FREE DVD

WAYLAND & MIR
THE FUTURE OF LINUX GRAPHICS

NOW FEATURING LINUXVOICE

LINUX PRO MAGAZINE

APRIL 2017

WAYLAND & MIR
The future of Linux graphics

Web Traffic Tools
Optimize your search rank with these analytics alternatives

digiKam 5
Manage your digital photo images

Qt Creator
Build your own mobile app

Wim Coekaerts
We Interview Microsoft's former open source VP

Bluetoothctl
Configuring Bluetooth in Linux

SSH Tricks
Secure login is only the beginning

LINUXVOICE

- Process Tracing
- Is the GPL Declining?
- Alternative Free OSs

WELCOME LINUXVOICE READERS! p65

FOSSPicks
- Calligra 3 Suite
- Systemd Genie

Tutorials
- Lock down your network with Tinc
- Work faster with a tiling menu

Issue 197
Apr 2017
US$ 15.99
CAN$ 17.99

APACHECON

16-18.05.17

MIAMI, FL.

APACHE:

BIG_DATA

16–18.05.17

MIAMI, FL.

ApacheCon *brings together Apache committers and the open source community to learn and collaborate on the technologies and projects driving the future of open source, web technologies and cloud computing.*

Apache: Big Data *gathers the developers, operators and users working in Big Data for education and collaboration across Apache big data related projects and technologies.*

One registration provides access to both events!

Linux Pro readers save 15% with code

LNM15

GO.LINUXFOUNDATION.ORG/APACHELNM

ApacheCon & Apache: Big Data 2017 are presented by THE LINUX FOUNDATION

BREAD AND RELEVANCE

Dear Reader,

PayPal cofounder and rocket maker Elon Musk keeps getting himself into the news. Sometimes he gets notice for his new products and projects, but he is also famous as a kind of self-appointed spokesman for the techno-future.

At a recent address in Dubai [1], Musk said "Over time, I think we will probably see a closer merger of biological intelligence and digital intelligence. What is needed, according to Musk, is a high-bandwidth interface with the brain that "helps achieve a symbiosis between human and machine intelligence and maybe solves the control problem and the usefulness problem."

To stay useful and relevant, you'll need to interface with a computer; otherwise, you won't be able to get a job, because all the robots and computers will do everything better than you can.

There was a time not so long ago when many people (not just science fiction writers, but many economists) believed the technological advances of the future would benefit everyone. The machines would do the work, and the humans would have their needs provided for, with more time for the things they cared about. The eminent economist John Maynard Keynes predicted that his grandchildren would only work a 15 hour work week [2]. However, the details of our economy have, instead, concentrated the wealth generated by our new technologies into the hands of a small number of people at the top of the pyramid, and the rest of us work just as much as we always did. This proposed merging of humans with computers appears to be a strategy for how we can continue to tread water and stay afloat in our economy as we start to lose our jobs to technology.

Musk does not give an example of how this symbiosis would work – at least in the public descriptions of his address. The article at the CNBC site says his concept "… would see a new layer of the brain able to access information quickly and tap in to artificial intelligence."

Notice the verb "tap in," which gives the warm impression that the human is in control and is sampling the best of what the computer offers in a dignified and proactive way. A more likely scenario is that the computer would provide the intelligence and the human would provide the body, or the human would serve as a specialized "interface" for dealing with other humans, smiling and looking polite to customers and serving as the computer's "front end." Neither of these options seems particularly appealing to me.

It seems odd that Musk and others like him would conclude we need to change our human biology rather than simply saying no to technology or changing the precepts of our economy. The underlying assumption is, instead of all humans benefiting from the labor of the robots, the only people who benefit will be the people who own the robots, and if you don't own a robot, you'd better darn well become a robot, or a half-robot at least, or you'll run the risk of starvation.

This sounds rather bleak, but to be fair, I don't think Musk really thinks about it that way. The way this future worship works is you just close your eyes, tune in to it, and start imagining the "possibilities." The implication is that you are being creative and brave for imagining remarkable things, but it is actually rather linear and predictable: First we connect the computers, then we connect the telephones, then we connect the home appliances, then we connect … THE PEOPLE!!!!!

I will credit Musk with raising awareness about the issue of economic displacement brought on by technology, but I do wonder if the solution might be to address the issue of unequal income distribution rather than going to the trouble to turn all the people into cyborgs.

I don't know if Elon Musk's weird vision of the future will come to pass. I kind of doubt it, but I could just as easily be wrong. One thing I do know is, if we ever reach this reality where every human has the equivalent of a souped-up USB port in the back of their skull so they can go into a Vulcan mind meld with the computers in their environment, they still won't exactly be able to declare victory on this relevance thing.

Joe

Joe Casad,
Editor in Chief

INFO

[1] "Elon Musk: Humans Must Merge with Machines or Become Irrelevant in AI Age": *http://www.cnbc.com/2017/02/13/elon-musk-humans-merge-machines-cyborg-artificial-intelligence-robots.html*

[2] "Economic Possibilities for Our Grandchildren, by John Maynard Keynes: *http://www.econ.yale.edu/smith/econ116a/keynes1.pdf*

LINUX MAGAZINE

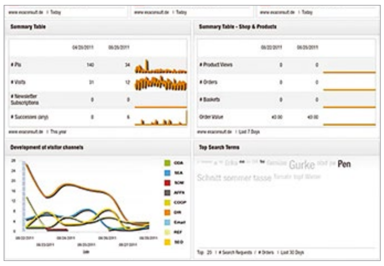
WHAT'S INSIDE

This month we show you why the X11 system that is the basis for Linux graphics might not be around much longer – and we explore a pair of alternatives that might replace it: Wayland and Mir.

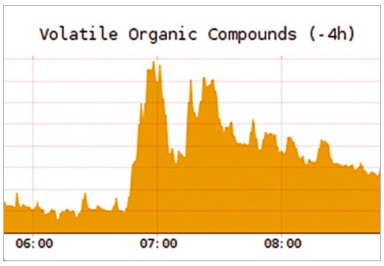
Other highlights:

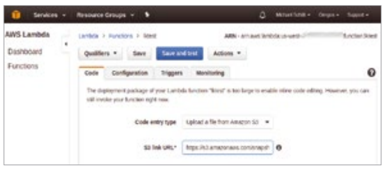
- **Web Traffic Tools** – Use these free apps to optimize your search rank – without the privacy complications of working with Google (page 34).
- **SSH Tricks** – SSH is much more than a secure Telnet alternative (page 44).

Also, lots more at Linux Voice (page 65), including a look at process tracing, a study of some alternative free operating systems, and a tutorial on tiling window managers.

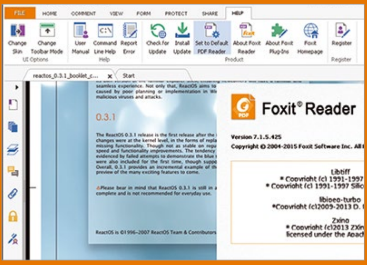


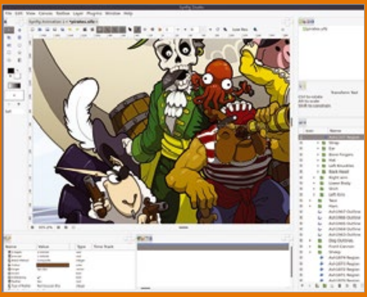
IN-DEPTH





LINUXVOICE





On the DVD

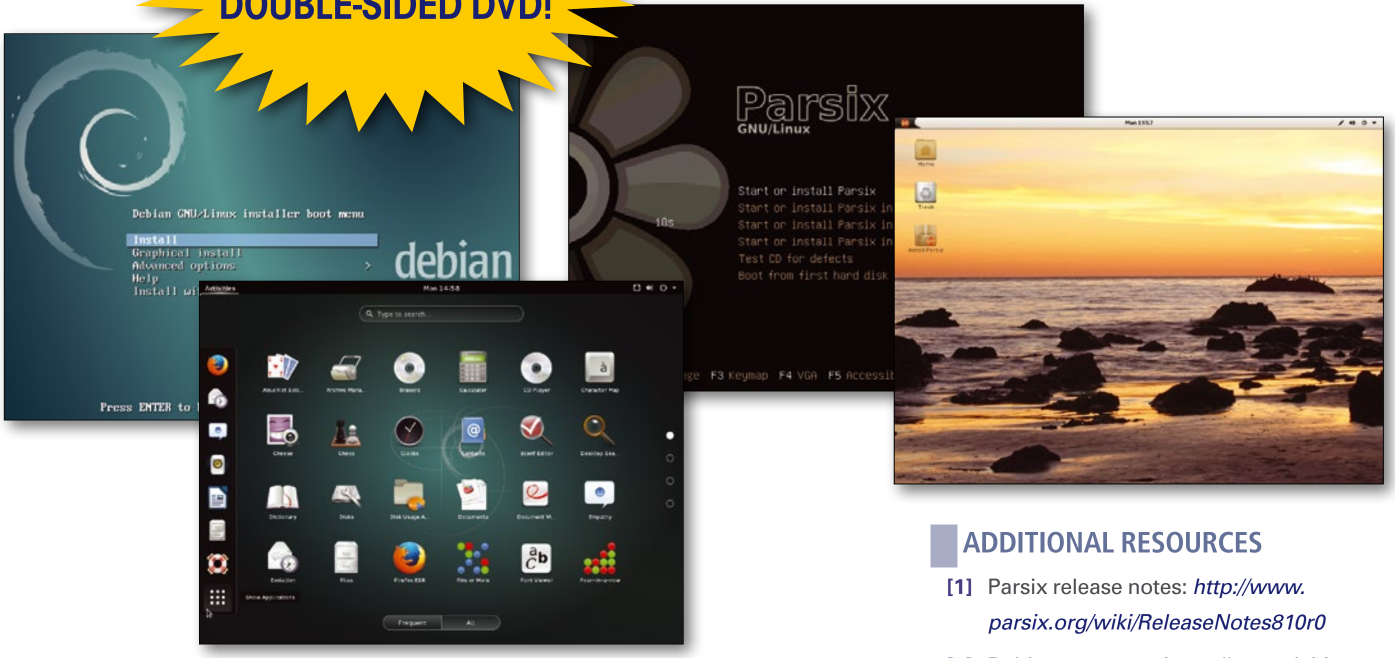
Parsix GNU/Linux (64-bit, Live)

Parsix is a "ready to use and easy to install" desktop Linux based on Debian's stable branch. Parsix GNU/Linux 8.10, code-named Erik, ships with a Gnome 3.20 desktop environment and Linux kernel 4.4.16 with TuxOnIce, the BFS process scheduler, and other patches. This version of Parsix syncs with Debian Jessie repositories. The 64-bit edition supports UEFI boot and installation. See the release notes [1] if you are an Nvidia or ATI/AMD user. Other features include:

- Xserver-X.Org 1.16.4
- Firefox 47.0.1
- Chromium Browser 51.0
- LibreOffice 4.3.3
- Glibc 2.19
- Gimp 2.8.14

Debian 8.7 Jessie (32-bit, Install)

The famous Debian all-free Linux is known for its stability and its vast software repositories. Debian 8.7 on this DVD is mainly a security update to fix problems to the stable release, along with "a few adjustments for serious problems" [2]. On installation, you have the choice of Gnome 3.14, KDE 4.11, Xfce 4.10, and LXDE desktop environments.


Debian GNU/Linux installer boot menu
Install
Graphical install
Advanced options
Help
Install wi
Press ENTER to
debian
Parsix GNU/Linux
Start or install Parsix
Start or install Parsix in
Start or install Parsix in
Start or install Parsix in
Test CD for defects
Boot from first hard disk
F3 Keymap F4 VGA F5 Accessib

ADDITIONAL RESOURCES

[1] Parsix release notes: *http://www.parsix.org/wiki/ReleaseNotes810r0*

[2] Debian 8.7 notes: *https://www.debian.org/News/2017/20170114*

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

NEW CloudNX

next generation Cloud Servers

UK data centres, UK support

Flexible

Pay as you use pricing

Enterprise features

- Load balancing
- VPN
- External firewall

Set up in seconds

- Easy to use control panel

Cloud Servers from:

£15.00

per month ex VAT charged at 20%

Call **0808 1686 777**

or visit **fasthosts.co.uk/cloud-servers**

SERVERS · WEB HOSTING · DOMAIN NAMES · EXCHANGE EMAIL

fasthosts

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

openSUSE Site Hacked

The news.opensuse.org site was hacked this week (*https://news.opensuse.org/*). Attackers defaced the site and posted the Kurdish flag and a message. The site was isolated from the rest of the openSUSE infrastructure, so critical services like the build, test, and download systems were untouched.

"Our offered downloads remain safe and consistent, and there was no breach of any openSUSE contributor data," openSUSE chairman, Richard Brown told us.

The hacked site runs WordPress, and it appears that the CMS software was not updated, allowing the attackers to exploit a known vulnerability.

The news.opensuse.org site is not managed by the SUSE or openSUSE IT teams but is, instead, administered by a team from SUSE's parent company Micro Focus.

LibreOffice Goes Online with 5.3 Release

The Document Foundation announced the release of LibreOffice version 5.3. The latest version is available for Mac OS, Windows, and Linux. This release has many new features, including an experimental ribbon-like interface reminiscent of Microsoft Word.

One of the biggest highlights of the new release is the source code for LibreOffice Online. Users can now install LibreOffice Online on their servers and use file sync and storage services like Nextcloud to create an experience similar to Google Docs and Office 365.

Although the source code is available for download, The Document Foundation has no plans to offer LibreOffice Online as a service. Italo Vignoli, a cofounder of The Document Foundation, told us that the foundation doesn't have the resources to build a Google-like infrastructure to offer such as service. LibreOffice Online is intended for ISPs and private cloud vendors. It also allows organizations and governments to build their own online document services.

The source code for LibreOffice Online is available immediately.

Microsoft Brings Clear Linux OS to Azure

Clear Linux OS is not just another desktop Linux operating system, it's an operating system by Intel designed for the cloud to compete with the likes of Container Linux. Now Microsoft is offering support for the operating system in its Azure Cloud 9 (*https://azure.microsoft.com/en-us/blog/announcing-the-availability-of-clear-linux-os-in-azure-marketplace/*).

Jose Miguel Parrella, Product Manager, Open Source, at Microsoft wrote in a blog post, "Microsoft Azure is the first public cloud provider to offer Clear Linux, and we're really excited about what it means for Linux users in the cloud and the community at large."

According to Parrella, Microsoft is offering a bare-bones virtual machine that can be used by customers to build out a system with bundles of their choice. It offers a container image that includes the popular Docker container run time and a sample solution image for developing machine learning applications preloaded with popular open source tools.

Parrella highlighted the performance of Clear Linux OS and said, "In addition to the performance features of Clear Linux, we believe that DevOps teams will benefit from the stateless capabilities of Clear Linux in Azure. By separating the system defaults and distribution best practices from the user configuration, Clear Linux simplifies maintenance and deployment, which becomes very important as infrastructure scales."

The move is not surprising because one out of three virtual machines in Azure run Linux, and Microsoft wants to ensure that no matter which distribution Azure customers run, it's fully supported on its cloud.

Hackers Infected DC Police Cameras Before Trump's Inauguration

A few days before the inauguration of Donald Trump, hackers infected 70% of storage devices that record police surveillance footage in Washington, DC.

The Washington Post reported, "City officials said ransomware left police cameras unable to record between Jan. 12 and Jan. 15. The cyberattack affected 123 of 187 network video recorders in a closed-circuit TV system for public spaces across the city, the officials said late Friday."

However, city officials didn't pay a ransom. They took the devices offline, formatted the software, and restarted the systems on site.

The initial discovery of the infection was made on January 12, 2017, when the police found a couple of sites not functioning properly. They informed the Office of the Chief Technology Officer (OCTO), which found two different kinds of ransomware on four recording devices. That's when they started investigating other sites and discovered more sites with ransomware.

The good news is that each site had four cameras connected to each recording unit; however, the recording unit was isolated from the department's environment, so the infection was contained to the recording devices.

Although it's comforting that they found the ransomware, it's unnerving that it was discovered because of malfunctioning devices and not because a system is in place that can detect malicious activities and mitigate them.

Let's hope after this incident and going forward, the OCTO will take steps to secure these devices.

Google Open Sources Chrome for iOS

Google has released the Chrome browser for iOS's source code. Google is usually very good at keeping its "open" projects open source, so why the delay in open-sourcing Chrome for iOS?

Copyright © Google/The Chromium Projects

Chrome comes with many binary blobs, so it is itself a proprietary technology. Chrome's source code (minus binary blobs) is released under the Chromium project upstream of Chrome.

However, Google was not able to keep the source of Chrome for iOS in the Chromium project because of the additional complexity of the way Apple's iOS works.

Rohit Rao, Upstream Angler at Google explained: "Due to constraints of the iOS platform, all browsers must be built on top of the WebKit rendering engine. For Chromium, this

MORE ONLINE

Linux Magazine
www.linux-magazine.com

Off the Beat • Bruce Byfield

KDE Comes to a Commercial Laptop
The newly announced KDE Slimbook is not free hardware. All the same, it is an important step in bringing Linux and related technologies to a commercial audience. Recently, I talked with Aleix Pol, the Vice President of KDE e.V, the governing body for the popular desktop environment, about how the project came about.

Adding a Bluetooth Speaker to Linux
These days, support for various technologies under Linux can often be taken for granted. An exception is Bluetooth, whose configuration remains arcane, as I found when trying to add a Bluetooth speaker to my workstation. My efforts were dogged by outdated information, and sometimes lack of information, but the audio improvement added by a high-end speaker made the effort more than worth my time.

Leaders, Rock Stars and Ninjas
In recent weeks, I find myself thinking about Ian Murdock, the founder of Debian and my former employee. Ian, you may remember, died a year ago after being beaten by the police. At the time, I described him as demonstrating "the modesty of a man who has nothing to prove." It's a description that applies to most of the best-known developers in free software.

ADMIN HPC
http://hpc.admin-magazine.com/

Modern Fortran – Part 3 • Jeff Layton
Fortran development is still progressing, with a new version scheduled to release in 2018. In this article, we look at Fortran 2008 and the upcoming Fortran 2015.

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

Network Backups with Amanda
Tim Schürmann
The free Amanda backup utility dates back to the days of tape drives, but it is still a powerful tool for centralized backup across the network.

Safeguard and Scale Containers
Thomas Fricke
Security, deployment, and updates for thousands of nodes prove challenging in practice, but with CoreOS and Kubernetes, you can orchestrate container-based web applications in large landscapes.

means supporting both WebKit as well as Blink, Chrome's rendering engine for other platforms. That created some extra complexities, which we wanted to avoid placing in the Chromium code base."

The source code is available on GitHub, and anyone can compile the iOS version of Chromium, just as they can for other versions of Chromium.

Raspberry Pi Compute Module 3 Released

The Raspberry Pi foundation has launched two versions of the Raspberry Pi Compute Module 3 (CM3).

The standard version is based on Raspberry Pi 3 hardware. It runs on a CM2837 processor (up to 1.2GHz) and comes with 1GB of RAM, plus 4GB of on-board eMMC flash storage.

A Lite version of CM3 (CM3L) comes with the same processor and the same amount of RAM, but instead of on-board storage, it comes with an SD card interface to the Module pins so a user can wire this up to an eMMC or SD card of their choice.

The foundation has also launched an updated version of the Compute Module IO Board V3 (CMIO3), which offers an easy-to-use "base" into which you can plug your compute modules, program them using the HDMI and USB connectors, and then use them in various applications.

The Compute Module has already gone beyond the DIY and maker communities and has made its foray into the commercial space. NEC is using the compute module in their commercial displays for corporate customers.

CM3 and CM3L are priced at $30 and $25, respectively. The Foundation has also dropped the price of the original Compute Module to $25. One of the core goals of the foundation is longer shelf lives of their devices. As a result, CM3 is kind of backward compatible with the CM1 design, which means you can simply remove the older CM and replace it with CM3. The only caveat is that CM3 can run a bit hotter than the previous board, and it's 1mm taller, so you do need a heatsink and take into consideration the 1mm of extra space.

D-Wave Announces 2,000-Qubit System

Canadian-based D-Wave Systems, Inc. has announced its new D-Wave 2000Q. D-Wave is the first commercial vendor to market quantum computing systems. The D-Wave 2000Q is a major upgrade from previous models, doubling the number of qubits from 1,000 to 2,000. According to the company, "With 2,000 qubits and new control features, the new system can solve larger problems than was previously possible, with faster performance, providing a big step toward production applications in optimization, cybersecurity, machine learning, and sampling."

D-Wave also announced their first customer for the 2000Q system, cybersecurity vendor Temporal Defense Systems (TDS). The list price for the 2000Q is $15 million.

D-Wave's systems have aroused controversy in the past. The details of implementing quantum computing in the real world are so arcane that you almost have to be a computer scientist and a quantum physicist to understanding how the system works. Some experts have questioned whether D-Wave systems are actually acting as quantum computers or whether the quantum speedup is significant enough to provide a viable alternative to conventional techniques. Others have expressed support for D-Wave systems and have presented evidence that the systems do indeed operate through quantum entanglement.

Zack's Kernel News

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

By Zack Brown

Increasing Maximum Address Space in x86-64

Liang Li pointed out that, currently, the x86-64 architecture only supports 46-bit physical memory addresses, which limits all x86-64 systems to a maximum of 64 TiB. But he said that some hardware vendors, notably Intel, were going to start building support for 52-bit addresses into their hardware. Liang said that Linux's Extended Page Table (EPT) code only supported four-level page tables, which would could go as far as 48-bit physical addresses, but no further. To reach 52-bit, he said, it would be necessary to extend EPT support to five levels.

This is the same story told in 2004 (see *lwn.net/Articles/106177*), except instead of migrating to four levels, Liang now wants to migrate to five levels. Liang posted a patch to do this.

Valdis Kletnieks had no major objection, but noticed that Liang's patch mentioned support for 52 bits in some places and 56 bits in others. He asked if Liang was trying to get all the way to 56 bits in an effort to "future-proof" the patch. Liang replied that 56 bits were "the virtual address width which will be supported in the future CPU with 52-bits physical address width. 5 level EPT can support a maximum 57-bit physical address width, as long as the future CPU use[s] no more than 57-bits physical address width; no more work is needed."

Kirill A. Shutemov liked Liang's patch and has already incorporated it into his own code tree.

Paolo Bonzini felt that Liang's patch seemed straightforward, though he wanted to investigate more. But he was slightly concerned that the way Liang had designed the patch might make it difficult to migrate virtual machines from one piece of hardware to another. Specifically, Paolo said, the LA57 mode in modern Linux would allow writing to high addresses, while older systems without LA57 mode couldn't. This meant that, as Liang's code stood, if a virtual system did a single write using LA57 mode, it would permanently block that system from migrating to an older system that didn't support LA57.

Ordinarily, Paolo said, the hypervisor would trap any unavailable features, but in this case, it wasn't able to trap LA57 – at least the code to do it hadn't been written yet, partly because the required change might slow the system down.

Essentially, Paolo said this wasn't a problem with Liang's code at all, but was a hardware issue. He said, "I am seriously thinking of blacklisting FSGSBASE completely on LA57 machines until the above is fixed in hardware."

Liang replied that he'd already forwarded that particular issue to the hardware people. But to some extent, that's that. If there's a hardware glitch with no viable workaround, it could simply delay adoption of this particular patch indefinitely, barring some new design idea.

Securing ext4

Yi Zhang described an exploit that a hostile user could use to cause memory corruption on systems running ext4. He posted a patch to prevent that sequence of actions from causing the corruption.

Andreas Dilger had minor quibbles but generally liked the patch. Valdis Kletnieks also liked Yi's work but wanted to find a technique to identify which filesystem on a multi-filesystem setup had been attacked. So Yi posted a modification of his patch, to identify the location of the problem as well as fix it. But Darrick J. Wong came up with an alternative solution, which Andreas liked better. However, Ted Ts'o pointed out that Darrick's improvement made some calls that wouldn't be possible from that particular context; so Yi's code stood as it was.

Yi also claimed that similar exploits existed on other filesystems. He asked, "do you think we should put these detections on the VFS layer? Thus other filesystems would not need to do the same things, but the disadvantage is that we cannot call `ext4_error` to report ext4 inconsistency."

ZACK BROWN

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**.

Ted replied that some filesystems didn't use inodes, and therefore wouldn't be susceptible to this kind of attack. But he added, "We'll have to see if Al and other filesystem developers are agreeable, but one thing that we could do is to do the detection in the VFS layer (which it is actually easier to do), and if they find an issue, they can just pass a report via a callback function found in the `struct_operations` structure. If there isn't such a function defined, or the function returns `0`, the VFS could just do nothing; if it returns an error code, then that would get reflected back up to userspace, plus whatever other action the filesystem sees fit to do."

Al Viro replied with some technical questions that seemed to indicate he was open to the idea, but at this point, the discussion petered out or went offline.

But clearly, everyone would be happy to plug a memory corruption hole, so we can expect the fix to go in very quickly, in whatever form is most pleasing.

Stabilizing RCU

Paul McKenney pointed out that the read/copy/update (RCU) implementation in Linux had some problems. RCU is a technique for making sure that certain data structures are available to all CPUs during bootup. Once the system is up and running, data structures can be accessed in RAM using normal means. But before all the resources of the system have been fully initialized, the CPUs all still need access to certain data. RCU is a way of copying the data between all CPUs in a way that's useful and won't cause race conditions or other problems.

Paul explained that RCU was initialized during boot-up in three distinct phases. During the first phase, you've got a single CPU running, with preemption disabled, so nothing else can interrupt it. At that point, any no-op is a Synchronous Grace Period, which means it's the moment that's OK for copying data.

In the second phase, preemption is enabled and the scheduler is running, but RCU has not yet got all its kthreads and workqueues up. During the second phase, if a Synchronous Grace Period comes along, the RCU isn't ready for data copying, and the system may crash.

In the third phase, the CPUs are running, the RCU is running, and everything's fine.

As Paul explained, that second phase has begun to be a problem, as Synchronous Grace Periods have begun to creep in and mess things up. He posted some patches to rework the second phase, to properly handle Synchronous Grace Periods during the second phase of RCU deployment.

Borislav Petkov liked the patch, though he had a few wording suggestions for the commit message and a couple of coding suggestions. He also tested the patch on a bunch of systems and confirmed that it seemed to work OK.

Paul posted a new patch, though regarding Borislav's coding suggestions, he said, "note that the code was buggy even before this commit, as it was subject to failure on real-time systems that forced all expedited grace periods to run as normal grace periods."

Rafael J. Wysocki wondered aloud if ACPI systems would need a separate fix for this whole issue, or if Paul's patch covered them as well. Borislav felt there was no need for a targeted ACPI patch, though he did say, "If you still think it would be better to have it regardless, you could pick it up (i.e., making ACPI more robust, yadda yadda). I dunno, though, perhaps it is only complicating the code unnecessarily and then can be safely ignored with a mental note for future freezes."

Lv Zheng also remarked, "ACPI fix is unnecessary as ACPI is just a user of the RCU APIs. And it's pointless to add special checks in the user side in order to use one of them."

Paul said that, if his RCU patch was delayed for any reason, the ACPI patch might be needed as a stopgap.

Boot time is one of those dark regions of the kernel, where few dare go and fewer still come out alive. If you ever want to feel fear, don't bother skydiving or bungee jumping. Just research what computers have to do in order to boot up successfully. Start with earlier history and work your way forward. The people who code in this area have the scars to go with every story. Paul's RCU patch was mercifully bloodless.

Supporting Larger Address Space, Pt 2

Previously, I described Liang Li's effort to extend EPT support to five levels in order to accommodate 52 to 56 bits of hardware address space. Coming at this issue from a different angle, Kirill A. Shutemov posted patches to allow userspace to successfully access 56-bit memory addresses. He said, "On x86, five-level paging enables 56-bit userspace virtual address space. Not all user space is ready to handle wide addresses. It's known that at least some JIT compilers use high bit in pointers to encode their information. It collides with valid pointers with five-level paging and leads to crashes."

Kirill wanted to accomplish this by setting an `rlimit`, which places limits on resource usage. But Andy Lutomirski felt that it would be more appropriate to be an ELF flag (aka note), which would allow the `setuid` bit to behave better, or a personality, which would allow flags to control the way in which a particular program could run. Kirill replied that he'd intended to implement ELF flags on top of the `rlimit` implementation, but admitted not fully grasping the `setuid` issue. Andy explained, "If a `setuid` program depends on the lower limit, then a malicious program shouldn't be able to cause it to run with the higher limit. The personality code should already get this case right because personalities are reset when `setuid` happens."

Meanwhile, Arnd Bergmann liked Kirill's patch, saying also that, "This seems to nicely address the same problem on arm64, which has run into the same issue due to the various page table formats that can currently be chosen at compile time."

Resolving these various issues led to a fairly technical consideration of the ins and outs of each implementation, and the various things that might go wrong with each, or the special powers that each might contain. For example, better/worse support for database users.

After quite a bit of discussion, Dave Hansen remarked to Kirill, "The farther we get into this, the more and more I think using an `rlimit` is a horrible idea. Its semantics aren't a great match, and you seem to be resistant to making *this* `rlimit` differ from the others when there's an entirely need to do so. We're already being bitten by 'legacy' `rlimit`. IOW, being consistent with *other* rlimit behavior buys us nothing, only complexity."

And elsewhere, Andy remarked at one point:

"Taking a step back, I think it would be fantastic if we could find a way to make this work without any inheritable settings at all. Perhaps we could have a per-mm value that is initialized to 2^47-1 on `execve()` and can be raised by ELF note or by `prctl()`? Getting it right for 32-bit would require a bit of thought. The ELF note would make a high stack possible and, without the ELF note, we'd get a low stack but high `mmap()`. Then the messy bits can be glibc's problem and a toolchain problem as it should be, given that the only reason we need a limit at all is because of messy userspace code.

Sure, the low stack prevents the *whole* address space from being used in one big block for databases, but 2^57 to 2^47 ought to be good enough.

I'm not 100% sure this is workable, but, if it is, it makes everyone's life easier. There's no need to muck around with `setarch(1)` or similar hacks."

Linus Torvalds stepped in at this point, saying definitely that "this is the right model. No inheritable settings, no suid issues, no worries. Make people who want the large address space (and there aren't going to be a lot of them) just mark their binaries at compile time."

But Kirill didn't want to give up on the inheritance idea just yet. He argued, "One thing that inheritance give[s] us is ability to change available address space from outside of binary. Both ELF note and `prctl()` don't really work here. Running a legacy binary with a full address space is a valuable option – as is limiting address space for a binary with ELF note or `prctl()` in case of breakage in a field. Sure, we can use `personality(2)` or invent [an]other interface for this. But to me, `rlimit` covers both normal and emergency use cases relatively well."

But Linus didn't agree. He felt that inheritance was "simply not valuable enough to worry about. Especially when there is a fairly trivial wrapper approach: Just make a full-address-space wrapper that acts as a binary loader (think 'specialized ld.so'). Sure, the wrapper may be fairly trivial but not necessarily pleasant: you have to parse ELF sections etc. and basically load the binary by hand. But there are libraries for that, and loading an ELF executable isn't rocket surgery; it's just possibly tedious."

Meanwhile Andi Kleen said he favored Kirill's inheritance-based approach. He felt Linus was minimizing the complexity of parsing ELF sections. Andi said, "Compile time is inconvenient if you want to test some existing random binary. I tried to write a tool that patched ELF notes into binaries some time ago for another project, but it ran into difficulties and didn't work everywhere. An inheritance scheme is much nicer for such use cases."

The discussion petered out at this point, though I'd expect Linus to have the last word eventually.

The difficulties of making a change of this kind – adjusting the way fundamental resources like RAM are handled by the system – is extreme. There are all sorts of caveats and corner cases, semantics that change in odd ways, speedups and slowdowns that occur in odd places, and also security concerns. It wouldn't surprise me to see this issue kicked around by the developers for months, before anything resembling an acceptable patch can emerge.

ARM SPE Support

Will Deacon said that the ARM 8.2 hardware "introduces the Statistical Profiling Extension (SPE). SPE provides a way to configure and collect profiling samples from the CPU in the form of a trace buffer, which can be mapped directly into userspace using the perf AUX buffer infrastructure."

He posted a patch to add a new perf driver to support ARM SPE. Peter Zijlstra asked for a high-level explanation of SPE, and Will replied:

"Sure, I can try, although there is no public documentation, yet so it's a bit fiddly.

SPE can be used to profile a population of operations in the CPU pipeline after instruction decode. These are either architected instructions (i.e., a dynamic instruction trace) or CPU-specific uops, and the choice is fixed statically in the hardware and advertised to userspace via `caps/`. Sampling is controlled using a sampling interval, similar to a regular PMU counter, but also with an optional random perturbation to avoid falling into patterns where you continuously profile the same instruction in a hot loop.

After each operation is decoded, the interval counter is decremented. When it hits zero, an operation is chosen for profiling and tracked within the pipeline until it retires. Along the way, information such as TLB lookups, cache misses, time spent to issue, etc. is captured in the form of a sample. The sample is then filtered according to certain criteria (e.g., load latency) that can be specified in the event config (described under `format/`), and, if the sample satisfies the filter, it is written out to memory as a record, otherwise it is discarded. Only one operation can be sampled at a time.

The in-memory buffer is linear and virtually addressed, raising an interrupt when it fills up. The PMU driver handles these interrupts to give the appearance of a ring buffer, as expected by the AUX code.

The in-memory trace-like format is self-describing (though not parsable in reverse) and written as a series of records, with each record corresponding to a sample and consisting of a sequence of packets. These packets are defined by the architecture, although some have CPU-specific fields for recording information specific to the microarchitecture.

As a simple example, a record generated for a branch instruction may consist of the packets shown in Table 1.

You can also toggle things like timestamp packets in each record.

Since SPE is an optional extension to the architecture, I'm sure there will be big.LITTLE systems where only one of the clusters has SPE support, so the driver is slightly complicated by handling that."

As you might imagine, a technical discussion ensued. With no public documentation, I would expect the kernel people to be reluctant to accept code into the actual kernel – though maybe they'd accept it in the staging branch for now. In either case, folks had questions about the implementation details as well. It's clear that this feature is in a very early stage of kernel support. But I wanted to include it here because of Will's very cool description of something that's barely hit the public eye. ■■■

TABLE 1: Branch Instruction Packets

0 (Address)	Virtual PC of the branch instruction
1 (Type)	Conditional direct branch
2 (Counter)	Number of cycles taken from Dispatch to Issue
3 (Address)	Virtual branch target + condition flags
4 (Counter)	Number of cycles taken from Dispatch to Complete
5 (Events)	Mispredicted as not-taken
6 (END)	End of record

Microsoft's former open source VP Wim Coekaerts

Equal Footing

By Swapnil Bhartiya

Microsoft has changed its ways since the old days, when the company gained a reputation for spreading what the FOSS community liked to call *FUD* (Fear, Uncertainty, and Doubt) about Linux. The new Microsoft has gradually released parts of the .NET platform to open source, and recently, CEO Satya Nadella stated that Microsoft "loves Linux." Some long-time FOSS watchers are skeptical, but the Linux Foundation recently showed their support by welcoming Microsoft as a Platinum-level member.

As part of that effort to upgrade their Linux position and improve their standing within the community, Microsoft hired Wim Coekaerts, who had helped jump-start the Linux effort at Oracle as the new Corporate Vice President of Enterprise Open Source. Coekaerts only stayed at Microsoft for eight months before he was lured back to Oracle, but according to Coekaerts, the decision to leave was strictly personal. He is quoted as saying "… from my short time [at Microsoft], I saw a company that really does love Linux and has a bright future in open source."

We caught up with Coekaerts at LinuxCon North America in Toronto (while he was still with Microsoft) and asked him about the company's changing attitude toward Linux.

Linux Magazine: Could you start by telling a little about yourself?

Wim Coekaerts: I joined Microsoft five months ago, and it's been a great five months so far. My role is Corporate Vice President for what we call Enterprise Open Source.

LM: What were you doing prior to joining Microsoft?

WC: Prior to joining Microsoft, I was at Oracle. I helped manage community contributions and worked to improve Oracle's Linux product. I'm doing similar things now at Microsoft: Making sure we work with the different open source projects in a natural way.

LM: Are you still contributing code to Linux?

WC: A little bit. I wrote a watchdog timer for a Sparc Linux several months ago, just because I felt like it, I guess. That went upstream to the main line. I like technology. My title at Microsoft is just a title; I'm a developer at heart.

LM: Being associated with Linux and open source for such a long time, you know it's more about culture than technology itself. When you came to Microsoft, what was the experience like, from a culture perspective, because at one point Microsoft was not as friendly towards Linux as it appears to be now.

WC: I think the reason I joined was because when I talked to Scott Guthrie and Mike Neil in January (we literally met at Starbucks just to chat), they made it very clear that Microsoft has evolved into a company with a very open culture. Different product groups can now send in pull requests to other groups. For example, groups working on SQL Server find something in Active Directory and send patches. They are trying to foster a sort of open development community within the company, and they've been working on that for a while.

That was one thing; the other was this big message around open source, around working with customers to make sure we do the right thing for customers and developers.

There have been enough open source projects at Microsoft that developers within the company can now see it as a career path. A lot of open source is happening within the company, and when something new happens people say "I should go and join that open source group."

When somebody is writing a product and they need a library, like some extension that makes it easier for them to do their job, it's so streamlined within the company that it has become natural. A lot of services, whether it's on [the] SDN side or on [the] Azure container side, are already running on Linux.

There are already internal Microsoft products and services being developed that use Linux platform. I think they've gone past a lot of those older stigmas that are no longer an issue.

At this point, open source is basically part of day-to-day life at Microsoft.

LM: Some people are skeptical of Microsoft, due to the history. But times have changed, and now Microsoft is building on top of Linux, so they can't really be hostile towards the technology that they are relying on.

WC: I think there are a number of things here. Azure is very important to the company. We need more Linux deployments (not just as Microsoft stack but in general) for Azure to grow. Customers who run various open source projects certainly run Linux. It's obviously in our best interest to make sure that customers know that we're not hostile towards Linux. In fact, Microsoft loves Linux. That's not just a tag line, because if we can't convince our customers that it's true, they won't come to us.

We recently hired Matthew Wilcox and Stephen Hemminger, who work on the Linux kernel side. We're going to expand the Linux work we do. We're contributing more code to the kernel that's beyond just enabling Linux on hypervisor or Linux in Azure.

We are doing upstream testing with Linux, and we'll do that for other open source projects as well. By doing those things, I believe we can change the opinion or the few skeptics that are still around.

LM: Beyond Hyper-V, what are the other areas that you are actively contributing to?

WC: Of course the Hyper-V stuff continues to evolve because there is a lot of work being done around containers. There is a lot of work done in the Docker space. Linux is a first-class citizen in Azure, so we will continue to work on technologies around it.

LM: Linux has had some security issues – not a lot, but some were serious. How can Microsoft help?

WC: Microsoft has a lot of development tools; there are a lot of static code analyzers, dynamic code analyzers, and many more tools like that. We're looking at how can we use those tools on open source code. We want to be able to run nightly static code analyzers on NTPD and OpenSSL, and when we find issues, we either report them or we try and fix them if we have the knowledge in house. But that's one area that we still have to investigate.

LM: The Linux Foundation is working on projects like the Core Infrastructure Initiative (CII) that aim at improving security. Microsoft is also part of that initiative, so what work are you doing there?

WC: There are two parts to this one. One of the reasons that the Core Infrastructure Initiative was created was that a lot of critical open source projects are done by one or two developers at home. They don't have the hardware and other resources to do the much-needed testing. As I mentioned earlier, for the work we want to do with developing the Linux kernel and testing upstream, CII is actually a good place to start.

The second part is that we will provide testing resources that should potentially help in finding bugs. I think that's a good way to contribute. As I've said for many years, a lot of open source enthusiasts (whether they do contribute code or not) see open source development as how many lines of code have you contributed. That's sort of the baseline for them. But QA is also an important part. People tend to forget that QA is actually very expensive. You need the infrastructure to run it. I think with Azure, we can contribute a lot of compute cycles to help run QA for these important infrastructure projects. It's important for security. Think about it: If the OpenSSL team runs tests on a small server, they might not find a large number of bugs because they don't have the same low latency ability they would have on a larger system. We can run these tests in a larger environment. I think that's a good contribution.

LM: Let's change gears and talk about Microsoft bringing PowerShell to Linux and Bash to Windows. How do you look at these two platforms?

WC: Well, I am not really an expert on the Windows side.

LM: Right, you are the Linux guy.

WC: I'm a Linux guy, sometimes Mac. But you know what? That's not an issue within the company. It's not a "you have to use Windows" company at all.

A lot of people used to install Cygwin on Windows to get a Bash front. But Cygwin is not really native; it's a separate compile set. By bringing Bash to Windows, we helped those developers that like the Windows desktop but want to use Bash. Consider this example. Someone uses Windows but then on occasion they have to write Python scripts or quickly test something that's on Linux. Of course, they can install a virtual machine, but it demands a lot more resources. So for these things, Bash on Windows is actually quite nice.

Then PowerShell does it the other way around; we have a lot of Windows system admins and DevOps folks who also have to do stuff on Linux, and this kind of makes it cross platform.

What I really like with PowerShell is that all these Linux distributions are slightly different. If you're an admin that has to deal with three or four Linux distributions or even different versions within the same one, then you have to customize your scripts. On Slack, you need to do it this way, whereas on Ubuntu you have to do it a different way. With PowerShell, commandlets make things easier in such cases; `useradd` is `useradd` no matter which distribution you're on. I think it's a management tool that can hide some of the complexities that come by using different Linux distributions. I'll make it a point to say, it's not a replacement for Bash. It's just a different solution.

LM: Earlier you mentioned a career path within Microsoft for developers who are interested in Linux and open source.

WC: Let's say there is a developer who has been doing Windows kernel development for 15 years. He loves Windows; he just feels that it's nice to do something else. He knows about the Linux work happening, and he would like to learn how to do Linux kernel development. Now he, and anyone else at Microsoft, can do that.

There are a lot of people that say the Linux kernel community specifically is sort of a static size team. It's very difficult to get into it because people are really worried that if they send a patch upstream, they might get yelled at, then back off and say "I'm never doing that again." Now what's interesting is that the core kernel developers just move around. They work at Red Hat, Microsoft, Intel, HP…all those companies. But they still do the same work, so the overall size of the core kernel teams remains the same.

To increase that size, to create more contributors, we are doing interesting things, and we are not the only ones – a lot of other companies are doing it too. As I said earlier, we have hired many core kernel developers, including Wilcox and Hemminger. We have been hiring or internally transferring people that are not Linux kernel developers but want to work on the kernel. They know operating systems very well, but they are new to Linux. We can create safe ways for them to learn to contribute. They can write patches and send them to Wilcox or others for review. They will get feedback, suggestions on the right way to do it. People have an opportunity to contribute, in a very safe environment.

LM: When it comes to using open source, what's Microsoft's approach towards upstream?

WC: Upstream first. Kernel development always goes upstream. It doesn't make sense to have private functionality. When it's upstream accepted, then it gets used. We work with our partners on it; we work with Red Hat, Canonical, SUSE, and others. A lot of container services are built on Ubuntu, so the way it works is, we send code upstream, and Ubuntu picks up the code. When we use Ubuntu, then we use the code that they ship. There are no forks of private branches that we know won't get accepted, because then you end up with an impossible situation.

LM: One last question: When you say that Microsoft loves Linux, what does it really mean?

WC: What it means is that customers love Linux and Azure is an important platform for customers.

LM: That's the corporate talk?

WC: No. When we say we love Linux, it means we have to make sure that it runs in the best way possible. Microsoft loves Linux means equal footing, first-class citizen status in Azure. When we release a new service, it goes out on Windows and Linux at the same time. When we provide new functionality in terms of SDKs or CLIs, it's available on both platforms. What we don't want to end up happening is that if I run Linux Azure, I can only do 50% of what Windows can do. We are not holding anything back from Linux. It's 100%. ■■■

DC { DevConf Panamá 2017 }

Connect and learn with software development people.

Some of the technologies we will be speaking about

Symfony Drupal™ 8 backdrop ANGULARJS node JS

An event to share, learn, meet and network with software developers and business leaders across different disciplines from Latin America & the US.

+300 attendees // +90 speakers // +80 Talks

Register Now at
devconfpanama.com
You can get tickets online now!

Radisson Decapolis Panama
Panama
May **25-26/2017**

marketing@devconfpanama.com

+507 209 9002

Organized by:
rootstack
Web Solutions | Mobile Applications

The new display servers

Changing of the Guard

The X Window System, which dates to the 1980s, still forms the basis of Linux's graphical user interface, but it does cause some despair among developers with its legacy ballast and outmoded technology. Wayland and Mir are two promising candidates for the changing of the guard. *By Tim Schürmann*

The good old X Window System consists of an X server that receives events from input devices and draws the graphics on the screen. On top of this system is a compositor, which records the positions of Windows and adds novel graphical effects as needed (Figure 1). Desktop environments add a window manager that decorates every window with a frame.

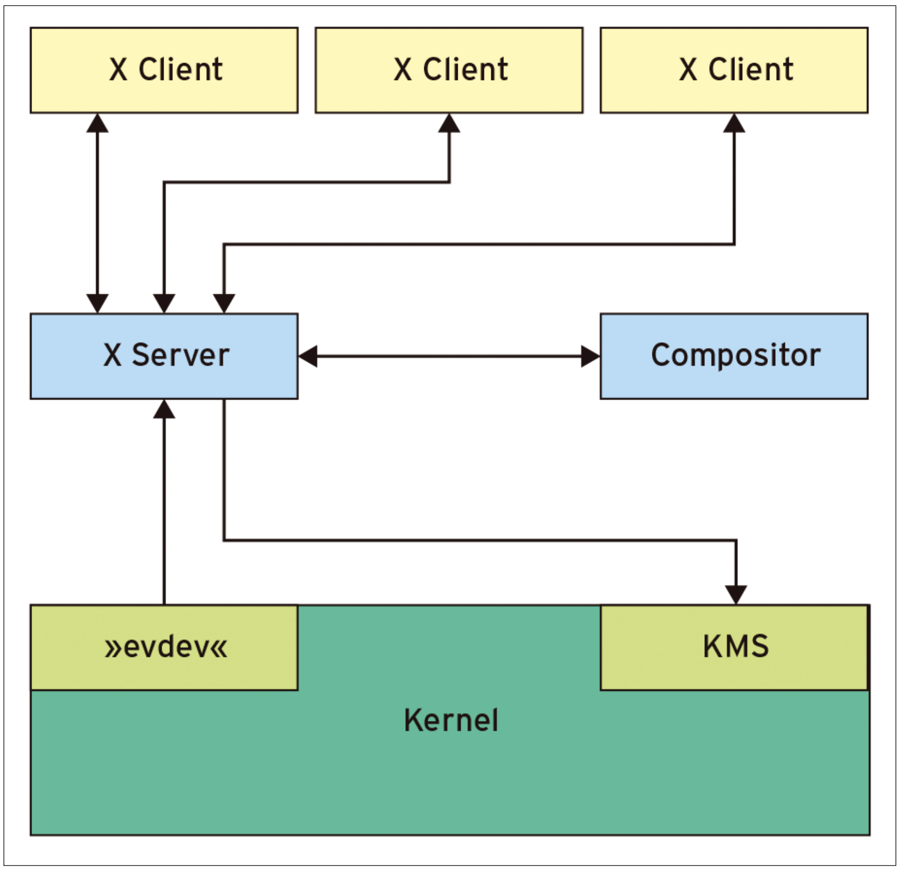


Figure 1: The X Window System: The kernel reports a mouse click to the server, which forwards the event to an X client. The client sends a render request to the X server, which also informs the compositor. The compositor recomposes the screen area and asks the X server to redraw it.

All of these components need to communicate through a time-consuming process. The cumbersome structure and outdated complexity of the X system bothered programmer Kristian Høgsberg, so he designed an alternative back in 2008, which he dubbed Wayland [1]. Høgsberg kept to the core functions that a display system currently needs to provide on Linux. He also set out to design Wayland so that it would avoid tearing, lag, flicker, and unnecessary redrawing of modified screen areas [2].

The result was a surprisingly sleek and efficient system. Media reports increasingly attracted the attention of supporters to Høgsberg's new display server architecture.

One organization that jumped on board with the Wayland development effort was Ubuntu parent company Canonical, but after a couple years in the trenches, Canonical announced that it would instead build its own graphical system called Mir, thus launching another alternative for an X11 replacement.

So far, most Linux distros think of Wayland and Mir as experimental technologies, and few systems actually use them as the default, but most observers believe that the new graphics systems will one day see wide adoption. We decided it was time for a closer look at what Wayland and Mir really are, how they work, and how they differ from the venerable X Window System.

One for All

On a system with Wayland, the Wayland compositor runs in the background. When a user clicks on a window, the compositor receives a matching kernel event. The compositor then determines which window the user clicked on. Wayland receives the necessary information faster than the X server. In contrast to its ancestors, the Wayland compositor always knows where the windows are located and what transformations it has applied to the windows. After finding the right window, the compositor sends the event to the associated application (Figure 2).

Programs that run on Wayland are known as Wayland clients. Clients need to draw their window content themselves – either manually, or using libraries like GTK+, or using hardware-accelerated interfaces such as OpenGL. After doing so, the client notifies the compositor, which puts together the contents of the screen and triggers the display.

In the old X Window System, a window manager draws borders and other decorations around the individual windows.

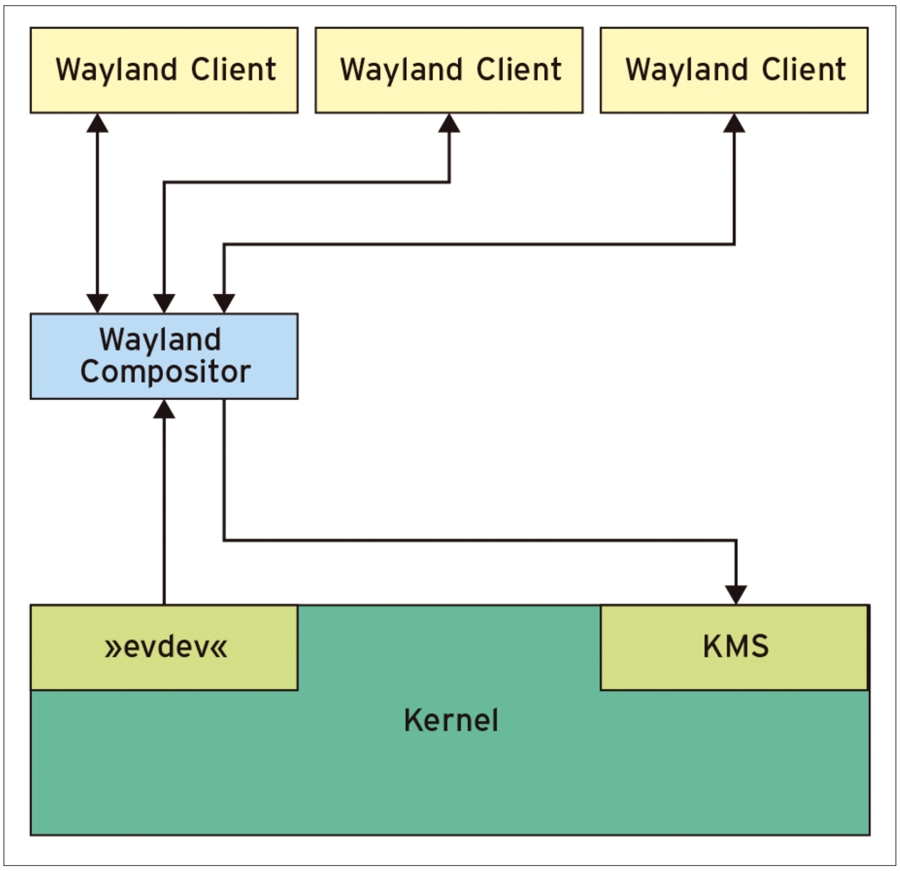


Figure 2: In contrast to the X Window System shown in Figure 1, Wayland merges the X server and compositor to form the Wayland compositor.

The Wayland specification does not clarify the responsibilities. Usually, the compositor decorates the windows with frames (server side decoration).

Name Confusion

The compositor and clients use a network protocol by the name of Wayland to talk to one another. In practice, developers also refer to the complete system of clients and the compositor as Wayland. The compositor and clients exchange their messages via Unix sockets. The name of the endpoint is found in the environment variable `WAYLAND_DISPLAY` and is usually `wayland 0`. Messages are always sent asynchronously, which translates to speed increases.

To make it easier for programmers to work on clients and the compositor, the Wayland developers provide the `libwayland-client` and `libwayland-server` C libraries (Figure 3). Confusingly, this reference implementation of the network protocol also goes by the name of Wayland. The source code of both libraries is available from the project page in the form of the `wayland` package [1].

The package also includes an XML file by the name of `wayland.xml`, which contains a formal definition of the permissible messages.

The Wayland scanner from the source package generates the two C libraries (semi-) automatically from the XML file. If you compile the libraries according to the guide, you remain blissfully ignorant of them.

