FREE DVD


VMware vs VirtualBox

NOW INCLUDING UBUNTU user

LINUX PRO MAGAZINE

NOVEMBER 2017

# VMware vs VirtualBox

## Virtual Lab: Model a network with KVM and Qemu

## Run Android in Linux

**KolibriOS**
Lightning-fast OS with a tiny footprint

**Data Rescue**
6 tools for saving lost files

**Firefox 54**
The best Firefox ever?

**Devuan 1.0**
Debian fork without systemd

# LINUXVOICE

- Minetest
- FAQ: Arduino
- cloud-init
- Can Patent Left Save Us?
- Standing Up for Malwaretech

**FOSSPicks**
- OBS Studio
- gtop
- Bitcoin Core

**Tutorials**
- COBOL
- moreutils

Issue 204
Nov 2017
US$ 15.99
CAN$ 17.99

HETZNER ONLINE

# RYZE ‘N’ SHINE

## More parallelization. More power.


## Dedicated Root Server AX60-SSD

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

monthly $ **65**

## The ideal solution for highly-parallelizable processes.

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

**www.hetzner.de/us**

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

# ON A HIGHWAY TO ...

## Dear Reader,

The Internet is a vast and beautiful thing – our ancestors would be amazed. I probably wouldn't have my job without the Internet, and if you work with Linux, the chances are your job, either directly or indirectly, depends on the Internet as well.

People in high tech like to talk about the Internet in glowing and heroic terms. The popular view is that the Internet is not just an information highway but is actually a highway on which we are all journeying to the future.

Part of the story is that the Internet is "good business," but the recent Equifax debacle illustrates how difficult it is to determine how much the Internet actually costs. A hack on the massive consumer credit reporting company comprised 143 million identities. The problem, according to several sources, was that the company failed to install routine security updates for the Apache Struts web application framework. A vulnerability in the platform was fixed back in March, but reports indicate that Equifax didn't get around to installing the update and therefore fell prey to the attack.

So now is the time when we all collectively say "What a bunch of slackers." Everybody knows you're supposed to keep current on security patches, and on Internet-facing servers, keeping up to date is an extremely critical and solemn responsibility. Internally, the company probably has its own "What a bunch of slackers" dialog going on. Some people have probably already been fired – or they will be soon.

Firing a few Equifax employees certainly seems appropriate, but it is a little too easy. We humans have a way of focusing blame on other humans, rather than on systems. When something goes wrong, we assign the blame to a person, and then when we punish that person, we all get the feeling that we're acting decisively to address the issue. Deeper down, though, the questions are a little more complicated – and thus more scary. For instance:

- Why was this vulnerability present in the first place and how did it go undetected until March of this year?
- What other vulnerabilities are still out there now that could be the cause of future events as bad as or worse than the Equifax debacle?

I don't really know the solution to the insecurity problems that face the Internet. In fact, I'm not sure I really believe an obvious solution actually exists – certainly not something that could happen within the next 5 to 10 years – but I think we would be in a better place if we would start understanding the *real cost* of operating the Internet and investing resources to address that cost.

The rosy picture we paint about Internet efficiency and convenience creates an imaginary world where a company can hide, making business decisions based on the illusion of security rather than on gritting out the labor-intensive reality of life in a jungle.

At Apache Struts, more code reviews, more testers, and bigger bounties would have helped find vulnerabilities sooner, but who is going to pay for it? Equifax probably could have used more training and a bigger, more qualified web admin staff, but who's going to pay for it? The way a company pays for overhead is to pass the costs back to the consumer, so they would have to raise their prices and would then lose business to competitors who are willing to live dangerously and do without enhanced security measures. (Pricing on the Internet is always a race to the bottom.)

Could the government step in and mandate security inspections or timely security patching for all companies, so failure to comply wouldn't just get you fired but would get you a fine or a jail term? Certainly not the US government, which is obsessed with reducing the regulatory burden on businesses to let them be "more efficient." The system encourages businesses to stay lean and unsafe, and the cost and inconvenience of all-too-frequent failures are passed to intrusion victims.

The effects of hidden costs are weird and difficult to trace; they are off the balance sheets used by traditional accounting, but they always show up somewhere. One of the possible effects of the Equifax intrusion, which compromised names and social security numbers, is that someone could theoretically hijack your income tax return. The remedy suggested by several experts is to file your taxes early. In other words, because you do business with a company that does business with a company that underfunded its security needs, instead of filing your taxes in April (which is your right under US law), you now have to file them in January or else someone you never met will steal your tax refund.

Isn't the Internet a marvelous thing?

Joe Casad,
Editor in Chief

# LINUX MAGAZINE

## WHAT'S INSIDE

**This month** we compare VirtualBox with the free version of VMware Workstation Player and explore the possibilities for modeling a network on a host computer with KVM and Qemu. Other highlights:

- **KolibriOS** – a teeny tiny operating system that runs in memory and boots in 10 seconds or less (page 34).
- **Devuan 1.0** – Developers who didn't like the switch to the systemd init forked Debian to found Devuan. We tour the first major release (page 44).

Check out LinuxVoice (page 67) for a look at Minetest, the open source answer to the Minecraft phenomenon.

## SERVICE



## NEWS





## REVIEWS



## COVER STORIES




## IN-DEPTH







# LINUXVOICE





# On the DVD


## Manjaro Gnome 17.0 (64-bit)

Manjaro is a user-friendly system based on the independent and well-regarded Arch Linux. Arch imbues Manjaro with the spirit of simplicity, and the Manjaro developers have added enough conveniences to satisfy both beginners and power users. The Manjaro developers recently added the Gnome edition as the third officially supported flavor. The new release comes with Linux kernel 4.9 and a collection of the best Gnome and Linux applications, including Firefox, Evolution, LibreOffice Fresh, and the Lollypop and Totem multimedia players.

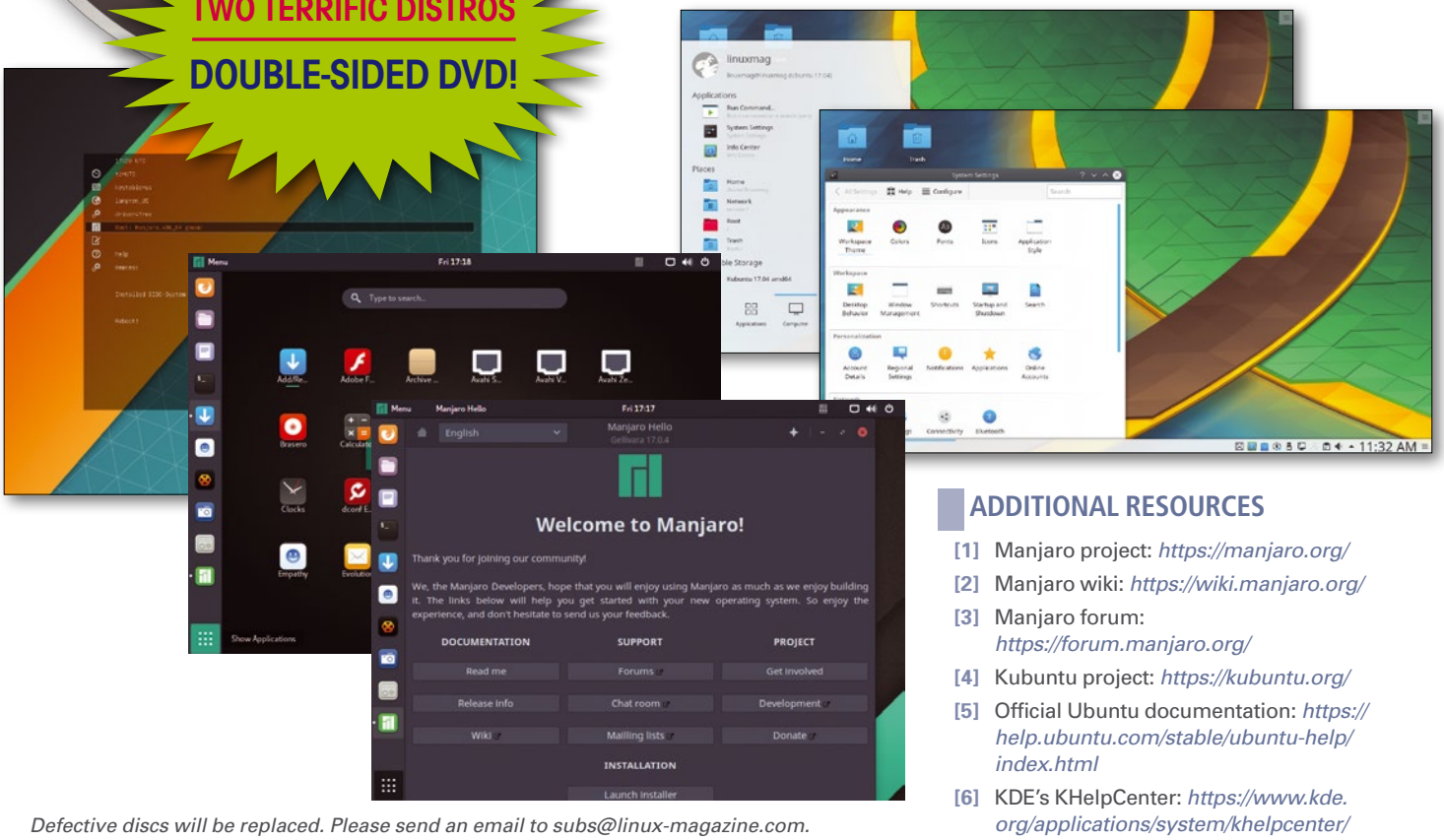
Use the arrow keys to navigate the Manjaro installer. Choose a language and then click on `Boot: Manjaro.x86_64 gnome` for the Live boot option.

## Kubuntu 17.04 (32-bit)

Kubuntu is Ubuntu with KDE's desktop and user environment. The latest edition comes with the KDE Plasma 5.9 desktop, which the developers describe as "Simple by default; powerful when needed." The new Kubuntu also includes many bug fixes and stability improvements, as well as a new 4.10-based Linux kernel and updates to Krita, Firefox, and LibreOffice.

**TWO TERRIFIC DISTROS**

**DOUBLE-SIDED DVD!**




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

## ADDITIONAL RESOURCES

[1] Manjaro project: *https://manjaro.org/*
[2] Manjaro wiki: *https://wiki.manjaro.org/*
[3] Manjaro forum: *https://forum.manjaro.org/*
[4] Kubuntu project: *https://kubuntu.org/*
[5] Official Ubuntu documentation: *https://help.ubuntu.com/stable/ubuntu-help/index.html*
[6] KDE's KHelpCenter: *https://www.kde.org/applications/system/khelpcenter/*

# Complete Your Open Source Library with Archive DVDs!

Each fully-searchable archive DVD includes past articles so you can find the content you need quickly.

**Save hundreds off the print and digital copy rate with a convenient archive DVD!**


## Order Your DVD Now!

**shop.linuxnewmedia.com**

# NEWS

Updates on technologies, trends, and tools

## THIS MONTH'S NEWS



## Reddit Closing Doors to Open Source

Reddit, the peanut gallery of the Internet, reported in a blog post that it is shutting down their open source repository on GitHub: "We're archiving reddit/reddit and reddit/reddit-mobile which are playing an increasingly small role in day to day development at Reddit. We'd like to thank everyone who has been involved in this over the years."

Reddit was open source in 2008, roughly 10 years ago, when it was a new company, because they wanted the source code of their product to be available as open source.

However, as the company grew, they found it difficult to keep up with their open source code, and their GitHub repo had not been updated for a long time. The company provided many reasons behind "doing a bad job of keeping their open-source product repos up to date."

Some reasons are legit and many others not. One such reason given by the company is that "Open-source makes it hard for us to develop some features 'in the clear' (like our recent video launch) without leaking our plans too far in advance. As Reddit is now a larger player on the web, it is hard for us to be strategic in our planning when everyone can see what code we are committing."

Companies like Red Hat, SUSE, Google, CoreOS, Docker, and others continue to innovate, yet all of their code is available as open source. The company has given many other reasons, but all fall flat compared with the way the larger open source world functions. Reddit will continue to open source some of their tools.

The good news is that Reddit is not a platform that's used by others to build their Reddit-like services, so Reddit source code might not be missed by the larger open source communities.

Lead Image © alphaspirit, 123RF.com

## VMware Brings Its Cloud to AWS

At the VMware World Conference, VMware announced the arrival of VMware Cloud to AWS. VMware will be selling and supporting the service as an on-demand, elastically scalable cloud service. This announcement is the culmination of the strategic partnership that the two companies forged last October.

"VMware and AWS are empowering enterprise IT and operations teams to add value to their businesses through the combination of VMware enterprise capabilities

and the breadth and depth of capabilities and scale of the AWS Cloud, providing them a platform for any application," said Pat Gelsinger, chief executive officer, VMware.

This service is initially available in the US West (Oregon) region through VMware and members of the VMware Partner Network.

The VMware Cloud on AWS supports custom-sized VMs to run any OS that is supported by VMware by using single-tenant, bare-metal AWS infrastructure.

"Each SDDC (Software-Defined Data Center) consists of 4 to 16 instances, each with 36 cores, 512GB of memory, and 15.2TB of NVMe storage. Clusters currently run in a single AWS Availability Zone (AZ) with support in the works for clusters that span AZs. You can spin up an entire VMware SDDC in a couple of hours and scale host capacity up and down in minutes," said AWS in a blog post.

The company also added that VMware Cloud on AWS runs directly on the physical hardware to avoid nested virtualization, while still taking advantage of a host of network and hardware features designed to support their security-first design model.

The entire stack of AWS compute, storage, database, analytics, mobile, and IoT services can be accessed directly from applications.

## Gnome Celebrates Its 20th Birthday

Gnome was started by Miguel de Icaza and Federico Mena-Quintero on August 15, 1997. The primary goal of the project was to create a fully open source alternative to KDE, which was based on the Qt widget toolkit that used a non-free licence back then.

Since its initial release in 1999, there have been 33 stable releases of Gnome to date. While Linux caters to power users, developers, and sys admins who prefer CLI, Gnome focuses on ease of use. No wonder Ubuntu, a distribution targeting PC users, picked Gnome as the default desktop environment.

Gnome has made some significant progress in the Linux desktop space with the 3.x family. They have built a distro agnostic software center that allows users of any distro to not only install and update applications, but also update the distribution itself.

Gnome also brought the capability of accessing Google Drive from within Linux desktops, a feature that's not officially supported by Google.

No wonder that even the creator of Linux, Linus Torvalds, runs Gnome as his favorite desktop.

© Bertold Werkmann, Fotolia.com

Gnome used to be the default desktop environment for Ubuntu, before Canonical introduced its own Unity shell. As a result of that decision, Gnome lost millions of users. But recently, Canonical decided to pull out of the desktop space and focus on enterprise. They ditched Unity and went back to Gnome. That means Gnome will return to millions of Ubuntu desktop users.

Happy 20th Birthday, Gnome.

## SQL Server Comes to RHEL; OpenShift Comes to Azure

Red Hat and Microsoft have expanded their partnership this week by bringing some of their core technologies to each other's platforms. The two companies have worked together to support Windows Server containers natively on Red Hat OpenShift, a Kubernetes-based container application platform.

According to Red Hat, OpenShift will be the first container application platform built from the open source Kubernetes project to support both Linux and Windows container workloads in a single platform across the multiple environments of the hybrid cloud. Red Hat will offer a technology preview in 2018. Red Hat is also bringing its OpenShift and OpenShift Dedicated to Azure, Microsoft's cloud platform.

## MORE ONLINE

### Linux Magazine
*www.linux-magazine.com*

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

**Multiprocessing in Python with Fortran and OpenMP • Jeff Layton**
Creating Python modules with Fortran OpenMP code makes all available cores accessible to Python functions.

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

**The Top Three SSH Tricks • Ken Hess**
Add extra security to your SSH service, securely copy files to and from remote computers, and configure passwordless SSH for automation scripting.

**Time-Series-Based Monitoring with Prometheus • Michael Kraus**
As Prometheus gave fire to mankind, the distributed monitoring software with the same name illuminates the admin's mind in native cloud environments, offering metrics for monitored systems and applications.

**Do It Yourself • Sven Lankes**
Oracle recently introduced Group Replication as a trouble-free, high-availability solution for the ubiquitous MySQL.

The companies also plan to collaborate on delivering enterprise performance standards and integrated support for Red Hat Enterprise Linux workloads running in Microsoft Azure Stack, an extension of Azure that brings cloud computing to on-premises environments.

The two companies are bringing .NET Core 2.0 as a container in OpenShift and Microsoft's SQL Server for Linux to Red Hat Enterprise Linux and Red Hat OpenShift.

Red Hat said in a press release that the two companies will extend the integrated, co-located Microsoft and Red Hat support to enable these new offerings across platforms.

## Google Releases Machine Learning Library

The Google PAIR (People + AI Research) initiative has released an open source, WebGL-accelerated JavaScript library for machine learning. The deeplearn.js library requires no installation or back end and runs entirely in the browser.

According to the blog post by Google engineers Nikhil Thorat and Daniel Smilkov, "While web machine learning libraries have existed for years, … they have been limited to the speed of JavaScript, or have been restricted to inference rather than training (e.g. TensorFire). By contrast, deeplearn.js offers a significant speedup by exploiting WebGL to perform computations on the GPU, along with the ability to do full backpropogation." The library lets you train a convolution neural network to recognize photos and handwritten digits "all in the browser without writing a single line of code."

© Buchachon Petthanya, 123RF.com

The deeplearn.js home page includes links to demos showing deeplearn.js at work on real-world machine learning tasks, such as classifying photos.

## FDA Recalls Nearly Half a Million Pacemakers Over Security Concerns

The Food and Drug Administration has issued a warning calling for a voluntary recall of over 465,000 pacemakers implanted in patients' bodies.

In a review, FDA found potential vulnerabilities in Abbott's (formerly St. Jude Medical) RF-enabled implantable cardiac pacemakers. Affected devices include Accent, Anthem, Accent MRI, Accent ST, Assurity, and Allure.

If the vulnerabilities are exploited, an attacker can gain access to a patient's device and then hurt the patient by either depleting the battery or triggering fatal pacing.

What's the fix? There is a firmware update that will patch these vulnerabilities. However, considering the nature of the device, you can't update it at home. The FDA said that the update requires an in-person patient visit with a health care provider.

The good news is that they won't have to operate on the patient to take out the device. It will be done wirelessly, and the whole process takes about three minutes.

"During this time, the device will operate in backup mode (pacing at 67 beats per minute), and essential, life-sustaining features will remain available. At the completion of the update, the device will return to its pre-update settings," FDA said in its advisory.

The firmware update was made available on August 29, 2017. All pacemakers manufactured after this date will have this update preloaded in the device and will not need the update.

If you or your friends and family members have pacemaker implants, please check the advisory to stay safe.

© Ljupco Smokovski, 123RF.com

# COMPLETE YOUR LIBRARY

## Order a digital archive bundle and save at least **50% off** the digisub rate!


**ORDER YOURS TODAY!**
**shop.linuxnewmedia.com**


You get an **entire year of your favorite magazines** in PDF format that you can access at any time from any device!

# Zack's Kernel News

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

*By Zack Brown*

## Container-Aware Cgroups

Roman Gushchin didn't like the way the out-of-memory (OOM) killer targeted individual processes for termination. On a system with many virtual systems on top, he said, the current OOM killer would not behave ideally. It would not recognize individual processes as belonging to particular containers, so it might unexpectedly kill some random process within the container. Or a very large container might not be recognized as a proper target for the OOM killer if it simply contained a large number of very small processes. The OOM killer might target a much smaller container instead, only because that container had a couple of large processes.

Roman wanted to address these problems by creating an OOM killer that would treat a single container as having the size of all processes running within it. Then the OOM killer might properly target that container and kill all the processes associated with it. In cases where no such containers existed, the OOM killer would fall back to its traditional per-process targeting system.

He posted a patch to implement this, but Michal Hocko objected. The real problem with the OOM killer is similar to the problem with context switching, in which the kernel switches rapidly between processes to give the illusion that they are all running simultaneously. The problem with context-switching algorithms is that different user behaviors call for different switching algorithms; the same is true for the OOM killer. There's no obviously correct way to choose which process to kill during OOM conditions.

Michal pointed this out and reminded Roman that among the kernel developers there was still no consensus about which processes the OOM killer should target in general. And he said that therefore trying to extend the OOM killer to handle cgroups might be jumping the gun.

Johannes Weiner, on the other hand, felt that Roman's patches were not dangerously related to OOM killer policy. He felt that in Roman's patches, the OOM killer was still expected to do the standard thing – identify which process to kill, according to its existing set of policies – but under Roman's patches, the OOM killer could simply consider a whole container as a single process and make its assessment the same as it would for any other process.

As Johannes put it, "All we want is the OOM policy, whatever it is, applied to cgroups."

But Balbir Singh agreed with Michal. He started to pose algorithmic policy questions related to how best to assess a given container as being a good or bad target for the OOM killer.

But Johannes explained, "The problem is when OOM happens, we really want the biggest *job* to get killed. Before cgroups, we assumed jobs were processes. But with cgroups, the user is able to define a group of processes as a job, and then an individual process is no longer a first-class memory consumer." He went on, "Without a patch like this, the OOM killer will compare the sizes of the random subparticles that the jobs in the system are composed of and kill the single biggest particle, leaving behind the incoherent remains of one of the jobs. That doesn't make a whole lot of sense."

Without talking past each other, it became clear that the two sides of the discussion were interested in different

things. The pro-patch folks wanted containers to be treated as discrete killable jobs, because killing a random part inside one of them might leave the whole container in an unworkable state, whereas the anti-patch folks – Balbir in particular – were more concerned with finding ways to avoid hitting an OOM condition in the first place. They wanted to find ways to handle memory that would be less likely to overcommit RAM. As Balbir put it, "OOM is a big hammer and having allocations fail is far more acceptable than killing processes. I believe that several applications may have much larger VM than actual memory usage, but I believe with a good overcommit/virtual memory limiter the problem can be better tackled."

At a certain point, Vladimir Davydov entered the discussion, with an objection to Roman's patch that was more along the lines of the kind of feedback Roman had probably hoped for initially. Vladimir said:

*I agree that the current OOM victim selection algorithm is totally unfair in a system using containers and it has been crying for rework for the last few years now, so it's great to see this finally coming.*

*However, I don't reckon that killing a whole leaf cgroup is always the best practice. It does make sense when cgroups are used for containerizing services or applications, because a service is unlikely to remain operational after one of its processes is gone, but one can also use cgroups to containerize processes started by a user. Kicking a user out for one of her process[es] has gone mad doesn't sound right to me.*

*Another example when the policy you're suggesting fails in my opinion is in case a service (cgroup) consists of sub-services (sub-cgroups) that run processes. The main service may stop working normally if one of its sub-services is killed. So it might make sense to kill not just an individual process or a leaf cgroup, but the whole main service with all its sub-services.*

*And both kinds of workloads (services/applications and individual processes run by users) can co-exist on the same host – consider the default systemd setup, for instance.*

Because of these two equally valid possibilities, Vladimir suggested allowing the user to choose which policy they preferred. He said, "we could introduce a per-cgroup flag that would tell the kernel whether the cgroup can tolerate killing a descendant or not. If it can, the kernel will pick the fattest sub-cgroup or process and check it. If it cannot, it will kill the whole cgroup and all its processes and sub-cgroups."

Roman saw a lot of value in Vladimir's scenarios. But he was hesitant to create a per-cgroup flag that would have to be supported in all future kernels. Presumably eventually such a thing would no longer be needed, so it would be good to avoid having the unnecessary flag available in the kernel for all time. But he did agree that there should be an option to disable the cgroup-aware OOM killer on a system-wide basis, if only for backward compatibility purposes.

The debate continued, especially between the "don't change policy" and the "we're not changing policy" positions. Ultimately it doesn't seem to be as much like bickering as it might appear. The folks protesting against policy changes are very sensitive to policy changes because that's the part of the code that concerns them, and they see the subtle ways that Roman's patch does actually change the way the OOM killer targets processes. The folks claiming Roman's patch doesn't implement any policy changes are sensitive to the plight of the virtual system, and they just don't want the OOM killer doing something that seems nonsensical, like destroying a core component of a container, leaving the rest of the container unusable, but without actually freeing up the memory used by the now-useless container.

## A Different Type of RAM Chip on a Single System

Ross Zwisler of Intel pointed out that modern devices would typically have multiple different kinds of RAM associated with a given CPU, and that the kernel needed to handle that properly. He said, "These disparate memory ranges will have some characteristics in common, such as CPU cache coherence, but they can have wide ranges of performance both in terms of latency and bandwidth." He went on, "consider a system that contains persistent memory, standard DDR memory and High Bandwidth Memory (HBM), all attached to the same CPU. There could potentially be an order of magnitude or more difference in performance between the slowest and fastest memory attached to that CPU."

He said the problem was that "NUMA nodes are CPU-centric, so all the memory attached to a given CPU will be lumped into the same NUMA node. This makes it very difficult for userspace applications to understand the performance of different memory ranges on a given CPU."

He suggested adding sysfs files to indicate performance characteristics of given memory ranges. Then user applications could choose the memory they really wanted to use for a given allocation.

To do this, Intel saw two possible options. Either they could directly export data into sysfs from the newly implemented Heterogeneous Memory Attribute Table (HMAT) code, which contains information about memory, or they could use a library and a running daemon to provide users with an API to access the same data via function calls. Or maybe, Ross said, there was a third way the Linux developers might prefer over either of those options.

There were no serious objections to Ross's proposal, nor did anyone seem to have a strong opinion on the best way to expose the data to user code. Mostly, folks just seemed interested in the whole issue, the types of devices, the use cases, and whatnot.

Bob Liu raised the only dissenting voice, suggesting that most users probably wouldn't care which piece of RAM they allocated, so he wouldn't want user code to be forced to pay attention to these various new APIs and exported sysfs files, to which Ross replied that there would be a decent set of default policies, so that only the users who did care about which memory they wanted would need to pay attention to the new interfaces.

This seems like one of those moments where Intel – or some other hardware maker – is doing something in hardware that obviously needs to be supported because the alternative is just not to support it; they're trying to let the kernel folks know that this is going to be a thing. So this whole thread seemed essentially like a little "heads-up" to the

kernel developers that something's coming and folks should be ready when the patches arrive. There's also an aspect of Intel covering its butt, making sure that the direction they're going internally with the patches is less likely to violate some kind of unexpected kernel requirement.

## New SARA Security Framework

Salvatore Mesoraca posted a patch to implement SARA (short for "SARA is Another Recursive Acronym"), a new general-purpose security framework intended to let users build security sub-modules to match particular needs. One sub-module Salvatore included was a USB filtering system to better control the kind of data that could pass along a USB connection. Another was a WX protection system to ensure that a piece of memory could be either writable or executable, but not both. Other sub-modules could be designed to meet other specific needs.

Mickaël Salaün suggested merging the WX protection sub-module with his own security project, TPE (Trusted Path Execution)/shebang LSM. He explained that TPE could prevent a user's binaries and even scripts from being executed, but he said, "there is always a way for a process to mmap/mprotect arbitrary data and make it executable, be it intentional or not." He suggested that Salvatore's SARA framework could "make exceptions by marking a file with dedicated xattr values. This kind of exception fit well with TPE to get a more hardened executable security policy (e.g. forbid an user to execute his own binaries or to mmap arbitrary executable code)."

Salvatore agreed that the two projects complemented each other, although he saw some difficulties with merging the two projects, in particular their very different configuration systems. SARA implemented its own, whereas TPE relied on `xattrs`. Salvatore said he wouldn't mind if TPE implemented the SARA configuration system, but he was loath to abandon his own system in favor of `xattrs`. He didn't really see a need to merge them, since they could both be used side by side without any cost.

Close by, Matt Brown announced that he'd merged his own TPE work with Mickaël's and added some additional shebang features.

The discussion veered off into technical implementation details. Clearly there's some motivation for these various security systems to merge together, although it seems to be too soon to point to any final front end as the most likely interface.

## Improving GPIO Interrupt Handling

Jerome Brunet asked Linus Torvalds and a lot of other people for guidance on some patches he wanted to write. The problem was how to have general-purpose input/output (GPIO) controllers provide a working IRQ to all GPIOs simultaneously. GPIO is a physical pin on a chip, with no particular purpose hard-coded into it, except that it can be used for either input or output as desired by the code using it. The interrupt request (IRQ) is a hardware signal used to interrupt whatever's happening (in this case whoever's using the GPIO pin) and to make something else happen instead.

If the Linux kernel is not able to interrupt something when it needs to, then it can't do process context switching in a timely fashion, and the user experience gets choppier. Fine-grained context switching is a crucial part of the Linux kernel, and uninterruptible operations are ideally always kept to a minimum.

Many hardware chips, Jerome said, had no trouble providing working IRQs to all GPIOs simultaneously. But some, like the Amlogic system on a chip (SoC), had trouble, because they didn't have enough IRQs available to match the number of GPIO pins.

To handle these cases, there would have to be some code to handle the GPIO resource properly, for example by manually associating particular interrupts with particular GPIOs on the fly. However, Jerome didn't like this possibility for a couple of reasons. For one thing, the mechanism for doing it was itself not interruptible, so the benefit to context switching was decreased. For another thing, once the code made the association between an interrupt and a GPIO, there was no corresponding mechanism to free up that association. It would just hang around even if it wasn't used anymore. So as Jerome put it, "this approach leaks mappings." He also pointed out that there were already quite a number of GPIO drivers using this approach, which indicated even more strongly that some kind of solution should be found.

Another solution, he said, was "to create an empty domain to get a virq for each pin but delay the setup of the interrupt hierarchy to the 'irq_request_resources' callback. Testing this approach led me to some nasty race conditions in the interrupt handlers. I discussed this solution with Marc [Zyngier] and he told me that doing the setup of the interrupt hierarchy in the irqchip callbacks is 'too late'."

Given the problems with each of these possibilities, Jerome felt there was no clean solution at the moment. He offered a third alternative that would require a fair bit of implementation. First, he would define a pair of prepare/unprepare functions, which any given GPIO driver could implement if needed. He would also add new functions to the GPIO kernel API that would implement reference counting in order to share scarce IRQs among multiple GPIOs. Finally, he'd go back to existing GPIO user code in the kernel and update them to use the new interface.

He emphasized that the new approach would be a complete no-op for any drivers that didn't need it. But for those that did, it would provide the working IRQs to all GPIOs simultaneously.

Linus Walleij, Grygorii Strashko, and Marc Zyngier immediately dove into a technical implementation discussion with Jerome, with various strong objections and new directions proposed. Ultimately, there was absolutely no consensus reached. But it's clearly something a lot of people care about, because it means the difference between cleanly supporting Linux on a piece of hardware, or not. ■■■

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**.

### Comparing VirtualBox and VMware Workstation Player

# Virtual Shootout

**VirtualBox and VMware Workstation Player are popular virtualization options that won't cost you a penny. Which is best for you?** *By Thomas Leichtenstern*

For many users, virtualization is a daily part of the computing experience. Test an application? Switch to a different OS without shutting down? Run a program that only works in Windows? The possibilities are endless.

Several popular options exist for supporting virtualization on a Linux workstation; two of the most popular alternatives are VirtualBox and VMware. The common conception is that VirtualBox is a free tool and VMware is a commercial solution with a price tag. The reality is a bit more complicated. VMware does make a no-cost version of their VMware Workstation solution called VMware Workstation Player [1]. VMware Workstation Player is only free for non-commercial uses; if you want to use it for business, you'll need to buy a license.

VirtualBox [2] is indeed a free and open source virtualization solution, but certain advanced features are collected into an extension pack that you'll need to install separately [3]. You can use the extensions without cost for non-commercial purposes. The project website is a little vague on what to do if you want to use the VirtualBox extensions for commercial purposes, but the answer appears to be the VM VirtualBox package from VirtualBox parent company Oracle [4].

If you're tired of dual-booting or switching computers whenever you need to access a different OS, and you're shopping for a virtualization alternative, read on for a look at VirtualBox and VMware Workstation Player. This article compares the free versions of both tools, although you'll also find some notes about the add-on extension packages. Keep in mind, however, that VirtualBox is free as in free speech as well as in beer (with a GPLv2 license) and VMware Workstation Player is free for personal use but does not have a FOSS-style free license.

## Installation: VirtualBox

See the box entitled "Testing Environment" for a description of the system used for the tests in this article.

Popular distributions such as Ubuntu or Debian may already provide VirtualBox in their package repositories, but the repositories usually don't have the latest version. You can get the latest VirtualBox from the project website [2].

### TESTING ENVIRONMENT

The operating system used in the test was Linux Mint 18.1 "Serena" KDE (64 bit), which is based on Ubuntu 4.16 LTS. The system used Linux kernel 4.4.0-53-generic, KDE Plasma 5.8.5, and X.org 1.18.4. Windows 10 Home (64 bit) served as the guest.

The test computer equipment included an Intel Core i5 (2 cores 3193 MHz), 8 GB of DDR3 RAM, an ATI Radeon HD 5670, and a Samsung SSD 750 EVO with 500GB of storage space on a SATA-II connector.

On most systems, you can download the package and click on it in the file manager, and the local package management system will automatically resolve missing dependencies and set up the software. Alternatively, set up the package, together with the `libcurl3` dependency, on the command line (Listing 1). If VirtualBox installs properly, you'll see an entry in the start menu under *System | Virtual Machine* or *Oracle VM VirtualBox*.

A more elegant approach is to integrate the VirtualBox repository provided by Oracle with your system. Using the official repository will put you in line to receive future updates without waiting for your distribution to integrate the new version. Open a terminal and type in the command from the first line of Listing 2. Import the required key with the commands from lines 2 and 3. After updating the sources (line 4), set up VirtualBox with the command from line 5.

The basic VirtualBox setup lacks some features, such as USB support. For this extended functionality, you need the proprietary extension pack, which you can still use free of charge for non-commercial purposes. After downloading the extension package, open *File | Settings* in VirtualBox and select the *Additional packages* section in the new window. Click on the little blue box with the down arrow, and navigate to the directory where the extension pack is located.

**LISTING 1: Installing VirtualBox**

```
$ sudo apt-get install libcurl3
$ sudo dpkg -i virtualbox*.deb
```

## Installation: VMware Workstation Player

VMware makes the VMware Workstation Player application available for download in generic form with the `.bundle` format [1]. At the time this article was written, the current version was 12.5.6. After downloading, open a terminal and launch the installation by entering:

```
sudo bash Vmware-Player-<version_number>.bundle
```

A wizard will take over and manage the installation (Figure 1).

A closing dialog in the installation wizard asks for a license key. If you do not have a key, skip this. When the

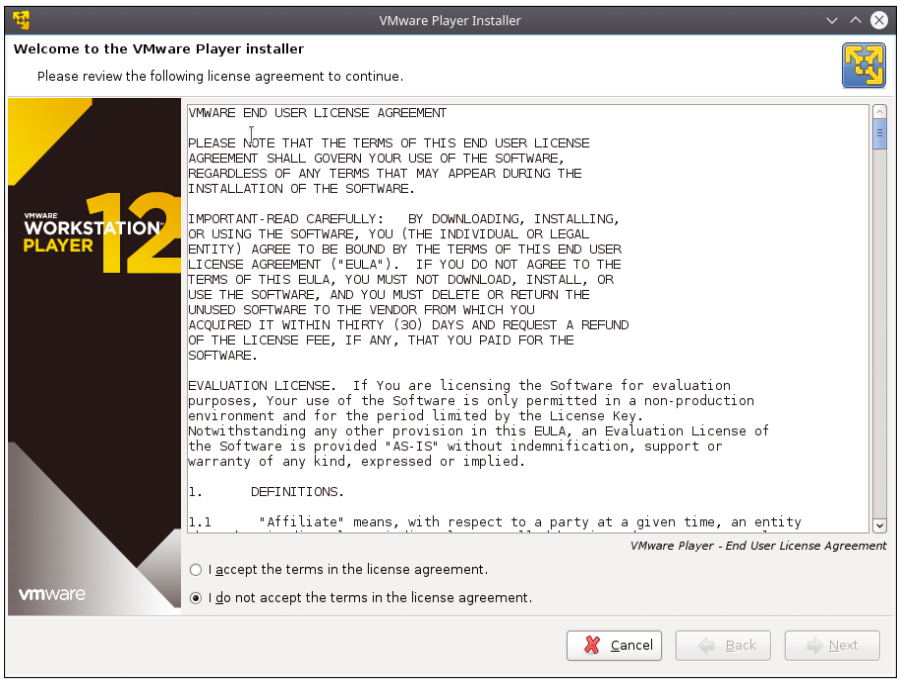



**Figure 1:** Launch the shell script in the console, and a wizard takes over the VMware Player setup.

**LISTING 2: VirtualBox Repository Install**

```
$ sudo echo "deb http://download.virtualbox.org/virtualbox/debian <Ubuntu-Version>
            contrib" >> /etc/apt/sources.list
$ sudo wget -q https://www.virtualbox.org/download/oracle_vbox_2016.asc -O- |
  sudo apt-key add -
$ sudo wget -q https://www.virtualbox.org/download/oracle_vbox.asc -O- |
  sudo apt-key add -
$ sudo apt-get update
$ sudo apt-get install Virtualbox
```

software is launched for the first time, the input window for the license key will reappear.

If you are using the software for your personal use only, it is enough to enter an email address for the license key. However, by entering your address, you automatically agree to receiving advertising. Commercial use of the software requires a license, which costs around $149.99 or EUR130.

VMware Workstation Player's main window is similar to VirtualBox (Figure 2). Use the menubar to set global parameters for the program; guest-specific options are right next to the list of guest systems. However, it is worth noting that the free version of Player has far fewer options than VirtualBox.

## New Systems

Both Player and VirtualBox include a self-explanatory dialog for creating guest systems. In each case, the dialog asks for the intended operating system and the size of the virtual disk. Virtually all major operating systems support both VirtualBox and VMware Workstation Player, including Windows, Linux, BSD, Solaris, and Novell Netware. Mac OS X interacts with VirtualBox but does not work with VMware Workstation Player.

In Player, you can complete the detailed configuration via *Edit virtual machine settings* once you have finished the basic program setup. In the *Hardware* tab (*Virtual Machine* | *Settings*), you will find settings for the network, the USB controller, the sound card, and the size of main memory.

The *Options* tab contains, among other things, settings for remote access, shared folders, and start behavior. The *Hard Disk* options are remarkably extensive (Figure 3): You can mount a virtual disk in the host, defragment, shrink, or enlarge. You will not find these features in the VirtualBox graphical interface. The exceptionally powerful command-line tool `VBoxManage` adds some additional options for VirtualBox users [5] if you are comfortable working in a terminal window.

In VirtualBox, you can set up the host system by clicking on the *Settings* icon. In the *Display* section, set the size of memory of the virtual graphics card, the scaling of the host window, and the number of screens.

**LISTING 3: Forcing BIOS Setup in Player**

```
bios.forceSetupOnce = "TRUE"
```

VMware Workstation Player allows the addition of up to ten network interfaces in three modes (*NAT*, *Bridged*, and *Host-only*) in the *Network Adapter* section. VirtualBox serves a maximum of four virtual interfaces that support five different modes in the *Network* section. Three of them (*NAT*, *Host-only*, *Bridged*) correspond to modes that are available on Player and play the largest role in real-world operations. VirtualBox provides two other adapters that are missing from Player.

Player provides a special function that lets you specify what bandwidth the virtual machine may use for inbound and outbound network traffic. This function ensures that the guest does not negatively affect the host operation. After activating the desired card, click the *Advanced* button, and a new window will open.

In addition to the parameters that can be accessed via the configuration dialog, VMware Workstation Player also offers other parameters in the BIOS. You can change, for example, the order of booting or USB settings. You need nimble fingers to open the virtual BIOS: The POST appears for just a split second, during which you must press F2. An entry in the configuration file (`.VMX`) of the virtual machine (Listing 3) makes sure that the BIOS setup appears first when launching the machine (Figure 4).

## Guest Extensions

Both VirtualBox and VMware Workstation Player provide guest extensions for Windows and Linux. These extensions include specially adapted drivers that better integrate the guest system into the virtual machine. In addition, the extensions provide advanced features such as drag and drop between guest and host.

In VirtualBox, you can find these extensions in the launched system under *Devices* | *Insert Guest Additions CD image*. If you select this item, the software mounts an ISO image with the drivers included and makes it available as a CD drive. VMware Worksta-

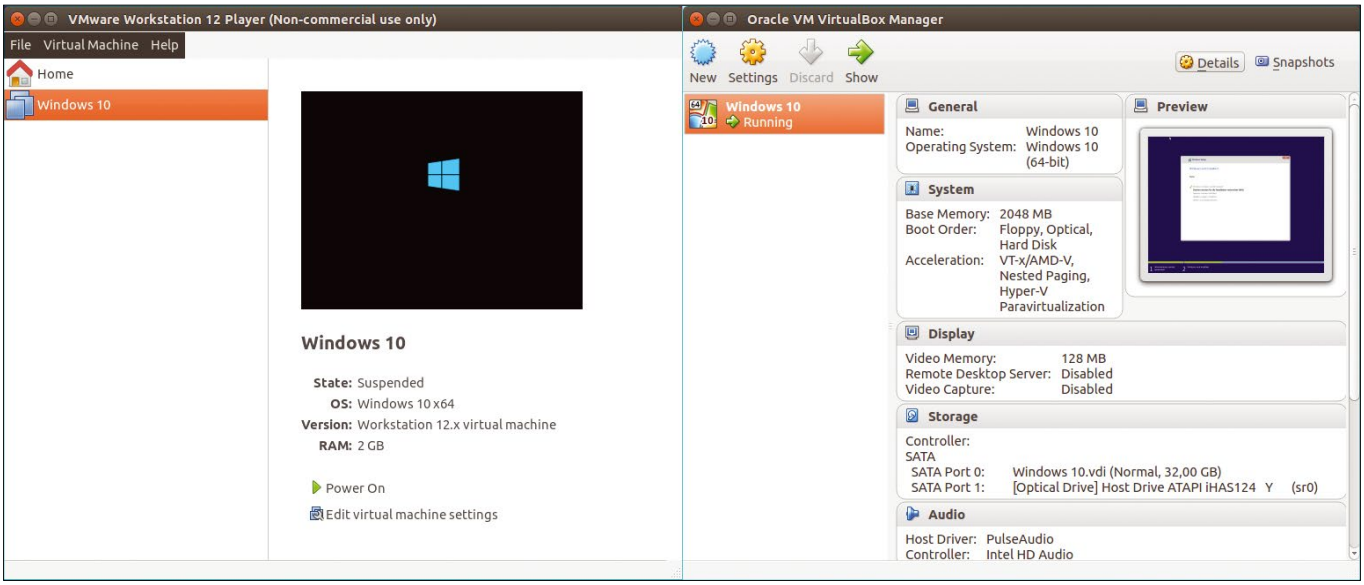



**Figure 2:** The main windows of VMware Workstation Player (left) and VirtualBox (right) appear very similar. The left side of the window contains the list of guest systems and the right side shows the guest configuration.

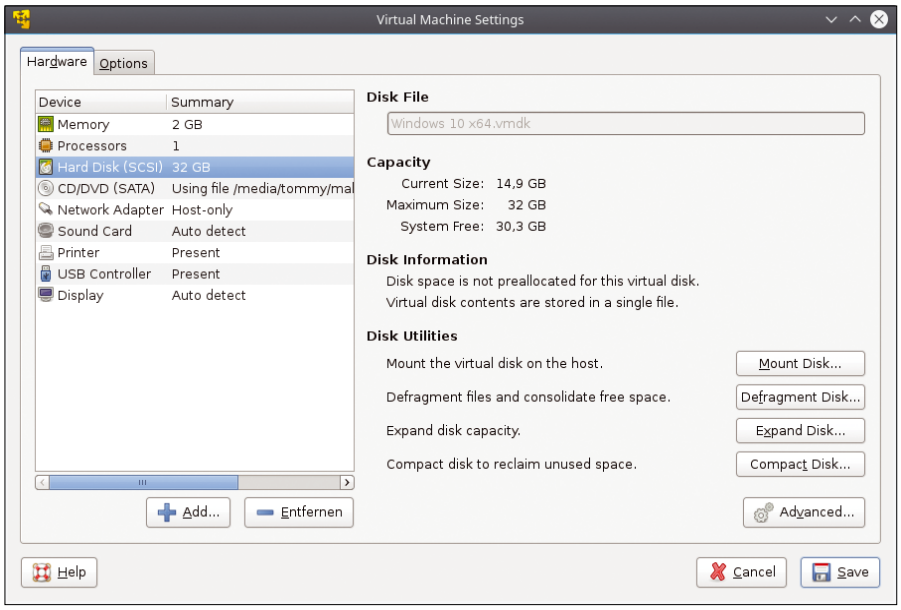



**Figure 3:** Beyond the sparse main window, VMware Workstation Player provides a variety of options for guests to customize the system's hardware.

tion Player has a similar approach, but the package is called *VMware Tools*, and it is available in the menu under *Virtual Machine*.

With Player, the boot time increased from around 16 seconds to almost 30 seconds after installing the extensions – no comparable effect occurred in VirtualBox.

## Virtual Systems

Bootable CDs or DVDs, as well as locally stored ISO images, are suitable for setting up a new virtual system in both applications. VirtualBox and VMware Workstation Player also support booting via Preboot Execution Environment (PXE) on the local network. However, both candidates fail when booting from a USB drive.

You can add support for USB boot with the Plop Boot Manager [6], for which you will find a detailed description on YouTube [7].

As an alternative to manual configuration for a virtual machine, fully configured guest images are available for download at various sources on the web. Many of these preconfigured guest images are equipped with special features, such as Nextcloud, WordPress, or Joomla [8]. But you will also find plenty of classical systems [9].

Both applications support the Open Virtualization Format (OVF, OVA), an open format for packaging software in virtual images. To open a virtual machine, click on *Open a Virtual Machine* in Player; under VirtualBox, navigate to *Import Appliance* in the *File* menu. Only VirtualBox allows the export of installed virtual machines.

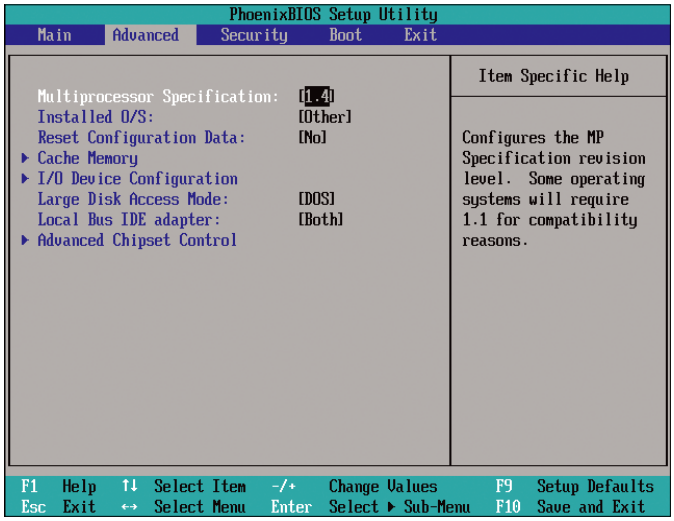



**Figure 4:** The VMware Workstation Player virtual machine even supports virtual BIOS configuration.

## Data Exchange

Both programs offer a range of possibilities for exchanging data between the guest and host. Both use the network or CIFS/SMB, and shared directories are mounted accordingly in the host.

The transmission rate for VirtualBox was over 60MBps on average from the mapped directory to the host system – and more than 100MBps in the opposite direction. The transfer in Player ran a tad faster; I copied the file in both directions at over 100MBps in the test.

Another way of exchanging data is to connect to a USB drive. Both programs support the 1 to 3 defaults and allow devices to be mounted and unmounted during operation. With VirtualBox, an important detail should be taken into account, because otherwise the system will not detect any USB devices at all (see the box titled "USB in VirtualBox").

The 2.5-inch removable drive from Western Digital used in the test has a capacity of 2TB and supports USB 3.0. Transfers to the host system can be done in both directions at about 30MBps. In Player, the measured speed was about 20MBps when transmitting in both directions. In the VirtualBox, the drive reads and writes at the same transfer rate, however, the rate was significantly behind the Player rate at 14MBps (see the box titled "Benchmark" and Table 1.)

Both VirtualBox and VMware Workstation Player support the use of a shared clipboard for the guest and the host, which makes it easy to copy text passages or URLs back and forth between the systems.

Player even goes a step further and allows the transfer of arbitrary data via copy and paste. You can also transfer data

### USB IN VIRTUALBOX

VirtualBox creates the `vboxusers` group, but it does not add any users to the group, which means you cannot access USB devices under the basic settings. Use the command in Listing 4 to add users to the group. Logging into the system again will activate the changes.

### LISTING 4: Adding Users to the vboxusers Group

```
# usermod -a -G vboxusers <user_name>
```

**TABLE 1: Comparing Values**

| Discipline | VirtualBox | VMware Workstation Player |
|---|---|---|
| Installation (operational and logged in) | 11:41 min. | 9:03 min. |
| Boot time until login (without extension*) | 25 sec. | 16 sec. |
| Boot time until login (with extension*) | 18 sec. | 27 sec. |
| Shutdown | 11 sec. | 17 sec. |
| Reading USB | 20MBps | 20MBps |
| Writing USB | 14MBps | 20MBps |
| Shared folder (reading) | 60MBps | 100MBps |
| Shared folder (writing) | 100MBps | 100MBps |

* Extensions: VirtualBox guest extension/VMware tools

## BENCHMARK

The virtual machines used identical parameters for the tests: A CPU, 2GB of RAM and 30GB of disk space. To rule out the influence of activity on the network, such as the automatic download of updates, the network interfaces were disabled. The tests revealed a fairly balanced picture: Both programs performed at roughly the same speed. However, Player's launch time significantly increased after installing the guest extensions, which was not the case with VirtualBox.

using drag and drop: Files of any type can easily be transferred from the guest to the host and back.

VirtualBox struggled with some significant clipboard issues: It only copied snippets of text from the clipboard but no files. When moving files from the guest to the host using drag and drop, a dialog opened with the options to unpack or link the file. The unpacking option had a tendency to crash the Dolphin file manager, but creating a link seemed to work.

Moving files from host to guest worked without any issues in the test.

## Integration

Virtual machines often provide various features for the desktop to make the handling of the guest system as pleasant as possible. These features include, for example, adjusting the size of the virtual desktops to the surrounding window: If you change the size of the window, content proportionally changes. This technique works easily with both programs (Figure 5) but requires the installation of guest extensions.

VirtualBox had several setbacks in full-screen mode: A small menubar appeared at the bottom of the screen in the guest, but it was not operational. The change between the scaled and full-screen mode led to strange and difficult-to-reproduce effects. Sometimes, everything seemed to be working smoothly; other times, the application minimized the screen to the size of the program window, and the host desktop occasionally froze.

Full-screen mode, as well as the switch to the window view, worked better in Player. The Help menu was also easy to use.

VirtualBox offers a seamless mode, in which the software displays a guest-launched application as a window on the host desktop and the virtualization application displays the guest start menu to run other programs. In our case, this seamless mode didn't work. On one occasion, the software partially displayed the Windows menu in the file explorer; another time the virtual machine crashed. Seamless mode is not available in VMware Workstation Player, although you could get something similar if you upgrade to a commercial-grade VMware version.

## Additional VirtualBox Features

VirtualBox has several other features that are not available in the free version of VMware Workstation Player, although some of these features are available the commericial VMware Workstation.

VirtualBox allows the encryption of a virtual machine with AES128 or AES256. The encryption takes about two minutes for a 30GB virtual drive. However, the negative impact proved to be considerable: Adding encryption increased the boot time from 16 to over 45 seconds, and the system ran significantly slower than before.

VirtualBox also lets you create a system snapshot, which you can later restore (Figure 6). A snapshot allows for the safe testing of new software – one click is enough to restore the original state. The restoration only succeeds in the switched-off state.

VirtualBox has a built-in recording function for documenting processes. You can enable recording under *View | Video Capture*. In the default setting, the software saves videos in WebM for-

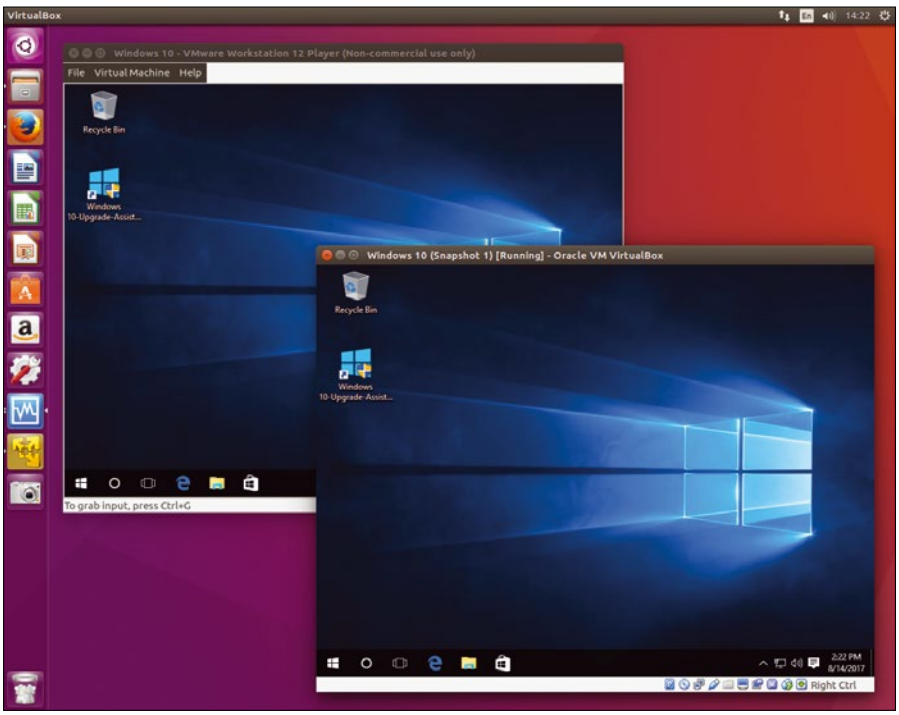



Figure 5: In *Automatically adjust the guest display* mode, the desktops of the guest systems comply with the size of the surrounding window. Player is in the foreground; VirtualBox is behind.

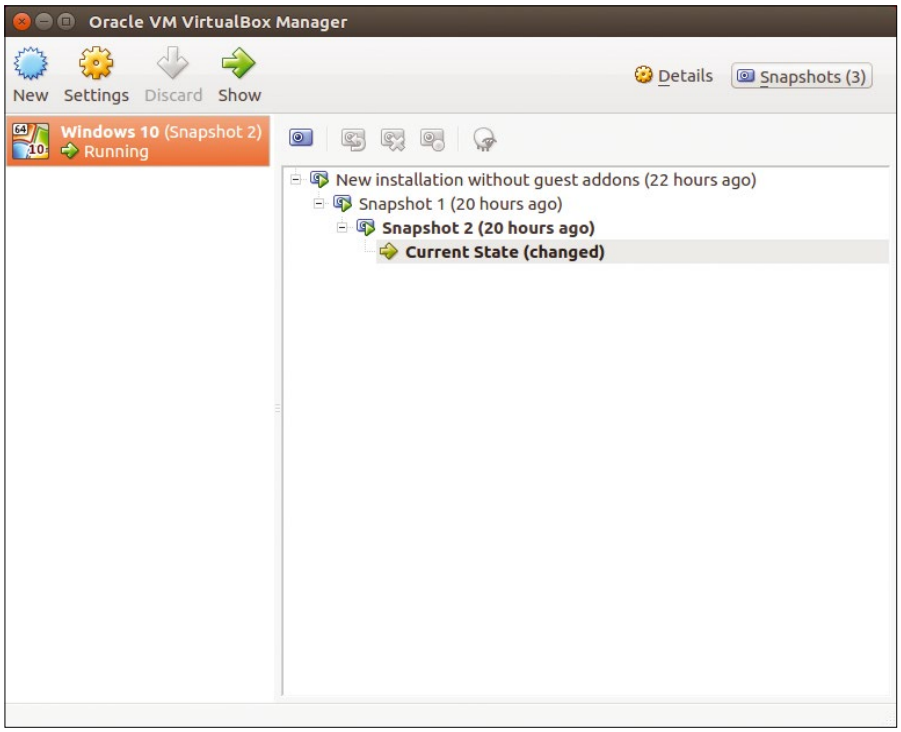



Figure 6: **VirtualBox allows you to create snapshots that will help you restore a virtual system to a safe state.**

mat in the directory of the running virtual machine.

## Conclusions

Looking for a simple-to-use and robust virtualization tool? You certainly can't go wrong with the VMware Workstation Player. The program may have relatively few options, but it is easy to configure, stable, and well designed. If you need the broader range of functions associated with VirtualBox, you can access the VMware Workstation commercial version, which costs around $250 (EUR200).

VirtualBox displayed a wider range of additional functions, but it seemed to have more problems. The previously well-functioning seamless mode was practically impossible to use, the encryption of the virtual machine reduced its pace considerably, and drag and drop from the guest to the host only partially worked. On the other hand, snapshots support and powerful tools for the command line speak in favor of using VirtualBox. ■■■

## INFO

[1] VMware Workstation Player: *https://my.vmware.com/web/vmware/free#desktop_end_user_computing/vmware_workstation_player/12_0|PLAYER-1257|product_downloads*

[2] VirtualBox: *https://www.virtualbox.org*

[3] VirtualBox Extension Pack: *http://download.virtualbox.org/virtualbox/5.1.28/Oracle_VM_VirtualBox_Extension_Pack-5.1.28-117968.vbox-extpack*

[4] Oracle VirtualBox Enterprise: *https://shop.oracle.com/apex/product?p1=OracleVMVirtualBoxEnterprise&p2=&p3=&p4=&p5=&intcmp=ocom_virtualization_vmvirtualboxenterprise*

[5] VBoxManage: *https://www.virtualbox.org/manual/ch08.html*

[6] Plop Boot Manager: *https://www.plop.at/*

[7] Booting from USB drives: *https://www.youtube.com/watch?v=aSS1bOF7sDA*

[8] Bitnami Images: *https://bitnami.com/stacks*

[9] OSboxes: *http://www.osboxes.or*

## KVM virtualization with Qemu and Aqemu

# Down in the Kernel

**KVM and Qemu provide a fast and powerful alternative to VirtualBox for virtualization in Linux.** *By Erik Bärwaldt*

Linux users often equate virtualization with VirtualBox, a versatile virtualization app that Oracle has maintained for several years [1]. VirtualBox is a favorite with both beginners and power users because of its convenient installation and simple configuration.