Independent

Clients need to render their own window content. To do so, the client first requests a memory area from the compositor that both elements can access, which is known as a shared memory buffer (SMB). The client pushes the new window content into the SMB. It then tells the compositor that something has changed, which rectangular section is affected, and in which buffer the window content is located. The compositor then grabs the buffer and arranges the stored image material, along with the other windows, to create a work of art.

Although the client must request the buffer from the compositor, the client is responsible for managing the buffer. It can use a new buffer for each drawing operation or request multiple buffers in advance and then switch back and forth between them.

You can avoid the cumbersome task of working with buffers and the `libwayland-client` library if you develop your programs with SDL, Clutter, Cairo, GTK+ 3, or Qt 5. These popular graphics libraries or GUI frameworks now support Wayland out of the box.

In the case of GTK+ 3 and Qt 5, the users can decide later on whether to use X11 or Wayland for the display. GTK+ 3 offers the environment variable `GDK_BACKEND` [3] for this purpose; for QT 5, the environment variable goes by the name of `QT_QPA_PLATFORM`. In addition, Qt programs support the `platform` command-line parameter to enforce output via one of the two systems.

In addition to the listed options, clients can also hardware-accelerate the window output using OpenGL or OpenGL ES. The client first allocates a buffer directly on the video card using the EGL interface. Like OpenGL, EGL [4] comes from the Khronos group, and it mediates between the window system – in this case Wayland – and the OpenGL and OpenGL graphics libraries.

Once the client has drawn the contents of its window, it hands control over to the compositor. The compositor then composes the screen content with EGL, OpenGL, or the OpenGL successor Vulkan [5].

Freedom!

Wayland does not prescribe a rendering mechanism. EGL and OpenGL are optional, although the official documentation mentions them [6]. You can replace the EGL implementation at any time independently of Wayland. (The clients and the compositor must agree on an implementation.) The system can even connect more graphics stacks if the compositor offers the appropriate buffer. For instance, you could have a special buffer for stereo images, which are sent to VR glasses.

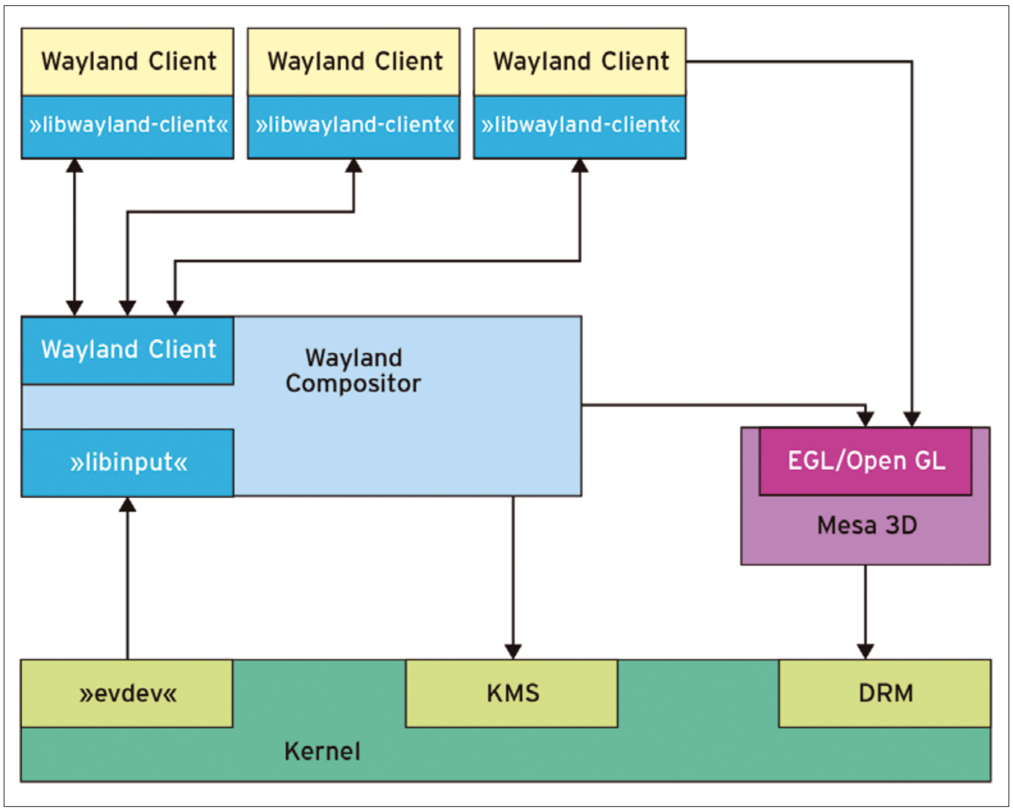


Figure 3: Under Wayland, the compositor and clients use established libraries and kernel interfaces. Confusing: The reference implementation of the network protocol is also named Wayland.

COMPOSITOR HELP

The compositor has to monitor and manage the input devices. To make the work easier for compositor manufacturers, the Weston developers have outsourced the code to a separate library. Libinput detects newly connected input devices, receives events sent by them, and passes the information on the compositor following preparation. Libinput communicates with udev and the evdev interface in the kernel (Figure 3). In addition to mice and keyboards, Libinput also supports touch devices.

If the compositor has the necessary support, the hardware can even directly adopt and display the buffer contents. (This feature is especially useful for games and videos in full-screen mode.)

The Wayland compositor does not need to run as a stand-alone display server but can itself be a Wayland or X11 client. In these cases, the Wayland compositor usually opens a window, in which the output from the Wayland clients appears.

A client could be a normal application, an X server, or another Wayland compositor. X-Wayland is an X server that runs as a client in Wayland. Thanks to X-Wayland, you can continue to use legacy X11 applications under Wayland. After launching, they connect to X-Wayland, which relies on Wayland to output the window content to the screen.

On to Weston

Weston is a reference implementation of the Wayland compositor by the Wayland developers. Like the client libraries, Weston is released under the MIT license. Weston also serves as a model for developers of desktop environments: To make desktop environments run under Wayland, the developers need to rework their window managers to complete Wayland compositors. (See the "Compositor Help" box.) These efforts are currently in progress at the KDE and Gnome projects. Gnome users can test the results obtained so far with the latest Fedora (Figure 4).

KDE users can test the state of development using the KDE Neon live system [7]. The following command

```
sudo apt-get install ⮧
plasma-workspace-wayland
```

first retrofits Wayland. In the login screen, the user then goes to

```
Desktop session:</C> Plasma (Wayland)
```

in the bottom left corner. Another option is RebeccaBlackOS, a live system that relies entirely on Wayland and Weston and comes with several compositor models you can try out (Figure 5, [8]).

On the Network?

In the X Window System, clients can communicate with the X server via a network if necessary. You can thus launch programs on a remote computer, and the windows will display on a local system. This process is transparent for the user and even works across different systems. This network transparency is missing in Wayland. Although there have been repeated

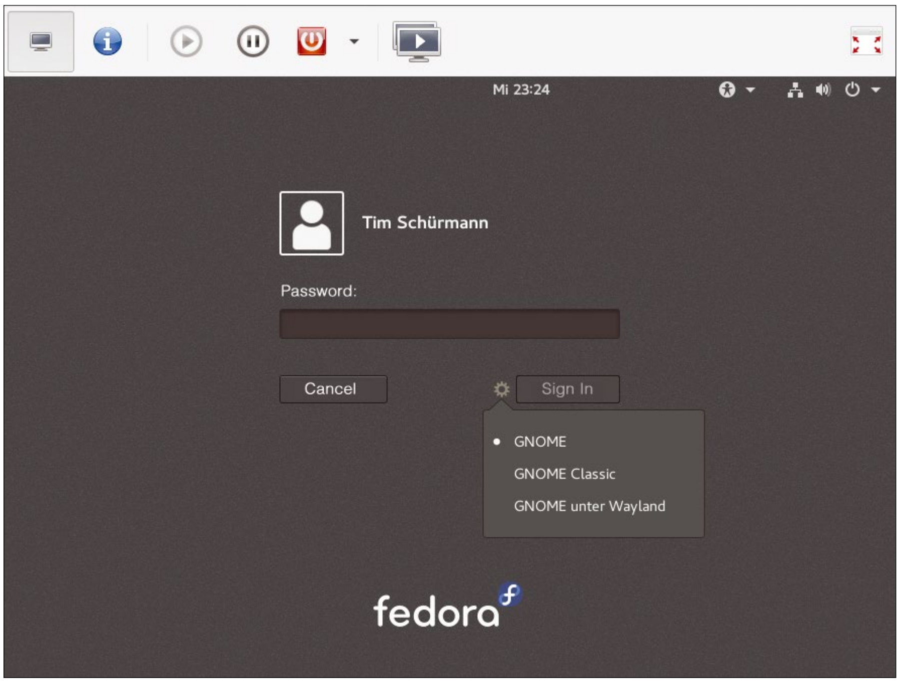
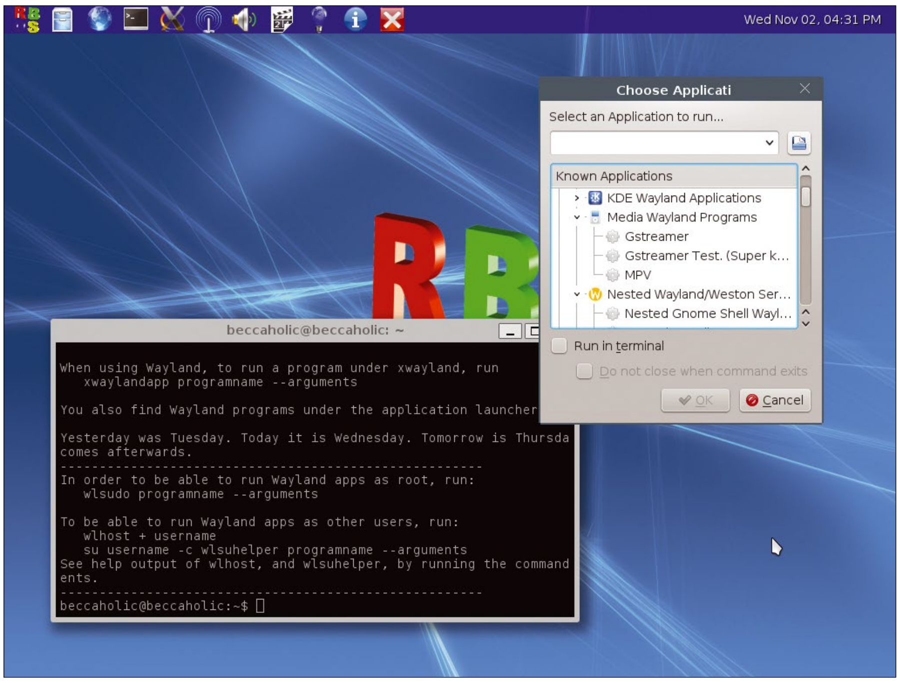


Figure 4: Most distributions with Wayland support let you change to a session with Wayland in the login screen. The figure shows this for Fedora 24.



```
When using Wayland, to run a program under xwayland, run
   xwaylandapp programname --arguments

You also find Wayland programs under the application launcher

Yesterday was Tuesday. Today it is Wednesday. Tomorrow is Thursda
comes afterwards.
-------------------------------------------------------
In order to be able to run Wayland apps as root, run:
   wlsudo programname --arguments

To be able to run Wayland apps as other users, run:
   wlhost + username
   su username -c wlsuhelper programname --arguments
See help output of wlhost, and wlsuhelper, by running the command
ents.
-------------------------------------------------------
beccaholic@beccaholic:~$ 
```

Figure 5: RebeccaBlackOS running Weston; the terminal is a native Wayland application.

attempts to extend Wayland, they have not made it into the official specification thus far. One of failed projects is Remote Wayland, which was launched in the scope of the Google Summer of Code 2011 [13]. The Gnome project has also conducted [14] its own experiments; remote control software such as VNC offers an alternative.

Pekka Paalanen explains why the introduction of network transparency is so complicated: "The Wayland protocol was written with shared memory in mind. The costs for the data transmission are effectively zero, thanks to shared main memory. Because of this optimization, the points where network transmissions can tap in are missing. You would need to evaluate many Wayland messages before you knew when and which parts of a buffer you need to transfer; therefore, simply forwarding individual messages does not work. When you start to build the necessary machinery that knows what and when to send, you soon realize that you've written a quarter of a compositor. [...] Also, you would need to evaluate quite a large number of Wayland messages to send content over a network.

Instead of artificially establishing network transparency and creating unnecessary overhead and complications that a local environment does not need, Wayland has limited its field of activity to local communication and is optimized for local activity. If you separate the network support and local communication, the translation layers between the two can be as complicated, smart, and pre-emptive as they want to be.

Last but not least, you can support an existing remote protocol, such as RDP, and thus avoid inventing a (bad?) new one. This design decision also completely evades the question as to whether you want to transmit a single window or the entire desktop. It allows for interesting applications, such as displaying the same window locally and on remote screens, and possibly even allowing people from different countries to use it at the same time."

Mir is Near

In 2010, Ubuntu founder Mark Shuttleworth announced in a blog post that the distribution would be moving to Wayland [15] in the long term. However, in the Spring of 2013, Ubuntu parent company Canonical surprisingly announced that it was developing a separate display server called Mir. The company said the existing solutions simply did not meet their needs [16]. In particular, Canonical was looking for a display server that would run efficiently on cellphones and also work with proprietary Android drivers. After looking at the Wayland protocol, Canonical realized that the specification did not meet these requirements. The Ubuntu wiki also reports that Canonical had issues with Wayland's inflexible event handling.

The Mir documentation available on the Internet is extremely terse. It is limited to a handful of pages on the Ubuntu wiki [17], a Launchpad page with the source code [18], and a few blog posts by Mir developers. Despite the lack of documentation, all devices that run Ubuntu should support Mir, including desktop and mobile devices, embedded systems, and kiosk systems.

Many users report that Mir works efficiently, supports conservative use of CPU capacity and memory, and provides a stable API. The developers place great emphasis on security, as Canonical system architect Thomas Voß assured *Linux Magazine*: "Real and virtual input streams are designed to reach only the focused application. If the input stream needs to be filtered/processed by a further component, either the system or the user needs to trust this component."

The Mir architecture outlined in the Ubuntu wiki is similar to Wayland and uses some Wayland concepts. For example, Mir uses EGL and provides X-Mir, an X server to run legacy X11 programs. Conversely, Mir can launch under the X Window System. Mir originally managed the input devices via the Android input stack. Then Canonical employees modified it so that it would compile without the Android source code.

At the end of 2015, Mir v0.18 changed to the Libinput library, which, ironically, comes from the Wayland project [19].

Fresh Mussels

Mir also includes a compositor. The compositor composes a complete representation of all the windows and then draws the results on the screen. Mir refers to the individual windows as surfaces. The compositor also has a renderer that applies effects on the individual surfaces. The compositor is synchronized with the refresh rate of the screen (Vblank), so as to avoid tearing effects and unnecessary (render) cycles.

The Mir concept supports trusted sessions. Trusted components send queries to the user, and the queries are then rendered in the window (surface) of the application. If an application wants to access positioning or location data, the associated service asks for the user's permission. Mir then displays the prompt on the surface.

In Mir, an additional component known as the shell handles window management. The shell also provides several

PROPRIETARY DRIVERS

The existing Wayland compositor types use a kernel mode setting to define the resolution of the graphics card. OpenGL interfaces are provided by the Mesa 3D graphics library, which uses the kernel's Direct Rendering Manager (DRM) (Figure 3). Initially, Wayland only ran with free graphics cards drivers.

Support for the proprietary NVidia driver was introduced in version 364.12 of Wayland. The support required modifications to Weston, although NVidia at least provided the appropriate patches. The Wayland and Weston developers resisted the idea of modifying Weston to accommodate proprietary drivers. Also KDE developer Martin Gräßlin ruled out adapting KWin – at least until the patches had become part of Weston [9].

At the XDC 2016 conference, the NVidia developers asked for help to design a solution that would allow their drivers to work with Wayland, which, according to Pekka Paalanen, boosted their popularity at the conference. The result of this cooperation is a new API named the UNIX Device Memory Allocator, which is still in its infancy ([10], [11]).

Meanwhile, the Gnome developers have modified their compositor so that it supports the proprietary NVidia drivers [12]. Users can evaluate the results in Fedora 25 Workstation. But the developers want to revert the changes to the compositor in favor of the emerging API. Also the reference compositor Weston has now integrated selected NVidia patches.

convenience functions, such as an application launcher and an ALT-tab switcher. In Ubuntu, Unity 8 serves as the shell; other shells did not exist when this issue went to press.

The Mir developers currently provide two libraries on Launchpad: `libmirserver` and `libmirclient`. `libmirserver` encapsulates the Mir server components and thus supports the development of a compositor. Mir clients use the `libmirclient` library to communicate with the Mir server. Alternatively, the applications can use the SDL, GTK+ 3, and Qt 5 toolkits, which Mir supports natively. Qt and QML form the basis for the Ubuntu SDK, which lets developers write Ubuntu applications [20].

Test Drive

Mir currently only runs on smartphones with Ubuntu Touch, but in the long run, Mir will replace the X Window System in Ubuntu. Many official Ubuntu derivatives do not want to make the change: The Kubuntu and Ubuntu Gnome developers have already announced that they will detour to Wayland in the future [21].

Users of Ubuntu 16.10 can test Mir and Unity 8 for the first time. However, the Mir/Unity duo leaves a very unfinished impression. For example, the pre-installed applications adapted for Mir are restricted to a simple browser and a terminal. X11 applications failed to launch, at least on the test system in our editorial office. The plans envisage Ubuntu completely changing from X11 to Unity 8 and Mir in the spring of 2018. Smartphones running Ubuntu Touch are already smooth and stable. Mir is also available in the new Snap package format for Ubuntu Core as of version 16.04 [22].

Just like Wayland, Mir relies on Generic Buffer Management (GBM) from Mesa 3D and DRM and KMS in the Linux kernel. Mir is thus – at least at the moment – also dependent on the free graphic stack. According to system architect Thomas Voß, the Mir team is collaborating with NVidia on a new alpha version of the proprietary NVidia driver adapted for Mir (see the “Proprietary Drivers” box). Mir already runs with the Intel driver. In addition, the developers are preparing Mir for various back ends. In particular, Mir will use the OpenGL successor Vulkan. Developers are also thinking of IoT devices, where rendering will need to occur in the software.

Mir is released under the GPLv3, but if you want to submit improvements, you first need to sign the Canonical Contributor License Agreement (CLA). CoreOS developer Matthew Garrett has criticized the CLA, which, according to Garrett, lets Canonical commercially distribute the code written by volunteers to manufacturers of smartphones [23].

Summary

In the past, several projects have attempted to replace the aged X Window System, including Y and Fresco [24]. The chances for Wayland are good, however. The X11 successor has received broad support from industry leaders, such as Intel and Red Hat, as well as projects such as Gnome and KDE. Even NVidia has approached the Wayland developers about providing support. The previous level of development looks promising under Fedora, and KDE is already relatively usable with Wayland. However, the biggest disadvantage of Wayland is in the lack of network transparency.

Mir cannot offer network transparency either. Although it has demonstrated its everyday usability on smartphones, Mir still has not arrived on the desktop. The fact that Canonical is largely creating Mir single-handedly is probably one reason for the long development. Since even major Ubuntu derivatives have come out in favor of Wayland, Mir is likely to remain an isolated solution, at least for the time being. ■■■

INFO

[1] Wayland: *https://wayland.freedesktop.org*

[2] Phoronix, Michael Larabel, “Wayland: A New X Server For Linux”: *http://www.phoronix.com/scan.php?page=article&item=xorg_wayland&num=1*

[3] Wayland support in GTK: *https://wiki.gnome.org/Initiatives/Wayland/GTK%2B*

[4] EGL: *https://www.khronos.org/news/press/khronos-releases-egl-1.5-specification*

[5] Vulkan 1.0 spec with Wayland support: *https://www.collabora.com/news-and-blog/blog/2016/02/16/vulkan-1.0-specification-released-with-day-one-support-for-wayland/*

[6] Wayland documentation: *https://wayland.freedesktop.org/docs/html/*

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

[8] RebeccaBlackOS: *https://sourceforge.net/projects/rebeccablackos/*

[9] Martin Gräßlin, “To EGLStream or not”: *https://blog.martin-graesslin.com/blog/2016/09/to-eglstream-or-not/*

[10] Unix Device Memory Allocation: *https://lists.freedesktop.org/archives/dri-devel/2016-October/120067.html*

[11] Unix Device Memory Allocator: *https://github.com/cubanismo/allocator*

[12] Wayland support for proprietary NVidia drivers: *https://bugzilla.gnome.org/show_bug.cgi?id=773629*

[13] Remote Wayland: *https://github.com/kempj/remote-wayland*

[14] Wayland Remoting: *https://wiki.gnome.org/Initiatives/Wayland/Remoting*

[15] Mark Shuttleworth, “Unity on Wayland”: *http://www.markshuttleworth.com/archives/551*

[16] Oliver Ries, “Taking Unity to the next level”: *https://lists.ubuntu.com/archives/ubuntu-devel/2013-March/036776.html*

[17] Mir specification: *http://wiki.ubuntu.com/MirSpec*

[18] Mir on Launchpad: *https://launchpad.net/mir*

[19] Mir release 0.18: *http://kdubois.net/2015/12/22/new-mir-release-0-18/*

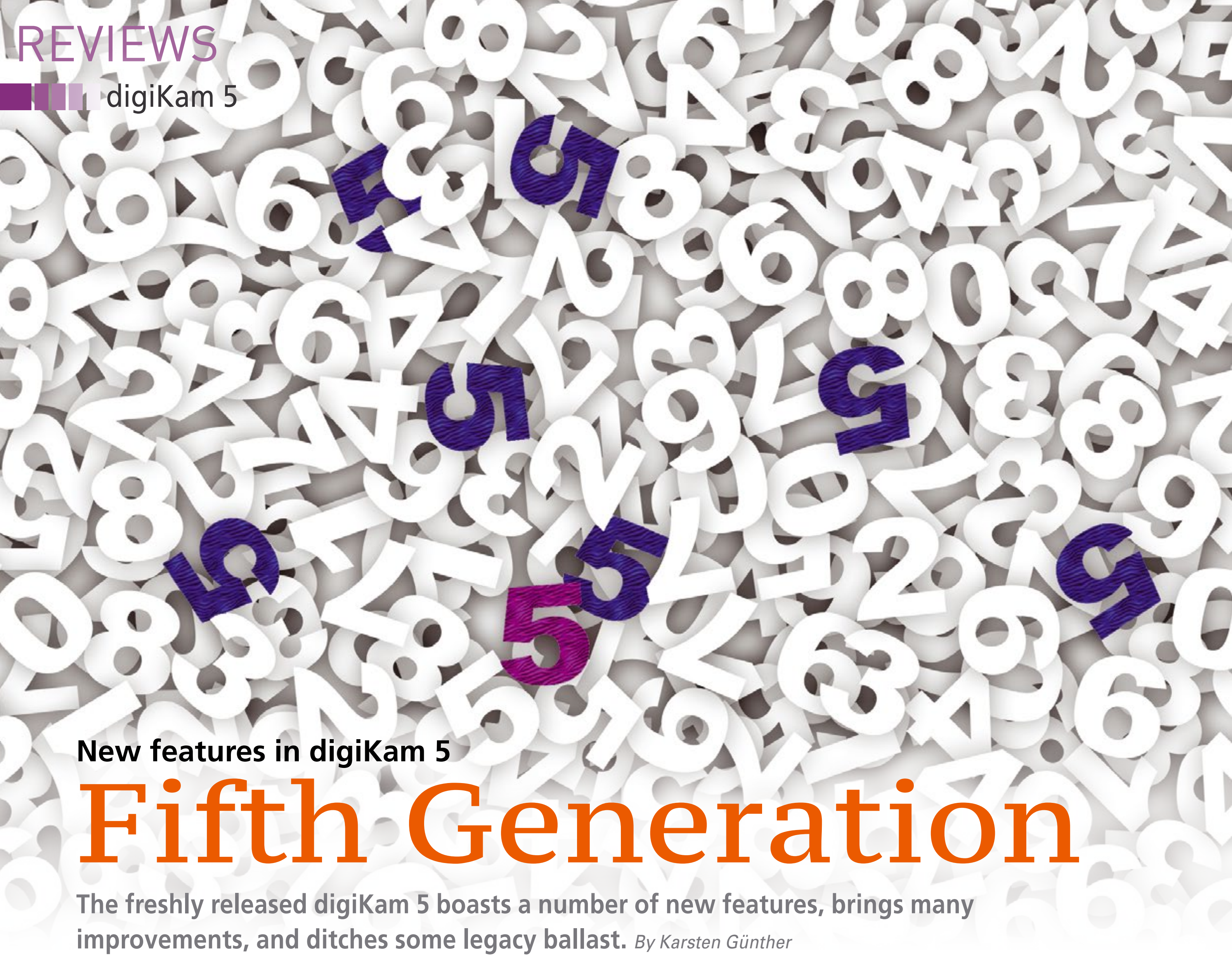
[20] Ubuntu SDK: *https://developer.ubuntu.com/en/phone/platform/sdk/*

[21] Ubuntu Gnome FAQ: *https://ubuntugnome.org/documentation/faq/*

[22] Mir Kiosk Snaps: *https://developer.ubuntu.com/en/snappy/guides/mir-snaps/*

[23] Matthew Garrett, “Mir, the Canonical CLA and skewing the playing field”: *http://mjg59.dreamwidth.org/25376.html*

[24] Fresco: *https://en.wikipedia.org/wiki/Fresco_(windowing_system)*



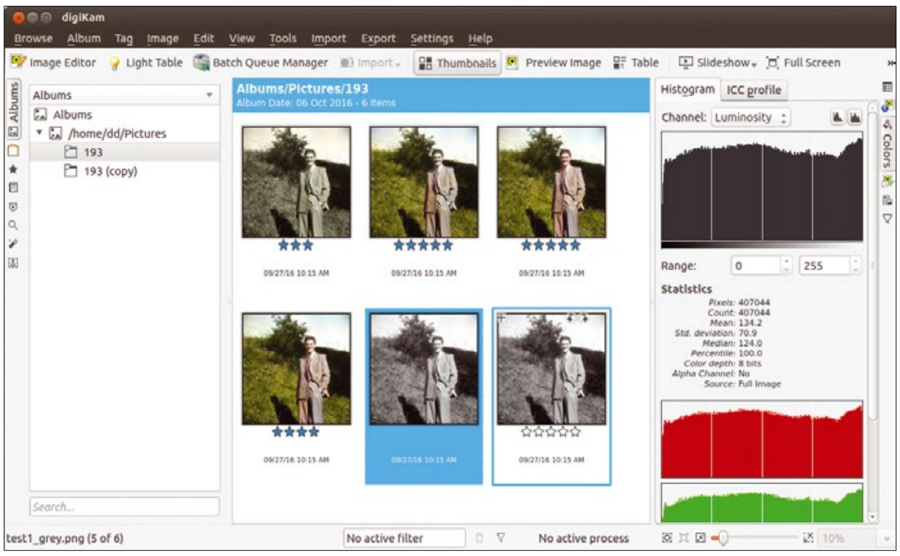
New features in digiKam 5

Fifth Generation

The freshly released digiKam 5 boasts a number of new features, brings many improvements, and ditches some legacy ballast. By Karsten Günther

If you have more than just a few snapshots in your photo archive, you need powerful software to manage them. The idea is not only to make the images accessible by filename, but in many other ways, as well, such as by time of recording, by keyword, or by geographic location.

For years, digiKam [1] has been considered the most important free software for managing and manipulate large volumes of images. The freshly published digiKam 5 ditches a fair amount of ballast and includes a number of conceptual changes, as well as many smaller additions. Version 5 largely replaces KDE dependencies with dependencies on Qt5 and introduces a number of new features. Version 5.1, the version used for this article, fixed a number of bugs; versions 5.2-5.4 made several improvements to the similarity search engine and include a complete re-write of video file support. Also new in v5.4 is the *Maintenance* tool (which runs processes in the background to maintain image collections and database contents) in the *Tools* menu. In this article, I'll take a detailed look at the fifth generation of digiKam.



Statistics
Pixels: 407044
Count: 407044
Mean: 134.2
Std. deviation: 70.9
Median: 124.0
Percentile: 100.0
Color depth: 8 bits
Alpha Channel: No
Source: Full Image

Figure 1: Structured design: On the left, select the source of the images that appear at the center; the details are shown on the right.

The Old and the New

DigiKam is based on a simple, clear-cut concept that is reflected in the design. The main window is divided into three panels (Figure 1): the image source (e.g., an album), the images it contains, and the details of the image(s) selected in the current folder or album. This structure has proven its value.

Along the left edge, you choose between individual albums (i.e., folders in the filesystem), keywords, (Exif) data, or search results. Whatever you enable by clicking on the appropriate button in the left bar is displayed by digiKam in the central main window. A second mouse click on the same button hides the left bar and thus expands the main window.

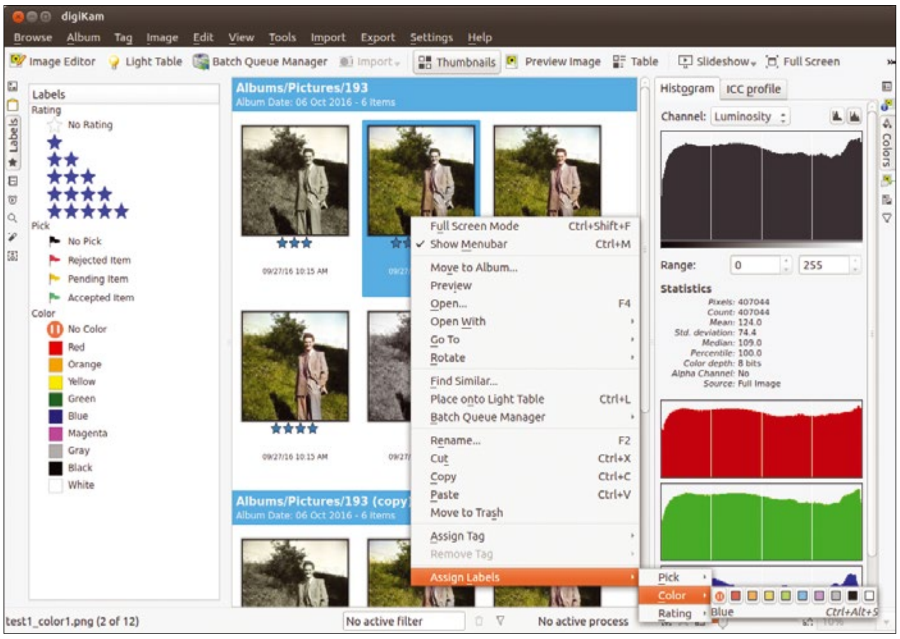
The new version of digiKam adds some new buttons in the left sidebar, which I will look at in more detail later. Showing the number of images in an album is a new function enabled by clicking *Settings* | *Configure digiKam*, choosing *Album View* in the Configure dialog, and checking the *Show a count of items in Tree Views* entry.

In the main window, digiKam features a new *Table* preview mode in which all the key data for the images are clearly summarized. Initially, you see a couple of columns, but you can add more with the contextual menu. Now, you also can select a theme for the entire program window in *Settings* | *Themes*.

The OpenGL extensions viewer is missing from the *Tools* menu. To make up for this, the function for creating panoramic views by merging multiple images now works again, although the Setup dialog complains about missing components. For this function to work, you need to install the Hugin program [2] with the package manager. DigiKam also provides an option for blending images of series of exposures.

DigiKam's main window has hardly changed. To select an image to be processed, either click on individual pictures with the mouse, Ctrl-click to select multiple images, or Shift-click to select whole areas. At top left are the plus and minus buttons for adding images to or removing them from the selection.

On the right side, the familiar tab applies special functions (e.g., adding keywords, accessing Exif tags, etc.) to the selected images. In the new release, version info for images, geodata, a filter to restrict the image selection in the main window, and access to the file attributes were added.



Statistics
Pixels: 407044
Count: 407044
Mean: 124.0
Std. deviation: 74.4
Median: 109.0
Percentile: 100.0
Color depth: 8 bits
Alpha Channel: No
Source: Full Image

Figure 2: In digiKam 5, images can be tagged with a number of abstract ratings that are easy to filter.

The functions in the status bar at the bottom of the window have also changed: You can no longer specify filters here; instead, the bar only shows the currently active filters, which you can disable by unchecking.

Details

The importance of searching grows as the number of images managed by digiKam increases; thus, you will want to familiarize yourself at an early stage with the buttons in the left sidebar so you can find your way around later when you have a larger image collection.

A search by labels and ratings is intuitive (Figure 2). In case of ratings, simply assign the number of asterisks desired to the images. Because this solution proved to be far too rigid, two new variants were added: picks (denoted by flags) and colors. For example, the flags can indicate the quality of the images (see the "Quality Management" box): The *Rejected Item* option lets you tag bad images that you only keep in the database because you lack a better image of the subject.

You can assign these flags in different ways: Either select the corresponding flag from the context menu of the main window below *Assign Labels* or – and this is far faster – use keyboard shortcuts: Alt+1 means "rejected," Alt+2 is "accepted," Alt+3 is "pending," and Alt+0 means you have assigned no rating. Unfortunately, many window managers intercept keyboard shortcuts (e.g., Alt+2 often opens a window in which you can enter a program name). That said, the keyboard shortcuts can be changed easily in *Settings* | *Configure Shortcuts*.

Color markings go a step further in the direction of abstraction: You can decide what they mean. For example, you could use them to tag particularly important images. Ten colors can be selected using the keyboard shortcuts Ctrl+Alt+1

QUALITY MANAGEMENT

digiKam uses the new flags in a very interesting way: Version 5 has an analysis tool to automatically determine image quality. DigiKam detects data on image sharpness, noise, exposure errors, and compression and formulates a characteristic value. First, enable sorting by image quality and define how exactly to sort under *Settings | Configure digiKam | Image Quality Sorter* (Figure 3). Then, choose *Tools | Maintenance | Image Quality Sorter* to assign quality labels to images according to your settings. Generally, quality analysis can be used only if no other maintenance task (e.g., face recognition) is active. Examining the images takes a huge amount of computing power and plenty of time, in particular for larger image stocks. Also, the analysis blocks the use other digiKam functions.

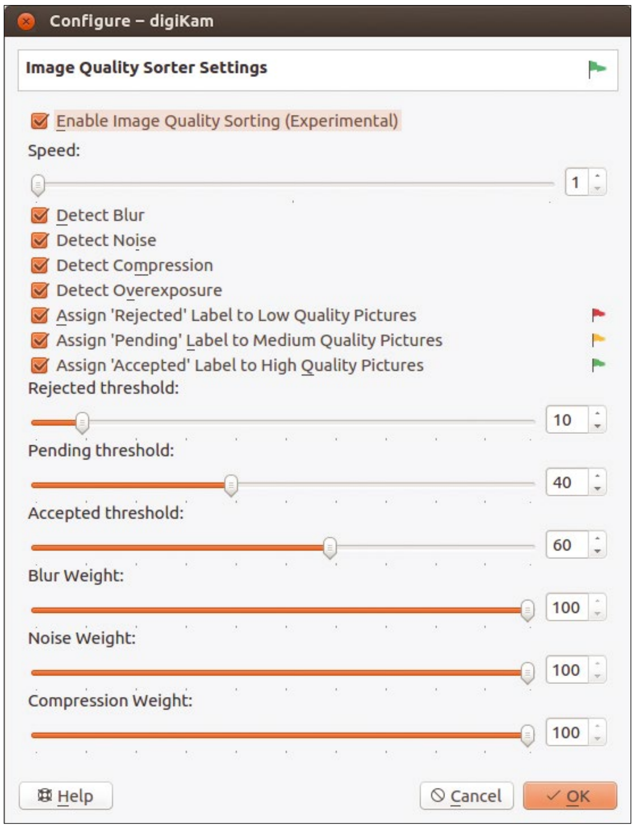

Configure – digiKam

Image Quality Sorter Settings

- ✓ Enable Image Quality Sorting (Experimental)

Speed: 1

- ✓ Detect Blur
- ✓ Detect Noise
- ✓ Detect Compression
- ✓ Detect Overexposure
- ✓ Assign 'Rejected' Label to Low Quality Pictures
- ✓ Assign 'Pending' Label to Medium Quality Pictures
- ✓ Assign 'Accepted' Label to High Quality Pictures

Rejected threshold: 10

Pending threshold: 40

Accepted threshold: 60

Blur Weight: 100

Noise Weight: 100

Compression Weight: 100

Figure 3: digiKam automatically determines image quality on request and retrieves the best images from the database without further intervention.

through Ctrl+Alt+0. Alternatively, you will also find all these labels in *Filters* tab in the right sidebar.

Geodata

Generally speaking, you have several ways to add geodata to images. The most universal and reliable method is still to do it manually, but modern cameras usually support automatic geotagging, so you can locate where you captured an image. Even GPS-enabled cameras, though, can make the occasionally error (e.g., failing to find satellites while taking the picture).

To modify the geodata, choose the pictures to be geolocated and call *Image* | *Geolocation* | *Edit Coordinates*. You can edit several images from different points of view in a single step. DigiKam always starts the wizard in overview mode, so you need to take a number of steps to assign the right location (Figure 4). Here, I'm using OpenStreetMap instead of the default, Marble.

Next, select the images for a location from the list and drag them onto the map at the location you took them. That is typically all you need to do to add the location coordinates. For digiKam to write the positions to the images, go to *Settings* | *Metadata* and check the *If possible write Metadata to RAW files* item. DigiKam now indicates that geolocations are present in the upper right corner of an image in the main window.

To use the *Map Searches* tab in the left sidebar, Ctrl-select the desired area in the map. Below *Search by area:* you can either select a region or a specific location (Figure 5). After clicking on image icons in the map, you will find your way around the view quickly.

In the map, you can exclude certain areas from the results (e.g., to remove unwanted images from the selection filter). The big *Show Non-Geolocated items* button lets you track down entries that lack geodata in the database.

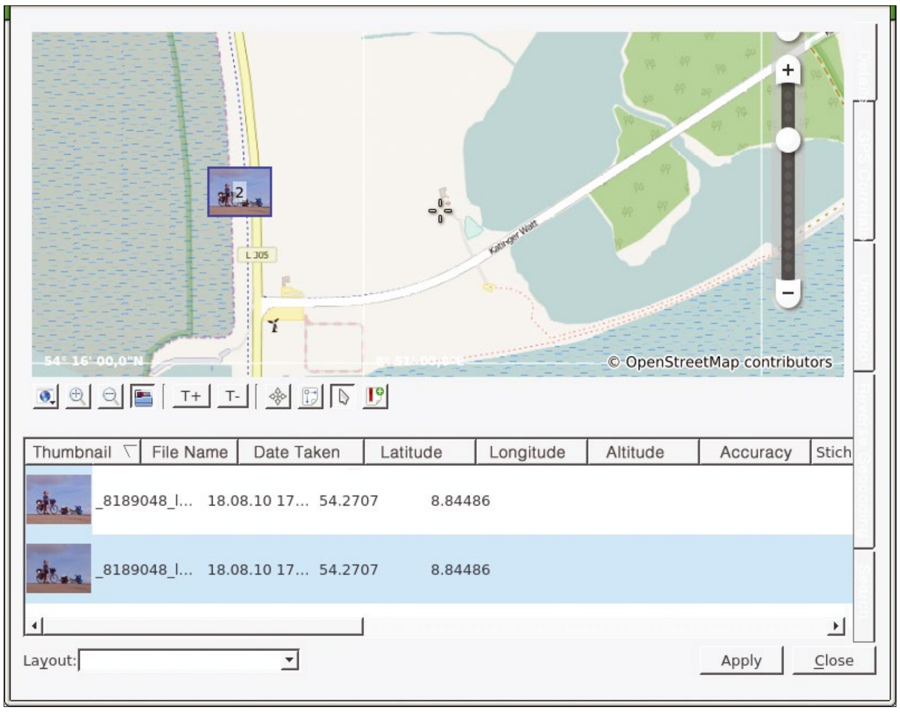
Familiar Faces

A topic that raises privacy questions on the Internet makes a great deal of sense when let loose on a local image database: face recognition. For a people search, you first need an additional database of faces, which digiKam creates as soon as you click the *People* tab in the left sidebar and enable *Scan collection for faces*.

In the Scanning faces dialog (Figure 6), digiKam shows you various options for creating the database. Preparing the data takes some time in a larger collection of images; later, you will want to update the database regularly in the *Maintenance* menu.

However, the underlying functions do not yet work well enough to use them without follow-up (Figure 7). That said, you have a few ways to influence the results in the Scanning faces dialog with the *Options* button at the bottom. For example, you can distribute the analysis across all processor cores, which greatly speeds up the editing but might also negatively effect the stability of the program. To watch digiKam at work indexing, start the program at the command line, and studying the output. This also works when creating other databases.

After selecting faces in the image database, you can use a new search function:



Thumbnail	File Name	Date Taken	Latitude	Longitude	Altitude	Accuracy	Stich
	_8189048_I...	18.08.10 17...	54.2707	8.84486			
	_8189048_I...	18.08.10 17...	54.2707	8.84486			

Figure 4: If your camera does not add GPS location data automatically to the snapshots, you can do so retroactively in digiKam.

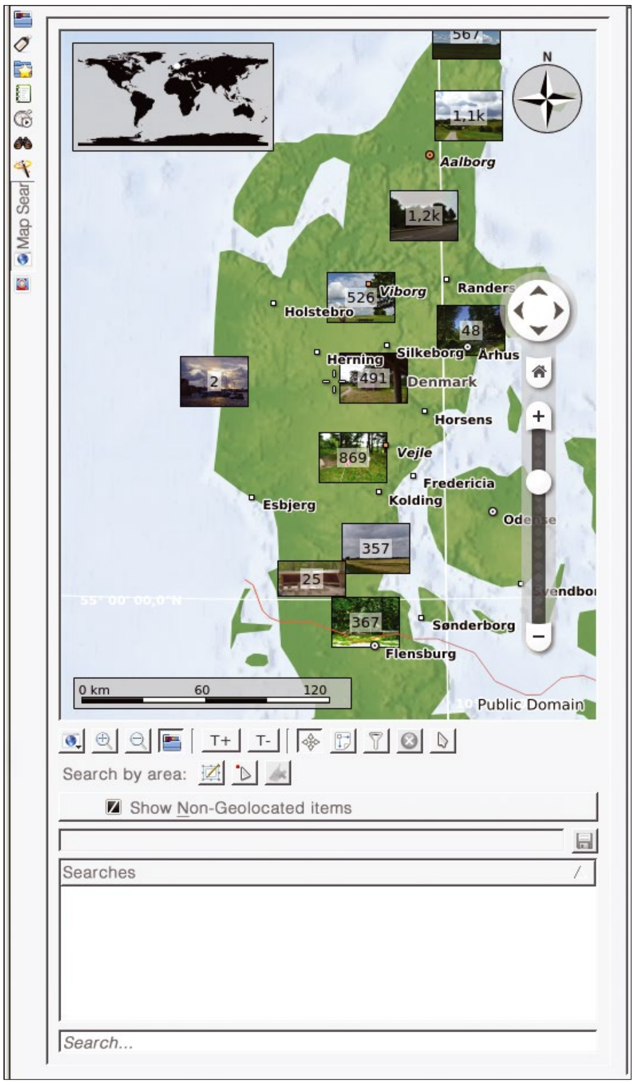
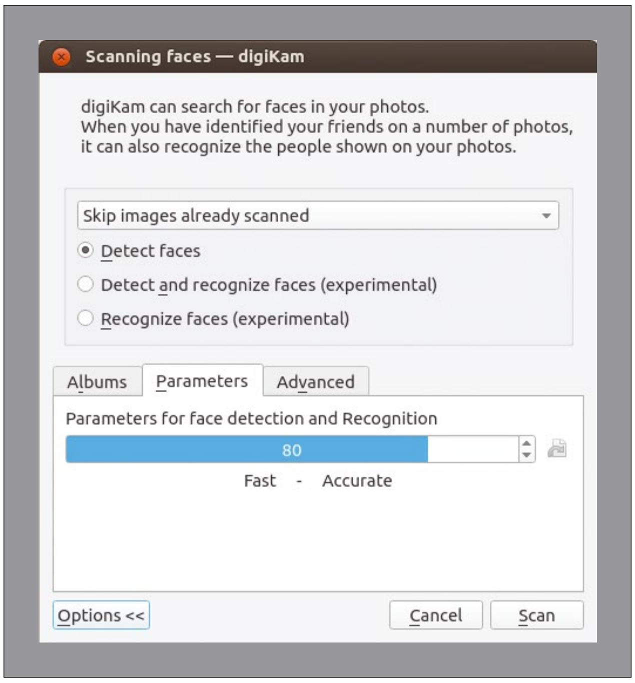


Figure 5: Searching for images in a given area.


Scanning faces — digiKam

digiKam can search for faces in your photos.
When you have identified your friends on a number of photos,
it can also recognize the people shown on your photos.

Skip images already scanned
(●) Detect faces
( ) Detect and recognize faces (experimental)
( ) Recognize faces (experimental)

Parameters for face detection and Recognition
80
Fast - Accurate

Figure 6: For a people search, digiKam needs to analyze all the images and create a database of faces.

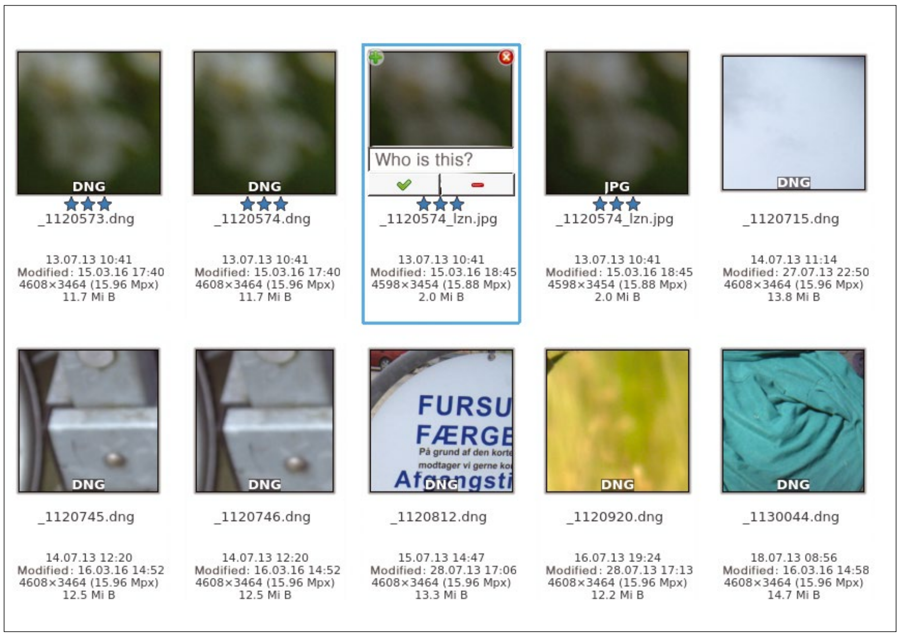


Figure 7: All of these image sections were recognized as faces by digiKam. You can click on the x at top right to remove false positives.

Browse | *Tag* helps you find names assigned to the faces.

Filters

Whenever the main window lists a large number of images, it is useful to check whether you can reduce the selection with a filter (symbolized by the funnel in the right sidebar) to essential shots only.

Filters are easy to enable in the *Filters* tab (Figure 8): In the various fields, select or enter what you want to search for. Filters such as the *Tags Filter* allow multiple entries. Others, such as the *MIME Type Filter*, have more limited support: *RAW Files* supports all RAW formats, *No RAW Files* supports all other formats. Special formats such as ORF and DNG cannot be specified in this way.

In the *Geolocation Filter*, you should not confuse the filter with the map search I looked at earlier: This setting filters on whether or not images have geolocation data. The *Tags Filter* now has a new function *Images Without Tags*. At the bottom of this dialog, the *Labels Filter* for flags is now available. There used to be a restricted version of this function in the main window; today, digiKam displays only the active filters and makes it possible to turn it off.

Various Innovations

DigiKam previously banished deleted images directly to the trash receptacle of the desktop environment, but the program now has its own *Trash* can (Figure 9), available as a regular album named *Trash*, and allows deleting and restoring from the album.

The new maintenance dialog under *Tools* | *Maintenance* (Figure 10) now acts as a hub for all regular tasks. In the dialog you can tell digiKam, if necessary, to renew the (auxiliary) databases or launch a comparison between internal databases and image files.

The developers also introduced a number of changes to the light table (Figure 11), where you compare and analyze similar photos. In practice, its op-

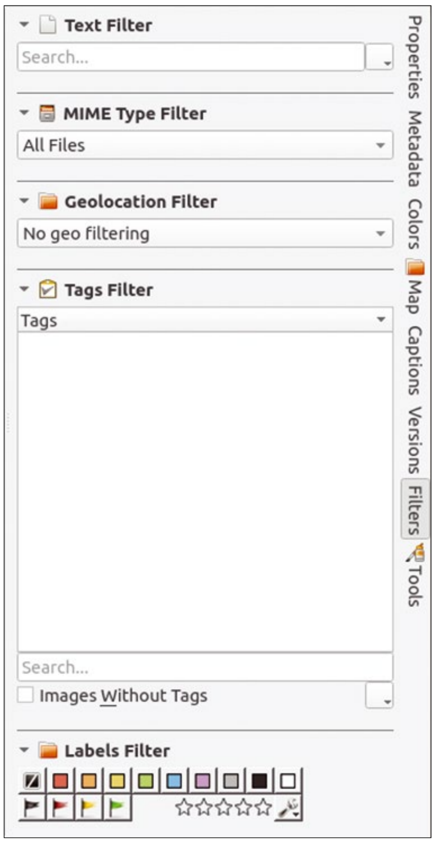
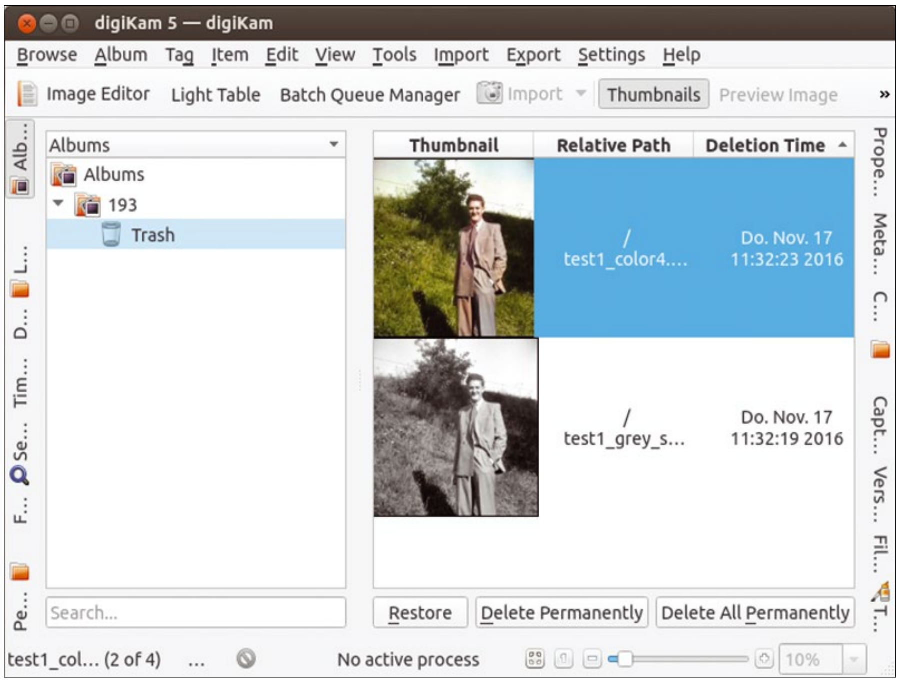

Text Filter
Search...
MIME Type Filter
All Files
Geolocation Filter
No geo filtering
Tags Filter
Tags
Search...
Images Without Tags
Labels Filter

Figure 8: The *Filters* tab provides a new, powerful way to restrict the images displayed in the main window.


digiKam 5 — digiKam

Albums
Albums
193
Trash

Thumbnail	Relative Path	Deletion Time
	/ test1_color4....	Do. Nov. 17 11:32:23 2016
	/ test1_grey_s...	Do. Nov. 17 11:32:19 2016

Search...
Restore | Delete Permanently | Delete All Permanently
test1_col... (2 of 4) ... No active process 10%

Figure 9: DigiKam's trash can is now better integrated into the system and allows the restoration of entries.

eration is intuitive, and you can display the relevant metadata alongside images. DigiKam always zooms and moves two images in parallel by default (the *By Pair* option), which facilitates comparisons at higher magnifications. You can now edit metadata (including geolocation) on both the light table and in the *Metadata* tab of the right sidebar.

Relatively little has happened in terms of functions for manipulating images. If you need an image editor, you are better advised to turn to an external RAW developer such as Darktable, or Gimp. The built-in RAW converter is now a little tidier, but it still only updates the preview after clicking and provides only a fraction of the kinds of features you would want.