With all the recent attention to VirtualBox, many users don't realize that a fully developed virtualization alternative is already built into the Linux kernel. Kernel-based Virtual Machine (KVM) has been part of the kernel since Linux 2.6.20 and is developed by a team at Red Hat [2]. Combining KVM with the Qemu hypervisor system and the Aqemu graphical interface can lead to greater flexibility and significantly faster virtual machines (VMs) than a solution based around VirtualBox.

## Operating Principle

KVM is actually a kernel module that acts as an operating system interface and, therefore, relies on a virtualization environment. Qemu [3], which is available in all major distributions, adds the necessary application-level virtualization features and is the perfect complement to KVM.

KVM/Qemu requires a CPU that supports hardware virtualization. If you use a computer with an Intel processor, you can enter the following command:

```
grep --color vmx /proc/cpuinfo
```

to find out whether the processor supports Intel VT. The `color` option highlights the `vmx` flag. If you have a computer with an AMD processor, the command sequence is similar:

```
grep --color svm /proc/cpuinfo
```

If you do not see the appropriate flags, and you suspect the CPU really does support hardware virtualization, take a look at the BIOS settings. Some BIOS versions let you switch the virtualization function on and off manually.

See your distribution's package manager for information on how to install Qemu. When you set up the KVM/Qemu duo, you should also install the Spice protocol and the QXL graphic driver, or else the system will only emulate an ancient Cirrus graphic card that does not support usable screen output on high-resolution monitors. Both components are also available in the repositories of all major distributions.

Aqemu [4] is a graphical environment for managing VMs. The Aqemu software is based on the Qt libraries and is strongly adapted to Oracle's VirtualBox. Aqemu is usually found in the repositories, so you can set it up along with the dependencies using

the package manager. Also, the application is available for download at SourceForge [5].

## First Steps

When you first launch Aqemu, a dialog guides you through the setup. In the course of the setup, the software scans the system for the active versions of KVM and Qemu and displays them in a table. If one of the two packages is missing, you need to check the installation.

After configuring all parameters, the routine jumps to the main window, which is very like VirtualBox: After creating the VM, on the left, you will find names listed in a table. On the right, Aqemu shows the settings in several dialogs, each grouped into tabs. Above is a horizontal bar with buttons for quick selection of the most important functions. A menu arranged horizontally at the top of the window rounds off the elements of the application.

## Settings

Aqemu displays the complete KVM/Qemu instruction set and combines it with the elements of a graphical interface. The dialogs for setting the parameters are extensive. You can find several configuration options in the File menu under the General Settings and Advanced Settings submenus. The basic settings let you set the path for the VMs and configure some options for the appearance of the program.

In addition, you can define a default template if necessary. The default template gives you specifications for creating VMs in the individual dialogs, although you can adjust these settings retroactively.

In the Advanced Settings menu, you can tweak settings for hardware emulation and specify the path to the log files. In *Advanced Settings | Information in Info Tab*, you can activate further options for display as needed (Figure 1).

## Creating VMs

To create a VM, first go to the bar on the left with the buttons and click on the icon with the magic wand. This step calls up a wizard that will help you to set up a functional VM in a few steps. The options in the dialog tabs for a simple system correspond to those of the defined template.

After completing the wizard, you now need to assign a boot medium to the new machine. The wizard creates a virtual hard disk according to your specifications. Usually, the boot medium is either an optical disc or an ISO image. In the *CD/DVD/Floppy* tab, enable the optical drive by checking the box next to the *CD/DVD-ROM* option. In the selection field below, select an appropriate image or a physical drive.

In the *General* tab you can adjust the keyboard layout. Then launch the VM by clicking on *Play*. The new system launches in a second window customized in size and resolution. If it responds particularly slowly, you will need to adjust other options after finishing the session.

The wizard sets the RAM size of the new VM to just 256MB. For many distributions, it is recommended to set a higher value in the *General* tab – the physical memory needs to be around 2GB or more for a typical Live system. For compact distributions such as SliTaz or Puppy Linux, 256MB will suffice.

Aqemu limits the number of usable core processors in the specification to just one. If the host has a modern CPU with HT technology enabled, you can assign two or more cores to the VM in the *General* tab beside *Number of CPU*.

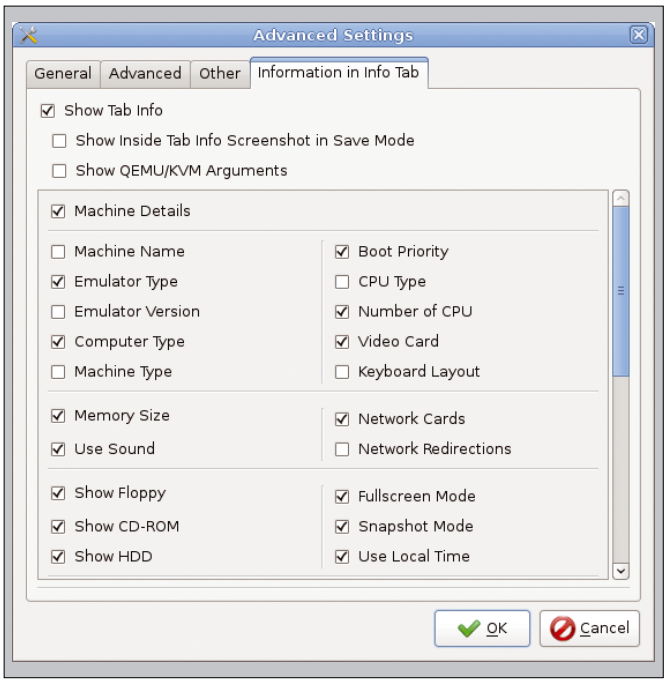



**Figure 1:** Aqemu's dialogs are very extensive – the software covers a wide range of functions.

To permanently install a launched Live system on the virtual disk, trigger your distribution's installation process and choose the virtual image in the installer as the mass storage medium. You'll find the intended mass storage in the *HDD* tab of the Aqemu window, usually listed as *HDA (Primary Master)*, including the maximum size of the virtual disk. If necessary, you can adjust the disk size by clicking the *Format* button when the VM is off.

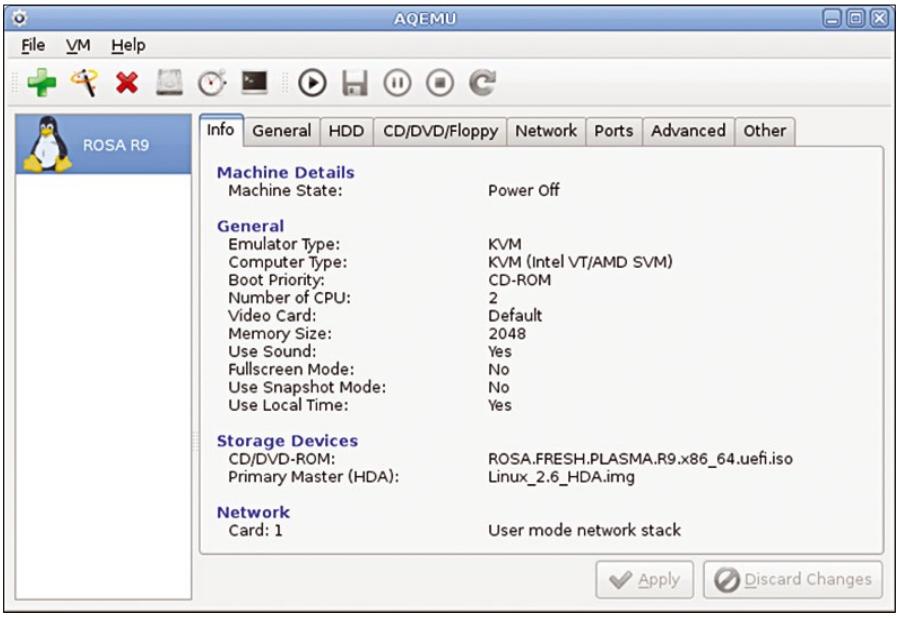



Figure 2: The Info tab provides a system overview of the VM.

After completing the setup, check the most important details in the *Info* tab (Figure 2). You can create several virtual systems, which Aqemu then lists in the program window to the left.

## Manual Creation

You can also create new VMs manually, instead of using the wizard. Left click on the button with the green plus sign. The software asks for a name for the new system; you can then enter all other settings in the various tabs.

ARM emulation is initially preset in the *General* tab in the *Computer Type* checkbox. Change the setting to *IBM PC 64Bit* so that you can then modify the number of core processors in the *Number of CPU* field.

When setting up multiple VMs in the *Primary Master (HDA)* segment, you will want to create a new image for a hard disk for each machine in the *HDD* tab (Figure 3). Otherwise, Aqemu assumes an image you created previously for the new VM. In the *CD/DVD/Floppy* tab, you can also modify the image that is used when the virtual system is first launched.

After you change all critical settings, close the dialog by clicking *Apply*. You can then launch the desired VM by clicking *Start*. Aqemu automatically adjusts the screen resolution. Additionally, it automatically starts the mouse cursor in most cases, so that you can easily switch between host and guest without the need to press any keyboard shortcuts.

To end a session, bring the Aqemu window to the foreground and click *Stop*. In a superposed dialog, the software will ask whether it should stop the VM. After confirming, Aqemu closes the window.


Figure 3: Creating an HDD image.

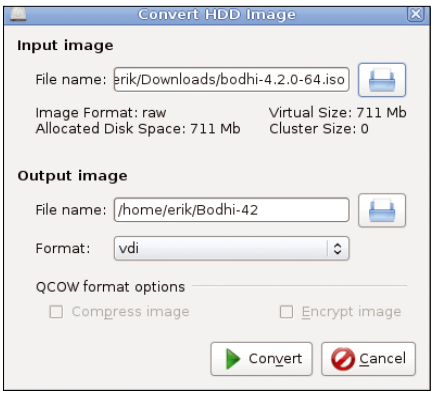



Figure 4: You can convert an image to an alternative virtualization format.

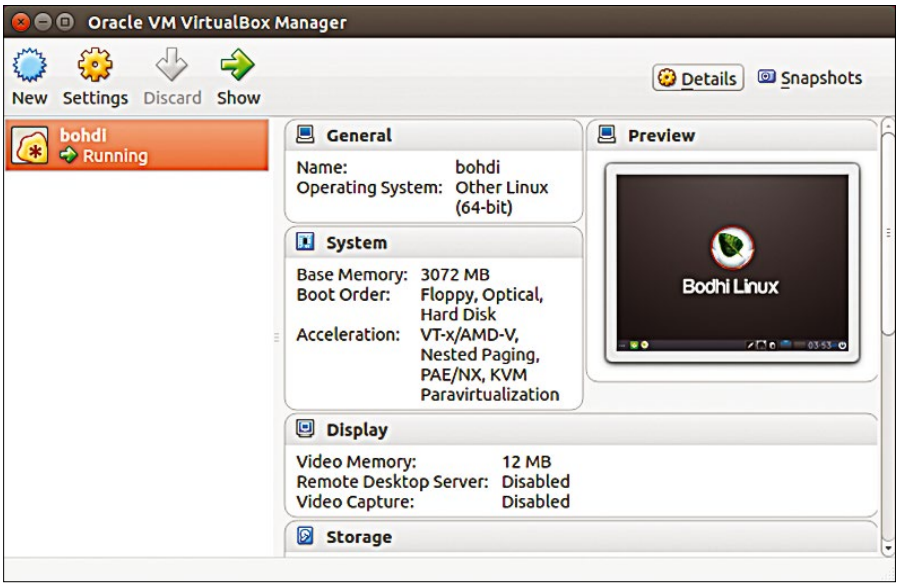



Figure 5: An image converted via Aqemu runs smoothly in VirtualBox.

## Snapshot

The KVM/Qemu virtualization environment lets you create a snapshot of a VM. A snapshot preserves a defined state of the VM, which lets you reconstruct the state in the event of a problem.

First, make sure that the VM is running when you create the snapshot,

then select the *Manage Snapshots* option in the VM menu in the main window. In the dialog box that opens, click on *+Create* and enter a name and, optionally, a description for the snapshot. After a final click on *OK*, Aqemu creates the snapshot.

In the same window, you will see existing snapshots in the *Snapshot List* box and, if desired, you can activate a snapshot with *Start*.

## Converting

Aqemu supports converting existing images to other formats. Open the *Convert HDD Image* dialog in the File menu and select the source file (Figure 4). Then enter a name for the new file and use the *Format* field to define the software for which the image is intended. You can use this method to create an image for VirtualBox (Figure 5).

After you click on *Convert*, the software gets to work. Bear in mind that, regardless of the file, you need to add the appropriate extension to the filename, so that other virtual environments will accept the file.

## On Record

Like other virtual environments, Aqemu offers a log feature. Logs are found in the *VM* menu in the main window. View the log using *Show QEMU Error Log Window*, or display the parameters of the currently selected VM using *Show QEMU Arguments*.

## Conclusions

The KVM/Qemu/Aqemu trio easily outpaces VirtualBox on desktop systems. Thanks to the graphical interface, Aqemu is suited to end users who do not want to spend hours learning the command-line parameters in order to leverage the options of KVM/Qemu.

KVM/Qemu is much more flexible than VirtualBox, and it supports alternative hardware architectures such as ARM and Sparc in addition to Intel. KVM/Qemu also offers significantly improved performance on guest systems by reducing the overhead. ■■■

## INFO

[1] VirtualBox: *https://www.virtualbox.org*
[2] KVM: *https://www.linux-kvm.org/page/Main_Page*
[3] Qemu: *http://www.qemu.org*
[4] Aqemu repository on GitHub: *https://github.com/tobimensch/aqemu*
[5] Aqemu download: *https://sourceforge.net/projects/aqemu/*

## Set up your own lab environment with KVM, Qemu, and Libvirt

# Playground

**If you don't have room on your desk for a whole laboratory of servers, simply hitch up a virtual playground on your own workstation.** *By Valentin Höbel*

It is better to test any network configuration you will one day have to depend on. If you're lucky, the application you are testing runs as a standalone tool. In the real world, however, you might not be so lucky. For instance, you might need to test a website before you make it live on the Internet, or perhaps you want to experiment with a network backup system before implementing it on your local network.

In the old days, programmers, testers, and documentation specialists often sat with three or four different computers on their desk as they tested various networking scenarios. In today's world, you can model a whole network on one computer using virtualization. Before you gum up your whole network adding a new music server or remote monitoring system, test the configuration on your laptop and work out the kinks.

This article describes how to set up a test network on a Linux portable computer. In this case, the configuration consists of three virtual systems that collect performance metrics, save the information to a database, and provide an interface for visualizing the data. You could easily adapt the techniques described in this article for other applications: file server, media center, web server, or any other networked configuration you need to test.

This solution depends on the KVM and Qemu virtualization tools described in another article in this issue, along with the libvirt virtualization API [1], which is present or available in the repositories of most popular Linux systems. The `virsh` command-line utility and the virt-manager graphical management tool serve as convenient interfaces to the libvirt virtualization environment. The configuration described in this article all took place on a single Lenovo ThinkPad computer (see Table 1). For a scenario like the one described in this article, I recommend a minimum of four usable CPU cores, 6GB RAM, and 200GB of free disk space.

**TABLE 1:** Reference System

| Component | Equipment/version |
|---|---|
| Model | Lenovo ThinkPad 460p |
| CPU | Intel Core i7-6700HQ (2.60GHz) |
| RAM | 32GB DDR4 (2133MHz) |
| Hard disk | Samsung SATA SSD (512GB) |
| Operating system | Arch Linux (64 Bit) |
| Kernel | 4.10.13 |
| Qemu | 2.9.0 |
| Libvirt | 3.2.0 |

## Careful Planning

Plan your virtual network carefully. You should only set up new systems when you really need them. Ideally, you should also delete

**The Author**

Valentin Höbel works as a Senior IT Consultant for the IT service provider open*i GmbH in Stuttgart, Germany. In his spare time, when he is not standing at the foosball table, he is looking at current open source technologies or tweeting under the @xenuser account.

these virtual machines (VMs) later through a targeted clean up operation or at least move them onto an external hard disk.

Every virtual system on the playground should serve only one purpose so that tests run isolated from one another. In some cases, virtual systems might need to communicate with each other (such as high-availability applications). Group related applications and systems in their own lab networks and isolate them from other systems.

The virtual lab in this article consists of three VMs (Figure 1). The *collectd01* VM collects metrics from the local system and sends them to the *influxdb01* VM. The Grafana visualization tool runs in the third virtual system *grafana01*, which visualizes the metrics. See Table 2 for some details on the virtual systems. The VMs all use Ubuntu 16.04 as the operating system, and the environment does not use a firewall. Unless noted, all commands within the VMs are issued as root; the commands on the host system assume a normal user with sudo rights.

**TABLE 2: Lab Systems**

| Host name | OS | CPUs | RAM | Disk | IP Address | MAC Address |
|---|---|---|---|---|---|---|
| *collectd01* | Ubuntu 16.04 | 1 | 1GB | 20GB | 172.100.100.11 | 52:54:00:b2:66:b6 |
| *influxdb01* | Ubuntu 16.04 | 1 | 2GB | 30GB | 172.100.100.12 | 52:54:00:ec:58:8e |
| *grafana01* | Ubuntu 16.04 | 1 | 2GB | 20GB | 172.100.100.13 | 52:54:00:f9:9e:9a |

## Configuration Work

If you use a typical tool such as `virsh` or virt-manager for creating the VMs, the configuration data will end up under `/etc/libvirt/`, with the VMs under `/var/lib/livirt/images/`. However, this approach creates problems later if you wish to transport the virtual lab to another host system. It is better to create a separate folder on an external or internal hard drive with enough space for each new playground.

In the example, I will format an external hard disk with the ext4 filesystem and mount it on my notebook system under `/srv/demo/`. Now create the `virtual-lab/` folder and some subdirectories in this directory; Figure 2 shows the directory structure.

The subdirectories shown in Figure 2 contain both the storage location of the VM images and adequate space for configuration files. You will need to create a storage pool and a network manually. Starting from the `/srv/demo/virtual-lab/` directory, create and open the `config/conf/networks/virtual-lab.xml` file. Fill it with the contents of Listing 1.

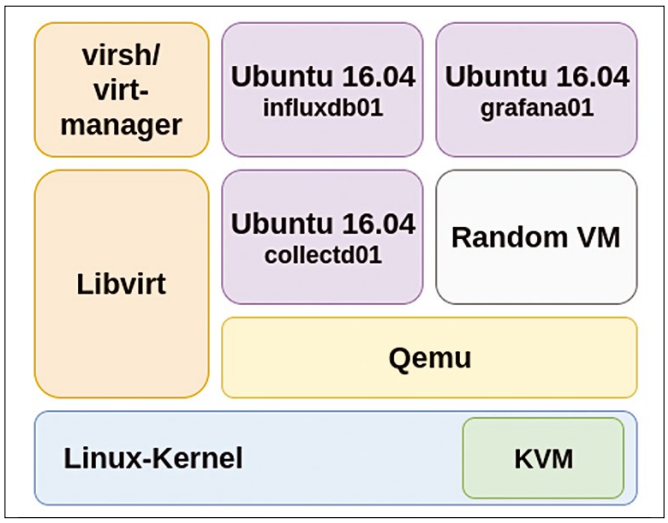



**Figure 1:** Schematic layout of the virtual laboratory. Random VM represents any other VM that is running in the laboratory.

**LISTING 1: Configuring Networking**

```
<network>
  <name>virtual-lab</name>
  <uuid>13f7adc7-475c-4e17-a0cd-4a80b94a70d7</uuid>
  <bridge name="virtual-lab" />
  <mac address='02:f1:21:33:07:9c'/>
  <forward mode='route'/>
  <domain name='virtual-lab.test' localOnly="yes" />
  <dns>
    <forwarder addr="8.8.8.8"/>
    <host ip='172.100.100.11'>
      <hostname>collectd01</hostname>
    </host>
    <host ip='172.100.100.21'>
      <hostname>influxdb01</hostname>
    </host>
    <host ip='172.100.100.31'>
      <hostname>grafana01</hostname>
    </host>
  </dns>
  <ip address="172.100.100.1" netmask="255.255.255.0">
    <dhcp>
      <range start="172.100.100.100" end="172.100.100.254" />
    </dhcp>
  </ip>
</network>
```

The next step is to create and open the `config/conf/storage/virtual-lab.xml` configuration file and transfer the contents from Listing 2. With the commands shown in Listing 3, activate the new virtual network and storage pool.

Then open the administration tool virt-manager and create three new VMs. To gain access to the Internet during the installation, issue an appropriate `iptables` command (Listing 4, first

```
[vhoebel@vhoebel-thinkpad virtual-lab]$ pwd
/srv/demo/virtual-lab
[vhoebel@vhoebel-thinkpad virtual-lab]$ tree
.
├── config
│   └── conf
│       ├── machines
│       ├── networks
│       │   └── virtual-lab.xml
│       └── storage
│           └── virtual-lab.xml
└── storage
    └── vm

7 directories, 2 files
```

**Figure 2:** The directory tree for the virtual lab.

### LISTING 2: Configuring Storage

```
<pool type='dir'>
  <name>virtual-lab</name>
  <uuid>1c2b3c2d-0140-1e49-a571-840f4c88210d</uuid>
  <capacity unit='bytes'>0</capacity>
  <allocation unit='bytes'>0</allocation>
  <available unit='bytes'>0</available>
  <source>
  </source>
  <target>
    <path>/srv/demo/virtual-lab/storage/vm</path>
    <permissions>
      <mode>0755</mode>
      <owner>-1</owner>
      <group>-1</group>
    </permissions>
  </target>
</pool>
```

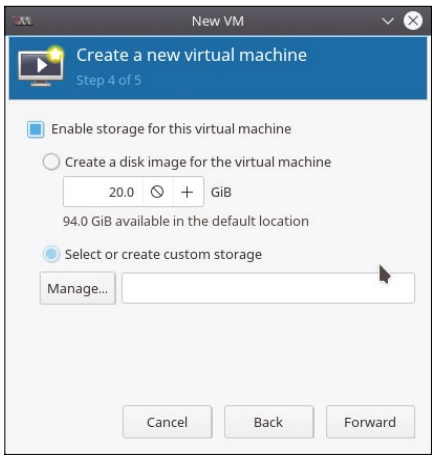



Figure 3: Caution: You need a user-defined location.

line). Replace the `wlp3s0` interface with an interface that has access to the Internet on your computer.

This configuration works if your system allows the forwarding of IPv4 packets. To allow IP forwarding, create the `/etc/sysctl.d/demo` file with administrative rights, store the appropriate entry, and then activate packet forwarding via `sysctl` (Listing 4, second and third lines).

Take care when installing the test systems to select *Select or create custom storage* under Step 4 and click on *Manage* (Figure 3). Then select the *virtual-lab* storage pool and click on the plus icon next to *Volumes* (Figure 4). A new window opens, where you can specify the name for the image file and the capacity (Figure 5).

For the first VM, select *collectd01.qcow2* and *20GB*. Press *Finish*, select the new disk, and press *Choose volume*. Return to the *Create a new virtual machine* dialog. Now enter the name for the VM (such as *collectd01*), unfold *Network selection*, and make sure that *Virtual network 'virtual lab': NAT* is active.

Boot the VM and the operating system installation by pressing *Finish*. Set up all three virtual systems in this way, and then shut down all the systems. I was asked for a user account during the installation and entered *demo* as the username and password.

To guarantee that each of the three new lab VMs will receive the same IP address with each launch, store the MAC address for each virtual system in the network configuration. You can find the MAC addresses by double clicking a VM in

### LISTING 3: Activating Network and Storage

```
$ sudo virsh net-create /srv/demo/virtual-lab/config/conf/networks/virtual-lab.xml
$ sudo virsh pool-create /srv/demo/virtual-lab/config/conf/storage/virtual-lab.xml
```

### LISTING 4: Accessing the Network

```
$ sudo iptables -t nat -A POSTROUTING -o wlp3s0 -j MASQUERADE
$ sudo sh -c "echo 'net.ipv4.ip_forward = 1' >> /etc/sysctl.d/demo"
$ sudo sysctl -p /etc/sysctl.d/demo
```

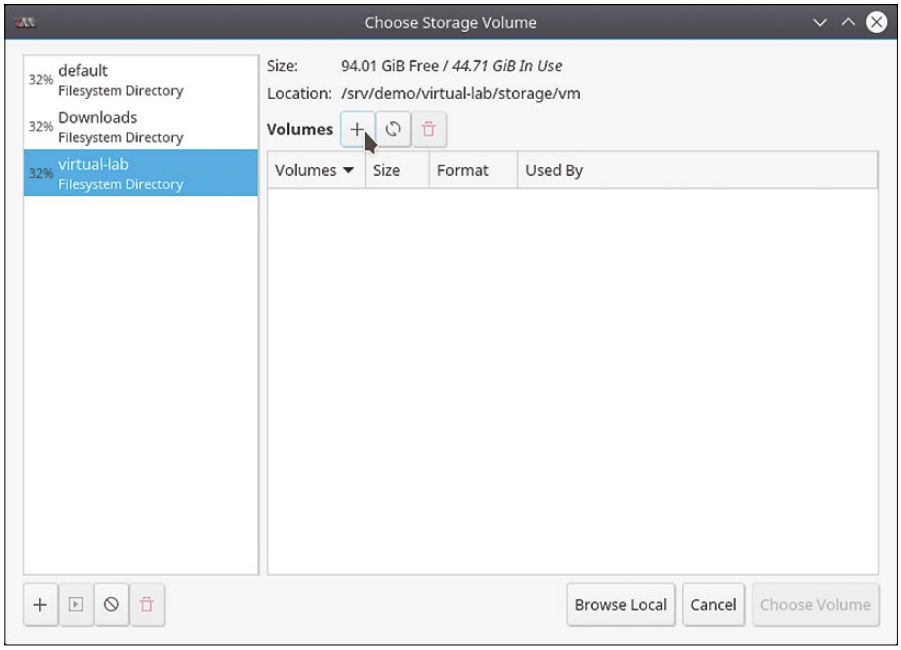



Figure 4: Create a new disk in the storage pool.

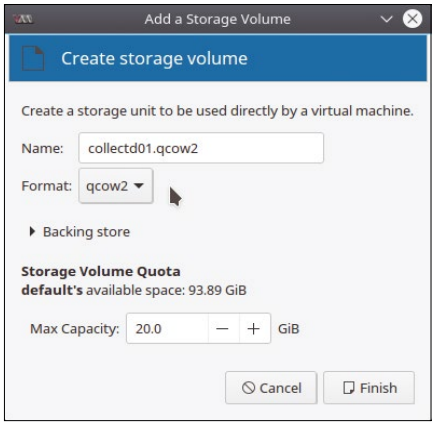



Figure 5: Specify the name and the capacity for the VM image.