The same applies for the image editor (Figure 12), which still has only a relatively small selection of features, such as sharpening or blurring images or noise reduction. The available filters are sufficient for many purposes and functions, but in terms of ease of use, it still has room for improvement. Only the new *Color* | *Auto-Correction* function turns out to be pretty user friendly.

In contrast to the image editor, image versioning takes high priority in digiKam. A first version of an image is even created during development with the program's RAW converter. Later, you can save new versions when applying filters using the edit function, so the original is always preserved and can be recovered, if needed.

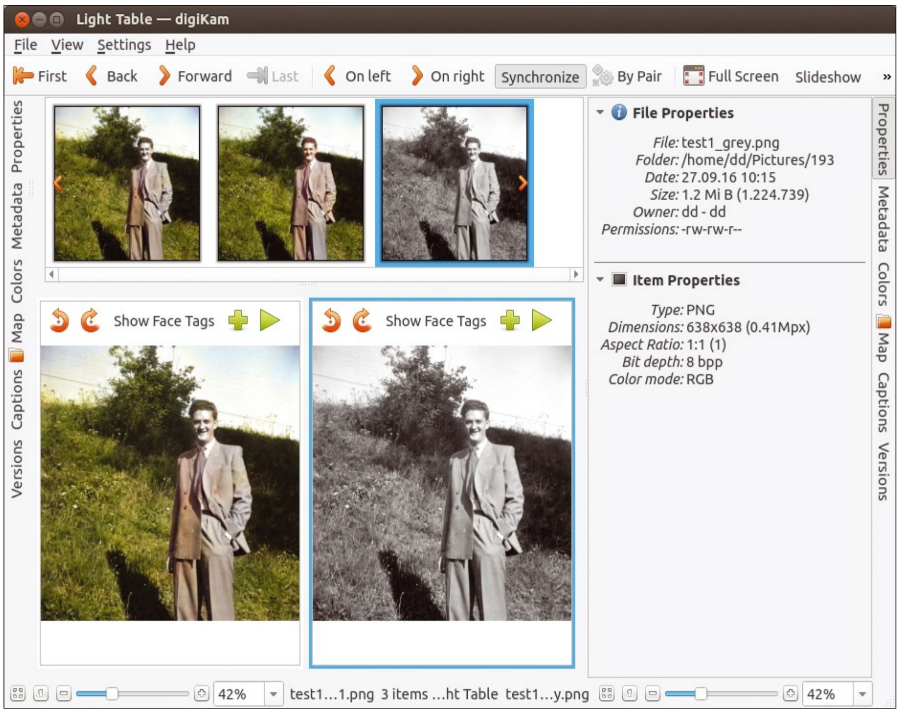
Conclusions

The fifth major version of digiKam displays significant improvements over its predecessors. The code and menus have been thoroughly cleaned up and more clearly structured, which benefits both the users and the developer community. DigiKam 5 has not yet fully resolved two problems: its relatively slow speed, despite the in-memory database, and some stability problems. ■■■

INFO

[1] digiKam: *https://www.digikam.org*

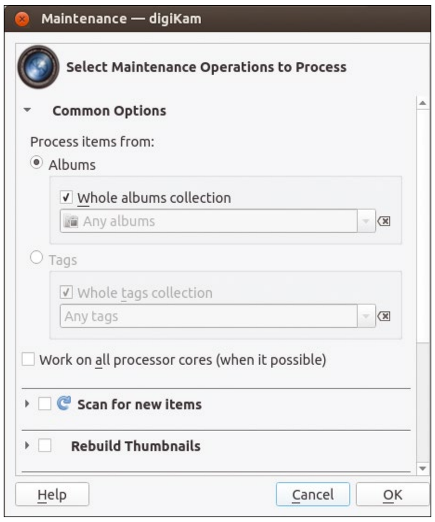
[2] Hugin: *http://hugin.sourceforge.net*


Light Table — digiKam
File View Settings Help
First Back Forward Last On left On right Synchronize By Pair Full Screen Slideshow
Versions Captions Map Colors Metadata Properties
Show Face Tags
Show Face Tags
42%
test1...1.png 3 items ...ht Table test1...y.png
42%
Properties Metadata Colors Map Captions Versions

File Properties
File: test1_grey.png
Folder: /home/dd/Pictures/193
Date: 27.09.16 10:15
Size: 1.2 Mi B (1.224.739)
Owner: dd - dd
Permissions: -rw-rw-r--

Item Properties
Type: PNG
Dimensions: 638x638 (0.41Mpx)
Aspect Ratio: 1:1 (1)
Bit depth: 8 bpp
Color mode: RGB

Figure 11: The Light Table displays the metadata next to the corresponding images.



Select Maintenance Operations to Process

Common Options

Process items from:

(●) Albums

[✓] Whole albums collection

Any albums

( ) Tags

[✓] Whole tags collection

Any tags

[ ] Work on all processor cores (when it possible)

[ ] Scan for new items

[ ] Rebuild Thumbnails

Figure 10: All important maintenance for digiKam is now managed in a single dialog.

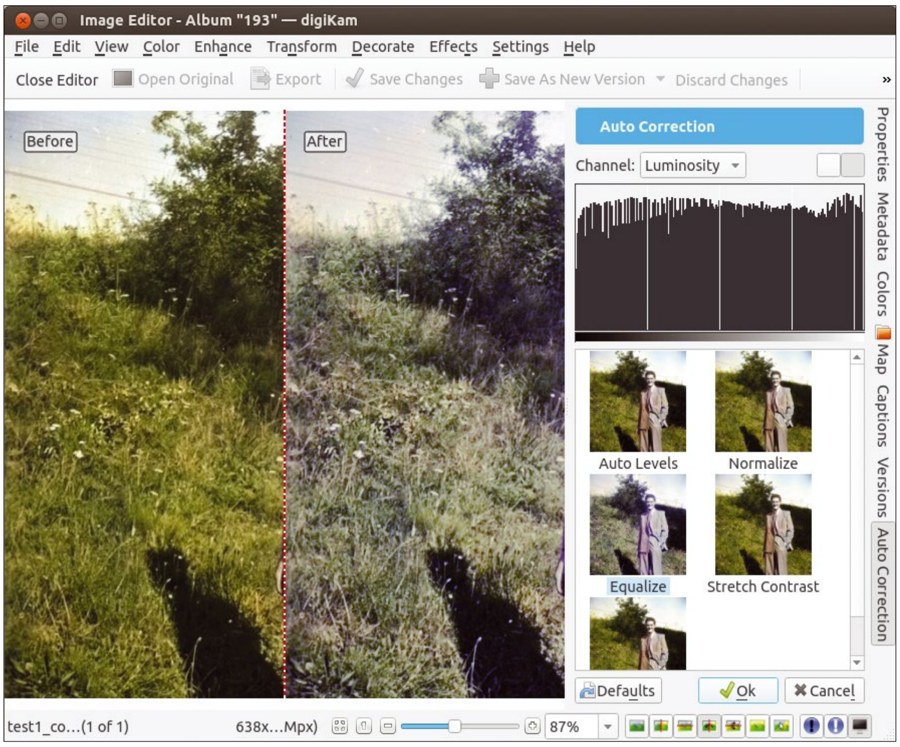


Image Editor - Album "193" — digiKam
File Edit View Color Enhance Transform Decorate Effects Settings Help
Close Editor Open Original Export Save Changes Save As New Version Discard Changes
Before
After
Auto Correction
Auto Levels
Normalize
Equalize
Stretch Contrast
Defaults
Ok
Cancel
Properties Metadata Colors Map Captions Versions Auto Correction
test1_co...(1 of 1)
638x...Mpx)
87%

Channel: Luminosity

Figure 12: The built-in image processing feature offers little in the way of functionality but meets the requirements in most cases.

Traffic analysis tools for websites

Data for Breakfast

If you are looking for an alternative to Google Analytics for studying website data, you can choose from a few free alternatives. In this article, we look at Piwik, Open Web Analytics, and eAnalytics. By Ferdinand Thommes

Admins who wanted details of the visitors to their websites in the early years of the Internet had to laboriously read the web server's logs. The first log file analysis applications appeared 20 years ago. Analog [1], Webalizer [2] and AWStats [3], which date from this period, are still occasionally in use (see the "Simple Web Analytics Tools" box).

In 2005, Google launched Google Analytics (GA) [4], a website analysis service that is hugely popular today. Open source tools such as Piwik [5] picked up on this trend towards graphical web analytics, but moved its focus to the customer's own server.

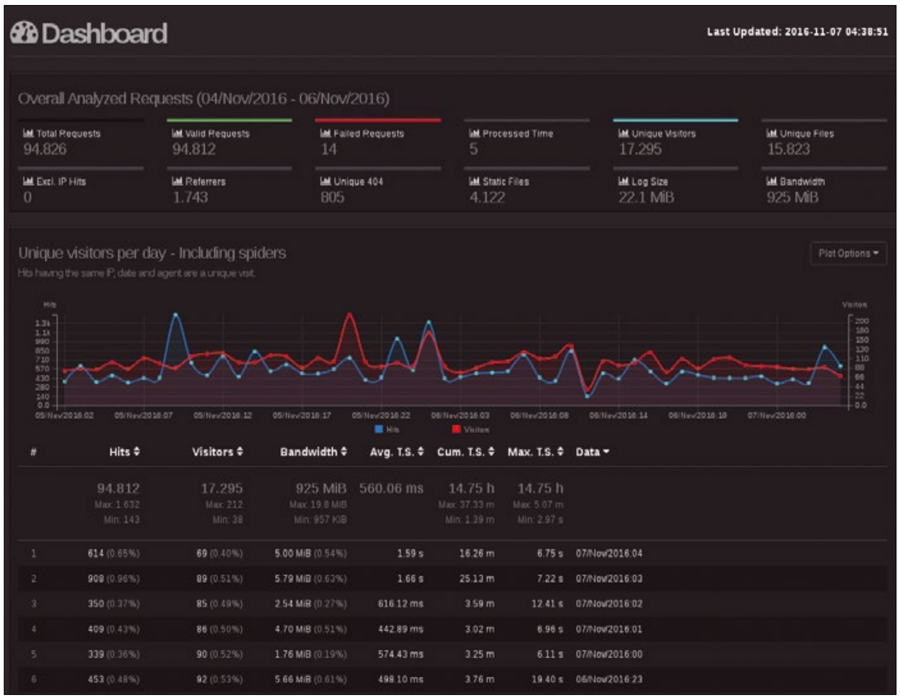
With the help of web analytics, site operators collect and evaluate data on the surfing habits of their visitors. The access data are of interest not only for commercial reasons; the companies behind the sites also often seek to better understand their customers and their interests. The following applies: The closer an operator knows the visitors and their preferences, the better

SIMPLE WEB ANALYTICS TOOLS

Many system administrators are quite happy with the simpler, resource-friendly log evaluations provided by statistics tools.

The oldest open source tools include Analog developed in 1995 and Webalizer first released in 1997. Both applications are still regularly updated today. The tools evaluate the logs several times a day, when run by the admin or a cron job. AWStats is also a simple analysis program. It has generated statistics about web page visits since 2000 and is still under active development. The script, implemented entirely in Perl, uses logfile analysis on web, mail, and FTP servers to produce its reports as HTML pages. Simple bar charts graphically enhance the results.

GoAccess [6] (Figure 1) gives the admin the ability to output and continuously update analyses in real time in a terminal or in a browser. GoAccess can handle virtually any log format used by Apache, Nginx, Amazon S3, Elastic Load Balancing, CloudFront, and others.


Dashboard
Last Updated: 2016-11-07 04:38:51

Overall Analyzed Requests (04/Nov/2016 - 06/Nov/2016)

Total Requests	Valid Requests	Failed Requests	Processed Time	Unique Visitors	Unique Files
94.826	94.812	14	5	17.295	15.823
Excl. IP Hits	**Referrers**	**Unique 404**	**Static Files**	**Log Size**	**Bandwidth**
0	1.743	805	4.122	22.1 MiB	925 MiB

Unique visitors per day - Including spiders
Hits having the same IP, date and agent are a unique visit.

#	Hits	Visitors	Bandwidth	Avg. T.S.	Cum. T.S.	Max. T.S.	Data
	94.812 Max: 1.632 Min: 143	17.295 Max: 212 Min: 38	925 MiB Max: 19.8 MiB Min: 957 KiB	560.06 ms	14.75 h Max: 37.33 m Min: 1.39 m	14.75 h Max: 5.07 m Min: 2.97 s	
1	614 (0.65%)	69 (0.40%)	5.00 MiB (0.54%)	1.59 s	16.26 m	6.75 s	07/Nov/2016:04
2	908 (0.96%)	89 (0.51%)	5.79 MiB (0.63%)	1.66 s	25.13 m	7.22 s	07/Nov/2016:03
3	350 (0.37%)	85 (0.49%)	2.54 MiB (0.27%)	616.12 ms	3.59 m	12.41 s	07/Nov/2016:02
4	409 (0.43%)	86 (0.50%)	4.70 MiB (0.51%)	442.89 ms	3.02 m	6.96 s	07/Nov/2016:01
5	339 (0.36%)	90 (0.52%)	1.76 MiB (0.19%)	574.43 ms	3.25 m	6.11 s	07/Nov/2016:00
6	453 (0.48%)	92 (0.53%)	5.66 MiB (0.61%)	498.10 ms	3.76 m	19.40 s	06/Nov/2016:23

Figure 1: GoAccess demo application in the browser.

the operator can optimize its offerings to suit the target group.

Good to Know

Site operators are often interested in where the visitors come from, what they are looking for, what items they click on, and how long they remain on the site. It can also be useful to know when they leave the site. Admins want to know what browsers and operating systems visitors to the site use, which files and documents they download and with what bandwidth, and how many visitors subscribe to newsletters or RSS feeds.

Web shop operators are interested in how many visitors add goods to their shopping carts, to then purchase them, or possibly not. If a website hosts advertising for third parties, web analysis is essential, because access figures and similar factors determine the prices for advertisers.

Open Access

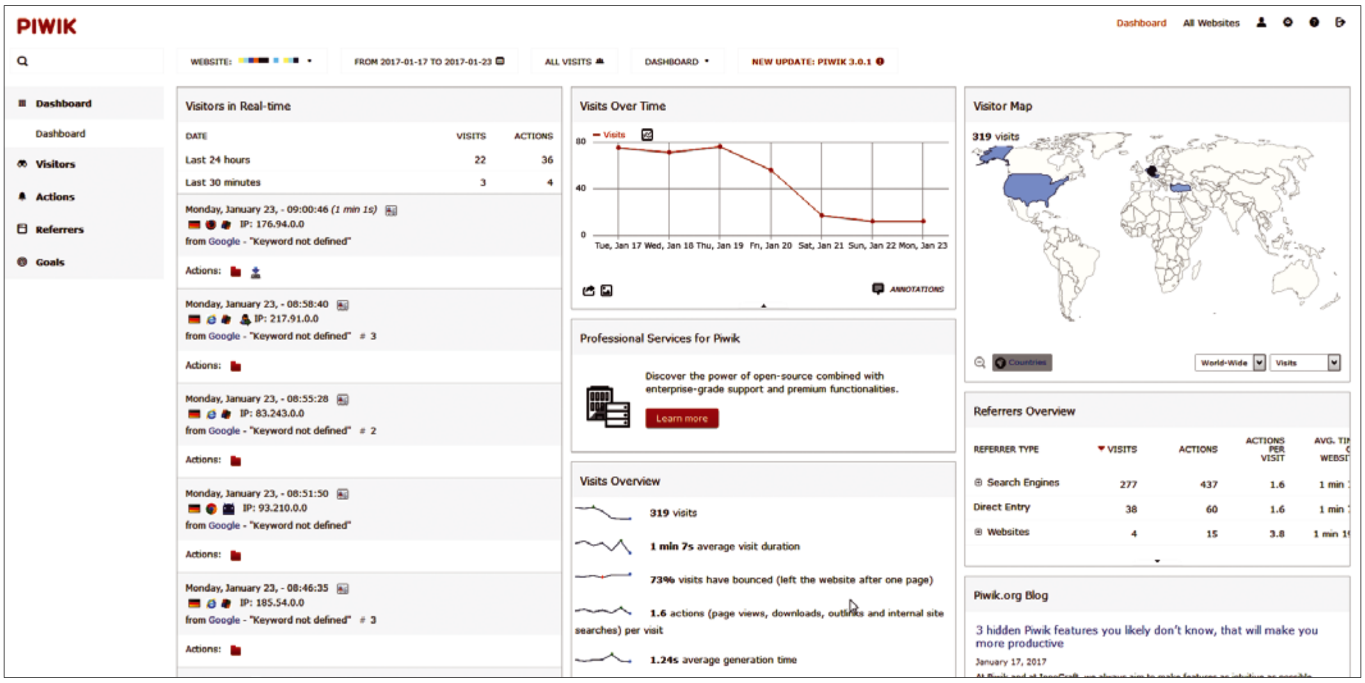
The market offers many different web analytics tools. They include around 150 commercial, typically proprietary applications, aimed at larger corporate websites. There are also some free and partly also open source tools. This article looks at Piwik, Open Web Analytics [7], and eAnalytics [8] (Table 1).

From a technical point of view, the web analytics tools either prepare web server logfiles, or special tags integrated into the HTML web pages giving the admin statistics and graphics for a quick overview and access to all necessary key indicators. Although the server-based method analyzes the logfiles of the web server, developers of the client-based variant add tracking pixels into the source code of the web page to determine the key indicators.

Although none of the two methods fully represents the actual traffic of a website, the client-based system of counting pixels, combined with the controversial use of cookies, is currently just about winning the accuracy stakes.

Privacy Issues

Because they evaluate cookies and store the visitors' IP addresses, web analytics tools always face a difficult legal situation. For example, Germany's Telemedia Act (TMG) [9] allows you to create user profiles if the user does not object to the purposes of advertising and market re-


PIWIK
Dashboard   All Websites
WEBSITE: FROM 2017-01-17 TO 2017-01-23   ALL VISITS   DASHBOARD   NEW UPDATE: PIWIK 3.0.1

Dashboard
Dashboard
Visitors
Actions
Referrers
Goals

Visitors in Real-time

DATE	VISITS	ACTIONS
Last 24 hours	22	36
Last 30 minutes	3	4

Monday, January 23, - 09:00:46 *(1 min 1s)*
IP: 176.94.0.0
from Google - "Keyword not defined"
Actions:

Monday, January 23, - 08:58:40
IP: 217.91.0.0
from Google - "Keyword not defined" # 3
Actions:

Monday, January 23, - 08:55:28
IP: 83.243.0.0
from Google - "Keyword not defined" # 2
Actions:

Monday, January 23, - 08:51:50
IP: 93.210.0.0
from Google - "Keyword not defined"
Actions:

Monday, January 23, - 08:46:35
IP: 185.54.0.0
from Google - "Keyword not defined" # 3
Actions:

Visits Over Time

Professional Services for Piwik
Discover the power of open-source combined with enterprise-grade support and premium functionalities.
Learn more

Visits Overview
319 visits
1 min 7s average visit duration
73% visits have bounced (left the website after one page)
1.6 actions (page views, downloads, outlinks and internal site searches) per visit
1.24s average generation time

Visitor Map

Referrers Overview

REFERRER TYPE	VISITS	ACTIONS	ACTIONS PER VISIT	AVG. TI… WEBSI…
Search Engines	277	437	1.6	1 min
Direct Entry	38	60	1.6	1 min
Websites	4	15	3.8	1 min 1

Piwik.org Blog
3 hidden Piwik features you likely don't know, that will make you more productive
January 17, 2017

Figure 2: Piwik is the best known and functionally the most similar open source alternative to Google Analytics.

TABLE 1: Three Web Statistics Tools at a Glance

	Piwik	Open Web Analytics (OWA)	eAnalytics
Platforms	Cross-Platform	Cross-Platform	Debian/Ubuntu
License	GPLv3 and others	GPLv2	AGPLv3
Under development since	2009	2009	2011
Language	PHP	PHP	Java and others
Methods	JavaScript tags, log analysis, tracking pixels	JavaScript tags, log analysis, tracking pixels	eAnalytics tag, tracking pixels
Functions	Visitors (visitors, unique visitors), operating system, browser version, downloads, IP address (pseudonymization capable), geolocation by city, page impressions, referrer, plugins	Visitors (visitors, unique visitors), operating system, downloads, browser version, IP address, geo location by country, page impressions, referrer, heat maps	Visitors (visitors, unique visitors), operating system, downloads, browser version, IP address (pseudonymization capable, can be switched off), geolocation by city, page impressions, referrer, plugins

search. Such a profile is only allowed to contain an anonymized IP address in addition to the data on the use of the website. IP addresses are typically automatically truncated to this end.

The TMG also requires the service provider to inform the user in a privacy statement on the website of whether, to what extent, and for what purpose it processes the IP address. And, the TMG stipulates that users must have an option to object to the creation of user profiles.

Probably the most controversial and at the same time most successful tool for website traffic analysis pages is the Google Analytics online service, which was launched in 2005, and which 50 percent of all websites employ. It is clearly the top dog. In contrast to the applications covered in this article, the data collected by GA leaves users' computers and heads to the United States, where data protection provisions are not as stringent as in Germany and the rest of Europe.

For example, GA delivers the unabridged IP address to the parent company. Also, the website visitor may not necessarily be informed of the fact that Google is collecting its data. Browser add-ons like Ghostery or NoScript can disable GA [10] to provide protection against unwanted data collection. GA doesn't cost anything up to a traffic volume of 10 million hits a month, but it only delivers certain data following a 24-hour delay. Also, the user has to agree to Google's using the data for its own purposes.

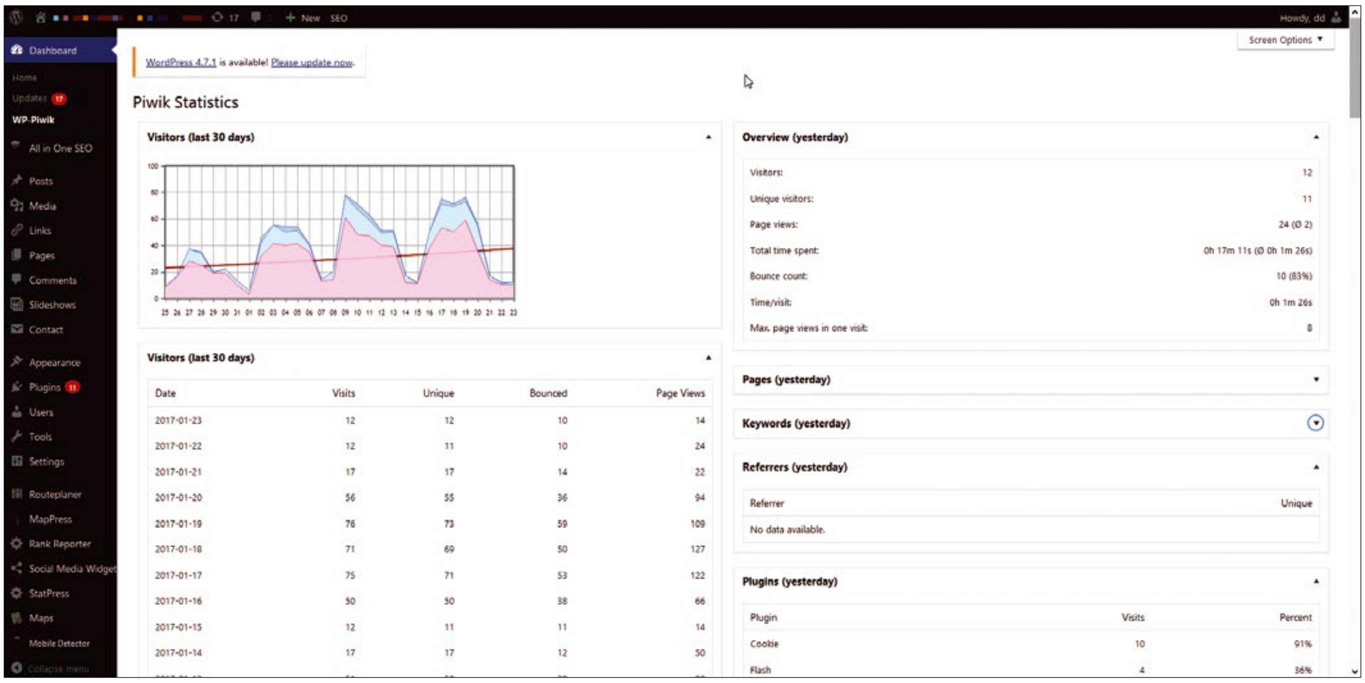
Piwik

Piwik (Figure 2) is growing in popularity around the world. Users have downloaded the cross-platform independent, open source program, which is written in PHP, almost three million times since 2008. Fans of GA will most likely find the functions they are familiar with from Google in Piwik, Site visitors are offered an opt-out in an IFrame, and Piwik respects the browser's Do Not Track feature.

Piwik collects data with tracking pixels, JavaScript, log analysis, and cookies, and stores these in a MySQL database. Access is via a web interface. The latest version, 3.0.1, introduces a new user interface on the basis of Material Design and Angular 1.4. Piwik is available under the GPLv3, but partly also under the BSD license.

Data Collection

Piwik Web Analytics collects relevant data such as visitor counts, keywords,



WordPress 4.7.1 is available! Please update now.

Piwik Statistics

Visitors (last 30 days)

Visitors (last 30 days)

Date	Visits	Unique	Bounced	Page Views
2017-01-23	12	12	10	14
2017-01-22	12	11	10	24
2017-01-21	17	17	14	22
2017-01-20	56	55	36	94
2017-01-19	76	73	59	109
2017-01-18	71	69	50	127
2017-01-17	75	71	53	122
2017-01-16	50	50	38	66
2017-01-15	12	11	11	14
2017-01-14	17	17	12	50

Overview (yesterday)

Visitors:	12
Unique visitors:	11
Page views:	24 (Ø 2)
Total time spent:	0h 17m 11s (Ø 0h 1m 26s)
Bounce count:	10 (83%)
Time/visit:	0h 1m 26s
Max. page views in one visit:	8

Pages (yesterday)

Keywords (yesterday)

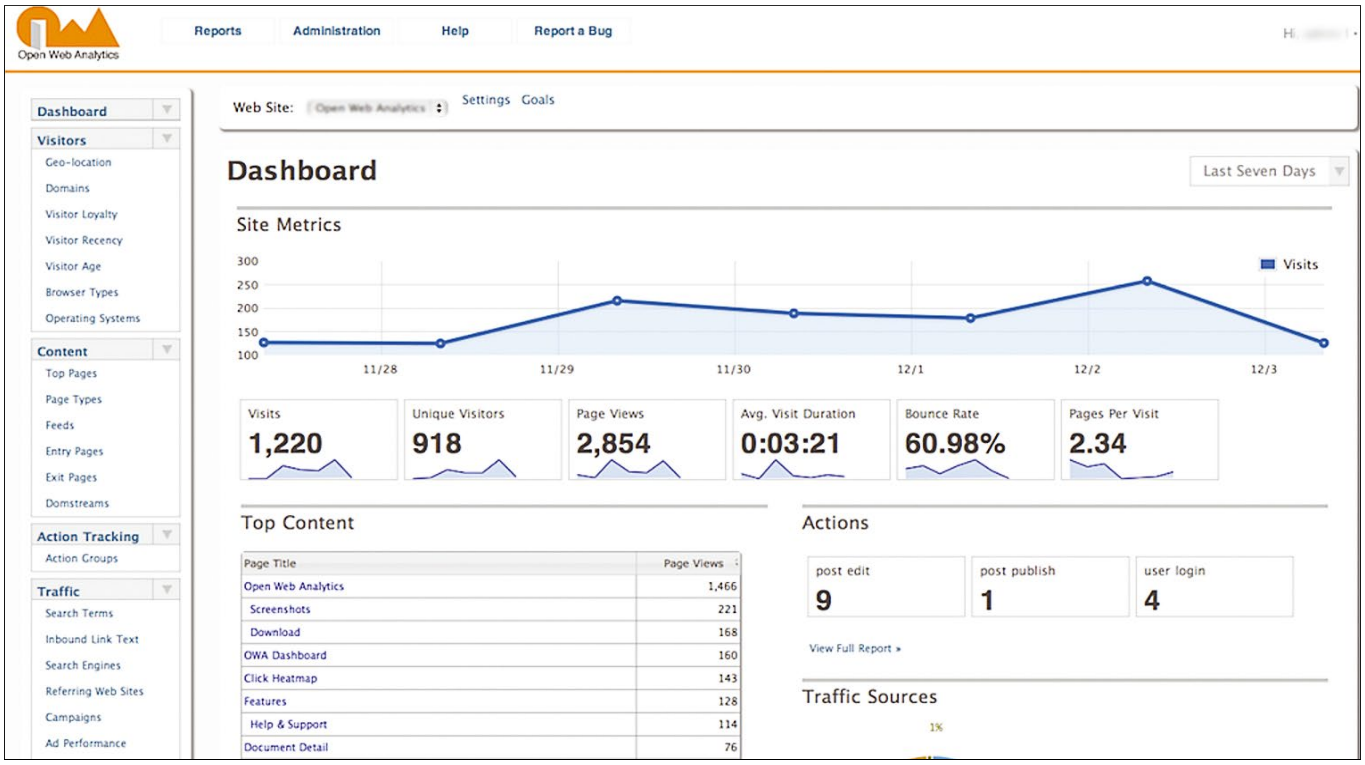
Referrers (yesterday)

Referrer	Unique
No data available.	

Plugins (yesterday)

Plugin	Visits	Percent
Cookie	10	91%
Flash	4	36%

Figure 3: Piwik can be combined with various web applications, for example, WordPress.



Dashboard

Site Metrics

Visits	Unique Visitors	Page Views	Avg. Visit Duration	Bounce Rate	Pages Per Visit
1,220	918	2,854	0:03:21	60.98%	2.34

Top Content

Page Title	Page Views
Open Web Analytics	1,466
Screenshots	221
Download	168
OWA Dashboard	160
Click Heatmap	143
Features	128
Help & Support	114
Document Detail	76

Actions

post edit	post publish	user login
9	1	4

View Full Report »

Traffic Sources

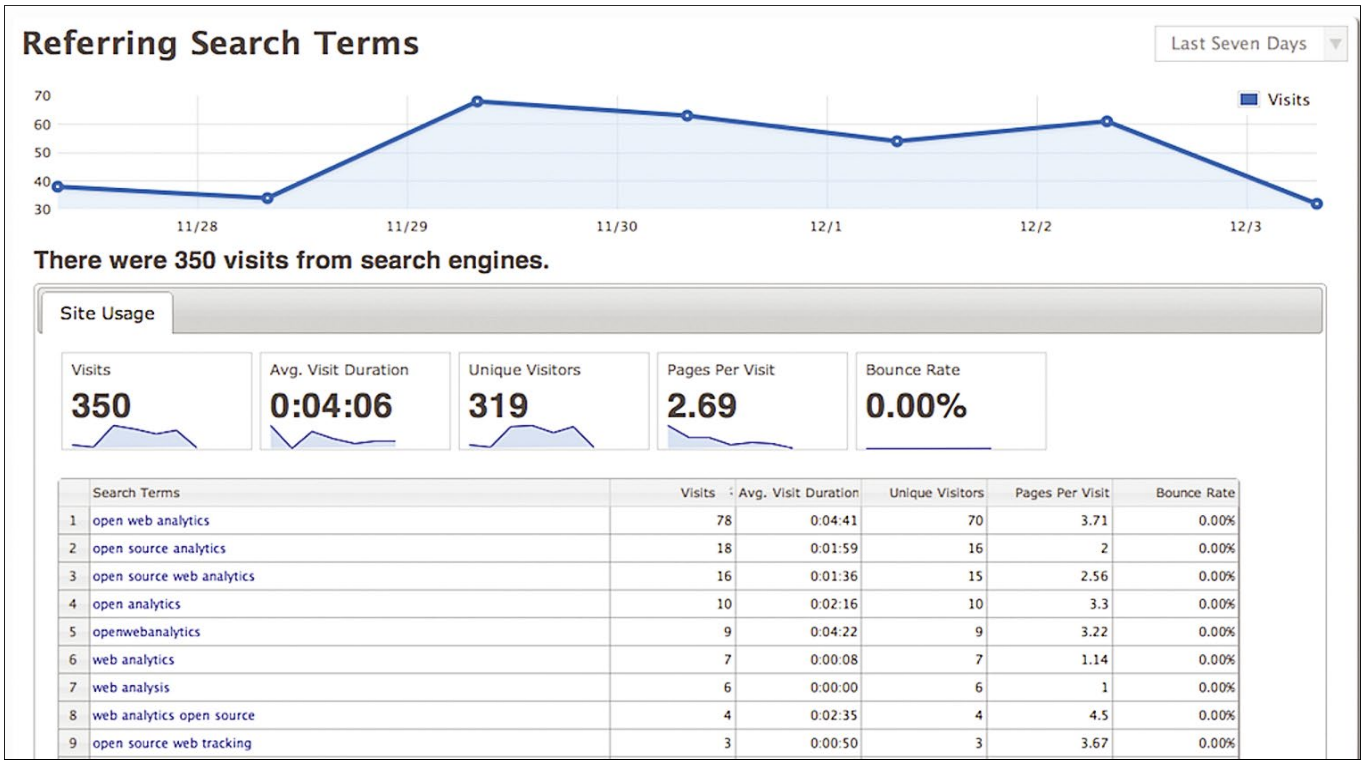
Figure 4: You can try out Open Web Analytics with the aid of an online demo.

referrers, and much more. This data tool prepares the data in a graphically appealing way and delivers the results in the form of reports. These include statistics on page views and unique visits. The visitor analysis also provides information on the countries of origin and the browsers and operating systems used. Referrers tell the operator which website sent a visitor to their offering.

The tool relies on plugins to implement its functions; you can add or remove these as needed. Piwik supports real-time updates, shows developments and trends, offers campaign and target tracking for online stores, and is multi-client capable for multiple websites.

Thanks to the configurable dashboard, admins can manage all their sites at a glance. A tool for aliasing the IP addresses is also part of the package and thus ensures data protection. There are also corresponding apps for the iOS and Android platform. Last but not least, users will find a detailed list of features with in-depth explanations [11] on the project website.

Admins can use plugins to extend the already abundant wealth of features that Piwik comes with out the box. A recent



Referring Search Terms

There were 350 visits from search engines.

Visits	Avg. Visit Duration	Unique Visitors	Pages Per Visit	Bounce Rate
350	0:04:06	319	2.69	0.00%

	Search Terms	Visits	Avg. Visit Duration	Unique Visitors	Pages Per Visit	Bounce Rate
1	open web analytics	78	0:04:41	70	3.71	0.00%
2	open source analytics	18	0:01:59	16	2	0.00%
3	open source web analytics	16	0:01:36	15	2.56	0.00%
4	open analytics	10	0:02:16	10	3.3	0.00%
5	openwebanalytics	9	0:04:22	9	3.22	0.00%
6	web analytics	7	0:00:08	7	1.14	0.00%
7	web analysis	6	0:00:00	6	1	0.00%
8	web analytics open source	4	0:02:35	4	4.5	0.00%
9	open source web tracking	3	0:00:50	3	3.67	0.00%

Figure 5: OWA collects statistics on referrers – as shown here – by integrating JavaScript into the landing page.

addition is the premium plugins, of which the project recently presented three [12]. Admins need to store Piwik software on the server and then install the system in a browser. If needed, you can integrate Piwik via enhancements in WordPress (Figure 3), MediaWiki, Magento, Joomla, vBulletin, and more than 60 other applications.

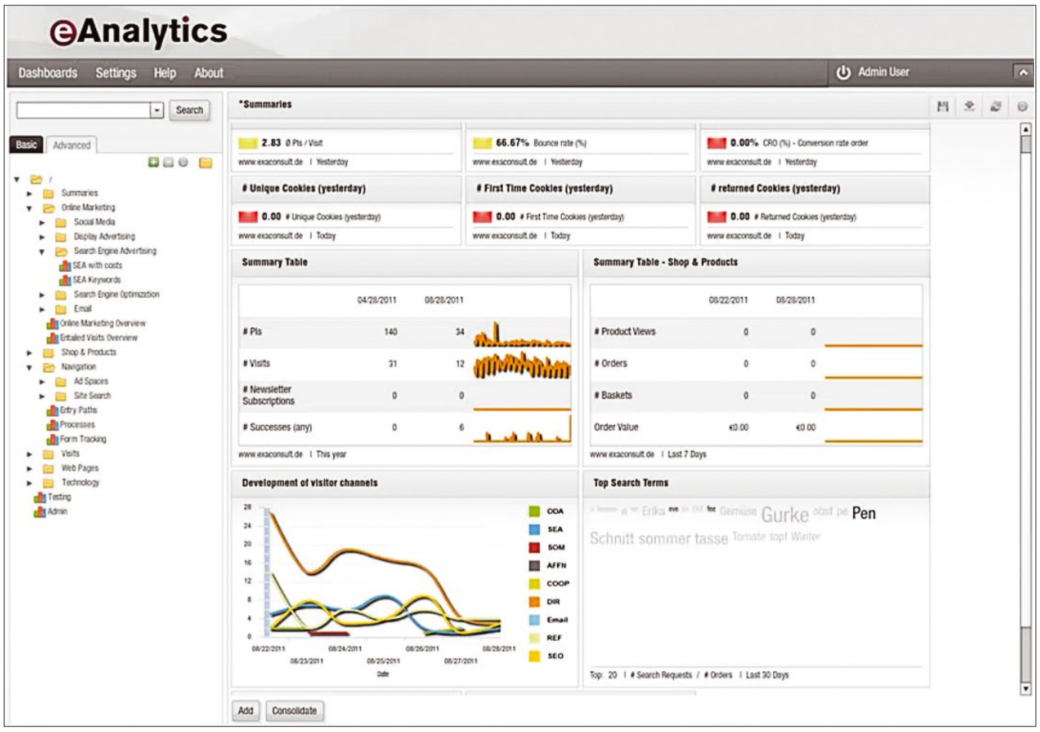
A demo version is available [13] on the project website. Piwik Pro sees the application run in the cloud [14]. Piwik is under active development and has a fairly lively community.

Open Web Analytics

Under development since 2009, Open Web Analytics (OWA) is not as well known as Piwik, but it keeps pace with Piwik and GA in terms of its feature set, even outpacing them in places (Figure 4). For example, it offers integrated heat maps, which competitors need to load as extensions. They help admins analyze mouse movements on web pages.

OWA uses a PHP front end with a MySQL back end and collects statistics by integrating a JavaScript into the target site (Figure 5). Admins can evaluate the results using JavaScript, but also directly via PHP or REST-based APIs. OWA supports WordPress or MediaWiki pages, and a third-party extension exists for Drupal.

The OWA framework is released under the GPLv2 license and is also suitable for campaign and e-commerce tracking.



*Summaries

2.83 Ø PIs / Visit — www.exaconsult.de | Yesterday

66.67% Bounce rate (%) — www.exaconsult.de | Yesterday

0.00% CRO (%) - Conversion rate order — www.exaconsult.de | Yesterday

Unique Cookies (yesterday): 0.00 # Unique Cookies (yesterday) — www.exaconsult.de | Today

First Time Cookies (yesterday): 0.00 # First Time Cookies (yesterday) — www.exaconsult.de | Today

returned Cookies (yesterday): 0.00 # Returned Cookies (yesterday) — www.exaconsult.de | Today

Summary Table

	04/28/2011	08/28/2011
# PIs	140	34
# Visits	31	12
# Newsletter Subscriptions	0	0
# Successes (any)	0	6

www.exaconsult.de | This year

Summary Table - Shop & Products

	08/22/2011	08/28/2011
# Product Views	0	0
# Orders	0	0
# Baskets	0	0
Order Value	€0.00	€0.00

www.exaconsult.de | Last 7 Days

Development of visitor channels

Top Search Terms

Top: 20 | # Search Requests / # Orders | Last 30 Days

Figure 6: eAnalytics is open source software from Germany. The figure shows the default view; the latest version of the software is available as a package for Ubuntu 14.04.

Users can define reports and dashboards that go beyond the standard selection to suit their needs. The application lets you integrate various web pages, which the OWA user can aggregate or view individually. It handles many administrative tasks directly at the command line of the server instead of in the browser.

The integrated event queuing is a unique feature among the applications presented here. If the database cannot process peak visiting times quickly enough, it first writes the data to a simple logfile (Flat File Based Event Queuing, [15]) and then parses it via a PHP call:

```
/path/to/php5 cli.php ↩
cmd=processEventQueue
```

Piwik offers a similar function in the form of Queued Tracking [16], which was added in the form of a plugin in 2015. But, in this case, the software writes its data to a Redis instance rather than directly to a classic database.

eAnalytics

eAnalytics (Figure 6) is not well known internationally, but is popular in Germany. The analysis tool, released by Integrated Analytics GmbH five years ago as open source, is designed for medium-sized enterprises. It seeks to make the technologies used in large-scale companies affordable for firms with fewer resources.

The focus is on merging data. eAnalytics seeks to meaningfully link data from web analytics with enterprise data from CRM systems, data mining and warehousing, and marketing systems.

The company offers support and managed services for eAnalytics and will build enterprise-specific extensions if necessary. eAnalytics provides a simple user opt-out like Piwik. At the same time, it honors Do Not Track requests from the browser. IP addresses can be pseudonymized in the configuration or not collected in the first place.

The software is released under the AGPLv3 license. Data from the tags of the page, external data from Google AdWords, a proprietary Twitter extension, and company-specific data serve as the data sources.

eAnalytics preconfigures 10 dashboards that can be extended using widgets. At the same time, it offers 55 interactive reports. You can install version 1.1.3 directly on the server as a Debian package; the current version is optimized for Ubuntu 14.04. Source code is available on SourceForge, but only for the older version 0.9.2 [17].

eAnalytics collects data with a JavaScript tool named eAnalytics Tag [18], which the server operator needs to additionally integrate [19]. One advantage of the web analytics software is that admins can distribute it to several machines in environments with many servers. For example, you can set up a separate server for the Tags component. You can explore eAnalytics upfront via a VMware image [20].

Conclusions

Fundamentally, the tools examined here – Piwik, OWA, and eAnalytics – do approximately the same thing as GA. Given appropriate hardware, the candidates can be used for websites with several million page views a day. The clear advantage is that all three keep the data on your own servers. This makes it easy for admins to comply with European data protection regulations.

Whether hosting locally is an advantage or a disadvantage for you is something you have to decide for yourself. But if you want to install, update, and maintain the applications yourself, you will need powerful hardware – in contrast to a scenario with GA. Piwik and eAnalytics, at least, offer supervised hosting.

If you completely reject GA, but do not have sufficiently powerful hardware for the mainline programs featured here, you can turn as an alternative to lean logfile analysis programs such as AWStats, Webalizer, or Analog. These may not provide the same wealth of information as the analytic applications, but they do still prepare the data in a clear-cut and graphically appealing way. They are fine if you only need an approximate overview.

Piwik has the edge in terms of popularity, dissemination, and developer community, which improves reliability and allows for long-term planning. The situation is not so clear with the two other candidates. Although both published their latest versions in the last twelve months, growth is far more restrained all told, and there are fewer developers on board. If you are interested in the integration of the acquired data with your business data, eAnalytics is the obvious choice; however, this unfortunately means doing without access to the source code for the current versions. ■■■

AUTHOR

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

INFO

[1] Analog: *https://en.wikipedia.org/wiki/Analog_(program)*

[2] Webalizer: *http://www.webalizer.org*

[3] AWStats: *http://www.awstats.org*

[4] Google Analytics: *https://analytics.google.com*

[5] Piwik: *https://piwik.org*

[6] GoAccess: *https://goaccess.io*

[7] Open Web Analytics: *http://openwebanalytics.com*

[8] eAnalytics: *http://eanalytics.de*

[9] Telemedia Act (in German): *http://www.gesetze-im-internet.de/tmg/__15.html*

[10] Google Analytics opt-out: *https://tools.google.com/dlpage/gaoptout*

[11] Piwik features: *http://piwik.org/features/*

[12] Premium plugins: *https://piwik.org/blog/2016/11/premium-plugins-now-available-marketplace/*

[13] Piwik demo: *http://demo.piwik.org*

[14] Piwik hosting: *http://piwik.org/hosting/*

[15] Event queuing: *http://www.openwebanalytics.com/?cat=9&paged=3*

[16] Queued tracking: *https://plugins.piwik.org/QueuedTracking*

[17] eAnalytics on SourceForge: *https://sourceforge.net/projects/eanalytics/*

[18] eAnalytics Tag: *http://eanalytics.de/resources/download/eanalytics-download.html*

[19] Tagging Guide: *http://eanalytics.de/uploads/media/eAnalytics_Page_Tagging_Guide_english_V1_7.pdf*

[20] VMware image: *http://eanalytics.de/resources/download/eanalytics-download.html*

Create mobile apps with Qt Creator

Easy Apps

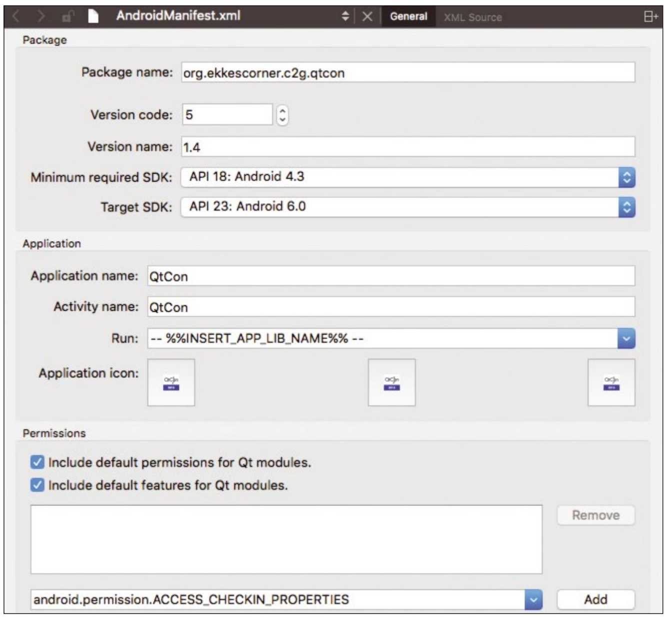
Find out how to create a cross-platform app with the veteran tool Qt Creator. By Ekke Gentz

Qt has been around for more than 20 years. During this time, developers have used several approaches to develop graphical interfaces. More specifically, you might recall Qt Widgets [1], Qt Quick [2], and Qt Quick Controls 2 [3]. Qt beginners who want to build mobile, cross-platform apps might perhaps be confused by this diversity, especially since all the technologies are still in use.

Qt runs on different platforms: Desktop, embedded, and mobile. For desktop applications (OS X, Windows, Linux), developers are more likely to go for Qt Widgets, which offers native controls for the user interface. In contrast, Qt Quick, based on an OpenGL implementation, seeks to abstract the UI controls to deploy Qt across platforms. Mobile apps developed with Qt Quick do

TABLE 1: Qt Quick Controls 2

Containers	Others
`ApplicationWindow` (top-level control)	`BusyIndicator`
`ButtonGroup`	`ComboBox`
`Container` (similar to the `Cascades` container)	`Dial`
`Frame`	`Label`
`GroupBox`	`PageIndicator`
`Page` (an area with a header and footer)	`ProgressBar`
`Pane` (area optimized for selected styles/ themes with the correct background)	`RangeSlider`
Buttons	`ScrollBar`
`Button`	`Slider`
`CheckBox`	`SpinBox`
`RadioButton`	`TextArea`
`Switch`	`TextField`
`TabButton`	`Tumbler`
`ToolButton`	**Navigation**
Pop-Ups	`Drawer`
`Menu`	`StackView`
`Popup` (toasts, dialogs)	`SwipeView`
`ToolTip`	`TabBar`
	`ToolBar`



Package
Package name: org.ekkescorner.c2g.qtcon
Version code: 5
Version name: 1.4
Minimum required SDK: API 18: Android 4.3
Target SDK: API 23: Android 6.0

Application
Application name: QtCon
Activity name: QtCon
Run: -- %%INSERT_APP_LIB_NAME%% --
Application icon:

Permissions
☑ Include default permissions for Qt modules.
☑ Include default features for Qt modules.
android.permission.ACCESS_CHECKIN_PROPERTIES

Figure 1: The Android Manifest Editor in Qt Creator.

Lead Image © Denys Rudyi, 123RF.com

not look native, nor is their performance very convincing. This changed, however, with Qt 5.7, which introduced the lightweight Qt Quick Controls 2. The name is a bit deceptive, because it is not a new version of Qt Quick, but a completely new development.

Mobile Controls

Qt Quick Controls 2 comes with many additional controls and with new containers that simplify development. Navigation elements, now standard in mobile applications, are also on board. Table 1 shows an overview of the elements provided by Qt Quick Controls 2.

Currently, three styles of control elements are available: Google Material, Windows Universal, and Default – a special iOS style is due to follow. Qt also has a dark and a bright theme.

LISTING 1: Floating action button (FAB)

```
Button {
    id: button
    // image should be 24x24
    property alias imageSource: contentImage.source
    // default: primaryColor
    property alias backgroundColor: buttonBackground.color
    property bool showShadow: true
    contentItem:
        Item {
        implicitHeight: 24
        implicitWidth: 24
        Image {
            id: contentImage
            anchors.centerIn: parent
        }
    }
    background:
        Rectangle {
        id: buttonBackground
        implicitWidth: 56
        implicitHeight: 56
        color: primaryColor
        radius: width / 2
        opacity: button.pressed ? 0.75 : 1.0
        layer.enabled: button.showShadow
        layer.effect: DropShadow {
            verticalOffset: 3
            horizontalOffset: 1
            color: dropShadow
            samples: button.pressed ? 20 : 10
            spread: 0.5
        }
    }
}
```

LISTING 2: Calling the FAB

```
FloatingActionButton {
    imageSource: "qrc:/images/"+iconOnPrimaryLightFolder+"/person.png"
    backgroundColor: primaryLightColor
}
FloatingActionButton {
    imageSource: "qrc:/images/"+iconOnPrimaryFolder+"/person.png"
}
```

Producers

If you have worked exclusively with Eclipse for more than 10 years, the switch to Qt Creator [4] might not be easy. However, the software is well-suited to building mobile apps. It helps you program the business logic with C++ and create the interface with QML, thus supporting the deployment process on platforms such as Android, iOS, and Windows 10. Developers can test mobile apps on simulators or real devices from within Qt Creator and build releases for the app stores.

Qt Creator supports the complete Android build process with support for an Android Manifest Editor (Figure 1), signatures, and icons. This also works without Android Studio; however, devel-

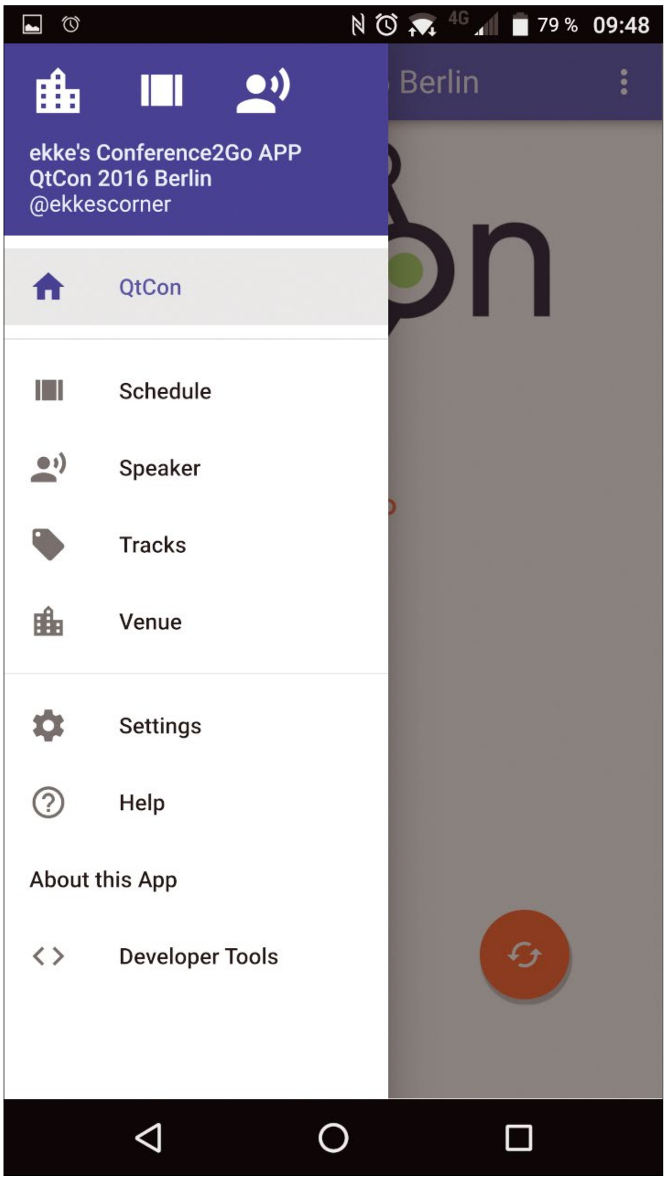


Figure 2: The `Drawer` control at top left allows access to all areas of the program.

LISTING 3: Save a list as a JSON array

```
void DataManager::writeToCache(const QString& fileName, QVariantList& data)
{
    QString cacheFilePath = dataPath(fileName);
    QJsonDocument jda = QJsonDocument::fromVariant(data);
    QFile saveFile(cacheFilePath);
    saveFile.write(jda.toJson(QJsonDocument::Compact:QJsonDocumen));
}
```

opers do need to install an SDK and the NDK up front.

An Xcode project for iOS is created by calling *Project* | *Archive* and then loading the app into the store, but if you want to use Xcode, you need an Apple computer. The path to the Windows App Store is through a Visual Studio project.

One for All

The beauty of Qt is that it serves different platforms with the same source code, and it even has interfaces to supplement native code for Android and iOS.

For my QtCon Conference App [5], I wanted to find out what is possible with a plain vanilla Qt version. The result is available on GitHub [6], and if you are interested, you will also find it in the Google Play Store or the Apple App Store.

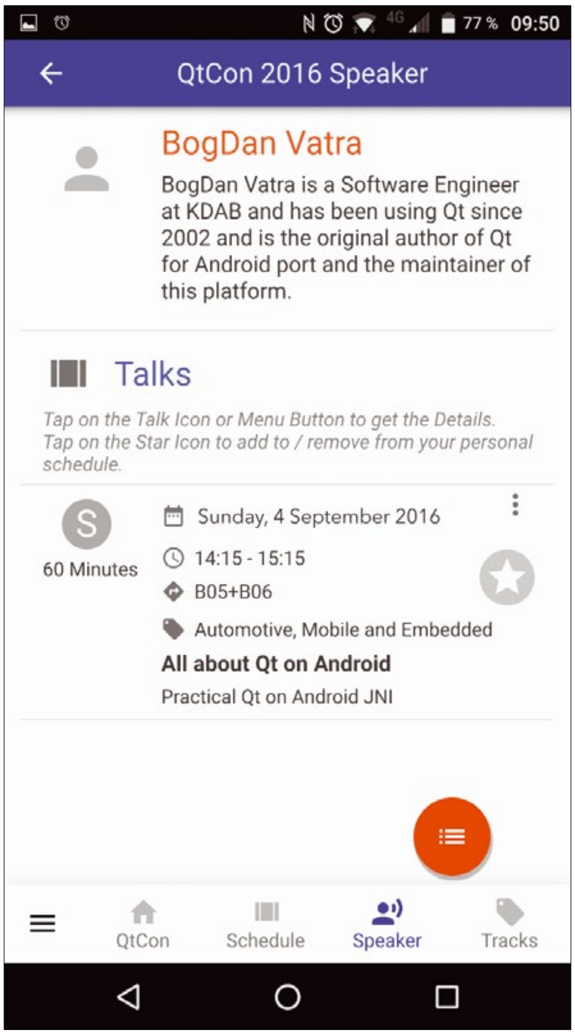
Extensible

The UI controls mentioned above were useful as a solid base that I could easily customize. All I was missing was a date and time picker in the style of Google Material. I programmed that within a few hours, and it is now available to the general public via the GitHub repository. I also found buttons, but not floating action buttons (FABs), which modern apps that use a Google Material style would need. Listing 1 shows the code to make a FAB from a button. Calling a FAB in the main program is then quite simple. Listing 2 shows you how.

Navigation, Data, and Structure

The major navigation elements are all used in the conference app. The `Drawer` navigation control with the hamburger menu (Figure 2) allows access to all areas of the program. A navigation bar at the bottom helps a user reach the main targets quickly.

The conference schedule uses the `SwipeView` and `TabBar` elements, which users can use to switch between the



QtCon 2016 Speaker

BogDan Vatra

BogDan Vatra is a Software Engineer at KDAB and has been using Qt since 2002 and is the original author of Qt for Android port and the maintainer of this platform.

Talks

Tap on the Talk Icon or Menu Button to get the Details.
Tap on the Star Icon to add to / remove from your personal schedule.

S
60 Minutes
Sunday, 4 September 2016
14:15 - 15:15
B05+B06
Automotive, Mobile and Embedded
All about Qt on Android
Practical Qt on Android JNI

Figure 3: The contextual details can be designed as `Pages` with a `StackView`.

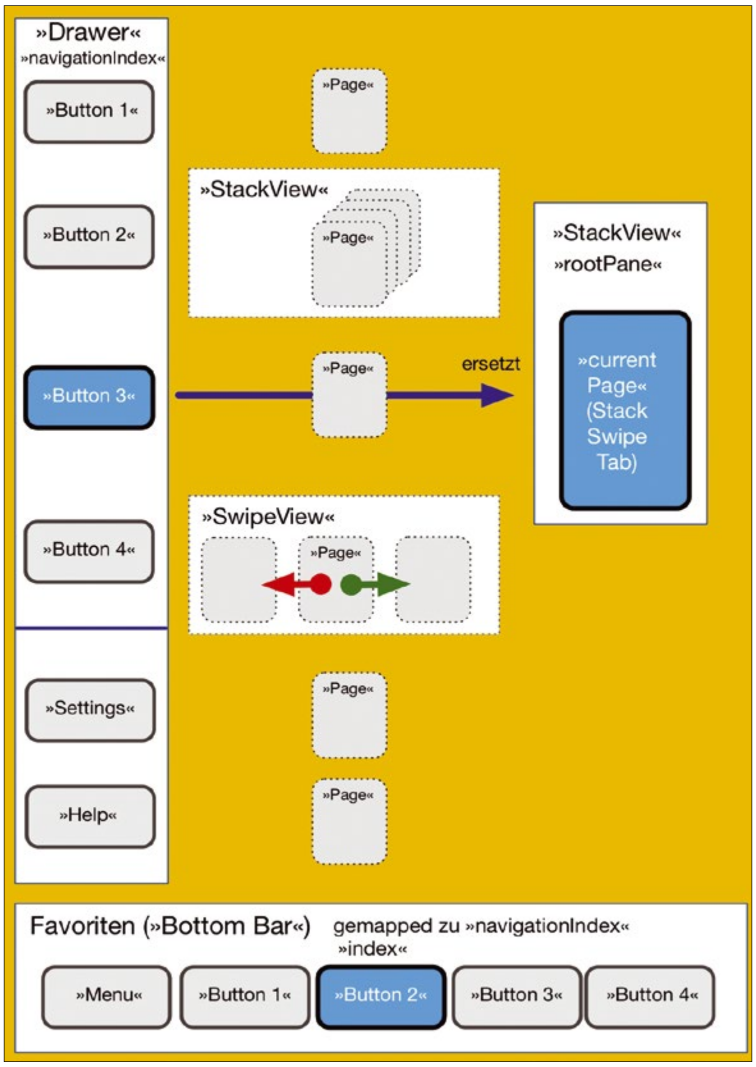


Figure 4: The complete app structure with Qt 5.7.

days of the conference using swipe gestures or tabs. The `StackView` element lets users climb up through the different levels and view contextual details (Figure 3) relating to, for example, sessions, details, speakers, talks, and venues.

Developers can easily combine all these navigation elements; thus, even very complex apps are no problem for Qt. Because components and loaders dynamically generate the controls and pages, memory usage remains manageable, and the app remains fast as a whole.

The application pushes data into the filesystem in the usual way in the form of JSON files or stores them permanently in a SQLite database. Listing 3 contains the code for saving a list as a JSON array, and Figure 4 shows a typical application structure.

Conclusion

Following the release of version 5.7 and thanks to Qt Quick Controls 2, Qt is now an excellent choice for developing visually appealing, high-performance mobile apps. The advantage is that you need only one codebase for all platforms.

It can be somewhat difficult to find the appropriate pages in the documentation, because Qt is not just available for mobile devices, but also for desktop and embedded devices. The blog [7] gives developers a little assistance in this matter.

Qt is open source, but also available with a commercial license. Independent developers and start-ups need a license, which you can secure from $79 (EUR72) per month, because, unfortunately, the LGPL is not suitable for apps in Apple's App Store. If you have any questions about the licenses, your best bet is to watch the video [8] that discusses the legal situation in more detail. ■■■

AUTHOR

Ekke Gentz is an independent software architect and has been developing software for almost 40 years. In the past few years, he has focused on mobile apps for business applications.

INFO

[1] Qt Widgets: *https://doc.qt.io/qt-5/qtwidgets-index.html*
[2] Qt Quick: *https://www.qt.io/qt-quick/*
[3] Qt Quick Controls 2: *http://doc.qt.io/qt-5/qtquickcontrols2-index.html*
[4] Qt Creator: *https://www.qt.io/ide/*
[5] QtCon website: *https://qtcon.org*
[6] QtCon Conference App on Github: *https://github.com/ekke/c2gQtCon_x*
[7] Blog on mobile apps with Qt: *http://j.mp/qt-x*
[8] Qt video on licensing issues: *https://www.youtube.com/watch?v=lSYDWnsfWUk*

Improve the way you work with Secure Shell

Safety First!

Many Linux users employ Secure Shell to log in remotely and work as if on a local machine. But SSH can do even more – the application will send commands, route other TCP connections through an encrypted tunnel, and provide multiplexing support. *By Heike Jurzik*

The man pages for the OpenSSH tool [1] read like novels. Most users know only a fraction of the options, and this article, too, is restricted to just a few features. It provides tips on SSH multiplexing, tunneling, and various configuration files.

On My Mark!

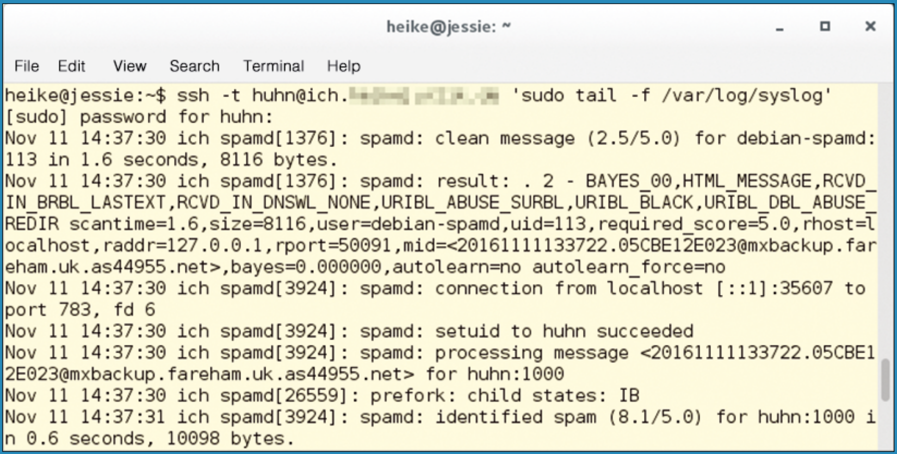
One of the oldest SSH tricks is to define a command that SSH runs at the opposite end when executed; this saves you time and typing. The output appears in the local terminal:

```
<heike@home:~$> ssh huhn@example.com pwd
/home/huhn
```

The command displays the working directory for the user `huhn` after logging onto the `example.com` machine; in other words, the home directory. Multiple commands are also possible. It is important to quote the commands to make sure they really run on the remote system:

```
heike@home:~$ ↩
  ssh huhn@example.com pwd;whoami
/home/huhn
heike
heike@home:~$ ↩
  ssh huhn@example.com 'pwd; whoami'
/home/huhn
huhn
```

If you want SSH to run commands that require user interaction on the other machine, the `t` option is also necessary to start a pseudo terminal. This ensures that users remain logged in until the program is finished. In this way, you can, say, edit files with a text editor, use `top` to list the active programs list, or view log files with `tail -f` (Figure 1).



```
heike@jessie:~$ ssh -t huhn@ich.[redacted] 'sudo tail -f /var/log/syslog'
[sudo] password for huhn:
Nov 11 14:37:30 ich spamd[1376]: spamd: clean message (2.5/5.0) for debian-spamd:
113 in 1.6 seconds, 8116 bytes.
Nov 11 14:37:30 ich spamd[1376]: spamd: result: . 2 - BAYES_00,HTML_MESSAGE,RCVD_
IN_BRBL_LASTEXT,RCVD_IN_DNSWL_NONE,URIBL_ABUSE_SURBL,URIBL_BLACK,URIBL_DBL_ABUSE_
REDIR scantime=1.6,size=8116,user=debian-spamd,uid=113,required_score=5.0,rhost=l
ocalhost,raddr=127.0.0.1,rport=50091,mid=<20161111133722.05CBE12E023@mxbackup.far
eham.uk.as44955.net>,bayes=0.000000,autolearn=no autolearn_force=no
Nov 11 14:37:30 ich spamd[3924]: spamd: connection from localhost [::1]:35607 to
port 783, fd 6
Nov 11 14:37:30 ich spamd[3924]: spamd: setuid to huhn succeeded
Nov 11 14:37:30 ich spamd[3924]: spamd: processing message <20161111133722.05CBE1
2E023@mxbackup.fareham.uk.as44955.net> for huhn:1000
Nov 11 14:37:30 ich spamd[26559]: prefork: child states: IB
Nov 11 14:37:31 ich spamd[3924]: spamd: identified spam (8.1/5.0) for huhn:1000 i
n 0.6 seconds, 10098 bytes.
```

Figure 1: The SSH `-t` switch allows user interaction at the other end.

Well Prepared

If you juggle several accounts on remote servers that listen on different ports, or each use a different SSH key pair for authentication, you can specify the options each time you run SSH. However, you may find a separate `~/.ssh/config` configuration file (Listing 1) more convenient.

Each line contains a keyword; this is followed by one or more arguments. In addition to options that apply only to one computer (sections following `host`

LISTING 1: Setup file ~/.ssh/config

```
Host work
        HostName 123.456.78.9
        Port 2220
        User <User>
        IdentityFile ~/.ssh/id_rsa
Host home
        HostName home.example.com
        User <User2>
        CheckHostIP no
Host *
        ServerAliveInterval 300
        LogLevel VERBOSE
        Compression yes
```

name), global settings (`host *`) are also possible. Because SSH evaluates the information in sequence, the general settings are available at the end of the file. The man page lists all the options for `ssh_config`.

The example in Listing 1 contains two blocks for individual hosts. For the first computer (accessible via the alias `ssh work`), it defines that the SSH server listens on port 2220. Also, the IP address is entered instead of a hostname, and the key to be used is defined, which removes the need to try out all the keys – this speeds up the connection if several key pairs are located in the `~/.ssh` directory.

The following applies for the second computer `home.example.com`: It is accessible using the `home` short name. `CheckHostIP no` prevents matching the IP address against the `~/.ssh/known_hosts` file, which is useful for hosts with a changing, dynamically allocated IP.

In the general instructions, `ServerAliveInterval 300` makes sure that the client sends a packet to the server after five minutes, to keep the connection alive and prevent a timeout. Additionally, the behavior of `log` is defined; users can choose from `QUIET`, `FATAL`, `ERROR`, `INFO` (standard), `VERBOSE`, `Debug`, `DEBUG1`, `DEBUG2`, and `DEBUG3`.

Data compression is turned on in this case, but this is only useful for hosts where the users are faced with a slow connection.

Bundled, We Are Strong

The multiplexing feature, which can send several signals via a TCP connection ensures a speed advantage. The SSH client uses an existing connection to the SSH server, which eliminates the handshake with the opposite end. You do not save time with the first SSH session, but from the second. Users control multiplexing with two options: `ControlMaster` and `ControlPath`:

```
ssh -o ControlMaster=yes ↩
  -o ControlPath=~/.ssh/↩
  control-%h_%p_%r user@example.org
```

The statement `ControlMaster = yes` generates a master link, which other sessions with the same destination can use. `ControlPath` defines the required socket, composed of the file name `control`, the hostname (`%h`), the port (`%p`), and user name (`%r`) of the target computer. For the next connection to this host, users state the socket; `ControlMaster` is left at the default value (`no`):

```
ssh -o ControlPath=~/.ssh/↩
  control-%h_%p_%r user@example.org
```

The second link attempts to dock onto the first. Should that fail, the SSH client establishes a normal connection as a fallback solution. Listing 2 shows the comparison with the `time` for the `who` command measured on the remote machine with and without multiplexing. In a small test setup, the difference is not particularly impressive, but in larger environments the feature should have a positive effect.

LISTING 2: SSH without and with multiplexing

```
$ time ssh huhn@example.com who
[...]
real    0m0.288s
user    0m0.008s
sys     0m0.012s
[Multiplexing enabled]
$ time ssh huhn@example.com who
[...]
real    0m0.037s
user    0m0.000s
sys     0m0.004s
```

The multiplexing configuration does not need to be enabled with `-o`; you can also do this in the SSH setup file. Besides the instructions, `ControlMaster` and `ControlPath` users can optionally define `ControlPersist` here; this determines how long the master connection waits for clients in the background after the first connection is done. Users can enter either `no` (master does not wait), `yes` (master waits forever) or a period of time in seconds. Listing 3 gives you an example.

By the way, it is not only SSH clients that benefit from the multiplexing feature, but also `scp`, `rsync`, `git`, and other commands that use the SSH protocol.

LISTING 3: Multiplexing settings in ~/.ssh/config

```
Host example
        HostName example.com
        ControlMaster auto
        ControlPath ~/.ssh/control-%h_%p_%r
        ControlPersist yes
```

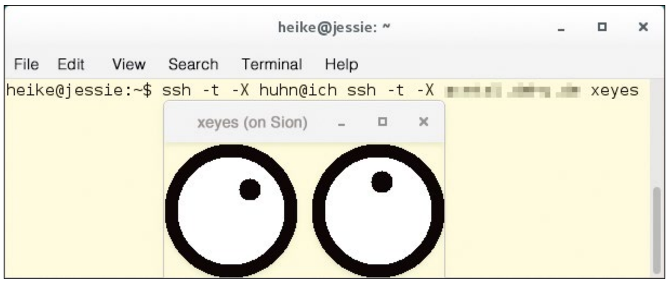
X, Please!

SSH can transfer graphical applications that are running on a remote computer to your own desktop. Assuming the SSH server on the other side allows X-forwarding (`X11Forwarding yes` in `/etc/ssh/sshd_config`) and the `xauth` package is installed, users can launch X11 programs on the remote machine and control them locally as a one off using `ssh -X` or permanently with an entry in the configuration file (`ForwardX11 yes`).

The whole thing becomes interesting when several computers are involved. Let's assume that `host1` is allowed to access `host2`, and `host2` can access `host3`, but a direct connection between `host1` and `host3` is not possible. Instead of multiple commands, users can type a single command (Figure 2):

```
ssh -t -X host2 ssh -t -X host3 xeyes &
```

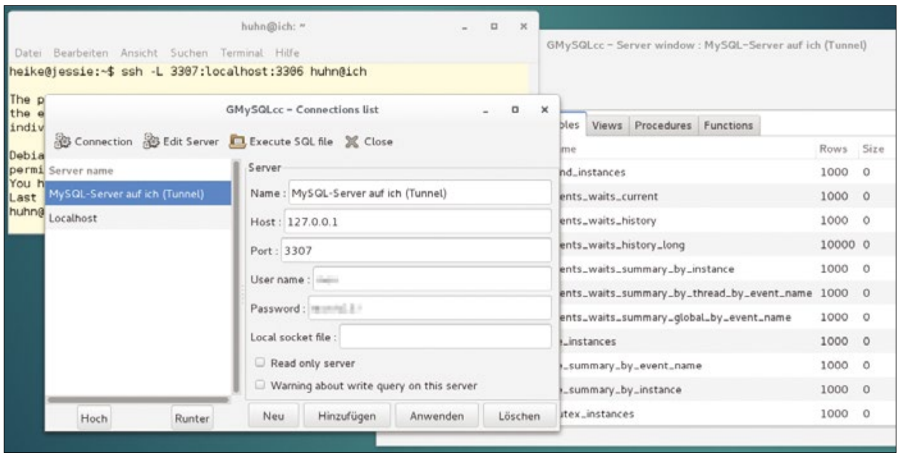
This is where you use the `-t` option (see the "On My Mark!" section). If an alias is defined in the configuration file, you can define the option `ForwardX11 yes` for `host2` and `host3`. The third computer is also given instructions for the `ProxyCommand` (Listing 4).



```
heike@jessie:~$ ssh -t -X huhn@ich ssh -t -X [redacted] xeyes
```

Figure 2: The `xeyes` program runs on the third host named Sion; however, the display appears in the graphical interface of the first computer (jessie).

Secure Shell routes other TCP connections through a secure tunnel, if desired. Much like a proxy, SSH accepts the connection on a port on one side and links it to a port on the remote side.


huhn@ich: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
GMySQLcc – Server window : MySQL-Server auf ich (Tunnel)
GMySQLcc – Connections list
Connection
Edit Server
Execute SQL file
Close
Server name
MySQL-Server auf ich (Tunnel)
Localhost
Hoch
Runter
Neu
Hinzufügen
Anwenden
Löschen
Views
Procedures
Functions

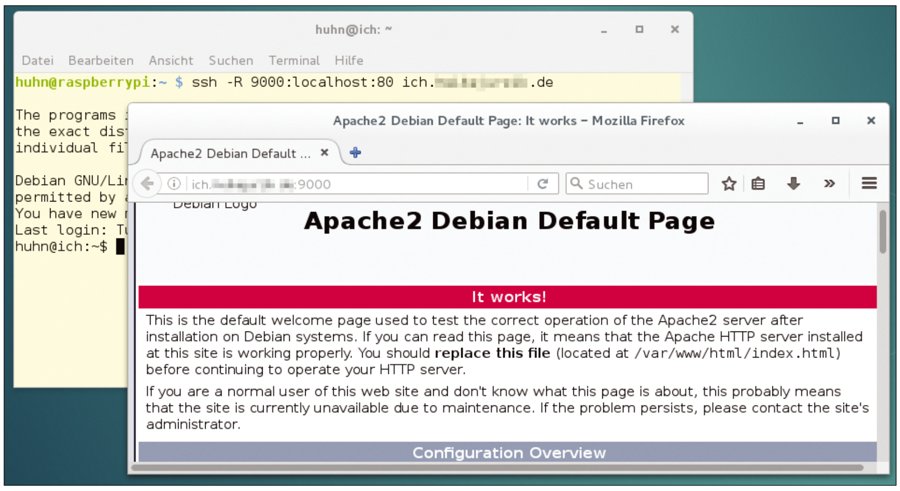
```
heike@jessie:~$ ssh -L 3307:localhost:3306 huhn@ich
```

Server

Name : MySQL-Server auf ich (Tunnel)
Host : 127.0.0.1
Port : 3307
User name : [redacted]
Password : [redacted]
Local socket file :
☐ Read only server
☐ Warning about write query on this server

me	Rows	Size
nd_instances	1000	0
ents_waits_current	1000	0
ents_waits_history	1000	0
ents_waits_history_long	10000	0
ents_waits_summary_by_instance	1000	0
ents_waits_summary_by_thread_by_event_name	1000	0
ents_waits_summary_global_by_event_name	1000	0
_instances	1000	0
_summary_by_event_name	1000	0
_summary_by_instance	1000	0
tex_instances	1000	0

Figure 3: In the graphical MySQL client, users define 127.0.0.1 as the address of the remote MySQL server and the port number specified in the tunnel (3307 here).


huhn@ich: ~
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
Apache2 Debian Default Page: It works – Mozilla Firefox
Apache2 Debian Default ...
ich.[redacted]:9000
Suchen
Debian Logo

```
huhn@raspberrypi:~ $ ssh -R 9000:localhost:80 ich.[redacted].de

The programs i
the exact dis
individual fi

Debian GNU/Li
permitted by
You have new
Last login: T
huhn@ich:~$
```

Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Figure 4: A local web server is now accessible from the outside on port 9000 through an SSH tunnel.

LISTING 4: X11 forwarding for multiple hosts in ~/.ssh/config

```
Host host2
        HostName host2.example.com
        User <User>
        IdentityFile ~/.ssh/id_rsa
        ForwardX11 yes
Host host3
        HostName host3.example.com
        User <User>
        ForwardX11 yes
        ProxyCommand ssh host2 -W %h:%p
```

Into the Pipe

Port forwarding works in both directions. Local port forwarding is enabled with the `-L` option and forwards a connection that arrives on an arbitrary local port through the SSH tunnel to a port on a remote server:

```
ssh -L <Local_port>:localhost:↩
  <Remote_port> <User>@<Remote_host>
```

Users can use such a tunnel to lift restrictions at the opposite end and, for example, reach a remote MySQL server that only listens to local connections (`bind-address = 127.0.0.1`) from the local computer (Figure 3).

Tunneling is also possible in the opposite direction: In remote port forwarding, the connection arrives at a port on the remote computer, and the data are routed through the SSH tunnel to any port on the client side. Users can take advantage of the feature to share a service offered by a computer on the local network with others outside the LAN.

Instead of `-L`, the option used is `-R`.

```
ssh -R <Remote_port>:localhost:↩
  <Local_port> <User>@<Remote_host>
```

To make this happen, the `GatewayPorts yes` option must be set on the SSH server at the other end so that the remote computers can connect to local, forwarded ports. Figure 4 shows an example in which the web server (port 80), a Raspberry Pi (which is available only on the local network) is connected via an SSH tunnel to port 9000 on a remote server. The remote server has a public IP address, so the web server using port 9000 is now also reachable from the outside.

If you want to define the settings for one or more SSH tunnels in the configuration file; Listing 5 shows an example. ■■■

INFO

[1] Man pages for OpenSSH: *https://www.openssh.com/manual.html*

LISTING 5: Local/remote port forwarding in ~/.ssh/config

```
Host host1
        HostName host1.example.com
        LocalForward 3307 localhost:3306
Host host2
        HostName host2.example.com
        RemoteForward 9000 localhost:80
```

The sys admin's daily grind: Airsensor

Exactly How Smelly?

If the air in a room smells stale, somebody will get up and open the window to let in some fresh air. Charly, however, wanted to measure its exact staleness, so he set off to find out armed with USB hardware and Linux. *By Charly Kühnast*

Stale air results from the accumulation of various gases in an enclosed room. Carbon dioxide is the classic case – just imagine 16 people in a small, windowless conference room. Add to this, a mixture of volatile organic compounds, known to the world of science as VOCs. Among other things VOCs are alcohols, volatile deodorant components, aldehydes from furniture, detergent fumes, nicotine, briefly: carbon-based emissions of all kinds.

You could use human noses to sniff VOCs, but sensors in the form of a USB stick are not only more suitable but also immune to attacks of nausea. The Rehau sniffer [1], for example, uses an LED with a color changer to indicate the state of the air in the room.

Pure Air

A tool helps me to read the measured values from the stick. Before you build the software [2], you first need to install the *libusb-dev* and *build-essential* packages. After unpacking the zip file to `/usr/local/`, the change directory to `usb-sensors-linux/trunk/airsensor/`. When you get there, do this:

```
gcc -o airsensor airsensor.c -lusb
```

which will compile the small C program, creating an `airsensor` binary. When launched, the sensor gives you a new air reading approximately every 10 seconds:

```
2016-12-09 11:27:26, VOC: 557, RESULT: OK
2016-12-09 11:27:37, VOC: 540, RESULT: OK
2016-12-09 11:27:48 AM, VOC: 542, RESULT: OK
2016-12-09 11:27:59 AM, VOC: 563, RESULT: OK
2016-12-09 11:28:11 AM, VOC: 515, RESULT: OK
2016-12-09 11:28:22 AM, VOC: 505, RESULT: OK
```

The unit of measurement for the VOC concentration is parts per million (ppm); my sensor is specified for values between 450 and 2000 ppm. It delivers even higher values, but an algorithm has probably interpolated these. For values below 1000 ppm, the built-in LED lights up green, yellow between 1001 and 1500 ppm, and red above that.

Using the `-o` parameter, I told the software to output a measured value and then terminate. That makes it easy to write the values to a database as datapoints and draw history graphs. Figure 1 shows what happens when you cook ham and eggs for breakfast. ■■■

INFO

[1] Rehau room air sensor: *https://www.amazon.de/gp/product/B00ZXP6EI4/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1*

[2] Airsensor sources: *https://storage.googleapis.com/google-code-archive-source/v2/code.google.com/usb-sensors-linux/source-archive.zip*

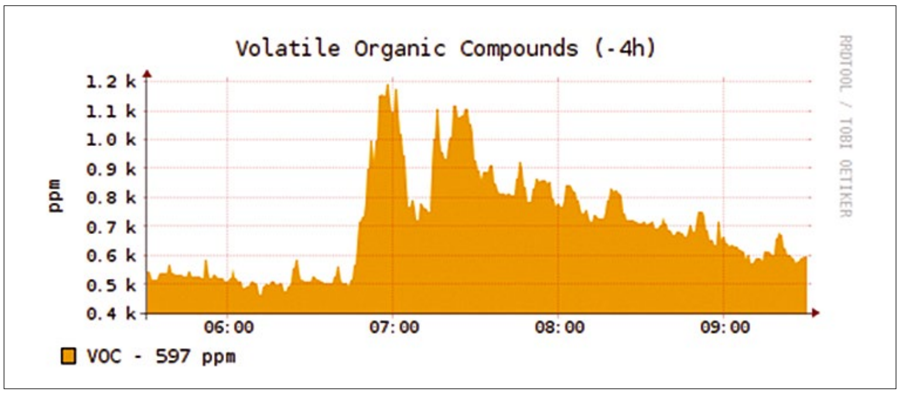


Figure 1: The event that led to the dramatic deterioration of Charly's indoor climate at 6:45 a.m. will be revealed at the end of the article.

CHARLY KÜHNAST

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.

Configuring Bluetooth devices with bluetoothctl

Blue Control

The bluetoothctl command will keep your Bluetooth devices talking to Linux.

By Bruce Byfield

`bluetoothctl` [1] is the main command for configuring Bluetooth devices on Linux. Contrary to what the name's structure might lead you to expect, `bluetoothctl` is not part of systemd, but rather a simple set of options for setting up Bluetooth devices.

As you probably know, the Bluetooth standard is a collection of protocols for exchanging data over short distances – typically, less than five meters. Named for the 10th-century king Harald Bluetooth, who unified Denmark under the banner of Christianity, Bluetooth was developed by the Swedish company Ericsson, largely under the direction of Sven Mattisson. It was first released in 1998 and is now developed by the Bluetooth Special Interest Group [2]. Since 2005, it has been fully supported by Linux, and it has been included in the kernel since 2001. Today, the Linux implementation of Bluetooth is maintained by the BlueZ project[3]. `bluetoothctl` is part of the *bluez-utilities* package.

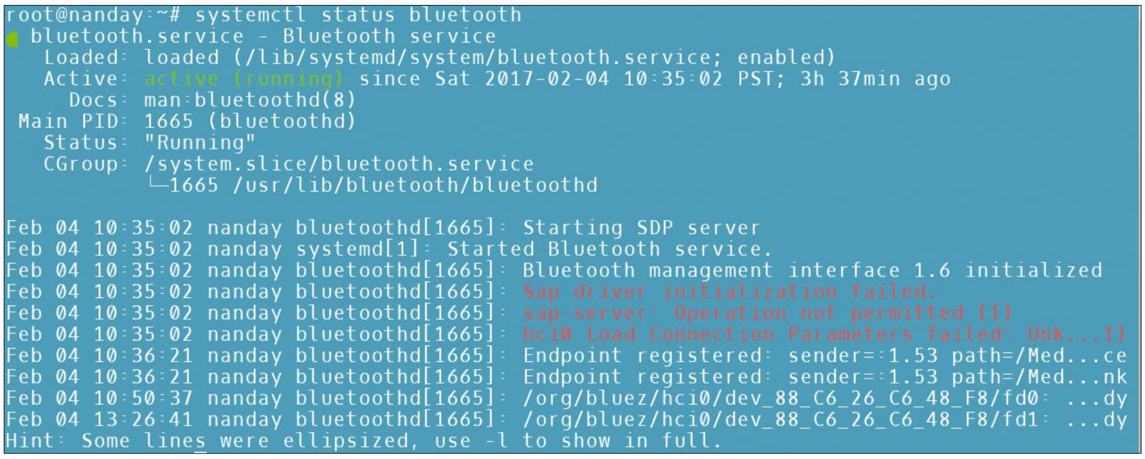
Bluetooth took several years to establish itself, but today, it can be used to add to your system almost every type of hardware, including speakers, printers, hard drives, keyboards, and mouses. The rise of the Internet of Things (IoT) has given the standard an enormous boost, because not only is Bluetooth's short range a security feature in itself, but the latest versions of the standard – version 5.0 as of mid-2016, although most available hardware uses version 4.0-4.2 – have exactly the built-in security features that companies are starting to realize are required by the IoT. Bluetooth smart light bulbs are already available, and more Bluetooth IoT devices are scheduled to follow.

Acceptance of Bluetooth is also helped by the fact that it is a mature standard whose user interface has become simpler over the years. In particular, `bluetoothctl` does not require users to concern themselves about which Bluetooth profile [4] (e.g., A2DP for headsets) devices need to communicate. Similarly, users no longer need to enter encryption PINs, which the latest versions of `bluetoothctl` now handle automatically unless an older device is involved. For that matter, `bluetoothctl` itself is in many ways an updated version of `hcitool` [5], the configuration tool that was used at the command line until a couple of years ago.

Preparing for Configuration

Most modern computers support Bluetooth by default. However, older or custom-compiled ones may not. Although you can add kernel modules to a system [6], the quickest way to add support is a USB Bluetooth dongle. Few dongles will actually advertise Linux support, but, as USB devices, almost any should work after you reboot the system.

If you are unsure whether a system supports Bluetooth, try installing the necessary packages – *bluetooth* or *bluez* on Debian, or `bluetoothctl` on Debian. Depending on the type of device, you may also need to install other packages. For example, for a speaker, you need to



```
root@nanday:~# systemctl status bluetooth
● bluetooth.service - Bluetooth service
   Loaded: loaded (/lib/systemd/system/bluetooth.service; enabled)
   Active: active (running) since Sat 2017-02-04 10:35:02 PST; 3h 37min ago
     Docs: man:bluetoothd(8)
 Main PID: 1665 (bluetoothd)
   Status: "Running"
   CGroup: /system.slice/bluetooth.service
           └─1665 /usr/lib/bluetooth/bluetoothd

Feb 04 10:35:02 nanday bluetoothd[1665]: Starting SDP server
Feb 04 10:35:02 nanday systemd[1]: Started Bluetooth service.
Feb 04 10:35:02 nanday bluetoothd[1665]: Bluetooth management interface 1.6 initialized
Feb 04 10:35:02 nanday bluetoothd[1665]: Sap driver initialization failed.
Feb 04 10:35:02 nanday bluetoothd[1665]: sap-server: Operation not permitted (1)
Feb 04 10:35:02 nanday bluetoothd[1665]: hci0 Load Connection Parameters failed: Unk...1)
Feb 04 10:36:21 nanday bluetoothd[1665]: Endpoint registered: sender=:1.53 path=/Med...ce
Feb 04 10:36:21 nanday bluetoothd[1665]: Endpoint registered: sender=:1.53 path=/Med...nk
Feb 04 10:50:37 nanday bluetoothd[1665]: /org/bluez/hci0/dev_88_C6_26_C6_48_F8/fd0: ...dy
Feb 04 13:26:41 nanday bluetoothd[1665]: /org/bluez/hci0/dev_88_C6_26_C6_48_F8/fd1: ...dy
Hint: Some lines were ellipsized, use -l to show in full.
```

Figure 1: Before pairing, check that Bluetooth control is enabled on your system. If not, then enable it.

install *pulseaudio* and *pavucontrol*, and for a printer, the necessary CUPS package. You can check whether Bluetooth is enabled with the command (Figure 1):

```
systemctl status bluetooth
```

If it is not, then you can start it with:

```
systemctl start bluetooth
```

Your goal is to pair your computer with the device. In other words, you want your computer and the device set to have encrypted communication with each other. Some devices pair automatically when they are turned on, while others like phones provide interfaces or apps. A few, like Logitech's UE Boom 2 speakers, can be readied by pressing physical buttons.

Typically, devices will stay ready to pair for as long as three minutes, after which they have to be readied again. Unfortunately, these methods are not standardized, but they are usually easy enough to use. If you do run into difficulty, many devices will not be shipped with a manual, but most will have a reasonably current online manual.

Configuring Support

`bluetoothctl` is the command for pairing the system with a device. You can use several desktop applications instead (including Bluedevil, Blueman, gnome-bluetooth, and Blueberry), but all except Blueman are specific to a desktop environment. None have any advantage over `bluetoothctl`.

Whereas most commands modify the bare command with options, running `bluetoothctl` starts its own prompt for entering options instead. You can see a list of commands by entering *help* (Figure 2). Above the prompt is a list of systems (which `bluetoothctl` calls controllers) and devices that are available for use.

In its current version, `bluetoothctl` works only with MAC addresses. Both controllers and devices may also have a human-friendly name. For controllers, the name is the computer's name. For devices, the default name is usually the product name.

Many devices allow you to change the name, which can reduce the possibility of confusion if you have two devices of the same kind from the same manufacturer. However, while `bluetoothctl` may list device names, its options do not work with the device names.

Depending on the device, you may be able to use only some of the `bluetoothctl` commands shown in Table 1. More often, you need to include all or some of these commands. Try them in the order listed.

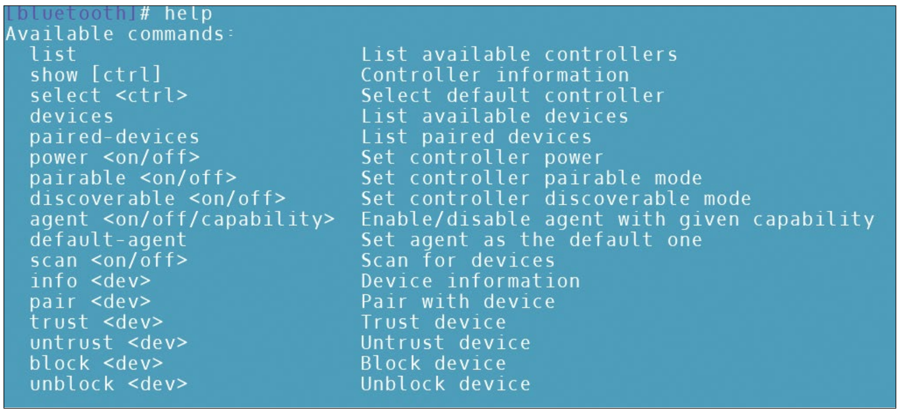
As you might have noticed, many of the commands shown in Table 1 can be used to turn a setting both on and off. In other cases, a separate command is used instead – for example, you can use `untrust` if you no longer want a device to pair immediately. Similarly, you can use `disconnect` to break the connection with a device or `remove` to make a device unavailable for pairing until you scan again. If you want to make a device unavailable on an ongoing basis, use `block` and its counterpart `unblock`.

Whatever commands you use, run `quit` to close the `bluetoothctl` shell and return to Bash. With some devices, you need to do additional setup, such as setting up speakers with *pavucontrol* or KDE's *Phonon*.

Troubleshooting and Fine-Tuning

Bluetooth is far more reliable today than it was as recently as five years ago. Still, it can sometimes behave erratically, so you may have to attempt pairing several times. Scanning with another Bluetooth device, such as a phone, can sometimes tell you if the problem is with the controller or the device.

If problems persist, check that the device is turned on and set to discover other devices. Other possibilities may be that the device is positioned outside the controller's range, or that the device or controller has exceeded the maximum of seven devices supported by Bluetooth.



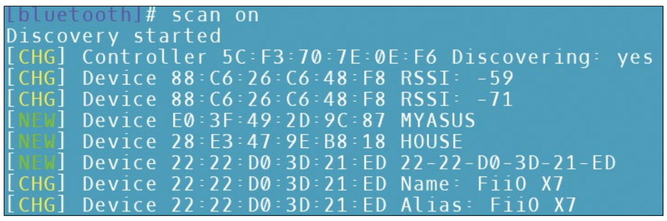
```
[bluetooth]# help
Available commands:
  list                       List available controllers
  show [ctrl]                Controller information
  select <ctrl>              Select default controller
  devices                    List available devices
  paired-devices             List paired devices
  power <on/off>             Set controller power
  pairable <on/off>          Set controller pairable mode
  discoverable <on/off>      Set controller discoverable mode
  agent <on/off/capability>  Enable/disable agent with given capability
  default-agent              Set agent as the default one
  scan <on/off>              Scan for devices
  info <dev>                 Device information
  pair <dev>                 Pair with device
  trust <dev>                Trust device
  untrust <dev>              Untrust device
  block <dev>                Block device
  unblock <dev>              Unblock device
```

Figure 2: `bluetoothctl`'s help shows the available commands.

TABLE 1: bluetoothctl Commands

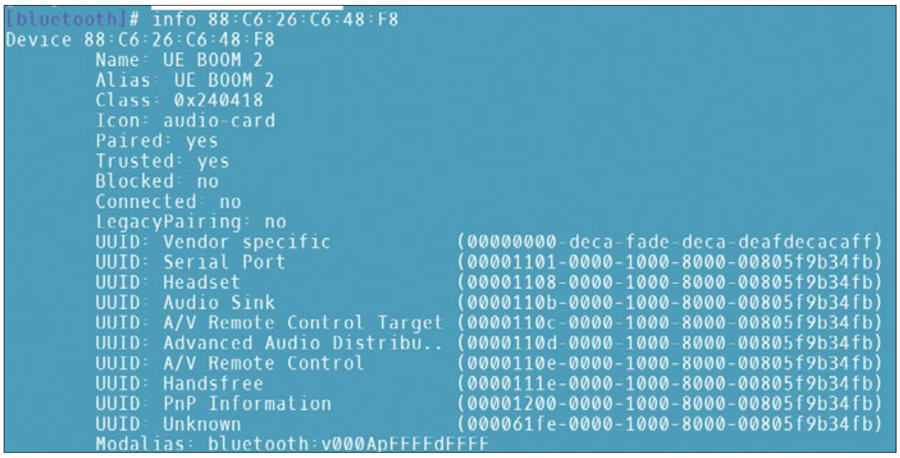
Command	Description
1. `list`	Generates a list of available controllers. You can pair a controller and a device while working from another controller, but on a home system you may have only one controller. However, you need the controller's MAC address to run other commands. If it is no longer visible on screen, run `list` again. You might be able to simplify the process by turning off all controllers except the one to pair.
2. `show`	Gathers information about available controllers, including their names and current state. This information can be useful for distinguishing one controller from the other.
3. `select <CONTROLLER-MAC-ADDRESS>`	Selects the controller to pair. Once you select the controller, all controller-related commands will apply to it for three minutes or until you select a new controller.
4. `power on`	Enables the selected controller. If you are pairing the controller on which `bluetoothctl` is running, this step is unnecessary.
5. `agent on`	Turns on Bluetooth support. If you use a USB adapter, it is on as long as it is plugged in.
6. `default agent`	Sets the current agent to the default.
7. `discoverable on`	Makes the controller visible to other devices. As a security precaution, run `discoverable off` after pairing.
8. `scan on`	Receives a list of detected devices (Figure 3). If a device you expect is not visible, check that it is turned on, ready to pair, and within range. Already paired devices will not be listed.
9. `pairable on`	Readies the controller for pairing. Remember that you have three minutes after running this command to pair.
10. `devices`	Lists available devices. You want the MAC address, not the name, to use with other commands.
11. `info <DEVICE-MAC-ADDRESS>`	Displays information about a particular device (Figure 4). This command is most often useful in identifying the correct device.
12. `connect <DEVICE-MAC-ADDRESS>`	Readies the device for pairing.
13. `pair <DEVICE-MAC-ADDRESS>`	Pairs the device with the default controller.
14. `trust <DEVICE-MAC-ADDRESS>`	Sets the device to re-pair automatically when it is turned on, which eliminates the need to pair all over again.
15. `discoverable off`	Hides the controller from other Bluetooth devices. Otherwise, any device that can detect it has access to it, leaving a major security hole. In this respect, running `bluetoothctl` resembles logging in as root: you want to spend as little time running it as possible.

In addition, for devices like printers, you may want to write a Bash script to autostart as you log in to your desktop environment. Login managers sometimes interfere with Bluetooth starting, so you might want to set the script to delay running until your desktop appears, however long that may be.



```
[bluetooth]# scan on
Discovery started
[CHG] Controller 5C:F3:70:7E:0E:F6 Discovering: yes
[CHG] Device 88:C6:26:C6:48:F8 RSSI: -59
[CHG] Device 88:C6:26:C6:48:F8 RSSI: -71
[NEW] Device E0:3F:49:2D:9C:87 MYASUS
[NEW] Device 28:E3:47:9E:B8:18 HOUSE
[NEW] Device 22:22:D0:3D:21:ED 22-22-D0-3D-21-ED
[CHG] Device 22:22:D0:3D:21:ED Name: FiiO X7
[CHG] Device 22:22:D0:3D:21:ED Alias: FiiO X7
```

Figure 3: Choose the device to pair from the list of detected devices.



```
[bluetooth]# info 88:C6:26:C6:48:F8
Device 88:C6:26:C6:48:F8
        Name: UE BOOM 2
        Alias: UE BOOM 2
        Class: 0x240418
        Icon: audio-card
        Paired: yes
        Trusted: yes
        Blocked: no
        Connected: no
        LegacyPairing: no
        UUID: Vendor specific           (00000000-deca-fade-deca-deafdecacaff)
        UUID: Serial Port               (00001101-0000-1000-8000-00805f9b34fb)
        UUID: Headset                   (00001108-0000-1000-8000-00805f9b34fb)
        UUID: Audio Sink                (0000110b-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: Advanced Audio Distribu.. (0000110d-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: Handsfree                 (0000111e-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: Unknown                   (000061fe-0000-1000-8000-00805f9b34fb)
        Modalias: bluetooth:v000ApFFFFdFFFF
```

Figure 4: Checking the information about a device can help you with identifying it and with troubleshooting.

One final tip: Some recent Bluetooth devices, such as the Fiio X7 portable stereo, can be set to turn off after a few minutes to preserve battery power. If the device turning off unpairs it, run `trust` from the `bluetoothctl` shell to have it reconnect automatically when turned back on.

Today, however, such problems are becoming rare. After all the years it has been available, Bluetooth is finally coming into its own – and `bluetoothctl` is the key to running it without difficulties on Linux. ■■■

INFO

[1] bluetoothctl: *http://www.bluez.org*

[2] Bluetooth Special Interest Group: *https://www.bluetooth.com/*

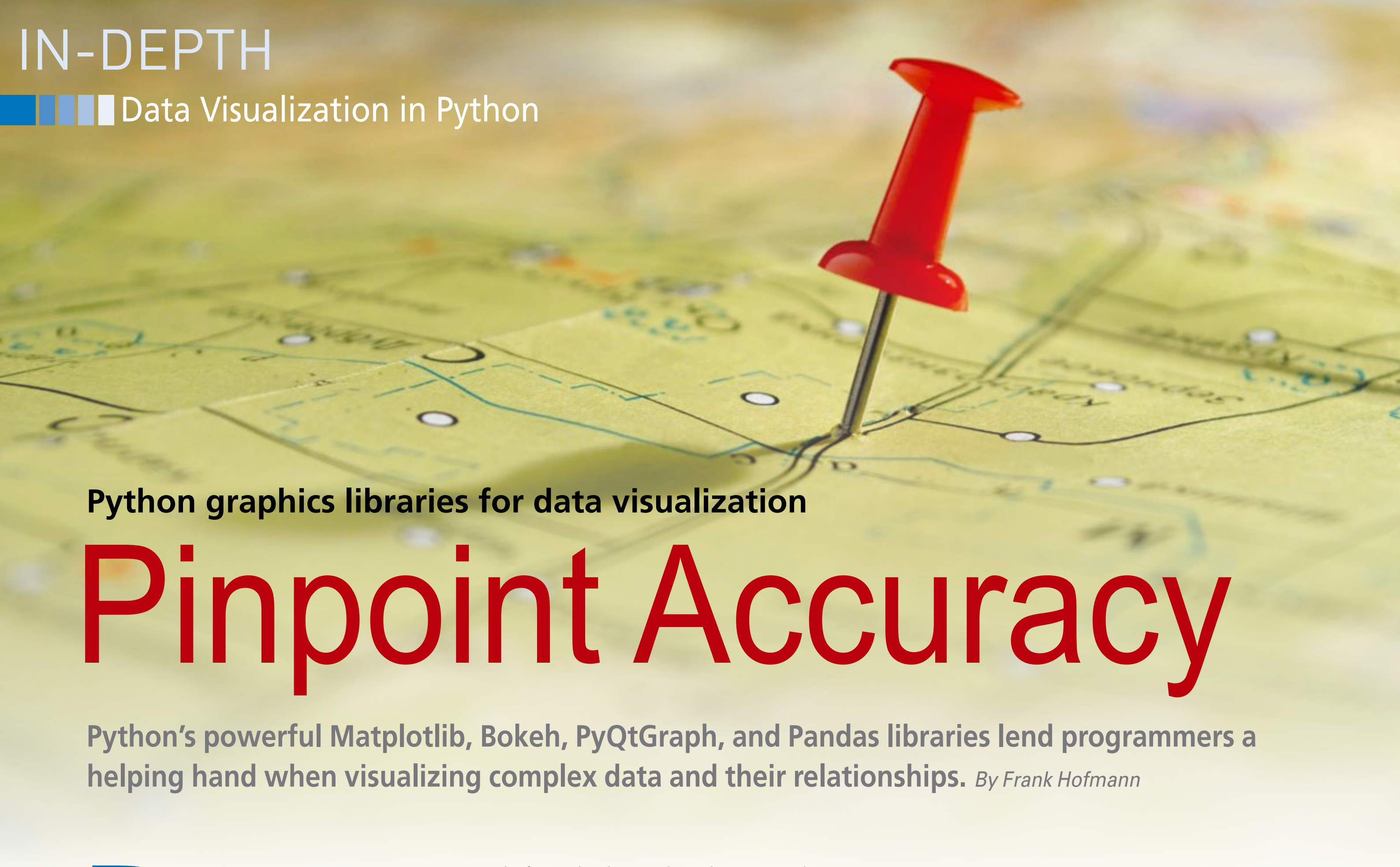
[3] The BlueZ project: *http://www.bluez.org/about/history/*

[4] Bluetooth profiles: *http://www.mobileburn.com/definition.jsp?term=Bluetooth+profile*

[5] hcitool: *http://linuxcommand.org/man_pages/hcitool1.html*

[6] Configuring Bluetooth: *https://wiki.gentoo.org/wiki/Bluetooth*

We're also in Google Play and the iTunes Store!



Python graphics libraries for data visualization

Pinpoint Accuracy

Python's powerful Matplotlib, Bokeh, PyQtGraph, and Pandas libraries lend programmers a helping hand when visualizing complex data and their relationships. *By Frank Hofmann*

Daily life is inundated by data collected, processed, and made available again in edited form. Examples include weather, temperature, precipitation, humidity, air pressure, wind, sales, and load measurements, as well as vehicle and services data.

A first step is to evaluate data in the simplest form as a table that provides an overview of individual values. Beyond this, a suitable graphical visualization helps you to understand the relationships at a glance. Python has excellent libraries to implement this visualization step, including Matplotlib [1], PyQtGraph [2], Bokeh [3], and Pandas [4]. Appropriate libraries for other languages are summarized the "Visualization in Other Languages" box.

In addition to showing data as two- or three-dimensional images, the respective API libraries contain interaction (PyQtGraph, Bokeh) and data analysis (Pandas) methods. With a short sample program, I can show you in each case how to take the initial hurdles in stride. (See the "Installation and Variants" box.)

Matplotlib

The sophisticated Matplotlib library focuses exclusively on presenting graphs, histograms, power spectra, and bar, pie, and error charts in the Cartesian coordinate system in two- and three-dimensional space. Supplementary toolkits like Basemap [20] and Cartopy [21] offer additional projection types and can combine your data with map data.

The website for the project has more suggestions. When browsing around the chart image gallery, you will find the associated, sometimes quite extensive, program code required to produce each chart.

Matplotlib is used in Python scripts, the Python shell, and the IPython Notebook [22] (for Python 2.x) or Jupyter Notebook [23] (for Python 2.7 and 3.3 or greater). The library blends well with web-based application servers and toolkits for creating graphical user interfaces.

As an initial example, I look at visualizing daytime and nighttime temperatures measured over five days. Table 1 lists the collected data; in Figure 1 you

VISUALIZATION IN OTHER LANGUAGES

If you are not a fan of the Python programming language, you might be able to find a matching library in another language. Perl has PLplot [5] and the chart modules Chart::Clicker [6] and GD::Graph [7], and PHP comes with the GD library [8] but can also be combined with JpGraph [9]. JavaScript-based applications can be fitted out with the InfoVis Toolkit [10], D3 [11], or Crossfilter [12]. Suitable candidates for combining JavaScript and jQuery include jqPlot [13], Visualize [14], and Flot [15]. Alternatively, R statistical language [16] incudes functions for plotting data out of the box. An overview of other, similar frameworks and libraries is available, for example, from the website at Datavisualization.ch [17], and Brian Suda's blog has a great overview of libraries [18].