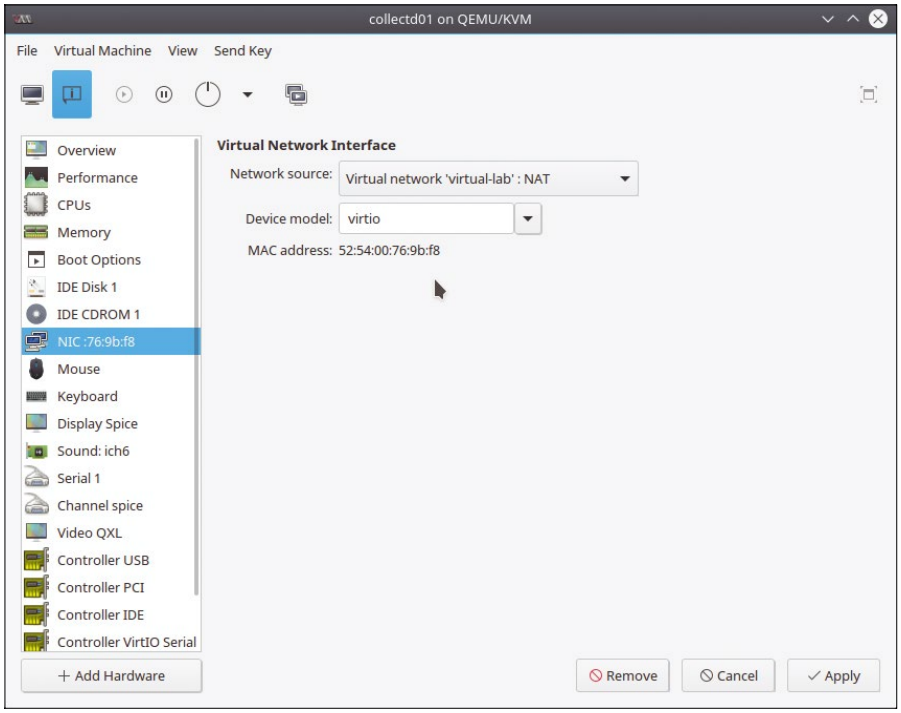



Figure 6: **Use virt-manager to find out the MAC addresses of the virtual systems.**

## LISTING 5: Stopping and Relaunching

```
$ sudo virsh net-destroy virtual-lab
$ sudo virsh net-create /srv/demo/virtual-lab/config/conf/networks/virtual-lab.xml
```

## LISTING 6: Adding DHCP

```
<network>
  <name>virtual-lab</name>
  <uuid>13f7adc7-475c-4e17-a0cd-4a80b94a70d7</uuid>
  <bridge name="virtual-lab" />
  <mac address='02:f1:21:33:07:9c'/>
  <forward mode='route'/>
  <domain name='virtual-lab.test' localOnly="yes" />
    <dns>
    <forwarder addr="8.8.8.8"/>
    <host ip='172.100.100.11'>
      <hostname>collectd01</hostname>
    </host>
    <host ip='172.100.100.21'>
      <hostname>influxdb01</hostname>
    </host>
    <host ip='172.100.100.31'>
      <hostname>grafana01</hostname>
    </host>
  </dns>
  <ip address="172.100.100.1" netmask="255.255.255.0">
    <dhcp>
      <range start="172.100.100.100" end="172.100.100.254" />
        <host mac='52:54:00:b2:66:b6' name='collectd01' ip='172.100.100.11'/>
        <host mac='52:54:00:ec:58:8e' name='influxdb01' ip='172.100.100.21'/>
        <host mac='52:54:00:f9:9e:9a' name='grafana01' ip='172.100.100.31'/>
    </dhcp>
  </ip>
</network>
```

### LISTING 7: Moving the Config Files

```
$ cd /srv/demo/virtual-lab/config/conf/machines
$ sudo mv /etc/libvirt/qemu/collectd01.xml .
$ sudo mv /etc/libvirt/qemu/influxdb01.xml .
$ sudo mv /etc/libvirt/qemu/grafana01.xml .
```

### LISTING 8: Launching the VMs

```
sudo virsh create /srv/demo/virtual-lab/config/conf/machines/collectd01.xml
sudo virsh create /srv/demo/virtual-lab/config/conf/machines/influxdb01.xml
sudo virsh create /srv/demo/virtual-lab/config/conf/machines/grafana01.xml
```

### LISTING 9: Installing InfluxDB

```
# curl -sL https://repos.influxdata.com/influxdb.key | apt-key add -
# source /etc/lsb-release
# echo "deb https://repos.influxdata.com/${DISTRIB_ID,,}
       ${DISTRIB_CODENAME} stable" | tee /etc/apt/sources.list.d/influxdb.list
# apt-get update
# apt-get install influxdb
# cd /etc/influxdb/
# wget http://hoebel.net/downloads/articles/influxdb.conf
# mkdir /usr/share/collectd
# cd /usr/share/collectd/
# wget https://raw.githubusercontent.com/collectd/collectd/master/src/types.db
# systemctl restart influxdb
```

### LISTING 10: Installing Collectd

```
# apt-get update
# apt-get install collectd collectd-utils
# cd /etc/collectd/
# wget http://hoebel.net/downloads/articles/collectd.conf
# systemctl restart collectd
```

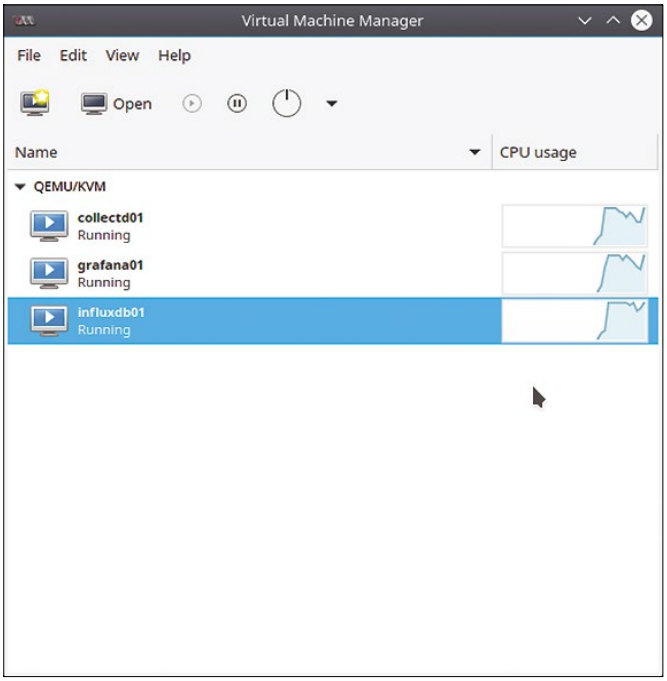



Figure 7: All three test systems appear together with loading information in virt-manager.

virt-manager and jumping to the *NIC* entry using the information icon (Figure 6).

Now make sure all new VMs are shut down and stop the virtual lab network (Listing 5, first line). Add the DHCP entries to the network configuration for each VM. The configuration should roughly match Listing 6. (Replace the MAC addresses with the addresses for your systems.) Then relaunch the network with the command in the second line of Listing 5.

At the end of the configuration work on the virtual lab, you need to move the VM configuration files. Navigate to the command line in the directory for the VM configuration (Listing 7, first line) and move the configuration files (Lines 2 to 4).

## Starting up Virtual Lab

At this point, you have set up the new virtual playground on your system and put all the necessary data in one directory, so you can easily transport it onto an external hard drive and integrate it with other systems.

Launch the three test systems using the `virsh create` command (Listing 8).

Virt-manager displays the three test systems (Figure 7). Now manage the trio using the console (or just via virt-manager)

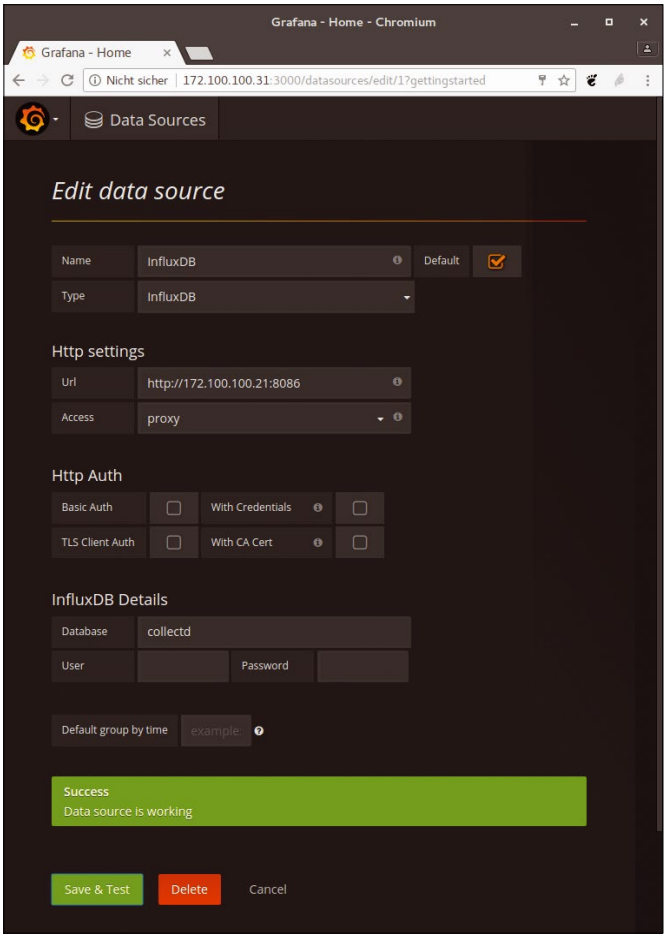



Figure 8: Configuration for Grafana data source.

**LISTING 11:** Installing Grafana

```
# echo "deb https://packagecloud.io/grafana/stable/debian jessie main" | tee /etc/apt/sources.list.d/grafana.list
# curl https://packagecloud.io/gpg.key | sudo apt-key add -
# apt-get update
# apt-get install grafana
# systemctl start grafana-server
```

or install the `openssh-server` package and connect to an SSH client (example: `ssh demo@172.100.100.11`). Perform a few connection tests to check whether the network connections are working. A short test with `ping linux-magazine.com` and `ping grafana01`, `ping influxdb01`, as well as `ping collectd01` is sufficient for each VM.

## Setting up the Systems

When you get the VMs up and running, the next step is to install the applications that will run on three virtual systems: InfluxDB [2], collectd, and Grafana.

The InfluxDB packages are not part of the official Ubuntu repositories; however, a third-party repository contains everything we need. Execute the commands in Listing 9 as root on the *influx01* VM.

Now check if the InfluxDB application is running:

```
ps aux |grep influx
```

The next step is to log into the *collectd01* system and set up the collectd daemon [3]. Collectd collects metrics of the local system and sends them to InfluxDB for permanent storage. Listing 10 shows the steps for installing the manic collectd collecting service on the `collectd01` test system. Check to see if the daemon is actually running:

```
ps aux | grep collectd
```

If it isn't running, stop all collectd processes and relaunch the service.

After a few minutes of run time, the time-series database fills InfluxDB with metrics from the *collectd01* test system. Check the status by running the `influx` command on the InfluxDB VM and listing all databases with `show databases`.

Use `use collectd` and `show measurements` to show which value categories ended up in the database. The `quit` command takes you back to the command line.

The virtual laboratory now collects metrics from one of the test systems and stores them in a database on another VM via the virtual network. The last step is to add visualization using Grafana [4].

### LISTING 12: Point Grafana to the Data

```
http://172.100.100.31:3000/dashboard/db/server-operating-system-metrics?refresh=
    1m&orgId=1&var-host=collectd01&var-interval=$__auto_interval
```

Open a command line on the *grafana01* test system and install Grafana (Listing 11). Line 1 sets up a software repository for Debian systems. (The system is Ubuntu, but Grafana developers have deliberately kept their package repository simple and have taken advantage of the fact that the package management works in the same way under Ubuntu and Debian.)

After you launch Grafana, the service listens on port 3000. Open the *http://172.100.100.31:3000* URL in a web browser, and you'll see a welcome to the login screen. Sign up as *admin* with the password *admin* and click on the *Add data source* link. In the *Name* field, enter the *InfluxDB* value and select the same value in the *Type* drop-down menu.

For *URL*, enter the *http://172.100.100.21:8086* character string – Grafana will later reach InfluxDB under this address. Skip the configuration fields for *Http Auth* and, under *InfluxDB Details*, enter the name *collectd* in the *Database* field. Now press the green *Add* button and then *Save & Test*. Figure 8 shows the complete configuration.

You'll find a Grafana dashboard prepared for this configuration at my website [5]. Download the configuration file and click on the Grafana icon in the top left in Grafana. The icon opens a menubar where you select *Dashboards* | *Import*. In the next dialog, install the recently downloaded JSON file using the *Upload .json File* button. Don't forget to select the correct data source *InfluxDB* before clicking *Import*.

Next, Grafana guides you to the recently imported dashboard. Enter the *collectd01* value under Host in the top left, or simply go to the link shown in Listing 12. Figure 9 shows the visualized data. The prefabricated dashboard displays all sorts of processed metrics for the *collectd01* test system.

## Conclusions

Creating a virtual test lab on your home computer requires less work than you might expect. Pay attention, take your time, and remain patient; you will very soon discover that the techniques described in this article will let you quickly configure and model complex test scenarios. ■■■

## INFO

[1] Libvirt: *https://libvirt.org/*
[2] Official InfluxDB website: *https://www.influxdata.com*
[3] Collectd: *https://collectd.org*
[4] Grafana: *https://grafana.com*
[5] Grafana dashboard for the virtual lab: *http://hoebel.net/downloads/articles/LinuxUser-Grafana-Dashboard.json*

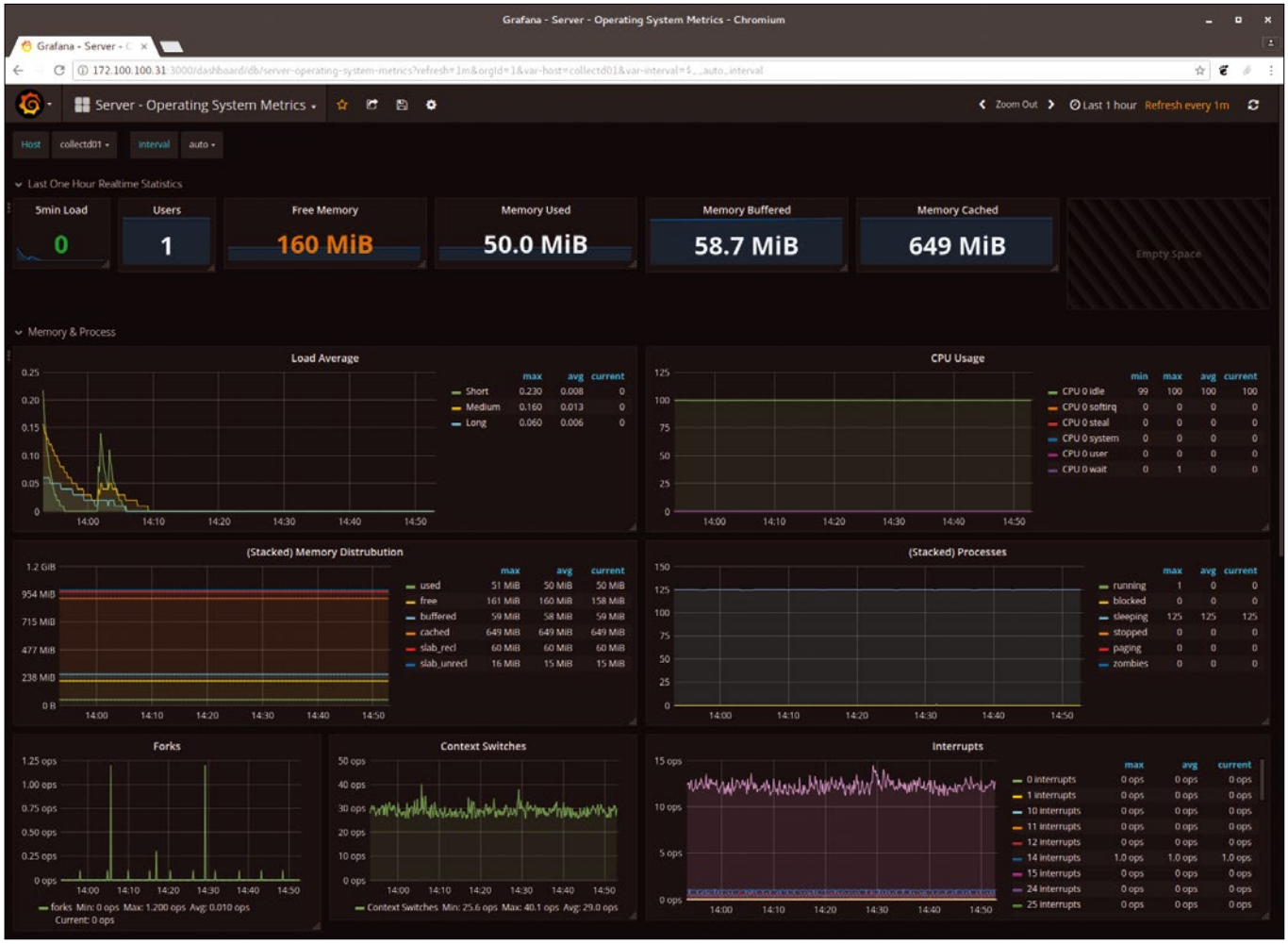



Figure 9: Grafana shows metrics to `collectd01`.

# Subscribe now!


**Don't miss a single issue of the magazine that delivers the in-depth technical solutions you'll use everyday!**

**shop.linuxnewmedia.com/subs**

## Exploring the extra tiny KolibriOS

# Little Friend

**KolibriOS, written in assembler, is especially suited to very old hardware – it even fits on a floppy disk.** *By Erik Bärwaldt*

Several small distros inhabit the Linux landscape. The best of these pared-down systems provide many of the same amenities associated with their more bloated counterparts, including GUI interfaces, games, and full-featured office suites. Some users prefer a minimal system just because they like to travel light and avoid the complications associated with unnecessary features. But beyond the personal aesthetics, though, tiny distros play a special role for the open source community: keeping old hardware alive.

Many users have old computers sitting around that are still perfectly functional but don't have the resources necessary to run contemporary mainstream systems. An old Windows 98 box, for instance, isn't nearly big enough or fast enough to run Windows 10, and mainstream Linux alternatives like RHEL and Ubuntu can't really offer a solution because they are just as resource hungry as Windows.

A small Linux, however, can easily fit on an older system. Lightweight distros such as Lubuntu, Puppy, and Damn Small Linux are all supported by loyal communities that see big value in a small footprint. But what if you want to get *really* small – and I mean *really really* small?

KolibriOS requires only a few Megabytes of disk space and it runs on 8MB of RAM, but this tiny size still provides a remarkably conventional user experience, including a GUI interface with word processor, image viewer, graphics editor, and over 30 games. Because KolibriOS runs in memory and is written entirely in assembly language, it is also very fast. According to the project website, KolibriOS "…boots in less than 10 seconds from power on to a working GUI on a $100 PC."

KolibriOS has been in development since 2004 and is a fork of the also-slim MenuetOS.

## Technical Matters

The GPLv2-licensed KolibriOS is a pure 32-bit operating system. Support is limited to single-core CPUs, but the system also runs on multicore processors.

The system requirements specified by the developers seem downright ridiculous for today's conditions: A single-core Pentium CPU with at least 100MHz frequency, as well as 8MB of storage space, are required. Additionally, the developers recommend a VESA-compliant graphics card from about 20 years ago that supports current screen resolutions up to 1024x768 pixels at 32-bit color depth.

You download KolibriOS as a tiny 7z archive in multiple versions on the project page [1]. KolibriOS requires 60MB of free storage space for installation on a mass storage device. The project provides an ISO image for burning onto a CD, as well as the files for use on a hard disk, in English, Spanish, Russian, and Italian; no multilingual packages are available.

The package archives, each with a volume of just 25MB, can be unpacked using archiving software such as PeaZip or the corresponding desktop applications. Then you can burn that large 60MB ISO image onto a CD, from which you can launch the system.

If you would like to start KolibriOS in a virtual machine (VM) or via the network using PXE, you will find extensive instructions in the wiki [2].

## Drivers

Despite the small size, KolibriOS offers a range of drivers, especially for old hard-

ware: Graphics cards from ATI and Intel can be used with the *radeon* and *i915* drivers, which largely correspond to their Linux counterparts. The VESA driver that is implemented in the kernel, which also supports the 16-, 24-, and 32-bit "high" color depths, uses other graphics cards. It is apparent that KolibriOS is able to correctly display the screen content on modern widescreen displays in the form factor 16:10 and 16:9, even with the hardware-independent VESA graphics card drivers.

With the audio support, KolibriOS relies on modern Intel drivers (AC97). However, it also provides several modules for sound cards that set the standards some twenty years ago: In addition to Sound Blaster drivers, you will also find those for SIS, Ensoniq, and VIA audio cards, which are hardly known today.

In the 1990s, many computer systems were already networked, and Internet standards were still in their infancy, but nevertheless already existed. KolibriOS, therefore, even supports the oldest 3Com590 or Realtek LAN cards. In addition, the system also offers support for Intel's Gigabit Ethernet hardware, so you can also use it on a modern LAN.

Floppy disks, optical drives (CD/DVD), and IDE hard drives can be used as mass storage devices. KolibriOS supports USB flash drives in accordance with the 1.x and 2.0 specifications, as well as USB hubs. Thus, the system can even be used on old notebooks, which provided USB interfaces for the first time at the end of the 1990s. The built-in drivers allow the connection of USB devices such as a mouse, keyboard, and external hard drives. Even a driver for controlling USB printers is available.

**LISTING 1: Copying to the USB flash drive**

```
# cp /usr/lib/syslinux/memdisk /dev/<KolibriOS-Partition>
# syslinux -s /dev/<KolibriOS-Partition>
```

## Start

Instead of GRUB, KolibriOS uses good old Syslinux as a boot manager. KolibriOS also works with GRUB, if you wish to add it to a configuration with several operating systems on a mass storage device.

After the launch via an ncurses boot screen, the system opens the graphical desktop within a few seconds. Even on a roughly nine-year-old machine with a Penryn dual-core processor, the launch in the test didn't even take five seconds.

You can also try out KolibriOS with the ISO image in a VM. Configure the image file as a virtual optical drive in the Mass Storage menu in VirtualBox. The default 64MB is sufficient as the main storage. In the VM, the entire system launches within much less than 10 seconds and is ready to run from the memory with all the applications. Since KolibriOS supports the virtual network components of VirtualBox, you also have direct access to the Internet from the VM (Figure 1).

## USB Flash Drive

If necessary, you can install KolibriOS on a flash drive – this again improves the operating system's launch speed. A flash configuration requires a few manual steps. First, create a partition on the stick using a tool like GParted and format it with the FAT16 or FAT32 filesystem. Don't forget to launch the boot flag, which you can do in GParted via the *Partition* | *Edit tags* menu.

Then make sure the *syslinux* and *mtools* packages are on your Linux system. If necessary, install it from the used distribution's software repositories. Then copy the `memdisk` file with root privileges using the command from the first line of Listing 1 into the USB flash drive root directory.

Subsequently, download the `latest-distr.7z` archive from the KolibriOS project, in which the universal image for the operating system is located. Unpack this archive, and then copy the `kolibri.img` file into the USB flash drive's root directory.

Afterwards, remove the USB drive from the system and enter the command from the second line of Listing 1 when prompted. The command basically sets up the system, including the bootloader. For Syslinux to work correctly, create a file called `syslinux.cfg` in the flash drive's main directory. You can expand this file using a text editor to include the following:

```
default memdisk initrd=kolibri.img
```

After that, the system is ready to launch from this USB drive.

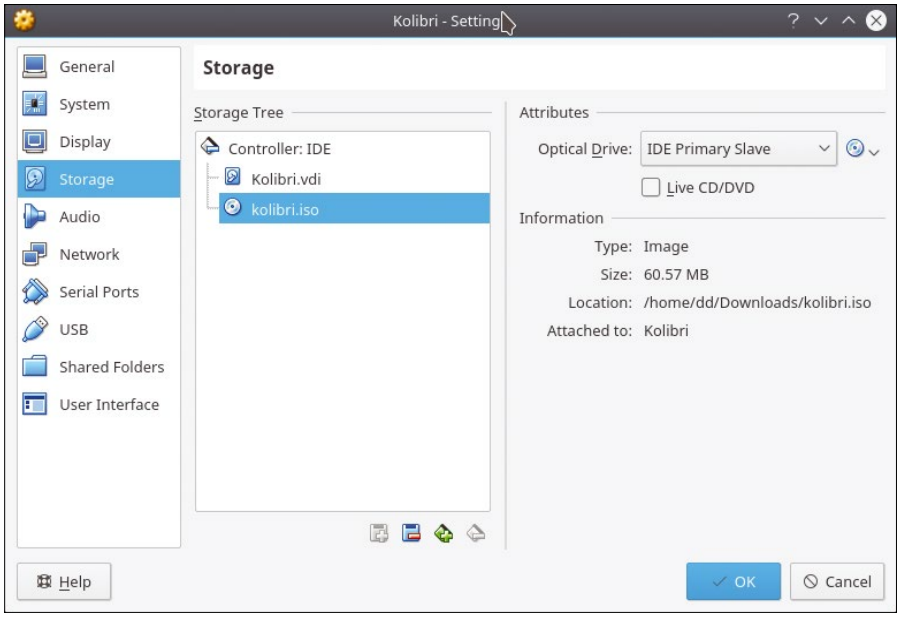



**Figure 1:** KolibriOS also cooperates well with low requirements with a VM from VirtualBox.

## In Operation

On the lower edge, the KolibriOS interface has a horizontal panel bar with a menu button at the bottom left. On the left edge – only visible at a second glance – there is a foldable switch bar that opens when clicked with the mouse.

To adjust the desktop, right-click in the working interface. You can then, for example, create an additional launcher on the desktop or change the appearance of the work environment via the corresponding entries in the context menu.

The system comes with a whole array of very slim applications. A total of around 30 icons can be found on the

desktop. At the top left, you will find the launcher for office and administration software. The file managers Eolite and KFAR are similar to the Midnight Commander or PCManFM, the default file manager for LXDE desktops. Many small applications are available for daily use, including the text reader, the calculator, and the text-based web browser, WebView, as well as the Tinypad notepad and the Animage drawing program.

At the top in the right corner of the screen are additional applications, mostly for programming work. These include a terminal, comparison software for text files, a debugger, an archiver, an assembler, and a hex editor. This group is rounded off with a debug and message board, as well as a display program for software documentation.

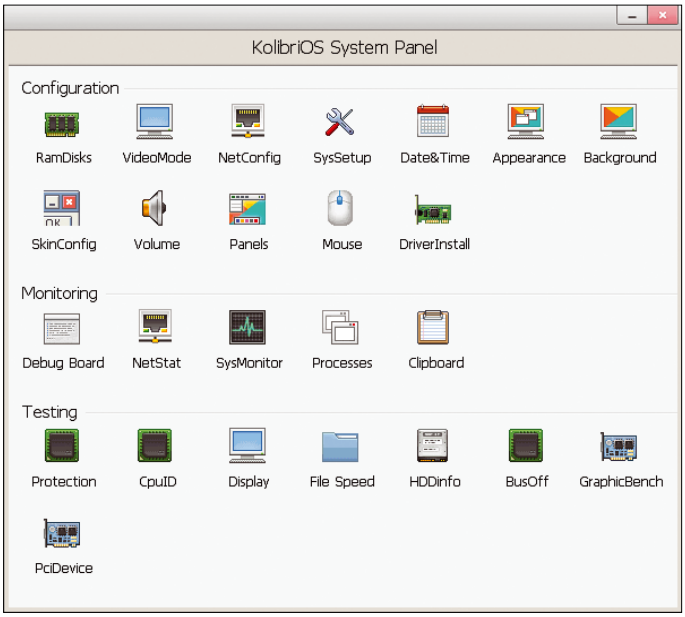



**Figure 2:** Syspanel brings together all the major configuration tools in an orderly manner.

The graphical tool Syspanel is reminiscent of the configuration tools of great desktop environments in Linux and comprises the most important tools for configuring the operating system (Figure 2).

Numerous games are clustered at the bottom of the screen in the right and left corners. These games include small classics such as Sudoku, Gomoku, Tetris, Snake, Mine, and Checkers, but also several puzzle and block games. Some of the games are only available with Russian localization and are therefore of limited use.

The foldable panel bar in the middle on the left edge of the screen lets you access more applications, such as an IRC client, a text editor, both an MP3 and Midi Player, and a volume control. A graphical benchmark program, with the unusual name KGB, and a CPUID routine complete the inventory.

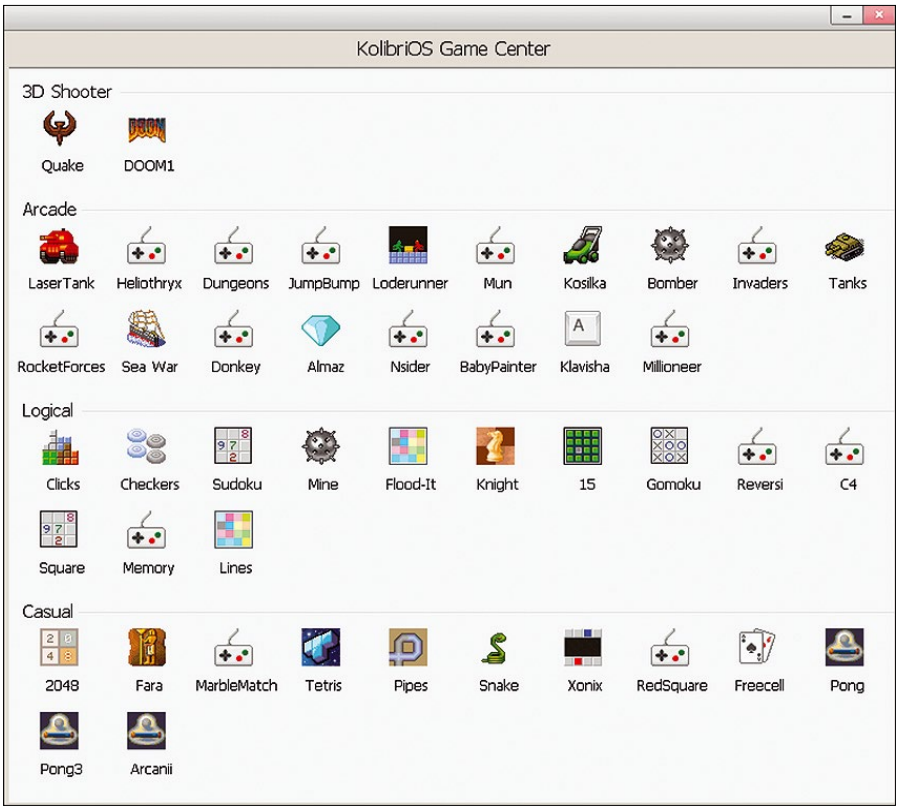



**Figure 3:** In the Game Center, you will find a number of smaller games and emulators for the formerly widespread ZX Spectrum by Sinclair.