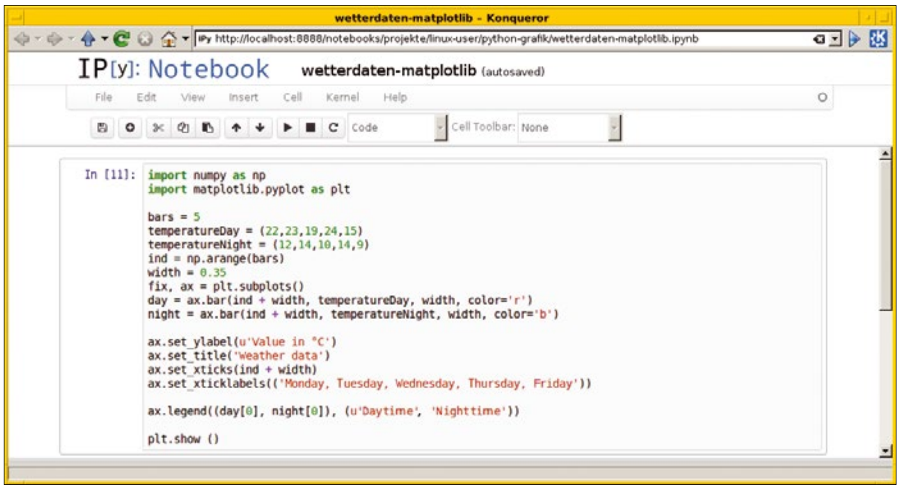
INSTALLATION AND VARIANTS

Matplotlib, PyQtGraph, and Pandas can all be set up conveniently from the repositories in the major distributions using their respective package managers. Under Debian and Ubuntu, for example, the corresponding packages go by the names *python-matplotlib*, *python-pyqtgraph*, and *python-pandas*. However, Bokeh is not available as a package in many distributions. The best idea is to retrieve the package from the project website and install it via Anaconda or Pip, the package management utility for Python packages from the Python Package Index (PyPI).

The listings in this article are mostly based on Python 2.x, but with few or no changes they will run in Python 3.x, which encodes all strings as a sequence of Unicode characters (e.g., to simplify the use of special characters). A handy migration guide provides insight into the differences between the two versions [19].

TABLE 1: Daily Temperature

	Monday	Tuesday	Wednesday	Thursday	Friday
Daytime (°C)	22	23	19	24	15
Nighttime (°C)	12	14	10	14	9


wetterdaten-matplotlib - Konqueror
http://localhost:8888/notebooks/projekte/linux-user/python-grafik/wetterdaten-matplotlib.ipynb
IP[y]: Notebook
wetterdaten-matplotlib (autosaved)
File
Edit
View
Insert
Cell
Kernel
Help
Code
Cell Toolbar: None
In [11]:

```
import numpy as np
import matplotlib.pyplot as plt

bars = 5
temperatureDay = (22,23,19,24,15)
temperatureNight = (12,14,10,14,9)
ind = np.arange(bars)
width = 0.35
fix, ax = plt.subplots()
day = ax.bar(ind + width, temperatureDay, width, color='r')
night = ax.bar(ind + width, temperatureNight, width, color='b')

ax.set_ylabel(u'Value in °C')
ax.set_title('Weather data')
ax.set_xticks(ind + width)
ax.set_xticklabels(('Monday, Tuesday, Wednesday, Thursday, Friday'))

ax.legend((day[0], night[0]), (u'Daytime', 'Nighttime'))

plt.show ()
```

Figure 1: The code in IPython Notebook for generating the chart.

can see the Python code in the IPython Notebook, and Figure 2 shows the graph it generated.

The chart is defined by importing the two libraries `numpy` and `matplotlib.pyplot` under the program-specific identifiers `np` and `plt` (Figure 1, lines 1 and 2).

Line 4 sets the appropriate number of bars for the number of days of measurements. Lines 5 and 6 contain the definition of the temperature values for day and night as a list. In line 7 the `arange` method is used to define a range based on the number of days with measurements, and line 8 sets the appearance of a single bar to 35 percent of the total width.

To prepare the graph with the bars in the background, the command in line 9 generates two objects `fig` and `ax`. Lines 10 and 11 define the appearance of the bars, that is, their position on the *x* axis, the output of the temperatures as the *y* value, and the width and color of the bar. The daytime values appear in red and the nighttime values in blue.

Lines 13 to 16 supplement the chart with a *y* axis label (°C), title, scale, and *x*-axis label (days of the week). In line 18, the legend for the two bars are set at the top right in a separate box. Finally, `plt.show()` renders the chart in the output window (Figure 2).

A second example – the code provided in Listing 1 – visualizes the partitioning of a memory device as a pie chart. Line 1 first includes the `matplotlib.pyplot` graphic library under the local identifier `plt`. Then lines 3 to 6 define for each piece of pie the name (`chartLabels`), the size in percent, the color, and the distance between the pie slice and the center of the graph. The greater the numeric value, the further away from the center the respective piece of pie is shown.

The call in line 8 renders the pie chart along with a shadow, and the method in Line 9 makes a uniform circle of what would otherwise be a somewhat oval appearance. Finally, line 10 calls `plt.show()` to output the prepared pie chart (Figure 3).

PyQtGraph

PyQtGraph, a package written entirely in Python, can be used across operating systems. In addition to graphical representations (visualizations), it offers an interface for application and GUI development by docking onto Qt with two frameworks: PyQt [24] and PySide [25].

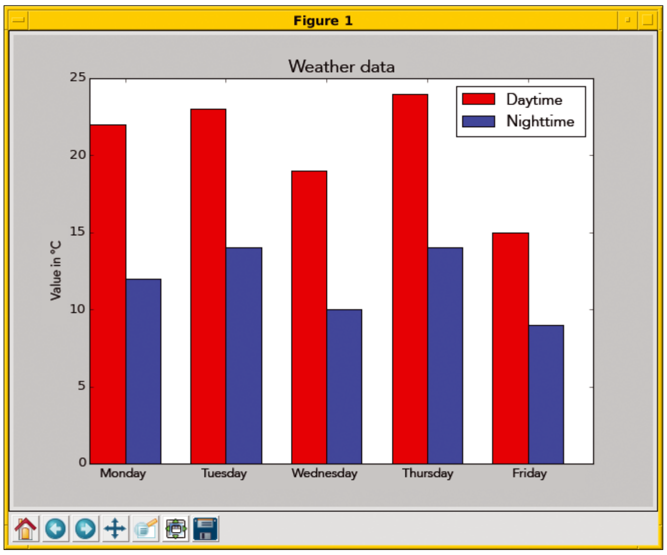


Figure 2: The weather chart generated with Matplotlib.

LISTING 1: Matplotlib Pie Chart

```
01 import matplotlib.pyplot as plt
02
03 chartLabels = '/', '/usr', '/home', '/var', '/tmp', '/opt'
04 chartSizes = [10, 15, 40, 10, 10, 15]
05 chartColors = ['yellow', 'green', 'blue', 'red', 'white', 'orange']
06 chartExplode = (0, 0, 0.2, 0, 0, 0)
07
08 plt.pie(chartSizes, explode=chartExplode, labels=chartLabels, colors=chartColors, autopct='%1.1f%%', shadow=True)
09 plt.axis('equal')
10 plt.show()
```

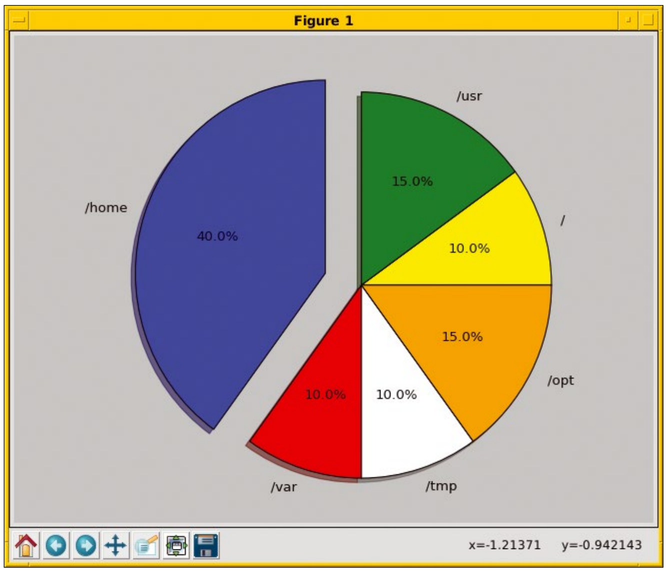


Figure 3: The partitioning scheme of a hard disk as a pie chart with Matplotlib.

In terms of content, it offers visualization for scientific purposes, and a correspondingly high degree of accuracy, by supplementing the functionality of Matplotlib, adding the ability to select a section of the graphical output interactively with the mouse, and then scaling the section and rotating and moving it along the coordinate axes. PyQtGraph is suitable for creating applications in which the representation of the data in the output changes at program run time, or in which the user can intervene (e.g., load curves, animations, video, and movie excerpts).

The example shown in Figure 4 shows a curve in two images side by side. The left curve provides an overview (value range 0 to 1,000) with two markers at 300 and 700 on the *x* axis; the right curve contains only the snippet inside these marks.

To move the section, first left-click in the highlighted area of the left diagram. Then, hold down the mouse button and drag the pointer left or right. To change the boundaries of the excerpt, left-click on the respective selection, hold down the left mouse button, and drag the boundary of the excerpt. In both cases, the representation in the right chart changes synchronously. You can use the right mouse button to access a contextual menu in which you can modify the presentation (e.g., for the coordinate grid system). You can configure each diagram separately (Figure 5).

Listing 2 shows the underlying program code. After defining the required modules (lines 1 to 3), the Qt application is initialized in line 5. Lines 7 to 9 define a graphic window, 700x200 pixels here, and a corresponding window title.

Starting in line 11 the two graphs are specified. First, anti-aliasing of the curves shown is enabled, and the scope and data points are defined. The variable `p1` represents the left diagram; the counterpart `p2` is on the right. Lines 15 and 21 define corresponding titles. Line 16 specifies the color for plotting the data points as an RGBA value. The curve appears in white with an alpha value of 200.

You define the initial section for the right curve in line 17 and the corresponding functions for updating in lines 24 to 28. Lines 30 and 31 create the link between the two charts by referencing the previously defined functions. The final lines 34 and 38 ensure that the Qt application is called correctly.

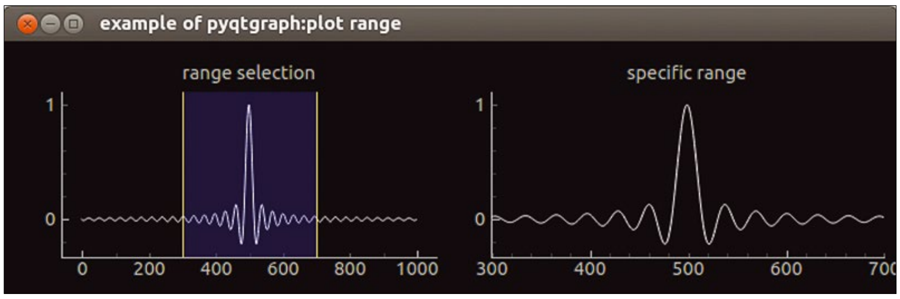


Figure 4: A graph with an interactively displaceable section shown separately on the right.

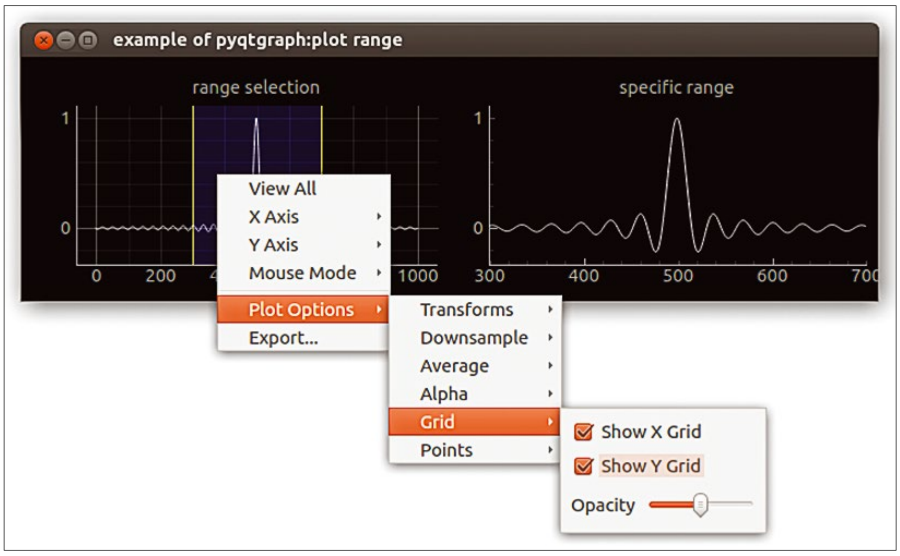


Figure 5: Using a context menu to modify the view shown in Figure 4.

Bokeh

Bokeh helps you create interactive graphics and pictures to be included in a web page. A web server such as Apache or Nginx then serves up this page with the figure. The library combines the figure with HTML and JavaScript, allowing the viewer to change the appearance of the graphic in their web browser (e.g., selecting a view, a data range, or the color of the data points).

Bokeh is designed to handle large amounts of data. It features a Matplotlib compatibility layer and can be combined with IPython Notebook. The package also includes extensive sample data you can immediately use for test purposes [26], such as the data in Figure 6, which shows an interactive periodic table.

Clicking on an element square opens a small window with further details, as shown for the element calcium (Ca) here. The additional information in-

LISTING 2: Graph and Excerpt

```
from pyqtgraph.Qt import QtGui, QtCore
import numpy as np
import pyqtgraph as pg

app = QtGui.QApplication([])

win = pg.GraphicsWindow(title="Move extract")
win.resize(700,200)
win.setWindowTitle('example of pyqtgraph:plot range')

pg.setConfigOptions(antialias=True)

x = np.linspace(-100, 100, 1000)
data = np.sin(x) / x
p1 = win.addPlot(title="range selection")
p1.plot(data, pen=(255,255,255,200))
lr = pg.LinearRegionItem([300,700])
lr.setZValue(-10)
p1.addItem(lr)

p2 = win.addPlot(title="specific range")
p2.plot(data)

def updatePlot():
    p2.setXRange(*lr.getRegion(), padding=0)

def updateRegion():
    lr.setRegion(p2.getViewBox().viewRange()[0])

lr.sigRegionChanged.connect(updatePlot)
p2.sigXRangeChanged.connect(updateRegion)
updatePlot()

## Start Qt event loop unless running in interactive mode or using pyside.
if __name__ == '__main__':
    import sys
    if (sys.flags.interactive != 1) or not hasattr(QtCore, 'PYQT_VERSION'):
        QtGui.QApplication.instance().exec_()
```

cludes the full name of the element, atomic number, element type, atomic mass, color code according to the CPK model [27], which is a convention for the color representation of atoms in molecule models (named after the chemists Robert Corey, Linus Pauling, and Walter Koltun), as well as the electronic configuration.

The program code for Bokeh examples is quite extensive and complex, so I have not printed it here. Basically, it is equivalent to Matplotlib and PyQtGraph code.

Pandas

The Python Data Analysis (Pandas) Library is available under a BSD-style license. The name is derived from "panel data" and suits multidimensional structured data. The developers have optimized Pandas for speed and therefore implemented critical sections of code in Cython and C. Cython is an optimizing static compiler for Python and the advanced programming language of the same name. It creates compiled C programs that run several times faster than the source code in the Python interpreter.

Figure 7 compares the frequency of selected children's names; see Listing 3 for the corresponding code. After defining the required modules (Lines 1 to 3), you set the data to be visualized. Lines 5 and 6 define two lists, one with the

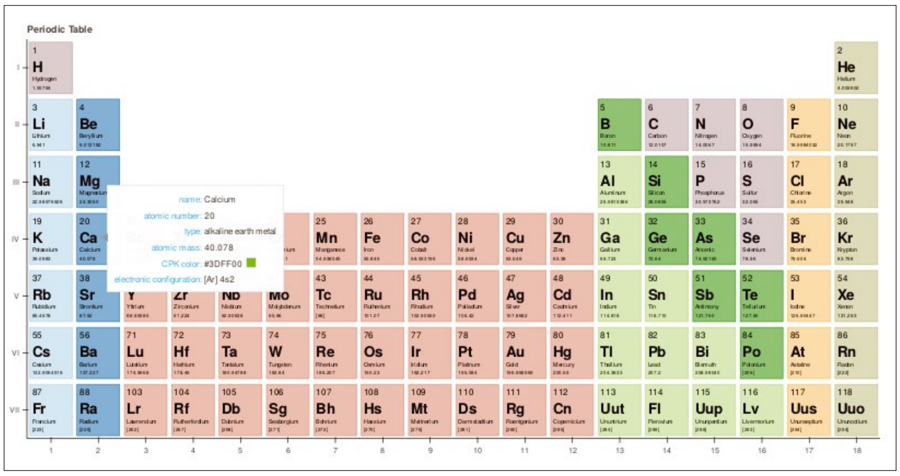


Figure 6: Interactive periodic table of the elements.

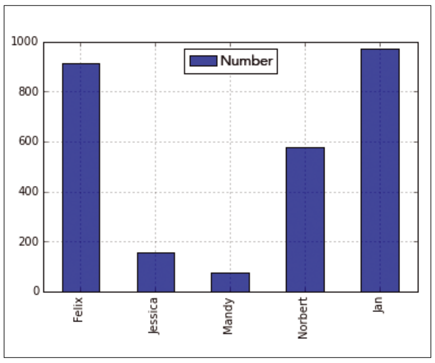


Figure 7: Frequency of children's names.

name and another with the respective frequency.

Using the internal Python functions `list` and `zip`, line 7 creates a two-dimensional dataset, from which Pandas generates a view in line 9. You need to assign the collected data to the `data` parameter, the data labels to `columns`, and the names defined in line 5 to `index`.

Pandas renders the frequency of the name along the *y* axis while outputting the name in the corresponding bar on the *x* axis as an identifier. The call in line 10 uses `df.plot()` and the parameter `kind='bar'` to produce the chart.

LISTING 3: Pandas Plot

```
01 import matplotlib.pyplot as plt
02 import pandas as pd
03 import numpy as np
04
05 names = ['Felix','Jessica','Mandy','Norbert','Jan']
06 births = [915, 155, 77, 578, 973]
07 babyDataSet = list(zip(names,births))
08
09 df = pd.DataFrame(data = babyDataSet, columns=['Names', 'Number'], index=names)
10 df.plot(kind='bar')
```

Conclusions

The Matplotlib, PyQtGraph, Bokeh, and Pandas libraries can help programmers convert abstract data to clear-cut representations with a reasonable amount of effort. Basic knowledge of Python considerably simplifies the use of these visualization libraries. If you are looking to create a specific look, you might need to experiment.

The featured libraries provide extensive capabilities and require a careful reading of the API to set the parameters correctly. Thus, you need patience and the motivation to work through the detailed documentation. The supplied examples will help you familiarize yourself with the order and effect of the calls. ■■■

AUTHOR

Frank Hofmann (*http://www.efho.de*) works in Berlin as a service provider at Büro 2.0, an open source experts' network specializing in printing and typesetting. Since 2008, he has coordinated the regional meeting of LUGs from the Berlin-Brandenburg Region and is coauthor of the Debian-Paketmanagement-Buch (Debian Package Management Book, *http://www.dpmb.org*).

INFO

[1] Matplotlib: *http://matplotlib.org/*

[2] PyQtGraph: *http://www.pyqtgraph.org/*

[3] Bokeh: *http://bokeh.pydata.org/en/latest/*

[4] Pandas: *http://pandas.pydata.org/*

[5] Perl module PLplot: *http://plplot.sourceforge.net/index.php*

[6] Perl module Chart::Clicker: *https://metacpan.org/pod/Chart::Clicker*

[7] Perl module GD::Graph: *https://metacpan.org/pod/GD::Graph*

[8] GD library (PHP): *http://php.net/manual/en/book.image.php*

[9] JpGraph (PHP): *http://jpgraph.net/*

[10] JavaScript InfoVis Toolkit: *http://philogb.github.io/jit/*

[11] D3: *https://d3js.org/*

[12] Crossfilter: *http://square.github.io/crossfilter/*

[13] jqPlot: *http://www.jqplot.com/*

[14] jQuery-Visualize: *https://github.com/filamentgroup/jQuery-Visualize*

[15] Flot: *http://www.flotcharts.org/*

[16] R: *https://www.r-project.org/*

[17] Data visualization tools: *http://selection.datavisualization.ch/*

[18] "The 38 best tools for data visualization" by By Brian Suda and Sam Hampton-Smith: *http://www.creativebloq.com/design-tools/data-visualization-712402*

[19] "What's new in Python 3" by Rainer Grimm, *Linux Pro Magazine*, issue 107, October 2009, pg. 42, *http://www.linuxpromagazine.com/Issues/2009/107/Python-3*

[20] Basemap: *http://matplotlib.org/basemap/*

[21] Cartopy: *http://scitools.org.uk/cartopy/docs/latest/*

[22] IPython Notebook: *https://ipython.org/ipython-doc/3/notebook/notebook.html*

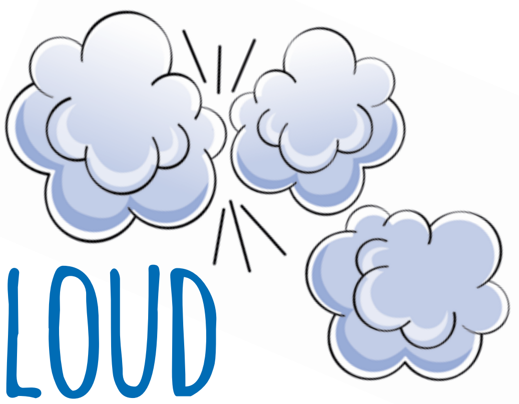
[23] Jupyter Notebook: *http://jupyter.org/index.html*

[24] PyQt on the Python wiki: *https://wiki.python.org/moin/PyQt*

[25] PySide on the Qt wiki: *http://wiki.qt.io/PySide*

[26] Bokeh examples: *http://bokeh.pydata.org/en/latest/docs/gallery/periodic.html*

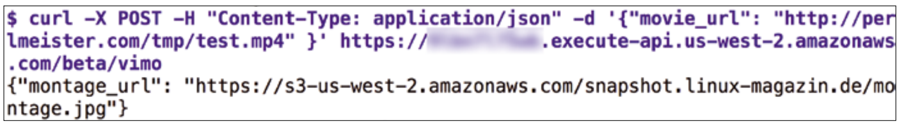
[27] CPK color model: *https://en.wikipedia.org/wiki/CPK_coloring*



Set up Amazon Web Services – Part 2

Home Run into the Cloud

DIY Python scripts run in container environments on Amazon's Lambda service – this snapshot example deploys an AI program for motion analysis in video surveillance recordings. By Mike Schilli



```
$ curl -X POST -H "Content-Type: application/json" -d '{"movie_url": "http://per
lmeister.com/tmp/test.mp4" }' https://[redacted].execute-api.us-west-2.amazonaws
.com/beta/vimo
{"montage_url": "https://s3-us-west-2.amazonaws.com/snapshot.linux-magazin.de/mo
ntage.jpg"}
```

Figure 1: The AI program for motion analysis runs on Amazon servers behind a REST API.

After some initial steps in a previous article [1] to set up an AWS account, an S3 storage server with a static web server, and the first Lambda function, I'll now show you how to set up an API server on Amazon to track down interesting scenes in videos from a surveillance camera.

The Lambda function triggered either by a web request from the browser or a command-line tool like `curl` retrieves a video from the web, runs it through an artificial intelligence (AI) algorithm implemented by the OpenCV library, generates a motion profile, and returns the URL of a contact sheet generated as a JPEG with all the interesting movements from the recording (Figures 1 and 2).

Sandbox Games

Unlike Amazon's EC2 instances with their full-blooded (albeit virtual) Linux servers, the Lambda Service [2] provides only a containerized environment. Inside a container, Node.js, Python, or Java programs run in a sandbox, which Amazon pushes around at will between physical servers, eventually going as far as putting the container to sleep in case of inactivity – just to conjure it up again when next accessed. Leaving data on the virtual disk of the container and hoping to find it still there next time would thus result in an unstable application. Instead, Lambda functions communicate with AWS offerings such as S3 storage or the Dynamo database to secure data and are otherwise "stateless."

Developers can upload things that an application cannot describe in a Python script to the (as rumor has it) CentOS-based containers as ZIP files (Figure 3).

A Lambda function that uses artificial intelligence capabilities from the OpenCV library, like the example, needs to compile the required binaries or libraries up front in a Unix environment similar to the Lambda container, package and upload the results, and call it with the Python script at run time. Existing Python bindings to shared libraries are used here, or the Python script calls precompiled binaries as external processes.

Lean and Mean

To prevent the AI program [3] from using too much compute time after installation in the Amazon cloud – and thus also using up money after exceeding the "free tier" quota – the improved code [4] (updated in Listing 1 from the previous article) no longer looks for movements in every frame (i.e., 50 times a second) but hops through the movie in increments of half a second in line 99. After a frame with detected motion, line 96 even skips forward two seconds. To accomplish this, `vid.grab()` called in line 50 no longer painstakingly decodes the frame in a complex process, as `vid.read()` did previously, but discards it to retrieve the next one.

Whereas the first version [3] only printed the number of seconds into the video in which the algorithm detected motion, to subsequently use MPlayer to extract the frames as JPEG files, lines 92 to 94 now use the `imwrite()` image processing functions included with OpenCV to write detected frames immediately as `000x.jpg` to the virtual disk. A second run and the shenanigans for installing MPlayer in the Lambda container are thus no longer required.

Based on these JPEG images, another Python script, `mk-montage.py`, then produces a contact sheet, also in `.jpg` format, with the help of the ImageMagick library. The Lambda program puts this file into Amazon's S3 cloud storage, and then sends a link to the file to the calling client.

RAM Is Money

How does a Python programmer now pick up a document from the web? A first approach would be the `read()` method provided by `urlopen()`, which

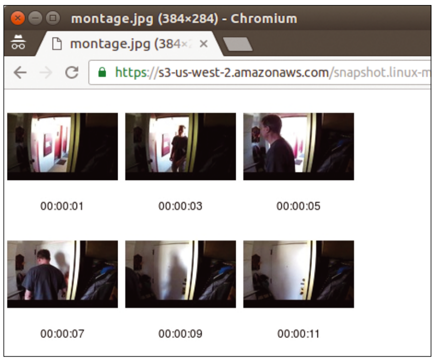

montage.jpg (384×284) - Chromium
montage.jpg (384×
https://s3-us-west-2.amazonaws.com/snapshot.linux-m
00:00:01
00:00:03
00:00:05
00:00:07
00:00:09
00:00:11

Figure 2: The contact sheet produced on AWS displays the seconds in the surveillance video during which something actually moved.

MIKE SCHILLI

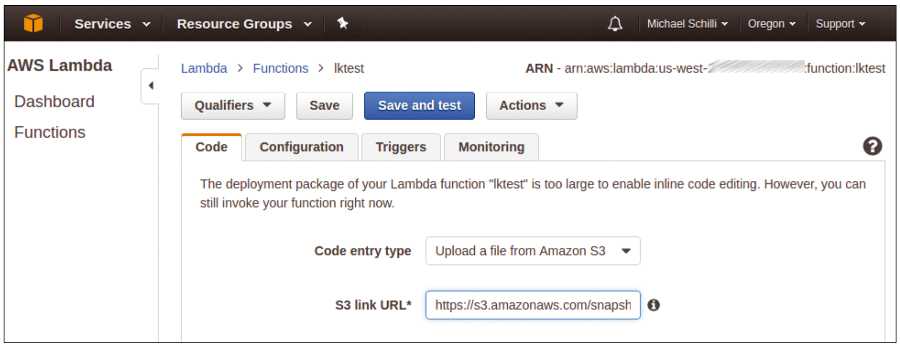
Mike Schilli works as a software engineer in the San Francisco Bay Area. He can be contacted at *mschilli@perlmeister.com*. Mike's homepage can be found at *http://perlmeister.com*.

then sends all the bytes it has obtained to a local file through `write()`. But, this would mean that a potentially large video file would be completely read into memory before Python finally starts writing it to disk.

The ample supply of RAM needed for this costs money on Amazon. To avoid this, the `urlretrieve()` method from the `urllib` module used in Listing 2 can buffer smaller data chunks – in a hopefully more or less intelligent way.

LISTING 1: max-movement-lk.cpp

```
#include "opencv2/opencv.hpp"
#include <stdio.h>

using namespace std;
using namespace cv;

const int MAX_FEATURES = 500;
const int MAX_MOVEMENT = 100;

int move_test(Mat& oframe, Mat& frame) {
    // Select features for optical flow
  vector<Point2f> ofeatures;
  goodFeaturesToTrack(oframe,
    ofeatures, MAX_FEATURES, 0.1, 0.2 );

    // Parameters for LK
  vector<Point2f> new_features;
  vector<uchar> status;
  vector<float> err;
  TermCriteria criteria(TermCriteria::COUNT
      | TermCriteria::EPS, 20, 0.03);
  Size window(10,10);
  int max_level   = 3;
  int flags       = 0;
  double min_eigT = 0.004;

    // Lucas-Kanade method
  calcOpticalFlowPyrLK(oframe, frame,
    ofeatures, new_features, status, err,
    window, max_level, criteria, flags,
    min_eigT );

  double max_move = 0;
  double movement = 0;
  for(int i=0; i<ofeatures.size(); i++) {
    Point pointA
      (ofeatures[i].x, ofeatures[i].y);
    Point pointB
      (new_features[i].x, new_features[i].y);

    movement = norm(pointA-pointB);
    if(movement > max_move)
        max_move = movement;
  }
  return max_move > MAX_MOVEMENT;
}

int frames_skip( VideoCapture vid, int n, int *i ) {
    for( int c = 0; c < n; c++ ) {
      if (!vid.grab())
        break;
      (*i)++;
    }
}

int main(int argc, char *argv[]) {
  int i = 0;
  Mat frame;
  Mat cframe;
  Mat oframe;

  if (argc != 2) {
    cout << "USAGE: <cmd> <file_in>\n";
    return -1;
  }

  VideoCapture vid(argv[1]);
  if (!vid.isOpened()) {
    cout << "Video corrupt\n";
    return -1;
  }

  int fps = (int)vid.get(CV_CAP_PROP_FPS);

  i++;
  if(!vid.read(oframe)) return 1;

  cvtColor(oframe, oframe, COLOR_BGR2GRAY);

  while (1) {
    if (!vid.read(frame))
      break;
    i++;

    int movie_second = i / fps;

    cframe = frame.clone();
    cvtColor(frame,frame,COLOR_BGR2GRAY);
    if(move_test(oframe, frame)) {
      cout << movie_second << "\n";

     char filename[80];
     sprintf( filename, "%04d.jpg", i/fps );
     imwrite( filename, cframe );

      frames_skip( vid, 2*fps, &i );
    } else {
        // fast-forward to next 1/2 sec
      frames_skip( vid, fps/2, &i );
    }

    oframe = frame;
  }

  return 0;
}
```



The deployment package of your Lambda function "lktest" is too large to enable inline code editing. However, you can still invoke your function right now.

Code entry type: Upload a file from Amazon S3

S3 link URL*: https://s3.amazonaws.com/snapsh

Figure 3: Uploading code in a ZIP file to the Lambda server via an Amazon S3 bucket.

LISTING 2: vimo.py

```
01 #!/usr/bin/python
02 import urllib
03 import tempfile
04 import shutil
05 import subprocess
06 import boto3
07 import os
08
09 def lambda_handler(event, context):
10     tmpd = tempfile.mkdtemp()
11
12     # fetch movie
13     movie_url  = event['movie_url']
14     movie_file = os.path.join(tmpd,
15         os.path.basename(movie_url))
16     urllib.urlretrieve(movie_url,movie_file)
17
18     # motion analysis
19     print subprocess.check_output([
20         "bin/max-movement-lk.py",
21         movie_file])
22
23     # generate montage
24     print subprocess.check_output([
25         "bin/mk-montage.py",tmpd])
26
27      # store montage in s3
28     s3 = boto3.resource('s3')
29     bucket = "snapshot.linux-magazine.com"
30     data = open(os.path.join(
31          tmpd,'montage.jpg')).read()
32     s3.Bucket(bucket).put_object(
33         Key="montage.jpg",
34         Body=data,ContentType="image/jpeg")
35
36     result = { "montage_url":
37       "https://s3-us-west-2.amazonaws.com" +
38       "/snapshot.linux-magazine.com/" +
39       "montage.jpg"}
40
41     shutil.rmtree(tmpd)
42     return result
```

Dichotomy: Python 2 and 3

The Python world suffers from the contradictions between Python 2.x and 3. The latter is considered to be a paradise-like state, in which all teething troubles and glaring inconsistencies have been fixed, and cool new developments are taking place; except, hardly anyone uses Python 3 in production environments, and Amazon only offers 2.7.

In Python 2.x, programmers battle with a ludicrous sprawl of libraries and need to decide between incompatible products like `urllib` and – I kid you not – `urllib2` when fetching web data. If you want to run external software with 2.x, you would use `check_output()` from the `subprocess` module. In Python 3.x, however, the `run()` method uses different parameters and `check_output()` has apparently vanished.

Lambda Go

The Lambda function in Listing 2 takes the URL for the video file to be analyzed in the `event` parameter dictionary under the `movie_url` key. In a genuine production environment, no Python script can modify data in a fixed directory like `data` and just hope that nobody else interferes. Because Amazon Lambda functions are called in parallel, they first need to create a temporary directory for the instance using Python's `tempfile` module and then clean it up again when done.

To ensure that this happens if one of the functions throws an exception after a failure, the last line in a production script would be an exception handler; this was left out in the test version shown here. Line 10 of Listing 2 calls the `mkdtemp()` method and uses the new directory to store the data determined in intermediate steps for the next stages in the script.

Line 16 stores the video file retrieved by a web request to a URL stored in the variable `movie_file`; the last part of the path determines the local file name. The next step is for lines 19 to 21 to call the `max-movement-lk.py` script in Listing 3 (this is a Python wrapper around the C program from Listing 1) and passes in the path to the video file located in the temporary directory.

On Site

If the analysis results in a number of JPEGs of format `0001.jpg`, `0002.jpg`, etc.,

LISTING 3: max-movement-lk.py

```
#!/usr/bin/python
import sys
import os
import subprocess

top_dir     = os.getcwd()
movie_path = sys.argv[1]

os.chdir(os.path.dirname(movie_path))

os.environ["LD_LIBRARY_PATH"] = os.path.join(top_dir,"lib")

print subprocess.check_output(
   [ os.path.join(top_dir, "bin/max-movement-lk") ] +
   [ os.path.basename(movie_path) ] )
```

LISTING 4: mk-montage.py

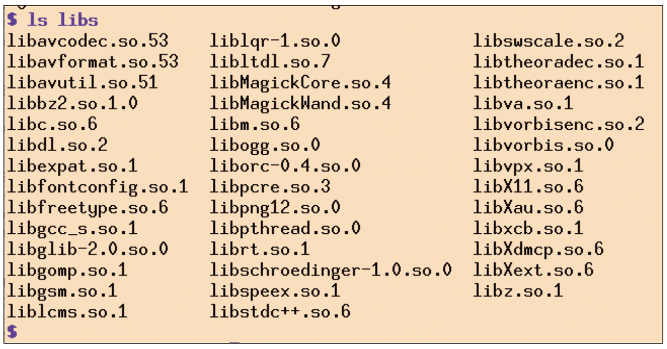
```
#!/usr/bin/python
import glob
import subprocess
import re
import time
import os
import sys

dir = sys.argv[1]
files = glob.glob(
       os.path.join(dir,'*.jpg'))
cmds  = ["bin/montage.py"]

r = re.compile('.*?(\d+)')

for file in sorted(files):
    match = r.match(file)
    if match:
        label =
         time.strftime("%H:%M:%S",
         time.gmtime
         (int(match.group(1))))
        cmds.append("-label")
        cmds.append(label)
        cmds.append(file)
    else:
        print "no match: " + file

cmds.append(os.path.join(
    dir,'montage.jpg'))

print subprocess.check_output(cmds)
```

the `mk-montage.py` wrapper script from Listing 4 runs as of lines 24 and 25 in Listing 2. The script converts the seconds into the movie embedded in the file names to an `HH::MM:SS` format and feeds the old file names and the formatted labels to ImageMagick's `montage` tool:

```
montage.py -label  00:00:01 ↲
  tmp/00001.jpg  [...]
```

The program uses this to build a contact sheet in the `montage.jpg` file, which ends up on Amazon's S3 storage system later, so that users can follow the link returned by the script later to download the content.

The Python script in Listing 5 acts as a wrapper around the `montage` binary, which is accompanied by a bunch of shared libraries in the `lib` directory so that the dynamically linked binary will run in the container. The `LD_LIBRARY_PATH` environment variable sets the



```
$ ls libs
libavcodec.so.53      liblqr-1.so.0                libswscale.so.2
libavformat.so.53     libltdl.so.7                 libtheoradec.so.1
libavutil.so.51       libMagickCore.so.4           libtheoraenc.so.1
libbz2.so.1.0         libMagickWand.so.4           libva.so.1
libc.so.6             libm.so.6                    libvorbisenc.so.2
libdl.so.2            libogg.so.0                  libvorbis.so.0
libexpat.so.1         liborc-0.4.so.0              libvpx.so.1
libfontconfig.so.1    libpcre.so.3                 libX11.so.6
libfreetype.so.6      libpng12.so.0                libXau.so.6
libgcc_s.so.1         libpthread.so.0              libxcb.so.1
libglib-2.0.so.0      librt.so.1                   libXdmcp.so.6
libgomp.so.1          libschroedinger-1.0.so.0     libXext.so.6
libgsm.so.1           libspeex.so.1                libz.so.1
liblcms.so.1          libstdc++.so.6
$
```

Figure 4: A collection of shared libs.

search path for shared libs on the nonstandard directory so that the binary actually finds them at run time.

Figure 4 shows the shared libs collected by Listing 6. Apparently, the AI program for motion analysis linked with OpenCV drags a trunk full of dependent libraries around with it. Video analysis requires sophisticated decompression technology if you want access to the raw frame data.

Safely Stored

Once the `montage.jpg` contact sheet has been created, the code from line 28 in Listing 2 copies the file from the temporary directory to a previously created S3 bucket on Amazon's cloud storage system. The `boto3` Python module is available by default on Lambda servers and offers various tools for communication with related service offerings.

The `put_object()` method in line 32 stores the output file read from the virtual disk as a type `image/jpeg` object in the cloud storage. From there, the S3 web server discussed in the previous issue delivers it to interested users under the URL the API returned. For this, lines 36 to 39 compose a JSON response, which tells the web client which S3 URL refers to the `montage` results file. In line 41, all that remains is to delete the temporary directory created previously.

To give the Lambda script write permissions for the `snapshot.linux-magazine.com` S3 bucket, the user needs to enable write access for the latter. Figure 5 shows that the S3 bucket grants write access to any designated AWS user. On the other side of the wall, files generated by the Lambda server in an S3 bucket must be globally readable for interested users. This is done using a bucket policy, the contents of which are shown in Listing 7. Each newly created file is thus readable for all; that is, the web server in the S3 bucket can deliver the file to any requesting web client.

Gateway to the World

Amazon helps you test Lambda functions; developers can either run uploaded scripts via the `aws` command-line utility or press the console's *Test* button in the browser. Ultimately, users need to be able to access functions via the web and Amazon's API Gateway. This service, which can also be configured in the console, creates a cloud server with a REST API, whose methods (as in the `/vimo` example shown) it redirects to user-defined Lambda functions, among other options.

AWS handles the connection between the web server and the application on the Lambda Service behind the scenes

LISTING 5: montage.py

```
#!/usr/bin/python
import sys
import os
import subprocess

os.environ["LD_LIBRARY_PATH"] = "lib"

print subprocess.check_output(
  [ "bin/montage" ] + sys.argv[1:])
```

LISTING 6: ldd-ls.py

```
#!/usr/bin/python
import subprocess;
import sys;

if len(sys.argv) != 2:
    print("usage:
     {} file".format(sys.argv[0]))
    sys.exit(1)

file = sys.argv[1]

output =
  subprocess.check_output(['ldd',file])
for line in output.split("\n"):
    words = line.split()
    if len(words) > 3:
        print words[2]
```

when the user stipulates the *Lambda Function* option as the Integration Type when creating the REST method (e.g., `GET` or `POST`), and defining the data center region and the name of the Lambda function further down (e.g., `vimo`; Figure 6).

In this case, the client will use the path `/vimo` to `POST` data in JSON format with named parameters like `movie_url`. If the web client, as shown in Figure 1, sets the `Content-Type` header of the request to `application/json`, then the API Gateway will expect JSON and decode it for consumers down the pipeline.

The Lambda function called later then receives the decoded value pairs from the JSON enchilada in the `event` function parameter as a Python dictionary. The client in Figure 1 provides the URL for the surveillance movie to be analyzed in JSON as `movie_url` , while the Lambda function in Listing 2 uses `event['movie_url']` to access it.

The REST API does not go live until the user clicks on the *Deploy API* function in the context menu below API Actions (Figure 7) and selects a production environment (called "Stage"), for example *Beta*. In the browser, AWS then displays the URL of where to reach the newly configured web service.

If you are installing in a production environment, the use of API tokens that manage access to the API is recommended. You can also slow down the access rate (e.g., to 1000 requests/second) to prevent an unexpected cost explosion if the link spreads like a wildfire.

While I was working on this article, I kept a watchful eye on the costs incurred, but they stayed within the "Free Tier" scope; it only cost me $0.01 to cover the bandwidth consumption for uploading the continuously updated and improved ZIP files with the test code and the libs. ■■■

LISTING 7: bucket-policy.json

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Sid": "",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::
      snapshot.linux-magazine.com/*"
  }
  ]
}
```

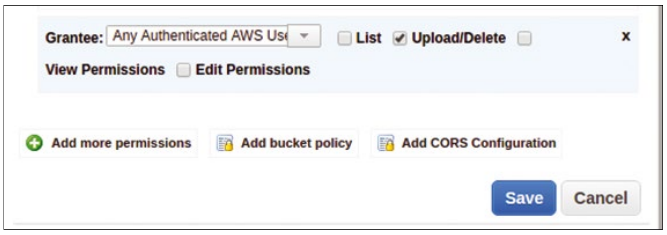


Figure 5: The Lambda server needs access privileges for the S3 bucket.

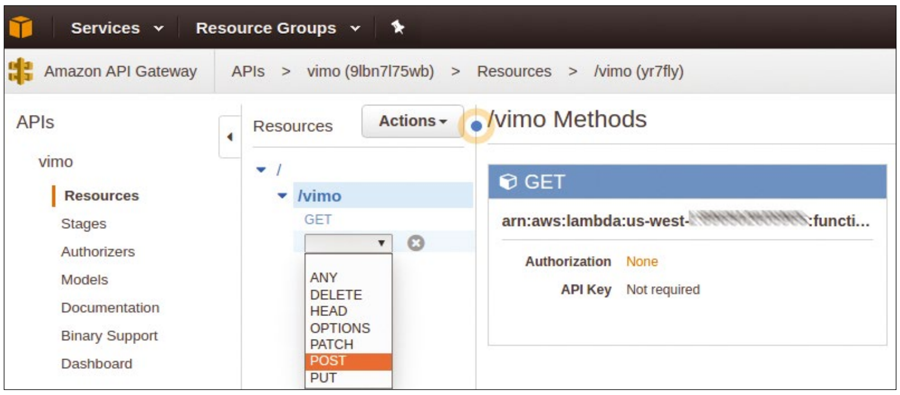


Figure 6: Creating the REST method with GET.

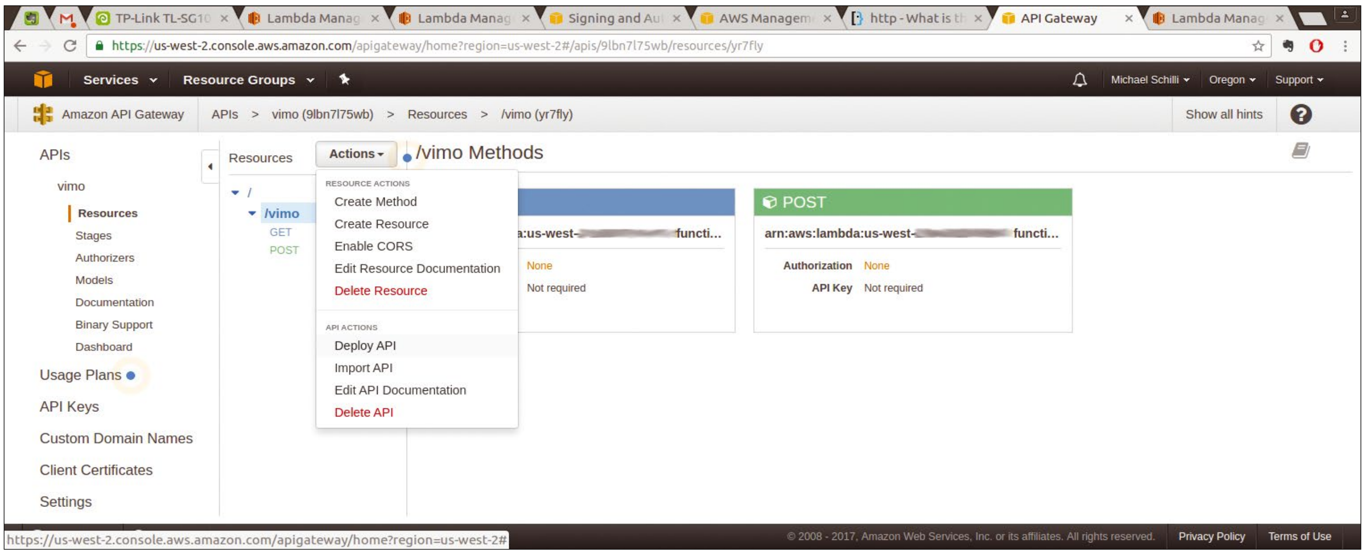


Figure 7: Going live with the *Deploy API* function.

INFO

[1] "Programming Snapshot – Amazon Web Services" by Mike Schilli. *Linux Pro Magazine*, issue 196, 2017, pg. 52.

[2] Poccia, Danilo. *AWS Lambda in Action*. Manning, 2017

[3] "Perl – Video Preview" by Mike Schilli. *Linux Pro Magazine*, issue 195, 2017, pg. 52. *http://www.linux-magazine.com/Issues/2017/195/Perl-Video-Preview*

[4] Listings for this article: *ftp://ftp.linux-magazine.com/pub/listings/magazine/197*

Ben Everard

One of the slightly ironic after-effects from the merger of *Linux Voice* and *Linux Magazine* is that I now spend a lot more time using Linux. OK, that's not quite right; I used Linux more-or-less all day before, and I still do now, but now I spend more time really getting my hands dirty with the internals of servers. As my interactions have changed, my preferred method of interacting with my computers has changed as well. I've started re-learning all the Vim keybindings that I forgot when most of my work was done in word processors, and I usually have an over-complex tmux session running in a terminal. (I'm sure I spend at least 10% of my time trying to find the right session, just like I spend about the same amount of time trying to find which web browser tab the thing I want is in.) All this means I spend most of my time with my hands on the keyboard rather than the mouse, and I've been thinking more and more about switching away from my trusty Unity desktop to a tiled window manager. It seems I'm not the only person with this on his mind at the moment, as Mike Saunders has already taken the plunge; I'll read his tutorial in this month's issue before making the switch myself.

For the rodent fans among you, we haven't abandoned the mouse. Graham picks out the latest and greatest free software in FOSS Picks, including Calligra, which I've always felt has been a bit unfairly overshadowed by LibreOffice.

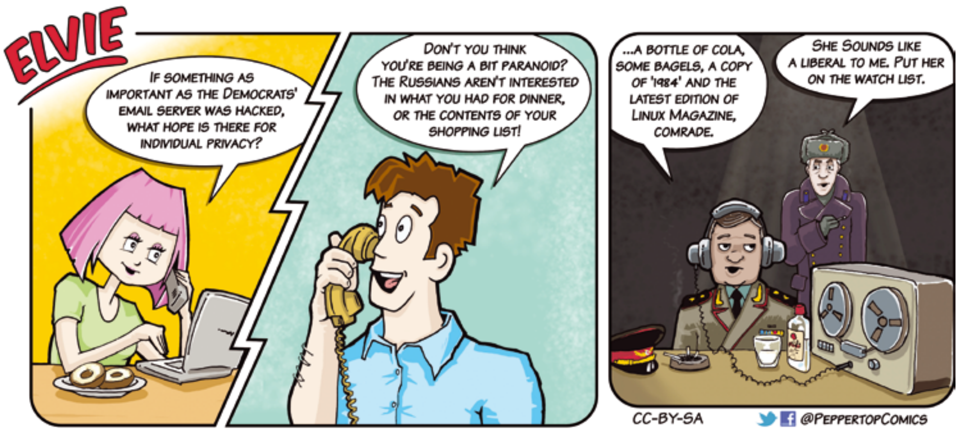
Our columnists bravely step outside to chat about open source software face-to-face. Simon Phipps ventures overseas to argue with geeks about free software licenses, Andrew Gregory hears about the virtues of FOSS from Windows users, and maddog visits the Brazilian Project Cauã. If you prefer to take your voyages into computers rather than the outside, Valentine Sinitsyn will, as always, guide you deeper into your Linux kernel than most people can navigate. Turn the page and choose your own Linux adventure.

– Ben Everard

Andrew Gregory

Graham Morrison

Mike Saunders


ELVIE
IF SOMETHING AS IMPORTANT AS THE DEMOCRATS' EMAIL SERVER WAS HACKED, WHAT HOPE IS THERE FOR INDIVIDUAL PRIVACY?
DON'T YOU THINK YOU'RE BEING A BIT PARANOID? THE RUSSIANS AREN'T INTERESTED IN WHAT YOU HAD FOR DINNER, OR THE CONTENTS OF YOUR SHOPPING LIST!
...A BOTTLE OF COLA, SOME BAGELS, A COPY OF '1984' AND THE LATEST EDITION OF LINUX MAGAZINE, COMRADE.
SHE SOUNDS LIKE A LIBERAL TO ME. PUT HER ON THE WATCH LIST.
CC-BY-SA
@PEPPERTOPCOMICS

LINUXVOICE

NEWSANALYSIS

The Linux Voice view on what's going on in the world of Free Software.

Opinion

Is the GPL Declining?

The state of copyleft in an age of permissiveness. BY SIMON PHIPPS

Simon Phipps is ex-president of the Open Source Initiative and a board member of the Open Rights Group and of The Document Foundation (makers of LibreOffice).

At the huge FOSDEM developer meetup in Brussels in early February, I attended a panel where speakers discussed whether the use of "permissive" open source licenses like the Apache License is now outstripping the use of "viral" licenses, such as the GPL. The discussion was spirited, with advocates associated with the Free Software Foundation pushing back on the assertion the GPL is "dying."

Reciprocal vs. Non-Reciprocal

I'm not keen on much of the language they were using to describe licenses, like "permissive" and "viral" to describe the Apache License and the GPL, respectively. The GPL is a fine open source license that grants all the permissions needed by developer communities that adopt it. There is no sense in which it is not "permissive," so to use that term as an antonym to "copyleft" verges on abuse. I prefer to consider the degree to which open source licenses anticipate reciprocal behavior.

The Apache License grants all its permissions without any expectation that those passing on the software to others will also pass on the same freedoms they enjoy. Copyleft licenses anticipate that developers will share the freedoms they enjoy as well as sharing the source code. So I term licenses like the Apache License "non-reciprocal" and those like the GPL "reciprocal."

You Can Prove Anything with Statistics

One reason for the dispute is that there's a fairly clear commercial motivation behind the statistics that are being used to support deprecation of the GPL. They come from companies that sell assurance and analysis services for open source, and who promote "license compliance" as a risk for developers. They have a vested interest in developers fearing reciprocal licensing as they aim to monetize the amelioration of that fear. Look for pretty much any scare story about "license compliance" and it will somehow be connected with what one speaker called the "compliance-industrial complex."

The complexity arises from the subjectivity of phrases like "a decline in the GPL." Even when linked to data apparently supporting the statement, underneath is an assumption that the popularity of a license is demonstrated by the metric being cited, such as the number of new projects on GitHub using non-reciprocal licenses. Anti-GPL articles almost never justify why their popularity metric is valid.

Certainty & Safe Spaces

The open source license the project uses defines and guarantees the things over which those developers need certainty. All give the right to use the code for any purpose, to share it with anyone, and to make whatever changes they want. Beyond those essential freedoms, different communities need different certainties.

Communities working on code that is normally used directly, alone, and in its entirety – application software such as LibreOffice, for example – may well want their license to also guarantee reciprocal grants of the same rights to other developers. Most of the LibreOffice core developers work for companies that offer support, training, customization, and deployment. Because everyone who offers these services has to contribute their work as a license requirement, it's much less likely that a freeloader will be able to undermine their business.

For developers mixing ingredients from multiple origins – frameworks, components, libraries – reciprocal license requirements increase the uncertainty rather than decrease it. Their employer may be concerned about managing the different reciprocal duties of different licenses, such as the Eclipse Public License (EPL) and the GPL. For these developers, it is much simpler to use non-reciprocal licenses for their code, especially if that code is not directly monetized.

Horses for Courses

Neither of these approaches are the One True Way. Both have their place, and both have substantial, growing international acceptance. All the same, the growing acceptance of open source for corporate use – some would now say "dominance" – could easily appear as the only trend to those whose gaze is fixed in that direction. There is no doubt that the growth of corporate open source development has seen new code aimed towards it tend to use non-permissive licenses.

So perhaps the best way to view the subject is to note that the open source world has grown enormously. The use and support of the GPL has also grown with it, but new strengths have also emerged related to corporate adoption of open source. Deciding which is dominant may well be a matter for your ideological biases rather than an objective absolute! ■■■

Free Software in the Wild

Even outsiders feel the benefits.

BY ANDREW GREGORY

Dear readers, I have found a new job. This means I have to leave the house at roughly the same time every day, I have to communicate in real sentences rather than grunts, and occasionally I have a shower. I also get to talk to people who aren't obsessed with free software.

These poor souls will never know the dirty thrill of updating Arch without reading the wiki first. They'll never spend ages configuring KDE to get it just right. They'll never switch their operating system for a new one, just to see if they like it.

I'm also having to use Windows on a daily basis for the last time ever. I'm used to Linux Mint and OS X, so I know that it's rubbish. But the muggles at work know that Windows is rubbish even though they've never known anything different. It's really impressive.

One chap, for example, has installed a Chrome plugin to take screenshots. Another uses Pixlr to take screenshots. Another uses Gimp to take screenshots and edit images – adding text to photos, color correction, cropping, and whatnot. Of course, he gets mocked for using software with such a stupid name, but at least it's getting used. Gimp works, and that's all that matters. Not the license, not the stupid name, just the fact that it works.

Likewise with development. There's a big web project that's taking a little while longer than expected; of course, and it is being developed using proprietary software. So the head of another department has gone off-piste and developed his own version of the site using WordPress – "the great thing about Wordpress is that it's open source and built-in PHP, so you're not locked in. You can get any developer to write you a plugin." That was part of a real conversation with a real person!

Non-developers are choosing languages based on the license. Non-geeks are choosing free software because it offers features that are superior to paid-for alternatives. And it's still way too much trouble to take a screenshot on Windows. There's a breakthrough coming, and for one reason: the software we've been promoting for all this time is really, really good. Keep up the good work, everyone. ■■■

Going Beyond **Linux**

We all love Linux, but there are many other free software operating systems out there that are worth exploring.

BY MIKE SAUNDERS

How many open source operating systems (OSs) can you name? GNU/Linux is obviously the first thing that comes to mind, and if you've been around in the FOSS world for a while you've probably come across (or even tried) one of the BSD flavors as well: FreeBSD, OpenBSD, and NetBSD. These are all Unix-like OSs, each with their own sets of strengths and benefits, but there's a whole world of other OSs out there. Many of them aren't well known right now and have smaller communities compared with Linux, but some hold plenty of potential and could become big names in a few years.

So over the next few pages, we'll look at what's going on in the "alternative OS" scene. Best of all, because these OSs are all free software and open source, you can try them out after you've read about them. If you have an old PC sitting around doing nothing useful, you can try running these OSs on real hardware – or you can simply install a PC emulator such as VirtualBox or Qemu on your main machine and take them for a spin.

ReactOS

Thanks to Wine, it's possible to run many Windows applications on Linux – at least, predominantly older applications, and often with a few glitches here and there. Wine does an impressive job of intercepting Windows system calls and rewiring them for their Linux equivalents; as a compatibility layer; however, it can only do so much. Instead of using this layer on top of an unrelated OS like Linux, wouldn't it make more sense to basically write a Windows clone from the ground up?

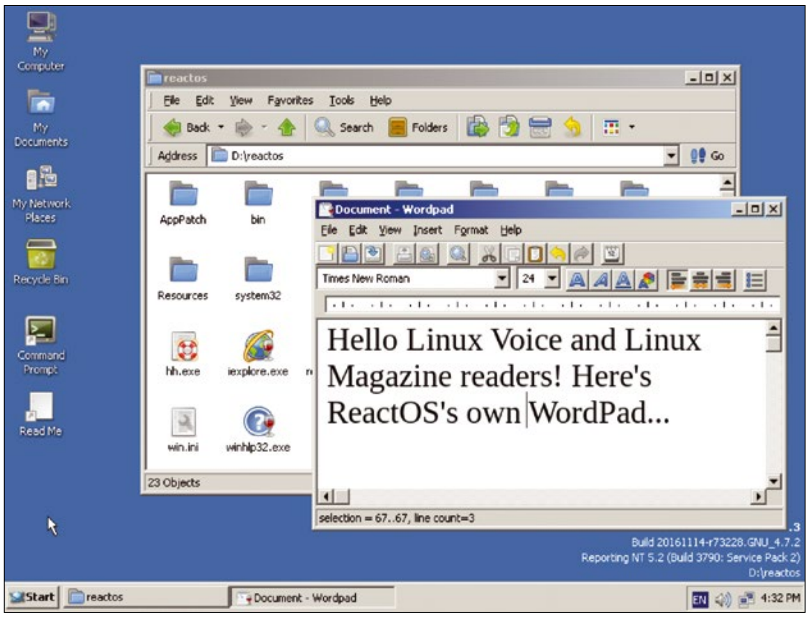
Well, that's what the team behind ReactOS [1] is trying to achieve. The project started in 1996 under the name FreeWin95, as an attempt to create a clone of Windows 95, but soon changed its name to ReactOS after the compatibility target was shifted to Windows NT. As with many FOSS projects, the early years were dominated by mailing list debates about design and implementation; not much real code was actually being written.

Indeed, it wasn't until ReactOS 0.2.0 in January 2004 that a usable version of the OS with a GUI was released. This was a major step forward for the project, and the ability to run a few (extremely simple) Windows applications won plenty of interest in the open source world. We remember trying it out at the time and being rather impressed – but also concerned about how Microsoft would respond. But more on that in a moment…

ReactOS is available as a live CD ISO image, so you can try it without installing. (Note: If you want to use it in VirtualBox, you'll need to tweak various settings as described in the wiki [2].) The OS boots up very swiftly and presents you with a desktop that look alarmingly similar to Windows 2000 (apart from the use of different icons). From here, it looks and feels very much like Windows of yesteryear and includes various common tools, such as a file manager, text editor, and registry editor (Figure 1). Although its appearance is rather retro, it's so much more polished than the 0.3.x releases we tried a few years ago.

So the big question is: How's the compatibility? It's very much a mixed bag, and ReactOS uses a lot of Wine libraries, so if something doesn't work well under Wine, you probably won't have much luck here either. It's worth trying your favorite apps

Figure 1: Here's ReactOS running its WordPad equivalent and Windows Explorer-like file manager.



Hello Linux Voice and Linux Magazine readers! Here's ReactOS's own WordPad...

selection = 67..67, line count=3

Build 20161114-r73228.GNU_4.7.2
Reporting NT 5.2 (Build 3790: Service Pack 2)
D:\reactos

and games and checking the Wine compatibility database [3] for tips and suggestions, though (Figure 2). Ultimately, the Windows API is a massive beast – full of undocumented calls and bugs, and changing with each new release – so chasing it is hard work. The ReactOS team has done a stellar job, but full Windows compatibility will probably always be a dream rather than reality.

What ReactOS really needs now is commercial backing. Look at the relationship between Wine and CrossOver, for example: The former is a totally free and open source project, while the latter is a commercial implementation backed by a company. That company (CodeWeavers) generates cash to hire developers and staff to support its commercial project, while contributing back to the upstream FOSS version (Wine). In the end, everyone wins, developers can put food on their tables, and we all get great new technology to play with.

So we'd love to see something similar happen with ReactOS. Many companies are still running hundreds or even thousands of PCs with old, unsupported versions of Windows. They don't want to upgrade to Windows 10 for various reasons (new hardware costs, compatibility issues, rampant spying in the OS), but ReactOS – if it had a commercial version with support contracts – could help them enormously. Indeed, it might make sense for a large company to sponsor ReactOS commercially to save costs "upgrading" to newer Windows releases, just like Sun Microsystems bought up StarOffice in 1999 because it was cheaper than installing Microsoft Office on 40,000+ PCs.

Of course, should ReactOS find commercial backing and more widespread popularity, there's the question: How would Microsoft respond? We've seen that Microsoft is taking steps to become more FOSS-friendly, at least superficially, so we don't think the company would simply try to shut down the ReactOS project overnight. But Microsoft's lawyers could potentially sow the seeds of doubt among potential ReactOS users (and customers), generating fear that the project could end up tangled up in legal issues, which would be enough to scare many companies away. Only time will tell.

Haiku

Whereas ReactOS is an attempt to clone Windows, Haiku uses BeOS as its inspiration. For those who've never heard of it, BeOS was a multimedia-centric OS that achieved limited popularity in the mid to late 1990s. Sporting SMP (multiprocessing support), pre-emptive multitasking, and a 64-bit journaling filesystem, BeOS certainly had its advantages over the reigning desktop OSs of the time, such as Windows 3.1/95 and classic Mac OS. Demo versions of BeOS were included on magazine cover discs – we remember being impressed by the ability to run multiple CPU-chomping applications simultaneously without the OS breaking into a sweat. Oh, and there was a nifty 3D teapot demo, as well.

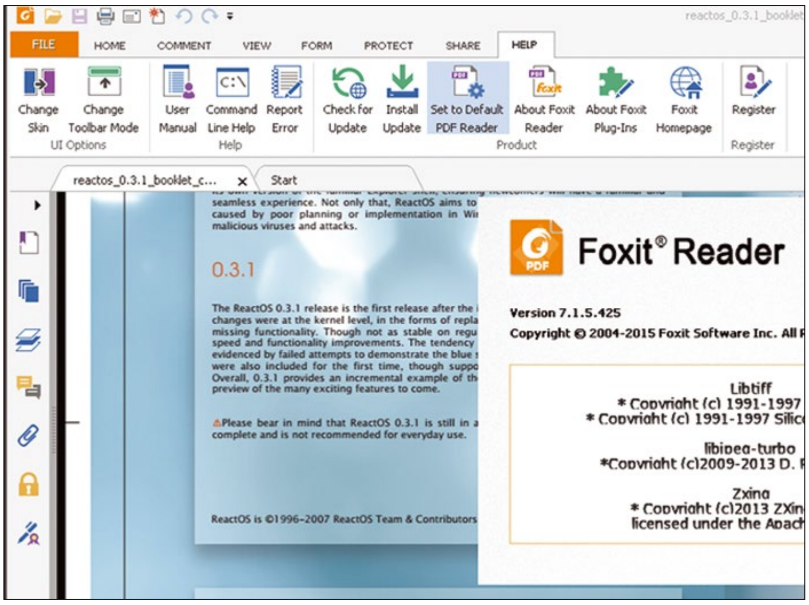
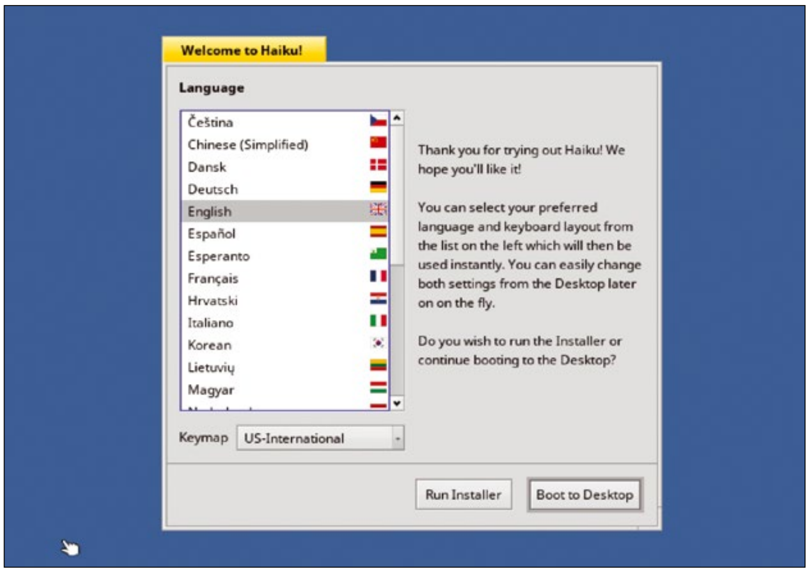


Figure 2: ReactOS can run various proprietary Windows apps like Foxit reader, but older apps work better.

So, what happened to BeOS? Ultimately, it failed to get any proper traction in the desktop market. Apple considered buying Be Inc., the company behind BeOS, to get a replacement for its aging OS (but ultimately went with NeXTSTEP and got Steve Jobs back as a result). In the early 2000s, Be Inc. took Microsoft to court, claiming that PC vendors had been strong-armed into shipping only Windows and no competing OSs. This case was eventually settled out of court; Be Inc. received $23 million from Microsoft, but the latter didn't have to admit any wrongdoing. And now, BeOS has faded into history.

Well, almost. As mentioned, Haiku [4] is an open source implementation of BeOS, and has been in development since 2001 (originally under the name OpenBeOS). Haiku aims to be both source and binary compatible with BeOS; source compatibility meaning that BeOS programs for which the source is available can be recompiled for Haiku with few changes, and binary compatibility meaning that original BeOS software can run flawlessly. This binary-compatibility requirement is causing development issues, though. Haiku needs to be built with an ancient version of GCC 2.95, released in 2001, although it's possible to compile the OS using a combined GCC 2/4 environment.

Haiku's last release – Alpha 4 – is now more than four years old and doesn't reflect the development activity in the project since then. So, to get the latest and greatest, it's worth getting a nightly release from the website [5]. Grab the "Anyboot" live CD ISO image and burn it to a CD-R for a real PC, or boot it up in VirtualBox or Qemu (Figure 3).



Thank you for trying out Haiku! We hope you'll like it!

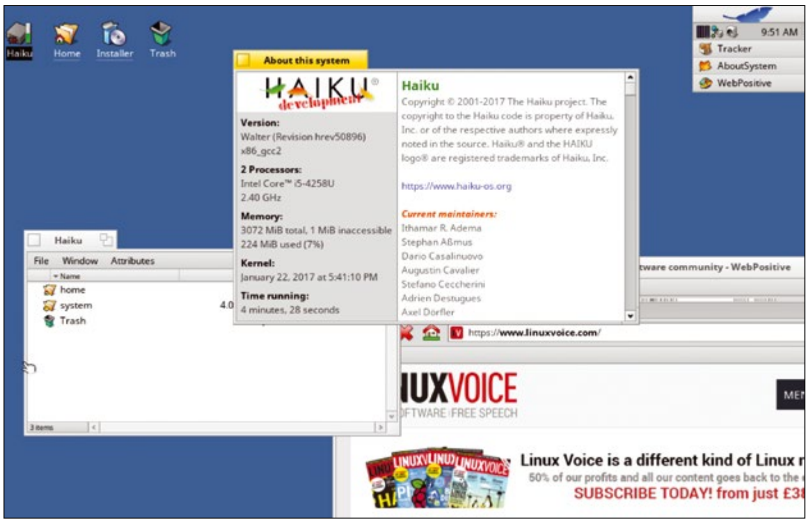
You can select your preferred language and keyboard layout from the list on the left which will then be used instantly. You can easily change both settings from the Desktop later on on the fly.

Do you wish to run the Installer or continue booting to the Desktop?

Figure 3: Haiku can be run in live mode from a CD or be installed directly to your hard drive.

Haiku starts up blazingly quickly. You're asked whether you want to install the OS to your hard drive or try out the desktop; choose the latter. You'll see straight away that Haiku's interface is rather spartan and minimal, with few graphical frills to woo you (Figure 4). In the top right corner, there's a leaf button, which opens up a Start-like menu for accessing programs and recent documents. Underneath that is a system-tray-esque area and then a list of running programs. If you find this horizontal design off-putting, click and drag the vertical dots in the system tray area to the top or bottom of the screen for a more familiar taskbar setup.

Click the leaf button and try browsing the included programs; you'll see that there's a mail client, web browser, media player, and other common tools. Under the Demos submenu, you'll find some multimedia showcases such as the aforementioned 3D teapot – although it runs like a dog in emulators! If you're on a real PC, though, you should be able to see how snappy and responsive Haiku is, even under load.


Haiku
Home
Installer
Trash
9:51 AM
Tracker
AboutSystem
WebPositive
About this system
File
Window
Attributes
Name
home
system
Trash
3 items
https://www.linuxvoice.com/
tware community - WebPositive
UXVOICE
FTWARE FREE SPEECH
Linux Voice is a different kind of Linux
50% of our profits and all our content goes back to the
SUBSCRIBE TODAY! from just £3

Version:
Walter (Revision hrev50896)
x86_gcc2
2 Processors:
Intel Core™ i5-4258U
2.40 GHz
Memory:
3072 MiB total, 1 MiB inaccessible
224 MiB used (7%)
Kernel:
January 22, 2017 at 5:41:10 PM
Time running:
4 minutes, 28 seconds

Haiku
Copyright © 2001-2017 The Haiku project. The copyright to the Haiku code is property of Haiku, Inc. or of the respective authors where expressly noted in the source. Haiku® and the HAIKU logo® are registered trademarks of Haiku, Inc.

https://www.haiku-os.org

Current maintainers:
Ithamar R. Adema
Stephan Aßmus
Dario Casalinuovo
Augustin Cavalier
Stefano Ceccherini
Adrien Destugues
Axel Dörfler

Figure 4: Haiku's desktop lacks wobbly windows and other shiny bits 'n' bobs, but it's elegant and fast.

Open up the terminal, and you may be surprised: Yes, it's Bash. But before you start treating it like a Linux or Unix system, bear in mind that Haiku (like BeOS) isn't a flavor of Unix and has its own design and filesystem layout. There are some similarities here and there, but Bash is included simply because it's a great shell, not because this is another wannabe Unix.

What does the future hold for Haiku? It's a shame that four years have passed since the last official alpha release, even though development is ongoing, and a small but dedicated team is beavering away to get R1 (1.0) out the door. Haiku's biggest problem, though, is the lack of native apps. Any Unix-like OS immediately has a wealth of software to run, and in the case of ReactOS, it has no shortage of potentially usable software as well.

There are ports of Linux/Unix apps to Haiku, but they often feel kludgy, and the OS needs modern, streamlined software that uses its API and features. Still, we're cautiously optimistic: If the Haiku team gets 1.0 released in the next 12 months, it could establish itself as a svelte and friendly desktop OS for low-spec machines, schools (e.g., web terminals), and other scenarios where you don't need the complexity of Linux or Windows, but just a clean and small desktop OS.

GNU Hurd

We're immensely grateful to the GNU project and Free Software Foundation for their contributions to the OS we use today – even though we rarely write GNU/Linux, opting for the shorter version in print. But as well as giving credit to GNU, the longer name also reminds us that Linux is technically just a kernel, and it's possible to run a GNU system with other kernels as well. Indeed, you can run the Debian GNU distro with a FreeBSD kernel in place of Linux [6] – although it's still rather experimental.

When Richard Stallman started his GNU project in the mid 1980s, he first worked on tools that could be used on other Unix-like OSs: a compiler, text editor, command-line tools, and so forth. Later, as the GNU project picked up pace, it started work on a kernel called Hurd, a doubly-recursive acronym: Hurd stands for Hird of Unix-Replacing Daemons, while Hird stands for Hurd of Interfaces Representing Depth. A nice geek joke, then, but not great for marketing purposes.

Anyway, Hurd's most prominent feature is that it's a microkernel, where many features are implemented as separate processes for maximum reliability. In the case of Linux, most features (hardware drivers, filesystem drivers, network protocols, etc.) run inside the kernel itself, so if one crashes, it can take down the whole system. With a microkernel like Hurd, though, many of those features run in their own process space, so they can be swapped out on the fly, and a crash won't

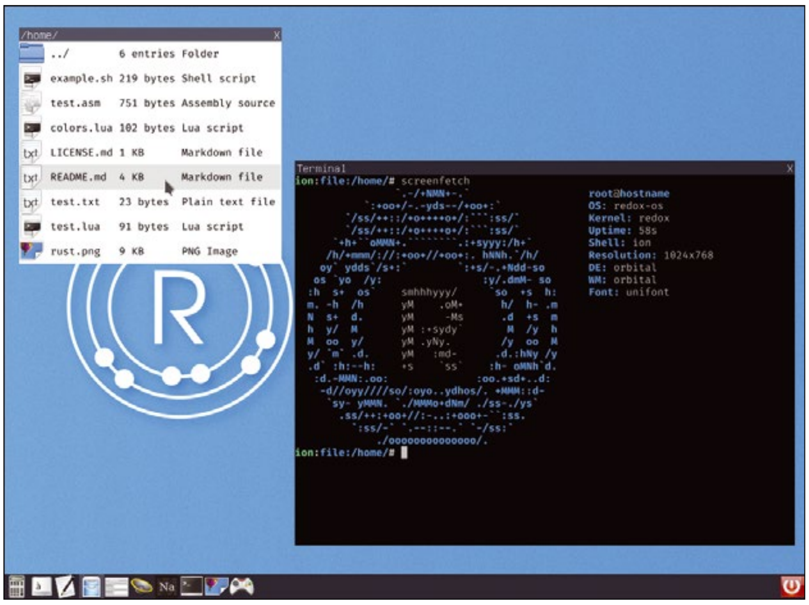
Redox: Unix Written in Rust

Although Hurd is unlikely to have a dazzling future, Redox OS [8] is a microkernel-based OS that's worth keeping an eye on. It's a Unix-like operating system written in the Rust programming language [9] (Figure 5). Rust was created by some Mozilla developers as a C-like language with extra features for security and concurrency and is being used in the development of Servo, Mozilla's experimental web browser layout engine.

As with Hurd, drivers in Redox run in their own separate spaces for reliability. A simple GUI called "Orbital" is available, along with a compatibility library for programs written in C. Although it's still relatively early days for the project, and it's more of an experiment rather than a production-ready OS, we're really happy to see the core concepts of Unix being re-implemented with new technologies and languages.

necessarily break the entire OS. This sounds great, but there is a downside: Switching between all of these processes takes more CPU time than when everything is lumped together in a big standalone kernel (see the "Redox: Unix Written in Rust" box for another microkernel option).

Partly because of this, and arguably because Linus Torvalds has been an excellent steward of the Linux kernel, Hurd has never taken off. You can run a Debian system with a Hurd kernel [7], but hardware support is very limited, and it's more of a tech demo than anything you'd want to use on a daily basis. Hurd may never become a serious competitor to Linux – and even Richard Stallman has admitted that it's not essential to focus developer effort on Hurd – but it's a fascinating part of Unix and FOSS history nonetheless. For more options, see the "Bring on the Clones" box. ■■■



```
/home/
../          6 entries Folder
example.sh 219 bytes Shell script
test.asm   751 bytes Assembly source
colors.lua 102 bytes Lua script
LICENSE.md 1 KB      Markdown file
README.md  4 KB      Markdown file
test.txt   23 bytes  Plain text file
test.lua   91 bytes  Lua script
rust.png   9 KB      PNG Image
```

```
Terminal
ion:file:/home/# screenfetch
                 `.-/+NMN+-.`                      root@hostname
            `:+oo+/-.-yds--/+oo+:`                 OS: redox-os
         `/ss/++::/+o++++o+/:```:ss/`              Kernel: redox
         `/ss/++::/+o++++o+/:```:ss/`              Uptime: 58s
       `+h+``oMMN+.````````.:+syyy:/h+`            Shell: ion
      /h/+mmm/://:+oo+//+oo+:. hNNh.`/h/           Resolution: 1024x768
     oy` ydds`/s+:`          `:+s/-.+Ndd-so        DE: orbital
    os `yo  /y:                 :y/.dmM- so        WM: orbital
   :h  s+  os`    smhhhyyy/     `so  +s  h:        Font: unifont
   m. -h  /h      yM    .oM+      h/  h- .m
   N  s+  d.      yM      -Ms     .d  +s  m
   h  y/  M       yM :+sydy`       M  /y  h
   M  oo  y/      yM .yNy.        /y  oo  M
   y/ `m` .d.     yM   :md-      .d.:hNy /y
   .d` :h:--h:    +s    `ss`    :h- oMNh`d.
    :d.-MMN:.oo:               :oo.+sd+..d:
     -d//oyy////so/:oyo..ydhos/. +MMM::d-
      `sy- yMMN. `./MMMo+dNm/ ./ss-./ys`
        .ss/++:+oo+//:-..:+ooo+-``:ss.
          `:ss/-` `.--::--.` `-/ss:`
              ./ooooooooooooooo/.
ion:file:/home/#
```

Figure 5: Redox is still in the early days of development, but shows that a proper OS can be developed in the Rust language.

Info

[1] ReactOS: *https://www.reactos.org*

[2] VirtualBox wiki: *http://reactos.de/wiki/VirtualBox*

[3] Wine compatibility database: *https://appdb.winehq.org/*

[4] Haiku: *https://www.haiku-os.org*

[5] Haiku nightly release: *https://download.haiku-os.org/nightly-images/x86_gcc2_hybrid/*

[6] Debian GNU/kFreeBSD: *https://www.debian.org/ports/kfreebsd-gnu/*

[7] Debian GNU/Hurd: *https://www.debian.org/ports/hurd/*

[8] Redox OS: *https://www.redox-os.org*

[9] Rust: *https://www.rust-lang.org*

[10] osFree: *http://www.osfree.org/*

[11] FreeVMS: *http://www.freevms.net*

Bring on the Clones

Name any popular piece of commercial, closed source software, and there's almost certainly a FOSS equivalent – even if it's in the very early stages of development. The same could be said about OSs: There's a FOSS clone of pretty much everything out there, although most of them see very little development activity or have been dormant for quite a while.

Take OS/2, for example, IBM's operating system that was once poised to budge Windows from the #1 slot on the desktop. OS/2 kind of lives on in the form of eComStation, but there's also a project to re-implement it as open source: osFree [10]. The goal here is to gradually replace all closed source components of OS/2 with open source equivalents, but although there were a few updates last year, development is still very slow.

Meanwhile, you may have heard of VMS or OpenVMS, a minicomputer-oriented OS originally from DEC that had its prime years in the late '70s and '80s. OpenVMS is still kicking around, albeit more in maintenance mode rather than as the OS we'll all be using tomorrow, and a FOSS project exists to recreate it as open source: FreeVMS [11]. Development appears to have stalled for a few years, but if you used to run VMS and know your way around it – and you fancy contributing to a FOSS project – maybe you can help!

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

MADDOG'S DOGHOUSE

"maddog" provides an update on Project Cauã, with new focus on helping university students and small businesses. BY JON "MADDOG" HALL

Changing Focus

Over the years ,I have written about Project Cauã [1], a project that I started more than 10 years ago. Project Cauã was first designed for Brazil, with its high-density cities but was always intended for spreading out to various parts of Latin America.

Project Cauã had many different business plans, all of which had various degrees of difficulty in getting started due to the use of hardware to leverage the plans. But, Project Cauã always had two main foci of generating jobs for people and making computers easier to use (not easy, just easier). While I am a staunch believer in Free and Open Source Software, I am also a pragmatist, and I realized that most business people use Microsoft or Apple products and that Project Cauã would have to address that issue.

In the past year, several people have been helping me change the focus of Project Cauã to one of helping university students afford university while gaining experience at running their own company and helping small and medium businesses utilize their computers better.

The Project Cauã Professionals (PCPs, as we call them) will become suppliers of front-line support, providing preventive maintenance so small business people can concentrate on running their business. The PCP will create a contract with five to six small businesses and work for them three to four hours each week making sure their computers are working properly. The PCP will therefore work about 24 hours a week, and the rest of the time they will be (hopefully) studying their university courses. Some students may spread their courses out over six years (instead of four) to maintain their grade levels, but since they are also running their own businesses working with computers, Project Cauã thinks this type of work is better than flipping hamburgers or waiting tables (typical jobs for university students).

While the main model of Project Cauã is for the student to be an independent business person, there are many liaisons that could also be formed with support companies who would like to have a local on-site person. Some of these support companies provide telephone support, but may have customers who do not work well with telephone support. Customers who employ a PCP might find a reduction in their support contracts with these companies, since the PCP might lower the time the support tech spends on the phone trying to eliminate the problem the customer is having.

Project Cauã will help the potential PCP learn how to run their company, how to apply for the proper licensing in their location, help with skeleton contracts, skeleton marketing brochures, determine their useful technical skills, and provide backup for "harder" technical problems, perhaps even sub-contracting issues to other PCPs with the necessary expertise.

One thing that was missing from all of the Project Cauã business plans until lately was the use of volunteer mentors to work with the PCPs, perhaps on a one-to-one level, to help make sure the PCP was successful. Mentors will come from the local business and technical community and will be people who have been successful in their own career – giving them a chance to "pay it back." As PCPs are successful in following the program, eventually they too will become mentors.

The structure of Project Cauã will be a top-level website that will be worldwide, which will explain general principles in multiple languages, followed by a second level that will concentrate on country issues, followed by a third level that will support local mentor groups that work with the individual PCPs.

All of the site will be available to be reviewed and studied by anyone. Anyone who wishes to ask questions will be able to create an account for free (to control users who might not act professionally), but there will be no charge for the login. People that wish to use the materials to create their own business will be free to do so.

To use the Project Cauã name and trademarks, however, a PCP will need to actually join Project Cauã, agree to follow a set of ethics and business practices, agree to a certain level of continuing education, and agree to general support of the project. There is no planned charge for this, although there may be charges for optional support mechanisms and products (professional insurance, group health plans, etc.) that will be made available to people reaching this level.

Project Cauã will be piloting in Ubatuba, Brazil for six months while the program is fine-tuned and evaluated, then piloted in several other cities around Brazil for another six months. In one year, Project Cauã will be opened up for other languages (particularly Spanish) and cultures (particularly Latin America). ■■■

Info

[1] Project Cauã: *http://projectcaua.org*

FAQ Mesa

Peek below the shiny interface of your machine and you'll probably find Mesa in the engine room. BY BEN EVERARD

Q I've decided to do my own Wikipedia research for this FAQ, so I'm way ahead of you. Mesa is the US-English term for a flat-topped mountain with steep sides. The word came to English from the Spanish for table, which is easy to remember as Table Mountain in Cape Town is a perfect example.

A Before you get too carried away, do you see the disambiguation link at the top of the Wikipedia page?

Q Err, yes.

A Click that and then scroll down to the technology section.

Q Ah, right. Got it. Mesa is a high-level programming language developed at Xerox PARC in Palo Alto. The name comes from a pun on it being a high-level programming language and a mesa being both high and level.

A That's a bit closer, but still not what we're talking about here. As well as being a flat hill and a programing language, Mesa is also the name of an open source 3D graphics library.

Q Ah, yes. In Wikipedia terms, you mean Mesa (computer graphics). According to this font of all knowledge, Mesa is an implementation of OpenGL and Vulkan. That, frankly, leaves me even more confused.

A 3D graphics have become so commonplace on everything from desktops to mobile phones that it's easy to forget that the computation behind them is hugely complex. The complexity really comes in two different ways: working out exactly what it is you want to display and doing the computation fast enough to display the scene at an acceptable frame rate. Both of these problems are solved through Mesa (and other software like it).

OpenGL (or Open Graphics Library) is a system for managing this complexity. It gives programmers a way of handling 3D graphics that is about as intuitive as a high-performance system for mathematical computation ever can be. Programmers can build objects in a 3D space (usually using triangles to map out the surface) and then define how this space is lit. The screen is then a window into this world (Figure 1).

What makes OpenGL particularly useful is that it's not just a way for programmers to understand how to structure their code, but a standard for hardware manufacturers to implement with their drivers. Essentially, it's a meeting point between the people who design and build 3D hardware and those that write 3D software.

Q Ah, right, so Mesa is what my graphics card drivers connect to?

A Yes, well sometimes. Your graphics card needs to speak OpenGL with the rest of your system, and Mesa is one

Figure 1: Mesa (and OpenGL) provides the graphical power for many desktop environments including Unity.

way of doing this. NVidia has their own software that works in a similar way to Mesa, but Intel and some AMD hardware use Mesa for their 3D graphical powers.

Q Graphics cards are often cited as a cause of non-free software on Linux systems. How does Mesa fit into this?

A Essentially, Mesa allows hardware manufacturers to work on a shared implementation of much of the graphics software. Companies such as AMD and Intel contribute to the development of the software, and as these are still in fierce competition with each other, the project can only succeed as open source software because neither party can close out the other. For supported hardware (and there's more every year), this means you can get open source graphics drivers and keep your machine as open as possible.

Q I've got an NVidia card on one of my machines, and you said that they don't use Mesa. Does this mean that I can't enjoy an open source graphics stack?

A Not necessarily. The official NVidia driver is proprietary, but there's also an open source driver for some NVidia hardware called Nouveau. This works with Mesa and provides graphics capabilities without the proprietary software. However, performance isn't as good as the official driver. It's fine for day-to-day usage, but may be a bit lacking if you want to play the latest games.

Q I don't have a graphics card. Do I still need Mesa?

A Probably, but this answer is a bit more complex than it first appears. The first thing to say is that it is possible to have graphics on Linux without Mesa (or any other implementation of OpenGL). However, even if you don't have a graphics card, you could still be using Mesa in a couple of ways. First, many modern CPUs come with graphics capabilities built in. This is true on phones, as well as desktops and laptops. In these cases, you'll still need Mesa or an equivalent to be able to use that power. Second, it's possible to use Mesa without dedicated 3D hardware. This works in exactly the same way as when running on graphics hardware, but the calculations are done on the CPU. This is sometimes (and erroneously in our opinion) called software rendering. However, this is much slower than running on a graphics card.

Q Why shouldn't you call that software rendering? It sounds like it's done in software.

A Well, yes, it is done in software, but all 3D rendering is done in software. When you play games using the latest-and-greatest chunk of silicon spitting out a few bazillion polygons on your super-duper-extra-even-more-high-definition screen at a refresh rate several times higher than the eye can see, all that 3D graphics is still being performed by software, it's just that the software is running on graphics chips.

Q Wait, if it's all in software anyway, what's the difference between rendering on the CPU and the graphics card, and why is the graphics card so much faster?

A Your CPU is designed to do almost anything that's thrown at it from database processing to web browsing to word processing. They're incredibly versatile, but this makes them big and bulky (in silicon chip terms). GPUs, on the other hand are designed to perform floating-point operations very efficiently. Not only that but the same floating-point operation on lots and lots of data. By focusing on just this type of processing, they can do it very quickly.

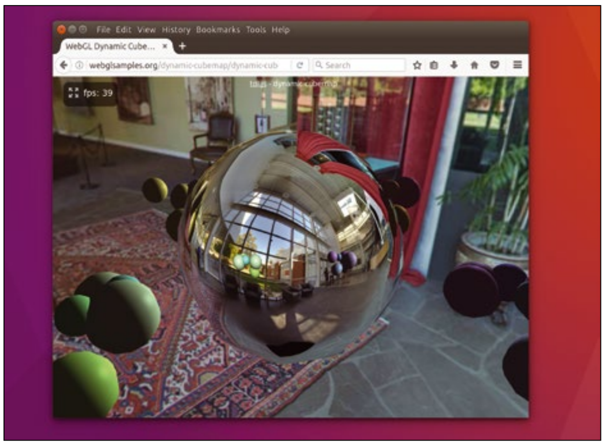

File Edit View History Bookmarks Tools Help
WebGL Dynamic Cube...
webglsamples.org/dynamic-cubemap/dynamic-cub
Search
fps: 39

Figure 2: Mesa can even supercharge your web browsing via WebGL. This example is a reflective sphere animation rendered in real time [1].

Q This is all very interesting, but I don't play games. Should I care about Mesa?

A Absolutely! The main reason being that many Linux desktop environments offload much of their graphics processing to the 3D hardware if it's available.

Q Ah yes, KDE's infamous wobbly windows. I'm not that fussed about the fancy 3D effects some desktops are peddling these days. So I can just not bother with Mesa, right?

A Nope! Even desktops that don't use gratuitous 3D effects often use Mesa to render the desktop because by using the GPU to do the graphics, it frees up the CPU for more important work. Even though the desktop could be rendered on the CPU, it can be much more efficient to render it on the GPU.

As well as graphics effects, video playback is often performed via Mesa (or similar software) to get smooth movement even on high definition screens (see Figure 2). It's one of those pieces of software that's vitally important (at least to desktop Linux users), but it's often forgotten about unless something goes wrong. ■■■

Info

[1] WebGL sample: *http://webglsamples.org/dynamic-cubemap/dynamic-cubemap.html*

Valentine Sinitsyn develops high-loaded services and teaches students completely unrelated subjects. He also has a KDE developer account that he's never really used.

CORE TECHNOLOGY

Ever wondered what processes are currently doing on your system? Linux has a capable mechanism to answer your questions.

BY VALENTINE SINITSYN

Process Tracing

Processes are, in general, units of isolation within a Unix system. This perhaps is the most important abstraction the kernel provides, because it implies that malicious or badly written programs can never affect proper ones. Isolation is the foundation of safety, but sometimes you want to turn it off.

Think of the interactive GNU Debugger (GDB) (Figure 1). You'd certainly want it to stop your code execution at specified points or execute it step-by-step, and it is hardly useful if it can't add watches or otherwise peek into the program being debugged; however, the debugger and the program it debugs are two different, isolated processes, so how could it ever happen?

You can't have rules without exceptions, and in Unix, a so-called process tracing mechanism called `ptrace()` answers this problem and has many other tricks up its sleeve.

Meet ptrace()

A gateway to process tracing in Linux is the `ptrace()` system call [2]. It allows one process, called a "tracer," to control execution, examine memory, modify CPU registers, and otherwise interfere with another process, known as the "tracee." Many popular tools, like the aforementioned GDB or the famous `strace` (system call tracer) command rely on `ptrace()` for their operation.

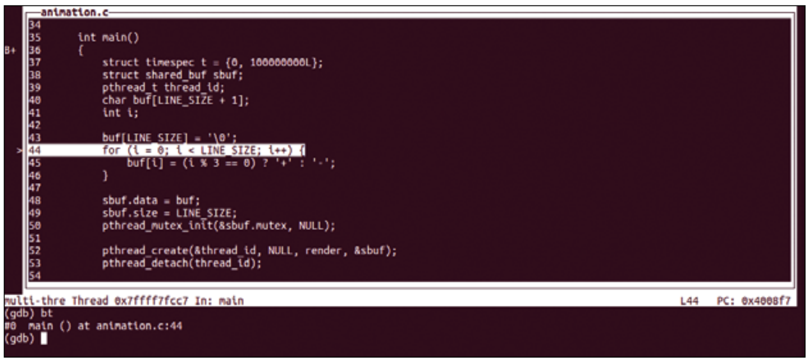
As many system calls with long histories (think `ioctl()`), `ptrace` is a multiplexer that does a myriad of things: attaches to the tracee and stops and resumes it, modifies memory and registers, updates signal handlers, and retrieves events, to name a few. Yet it accepts just four arguments: the type of request to issue, a process ID of the tracee, and two `void *` pointers, `addr` and `data`. The `addr` pointer usually conveys the address in the tracee's memory, and `data` is an exchange buffer in the tracer's address space. You get a result either as a `prtrace` return value (if it fits within `long`, which is typically 64 bits these days) or through the `data` buffer.

The kernel treats processes that are being traced somewhat differently from the others. When such a process receives a signal, the kernel stops it. This happens even if the tracee is set to ignore the signal. The kernel may also stop the tracee when its forks, calls `execve()` to run a new executable, or in fact does any system call. If single-step execution is desired, the kernel employs hardware-specific mechanisms (e.g., a TF flag in x86) to stop the tracee after each machine code instruction.

Before you can do useful things with `ptrace()`, you must attach a tracer to the tracee. This happens when you run:

```
gdb -p PROCESS_ID
```

Attachment is per thread, not per process as a whole, so in theory, you can attach a debugger to one thread, leaving others intact.



```
animation.c
    int main()
    {
        struct timespec t = {0, 100000000L};
        struct shared_buf sbuf;
        pthread_t thread_id;
        char buf[LINE_SIZE + 1];
        int i;

        buf[LINE_SIZE] = '\0';
        for (i = 0; i < LINE_SIZE; i++) {
            buf[i] = (i % 3 == 0) ? '+' : '-';
        }

        sbuf.data = buf;
        sbuf.size = LINE_SIZE;
        pthread_mutex_init(&sbuf.mutex, NULL);

        pthread_create(&thread_id, NULL, render, &sbuf);
        pthread_detach(thread_id);

multi-thre Thread 0x7ffff7fcc7 In: main                      L44   PC: 0x4008f7
(gdb) bt
#0  main () at animation.c:44
(gdb) 
```

Figure 1: GDB comes with an interactive text-mode user interface, too. To enter or leave, type C-x C-a (TUI key bindings [1]).

To attach to a tracee, you issue a `PTRACE_ATTACH` request and set the `pid` argument to the tracee's process ID. In this case, `addr` and `data` are unused:

```
#include <sys/ptrace.h>
ptrace(PTRACE_ATTACH, pid, 0, 0);
```

Originally, process tracing was permitted between any processes running under the same UID, unless a process was especially put in an undumpable state with a `prctl()` system call or (sometimes) via a `setuid` operation [3].

```
#include <sys/prctl.h>
prctl(PR_SET_DUMPABLE, 0, ...);
```

This solution wasn't perfect security-wise, so the Yama security module, which first appeared in Linux 3.4, introduced a `ptrace_scope` concept. A `sysctl` setting allows you to switch between classic behavior and restricted mode, in which case, parents can only trace their children. Alternatively, a process must declare some PID as a debugger, again with the `prctl()` system call:

```
prctl(PR_SET_PTRACER, pid, ...);
```

Desktop crash handlers (e.g., Dr. Konqi in KDE) often exploit this opportunity. Finally, you can enable admin-only attachment, in which case only root processes with `CAP_SYS_PTRACE` capability can act as tracers, or you can disable the feature altogether.

When you attach a tracer to the process, the kernel sends the tracee a `SIGSTOP` signal to stop it. If you don't want this to happen, use the `PTRACE_SEIZE` request, again introduced in Linux 3.4. To stop such a tracee at any later time, issue `PTRACE_INTERRUPT`.

You can also set up process tracing the other way around. In this case, the tracee issues a `PTRACE_TRACEME` request to have its parent start tracing itself. This sounds a bit counterintuitive and is hardly useful unless the parent is prepared to trace the child. A typical approach is to fork the tracer, issue a `PTRACE_TRACEME` from the fork, and then make the child run whatever program you want to trace.

When you no longer want to trace a process, issue a `PTRACE_DETACH` request. This is what the `detach` command in GDB does internally. The tracee must be stopped beforehand, usually when it gets a signal or issues a system call. Remember, you usually type Ctrl+C before detaching in GDB. Although this seems natural, now you know the real reason.

Some (Executable) Pseudocode

`ptrace()` is a Unix system call, so its native API is in C, which is okay for the low-level mechanism that `ptrace()` is, but for the sake of this article, I want something with fewer nuts and bolts involved. Luckily, such a tool exists. Python-ptrace [4] wraps all `ptrace()` goodies in a neat Python interface. Moreover, it includes a fully functional (yet relatively simple) debugger that you can dissect to learn `ptrace()` operation from a real-world example.

Python-ptrace uses ctypes to build a high-level ptrace API and is Python 2/3 compatible. It also includes faster (but not pure Python) bindings in a module called cptrace. Two high-level classes are provided, `ptrace.debugger.PtraceDebugger` and `ptrace.debugger.PtraceProcess`, which represents a process traced by a `PtraceDebugger`. Many `PtraceProcess` methods simply wrap corresponding `ptrace()` calls, but a few others are a bit more sophisticated.

The debugger is found in `gdb.py` (Figure 2), and it implements most basic GDB commands but ignores anything not directly related to the debugging itself. Thus, it won't load debugging symbols or show you the sources, which is a problem of its own (see the "Source-Level Debugging" box). In fact, it won't even show you the disassembly unless you have the diStorm disassembler [5] installed. All basic features are present and functional, though.

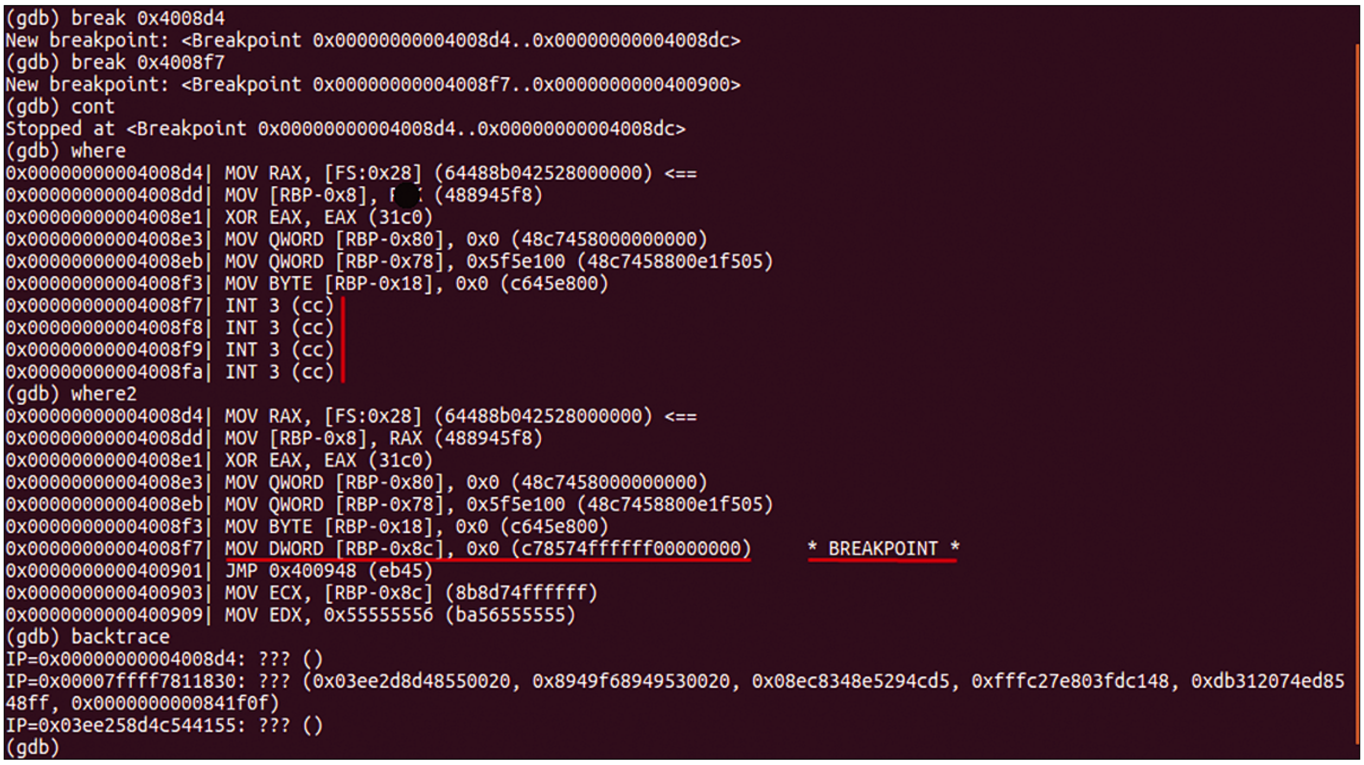
The tracer process often runs a sort of event loop. It instructs the kernel to schedule a tracee with `PTRACE_CONT` and then calls `waitpid()` to wait for the tracee's status change. When the tracee stops, `waitpid()` returns, and the tracer goes into action. It examines the `status` output argument to `waitpid()` with `WIFSTOPPED()`, `WIFSTOPSIG()`, and other macros, as usual, to learn what caused this stop. It can be an "ordinary signal" (e.g., `SIGINT`), which the tracer

Source-Level Debugging

Although it is possible to install breakpoints and read a program's memory (including code) with `ptrace()`, you only get machine instructions. However, programmers prefer to think in terms of source code lines. Mapping these to each other is a separate and non-trivial task that involves two major components: the sources and the debugging symbols. The debugging symbols link machine code locations to source code lines.

When you compile your code with `gcc -g` (or the equivalent clang option), debugging symbols are embedded within the resulting executable, which makes the binary bigger (much bigger) and is usually unwanted on production systems. So many distributions now ship symbols in separate packages, often with `-dbg` or `-debuginfo` suffixes. Symbols are usually installed under `/usr/lib/debug`, where debuggers such as GDB can find them and load at run time.

A de facto standard format to convey debugging information is DWARF (naturally, a companion term to ELF). The DWARF specification is several hundred pages in size, which hopefully gives an indication of the amount of work required to create a source-level debugger and provides a hint as to why `gdb.py` (with a thousand lines of Python code) doesn't step further than disassembly.