## Menu

You will find additional software categorized into subgroups in the operating system's menu, which you can reach using the corresponding button at the bottom left in the panel bar. The focus is on games, emulators, and developer tools. In particular, the Emulators menu reveals some interesting applications: in addition to the famous DOSBox for launching old DOS software, you will also find emulators for Super Nintendo and Game Boy gaming consoles.

Of particular interest is the ZX Spectrum emulator, which emulates a 1980s home computer by the British manufacturer Sinclair. Many applications originally developed for the Spectrum are still in use today. You fill find additional games in the Game Center (Figure 3). Playback software is available in the Multimedia menu with Fplay + for videos and movies.

## Conclusions

KolibriOS is particularly impressive thanks to its fantastic speed that even outshines minimalistic Linux derivatives such as SliTaz. The system pleases with its modest hardware requirements.

With its many preinstalled emulators, KolibriOS is a good choice for gamers who have a soft spot for older game consoles or DOS games. However, as a conventional desktop system, KolibriOS is only suitable for a limited number of applications, such as a word processor or web browser. ■■■

## INFO

[1] KolibriOS: *http://kolibrios.org/en/download*
[2] Installing Wiki: *http://wiki.kolibrios.org/wiki/HowTo*

Tools for reconstructing deleted data

# Rescue Mission

**One false click can quickly delete important data, or even an entire partition. If a backup tool is missing, only a rescue specialist can help.**

*By Erik Bärwaldt*

Accidentally deleting data without a backup is a nightmare scenario for many users. Even if the application prompts for confirmation before deleting, the user only has to click too fast, and the data is gone. The power of the command line is another threat. An incorrect command parameter could send an entire directory tree into the black hole of oblivion.

Despite the danger of losing data, surveys show that users often create inadequate or no backups [1]. Luckily, the Linux environment includes several tools for reconstructing lost data.

## Organizational Matters

Mass storage – whether it be hard drives, solid state drives, or optical discs – always manages and stores data in an organizational structure. Some filesystems also inform the operating system about the size, location, and directory attributes of the file resource using metadata.

The filesystem maintains a table of contents, so the operating system can track down lost files on the storage medium. The system tracks the initial and subsequent clusters, as well as the number of clusters occupied by a file. The internal directory either creates references to the corresponding data or generates a table.

If you delete a file, the data does not automatically disappear. Instead, the filesystem simply drops the corresponding entry from its table of contents. In some cases, it might only delete the first letter of the filename in the directory, while the file lives on anonymously. The original file really only disappears when the system overwrites the occupied sectors of the storage medium with other data.

If you accidentally delete a file, the chances are quite good that the system administrator can still reconstruct the file or at least fragments of it with the appropriate tool, especially on large storage devices. (See the "Preparations" box for information on getting your storage medium ready for data recovery.)

## PhotoRec

PhotoRec [3], which is published under the GNU GPLv2 and is already part of most rescue distributions, is a classic tool for deleted database reconstruction. Also, many Mandriva-based distributions, openSUSE, Fedora, and CentOS offer PhotoRec as a separate package with an optional GUI.

In Linux distributions, PhotoRec is often part of the TestDisk package that reconstructs whole mass storage partitions. It can be installed from the repositories on Debian and Ubuntu. The TestDisk project website [4], in addition to the source code, offers current versions of TestDisk that work with kernel 2.6.18.

PhotoRec – contrary to what the name suggests – does not just recover image files, but also archives, various document formats, and multimedia data. In total, PhotoRec supports hundreds of formats from more than 100 format families [5].

At the same time, PhotoRec operates independently of the underlying filesystem. It doesn't matter whether the data to be reconstructed is on an ext2, ext3, or ext4 partition; a FAT, VFAT, or NTFS drive; or – in the Apple world – on an HFS filesystem.

PhotoRec can also cope with a wide variety of media: In addition to traditional hard drives and USB flash memory devices, source media also include SSDs, SD and compact flash cards, and optical discs.

After installation, the software is immediately available at the command line but requires administrative privileges. The QPhotoRec GUI [3] (Figure 1) for Qt-based work environments, such as KDE Plasma 5, also requires root privileges. After installation, it can be

## PREPARATIONS

If the storage medium no longer has the data, the administrator needs to consider some factors for successful reconstruction.

You will need to use the affected medium in read-only mode to prevent accidental overwriting of the residual data. Specifying a directory on the affected storage medium as a target path for reconstructed files is not a good idea. It is best to launch the computer system from a Live CD or a USB flash drive with a Live system on it.

Sometimes third parties want to edit the data carrier professionally and forensically – for example, in the case of official inquiries or legal conflicts that require watertight data reconstruction documentation. In this case, the administrator cannot work with the original storage medium anyway. Instead, the admin needs to make a complete copy of the medium and verify it using a checksum; this copy can then be used to attempt data recovery.

A copy of a faulty disk can be created by the fairly ancient but reliable command-line program `dcfldd` [2], which exists as an extension of the `dd` command in Linux. It is also found in many software repositories, such as Debian, Ubuntu, Fedora, Slackware, CentOS, and Mageia.

found under the *System* menu of the KDE desktop.

After running the command, first select the drive from which you want to reconstruct the data. At the command line, PhotoRec offers a selection from a table of detected drives, whereas QPhotoRec shows a selection box in the upper part of the window.

The GUI already anticipates the next step, which the user otherwise needs to type at the command line: It shows the partition table of the activated mass storage device, from which you then select a partition if necessary. It then determines the filesystem of the selected partition; the software only supports two options here. If the source drive is not an ext2, ext3, or ext4 filesystem, it will choose the *Other* option.

In the command-line version, PhotoRec asks for the destination where you would like to store the reconstructed files and then starts the search when you press *C*; it informs you of the progress in real time. The software stores the recovered files in numbered subdirectories within the target directory.

While the console displays the options in separate dialogs, the GUI user can change options in just two displays: After specifying the source drive, you specify the desired partition in a tabular list. Define the filesystem and the target path in the same dialog with a selection box and an input cell.

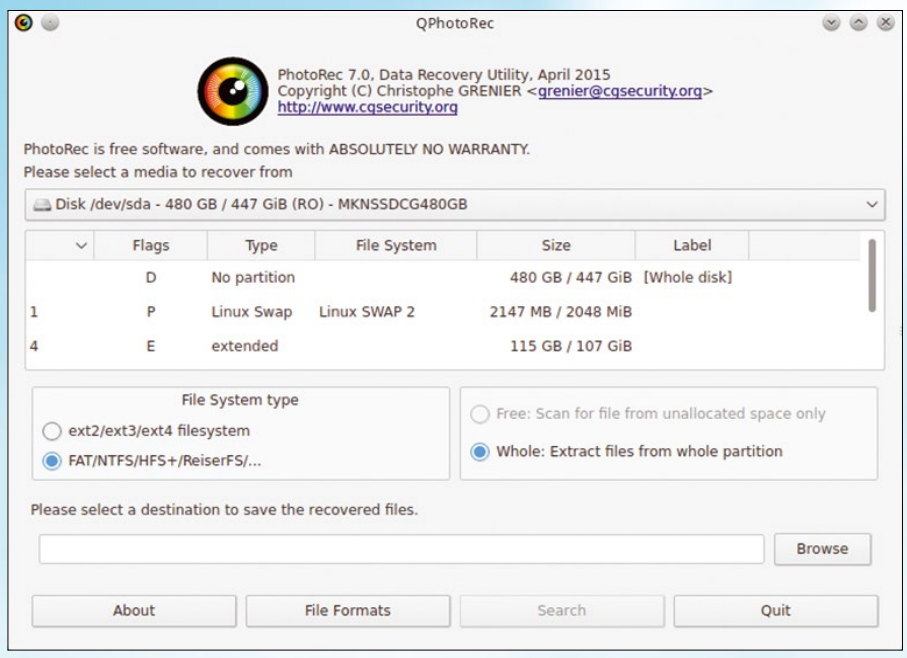



**Figure 1:** QPhotoRec makes PhotoRec easier to use with an intuitive user interface.

Over time, mass storage can accumulate hundreds of thousands of files of various formats, which PhotoRec can recover. Reconstructing all available file types not only takes a few hours, it uses a huge amount of storage space in the target directory. The numerous subfolders very quickly make this type of complete reconstruction confusing.

The software therefore provides the option to exclude arbitrary file formats from the reconstruction. In the PhotoRec command-line version, a separate dialog lists the available formats one by one; an *x* to the left selects the file type. You can deactivate unwanted formats by removing the *x*; deselected formats are then ignored in the subsequent recovery.

QPhotoRec lets you ignore file formats much more conveniently by clicking on the *File Formats* button at the bottom of the program window. You can then select your favorites from a tabular list. If you only want to rescue a few file types, you first need to deactivate all formats by clicking the *Reset* button and add the desired file types by checking the boxes (Figure 2).

After setting the options in the dialog box, you can launch the reconstruction by clicking the *Search* button at the bottom of the window. QPhotoRec then displays the progress with a bar chart and a table. When finished, a click on *Quit* terminates the program.

Out of the box, QPhotoRec also provides a logging function that stores data carrier settings in an ASCII file located in the logged-on user's home directory. In addition to information about partitions and filesystems, there are also hardware and operating system-specific settings in the file that particularly help in forensic work.

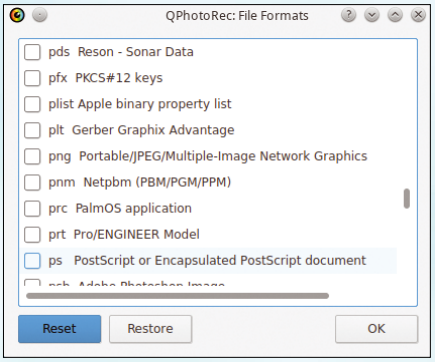



**Figure 2:** Among other things, QPhotoRec provides a simple dialog for format selection.

## TestDisk

TestDisk [4], which many distributions provide in the same package as PhotoRec, reconstructs partitions and filesystems. The command-line program is included with many Live systems for data reconstruction [6], but it is also found in the repositories of all major Linux distributions. If you run the application with root privileges, a text screen pops up suggesting that you create a logfile. Later, all your work steps can be traced.

In another dialog box, you select the corrupt partitions from a list of detected drives. If a valid partition table is missing on the medium, TestDisk lets you search for one with the *Analyse* option. This includes GUID Partition Tables (GPTs) that are used with today's very large data carriers instead of conventional master boot record (MBR) partition tables.

Depending on the storage medium's size, the data structure analysis can take a while. If TestDisk detects damaged partitions, it displays them in the terminal, including the filesystem used (Figure 3).

Once the software has identified the partition to be repaired, TestDisk will search for the partition type. For large disks with partitions larger than 2TB, select *EFI GPT* in the partition table type; in the case of smaller partitions, you can usually keep the *Intel* default. The software then displays the available tools for repair. The first two options: *Analyse* and *Advanced* are used to recover damaged partition tables and work with files and entire partitions.

You can copy individual files or create images here. By using the Boot menu, you can also repair faulty boot sectors or set up the backup for a boot sector. In this case, you control the software by entering individual letters. Dialogs describe what the respective letters do. You need to take into account that the letters are partially case sensitive and trigger different actions depending on whether they are uppercase or lowercase.

The other options: *Geometry*, *Options*, *MBR Code*, and *Delete* help to adapt faulty disk geometries, repair a storage medium's MBR, or completely empty a partition table.

TestDisk also provides the option to create a logfile immediately after startup in the user's home directory. Because of an unformatted text file, the format is deliberately independent of the application and operating system and logs all your work steps. The steps can be easily traced at any time.

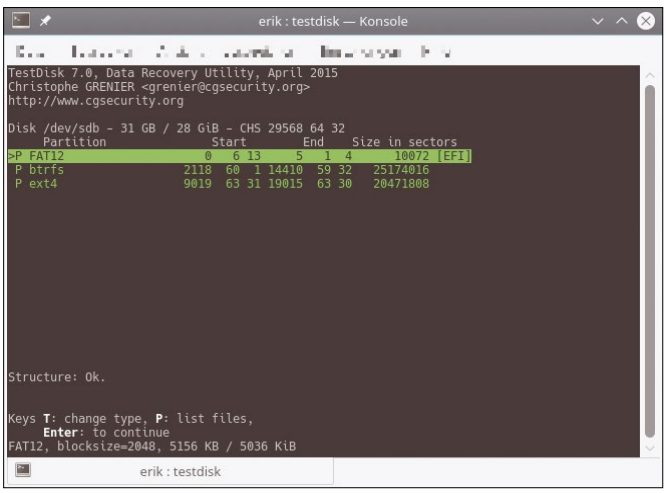



**Figure 3:** In this case, TestDisk found three partitions.

## Magic Rescue

The Magic Rescue [7] command-line tool provides another approach to recovering corrupted partition tables or data lost by accidental deletion. However, the software does not rely on filesystem allocation tables; instead, it evaluates the magic numbers that identify each file type. In almost all files, the magic numbers are inserted in the headers before the actual user data and are standardized by the file type.

Corrupt file allocation tables, which no longer allow the reconstruction of individual file fragments, do not negatively affect content recovery. Magic Rescue is included in the repositories for many distributions, such as Debian and Ubuntu, as well as Mageia and openSUSE.

Magic Rescue relies on recipes for detecting the file type. These recipes in file form are used as a template for the software and contain parameters of the respective file types for the command input at the prompt. When you install the software, they end up on the mass storage and accumulate in the `recipes` subfolder in the `/usr/share/magicrescue/` program directory.

The application launches as soon as the user types `magicrescue`, with an overview of the program options initially displayed. Before starting the actual search, create a target folder with `mkdir`, in which you store the found data. To reconstruct files on a block device, enter

```
magicrescue -r Recipe_1 <[...]>
  -r Recipe_<n>
  -d <target file> <source>
```

at the prompt. The `-r` parameter allows multiple recipe files to be recovered simultaneously in several formats. As a source, you specify a partition (e.g., `/dev/sdb1`). If you accidentally stipulate a directory or just the name of a block device, the software cancels the reconstruction with an error message.

If the message *Command not found* appears as a response to a correct command, the application is missing a third-party program that it needs to reconstruct certain file types. This happens especially with JPEG files, for which you need to install the *jpeg-progs* or *libjpeg-progs* package from the software repositories. Magic Rescue needs the `jpegtran` program to save to JPEG image files (Figure 4).

The subsequent reconstruction of selected databases takes quite some time, because Magic Rescue individually searches all sectors of the source partition. Therefore, the program is more suitable for smaller removable disks such as SD memory cards or USB flash drives. The application subsequently stores the recovered files in the target directory, without sorting them.

If you have specified several file types to be reconstructed, the target directory quickly becomes cluttered. Magic Rescue comes with a small command-line tool named `magicsort` as part of the program package and is used without further parameters. Simply enter `magicsort` at the prompt in the target directory; the software then sorts the files located in the target directory into their own subdirectories by file type.

## R-Studio

One of the most powerful tools for data reconstruction is the commercial R-Studio [8] developed by Canadian R-Tools Technology Inc. R-Studio for Linux is

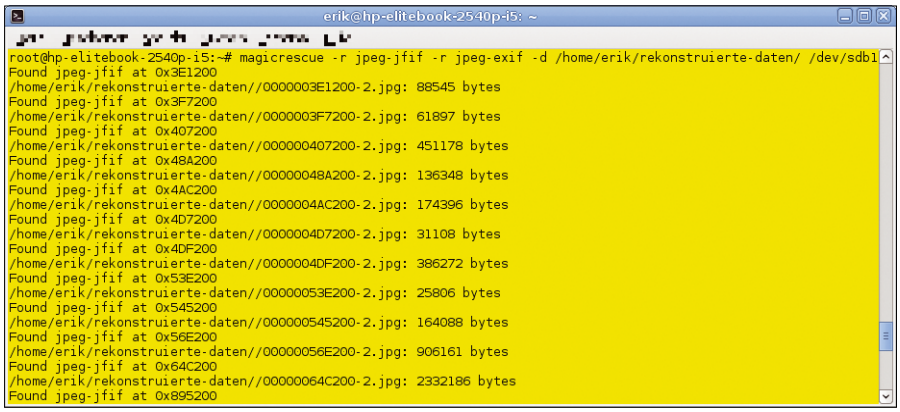



**Figure 4:** Magic Rescue rescuing an impressive number images that have been long forgotten.

available for around $80 (single-user license) or around $180 in the network variant, R-Studio for Linux Network, from the online store [9]. The manufacturer, who specializes in data recovery and data backup solutions, offers the program as an unlimited demo version for download from the website. However, the maximum size of files you can reconstruct is 256KB – sufficient for simple text files and small images.

R-Studio is available in RPM and DEB package versions for 32- and 64-bit architectures. To convert the demo version into a full version, you just need a license key, which you can purchase online [9].

R-Tools Technology also offers a free full version of R-Linux that doesn't require registration and is specifically for data recovery on ext2, ext3, and ext4 systems. R-Linux is decoupled from R-Studio and offers the same feature scope as the commercial version when reconstructing ext filesystems. However, R-Linux only runs only on single-user systems and does not support other filesystems.

R-Linux and R-Studio both require a 32- or 64-bit machine with 256MB memory and kernel 2.6. To store the reconstructed data, you need to have sufficient free space on the target medium. R-Studio reconstructs image files that can weigh in at several gigabytes. The graphical interface requires a standard Linux desktop.

R-Studio for Linux Network has an agent available for use on the client: The agent comes in 32- and 64-bit versions for Linux systems [10].

R-Studio packages run without problem on all standard Linux distributions that support the RPM or DEB package format.

In addition to the conventional menubar and buttonbar, the program window has three panes. After launch, the top left pane displays all the relevant data on the drives that are connected to the system, as well as the partitions. In the pane on the right, the top level shows device properties or some of the computer's properties (e.g., operating system, kernel version, CPU and RAM data). In the bottom pane, the log lists actions that the software performs during the session; this segment is initially empty.

To scan a storage device and reconstruct the data, first select the desired source partition. Right-click on it and select *Open Drive Files* to display the contents in the right segment of the program window. A small red cross to the left of the file name or folder symbol represents deleted files and directories that R-Studio can reconstruct. If the folder symbol also displays a red question mark, then it is likely that the contents of the folder in question cannot be reconstructed.

Under the upper left pane, you will find the *Extensions*, *Creation Time*, *Modification Time*, and *Access Time* tabs (Figure 5). These represent additional sorting criteria for organizing the disk structure.

If you are not sure what file content is hiding behind which name, you can right-click on the file and select *Preview* – as long as it can be reconstructed – to display the file content. The software will open a small overlay window that displays graphics in their original format. The software plays back video files in multiple formats and also visualizes PDF files.

If you have deleted and overwritten partitions by reformatting a disk, data recovery is not possible using the conventional methods. In this case, R-Studio offers an enhanced reconstruction option (Figure 6). Select the desired drive and either press *Scan* in the top buttonbar or, after right-clicking, choose the same option from the context menu.

R-Studio opens a small superimposed window, in which, on request, it defines not only an area to be scanned but also a destination for the scanned file. R-Studio searches the entire disk sector-by-sector for reconstructible data, which can take

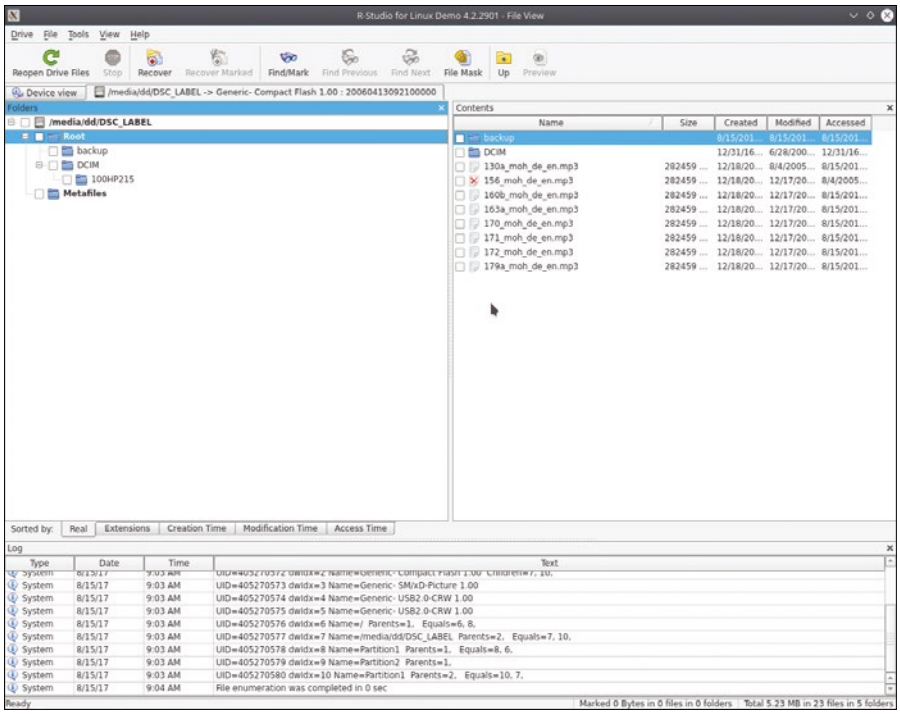



**Figure 5:** R-Studio displays recoverable content with small cross symbols.

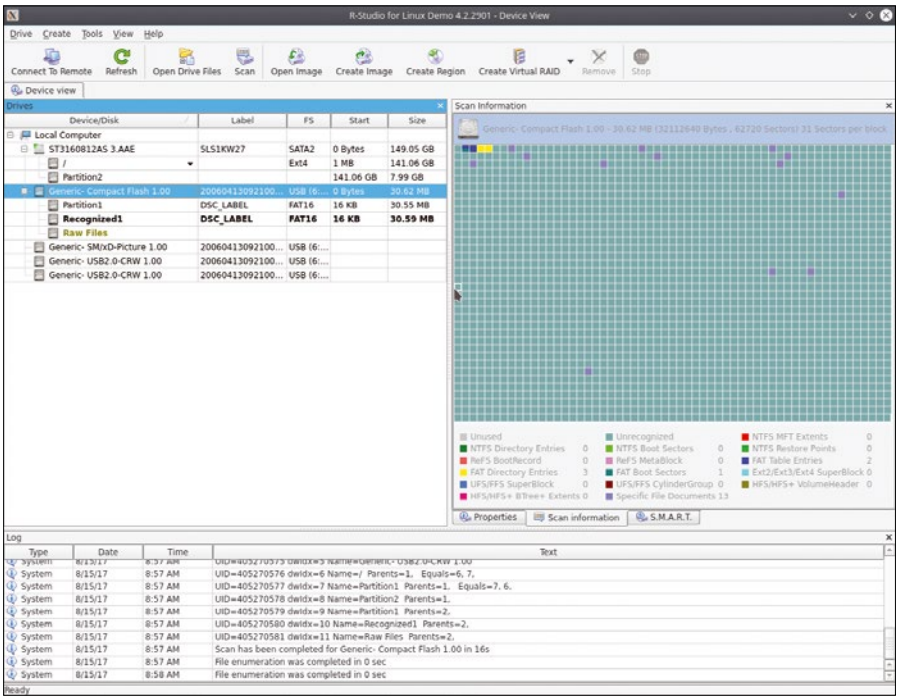


Figure 6: The in-depth scan in R-Studio brings further fragments of old partitions to light.

a considerable amount of time for larger storage media.

If you want to reconstruct data from a RAID system to which direct access is no longer possible (i.e., because the RAID controller has died), R-Studio can stimulate the RAID system using virtual RAID software. However, unlike a conventional RAID system, R-Studio doesn't give you write access.

## Foremost

As a command-line-only tool, Foremost [11], which uses a process called data carving, is one of the standard data recovery tools in Linux. The software tries to identify the existing data structures and recover occupied sectors by referencing the metadata, including the header, the footer, and – much like Magic Rescue – the magic number of the file. The program reanimates files up to a size of 2GB. Foremost v1.5.7 is available from the repositories of virtually all the major Linux distributions.

The software is designed to reconstruct numerous data formats out of the box: In addition to various multimedia files, it also recovers some binary and document formats without manual rework. It also recognizes some types of archives. Foremost also lets you integrate new file formats: This is done by editing the `/etc/foremost.conf` configuration file, which lists the known format types.

After installation, you call the software by entering the program name at the prompt followed by the `-t` parameter with the file type of the file to be recovered and the source disk, which can be an image or a block device, preceded in both cases by the `-i` parameter.

Foremost then creates the `output` folder in the current directory, followed by the file-type-specific subdirectories in which the reconstructed data is stored. Additionally, the software generates an audit file containing information on the disk reconstruction (Figure 7).

## The Sleuth Kit

One of the most powerful tool collections for forensic work is The Sleuth Kit (TSK) [12], which has been under continuous development for several years. The developer, Brian Carrier, also has a graphical front end called Autopsy [13]. However, the latest version 4.4.1 is only available as a Windows package. To use Autopsy with Linux, you must build it from the source code, which poses massive problems because of partly missing, partly incorrectly designated dependencies.

Because TSK is intended as a command-line tool for users who perform professional data analysis and forensic work, the tool collection is also suitable for many tasks beyond the actual reconstruction of data.

The tools can be found in the software repositories of all the major Linux distributions, as well as in many Live systems with a focus on IT security and data maintenance [14]. TSK includes many tools for data analysis and reconstruction; the individual modules collect a variety of information and take metadata, the filesystem, object names, journals, and data blocks into account.

As supported filesystems, the documentation refers to ISO-9660, standard FAT variants, NTFS, ext2/3/4, and the UFS1 and UFS2 filesystems used by several Unix derivatives. Because of the many parameters, TSK requires a longer training period and therefore is less suitable for occasional data recovery. The `fls` and `tsk_recover` com-

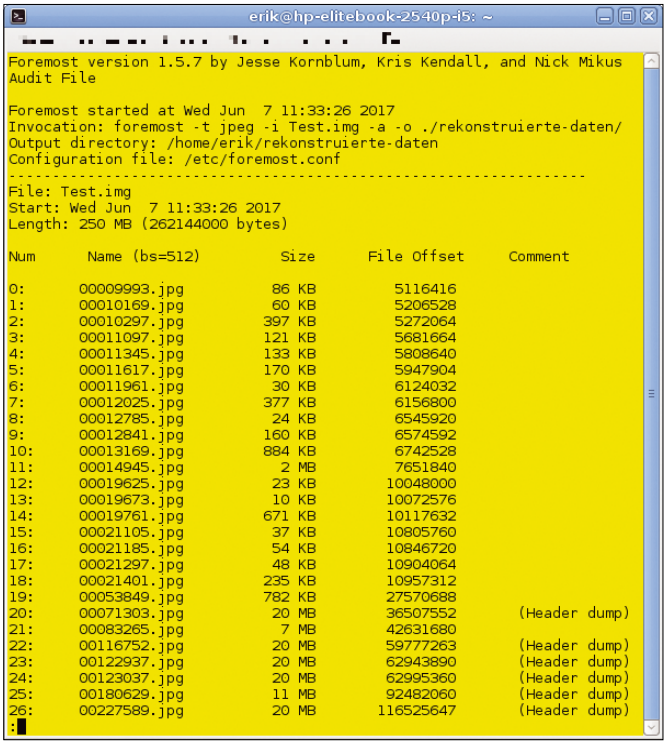



Figure 7: Foremost provides detailed information on the disk in an audit file.

## NOT INCLUDED

This test did not include the Scalpel [15] file carver. Although it provides functionality similar to Foremost, it only supports a few filesystems, which is not optimal, especially in environments with many different operating systems.

I also did not cover `ddrescue` [16]. Although the software is very well documented, it was not convincing during a practical test on larger drives with a large amount of lost data and a corrupted partitions table. Because `ddrescue` scans the disk several times, depending on the severity of the damage, an attempt to reconstruct the data can take several hours for USB memory sticks of just a few gigabytes. It was also impossible to repair disks that exhibited pronounced fragmentation of residual data as a result of overwriting in a reasonable amount of time.

mands are useful for simple data reconstruction; TSK expects to work on an image generated from the original file as a template, so that it does not destroy reconstructible content by means of accidental write processes on the original disk.

To reconstruct some data on a USB memory stick, first create an image of the damaged drive by using the `dd` Linux command; the content can then be displayed with the `fls <imagefile>` command. Files with an asterisk are deleted.

The inode number, which is also listed, then lets you reconstruct a file as follows:

```
istat <imagefile> <inode_number>
```

and

```
icat <imagefile> <inode_number> > ➥
     <filename>
```

`tsk_recover` also relies on other TSK modules when localizing and recovering deleted data; for example, it recovers all files of an image using the

```
tsk_recover ➥
  -e <imagefile> <target_directory>
```

command sequence. These then end up in subdirectories. The `-v` option in the command sequence also outputs details to the screen (Figure 8).

## Conclusions

All tested programs are up to the task of reconstructing accidentally deleted data or content from corrupted disks (see also the "Not Included" box). However, the approaches to restoring the data differ significantly: Whereas TSK is suitable for professional forensic applications by providing a variety of parameters, at the other end of the spectrum, Foremost and Magic Rescue are easy to use and provide rapid results.