```
(gdb) break 0x4008d4
New breakpoint: <Breakpoint 0x00000000004008d4..0x00000000004008dc>
(gdb) break 0x4008f7
New breakpoint: <Breakpoint 0x00000000004008f7..0x0000000000400900>
(gdb) cont
Stopped at <Breakpoint 0x00000000004008d4..0x00000000004008dc>
(gdb) where
0x00000000004008d4| MOV RAX, [FS:0x28] (64488b042528000000) <==
0x00000000004008dd| MOV [RBP-0x8], RAX (488945f8)
0x00000000004008e1| XOR EAX, EAX (31c0)
0x00000000004008e3| MOV QWORD [RBP-0x80], 0x0 (48c7458000000000)
0x00000000004008eb| MOV QWORD [RBP-0x78], 0x5f5e100 (48c7458800e1f505)
0x00000000004008f3| MOV BYTE [RBP-0x18], 0x0 (c645e800)
0x00000000004008f7| INT 3 (cc)
0x00000000004008f8| INT 3 (cc)
0x00000000004008f9| INT 3 (cc)
0x00000000004008fa| INT 3 (cc)
(gdb) where2
0x00000000004008d4| MOV RAX, [FS:0x28] (64488b042528000000) <==
0x00000000004008dd| MOV [RBP-0x8], RAX (488945f8)
0x00000000004008e1| XOR EAX, EAX (31c0)
0x00000000004008e3| MOV QWORD [RBP-0x80], 0x0 (48c7458000000000)
0x00000000004008eb| MOV QWORD [RBP-0x78], 0x5f5e100 (48c7458800e1f505)
0x00000000004008f3| MOV BYTE [RBP-0x18], 0x0 (c645e800)
0x00000000004008f7| MOV DWORD [RBP-0x8c], 0x0 (c78574ffffff00000000)     * BREAKPOINT *
0x0000000000400901| JMP 0x400948 (eb45)
0x0000000000400903| MOV ECX, [RBP-0x8c] (8b8d74ffffff)
0x0000000000400909| MOV EDX, 0x55555556 (ba56555555)
(gdb) backtrace
IP=0x00000000004008d4: ??? ()
IP=0x00007ffff7811830: ??? (0x03ee2d8d48550020, 0x8949f68949530020, 0x08ec8348e5294cd5, 0xfffc27e803fdc148, 0xdb312074ed85
48ff, 0x00000000000841f0f)
IP=0x03ee258d4c544155: ??? ()
(gdb)
```

Figure 2: The gdb.py command-line debugger showing some disassembly (note the breakpoint instructions) and a backtrace. It works, but it isn't too informative.

probably injects back into the tracee, or it can be a `SIGTRAP`, which informs the tracer that something of interest has happened, such as a breakpoint hit or system call entered or returned.

After the tracer decides what to do with the signal, it issues a `PTRACE_CONT` request, telling the kernel which signal it wants to inject into the tracee (if any). Then the tracee resumes and the loop commences the next iteration.

Peeking into Memory

Imagine you run a program under `gdb.py`, and it stops for whatever reason. You want to examine which instructions it was executing before the stop and type the `where` command. The debugger prints some machine code or a raw hex dump (Figure 2) if you don't have diStorm. How has `gdp.py` achieved this?

The `where` command handler does some argument parsing and then calls the `PtraceProcess.dumpCode()` method, which retrieves an instruction pointer value (%rip register, if you are on x86-64) with `PtraceProcess.getInstrPointer()`. Next, it calls into a private `PtraceProcess._dumpCode()` method, which reads the tracee memory word-by-word with `PtraceProcess.readBytes()` and either passes it to the disassembler if it's present or just dumps hex data. Simplified versions of the `getInstrPointer()` method and the `readWord()` method, which reads a word of the tracee's memory, are shown in Listings 1 and 2, respectively.

Listing 1: Getting tracee's instruction pointer

```
CPU_INSTR_POINTER = "rip"

class PtraceProcess(object):
  def getInstrPointer(self):
    if HAS_PTRACE_GETREGS:
      regs = ptrace_getregs(self.pid)
      return getattr(regs, CPU_INSTR_POINTER)
    else:
      # Use PTRACE_PEEKUSER
```

Listing 2: Reading the tracee's memory

```
class PtraceProcess(object):
  def readWord(self, address):
    """Address have to be aligned!"""
    word = ptrace_peektext(self.pid, address)
    return word
```

As you can see, they rely on two `ptrace()` requests. `PTRACE_GETREGS` returns the tracee's general purpose registers, which are naturally architecture dependent. You can find the exact layout in the `sys/user.h` file in the standard C library. Python-ptrace re-implements it in the `ptrace.binding.linux_struct` module, which you may find more human-readable. The register file is usually several hundred bytes in size, so `ptrace` puts it where the `data` argument points.

Not all architectures recognize the `PTRACE_GETREGS` request, so python-ptrace introduces a workaround. If support is missing, it issues `PTRACE_PEEKUSER` to read memory in a so-called "user area." The exact layout of this area is defined again in `sys/user.h` (hence the name) as `struct user`. The user area stores various tracee process data (e.g., the code or stack starting address), which may aid debugging. The `addr` argument stores the offset within the structure. Because the `PTRACE_PEEKUSER` result is not longer than a machine word, `ptrace` conveys it as a return value and ignores the `data` argument.

`PTRACE_PEEKTEXT` has the same semantics as `PTRACE_PEEKUSER`, except it reads process text (or

Listing 3: Installing a breakpoint (simplified)

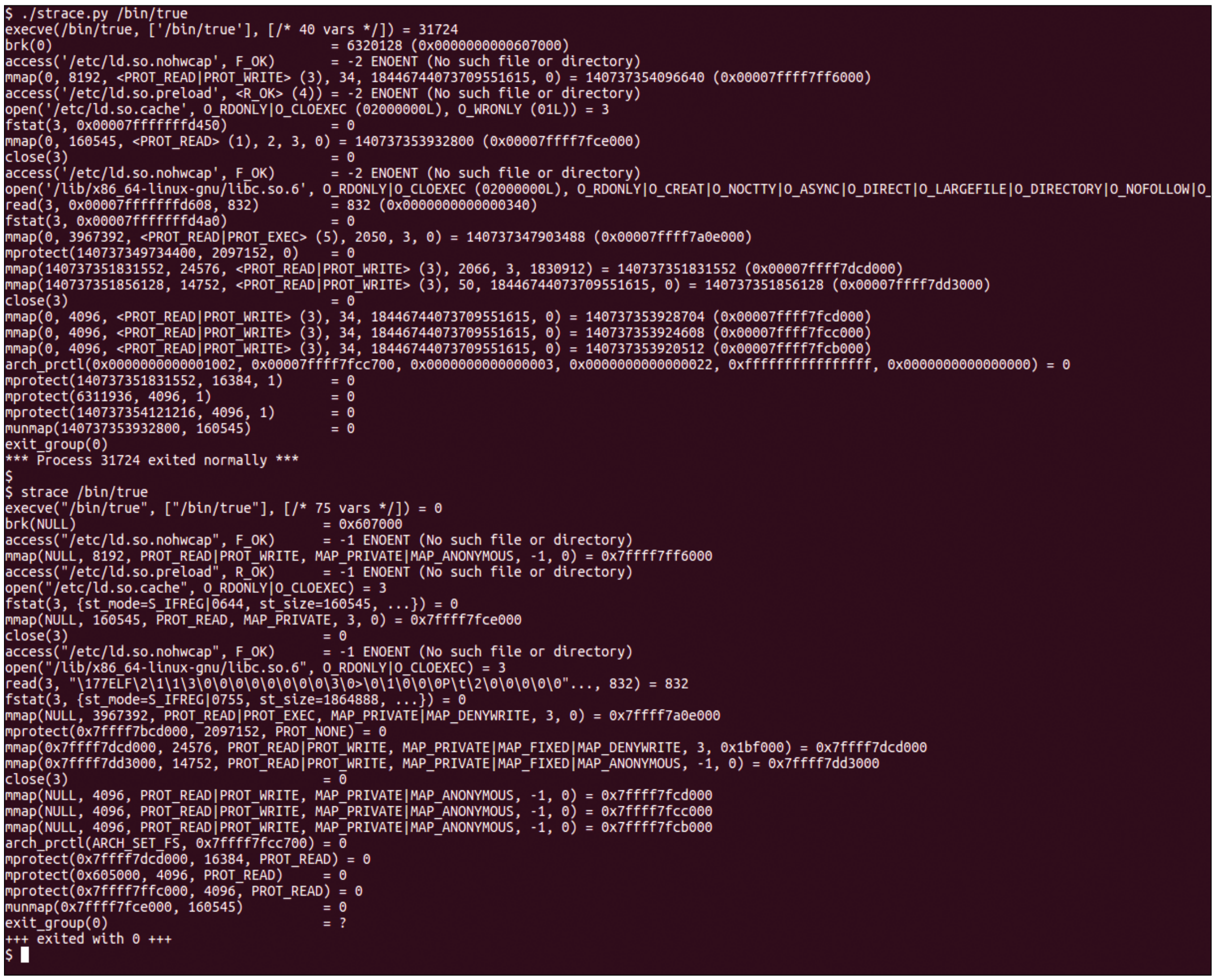
```
class Breakpoint(object):
  def __init__(self, process, address, size):
    self._installed = False
    self.process = ref(process)
    self.address = address
    self.size = size

    # Store instruction bytes
    self.old_bytes = process.readBytes(address, size)

    new_bytes = b("\xCC") * size
    process.writeBytes(address, new_bytes)
    self._installed = True
```

code), not the user area. Also, `PTRACE_PEEKDATA` reads the program data, but in Linux, code and data live in a single address space, so these are synonymous. `addr` represents a virtual address to read from, and `readBytes()` loops as many times as needed to read the amount required. It is a good idea to supply `PTRACE_PEEK*` requests (and `PTRACE_POKE` requests, which you'll see in a second), with a word-aligned `address` (i.e., it starts at an 8-byte boundary on x86-64).

Installing Breakpoints

Imagine now you want to discover what calls a buggy function. Normally, you'd set a breakpoint on it and print the backtrace when it fires. Although `gdb.py` can do this, because it knows nothing about your sources, you supply the breakpoint as a raw address in hex and receive the backtrace in much the same form (Figure 2).

Breakpoints come in two flavors – hardware and software – although `gdb.py` supports only software breakpoints. To install such a breakpoint, you modify the instruction at the location of



```
$ ./strace.py /bin/true
execve(/bin/true, ['/bin/true'], [/* 40 vars */]) = 31724
brk(0)                                  = 6320128 (0x0000000000607000)
access('/etc/ld.so.nohwcap', F_OK)      = -2 ENOENT (No such file or directory)
mmap(0, 8192, <PROT_READ|PROT_WRITE> (3), 34, 18446744073709551615, 0) = 140737354096640 (0x00007ffff7ff6000)
access('/etc/ld.so.preload', <R_OK> (4)) = -2 ENOENT (No such file or directory)
open('/etc/ld.so.cache', O_RDONLY|O_CLOEXEC (02000000L), O_WRONLY (01L)) = 3
fstat(3, 0x00007fffffffd450)            = 0
mmap(0, 160545, <PROT_READ> (1), 2, 3, 0) = 140737353932800 (0x00007ffff7fce000)
close(3)                                = 0
access('/etc/ld.so.nohwcap', F_OK)      = -2 ENOENT (No such file or directory)
open('/lib/x86_64-linux-gnu/libc.so.6', O_RDONLY|O_CLOEXEC (02000000L), O_RDONLY|O_CREAT|O_NOCTTY|O_ASYNC|O_DIRECT|O_LARGEFILE|O_DIRECTORY|O_NOFOLLOW|O_
read(3, 0x00007fffffffd608, 832)        = 832 (0x0000000000000340)
fstat(3, 0x00007fffffffd4a0)            = 0
mmap(0, 3967392, <PROT_READ|PROT_EXEC> (5), 2050, 3, 0) = 140737347903488 (0x00007ffff7a0e000)
mprotect(140737349734400, 2097152, 0)   = 0
mmap(140737351831552, 24576, <PROT_READ|PROT_WRITE> (3), 2066, 3, 1830912) = 140737351831552 (0x00007ffff7dcd000)
mmap(140737351856128, 14752, <PROT_READ|PROT_WRITE> (3), 50, 18446744073709551615, 0) = 140737351856128 (0x00007ffff7dd3000)
close(3)                                = 0
mmap(0, 4096, <PROT_READ|PROT_WRITE> (3), 34, 18446744073709551615, 0) = 140737353928704 (0x00007ffff7fcd000)
mmap(0, 4096, <PROT_READ|PROT_WRITE> (3), 34, 18446744073709551615, 0) = 140737353924608 (0x00007ffff7fcc000)
mmap(0, 4096, <PROT_READ|PROT_WRITE> (3), 34, 18446744073709551615, 0) = 140737353920512 (0x00007ffff7fcb000)
arch_prctl(0x0000000000001002, 0x00007ffff7fcc700, 0x0000000000000003, 0x0000000000000022, 0xffffffffffffffff, 0x0000000000000000) = 0
mprotect(140737351831552, 16384, 1)     = 0
mprotect(6311936, 4096, 1)              = 0
mprotect(140737354121216, 4096, 1)      = 0
munmap(140737353932800, 160545)         = 0
exit_group(0)
*** Process 31724 exited normally ***
$
$ strace /bin/true
execve("/bin/true", ["/bin/true"], [/* 75 vars */]) = 0
brk(NULL)                               = 0x607000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ffff7ff6000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=160545, ...}) = 0
mmap(NULL, 160545, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7ffff7fce000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\t\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=1864888, ...}) = 0
mmap(NULL, 3967392, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7ffff7a0e000
mprotect(0x7ffff7bcd000, 2097152, PROT_NONE) = 0
mmap(0x7ffff7dcd000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bf000) = 0x7ffff7dcd000
mmap(0x7ffff7dd3000, 14752, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7ffff7dd3000
close(3)                                = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ffff7fcd000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ffff7fcc000
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7ffff7fcb000
arch_prctl(ARCH_SET_FS, 0x7ffff7fcc700) = 0
mprotect(0x7ffff7dcd000, 16384, PROT_READ) = 0
mprotect(0x605000, 4096, PROT_READ)     = 0
mprotect(0x7ffff7ffc000, 4096, PROT_READ) = 0
munmap(0x7ffff7fce000, 160545)          = 0
exit_group(0)                           = ?
+++ exited with 0 +++
$ █
```

Figure 3: strace.py (top) is nearly indistinguishable from the original (bottom).

interest and put a "breakpoint trigger" instruction there instead. For x86, it's `INT 3` (opcode 0xCC). It triggers an interrupt, which the kernel handles and delivers as `SIGTRAP` to the tracer. The tracer then puts the original instruction back and resumes the tracee.

In the Python-ptrace implementation, a breakpoint is encapsulated in a class of its own, `ptrace.debugger.Breakpoint`. To create a breakpoint, you supply its constructor a tracee process and the address to put the breakpoint into. As you can see from Listing 3, it calls `readBytes()` first to read original instructions from the tracee's memory. Then, it calls `writeBytes()` to put `INT 3` at the `address`. `PtraceProcess.writeBytes()` translates the `PTRACE_POKETEXT` request, which copies a machine word from `data` to the `address`.

Tracing System Calls

For dessert, I'll briefly skim system call tracing. A de facto standard tool for this is strace (see the "Command of the Month" section), but Python-ptrace bundles its own pure Python version, `strace.py` (Figure 3).

When you execute

```
strace.py /usr/bin/<something>
```

`strace.py` runs the program you specify as a child and issues a `PTRACE_TRACEME` request to

Command of the Month: **strace**

Strace (Figure 4) is a venerable tool with noble SunOS origins that dates back to the early days of Linux. It appears you can teach an old dog some new tricks, though. Version 4.15, released around last Christmas, brings some "cool stuff" created in the course of the last year's Google's Summer of Code program.

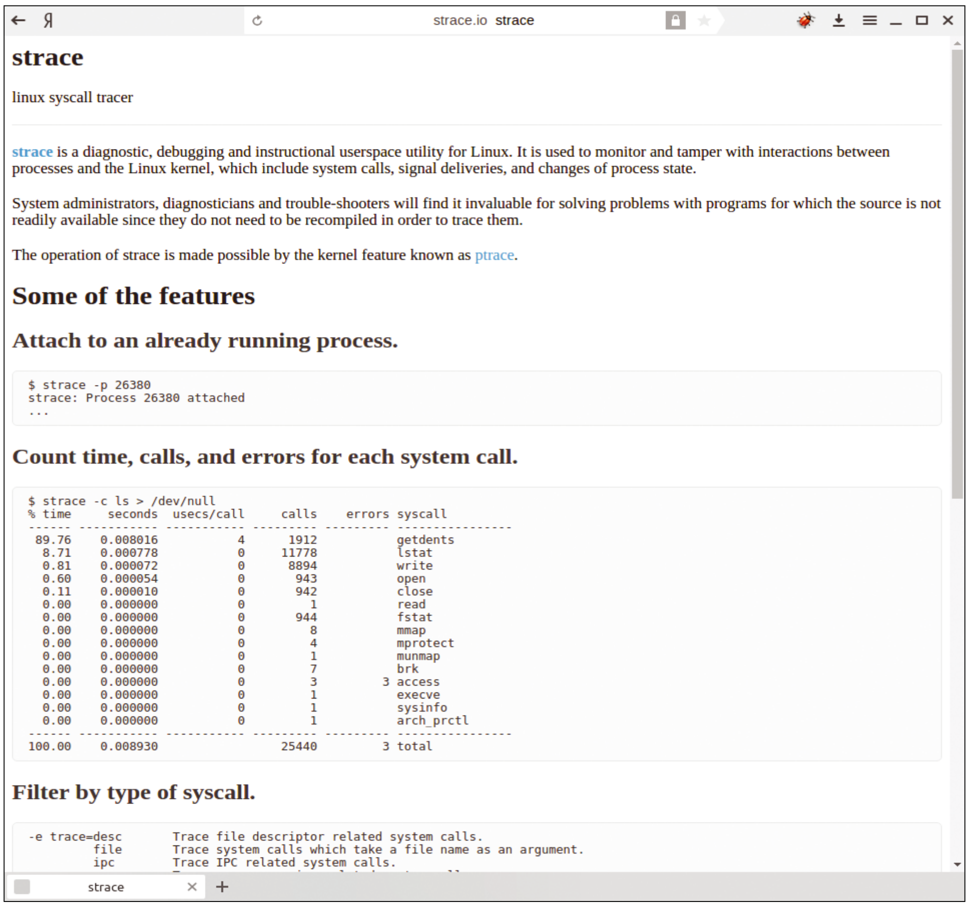
I'm speaking about fault injection. Handling error conditions in system programming can be tricky, so how do you check if you have accounted for all of them? It's relatively easy to test if an application misbehaves when it can't open a file, but how do you test for more convoluted things such as interrupted system calls or per-process limits that have been reached?

As of strace 4.15, you can instruct the tool to forge the return value for a selected system call. Consider the example

```
strace -e fault=open:error=ENOSPC:when=5+ ↩
  /some/program
```

which makes strace return an `ENOSPC` error for the fifth and subsequent `open()` system calls. According to the man page [6], this happens when a filesystem `open()` can hold no more files. This state isn't trivial to achieve in the real world, but strace makes testing for such tricky conditions a breeze.

Internally, strace sets the syscall number to `-1` before resuming a tracee. Because it is an invalid syscall number, the kernel replies with `ENOSYS`, but the `error` specification overrides this return value. The `when` tells strace when to inject the fault: You can do it for every matching system call or for the first invocation, for instance. A newer strace (unreleased at the time of writing) allows you to inject a signal alongside the error code. You already know this works because you supplied a `data` argument to a corresponding `PTRACE_SYSCALL` request. The only catch is that your distribution (if it's not Arch, you know) probably still ships the old strace. Packages for selected distributions are available through the openSUSE Build Service [7] ■■■



strace

linux syscall tracer

strace is a diagnostic, debugging and instructional userspace utility for Linux. It is used to monitor and tamper with interactions between processes and the Linux kernel, which include system calls, signal deliveries, and changes of process state.

System administrators, diagnosticians and trouble-shooters will find it invaluable for solving problems with programs for which the source is not readily available since they do not need to be recompiled in order to trace them.

The operation of strace is made possible by the kernel feature known as ptrace.

Some of the features

Attach to an already running process.

```
$ strace -p 26380
strace: Process 26380 attached
...
```

Count time, calls, and errors for each system call.

```
$ strace -c ls > /dev/null
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 89.76    0.008016           4      1912           getdents
  8.71    0.000778           0     11778           lstat
  0.81    0.000072           0      8894           write
  0.60    0.000054           0       943           open
  0.11    0.000010           0       942           close
  0.00    0.000000           0         1           read
  0.00    0.000000           0       944           fstat
  0.00    0.000000           0         8           mmap
  0.00    0.000000           0         4           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         7           brk
  0.00    0.000000           0         3         3 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         1           sysinfo
  0.00    0.000000           0         1           arch_prctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.008930                 25440         3 total
```

Filter by type of syscall.

```
-e trace=desc       Trace file descriptor related system calls.
         file       Trace system calls which take a file name as an argument.
         ipc        Trace IPC related system calls.
```

Figure 4: The good old strace tool recently got a facelift and the fashionable strace.io domain name.

make the parent (i.e., `strace.py` itself) trace it automatically. Then, a slightly modified version of the above "event loop" begins. It starts with a `PtraceProcess.syscall()`, which translates to a `PTRACE_SYSCALL` request. The kernel then stops the tracee on each syscall entry and exit with a `SIGTRAP` signal. This signal is somewhat oversubscribed, so to distinguish syscall traps from everything else, Linux (and some other Unices) introduces a `PTRACE_O_SYSGOOD` option. When it's enabled with a `PTRACE_SETOPTIONS` request (python-ptrace does this automatically if supported), the kernel delivers system call traps as `SIGTRAP | 0x80` – that is, with bit 7 in the signal number raised (| denotes bitwise OR).

You might wonder why it is important to notify the tracer both on syscall entry and exit. The assumption is that you use the first trap to decode the syscall arguments and the second to obtain the return value. Although it sounds simple, in fact it is rather hairy. Ptrace-python devotes a whole package, `ptrace.syscall`, for these purposes. Consult it if you are interested. In short, system calls are distinguished by their numbers, which are dependent on architecture and come through architecture-dependent registers. Where to get the arguments and return value also depends on the application binary interface (ABI). This is not to say you'd expect to see flag names such as `O_RDONLY` instead of raw numerical values.

When all this grunt work is finished, `strace.py` issues `PTRACE_SYSCALL` once again to run the tracee until the next syscall entry and exit, and the loop commences. The `addr` argument is unused, and `data` stores a signal to inject into the tracee when it's resumed. ■■■

Info

[1] TUI key bindings: *https://sourceware.org/gdb/current/onlinedocs/gdb/TUI-Keys.html#TUI-Keys*

[2] ptrace(2) man page: *http://man7.org/linux/man-pages/man2/ptrace.2.html*

[3] prctl(2) man page: *http://man7.org/linux/man-pages/man2/prctl.2.html*

[4] python-ptrace home: *http://python-ptrace.readthedocs.io/en/latest/*

[5] diStorm disassembler home: *https://github.com/gdabah/distorm*

[6] open(2) man page: *http://man7.org/linux/man-pages/man2/open.2.html*

[7] openSUSE Build Service page for strace: *https://build.opensuse.org/package/show/home:ldv_alt/strace/*

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

Productivity Suite

Calligra 3.0

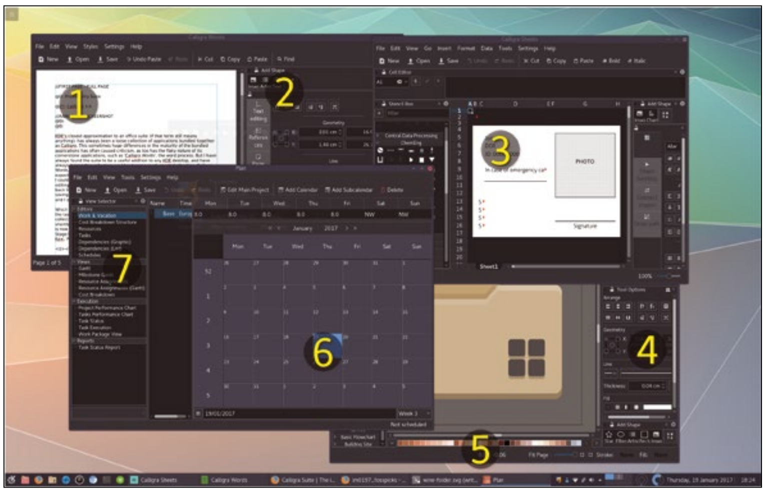
KDE's closest approximation to an office suite (if that term still means anything) has always been a loose collection of applications bundled together as Calligra. The occasionally huge difference in the maturity of the bundled applications has often caused criticism, as too has the flaky nature of its cornerstone applications like Calligra Words, the word processor. But I have always found the suite to be a useful addition to any KDE desktop and have hoped this collection would get some much needed attention.

Calligra Words, in particular, is one of my favorite KDE applications. As you might expect, it can be reconfigured endlessly. As a writer, this is important because it allows the user to remove the many distracting elements on display and leave the editing environment front and center, complete with word count. When pared back like this, Words was my favorite distraction-free word composer. But saving and loading issues, plus its adherence to ODT, often caused problems, and I eventually moved away.

That's why the release of Calligra 3.0 is so important. Not only has it given the development team an opportunity to rationalize the disparate nature of the collection, it has also meant addressing many of the old bugs and shortfalls. Krita, formerly of Calligra 2.x, is now big and successful enough to stand on its own, while Author, Flow, and Stage have been dropped, leaving the suite with five core applications: Karbon, the vector image editor; Kexi (released separately) the database; Plan, the project planner; Sheets, the spreadsheet; and the aforementioned Words.

The main feature for this release is that everything has been ported to Qt5 and KDE Frameworks 5, finally bringing the suite into the modern era. This was a major undertaking and is responsible for Calligra running so fast and feeling well integrated with the remainder of the KDE desktop. My favorite application is still the word processor, Words. It's never going to be as powerful as LibreOffice, but this means it's much easier to use and much quicker, too. In particular, in true KDE style, many different kinds of panels can be added and removed according to your needs allowing a level of visual configuration I've not seen in any other word processor. This feature is common to all of Calligra's applications, and it's something that KDE users will appreciate.

Karbon and Sheets are perhaps eclipsed by more famous alternatives, but they're both excellent options to have installed alongside Words. Karbon is perfect for editing of SVG icons, for instance, or creating a quick scalable diagram. Sheets is equally good at creating tables of numbers, and if you need a more ambitious solution, exporting as ODS works flawlessly. The project planning tool, Plan, is perhaps the weakest application here, but it's a valuable option simply because there's so little software of this nature on Linux, and it's perfectly suited to personal goals and projects. If you're a KDE user, or you've been looking for a lightweight and functional office suite that isn't LibreOffice, Calligra is definitely worth an installation.



1 Words. An excellent word processor with lovely font rendering and a distraction-free mode. **2 Dockers.** Each Calligra app can have its UI completely overhauled by adding and removing these panels. **3 Sheets.** A spreadsheet that's great for lots of things other than just numbers. **4 Karbon.** Not as complex as Inkscape but well suited for quick edits and diagrams. **5 KDE integration.** If you're a KDE user, Calligra feels part of the whole desktop. **6 Plan.** Project management applications like this are few and far between on Linux. **7 Views.** Windows can be as complex or as simple as you need them to be.

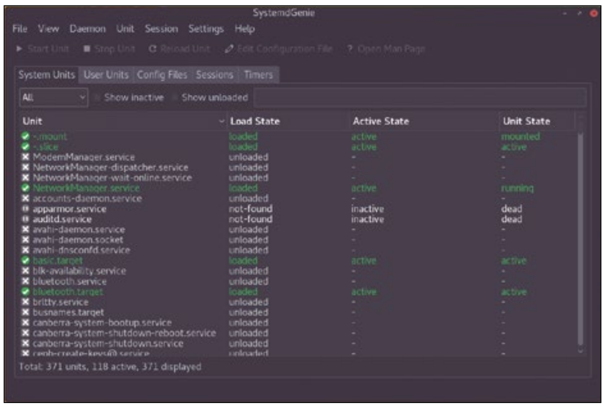
Project Website
https://www.calligra.org/

Systemd Editor

SystemdGenie

One of the most common complaints leveled against systemd is that it's difficult to understand and edit, especially when you're used to your system booting off a series of easily editable scripts. Which is why it's surprising there aren't more visual systemd editors for the entire system. There was one editor, though, and that was an extra KDE settings panel that could be installed. Unfortunately, because it was a KDE settings panel, you obviously needed to be running KDE. But the author has now updated the panel concept with a fully fledged Qt/KDE application, complete with menus, toolbars, and a new name. It's still KDE-based, but hopefully not riddled with huge dependencies if you want to run this on an alternative desktop.

SystemdGenie offers the same features as the old panel. Each of systemd's units is visible in a huge list that can be filtered, searched, and sorted. You can enable or disable a unit, start or stop a unit, and edit a unit's configuration file using SystemdGenie's integrated text editor. This is all simple enough to do on the command line if you know what you're doing, but doing the same within a GUI feels far more refined and will give some users that extra confidence to make changes without remembering systemd's often convoluted syntax. There are extra tabs for global configuration files, user sessions, and timers. Admittedly, none of this is going to make systemd any easier to use if you don't already have a good idea of what you're doing and want to accomplish, but the colored visual feedback and searching is very useful. I'd love to see extra features such as automatic version control or backup, but even when you don't change anything, SystemdGenie gives a great overview of what's running on your system.



```
Unit                                        Load State   Active State   Unit State
-.mount                                     loaded       active         mounted
-.slice                                     loaded       active         active
ModemManager.service                        unloaded     -              -
NetworkManager-dispatcher.service           unloaded     -              -
NetworkManager-wait-online.service          unloaded     -              -
NetworkManager.service                      loaded       active         running
accounts-daemon.service                     unloaded     -              -
apparmor.service                            not-found    inactive       dead
auditd.service                              not-found    inactive       dead
avahi-daemon.service                        unloaded     -              -
avahi-daemon.socket                         unloaded     -              -
avahi-dnsconfd.service                      unloaded     -              -
basic.target                                loaded       active         active
blk-availability.service                    unloaded     -              -
bluetooth.service                           unloaded     -              -
bluetooth.target                            loaded       active         active
brltty.service                              unloaded     -              -
busnames.target                             unloaded     -              -
canberra-system-bootup.service              unloaded     -              -
canberra-system-shutdown-reboot.service     unloaded     -              -
canberra-system-shutdown.service            unloaded     -              -
ceph-create-keys@.service                   unloaded     -              -
Total: 371 units, 118 active, 371 displayed
```

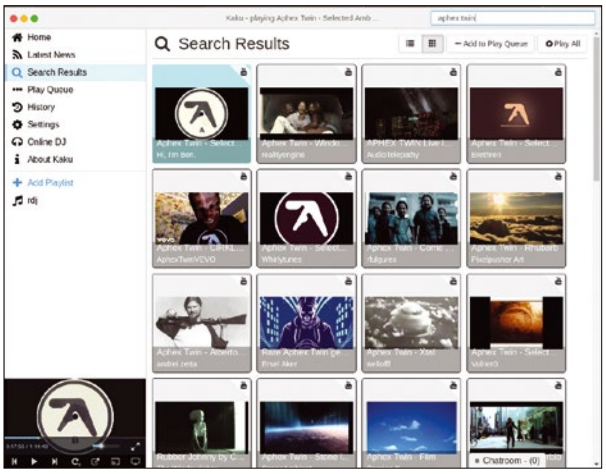
If you don't like systemd, you might like SystemdGenie.

Project Website
https://cgit.kde.org/systemdgenie.git/

Music Player

Kaku 1.8.5

Plenty of audio players attempt to make online music as accessible as local music files. But with recent releases, Kaku has become one of the few to make the concept work. It succeeds because it's simple and easy to use, requiring no configuration tinkering or weird dependencies. It's even available as an AppImage, which means it can be run without installing anything if you're prepared to accept a static binary running from an untrusted source – the latest release will also update itself automatically. When run, Kaku looks like an old version of Apple's iTunes, complete with colored window control icons and a complete disrespect for your chosen window manager, but it works brilliantly. Enter a search term and click play on one of the results. You'll hopefully hear what you were looking for. The default search destination is YouTube, but you can change this to SoundCloud, Vimeo, or All, which aggregates the results from all three. You can also choose which country code to use to rank the results, and whether you want to prioritize results from a video or audio-only search. As this is an audio player, video isn't supported, but the audio is played automatically and transparently, regardless of whether Kaku is ripping the audio or not. This is a fantastic feature considering the kind of rare or live material you find on sites like YouTube, where it can often be the only source for such material. You often forget this music is streaming from predominately video sources. You can then use any results to populate your own playlists and play queue, just as you would with your own music or a service like Spotify. The only downside is that those sources can change or disappear, just like they can with a browser, except Kaku is much nicer to use.



Kaku is much better than its name implies.

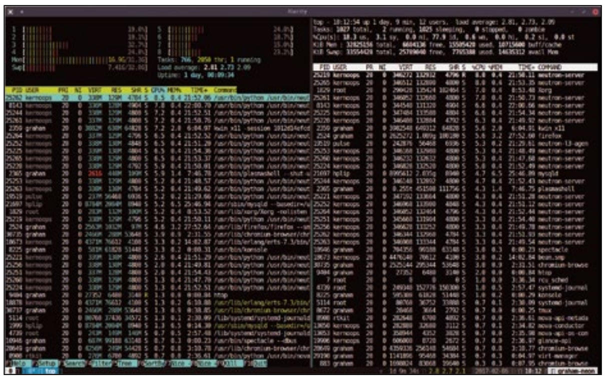
Project Website
http://kaku.rocks/

Superfast Terminal Emulator

Alacritty 0.1

We don't often think of an actual terminal emulator being either fast or slow. The commands we run in them are nearly always the culprits when it comes to clogging up your system, but that doesn't stop developers from trying to create the fastest terminal in the land. This is what Alacritty is – a terminal emulator that promises "blazing fast" speed. It's even brave enough to claim to be the fastest terminal emulator available, and I certainly found it faster than anything I had installed. But the clever part is that this speed comes from a part of your system you're unlikely to be fully utilizing while typing on the command line, and that's your GPU. Alacritty uses OpenGL directly to harness the power of your graphics hardware and is capable of rendering around 500 frames per second with a high-resolution screen full of text, according to the developer.

The motivation behind developing Alacritty is to give terminal-intensive applications, like vim or tmux, a much needed performance boost, especially when running on high-resolution, high-pixel density displays. It draws a new frame whenever anything changes within the terminal, and not when the terminal is sitting idle, and you notice this whenever you deal with screenfuls of scrolling text. What's even more remarkable is that Alacritty is written in Rust, a project under the auspices of Mozilla Research. That a new and modern language can perform as well as traditionally speedy but dangerous languages like C or C++ is a sure sign of the future in terms of both code security and validity. It also helps that the terminal looks so good, thanks to the sub-pixel anti-aliasing, which is presumably coming for free with OpenGL. If you use a terminal all the time, as many of us are now doing, it's only when you use a super-fast terminal like Alacritty that you realize you were using an old one.


You don't normally use the GPU while using the terminal, so why not use a terminal that uses the GPU?

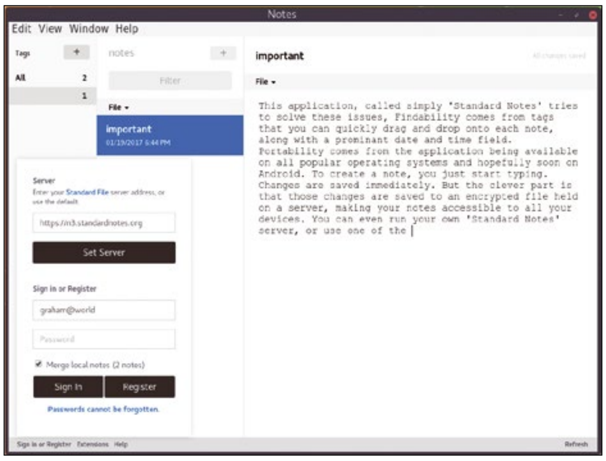
Project Website
http://blog.jwilm.io/announcing-alacritty/

Note Taker

Standard Notes

Note taking is never given the attention it deserves. It's an important skill that many of us neglect. But it's not just the notes themselves that we neglect, it's the medium we use to take them. Many of us still seem to take notes using either a pen and paper, for example, or by emailing ourselves or creating random files in something like Google Drive. But these methods, like others, suffer from the same problems of findability, portability, and security. You often need to take notes at short notice, and equally, store those notes away. This makes them hard to find and hard to contextualize later when you need them.

This application, called simply Standard Notes, tries to solve these issues. Findability comes from tags that you can quickly drag and drop onto each note, along with a prominent date and time field. Portability comes from the application's being available on all popular operating systems and hopefully soon on Android. To create a note, you just start typing. Changes are saved immediately. But the clever part is that those changes are saved to an encrypted file held on a server, making your notes accessible to all your devices. You can run your own Standard Notes server, or use one of the publicly accessible servers provided by the developers. Notes are encrypted with your own password and can't be retrieved if you happen to forget it, but that's a privacy feature rather than an inconvenience. You can also install extensions, and there are currently two: one for syncing your notes with a Dropbox account and another for turning your notes into a personal blog. Separate tools can also be used to export notes to Evernote. But even without these features, a simple note-taking application that silently and securely saves your notes to the cloud is still very useful.



important

```
This application, called simply 'Standard Notes' tries
to solve these issues, Findability comes from tags
that you can quickly drag and drop onto each note,
along with a prominant date and time field.
Portability comes from the application being available
on all popular operating systems and hopefully soon on
Android. To create a note, you just start typing.
Changes are saved immediately. But the clever part is
that those changes are saved to an encrypted file held
on a server, making your notes accessible to all your
devices. You can even run your own 'Standard Notes'
server, or use one of the |
```

Enter your Standard File server address, or use the default.
https://n3.standardnotes.org
graham@world
Password
✓ Merge local notes (2 notes)

Standard Notes is a simple note taking application that saves your notes to the cloud.

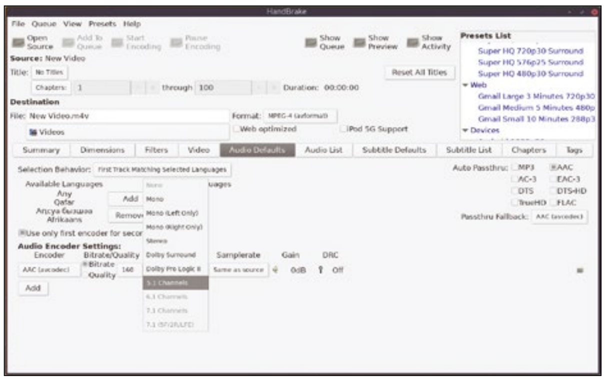
Project Website
https://standardnotes.org

Media Transcoder

Handbrake 1.0

It's taken 13 years to get to a 1.0 release, but there can't be many Linux users who haven't heard of Handbrake. It's a brilliant utility that converts media, usually videos, from one format to another. Back in the dark days of Linux, back when Internet Explorer was considered a web standard, Handbrake was often the only way we could play formats created by recording equipment, or even purchased online. It was also fabulous at archiving video off optical media, or converting it for use with MythTV or Kodi, and it was one of the only ways to get Linux-generated video and squeeze it into a format Apple's products could use. It's still great for all these things, and what makes Handbrake so powerful is that you don't need to be an FFmpeg expert. Thanks to its use of presets that act as target profiles for specific uses, it's easy to create a file that will play on Android at 1920x1200/60 FPS, for instance, or on a Chromecast, or even as an autoplaying Gmail attachment. Select a source, select a preset, and start encoding.

The big update to accompany this release is the addition of new online documentation. At the moment, it feels more like a series of articles than a comprehensive overview of the software, but it will still be incredibly useful for beginners. There are excellent documents that walk you through the encoding process, for instance, which will help you get your head around the sometimes perfunctory user interface.


One of the best uses for Handbrake is to extract the 5.1 surround audio from a music DVD or performance.

But the best thing about Handbrake is that it works without crashing or breaking compatibility. It does a great job at hiding the transcoding complexity and producing high-quality conversions, which is remarkable when you consider how the codec/DRM/video landscape has changed over the past 13 years.

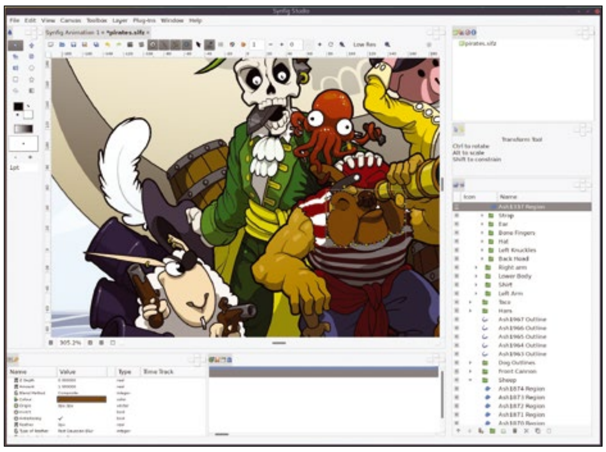
Project Website
https://handbrake.fr

Animation Software

Synfig Studio 1.2.0

There have been some significant updates to some significant projects recently, and the release of Synfig Studio 1.2 is one of those. Synfig is an amazing vector graphics editor with a focus on cartoon-like images, that's also an animation tool that can turn those images into fully fledged cartoons. It's brilliant. And complicated. And requires not only proficiency with the application itself but a huge dose of talent, too. Luckily, lots of its users seem to have met these requirements, and many of the results I've seen are easily worth the effort. That all this comes from a piece of open source software is even more amazing, especially when compared with spending money on commercial alternatives.

This version is the result of 16 months' work and comes off the back of a modest crowdfunding campaign that was completed in the middle of 2014. Training courses and a Patreon page have also helped with development, and the main improvement for this release is a completely rewritten rendering engine. The main problem with Synfig has always been its performance. This is because those vectors preserve the original resolution and need to be maintained even in very complex scenes and animations – they're the reason why the output is always so perfectly crisp. The new rendering engine is noticeably faster; it's now multithreaded and less memory demanding. The other major feature is integration with Papagayo lip sync, a tool designed to help line up mouth shapes with the recorded sound of the actors within the animation. Of course, this may cause some anime titles to lose a bit of their crudely translated charm, but overall it's going to help Synfig and its users create much more professional results.


Conquer YouTube with your own cleverly cynical parodies of modern politics.

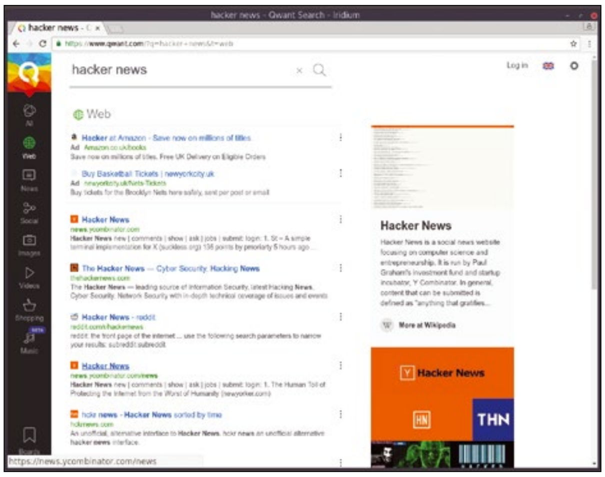
Project Website
https://morevnaproject.org/

Web Browser

Iridium

There are many things to love about Google's Chrome browser. It works with almost every site; it's about the best performing, full-fledged browser I've used; and it's secure, thanks to a constant stream of updates. But there's one thing you can't quite trust in Chrome, and that's its privacy credentials. Fortunately, like Android, the core of Google Chrome is open source and can be merged and built into other projects. Chromium, for example, is an open source version of Google Chrome. But this also means the source code can be taken, changed, and rolled into a completely new project, which is exactly what the Iridium browser is.

Iridium is a web browser for modern times. Every modification made to the open source code base has been made to enhance the privacy of the user, and it does this without sacrificing Chrome's speed or ease of use. The project's GitHub page lists the changes made to the Chromium code, and these include always sending the Do Not Track header, changed default search, blocked third-party cookies, disabled battery status API, no password saving, and many more. Additionally, many of Google's embedded services have been disabled, too. These include cloud printing, embedded hot words, translation features, and automatic update check. All this means you get the best of Chrome, offering the latest web technologies such as WebRTC, without many of the features and services that could compromise your privacy. And although many of these options can be changed within both Chrome and Chromium, it's reassuring to know a skilled set of developers have created a set of baked-in defaults that should help the majority of us reclaim our privacy.


hacker news - Qwant Search - Iridium
https://www.qwant.com/?q=hacker+news&t=web
hacker news
Web
Hacker News
Hacker News
THN
https://news.ycombinator.com/news

All the things you like about Google's browser, without the Google.

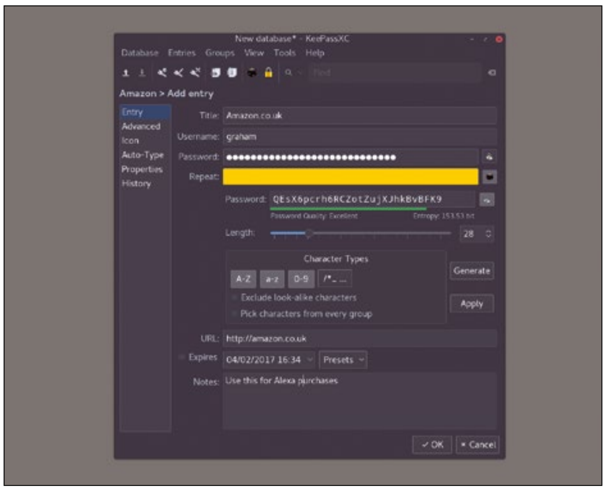
Project Website
https://iridiumbrowser.de/

Password Manager

KeePassXC 2.1.0

Password security is still a nightmare, and no one has really come up with a better solution than having a strong, randomly generated password for each site or service you use. This of course makes remembering them impossible, which is why password managers exists. But putting all your passwords into the same virtual pocket is the encryption equivalent to putting all your eggs in one basket. One mistake and you've lost everything. This is why open source password managers are so important. My favorite is a tool called "pass," which uses a normal filesystem layout of folders and files, alongside your GnuPG key, to secure your information in plain site. But another popular option is KeePassX, a GPL Linux port of the same tool on Windows and OS X.

Unfortunately, the original port of KeePassX has stalled, with lots of written and proposed new features and bug fixes being kept from the main project's repository – there were 69 opened pull requests without maintainer comment waiting to be merged, for example. This feature stalling has led to KeePassX to be forked into a new project called KeePassXC, which now includes all the stalled features and fixes along with a renewed vigor to provide users with the best password manager for your systems. The new application looks the same and has the same database management with hooks to your browser. It also opens and saves a 2.x compatible database, so you can switch between the two applications if you've not yet got the confidence with the new version. But because the new application includes a fantastic password generator, complete with strength meter for your own, favicons for website entries, and the ability to merge databases that can reload when there's a change, I don't see any immediate reason to go back to the old pre-fork version.



Amazon > Add entry

Title: Amazon.co.uk
Username: graham
Password: ●●●●●●●●●●●●●●●●●●●●●●●●●●●●
Repeat:
Password: QEsX6pcrh6RCZotZujXJhkBvBFK9
Password Quality: Excellent Entropy: 153.53 bit
Length: 28
Exclude look-alike characters
Pick characters from every group
URL: http://amazon.co.uk
Expires 04/02/2017 16:34
Notes: Use this for Alexa purchases

The excellent KeePassX has forked, adding a C to its name alongside some excellent new features.

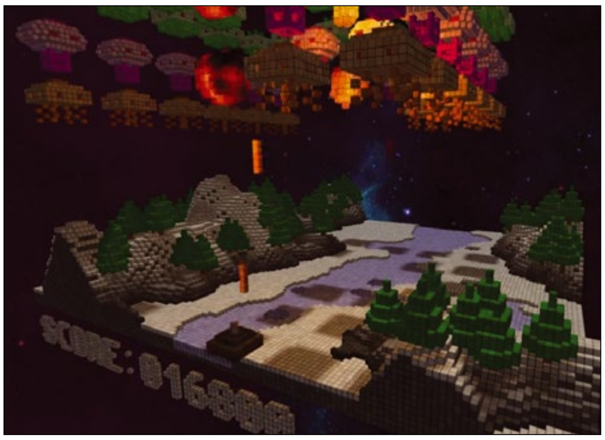
Project Website
https://keepassxc.org

Space Invaders

Voxeliens

This is a game that was originally proprietary and first available in 2012. Because of its nature and the way the development studio works, it has never been fully completed and never made it to a 1.0 release. In recognition of this, and the unlikelihood of the game being further developed by the original team, Voxeliens has been released as open source under an MIT license. This is great news for us, and great news for the project in general as I'd obviously much rather see its code being used than languishing in an unloved private repository.

The game itself is a modern interpretation of the ancient Space Invaders arcade game. The modern interpretation actually has more to do with the post-Minecraft world of 2012 than something truly modern like virtual reality, because like many other games of the 2012, it's a recreation of an old concept using voxels. Voxels, as used in Minecraft, are 3D pixels where the squares are not only positioned in an X and Y coordinate, they're also positioned along a Z axis for depth. In Voxeliens, you skim across a static voxel terrain while 3D pixelated aliens attempt to shoot you from above. It's just like Space Invaders with a Z axis and is equally playable. The 8-bit retro gaming sounds and quick action help recreate the old atmosphere, and even though you're moving across a 3D plane and can change the angle, the gameplay is fundamentally the same as the old classic. Which is a good thing. The aesthetics are lovely, and it's genuinely great news that all this is now available for use within your own projects, or for your own study.



Study voxel graphics and great gameplay with the open source release of an old game.

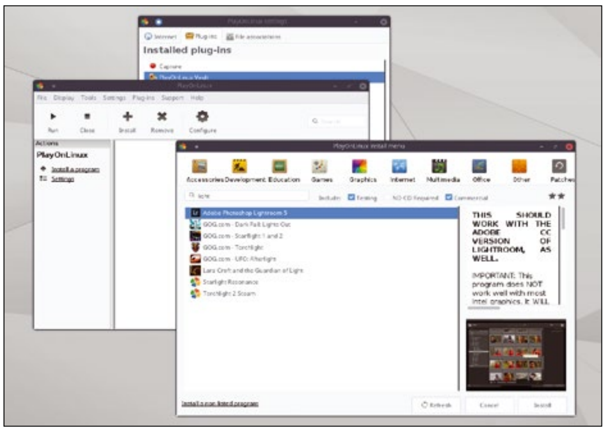
Project Website
https://bitbucket.org/volumesoffun/voxeliens

Windows Emulator

Wine 2.1

There have been a couple of major updates to this incredible project recently. The first was the milestone 2.0 release, finally, after 23 years of development, quickly followed by the 2.1 release. The second is that Wine is no longer a recursive acronym for Wine is Not an Emulator. There's no further clue as to whether it's still an acronym at all. Wine is now just Wine. However, Wine attempts to do the same job it's been trying for decades – allowing you to run Windows and DOS software on your Linux box without a Windows or DOS installation or license. The supreme effort in reverse engineering those APIs and re-implementing them as open source equivalents deserves a lot of credit, and I'm incredibly grateful to the developers for keeping this project going. However, as many of us know, getting Windows applications to run on Wine is often hit and miss, and often complicated by playing with lots of different configuration options.

This release has far too many new features to list. It finally supports GStreamer 1.0. DirectX 3D 11 is getting very close to being usable. Shader Models 4 and 5 work. There's a new clipboard. Microsoft Office 2013 works, and more than 6,600 individual changes have been rolled into the codebase. You still won't find support for the latest games, but I've found most games from three-four years ago can be made to work, although this is negated somewhat by the GPU pass-through mode of some virtual machines, which is a better option if you have a Windows license. But for those many small utilities, often that accompany hardware, or for the many niche tools and applications that have never been ported to Linux, Wine is worth a try and can often work if you're not too ambitious. And, you don't have to opt out of all the Windows 10 spyware.



THIS SHOULD WORK WITH THE ADOBE CC VERSION OF LIGHTROOM AS WELL.

IMPORTANT: This program does NOT work well with most Intel graphics. It WILL

One of the best ways of getting software to work with Wine is to use the Play On Linux front end.

Project Website
https://www.winehq.org/

Lock Down Your Network with Tinc

Block attackers from all your machines with a virtual mesh network.

BY BEN EVERARD

We'll be honest, computer security is something we'd rather not have to deal with. It'd be much nicer if we lived in a world where we didn't have to remember different complex passwords for every service we used, or have to carefully configure every network-facing service to ensure it remains secure against attack, but we have to live with the reality of the world we live in.

Attacks are commonplace and getting more sophisticated every day. Weak passwords will be broken sooner or later, and any open port on the Internet is under a constant bombardment from scanners trying to find vulnerabilities (just take a look at your logfiles if you don't believe us). Network security is usually dealt with on a service-by-service basis, but we're going to look at a way of locking everything down with a single bit of software.

Where most secure protocols (e.g., SSH or HTTPS) are designed to bring two computers together securely, Tinc brings the same technology to groups of computers. It allows you to create a mesh network between multiple machines regardless of where they're physically located where all communication between the machines is encrypted and authenticated. (See the "A Whirlwind Guide to Communications Protocols" box for more information.)

We're all familiar with encryption – where data is obfuscated so that only the people who should have access to it do – but authentication is a different aspect of security entirely. Authentication means knowing that you're really communicating with who you think you are. For example, the HTTPS protocol for secure web traffic is authenticated, but only at one end. This means that when you go to *https://www.linuxvoice.com*, you can be confident that you're really communicating with the Linux Voice server, but the Linux Voice server doesn't know who you are unless it goes through some additional steps (such as a password login).

Regular SSH is what we would term slightly authenticated. When you connect to a server for the first time, you're shown the fingerprint of the server to verify (but honestly, who does?), and you should see a warning any time this changes. With Tinc, every machine has a public key, and both sides of any communication are verified to be who they really claim to be.

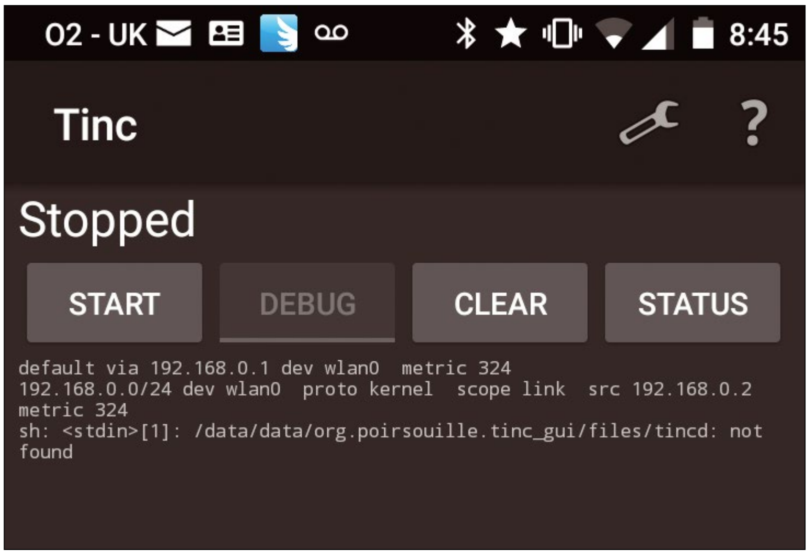
You might find Tinc in your software repositories, or you can install it from source fairly easily (Figure 1). For example, to install on a CentOS machine, you first need to grab the dependencies (which are just the gcc compiler and some header files for the libraries used).

```
sudo yum install gcc zlib-devel ⮐
  lzo-devel openssl-devel
```

Grab the source code from the website [1]. The latest version at the time of writing was 1.0.30. Then unzip it and open a terminal in the new directory. Building the software is then a case of running the three classic build commands

```
./configure
make
sudo make install.
```

You can check that it's worked by entering `tincd` at the command line. You should get a message telling you about the various options you need to run the network successfully.