Anyone who needs professional data reconstruction across an intranet might be interested in the commercial R-Studio. Although the software is not as good as TSK when it comes to court-proof documentation of forensic investigations, its ability to restore lost files outshines PhotoRec and TestDisk.

PhotoRec is well suited for occasional use in multiformat databases, because it provides the most extensive format support.

When it comes to finding a suitable solution, you should first determine your needs before choosing one of these programs for data recovery ■■■

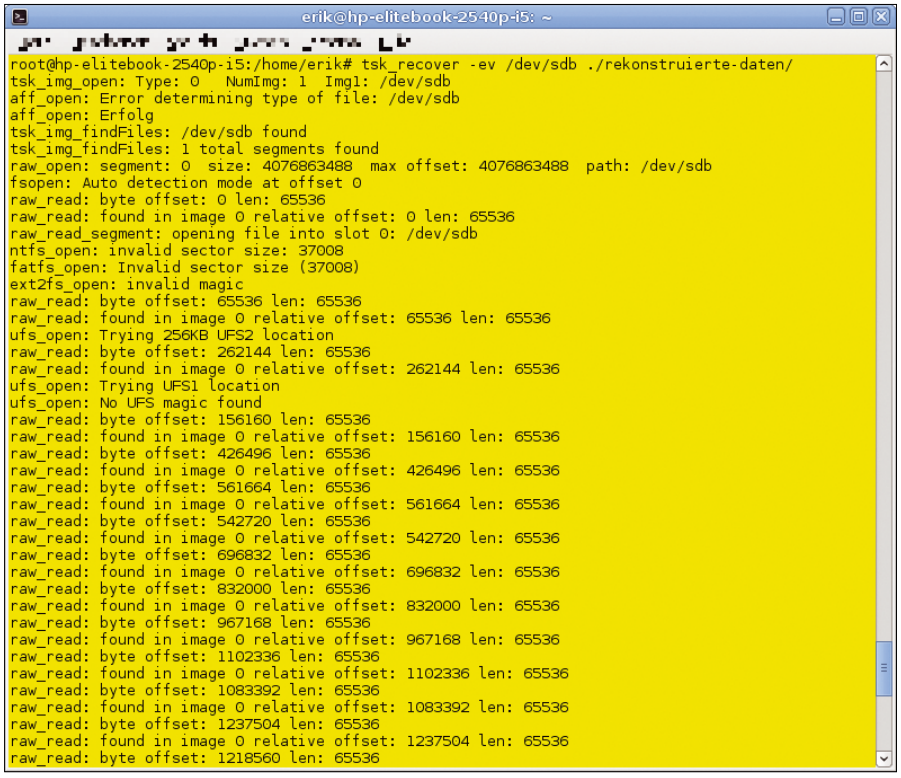



**Figure 8:** TSK modules aid in data reconstruction.

## INFO

[1] Backup survey: *https://www.backblaze.com/blog/data-backup-survey/*

[2] dcfldd: *http://dcfldd.sourceforge.net*

[3] PhotoRec: *http://www.cgsecurity.org/wiki/PhotoRec*

[4] TestDisk: *http://www.cgsecurity.org/wiki/TestDisk_Download*

[5] PhotoRec supported file formats: *http://www.cgsecurity.org/wiki/File_Formats_Recovered_By_PhotoRec*

[6] List of Live systems with TestDisk: *http://www.cgsecurity.org/wiki/TestDisk_Livecd*

[7] Magic Rescue: *http://freecode.com/projects/magicrescue*

[8] R-Studio: *http://www.r-studio.com*

[9] R-Studio online store: *http://www.r-tt.com/BuyOnLine.shtml*

[10] Downloading the agent: *http://www.r-studio.com/data_recovery_linux/Download.shtml*

[11] Foremost: *http://foremost.sourceforge.net*

[12] TSK: *https://www.sleuthkit.org*

[13] Autopsy: *https://www.sleuthkit.org/autopsy/*

[14] List of distributions and tools with TSK or Autopsy: *http://wiki.sleuthkit.org/index.php?title=Tools_Using_TSK_or_Autopsy*

[15] Scalpel: *https://sourceforge.net/projects/scalpel/*

[16] ddrescue: *https://www.gnu.org/software/ddrescue/*

## Devuan: Debian without systemd

# New Planet

**In 2014, the Debian project decided to replace the old init system with systemd, but a small group of developers resisted, forking Debian to start the systemd-free Devuan. We decided to take a look at Devuan 1.0.0, the first stable release.** *By Antony Stone and Heike Jurzik*

Init has a crucial role on any Unix or Linux computer: The init service, which is often referred to as a "daemon-starting daemon," handles the process of launching, managing, and terminating other services running on the system. Linux has depended on the venerable SysVinit for most of its history, but a few years ago, many Linux developers began to grow restless with System V. The basic argument was that SysVinit, which had served the community well for many years, was inefficient and out of step with contemporary hardware and programming practice.

A pair of competing alternatives eventually emerged: Upstart, which was backed by Canonical, and systemd, which began as a Red Hat project. The two new inits competed for market share and public awareness for a few years, but gradually, Linux distros began to express a preference for systemd. In 2014, Canonical founder Mark Shuttleworth announced that even Ubuntu would migrate to systemd instead of Upstart, which left systemd as the last alternative to System V.

In recent years, most major Linux distros have adopted systemd as their init system, including Red Hat, Fedora, SUSE, Arch, and Ubuntu. The discussion of whether to replace System V with systemd was particularly intense and polarizing at Debian. The Debian project, which is the largest community-driven Linux distribution, is particularly influential because it serves as a basis for several other Linux distros.

Many developers approved of the new features introduced with systemd, including parallel processing and a versatile system of interfaces. Proponents argued that systemd would bring better performance and reduce compatibility issues among different Linux distros. Another group of developers argued that, regardless of whether systemd was actually better than the old SysVinit, the task of keeping the old System V compatible alone, without the help of the rest of the Linux community, would require too much overhead.

In spite of these advertised benefits, a large number of developers at Debian (and elsewhere in the Linux community) were not happy with the change to systemd. In fact, many were quite angry.

The main argument against systemd was that it was way too big and complex, and it was therefore a violation of the time-honored Unix principle of simplicity. These coders saw no benefit in replacing a simple and well tested system component with a massive and many-layered alternative. In addition to the aesthetic issues with abandoning the principle of simplicity, these developers argued that a bigger and more complex init system posed some very practical problems, including a higher probability of bugs and a bigger attack surface for cyber intruders. (See *without-systemd.org* for a summary of the arguments against adopting systemd init [1].)

In November 2014, a group of developers who objected to Debian's systemd migration forked Debian and started a new project, which they dubbed Devuan [2]. Devuan Linux was envisioned as a Linux that would keep the System V system alive for developers and users who rejected the code bloat and mission creep of systemd. (The Debian wiki contains instructions [3] on successfully installing Jessie without systemd using preseeding, but this procedure only works on servers without graphical working environments.)

Since 2014, a group called the Veteran Unix Admins has maintained Devuan. May 2017 marked the first stable release of the systemd-free Devuan fork. Devuan 1.0.0 carries


**Figure 1:** The Devuan installer looks very similar to its Debian counterpart.

the LTS seal of approval – the developers say its long-term support will be at least as long as the support for Debian Jessie.

With the arrival of the first stable release, we decided it was time to boot up Devuan 1.0 and have a look around.

## Introducing Devuan

The first major release of Devuan uses the nickname "Jessie," signifying its compatibility with the Debian 8.0 series. Later releases, however, will be named for minor planets, as classified by the International Astronomical Union [4]. The current testing branch is called ASCII (which, in addition to its meaning within the computer industry, is also the name of a minor planet). For the next release, the developers will choose a minor planet starting with the letter B, and subsequent releases will progress through the alphabet. The Unstable branch is always named Ceres. To remain compatible with Debian, the developers have set up an alias for Unstable, named Sid.

Various Devuan images are available in the download area [5], including hybrid DVD images for Live operations, ARM images for embedded devices, as well as images for Vagrant, Qemu, and VirtualBox. If you wish to install Devuan, you will find CD and DVD images (32- and 64-bit). In our lab, we used the 64-bit DVD image for a fresh Devuan installation. We also transformed an existing Debian Jessie into a Devuan Jessie via an online update. Finally, we looked at what happens when you upgrade Debian Wheezy to Devuan Jessie.

## First Look

At first glance, a fresh Devuan installation hardly differs from Debian. After booting (Figure 1), select the language and wait for the hardware detection and network setup. After creating the user account and choosing the time zone, partition the hard disk. Then install the basic system on your disk. When setting up the online sources for the package manager, you will notice that the mirrors list is significantly shorter than the Debian list. Two servers are available for each listed country: one with the country code and `auto.mirror.devuan.org`.

You start to notice some divergence from Debian when you select a software package. If you opt for a graphical environment, you can choose between Xfce (standard), LXDE, and Mate – no trace of KDE or Gnome. In the announcement for the second release candidate [6], the developers reported that they have removed the major desktop environments from the installer for stability reasons. You can add KDE or Gnome manually, but some components still cause problems. If you stumble across an error, report it in the bug tracker [7].

We couldn't get the Gnome desktop to work on our lab computer, but we managed a retroactive installation of KDE Plasma using the `task-kde-desktop` package without any problems. When importing from KDE, the post-installation routine proposes the KDM login manager as the default. We recommend you decline the KDM option and stay with SLiM.

## Quick-Change Artist

Even after Debian 9 "Stretch" is released, and Jessie is thus the Old Stable, converting from Jessie to Jessie will still be the best way to move to Devuan. The developers explicitly advise against upgrading to Devuan Testing or Unstable.

As with a Debian distribution upgrade, the Devuan documentation [8] recommends disabling third-party sources and removing the packages from them. Adjust the list of repositories, enter the Devuan mirror (Figure 2), and use `apt-get update` to update the package list. Then install the GPG keyring of the Devuan project to allow a non-authenticated package:

```
apt-get install devuan-keyring
  -y --allow-unauthenticated
```

Then update the packages and perform the upgrade:

```
apt-get update
apt-get dist-upgrade
```

The package manager removes `systemd-sysv`. Contrary to the description in the documentation, Xfce and SLiM installed automatically on our lab machine


**Figure 2:** First adjust the list of sources and enter the Devuan mirror.

and did not require manual installation. If the post-installation routine asks for the default display manager (Figure 3), choose SLiM.

For the last step, end the current X session and continue to work at the console. After the upgrade, check to see if any remnants of systemd are still present. The `systemd-shim` package was still installed on the lab computer. If you remove it using `apt-get purge`, you will see some packages disappear, including `gdm3`, `gnome`, and the Wicd Network Manager.

## Challenging

If you are still running Debian Wheezy and want to convert to Devuan Jessie, you'll experience some of the same problems that occur when upgrading from Debian Wheezy to Debian Jessie. After adjusting the repositories, install the Devuan keyring and run the `apt-get dist-upgrade` command. The package manager will ask you the usual questions about receiving the configuration files and then proceed to install the updates and additional required software.

On our lab machine, we had an unpleasant surprise with the KDE desktop. The new KMail version, which relies on the Akonadi data management service, did not install properly. If you follow the program's prompt and start the conversion wizard, you may be waiting for some time, depending on the volume of messages and folders. All the filters were missing on the test system. The following commands gave us the right results:

```
akonadictl stop
rm -rf ~/.local/share/akonadi/
rm -rf ~/.config/akonadi/
rm ~/.kde/share/config/akonadi*
akonadictl start
```

We then relaunched the migration. We initially did not enter the passwords for the accounts and imported all the filter rules; this time the migration was successful. After several hours, KMail was still unusable – it was impossible to read or delete messages.

On the second computer, we found two packages with configuration files that had not been removed (`libsystemd-daemon0` and `libsystemd-login0`). The `libsystemd0` package was marked as automatically installed, and it remained as a dependency of `xsane`, `libsane`, and `sane-utils` – support for the scanner does not seem to be fully developed.

The `init` metapackage in Devuan ensures that one of the available init system (`systemd`, `sysvinit`, or `upstart`) is installed if a program requires it. Because many applications from graphical environments depend on D-Bus, and the standard D-Bus component requires systemd on Debian, the Devuan team built their own D-Bus from the upstream sources.

## Reach for the Stars

We noticed a few minor issues with setting up Devuan. The auto-mount feature caused a couple of hiccups on both lab computers. One system required us to mount the removable disk manually; the other asked for the root password before mounting.

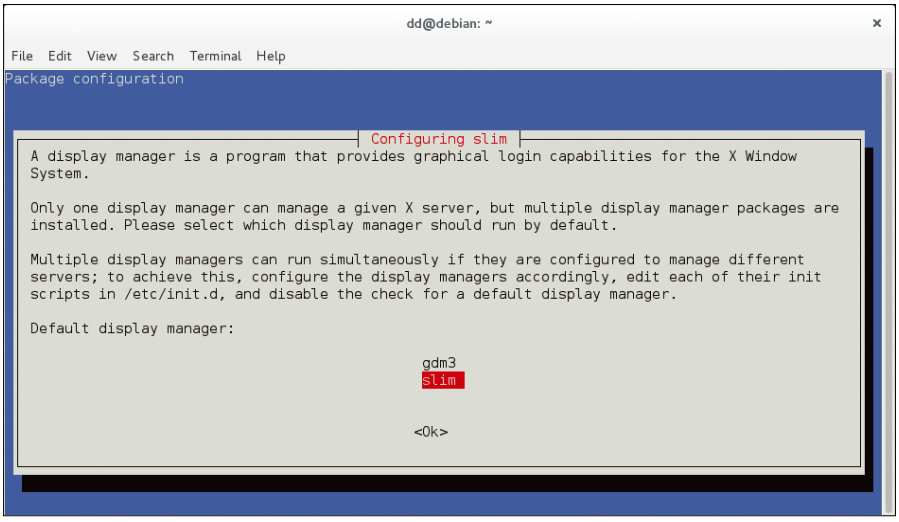



**Figure 3:** Opt for SLiM, the lean login manager, rather than GDM or KDM.

Overall, Devuan is a success. The fact that switching from Debian Wheezy to Devuan Jessie caused such problems appeared to be an issue with KMail and not the distribution, as Debian users have also reported difficulties in forums. The change to the SLiM login manager does not pose a problem. If you cannot get along with Xfce, install KDE, LXDE, or Mate.

Only Gnome fans are left out in the cold – and that will not change in the future, as we learned from the developers, who are more interested in pushing forward with the next release than integrating the huge Gnome codebase. If everything runs smoothly, the first images for the upcoming ASCII release will soon be available on the Devuan servers.

It is not just long-serving Unix veterans who are interested in a systemd-free distribution. The project page shows more than ten distributions based on Devuan. And things are also happening in the enterprise field: Centurion Computer Technology (New Zealand) and Dyne Solutions (Netherlands) are in the starting blocks to provide customized Devuan derivatives and support [9] to their customers. ■■■

## INFO

[1] Arguments against systemd: *http://without-systemd.org/wiki/index.php/Arguments_against_systemd*

[2] Devuan: *https://devuan.org*

[3] Install Jessie without systemd: *https://wiki.debian.org/systemd#Installing_without_systemd*

[4] Minor planets: *http://www.minorplanetcenter.net/iau/lists/MPNames.html*

[5] Devuan download: *https://files.devuan.org*

[6] Blog post on the release candidate: *https://devuan.org/os/debian-fork/stable-2nd-candidate-announce-050517*

[7] Devuan bug tracker: *https://bugs.devuan.org*

[8] Instructions for the Devuan upgrade: *https://devuan.org/os/documentation/dev1fanboy/Upgrade-to-Devuan*

[9] Devuan 1.0 release notes : *https://devuan.org/os/debian-fork/stable-jessie-announce-052517*

# What?!

## Archives

come with my

digital subscription?


## Archives + Current!

Sign up for a digital subscription to get the latest issues of Linux Magazine, **PLUS access to archive articles.**

shop.linuxnewmedia.com/digisub

**That's a lot of articles!**

## The sys admin's daily grind: Rclone

# Drive-In

**Having a good backup is a matter of course for sys admin columnist Charly Kühnast, but devices could still fall victim to fire or theft some day. Because he has enough free space on Google Drive, he doesn't need to search long for a solution. The only thing missing is the right tool.** *By Charly Kühnast*

I never wanted to have to lament lost data, which is why I have functioning backups at home. Nevertheless, having an offshore backup seemed to be a good idea in case of a chain of ridiculous circumstances (i.e., absolute disaster). As an Android user, Google gives me 15GB of disk space, which I don't currently use. But the right tool is still missing: Rclone.

Completed binary packages for Linux, Windows, Mac OS, the BSDs, Plan 9, and Solaris can be downloaded from Rclone's website [1]. If you would prefer to build your own and you have a Go compiler on your system, you can proceed with the command

```
go get -u -v github.com/ncw/rclone
```

at the start. I launch the configuration of the storage back ends with `rclone config` and, with `n`, launch the path for *New remote*. I use `mygdrive` for the name now requested for the remote connection. Figure 1 makes it clear that there should be something for everyone in the supported storage services – even generic FTP/SFTP to set up your own backup corner on a rented server or web space.

After choosing *Google Drive* and the automatic configuration, a browser opens so that I can let Rclone access my file storage on Google. This is the end of the configuration.

```
root@funghi:~# rclone config
2017/07/16 15:09:01 NOTICE: Config file "/root/.config/rclone/rclo
e.conf" not found - using defaults
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q> n
name> mygdrive
Type of storage to configure.
Choose a number from below, or type in your own value
 1 / Amazon Drive
   \ "amazon cloud drive"
 2 / Amazon S3 (also Dreamhost, Ceph, Minio)
   \ "s3"
 3 / Backblaze B2
   \ "b2"
 4 / Dropbox
   \ "dropbox"
 5 / Encrypt/Decrypt a remote
   \ "crypt"
 6 / FTP Connection
   \ "ftp"
 7 / Google Cloud Storage (this is not Google Drive)
   \ "google cloud storage"
 8 / Google Drive
   \ "drive"
 9 / Hubic
   \ "hubic"
10 / Local Disk
   \ "local"
11 / Microsoft OneDrive
   \ "onedrive"
12 / Openstack Swift (Rackspace Cloud Files, Memset Memstore, OVH)
   \ "swift"
13 / SSH/SFTP Connection
   \ "sftp"
14 / Yandex Disk
   \ "yandex"
Storage>
```

**Figure 1:** Rclone helps you with all major cloud driver providers.

To check what I've saved on the remote side, the following command suffices:

```
rclone ls mygdrive:
```

The colon is mandatory. Because my drive is empty, there is no output. To test it, I copy my Apache config files into the drive:

```
rclone copy -L -v /etc/apache2/ ➥
  mygdrive:backup
```

This takes a second, because Google has relatively aggressive rate limiting (about two files per second). I check straightaway whether the files have arrived in Google's web interface (Figure 2).

## With History

Google provides a small versioning drive. If I upload changed Apache files again, the drive recognizes this and keep both files for a maximum of 30 days or 100 changes, whichever comes first. I now use the `sync` sub-command instead of `copy`:

```
rclone sync -L -v /etc/apache2/ ➥
  mygdrive:backup
```

Rclone synchronizes the local and remote pages and only copies the files that have changed. Files that you delete locally meet the same fate on Google Drive – not vice versa. Rclone cannot manage to synchronize both sides like Rsync. However, I can encrypt files for storage by choosing *Encrypt/Decrypt a remote* in `rclone config`.

Rclone is still in active development, so you can watch it closely at work. ■■■

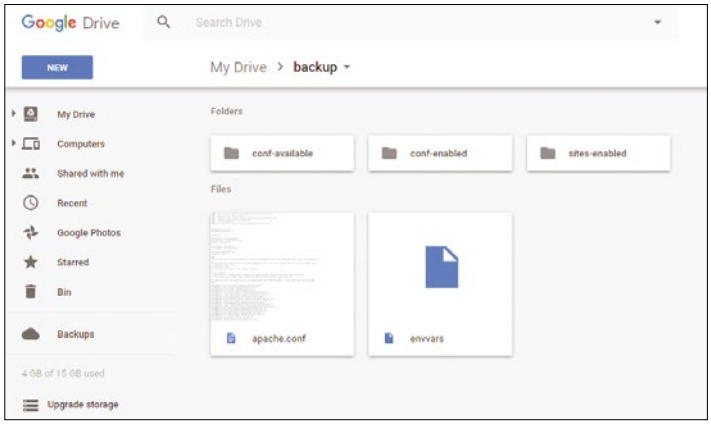



**Figure 2:** Google's web front end shows that the example files transmitted via Rclone have arrived safely.

## CHARLY KÜHNAST

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

## INFO

[1] rclone: *https://rclone.org/downloads/*

## Android in an LXC container

# Well Packed

**Need Android apps in a separate window on the Linux desktop? Anbox is the answer. Unlike common Android emulators, the software relies on LXC containers and kernel namespaces. We tested the pre-alpha version.** *By Heike Jurzik*

In April 2017, developer Simon Fels introduced an early development version of his Android in a Box (Anbox) program [1]. Released under a GPLv3 and Apache license, Anbox brings favorite apps from the Android smartphone or tablet to the Linux desktop. The software uses a different approach than emulators like Shashlik [2] and Genymotion [3] or the development environment Android Studio [4]. To bring the apps natively to the desktop, Anbox encapsulates the Android system and the apps in a LXC container, to isolate them from the Linux system.

According to statements by the developer, the project is the result of a technical incentive and not aimed at a particular target group. The idea behind the packaged Android is good: The Anbox run time essentially comprises the session manager and the container manager, which is responsible for setting up the LXC container and exchanges data with the session manager.

In turn, the session manager communicates with the Android container via multiple sockets and ensures the integration of apps on your desktop when logged in.

For a more detailed overview of the architecture and interaction of the components, you can read an article in the project's GitHub repository [5].

## In the Box

Officially, Anbox currently will only run on Ubuntu LTS v16.04, but according to the readme in the GitHub repository, it should also work on 14.04, 16.10, and 17.04. I ran Ubuntu 16.04 with the Unity desktop on the lab computer.

Snap loads the Anbox installer onto the system:

```
snap install --classic anbox-installer
```

Alternatively, the installer script is available for download from the project page. You can then launch the tool with the `anbox-installer` command; the script uses `sudo` and asks for a password if required.

The installer sets up the developer's PPA as a packet source, retrieves a package with two kernel modules (`ashmem` and `binder`) for the Android container, installs Anbox itself, and sets up an Upstart job on login. For the Android apps to appear in the desktop environment's application launcher, you must reboot the computer.

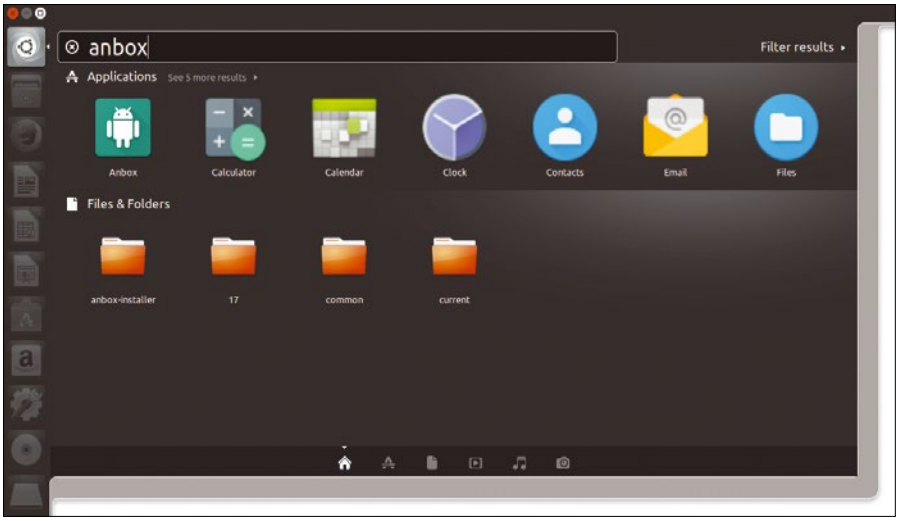



**Figure 1:** The Anbox Application Manager and all installed apps are available in the Unity Dash.

Lead Image ©bekas007, 123RF.com

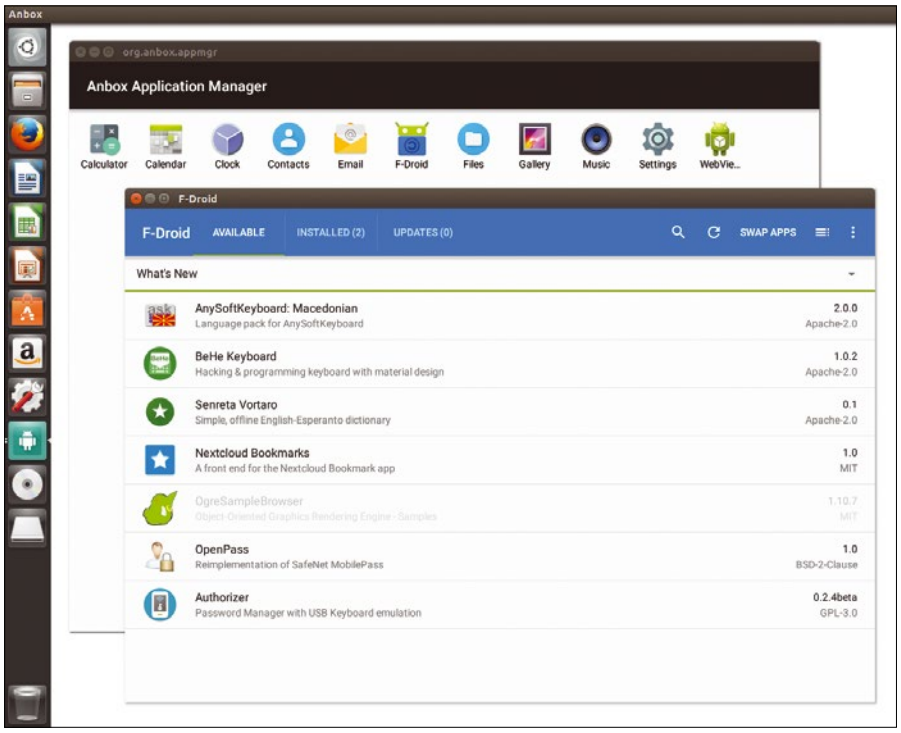



**Figure 2:** The alternative app store F-Droid offers free and open source Android apps for the Anbox system.

Searching for *anbox* in the Unity Dash shows several hits after the next login (Figure 1): *anbox* launches the Anbox Application Manager, which leads to all the installed apps. Alternatively, you can start the program by clicking on the icon in the Start menu.

## Filler Material

A minimal Android image, which is based on Android 7.1.1, runs in the Android container. The Anbox developers used the sources from the Android Open Source Project [6] and adapted them to their needs [7]. Support for alternative factory images from Google or custom ROMs is not intended. These are tailored to the respective devices and have no relevance to Anbox.

The DIY Android does not support Google accounts, so Play Store integration is missing. New apps are therefore installed on the Android system via the Android Debug Bridge [8]. Install the `android-tools-adb` packet, load the desired APK file from a mirror, and install using `adb install file.apk`. Alternatively, install the F-Droid app [9] and get access to an alternative app store with exclusively free and open source apps (Figure 2).

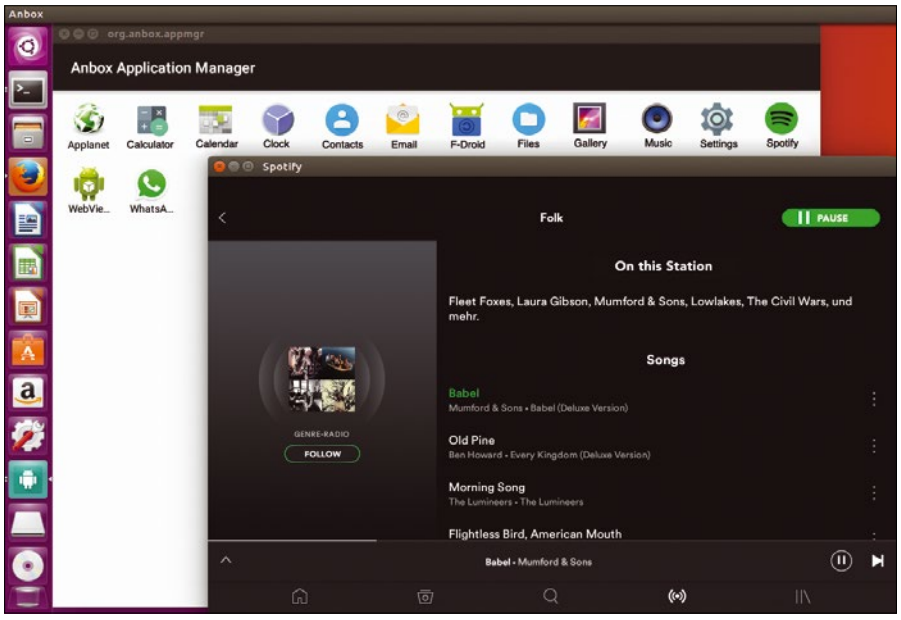



**Figure 3:** Anbox is network capable and makes music; therefore, apps like Spotify already work.

## Impressive

At the time of testing, the network and PulseAudio access already worked from within Anbox (Figure 3). Some apps ran well, whereas others would not cooperate. If you encounter a bug and want to discuss it with other Anbox users, search for the GitHub bug tracker on the project page [10]. The pre-alpha edition still does not let you exchange images, videos, or music between Android and Linux computers, but this feature will be available in one of the future versions.

Many Android apps are not yet ready for use on the Linux desktop, but this will probably improve in the next few versions of Android. *Linux Magazine* asked how the project itself is looking to move forward. According to statements by Simon Fels, the main focus is currently on stability. In order for Anbox to reach the alpha or beta status, some work on the Android run time is still necessary – the project welcomes capable developers from the Android camp. ■■■

## INFO

[1] Anbox: *https://anbox.io*
[2] Shashlik: *http://www.shashlik.io*
[3] Genymotion: *https://www.genymotion.com*
[4] Android Studio: *https://developer.android.com/studio*
[5] Anbox run time (overview): *https://github.com/anbox/anbox/blob/master/docs/runtime-setup.md*
[6] Android Open Source Project: *https://source.android.com*
[7] Building an Android image for Anbox: *https://github.com/anbox/anbox/blob/master/docs/build-android.md*
[8] Android Debug Bridge: *https://developer.android.com/studio/command-line/adb.html*
[9] F-Droid: *https://f-droid.org*
[10] Anbox issues: *https://github.com/anbox/anbox/issues*

AI learning with decision trees

# Horses and Riders

**Decision trees help machines emulate human behavior. Thanks to artificial intelligence, programs can autonomously build decision trees by learning from existing real-life data. Can an AI system determine who was operating a particular vehicle by analyzing driving style?**

*By Mike Schilli*

In the 1950s, TV stations were broadcasting howlers that probably wouldn't interest anyone today. One was "What's My Line?" with presenter John Daly, in which a blindfolded team of four panelists had to guess the identity of a star guest using only yes or no questions.

A popular strategy was to pose general questions first ("Are you male?"), in order to ask more specific questions in the final round, until the panelists began to close in on the guest star and their identity was finally revealed.

**TABLE 1: *x*/*y* AND Connection**

| Input | | Output |
|---|---|---|
| *x* | *y* | *x* AND *y* |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

In machine learning, programmers employ similar techniques to teach computers to imitate learned behaviors. As a somewhat contrived example, take the behavior of an AND gate (Table 1), which always shows the `0` value at output until a `1` is applied to both inputs. The gate is normally implemented by binary operators, but for the purpose of this investigation, I'll be using the decision tree in Figure 1.

The algorithm starts at the head node and checks whether the `X` input variable is set to `1`. If this is not the case, further testing is unnecessary, and the result of `0` is final. If `X` is `1`, on the other hand, the algorithm proceeds to the lower left to the `Y==1?` node. After evaluation, the end result is clear: If `Y` is also equal to `1`, the AND gate displays `1`, but if `Y` is `0`, the gate value is `0`.

## Supervised Learning

Now, of course, artificial intelligence (AI) can capture and simulate far more complex relationships than simple AND gates. However, the decision trees used there follow exactly the same procedure as in our simple example. In supervised learning, they receive input values such as `X` and `Y` and the expected output values, like the output values of the AND gate. From this, decision trees are automatically being built internally and gradually honed until, in a fully trained state in production, the system generates exact results from new live input values – or at least values close to the ideal result.

AI kits [1] like the Python module `scikit-learn` help automatically create these decision trees, which are based on live input data; it is easily installed using

```
pip3 install scikit-learn scipy
```

preferably in a `virtualenv` environment, to avoid clutter on your computer:

```
virtualenv -p /usr/local/bin/python3 dt
source dt/bin/activate
pip3 install [...]
```

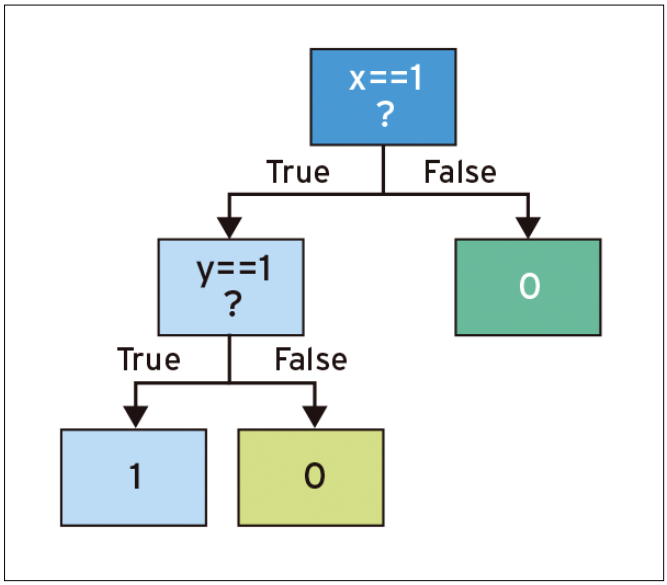



**Figure 1:** Manually created decision tree for AND gates.

### LISTING 1: decision-tree.py

```
01 #!dt/bin/python3
02 from sklearn import tree
03 import pydotplus
04
05 X=[[0,0],\
06    [0,1],\
07    [1,0],\
08    [1,1]]
09
10 Y=[0,0,0,1]
11
12 clf = tree.DecisionTreeClassifier()
13 clf = clf.fit(X, Y)
14
15   # dump decision tree
16 dot_data = tree.export_graphviz(
17     clf, out_file=None,
18     filled=True, rounded=True,
19     feature_names=['X','Y'],
20     class_names=['0','1'])
21 graph = \
22     pydotplus.graph_from_dot_data(dot_data)
23 graph.write_png("tree.png")
24
25   # Check predictions
26 for input in X:
27     print( input, ":",
28            clf.predict( [input] ) )
```

Listing 1 [2] additionally imports the `pydotplus` module to print the decision tree for study purposes.

## Mandatory Algorithm

The `X` Python list contains the previously known input values of the gate as a series of *x*/*y* values, also in list form. The `Y` variable contains the corresponding output values in the same order. The `tree` class from the `scikit-learn` module (the abbreviated name `sklearn` is also valid) now offers the `DecisionTreeClassifier`, which approximates relationships by fitting with the `fit()` method; internally, it builds a tree and later predicts the results for new input values using `predict()`.

To take a look at the internal processes in the production line of this procedure, the `export_graphviz()` method collects the generated tree and outputs it in text notation for the `graphviz` chart tool. The `pydotplus` Python module (installed with `pip3`) creates the PNG file in Figure 2, which requires the `graphviz` package.

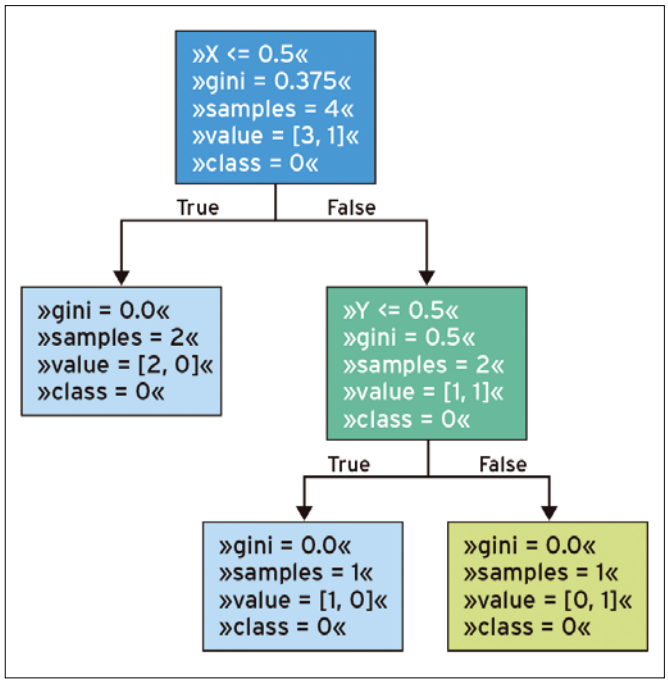



**Figure 2:** Automatically generated decision tree for the AND gate.

```
$ ./decision-tree.py
[0, 0] : [0]
[0, 1] : [0]
[1, 0] : [0]
[1, 1] : [1]
$
```

**Figure 3:** After the learning phase, the script can reproduce the output values of the AND gate.

You can install the `graphviz` package on Ubuntu by typing:

```
sudo apt-get install graphviz
```

The output from the script in Figure 3 shows that, after the fitting phase, perfect results are obtained from all possible input values.

At first glance, the machine-generated decision tree in Figure 2 looks a bit unorthodox, but machines think differently from humans; they just rigidly follow their algorithms. For example, the classifier first checks whether `X` is less than 0.5, and if that is the case, it outputs `class=0` as a result in the True node below left. If `X` is greater than 0.5, however, the False node below right checks to see whether `Y` is less than 0.5 and displays `0` if yes and `1` if no; that is, it's equal to `1` in real life.

The procedure performs as optimally as the tree in Figure 1 but – typical of a machine – somewhat non-humanly, just like the AlphaGo mastermind [3], which now ranks among the world's top Go board players, with some seemingly absurd moves that no one has ever tried before.

## From Insecure to Secure

The method used by `scikit-learn` [4] is mathematically proven and builds the tree by splitting the total of all possible input tuples and their preexisting results into subsets. Each division aims to minimize uncertainty in the system – its entropy. At the head node, the entropy is still high because nothing is yet known about the result, so the algorithm would have to select either `0` or `1` as a result. Because the output is `0` in 75 percent of all cases, and 1 in 25 percent of cases (Table 1), the entropy of the system is initially:

$$-[(1/4) \times \ln(1/4)] - [(3/4) \times \ln(3/4)] = 0.56$$

If the first node separates the possible input values into two subsets by the `X<=0.5` comparison,

```
[[0,0],[0,1]]
```

and

```
[[1,0],[1,1]]
```

then the entropy of the first subset is equal to 0 (the result is 0 in any case; that is, it's certain), and the second stays 0.56, so the total entropy is:

$$-(1/2 \times 0) + (1/2 \times 0.56) = 0.28$$

With only one node in the tree, the system's entropy has decreased from 0.56 to 0.28. An additional node splits the remaining two result tuples into two individual results. In the case of `X>0.5` where `Y<=0.5`, the entropy of the whole system drops to 0. Therefore, with two nodes, the tree can correctly predict the result for each arbitrary input tuple with 100 percent probability.

In this case, it is of academic interest at most, because an AND gate can be set up in hardware or firmware much more efficiently. However, an algorithm that knows nothing of the internal structure at first and only learns from combinations of input and output values, mimicking the behavior of the unknown system, can be used to tackle arbitrarily complex problems. As a bonus, it even supports non-permitted input values such as `[0.9,0.8]`, which it transforms into useful results (in this case, `1`).

## Who Drove?

As a practical AI application, let's look at a decision tree based on car driving data to determine who operated a vehicle. As in the previous Programming Snapshot [5], I prepared data collected by my Automatic adapters [6] in my two cars, which recorded when, where, and how fast the vehicles were moving.

The only thing the adapter doesn't know is who was driving, and since my wife and I alternately drive both cars, I'd like to teach an algorithm to guess the driver by looking at the trip data. To get the project off the ground with some sample data from which to learn, I added driver abbreviations `M` or `A` in Listing 2 on selected trips where I knew who was driving at the time.

Each line in the CSV file in Listing 2 represents a recorded drive; the second to last column, `vehicle`, indicates whether commute mule Honda Fit (`1`) or my sportier 1998 Acura Integra (`2`) was used. My wife rarely drives the latter, but she often drives the mule to work during the week (`dow = 1-5`), while I tend to take the Integra for a spin on the weekends (`dow = 6-7`).

Whether the Honda or the Acura, the `speed` column provided by the Automatic adapter seems to give me more points than my wife as a driver, for reasons that are incomprehensible to me. The brake or the acceleration ratings (`brakes` and `ac-`

**LISTING 2: trips-learn.csv**

```
01 dow,miles,brakes,accels,speed,
   vehicle,driver
02 3,5894.2,0,0,0,1,A
03 3,471.4,0,0,0,1,A
04 2,1279.4,0,0,0,1,A
05 4,21876.9,5,2,0,1,A
06 4,1510,1,1,0,1,A
07 4,20586.9,3,0,0,1,A
08 3,22381.9,1,1,0,1,A
09 2,39883.3,2,2,18,1,A
10 1,2005.6,2,4,0,1,A
11 3,16131.6,4,2,6,2,M
12 7,11788.7,1,0,14,1,M
13 6,19103.8,0,2,20,1,M
14 5,21384.3,1,0,15,2,M
```

**LISTING 3: driver.py**

```
01 #!dt/bin/python3
02 from sklearn import tree
03 import pandas as pd
04
05 df = pd.read_csv("trips-learn.csv",
06         header = 0)
07
08 Y = df['driver']
09 X = df[['dow', 'miles', 'brakes', 'accels',
10         'speed', 'vehicle']]
11
12 clf = tree.DecisionTreeClassifier()
13 clf = clf.fit(X, Y)
14
15   # Check predictions
16 for input in [[6,2000,0,2,20,1],
17               [6,2000,0,2,0,1]]:
18     print( input, ":",
19            clf.predict( [input] ) )
```

```
$ ./driver.py
[6, 2000, 0, 2, 20, 1] : ['M']
[6, 2000, 0, 2, 0, 1] : ['A']
$
```

**Figure 4:** After the learning phase, the decision tree identifies the driver from their driving behavior.

cels) seem to be evenly distributed. Are these criteria enough to teach the system to guess the driver correctly on newly recorded trips with unknown drivers?

## Driving Experience

Listing 3 loads the CSV file into a Python Pandas dataframe. The `Y` list takes the entries that have been added to the `driver` column manually as desired results as `M` or `A`. `X` contains a two-dimensional list defined in line 9, in which the trip data are available in rows with the day of the week (`dow`: 1-7), the number of miles traveled (`miles`), hard brake and acceleration counters (`brakes` and `accels`), a speed rating (`speed`), and the vehicle ID (`vehicle`).

As in the previously discussed, more academic case, the `sklearn` class `DecisionTreeClassifier` is also used in Listing 3; its `fit()` method processes the training data to enable the model to later predict new results with `predict()`.

Somewhat surprisingly, for new trip records shown in `driver.py` (Figure 4), the model guesses pretty well, even when some columns like the mileage contain values that haven't been seen before. For the same car (`1`: Honda Fit), only because of the driver's higher speed rating, the algorithm assigns the trip to `M`, and to `A` in the other case. After an in-depth analysis, the decision tree must have determined that this is the key distinguishing feature between the two drivers.

For a first attempt, the process produces very good results; more collected live data later on will show how reliable they really are. If improvements are needed, more training data will help produce a more accurate decision tree. ■■■

## INFO

[1] Vo.T.H, Phuong, Martin Czygan, Ashish Kumar, and Kirthi Raman. *Python: Data Analytics and Visualization*. Packt Publishing, 2017

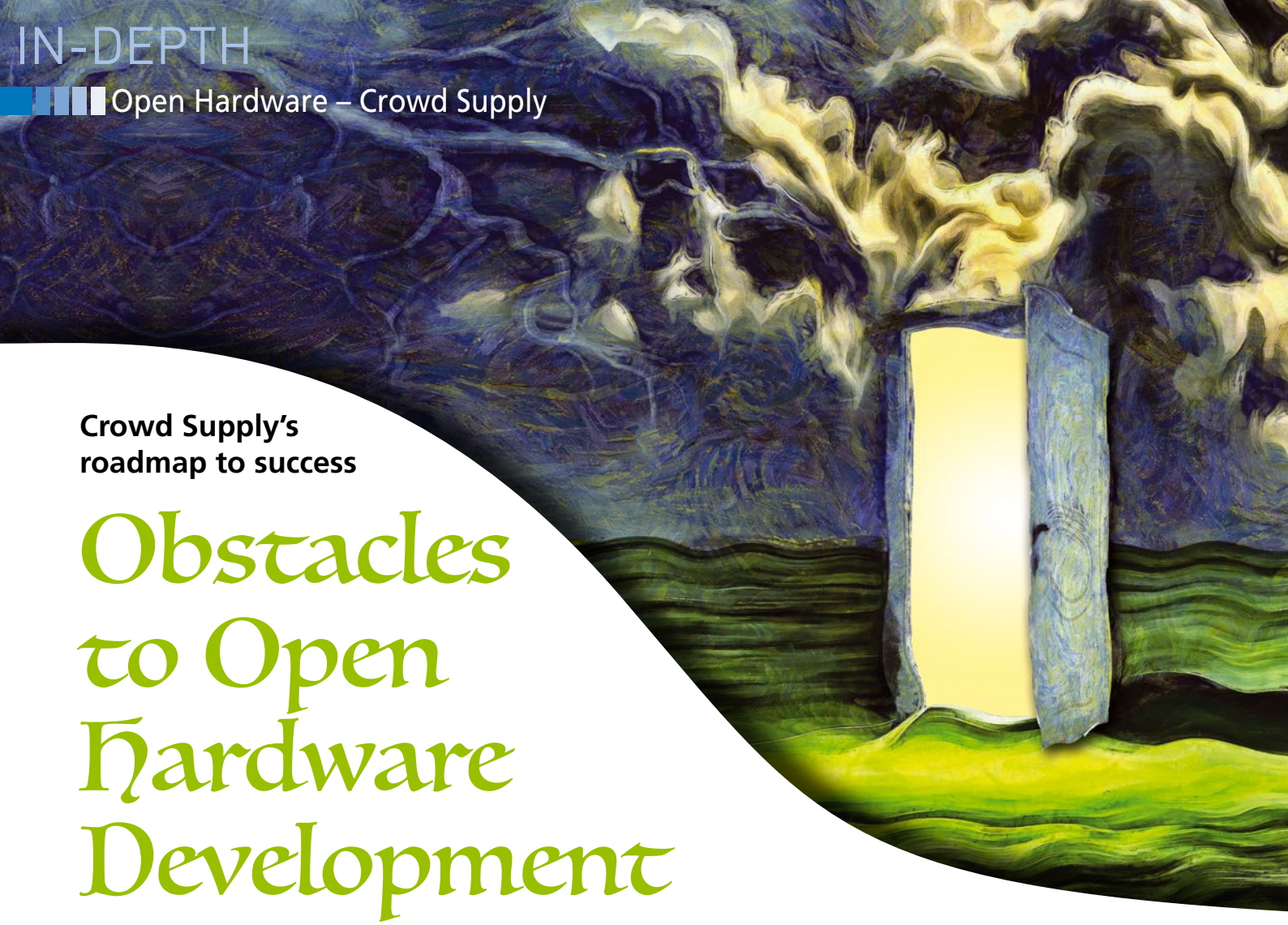
[2] Listings for this article: *ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/204/*

[3] AlphaGo: *https://deepmind.com/research/alphago/*

[4] Joshi, Prateek. *Artificial Intelligence with Python*. Packt Publishing, 2017

[5] "Programming Snapshot – Mileage AI" by Mike Schilli, *Linux Magazine*, issue 203, October 2017, pg. 56: *http://www.linux-magazine.com/Issues/2017/203/Programming-Snapshot-Mileage-AI*

[6] "Programming Snapshot – Driving Data" by Mike Schilli, *Linux Magazine*, issue 202, September 2017, pg. 50, *http://www.linux-magazine.com/Issues/2017/202/Programming-Snapshot-Driving-Data*



## Crowd Supply's roadmap to success

# Obstacles to Open Hardware Development

**Crowdfunding open hardware makes it easier to realize your dreams, but the road to success has obstacles.** *By Bruce Byfield*

Thinking of developing an open hardware product? Thanks to crowdfunding, your ambition is now within reach. You may not found the new Red Hat, but you can realistically hope to be successful at making a living doing something you are passionate about. However, between your dream and its realization are numerous obstacles, as Joshua Lifton, the founder of Crowd Supply, points out (Figure 1).

Crowd Supply [1] is a crowdfunding site like Kickstarter [2] and Indiegogo [3]. The difference is that Crowd Supply is not simply a place to raise funds. Specializing only in physical products, Crowd Supply advises its clients on business plans, offers fulfillment services and marketing, and acts as a reseller for released products (Figure 2). Open hardware is one of the site's top-level categories [4], and an open hardware credo is offered on the About Us page [5]. Lifton's boast is that, with this support, all campaigns on Crowd Supply that reach their goals deliver a product.

In the five years of Crowd Supply's existence, Lifton has become a close ob-

## BRUCE BYFIELD

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art. You can read more of his work at *http://brucebyfield.wordpress.com*

Lead Image © Bruce Rolff, 123RF.com

**Figure 1:** Joshua Lifton, the founder of Crowd Supply.

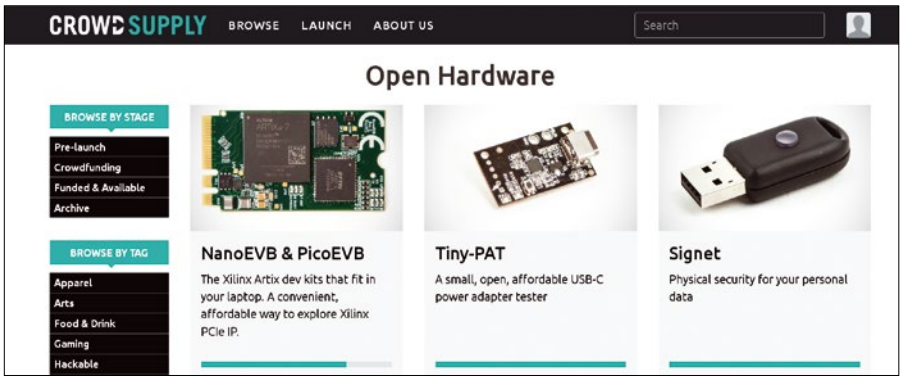



**Figure 2:** Crowd Supply's Open Hardware page.

server of open hardware and is optimistic about its future. "The open hardware market has picked up a lot of steam in the last few years," Lifton says. "I think the best analogy I've heard compares today's open hardware market with the open software market of the 1990s – no one quite knows what's going to happen, but a lot of exciting things are happening and the future is unwritten."

Lifton sees two main reasons for the growing popularity of open hardware. First, he cites Bunnie Huang's keynote at the RISC-V workshop [6] to the effect that Moore's Law – the tendency of computer chips to double their capacity every two years or so [7] – is slowing down after years of being a rough truism. Consequently, "developers have more time to work on and create open hardware before the underlying chipsets and technologies become obsolete." Since open hardware is often modestly funded and tends to develop at a slower pace than proprietary products, this change means that open hardware "actually has a chance at competing against proprietary, well-funded products" for the first time."

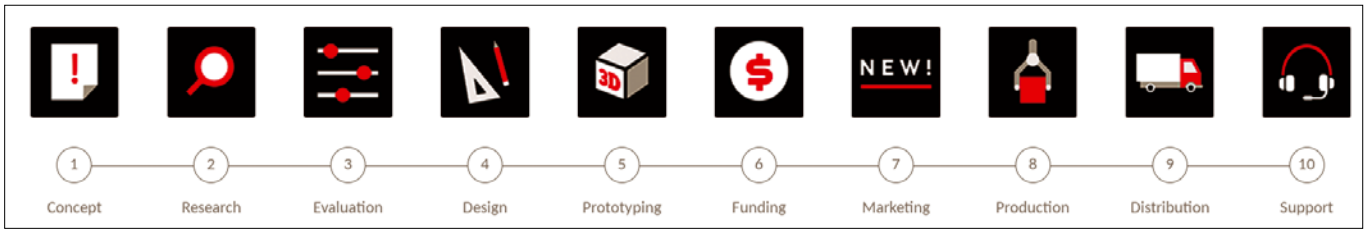



**Figure 3:** Digi-Key's Maker.io diagrams the steps to bring an open hardware product to market [9].

Second, security and privacy are becoming a consumer concern, and no longer evoke jokes about "tinfoil hats." Today, there are notes of agreement and shared anecdotes of missteps and overreach by governments and corporations. Most people still don't know what to do about it, or are perhaps unwilling to sacrifice the conveniences afforded by insecure, prying technologies, but the tide is turning. There are only so many times consumers can find out about a debacle like Lenovo's Superfish malware [8] before they take their trust and money elsewhere. "What all of this means is that there is both the competitive advantage and demand needed to fertilize a healthy market for open hardware, which promises to address these growing concerns."

## Avoiding Failure

"Crowdfunding and open hardware go hand in hand," Lifton adds, "but there is nothing magical about either – they are simply modern versions of the millennial-old art of selling a product to people who need it." Realistically, many – even most – open hardware products are likely to fail, not because there is anything wrong with the products or with open hardware, but because most new products fail. "Open hardware fails for the same three reasons any product fails," Lifton observes: lack of demand, failure to reach the right audience, and poor execution.

Like any product, open hardware development needs to follow the steps summarized on Digi-Key's Maker.io site (Figure 3): concept, research, evaluation, design, prototyping, funding, marketing, production, distribution, and support [9]. Omitting one of these steps, doing them in the wrong order, or skimping on one can be enough to sabotage your efforts. Moreover, since many open hardware developers are either amateurs or experts in only some of these areas – often, just research and design – the chances of under-emphasizing some of these steps is often high.

For example, Lifton says, "Assuming you get to the point of actually launching the product as a crowdfunding campaign, there's the very public success or failure of meeting your funding goal or not. However, even not meeting your funding goal is a type of success – you learned that the product you offered isn't the right one. If that's the case, the real failure would be to ignore that data point and continue to pour money and time into the project without changing anything else."

Conversely, "If you do meet your funding goal, there's the question of whether you actually raised the right amount of money. If your pricing and cost structure isn't right, you might actually lose money on every sale, so the more you raise, the worse off you'll be."

Yet even a successful fundraising campaign does not automatically mean that the product will be delivered. "It's not that things might go wrong," Lifton observes, "[But that] they definitely will go wrong and you've got to plan for that." The Keyboardio blog [10] and the EOMA68 campaign updates [11] are harrowing reminders of how much can go wrong, especially when dealing with manufacturers for the first time.

Moreover, even delivering a product may not be enough. "Did you deliver it on time?" Lifton asks. "Is the product so late that it's already obsolete? Now that customers finally have the product on their hands, do they actually like it? Do they like it enough to merit a second version? Do other people use your product as the inspiration or foundation for other great products?" For example, the Novena Open Laptop [12] became the basis for several projects, including the CrypTech hardware security module [13].

However, regardless of the actual problem, Lifton identifies seven "tell-tale signs that a project might fail." Briefly, they are:

- Hubris of the team
- Unclear or unknown pricing and cost structure
- Too broad of a target market
- Disregard for the competition
- Lack of technical knowledge
- Optimism about timelines
- Greed

To help clients avoid these problems, Crowd Supply has posted a guide for clients at each step of the way, ranging from high-level planning, such as how to decide whether to work with off-shore manufacturers, to the routine tactics for working with other businesses [14].

Asked how to avoid the usual obstacles, Lifton's first advice is do the opposite of all his tell-tale signs. He warns, too, that "the people working on the project should be the people who came up with the idea." He adds, "Ideas are a dime a dozen, so execution is everything. There is no wildly successful product that had never been thought of before. The difference is simply the execution."

## The Unfolding Roadmap

A significant number of open hardware projects has only existed for a few years, so the means to success is still being worked out. There are simply too few data points to create a definitive roadmap, and too many differences from conventional hardware manufacturing. However, the gaps are starting to be filled in by sites like Crowd Supply.

Following the advice of Lifton – or anyone else – is not a guarantee of success. However, entrepreneurship of any sort is a calculated gamble, and open hardware production is no exception. Resources like Crowd Supply are starting to become available, and, used carefully, they might just be enough to tip the odds in your project's favor. ■■■

## INFO

[1] Crowd Supply: *https://www.crowdsupply.com*
[2] Kickstarter: *https://www.kickstarter.com/*
[3] Indiegogo: *https://www.indiegogo.com*
[4] Open Hardware: *https://www.crowdsupply.com/open-hardware*
[5] Crowd Supply About page: *https://www.crowdsupply.com/about*
[6] RISC-V workshops: *https://riscv.org*
[7] Moore's Law: *https://en.wikipedia.org/wiki/Moore%27s_law*
[8] Levono Superfish malware: *https://www.cnet.com/news/superfish-torments-lenovo-owners-with-more-than-adware/*
[9] Digi-Key's Maker.io site: *https://www.digikey.com/en/maker/*
[10] Keyboardio blog: *http://blog.keyboard.io/*
[11] EOMA86 updates: *https://www.crowdsupply.com/eoma68/micro-desktop/updates*
[12] Novena Open Laptop: *https://www.crowdsupply.com/sutajio-kosagi/novena*
[13] CrypTech hardware security module: *https://www.crowdsupply.com/cryptech/open-hardware-security-module*
[14] Crowd Supply Guide: *https://www.crowdsupply.com/guide*

diff and merge

# Repurposing Old Tools

**Diff and merge: They're not just for developers.** *By Bruce Byfield*

Recently, a friend of mine returned to a manuscript after several months. The manuscript had half a dozen versions, and she could no longer remember how each one differed. Listening to her problem, I had a blinding flash of the obvious: `diff` [1], and related commands like `diff3` [2] and `merge` [3], can be as much help to her as they have been to coders over the decades.