```
default via 192.168.0.1 dev wlan0  metric 324
192.168.0.0/24 dev wlan0  proto kernel  scope link  src 192.168.0.2
metric 324
sh: <stdin>[1]: /data/data/org.poirsouille.tinc_gui/files/tincd: not
found
```

Figure 1: Tinc isn't only for your Linux machines; there are clients for most desktop OSs (including OS X and Windows) and Android as shown here.

Configuring Tinc is quite easy, but there are a few things that you have to get in place. Before any of that, though, you need to decide on how to allocate your IP addresses. Tinc doesn't do this automatically, so you need to make the decision for yourself. There are several private IP address ranges that you can choose from including 192.168.x.x (but this is often used for physical networks), and 10.x.x.x (each x can be between 1 and 255).

The most important thing to decide when picking an IP address range is which other networks your machines are connected to. You can use the `ip addr` command to show what IPs are currently associated with a machine and then pick an entirely different range. For our network, we're going to use 10.0.0.1 to 10.0.0.255 (or at least we would if we had enough machines to fill this range). It's not too difficult to change the IP in this range that a particular machine has, but it's best to get it right the first time. If your machines are already ordered in some way, give them IPs ordered in the same way.

There's no distinction between the server and client in Tinc. Every machine is just a part of the network. However, for all the machines to connect together, some machines need to act as points of contact that the others reach out to when the network is first established. The most important thing to think about here is which machines are reachable from which other machines.

For example, you might be using a cluster of cloud servers to provide services for each other and want them to connect securely. If they're all in the same hosting provider, then they probably all have private IP addresses (or a private DNS hostname) that they can use to communicate. In this case, any or all of them could be the hosts to connect to. However, if you want to connect to this network from outside, you'll need at least one machine with a public IP address that you can connect to. Once you're connected to one machine, Tinc will route traffic to other machines even if you can't connect directly to their main network interfaces. For this reason, it's a great way of securing machines spanning multiple physical networks.

Each machine requires five bits of configuration: a conf file, a tinc-up script, a tinc-down script, a private key, and a set of hosts files. Let's look at them in this order. First, the configuration file. This should look something like the following:

```
Name = machine1
AddressFamily = ipv4
Interface = tun0
ConnectTo = MainMachine
```

In this example, `machine1` is the name of the current computer and `MainMachine` is the name of the computer that this should connect to when Tinc starts. You can have multiple `ConnectTo` lines, so your network is resilient if one or more machines go down. Tinc can have multiple networks on the same machine, so all the configuration files are separated out by network. As such, this file should go in the configuration for the network you're building (which we'll call LinuxVoiceNet, but you can replace this with your own name).

Installing from source puts the configuration files in `/usr/local/etc/tinc`, but if you've installed via a package manager, this might be `/etc/tinc`. Whichever, put the above file inside this configuration root as `LinuxVoiceNet/tinc.conf`.

The next thing we need to consider is tinc-up, which is a script that runs when the network is started and just makes sure the interface is configured properly. This might sound a little scary, but it's actually really simple. The following is all we need for this machine:

```
#!/bin/sh
ifconfig $INTERFACE 10.0.0.1↩
netmask 255.255.255.0
```

This will assign the machine the IP 10.0.0.1. The netmask of 255.255.255.0 means that anything matching an IP like 10.0.0.x can be found on this network. The `$INTERFACE` part isn't a typo on our part. Tinc will insert the correct network interface here when it runs the script.

Tinc-down is even simpler and just needs to contain the following:

```
#!/bin/sh
ifconfig $INTERFACE down
```

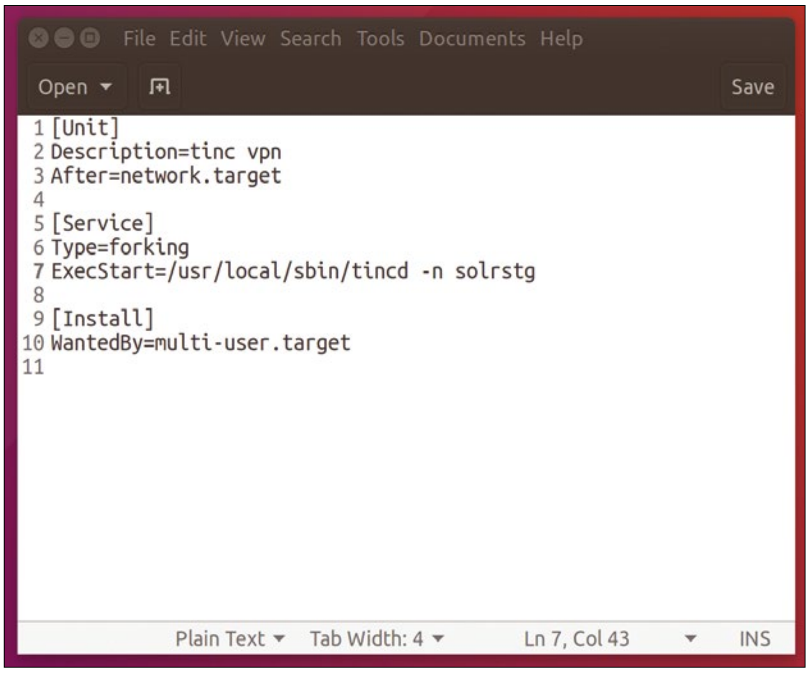
Both of these should live in the LinuxVoiceNet folder (as tinc-up and tinc-down, respectively).

So far, we've just been configuring the network, but now we need to look at security, and this comes down to keys. You can generate a good, strong key with:

```
sudo tincd -n LinuxVoiceNet -K4096
```

This will put the appropriate files in the appropriate places. You should have a private key in the main configuration directory and a file at `LinuxVoiceNet/hosts/machine1` (or whatever you called this machine). In this hosts directory, we'll keep a copy of the public keys of all the machines we connect to, but we'll deal with that in a minute. First, we need to edit the newly created `machine1` file to include a bit more information. Open it up in a text editor and add the following lines to the top of it:

```
Address = <main IP or domain>
Subnet = 10.0.0.1/32
```



```
[Unit]
Description=tinc vpn
After=network.target

[Service]
Type=forking
ExecStart=/usr/local/sbin/tincd -n solrstg

[Install]
WantedBy=multi-user.target
```

Figure 2: Systemd caused a lot of controversy when it first came out, but it's been adopted by most major distros and makes services easy to create.

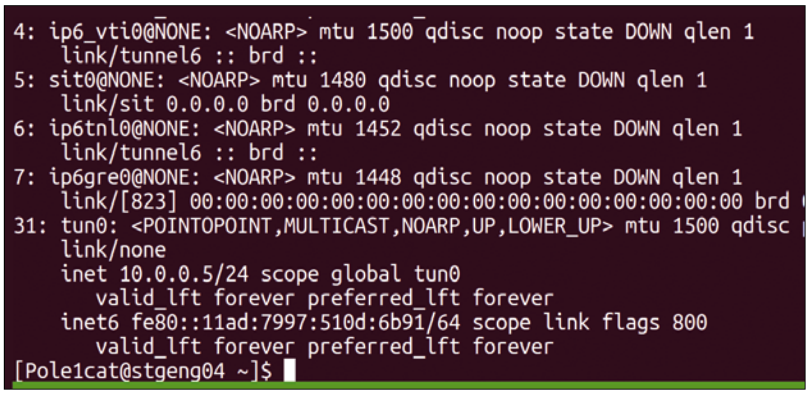
Here we need to tell the other machines how they can contact this machine in order to make a Tinc connection. This could be a public IP, a private IP on the same network as the other machines, or a domain name. Strictly speaking, this is only necessary on machines that others connect to, but it's best to have it on as many machines on your Tinc network as possible, as this can ease congestion on the network.

You need to follow all of the above steps on all of your machines (this is a great candidate for automating via a configuration management tool, such as Ansible). Once you've done that, gather all the files from the hosts directory (there should be one named after each machine) and copy them so that every machine has a copy of every other machine in the hosts folder. Once that's done, you can start the network with a simple:

```
tincd -n LinuxVoiceNet -D d3
```

Remember that you need to do this on the machine that the others connect to first.

Figure 3: Launching Tinc will create a new network interface (`tun0`), which has its own IP address. This interface is only for traffic on the virtual network.



```
4: ip6_vti0@NONE: <NOARP> mtu 1500 qdisc noop state DOWN qlen 1
    link/tunnel6 :: brd ::
5: sit0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN qlen 1
    link/sit 0.0.0.0 brd 0.0.0.0
6: ip6tnl0@NONE: <NOARP> mtu 1452 qdisc noop state DOWN qlen 1
    link/tunnel6 :: brd ::
7: ip6gre0@NONE: <NOARP> mtu 1448 qdisc noop state DOWN qlen 1
    link/[823] 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00 brd
31: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc
    link/none
    inet 10.0.0.5/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::11ad:7997:510d:6b91/64 scope link flags 800
       valid_lft forever preferred_lft forever
[Pole1cat@stgeng04 ~]$
```

This will run the network daemon in the current terminal, which makes it easy to see where any problems are. However, for normal running, you'll want a way of starting the service in the background. This varies a little between distros, particularly at the moment as most modern distros use systemd (Figure 2), but many older (but still supported) distros are using `sys-v init`. It's easy to tell if you're using systemd. Just run the following command:

```
cd /etc/systemd
```

If you get an error saying that this folder doesn't exist, then you're not running systemd and you need to create a script. If you move into the directory, then you're running systemd, and you need to create a unit file.

The start scripts and unit files only define how Tinc starts and not how it runs, so there's absolutely no problem running a network with some machines using unit files and some using startup scripts. You can also launch Tinc manually on some machines as well if you want (Figure 3).

Let's look at systemd unit files, as this covers most modern distros.

```
[Unit]
Description=tinc vpn
After=network.target

[Service]
Type=forking
ExecStart=/usr/local/sbin/tincd ↩
  -n LinuxVoiceNet

[Install]
WantedBy=multi-user.target
```

If you're using `sys-v init`, however, you'll need to write a shell script that's run at the appropriate time to start and stop the service. These scripts are rather big, so we can't fit the whole thing in here, but you can follow the example online [2] to quickly get one running (Figure 4).

Whichever option you choose, you're ready to get started. First, stop the Tinc process running if you haven't already with Ctrl+/, then run

```
service tincd start
```

to start `tincd`.

You can configure your system to automatically start Tinc on boot with either

```
systemctl enable tincd.service
```

if you used a systemd unit file, or the following if you've used a `sys-v init` script:

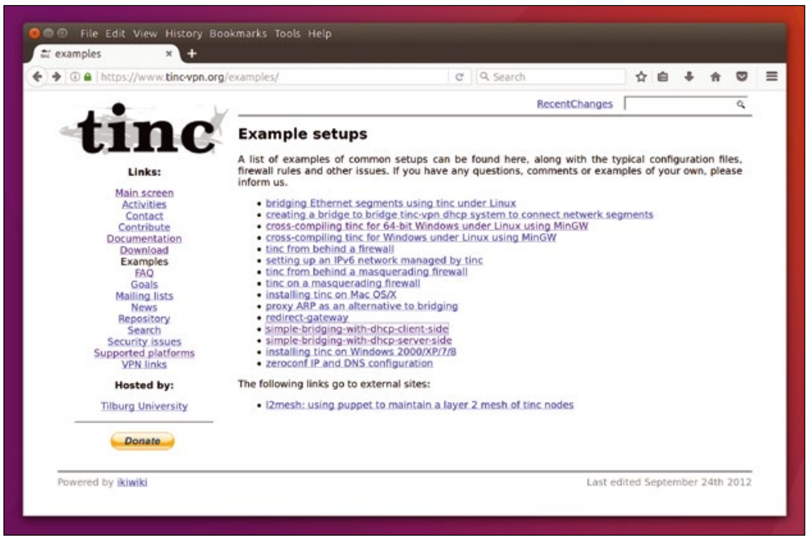
```
chkconfig tincd on
```

That really is all it takes to lock down a distributed network of machines. With Tinc running, you can share whatever you need safely and secure in the knowledge that prying eyes can't see what you're doing. Using Tinc isn't an excuse to ignore good security practices, but it does give you an extra layer of protection against the hordes of attackers roaming the Internet. ■■■

Info

[1] Tinc: *https://www.tinc-vpn.org/*

[2] Init script template: *https://github.com/fhd/init-script-template*


File Edit View History Bookmarks Tools Help
examples
https://www.tinc-vpn.org/examples/
Search
RecentChanges
tinc
Donate

Example setups

A list of examples of common setups can be found here, along with the typical configuration files, firewall rules and other issues. If you have any questions, comments or examples of your own, please inform us.

- bridging Ethernet segments using tinc under Linux
- creating a bridge to bridge tinc-vpn dhcp system to connect netwerk segments
- cross-compiling tinc for 64-bit Windows under Linux using MinGW
- cross-compiling tinc for Windows under Linux using MinGW
- tinc from behind a firewall
- setting up an IPv6 network managed by tinc
- tinc from behind a masquerading firewall
- tinc on a masquerading firewall
- installing tinc on Mac OS/X
- proxy ARP as an alternative to bridging
- redirect-gateway
- simple-bridging-with-dhcp-client-side
- simple-bridging-with-dhcp-server-side
- installing tinc on Windows 2000/XP/7/8
- zeroconf IP and DNS configuration

The following links go to external sites:

- l2mesh: using puppet to maintain a layer 2 mesh of tinc nodes

Links:

Main screen
Activities
Contact
Contribute
Documentation
Download
Examples
FAQ
Goals
Mailing lists
News
Repository
Search
Security issues
Supported platforms
VPN links

Hosted by:

Tilburg University

Powered by ikiwiki

Last edited September 24th 2012

Figure 4: Take a look at the Tinc website for inspiration on more advanced configurations.

A Whirlwind Guide to Communications Protocols

There are loads of different ways computers communicate with each other. Here's the information you need to know about the security of some of the most popular:

- HTTP – When most people think of computers communicating over the internet, they think of the Hypertext Transfer Protocol (HTTP) that serves up web pages. It has essentially no security features.
- HTTPS – With the web becoming more important, HTTPS was developed to improve on the shortcomings of HTTP. By pushing plain HTTP over Secure Socket Layer (SSL) and more recently Transport Layer Security (TLS), the same functionality is retained, but the browser can be reasonably sure that the web server is who they say they are. HTTPS relies on trusted certificate authorities to verify the identity of websites; if one of these authorities is compromised, then the entire authentication of HTTPS is in question.
- RCP – Rarely seen these days, Remote Copy (RCP) is the traditional way of moving files between machines on a network. It has no security.
- SCP – Like RCP, SCP copies files between machines via a network; however it adds the security afforded by SSH, which includes some authentication and strong encryption (provided it is set up).
- Rsync – Rsync builds on SCP with more features for deciding which files to copy. While the security aspects are similar, Rsync is more powerful.
- FTP – File Transfer Protocol is a communications protocol optimized for downloading large files. Like HTTP, it was designed in a simpler time when computer security wasn't seen as a significant problem and should be avoided if security is a concern.
- SFTP – By wrapping a secure layer over FTP, SFTP gives the user more security guarantees than the original protocol.
- Gopher – A protocol for retrieving documents stored in a hierarchical structure that was briefly popular during the mid-nineties. Nowadays, the protocol's main purpose seems to be eliciting nostalgia from aging geeks. It is unsecure.
- Email – Although it's not a protocol on its own, the global email system works using a range of protocols including SMTP, IMAP, and POP. The major problem with email from a security perspective is that it's impossible to know exactly how your message is transmitted once it leaves your computer. It may be secure; it may not be.
- PGP/GPG – Pretty Good Privacy (also implemented by Gnu Privacy Guard) is a protocol for encrypting data that is to be sent over another protocol. It's most commonly used for email, but also works on any system that allows you to send text. It can be technical to use but offers strong encryption and authentication even over channels that don't normally have such protections.
- Telnet – A remote access protocol that allows you to log into a terminal session via a network connection. It has no encryption and should not be deployed. Treat any Telnet system you encounter with deep suspicion.
- SSH – Secure Shell is a remote access protocol that's more advanced than Telnet both in terms of features and security. Care should be taken when configuring an SSH server as it is possible to configure insecurely, and SSH ports on the Internet are heavily attacked.
- Signal – An instant messaging protocol developed by Whisper Systems for instant messaging. It's used in the app of the same name and WhatsApp to provide strong end-to-end encryption. It's our recommended protocol for instant messaging.

Master Your Desktop
Be a Window Wizard

Regular window managers are so 2016 – install a tiling WM and work faster, smarter, and cooler.

BY MIKE SAUNDERS

Despite all of the vast gains seen in technology over the last few decades, user interfaces on desktop computers have remained pretty much the same. We have insanely fast CPUs and graphics cards to make our windows wobbly, and we have truckloads of RAM to run countless apps simultaneously. But we still tend to have the same kind of setup on our desktops: a program launcher, a taskbar or list of running programs, a "system tray" area, various windows scattered around, and so forth. Even the desktop environments that have bucked the trend in recent years, like Gnome 3 and Ubuntu's Unity, still keep most of the basics that we've been using since the Amiga Workbench days.

Now, you could argue that this is a good thing. User interface design has been refined over the decades, and we're all used to how WIMP (windows, icons, menus, and pointing device) setups work, so why change it? Why throw all of this away for something novel and "experimental" that requires learning a whole new workflow?

Well, sometimes that workflow can make you faster. A lot faster. Look at the command line, for instance: We Linux and Unix geeks know that it's not as pretty or welcoming as a flashy GUI, but it gets many things done much more quickly. Prodding the mouse, going through tabs in a dialog box, fiddling with sliders and radio buttons – who wants that when you can just type in a command and get your flippin' work done?

So Long, Rodent

Let's apply this thinking to desktop environments and window managers. Having lots of buttons, menus, and titlebars may look nice and helpful, but arguably they just eat up screen space. And, although many window managers have "smart" window placement modes and try to guess where you want a window to be, they usually do a pretty bad job of it. If you have a large screen (or even better, you're working with a multi-monitor setup), you've probably wasted enough time repositioning all of your favorite apps after every fresh boot of your distro.

This is where "tiling" window managers (WMs) are godsends. Tiling WMs do the job of positioning windows for you, placing them strategically and using up every spare pixel of screen space at the same time. You don't have bits of windows overlapping other ones; you don't have to drag them around and clumsily position them in a sort-of-about-right place – the tiling WM does all this for you. Of course, you still have manual control over your window placement, but with the right setup you rarely have to get involved. Oh, and to top it all off, tiling WMs are heavily keyboard driven, so you can switch between apps and rearrange your desktop layout right from the middle of your typing work – no need to reach for the mouse or trackpad.

One of the best tiling WMs doing the rounds is i3; it's especially popular in power-user distro circles (e.g., Arch Linux), but it can be installed in pretty much every other distro. And, although i3 takes a bit of getting used to, it's not especially hard to master and can save you huge amounts of time in the long run. In this article, we'll show you how it works, how to configure it, and how to get the most out of it.

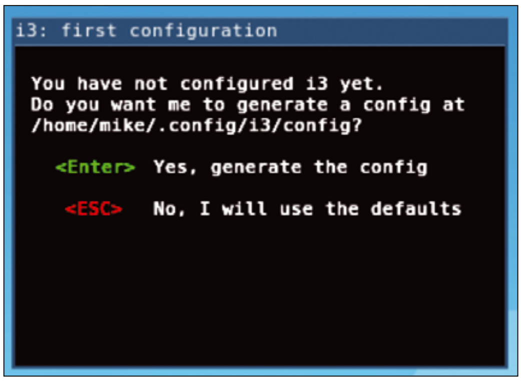
i3 Basics

The first thing you'll need to do is install i3 from your distribution's package repositories. Have a look in your package manager – or on Debian and Ubuntu-based distros, you can install it with a simple:

```
sudo apt-get install i3
```

If you can't find it, grab the source code from i3's website [1] and build it yourself. If you have trouble compiling it, try posting a message on the i3 subreddit [2], which has plenty of users around to help.

Once you have i3 installed, log out of your current desktop environment and then choose i3 at your distro's login screen. Exactly how you do this will vary depending on your distro's login manager – usually there's a small icon showing your current desktop environment (Gnome, KDE,



```
i3: first configuration

 You have not configured i3 yet.
 Do you want me to generate a config at
 /home/mike/.config/i3/config?

   <Enter> Yes, generate the config

    <ESC>  No, I will use the defaults
```

Figure 1: When first launched, i3 will offer to set you up with a default config file.

Xfce, etc.) that you can click to change it. So, click that and select i3.

Then log in, and you'll immediately see a "first configuration" box like in Figure 1. Here i3 is offering to generate and save a default config file for you – and that's definitely what you want at this stage, so hit Enter to proceed. Then you'll be posed another question: do you want to use your Windows key as the "default modifier," or the Alt key? This default modifier is the key you use in combination with other keys to manage and manipulate i3. We're going to choose Alt for this tutorial, but you can choose whatever suits your workflow best. (Of course, if you choose the Windows key, replace Alt with Windows in the instructions we give throughout this tutorial.)

And then… Well, there's nothing. Or almost nothing. Your screen will go completely blank – or at least show the wallpaper from your login manager – with nothing obvious going on. Look down at the bottom of your screen, though, and you'll see a thin black bar containing various bits of information. This is the i3bar, and it's running a tool called i3status inside of it. Have a look at the data it displays: You'll see that there's information about your network connection, free hard drive space, IP address, CPU load, along with the date and time. All rather handy and taking up very little space.

Now try clicking the bits of data on the i3bar though – notice how nothing happens? As mentioned, i3 is very much keyboard-centric, so the mouse does very little out of the box. i3 encourages you to keep your hands firmly on the keyboard, with pretty much every feature available via keyboard shortcuts. Indeed, you can't even launch a program via the mouse; if you've been searching around for a Start-like menu, you won't have had any luck.

To open a terminal, hit Alt+Enter. You'll see straight away that the terminal fills up the entire screen. This is where we start to see the basics of tiling: i3 sees the whole empty desktop as one big "tile" in which it can place a newly launched application. i3 automatically makes use of all available space – you don't need to maximize your windows manually.

Managing Windows

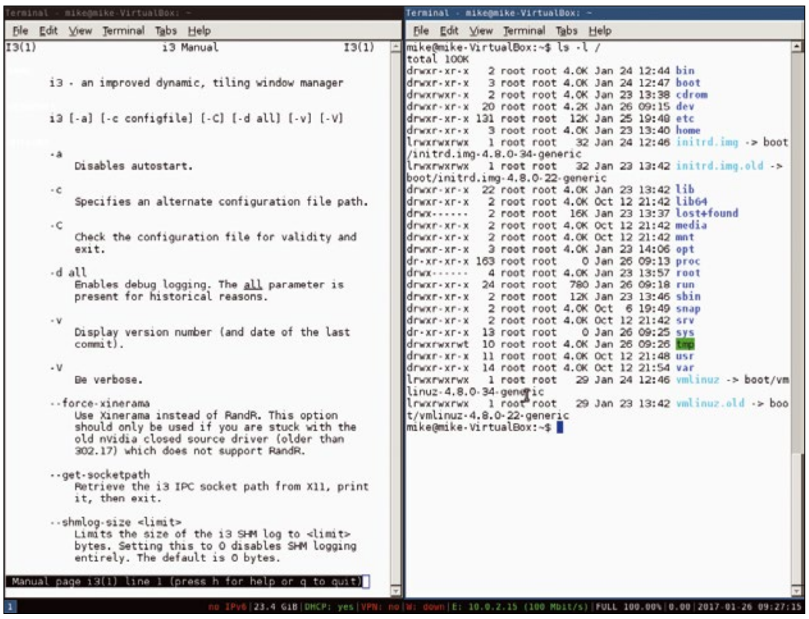
Now, with that single terminal window open, hit Alt+Enter again to open another terminal. i3 immediately splits the screen into two tiles: one on the left, and one on the right, leaving you with two terminals open side-by-side, like in Figure 2. There's no empty space on the screen here. The currently focused window is the one on the right-hand side, so what if you want to switch to the other one?

You could move the mouse over (i3 operates under the "focus follows mouse" model, so you don't have to click on a window to make it active), but that's not ideal when we want to keep our hands on the keyboard. Instead, use Alt along with your arrow keys; you can tell which window is focused by the color of the title bar and the thin frame around it (in the default configuration, focused windows are dark blue while unfocused ones are black).

Now, you can keep hitting Alt+Enter to pop up terminals so that they're all shown side-by-side, but sometimes you'll want to make one window appear below the current one. Make sure you have two terminals open like in Figure 2, with the right-hand one active. Tap Alt+V to prepare i3 for a vertical split, and then hit Alt+Enter again – and you should now have three terminals visible, with one taking up the left-hand part of the screen and the other two on the right-hand part, stacked on top of each other (Figure 3).

If you hit Alt+Enter again, another terminal will appear in that right-hand section. Press Alt+H to return to horizontal splits and hit Alt+Enter – this time the new terminal window appears to the right of the current one. It might sound rather odd and unintuitive,

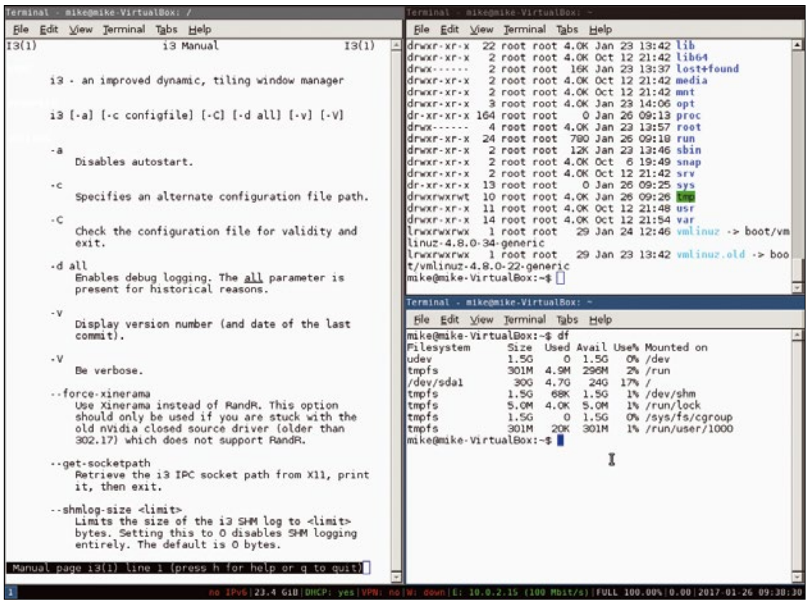
Figure 2: Here's our first exposure to tiling, with two windows being automatically placed side-by-side.



```
Terminal - mike@mike-VirtualBox: ~
File Edit View Terminal Tabs Help
I3(1)                     i3 Manual                    I3(1)

       i3 - an improved dynamic, tiling window manager

       i3 [-a] [-c configfile] [-C] [-d all] [-v] [-V]

       -a
           Disables autostart.

       -c
           Specifies an alternate configuration file path.

       -C
           Check the configuration file for validity and
           exit.

       -d all
           Enables debug logging. The all parameter is
           present for historical reasons.

       -v
           Display version number (and date of the last
           commit).

       -V
           Be verbose.

       --force-xinerama
           Use Xinerama instead of RandR. This option
           should only be used if you are stuck with the
           old nVidia closed source driver (older than
           302.17) which does not support RandR.

       --get-socketpath
           Retrieve the i3 IPC socket path from X11, print
           it, then exit.

       --shmlog-size <limit>
           Limits the size of the i3 SHM log to <limit>
           bytes. Setting this to 0 disables SHM logging
           entirely. The default is 0 bytes.

 Manual page i3(1) line 1 (press h for help or q to quit)
```

```
Terminal - mike@mike-VirtualBox: ~
File Edit View Terminal Tabs Help
mike@mike-VirtualBox:~$ ls -l /
total 100K
drwxr-xr-x   2 root root 4.0K Jan 24 12:44 bin
drwxr-xr-x   3 root root 4.0K Jan 24 12:47 boot
drwxrwxr-x   2 root root 4.0K Jan 23 13:38 cdrom
drwxr-xr-x  20 root root 4.2K Jan 26 09:15 dev
drwxr-xr-x 131 root root  12K Jan 25 19:48 etc
drwxr-xr-x   3 root root 4.0K Jan 23 13:40 home
lrwxrwxrwx   1 root root   32 Jan 24 12:46 initrd.img -> boot
/initrd.img-4.8.0-34-generic
lrwxrwxrwx   1 root root   32 Jan 23 13:42 initrd.img.old ->
boot/initrd.img-4.8.0-22-generic
drwxr-xr-x  22 root root 4.0K Jan 23 13:42 lib
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 lib64
drwx------   2 root root  16K Jan 23 13:37 lost+found
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 media
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 mnt
drwxr-xr-x   3 root root 4.0K Jan 23 14:06 opt
dr-xr-xr-x 163 root root    0 Jan 26 09:13 proc
drwx------   4 root root 4.0K Jan 23 13:57 root
drwxr-xr-x  24 root root  780 Jan 26 09:18 run
drwxr-xr-x   2 root root  12K Jan 23 13:46 sbin
drwxr-xr-x   2 root root 4.0K Oct  6 19:49 snap
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 srv
dr-xr-xr-x  13 root root    0 Jan 26 09:25 sys
drwxrwxrwt  10 root root 4.0K Jan 26 09:26 tmp
drwxr-xr-x  11 root root 4.0K Oct 12 21:48 usr
drwxr-xr-x  14 root root 4.0K Oct 12 21:54 var
lrwxrwxrwx   1 root root   29 Jan 24 12:46 vmlinuz -> boot/vm
linuz-4.8.0-34-generic
lrwxrwxrwx   1 root root   29 Jan 23 13:42 vmlinuz.old -> boo
t/vmlinuz-4.8.0-22-generic
mike@mike-VirtualBox:~$
```

```
1        no IPv6|23.4 GiB|DHCP: yes|VPN: no|W: down|E: 10.0.2.15 (100 Mbit/s)|FULL 100.00%|0.00|2017-01-26 09:27:15
```



```
Terminal - mike@mike-VirtualBox: /
File  Edit  View  Terminal  Tabs  Help
I3(1)                    i3 Manual                    I3(1)

       i3 - an improved dynamic, tiling window manager

       i3 [-a] [-c configfile] [-C] [-d all] [-v] [-V]

       -a
           Disables autostart.

       -c
           Specifies an alternate configuration file path.

       -C
           Check the configuration file for validity and
           exit.

       -d all
           Enables debug logging. The all parameter is
           present for historical reasons.

       -v
           Display version number (and date of the last
           commit).

       -V
           Be verbose.

       --force-xinerama
           Use Xinerama instead of RandR. This option
           should only be used if you are stuck with the
           old nVidia closed source driver (older than
           302.17) which does not support RandR.

       --get-socketpath
           Retrieve the i3 IPC socket path from X11, print
           it, then exit.

       --shmlog-size <limit>
           Limits the size of the i3 SHM log to <limit>
           bytes. Setting this to 0 disables SHM logging
           entirely. The default is 0 bytes.

 Manual page i3(1) line 1 (press h for help or q to quit)
```

```
Terminal - mike@mike-VirtualBox: ~
File  Edit  View  Terminal  Tabs  Help
drwxr-xr-x  22 root root 4.0K Jan 23 13:42 lib
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 lib64
drwx------   2 root root  16K Jan 23 13:37 lost+found
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 media
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 mnt
drwxr-xr-x   3 root root 4.0K Jan 23 14:06 opt
dr-xr-xr-x 164 root root    0 Jan 26 09:13 proc
drwx------   4 root root 4.0K Jan 23 13:57 root
drwxr-xr-x  24 root root  780 Jan 26 09:18 run
drwxr-xr-x   2 root root  12K Jan 23 13:46 sbin
drwxr-xr-x   2 root root 4.0K Oct  6 19:49 snap
drwxr-xr-x   2 root root 4.0K Oct 12 21:42 srv
dr-xr-xr-x  13 root root    0 Jan 26 09:25 sys
drwxrwxrwt  10 root root 4.0K Jan 26 09:26 tmp
drwxr-xr-x  11 root root 4.0K Oct 12 21:48 usr
drwxr-xr-x  14 root root 4.0K Oct 12 21:54 var
lrwxrwxrwx   1 root root   29 Jan 24 12:46 vmlinuz -> boot/vm
linuz-4.8.0-34-generic
lrwxrwxrwx   1 root root   29 Jan 23 13:42 vmlinuz.old -> boo
t/vmlinuz-4.8.0-22-generic
mike@mike-VirtualBox:~$
```

```
Terminal - mike@mike-VirtualBox: ~
File  Edit  View  Terminal  Tabs  Help
mike@mike-VirtualBox:~$ df
Filesystem      Size  Used Avail Use% Mounted on
udev            1.5G     0  1.5G   0% /dev
tmpfs           301M  4.9M  296M   2% /run
/dev/sda1        30G  4.7G   24G  17% /
tmpfs           1.5G   68K  1.5G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           1.5G     0  1.5G   0% /sys/fs/cgroup
tmpfs           301M   20K  301M   1% /run/user/1000
mike@mike-VirtualBox:~$
```

```
1      no IPv6 | 23.4 GiB | DHCP: yes | VPN: no | W: down | E: 10.0.2.15 (100 Mbit/s) | FULL 100.00% | 0.00 | 2017-01-26 09:30:30
```

Figure 3: We can split the screen horizontally or vertically, and indeed do combinations of those splits.

but try playing around with different combinations: Open and close terminals, using Alt+V and Alt+H to switch between different layouts as you do so. Along with Alt and the cursor keys to navigate between windows, you can use the JKL; keys for left, down, up and right, respectively.

If you want to move a window into a different tile, use Alt+Shift together with the cursor keys (or JKL;). Resizing tiles is possible using the keyboard: press Alt+R to switch to resize mode (you'll see the word "resize" in the left-hand part of the status bar), and then use the arrow keys to change the size. When you're done, hit Esc. If you find this approach a bit unintuitive with certain layouts, you can still resort to the mouse – just click and drag the separators between windows (or the titlebars in vertical layouts).

Launching and Closing Apps

So far we've only been opening terminal windows, which is fine if you do most of your work at the command line, but obviously isn't great for general desktop usage. Fortunately, there's a dedicated program launcher for i3 called dmenu that is – as you've probably guessed by now – keyboard-centric. To activate it, press Alt+D and look at the top of the screen: You should see a bar with a gap on the left and various programs listed on the right. (If you don't see this, make sure dmenu is installed – have a search for it in your distro's package manager.)

Now, with dmenu open, start typing something – for example, f. You'll see that dmenu narrows down the list of displayed apps to those beginning with f. Type "fire" and (unless you have other apps installed beginning with fire), it should narrow down to "firefox", so hit Enter to launch it. As you're seeing, dmenu is a useful way to quickly launch applications without having to type in the full executable names – at least, most of the time!

To close an window, press Alt+Shift+Q. This tells i3 to send a regular "close window" command to the application, that is, shutting it down gracefully. So if you're running an editor and haven't saved your work, the app will prompt you. Try it out with, say, LibreOffice – start a word processor document, tap in some text, and then press Alt+Shift+Q. LibreOffice's *Save Document?* dialog will pop up, and you'll also notice that i3 lets it appear in the middle of the screen, as usual (see the "Using Workspaces" box for more info).

So i3 isn't overly zealous about tiling; it tries to give each window and application its own dedicated space on the screen, making sure no pixels are wasted, but it's also clever enough to respect "floating" windows that are temporary and don't need to be assigned their own tiles. You can see this in Figure 4.

You're now ready to start using i3 for your daily work – it may take some time to get used to, but after a while you'll find it much more elegant to work with than regular point-and-click window managers (Figure 5).

But one final thing we'll look at is the default configuration file that i3 created when you first launched it. This is called *config*, and you'll find it in the `.config/i3/` subdirectory of your home directory.

Apart from the *font* option, which lets you customize the font used in window titles and the i3bar, most of the other options are preceded by

Using Workspaces

Like most desktop environments and window managers, i3 supports workspaces – separate virtual "screens" where you can open additional applications. You can see the number of the currently active workspace in the far-left of the status bar at the bottom (it should show "1" at this stage). To switch to another workspace, use Alt along with a number (e.g., Alt+2 to switch to workspace 2). These are created on the fly, so you don't have to worry about creating or managing them.

In many cases, you'll want to move windows between different workspaces; this is achieved by using Alt+Shift along with a number. So, if you're in workspace 1 with two windows open side-by-side and press Alt+Shift+2, the currently selected window will be moved into the second workspace. And, if that workspace was empty before the move, the window will take up the whole screen, like when we first started i3.

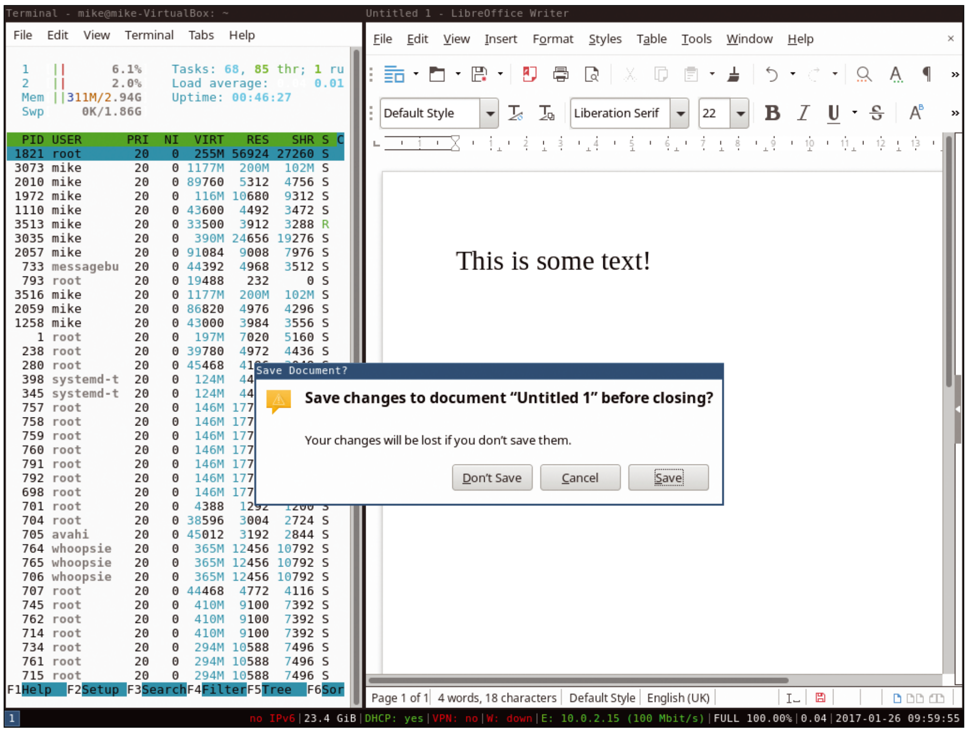
bindsym and configure the keybindings for the various window management options. For instance, look at this:

```
bindsym $mod+Shift+q kill
```

Here, *$mod* is the modifier key (Alt or Windows) that we selected during the first run of i3 – and it's also defined at the top of the configuration file. So, this line of code says "bind the key combination $mod+Shift+q to the kill command" with *kill* meaning close the window.

Look through the file to explore the available keybindings – there are a few lesser-used ones that we haven't talked about in this guide, but that you may find useful as time goes on. If there's a particular default keybinding that you find awkward to use, you can simply change it here, save the file, and press Alt+Shift+C to reload the configuration file without having to restart i3 completely. (To exit i3, use Alt+Shift+E and notice the confirmation bar at the top of the screen.)

One last thing: When you're getting to grips with i3, we recommend loading up the i3 Reference Card [3] and making a printout of it for your wall. For the complete i3 documentation, describing many different ways to divide up your screen, see the User's Guide [4]. Also, check out the "Other Tiling WMs" box for alternatives to i3. ∎∎∎



```
Terminal - mike@mike-VirtualBox: ~
File  Edit  View  Terminal  Tabs  Help

  1  [||        6.1%]   Tasks: 68, 85 thr; 1 ru
  2  [||        2.0%]   Load average: 0.04 0.01
  Mem[||311M/2.94G]     Uptime: 00:46:27
  Swp[       0K/1.86G]

  PID USER      PRI  NI  VIRT   RES   SHR S C
 1821 root       20   0  255M 56924 27260 S
 3073 mike       20   0 1177M  200M  102M S
 2010 mike       20   0 89760  5312  4756 S
 1972 mike       20   0  116M 10680  9312 S
 1110 mike       20   0 43600  4492  3472 S
 3513 mike       20   0 33500  3912  3288 R
 3035 mike       20   0  390M 24656 19276 S
 2057 mike       20   0 91084  9008  7976 S
  733 messagebu  20   0 44392  4968  3512 S
  793 root       20   0 19488   232     0 S
 3516 mike       20   0 1177M  200M  102M S
 2059 mike       20   0 86820  4976  4296 S
 1258 mike       20   0 43000  3984  3556 S
    1 root       20   0  197M  7020  5160 S
  238 root       20   0 39780  4972  4436 S
  280 root       20   0 45468  41
  398 systemd-t  20   0  124M  44
  345 systemd-t  20   0  124M  44
  757 root       20   0  146M  177
  758 root       20   0  146M  177
  759 root       20   0  146M  177
  760 root       20   0  146M  177
  791 root       20   0  146M  177
  792 root       20   0  146M  177
  698 root       20   0  146M  177
  701 root       20   0  4388  1292  1200 S
  704 root       20   0 38596  3004  2724 S
  705 avahi      20   0 45012  3192  2844 S
  764 whoopsie   20   0  365M 12456 10792 S
  765 whoopsie   20   0  365M 12456 10792 S
  706 whoopsie   20   0  365M 12456 10792 S
  707 root       20   0 44468  4772  4116 S
  745 root       20   0  410M  9100  7392 S
  762 root       20   0  410M  9100  7392 S
  714 root       20   0  410M  9100  7392 S
  734 root       20   0  294M 10588  7496 S
  761 root       20   0  294M 10588  7496 S
  715 root       20   0  294M 10588  7496 S
F1Help  F2Setup F3SearchF4FilterF5Tree  F6Sor
```

Untitled 1 - LibreOffice Writer

This is some text!

Save Document?

Save changes to document "Untitled 1" before closing?

Your changes will be lost if you don't save them.

Don't Save | Cancel | Save

Page 1 of 1 | 4 words, 18 characters | Default Style | English (UK)

```
1    no IPv6|23.4 GiB|DHCP: yes|VPN: no|W: down|E: 10.0.2.15 (100 Mbit/s)|FULL 100.00%|0.04|2017-01-26 09:59:55
```

Figure 4: i3 doesn't try to tile absolutely everything, though and respects the wishes of floating dialogs.

Info

[1] i3: *https://i3wm.org*

[2] i3 subreddit: *https://www.reddit.com/r/i3wm/*

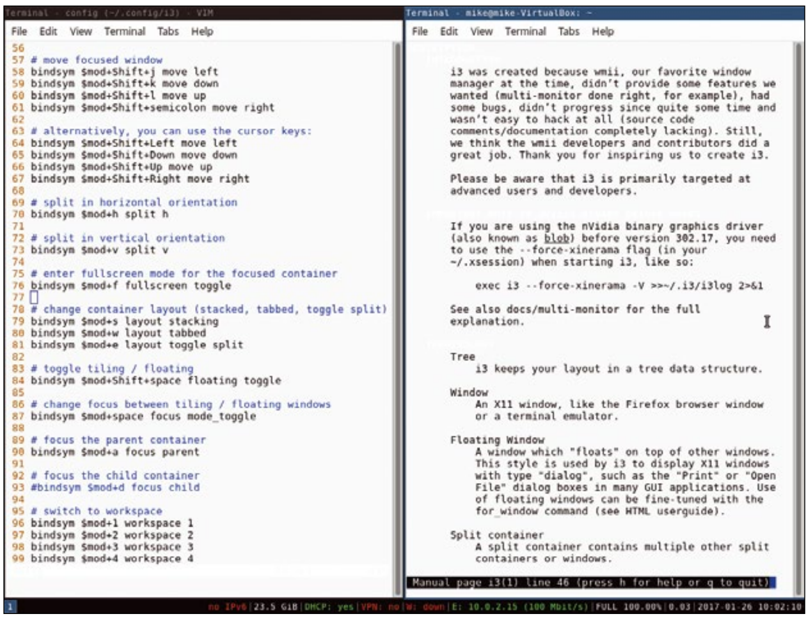
[3] i3 Reference Card: *https://i3wm.org/docs/refcard.html*

[4] User's Guide: *https://i3wm.org/docs/userguide.html*

[5] Awesome: *https://awesomewm.org*

[6] Dwm: *http://dwm.suckless.org*

Figure 5: A great example of i3 at its best: editing a config file in one tile while viewing its manual page in another.



```
Terminal - config (~/.config/i3) - VIM
File  Edit  View  Terminal  Tabs  Help

# move focused window
bindsym $mod+Shift+j move left
bindsym $mod+Shift+k move down
bindsym $mod+Shift+l move up
bindsym $mod+Shift+semicolon move right

# alternatively, you can use the cursor keys:
bindsym $mod+Shift+Left move left
bindsym $mod+Shift+Down move down
bindsym $mod+Shift+Up move up
bindsym $mod+Shift+Right move right

# split in horizontal orientation
bindsym $mod+h split h

# split in vertical orientation
bindsym $mod+v split v

# enter fullscreen mode for the focused container
bindsym $mod+f fullscreen toggle

# change container layout (stacked, tabbed, toggle split)
bindsym $mod+s layout stacking
bindsym $mod+w layout tabbed
bindsym $mod+e layout toggle split

# toggle tiling / floating
bindsym $mod+Shift+space floating toggle

# change focus between tiling / floating windows
bindsym $mod+space focus mode_toggle

# focus the parent container
bindsym $mod+a focus parent

# focus the child container
#bindsym $mod+d focus child

# switch to workspace
bindsym $mod+1 workspace 1
bindsym $mod+2 workspace 2
bindsym $mod+3 workspace 3
bindsym $mod+4 workspace 4
```

```
Terminal - mike@mike-VirtualBox: ~
File  Edit  View  Terminal  Tabs  Help

       i3 was created because wmii, our favorite window
       manager at the time, didn't provide some features we
       wanted (multi-monitor done right, for example), had
       some bugs, didn't progress since quite some time and
       wasn't easy to hack at all (source code
       comments/documentation completely lacking). Still,
       we think the wmii developers and contributors did a
       great job. Thank you for inspiring us to create i3.

       Please be aware that i3 is primarily targeted at
       advanced users and developers.

       If you are using the nVidia binary graphics driver
       (also known as blob) before version 302.17, you need
       to use the --force-xinerama flag (in your
       ~/.xsession) when starting i3, like so:

           exec i3 --force-xinerama -V >>~/.i3/i3log 2>&1

       See also docs/multi-monitor for the full
       explanation.

       Tree
           i3 keeps your layout in a tree data structure.

       Window
           An X11 window, like the Firefox browser window
           or a terminal emulator.

       Floating Window
           A window which "floats" on top of other windows.
           This style is used by i3 to display X11 windows
           with type "dialog", such as the "Print" or "Open
           File" dialog boxes in many GUI applications. Use
           of floating windows can be fine-tuned with the
           for_window command (see HTML userguide).

       Split container
           A split container contains multiple other split
           containers or windows.
 Manual page i3(1) line 46 (press h for help or q to quit)
```

```
1    no IPv6|23.5 GiB|DHCP: yes|VPN: no|W: down|E: 10.0.2.15 (100 Mbit/s)|FULL 100.00%|0.03|2017-01-26 10:02:10
```

Other Tiling WMs

i3 is the most popular tiling window manager, and for good reason: Its default configuration is sensible, it's insanely fast and light, and it has very few dependencies (so it doesn't pull in loads of libraries or graphical toolkits when you run it). But there are plenty of others worth exploring as well if you really get into the tiling groove.

Awesome [5] works well out of the box, but is designed as a "framework" window manager that can be extended using the Lua programming language. It implements many Freedesktop.org standards, so it fits in well with add-on tools like docks, program launchers, and system trays.

Dwm [6], meanwhile, is an extremely minimalist tiling window manager – indeed, the developers have set its source code a hard limit of 2,000 lines! And it's even possible to do tiling inside a terminal window, thanks to Tmux – see our guide on page 92 of *Linux Magazine* issue 193 for all the juicy details.

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*.

2017 HPC for Wall Street

Date: April 3, 2017

Location: New York City, New York

Website: *http://www.flaggmgmt.com/linux/*

Register now for HPC for Wall Street for an All-Star Conference Program! This event is designed for forward-thinking industry veterans to get the latest on cloud and data centers. Visit the website to learn more and register today.

DrupalCon Baltimore

Date: April 24–28, 2017

Location: Baltimore, Maryland

Website: *https://events.drupal.org/baltimore2017*

The Drupal community is one of the largest open source communities in the world. We're developers, designers, strategists, coordinators, editors, translators, and more. Once a year, we comes together in a US city for one of our biggest events: DrupalCon. This year we bring DrupalCon to Baltimore. Visit our website to learn more!

OSDC Berlin

Date: May 16–18, 2017

Location: Berlin, Germany

Website: *https://www.netways.de/events/osdc/overview/*

OSDC offers the opportunity to meet open source professionals and insiders and gather and share expertise over three days of presentations, hands-on workshops, and social networking. OSDC aims to "simplify complex IT infrastructures with open source" for experienced administrators and architects.

EVENTS

Icinga Camp Berlin	March 7	Berlin, Germany	https://www.icinga.com/community/events/icinga-camp-berlin/
Chemnitzer Linux-Tage 2017	March 11–12	Chemnitz, Germany	https://chemnitzer.linux-tage.de/2017/en
CeBIT 2017	March 20–24	Hanover, Germany	http://www.cebit.de/home
Maker Faire Ruhr	March 25-26	Dortmund, Germany	https://www.makerfaire-ruhr.com/
World Hosting Days Global	March 25–31	Rust, Germany	http://worldhostingdays.com/global/
SPTechCon 2017	April 2–5	Austin, Texas	http://www.sptechcon.com/
2017 HPC for Wall Street – Cloud and Data Centers Show and Conference	April 3	New York, New York	http://www.flaggmgmt.com/linux/
JAX DevOps	April 3–6	London, United Kingdom	https://devops.jaxlondon.com/
Linux-Infotag	April 22	Augsburg, Germany	https://www.luga.de/start/
DrupalCon Baltimore	April 24–28	Baltimore, Maryland	https://events.drupal.org/baltimore2017
Grazer Linux-Tage 2017	April 28–29	Graz, Austria	https://www.linuxtage.at/
Check_MK Conference #3	May 2–4	Munich, Germany	http://mathias-kettner.de/
Linux Presentation Day 2017.1	May 6	Europe-wide in numerous cities	https://linuxday.ch/index.php/en/
Open Source Data Center	May 16–18	Berlin, Germany	https://www.netways.de/events/osdc/
DevConf Panamá 2017	May 25-26	Panama City, Panama	https://www.devconfpanama.com//#/
openSUSE Conference I OSDC 2017	May 26-28	Nürnberg, Germany	https://events.opensuse.org/conference/oSC17/
ISC High Performance (ISC 2017)	June 18–22	Frankfurt, Germany	http://www.isc-hpc.com/
AnDevCon	July 17–19	Washington, DC	http://www.andevcon.com/

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.

AUTHORS

Swapnil Bhartiya	8, 16
Zack Brown	12
Bruce Byfield	50
Joe Casad	3
Mark Crutch	65
Ben Everard	65, 74, 88
Ekkehard Gentz	40
Andrew Gregory	67
Karsten Günther	28
Jon "maddog" Hall	72
Frank Hofmann	54
Heike Jurzik	44
Charly Kühnast	48
Vincent Mealing	65
Graham Morrison	82
Simon Phipps	66
MIke Saunders	68, 92
Mike Schilli	60
Tim Schürmann	20
Valentine Sinitsyn	76
Ferdinand Thommes	34

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 Editors
Amber Ankerholz, Amy Pettle

Layout
Dena Friesen, Lori White

Cover Design
Lori White

Cover Image
© Dejan Bozic, 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
Fax: +49 89 99 34 11 98

www.linuxpromagazine.com – North America

www.linux-magazine.com – Worldwide

Printed in Germany

Distributed by COMAG Specialist, Tavistock Road, West Drayton, Middlesex, UB7 7QE, 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.

Approximate

UK / Europe	Apr 03
USA / Canada	Apr 28
Australia	May 29

On Sale Date

Issue 198 / May 2017

Efficient Office

The whole reason for computers is to make life easier for humans. Next month, we study some tools for a more efficient desktop experience, including the Zim desktop wiki editor and the cool productivity tool Getting Things GNOME!

Lead Image © Rabia Elif Aksoy, 123RF.com

open
build
service

A generic system to build
and distribute packages
from sources in an automatic,
consistent and reproducible way

openbuildservice.org

openSUSE®

Lightning Fast All Flash NVMe SERVER SOLUTIONS

BigTwin™
2U/4 DP Nodes, 6 NVMe/SAS3/SATA3 per Node

1U 10 NVMe Ultra
SYS-1028U-TN10RT+

2U 48 NVMe Simply Double
SSG-2028R-NR48N

2U 24 NVMe Ultra
SYS-2028U-TN24R4T+

2U 40/20 Dual Port NVMe
SSG-2028R-DN2R40L
SSG-2028R-DNR20L

7U 42 NVMe
Datacenter Blade®

- **Supports Intel® Xeon® processor E5-2600 v4/v3 product families**
- **Over 60 SKUs Enabled**
- **Optimized Hot-Swap NVMe Designs**
- **Highest Quality**
- **The Leader in NVMe Server Development**

Intel Inside®. Powerful Productivity Outside.

Learn More at www.supermicro.com