`diff` is a utility that compares two files line by line. For coders, `diff` is a command that defines Unix-like operating systems like Linux. Although file comparison utilities are as old as Unix, `diff` itself was first released in 1974 for text files, with support for binary files added later. `diff` presents users with a summary of the comparison in two different formats, which can also be merged into a single file. `diff3` [2], a similar utility, operates in a like manner on three files, although it does not support binary formats. More sophisticated tools like `patch` have been developed, but `diff` is still installed by default in many distributions, and its output files, or diffs, remain a standard name for any patch, just as the `grep` command has given its name to any file search.

## Basic Comparisons

Typing `info diff` (the man page is incomplete) quickly shows how `diff` can be as useful to a writer as a programmer. The command follows the standard format of a command followed by options and two files. The first file is the original, or any file if, as in my friend's case, the original is unknown or irrelevant:

```
diff OPTIONS ORIGINAL-FILE OTHER-FILE
```

Just by adding the `--brief` (`-q`) option, a writer can tell if the files are different –

```
bb@nanday:~$ diff -q original.txt revision1.txt
Files original.txt and revision1.txt differ
bb@nanday:~$ diff -s original.txt original1.txt
Files original.txt and original1.txt are identical
```

**Figure 1:** The quickest way to use `diff` is to check whether two files differ or are the same.

**BRUCE BYFIELD**

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art. You can read more of his work at *http://brucebyfield.wordpress.com*

```
bb@nanday:~$ diff original.txt revision1.txt
1c1
< Diffs and merges: They're not just for developers
---
> Diffs and merges:
3c3,6
< Recently, a friend of mine returned to a manuscript after several months. The manusc
ript had half a dozen versions, and she could no longer remember how each one differed
. Listening to her problem, I had a blinding flash of the obvious: diff and related co
mmands like diff3 and merge can be as much help to her as they have been to coders ove
r the decades.
\ No newline at end of file
---
> Recently, a friend of mine returned to a manuscript after several months. The manusc
ript had half a dozen versions, and she could no longer remember the differences betwe
en them all. Listening to her problem, I had a blinding flash of the obvious: diff and
 related commands like diff3 and merge can be as much help to her as they have been to
 coders over the decades.
```

**Figure 2:** `diff` **output consists of a comparison summary, plus context lines around where differences occur.**

something that file attributes alone cannot always show. Similarly, `--report-identical-files` (`-s`) either reports when the files are the same or displays the differences (Figure 1). In some situations, like my friend's, this information alone may be enough to let some files be ignored.

Even more efficiently, directories can be specified instead of files, with `--recursive` (`-r`) added to include subdirectories in order to locate identical files. In the same way, the `--from-file=DIRECTORY1` and `--to-file=DIRECTORY2` options can be used to compare files of the same name in different directories. With `--exclude=PATTERN` (`-x`), files that match the pattern are excluded, while `--exclude-from=FILE` (`-X`) excludes files that match the patterns that are listed, one per line, in the designated file. Still other options when comparing directories are the self-explanatory `--starting-file=FILE` (`-S FILE`), `--exclude=PATTERN` (`-x PATTERN`), `--ignore-file-name-case`, and `--no-ignore-file-name-case`. All these options make for a more targeted search, and, although they take a while to set up, are still much faster than opening all the files for comparisons.

However, the comparison can be far more specific. Some options, such as `--show-c-function` (`-p`) are specific to programming, but others apply to regular text as easily as code. You can, for example, use `--ignore-all-space` (`-w`) so that differences in white space are not considered. Similarly, when comparing plain text files, using `--ignore-blank-lines` (`-B`) ignores the blank lines that are being used to separate paragraphs. A particularly useful option is `--ignore-matching-lines=REGULAR-EXPRESSION`, which can help to focus results.

More specifically, experts can specify what to display with `-GTYPE-group-format=`. The option can be completed to specify, in this order, lines from the original file (`NUMBER <`), lines from the second file (`NUMBER >`), or lines common to both (`NUMBER=`). Similarly, `--LTYPE-line-format=` can be completed by the first line number (`F=`), last line number (`L=`), and the number of lines (`N=`). Both options have a number of other completions, so consult the `man` or `info` page for more details.

## Output Formats

By default, `diff` displays the lines where differences occur in a set format (Figure 2). If the files are identical, there will be no output whatsoever. However, assuming some output is produced, at the top of the display is a summary, such as *5,6c7*. This summary displays the line number or lines where differences occur in the original file on the left, and the line number in the other file on the right. In between is one of three letters: *c* (change), *a* (append), or *d* (delete). Below the summary, the name of the original file is given first, marked by a lesser than (`<`) sign. Below it, the second file is marked by a greater than sign (`>`). For each difference, context lines are given to make the difference easier to find. The default number of context lines is three, but you change them by adding the option `--context=NUMBER` (`-c` or `-C NUMBER`).

An even easier output display can be had by adding `--side-by-side` (`-y`) to the command. This option displays the original file's contents on the left and the second file on the right, making detailed comparisons easy (Figure 3). You can adjust the column widths for a side-by-side display up to a maximum of 130 characters with `--width=NUMBER` (`-W NUMBER`). Another option is to set `--left-column`, so that only common lines are shown.

Regardless of which of these two output formats you use, the display is noticeably more flexible than that offered by LibreOffice's *Edit* | *Track Changes*,

```
bb@nanday:~$ diff --side-by-side original.txt revision1.txt
Diffs and merges: They're not just for developers             | Diffs and merges:

Recently, a friend of mine returned to a manuscript after sev \ Recently, a friend of
mine returned to a manuscript after sev
                                                              >
                                                              >
                                                              > Diff is a utility that
 compares two files line by line. For cbb@nanday:~$
```

**Figure 3:** **One of the easiest ways to use** `diff` **is to display output in two columns.**

```
bb@nanday:~$ diff --ifdef=NAME original.txt revision1.txt
#ifndef NAME
Diffs and merges: They're not just for developers
#else /* NAME */
Diffs and merges:
#endif /* NAME */

#ifndef NAME
Recently, a friend of mine returned to a manuscript after several months. The manuscri
pt had half a dozen versions, and she could no longer remember how each one differed.
Listening to her problem, I had a blinding flash of the obvious: diff and related comm
ands like diff3 and merge can be as much help to her as they have been to coders over
the decades.
#else /* NAME */
```

**Figure 4:** `diff` **can also create a merged output file.**

which can require far more concentration to read. If you open a second copy of the original, you can merge the files manually as you compare `diff`'s results. A manual comparison is laborious, but it may be the best way to compare results.

A third alternative is to to use `--ifdef=NAME` (`-D NAME`) to create an output merged file (Figure 4). This output can be copied and pasted into a new file, where a writer can manually merge. However, if you are confident that the two files can be merged to get the results that you want, you can use `--ed` (`-e`) to actually merge the file. In programming, `--ed` is used to generate a patch, yet it can serve a writer's purpose just as well.

In all formats, you can further customize by adding `--color` (Figure 5). Left unspecified, the `--color` option will use color when standard output is to a terminal. However, you can also complete the option with `=none` to never use color or `=always`. By default, red is used for deleted lines, green for added lines, cyan for line numbers, and a bold font weight for the header. Colors can be customized with `--palette=PALETTE`, as specified in the `diff` info file.

## diff3 and merge

`diff`'s obvious limitation is that the original file must be compared against each of the other files one at a time. A quicker method is to use `diff3` or `merge` to compare two files simultaneously with the original.

Like `diff`, the first file listed by `diff3` is the original. The default output of the two commands is also similar, although `diff3` uses a back slash (\) for the original file, a lesser than sign (<) for the second, and a greater than sign (>) for the third. In addition, `diff3` can add `--show-all` (`-A`) to output all changes, with conflicts listed in brackets. `diff3`'s output can also be set to show only overlaps with `--show-overlap` (`-E`) or non-overlaps with `--easy-only` (`-3v`). Other options for input include `--ed` (`-ed`), which `diff3` shares with `diff`, and `--merge` (`-m`), `diff3`'s more sensibly named version of `diff`'s `--ifdef=NAME` (`-D NAME`).

`merge` is a near-duplicate of `diff3`. However, instead of providing output that can be copied and pasted into a new file, `merge` adds everything to the original file. This behavior is not a problem if there are no conflicts. However, if conflicts do exist, `merge` warns of them, and the original file will need editing. This extra effort is not much trouble in a plain text file, but in a binary format like Open Document Format, it could potentially corrupt the original file. The same is true for `-A`, which, as in `diff3`, offers more verbose output. For this reason, only use `merge` after making a backup of the original.

## Repurposing Tools

`diff`, `diff3`, and `merge` are not the only programming tools that can be borrowed by writers. For example, multiple text files, including HTML, can be edited all at once by placing them in the same directory and using `sed`. I can't help thinking, as well, that my friend would have had less trouble sorting her files in the first place if she had been using a form of version control like Git; in fact, browsing GitHub shows that online help and manuals are already being stored there.

In the case of `diff`, `diff3`, and `merge`, the tools are even friendlier than they are for development, because perhaps a quarter of the options can be ignored. The major barrier for using them is that they require a different workflow than most writers have. Yet once the idea that these tools save time is understood, they can be as useful to writers as they are to programmers. ■■■

## INFO

[1] diff: *https://linux.die.net/man/1/diff*
[2] diff3: *https://linux.die.net/man/1/diff3*
[3] merge: *https://linux.die.net/man/1/merge*

```
bb@nanday:~$ diff --color=always original.txt revision1.txt
1c1
< Diffs and merges: They're not just for developers
---
> Diffs and merges:
3c3,6
< Recently, a friend of mine returned to a manuscript after several months. The manusc
ript had half a dozen versions, and she could no longer remember how each one differed
. Listening to her problem, I had a blinding flash of the obvious: diff and related co
mmands like diff3 and merge can be as much help to her as they have been to coders ove
r the decades.
\ No newline at end of file
```

**Figure 5:** **Adding color can make** `diff` **output easier to read.**

## Mozilla adds multiprocessing with Electrolysis in Firefox 54

# FOUR CYLINDERS

**Developers are praising Firefox 54 as the "best Firefox ever." The revamped web browser adds multiprocessing and promises a significant boost in speed.** *By Christoph Langner*

In the mid-1990s, Netscape Navigator was the leading web browser, but when Microsoft decided to bundle Internet Explorer (IE) with the Windows operating system, the tide began to turn. After years of competition (and litigation), Redmond lost the antitrust case, but won the browser war. At its peak, IE had an 80 percent share of the browser market. But the Mozilla project, which was built from remnants of Netscape, bounced back with Firefox. The Firefox browser has had a long and glorious run, attracting up to 48 percent of the market at its peak, including many Windows users who opted for Firefox because they just liked it better than IE [1].

Firefox's fortunes have waned in recent years. Chrome has emerged as the leading third-party browser alternative, with Firefox market share falling somewhere in the range of 11 to 14 percent. Part of the change is the ordinary dynamics of the software development lifecycle. An application is designed for the time in which it is built, with state-of-the-art programming techniques and implicit assumptions about the state of available hardware. As the world changes, the software continues to receive updates, but a large-scale overhaul is never easy midstream, especially for an application with millions of users.

Chrome was built later and isn't showing the same signs of age. IE is definitely aging, but Microsoft solved the problem by replacing IE with a whole new browser – the Microsoft Edge browser, which debuted with Windows 10.

Chrome and Edge distribute the computing work when rendering a web page over multiple processes. On a modern system, this approach should result in increased performance, stability, and security. The Mozilla developers knew they needed this multiprocessing capability to stay relevant for the next generation. The result of this effort is the Electrolysis project [2], also called Multiprocess Firefox [3]. At least one of the developers involved with the Electrolysis project has called the newly released Firefox 54 [4] with Electrolysis support "the best Firefox ever" [5].

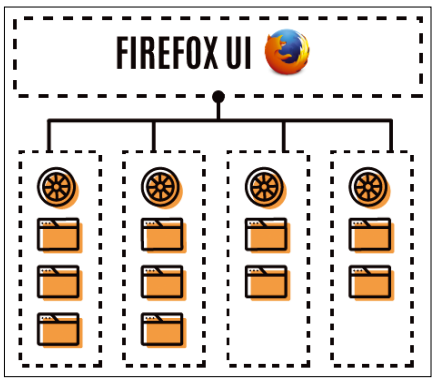



**Figure 1: Firefox 54 with Electrolysis transfers the interface and web content into their own processes. (CC BY-SA 3.0 [6], source: Dan Callahan, Belén Albeza)**

## Multiprocess Firefox

With Firefox e10s (the abbreviation for Electrolysis) each tab and the interface need to run in a separate process (Figure 1). This promises faster load times, as well as greater security and stability. All tabs work independently: If a corrupted web page crashes, only the corresponding process and the associated tab will pend. All other web pages and the browser itself continue to operate unhindered.

With Firefox 54, Mozilla automatically activates Electrolysis for the user for the first time – but only for those not using add-ons. Previously, this was only possible with the developer's "nightly" version. As a user, you will not initially be aware of the change. To start, the new Firefox version looks and feels the same as the previous browser editions.

To see if Electrolysis is ac-

Lead Image © Karols Stefan Sobolewski, 123RF.com

tive, go to the *Troubleshooting Information* page (click the hamburger menu, then *?* | *Troubleshooting Information*) and enter *about:support* in the address bar. If the field that follows *Multiprocess Windows* shows *1/1 (Enabled by default)*, Firefox uses several processes (Figure 2). The *1/1* stands for the number of Firefox windows currently open.

Firefox logically separates the individual tabs from one another with Electrolysis plugins, so a number of popular extensions block the function. In the past, this included NoScript [7], for example. Recently, most add-on developers have responded and updated their applications accordingly. However, various browser add-ons in the Mozilla extensions shop still have not fixed this problem. If you use one of these add-ons, the corresponding field in *about:config* will only show *0/1 (disabled by add-ons)*.

## Incompatibilities

In such cases, you need to disable the incompatible add-on. The easiest way to find the culprit is via the Add-on Compatibility Reporter [8], which adds an additional column in the extension overview under *about:addons*. Beside each entry is *Compatible with multiprocess* or *Not compatible with multiprocess*.

Ubuntu's automatically preinstalled Ubuntu Modifications (Figure 3) is a typical example of an incompatible add-on. To activate e10s in Ubuntu 17.04, you first need to disable this add-on or force-activate Multiprocess Firefox through configuration (see the "On Command" box).

However, Firefox 54 still does not necessarily use multiple processes. You need to change the *about:config dom.ipc.processCount* option in the configuration database. A *1* is usual, and any value greater than *1* stands for another Firefox process that the browser launches as required. Mozilla recommends a maximum value of *4* because the memory consumption increases with each process – too many processes hardly offers any more advantages (Figure 4). The System Monitor shows the individual processes (Figure 5). Alternatively,

```
pgrep -c -f firefox
```

provides the number of processes launched by Firefox (plus one for the interface).

## Always Responsive

However, the performance values between Firefox 53 and Firefox 54 only changed marginally in a test over several benchmarks (see Table 1). The difference is rather striking in practice: When I opened Firefox 53 without e10s (see the "Switched Off" box) and launched the JetStream [10] JavaScript benchmark in two browser windows, it was no longer possible to run Firefox on the test computer. Other web pages would no longer load, and the interface froze.

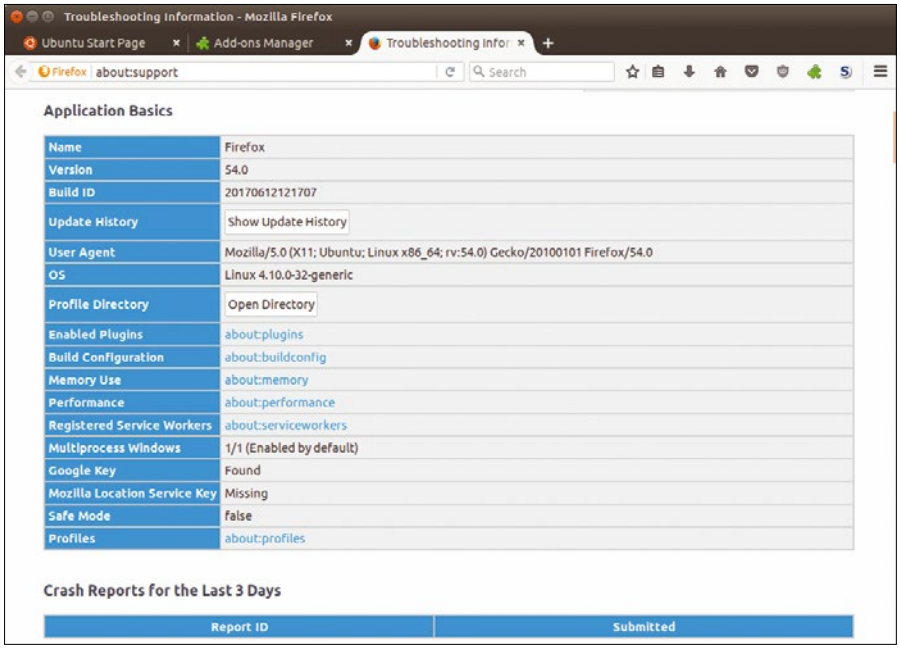



Figure 2: The output for *Multiprocess Windows* under *about:support* shows that Electrolysis is active.

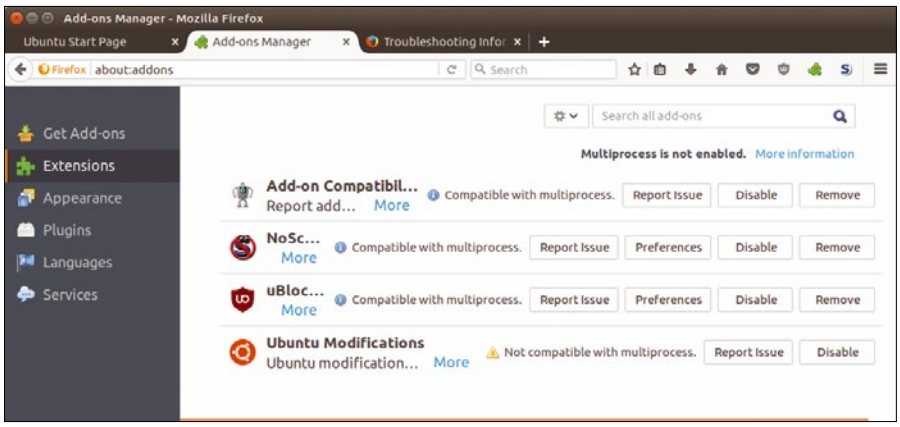



Figure 3: Ubuntu users need to disable the Ubuntu Modifications add-on by hand for Multiprocess Firefox to work.

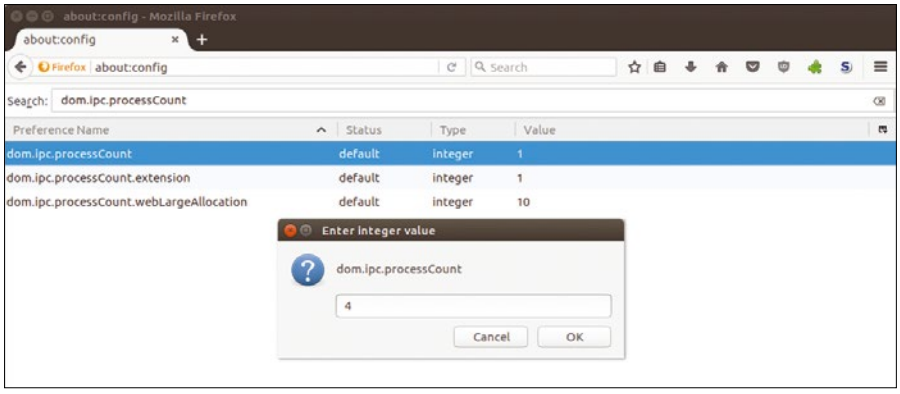



Figure 4: The number of processes launched by Firefox can be adjusted in the settings.

## ON COMMAND

Traditional add-ons are on the way out at Mozilla – Firefox developers are planning to switch completely to WebExtensions [9] with version 57. Using this browser API makes it possible to develop add-ons with relatively little overhead for all popular browsers (Firefox, Chrome, Edge). Classic extensions often can't cope with Multiprocess Firefox; therefore, browsers don't activate this function if they encounter an incompatible extension. You are then faced with the choice of continuing to use a beloved add-on or Multiprocess Firefox. As a way out, you can force e10s active in *about:config*. There, you can right-click and choose *New | Boolean*, name it *browser.tabs.remote.force-enable*, and select *true* as the value. A reboot then forcibly enables Multiprocess Firefox. In case of problems, you should do without the extension in question – or even e10s.

However, with Firefox 54 and Electrolysis activated, it was possible to run the browser without impairments, even with two or more benchmarks running. In this case, the results of the performance measurements might have been slightly poorer than those of Firefox without e10s, but in practice, I can deal with this in favor of an interface that is easy to use. In everyday testing, the new fluidity particularly stands out with power-hungry websites such as Firefox or Google+.

Firefox keeps limits memory consumption. Measurements by Mozilla developers showed that Firefox in Linux hogs less memory than Chrome [11]. The same applies to Firefox in Mac OS X and Windows. The current Firefox is therefore only topped by IE. However, Microsoft's current Edge browser uses significantly more memory. Firefox is therefore a possible alternative to Chrome, particularly on systems with little memory.

## Conclusions

According to software developer Ronan Cremin, an average website's volume has more than tripled over the past six years [12]. Accessing such a website today loads a volume of data comparable to downloading the compressed setup of the computer game *Doom*. The new Multiprocess Firefox 54 is now finally adapting to this development. Anyone who works with tabs a lot and jumps back and forth between Firefox, Chrome, and other browsers will certainly notice the improvements.

However, a lot still needs to be done: In this test, Electrolysis occasionally crashed browser tabs, and numerous incompatible extensions [13] still exist, so the "best Firefox ever" still has room to improve – especially considering how Phoenix and Firebird (Firefox has had to change its name several times) released users from IE's clutches. But in any case, this is the right path. ■■■

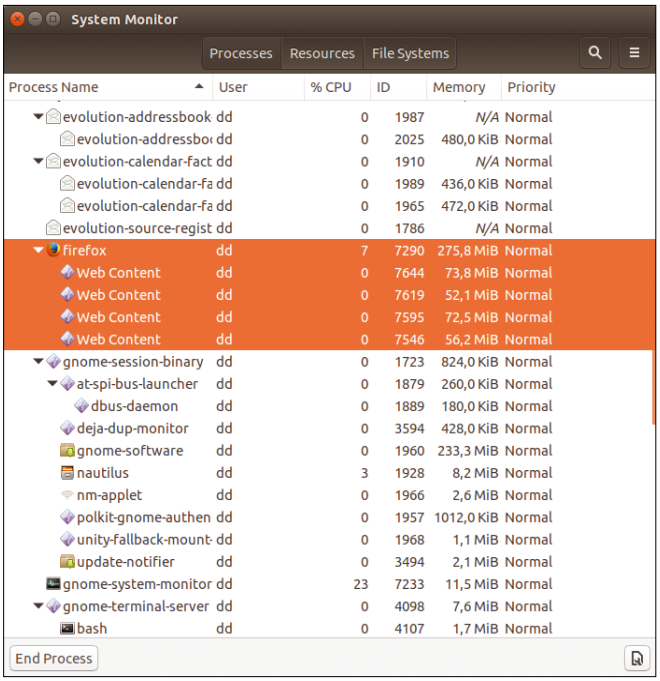



**Figure 5:** The System Monitor shows Firefox 54 distributing the rendering of web content (*Web Content*) to multiple processes.

## TABLE 1: Performance Comparison

| Benchmark | Firefox 53 | Firefox 54 | Chrome 59 | Opera 45.0 |
|---|---|---|---|---|
| JetStream 1.1 | 142.7 | 141.0 | 315.8 | 223.1 |
| SunSpider 1.0.21 | 297.7 | 259.3 | 315.8 | 223.1 |
| Peacekeeper [1] | 4447.3 | 5764.7 | 4385.7 | 3853 |

Higher values are better.
[1] Benchmark is no longer maintained.

## SWITCHED OFF

During the test, Multiprocess Firefox proved to be subjectively less stable than the conventional Firefox. Crashes often occurred when loading several web pages that used JavaScript extensively (e.g., the browser benchmarks). Thanks to e10s, crashes don't affect the whole browser, but they often cause so much interruption that you'll want to disable Electrolysis. To do so, open *about:config* and set the *browser.tabs.remote.autostart* option from *true* or *false*. Firefox only uses one process after a reboot.

## INFO

[1] Browser statistics: *https://www.w3schools.com/browsers/*
[2] Electrolysis: *https://wiki.mozilla.org/Electrolysis*
[3] Multiprocess Firefox: *https://developer.mozilla.org/de/Firefox/Multiprocess_Firefox*
[4] Firefox 54.0 release notes: *https://www.mozilla.org/en-US/firefox/54.0/releasenotes*
[5] "The Best Firefox Ever": *https://blog.mozilla.org/blog/2017/06/13/faster-better-firefox/*
[6] Attribution-ShareAlike 3.0 Unported: *https://creativecommons.org/licenses/by-sa/3.0/legalcode*
[7] "NoScript add-on does not work with e10s": *https://bugzilla.mozilla.org/show_bug.cgi?id=1058542*
[8] Add-on Compatibility Reporter: *https://addons.mozilla.org/en-US/firefox/addon/add-on-compatibility-reporter/*
[9] WebExtensions: *https://developer.mozilla.org/en-US/Add-ons/WebExtensions*
[10] JetStream: *http://browserbench.org/JetStream*
[11] "Firefox memory usage with multiple content processes": *http://www.erahm.org/2017/05/15/firefox-memory-usage-with-multiple-content-processes/*
[12] "The web is Doom": *https://mobiforge.com/research-analysis/the-web-is-doom*
[13] "Are we e10s yet?": *https://www.arewee10syet.com*

Ben Everard

**It's become trite** to say that children are the future, but that doesn't make it any less true. The people learning how to use computers today are the ones that will be designing and building the software of tomorrow. It is, therefore, incredibly important to get open source software that's not just available for kids, but exciting and relevant for them. This month, Mike takes a look at one project hoping to be just this: Minetest, an open source project inspired by the hugely popular Minecraft game.

For those readers at, ahem, the other end of the age spectrum, we've also investigated one of the oldest programming languages around, COBOL. Mike takes a look at this language that steadfastly refuses to fade away.

In between these two, we've managed to wedge a whole lot of Free Software goodness covering everything from the cloud (thanks to Valentine Sinitsyn) to my article on increasing the power of your command line. Graham Morrison, as always, covers the very latest software releases both well known and more niche. We've also got our usual trio of talking heads discussing the most pertinent topics in Open Source.

The one thing that remains is for me to say goodbye. This is my last issue writing for *Linux Magazine*. I'm off to take on new challenges – not least the fact that I'm about to become a father. I've steadfastly refused to count how many issues of various Linux magazines I've worked on for fear of feeling too old, but I'd like to thank everyone who joined me on my journey into Linux.

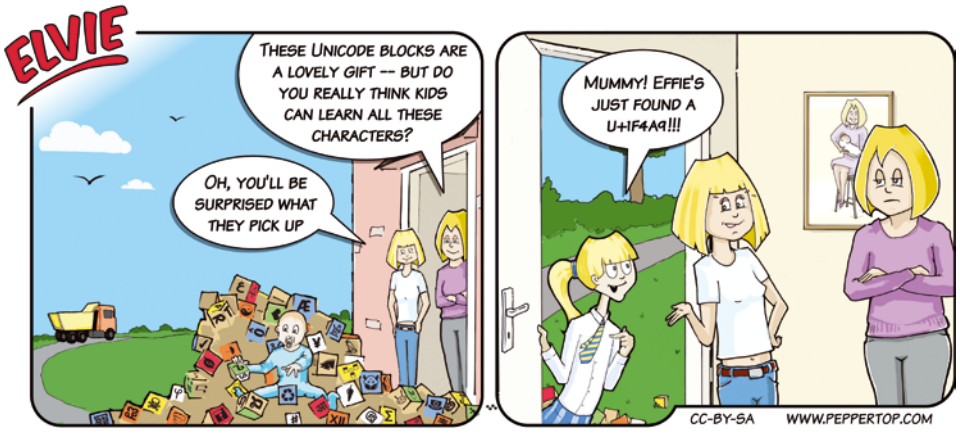
Now, that's enough sentimentality for one issue. Let's crack on with the Free Software.

*– Ben Everard*

Andrew Gregory

Graham Morrison

Mike Saunders




## LINUXVOICE

# NEWSANALYSIS

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

Opinion

## Can PatentLeft Save Us?

### A hack on patent law, based on copyleft, could render patents useless. BY SIMON PHIPPS

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

The word "copyleft" arises from a clever hack by Richard Stallman, who used the laws relating to copyright – a statutory device to incent creativity by granting limited monopolies to creators – to create a world where creators are incented to share instead of monopolize their work.

Since the Berne Convention, a creative work is the automatic sole property of its creator, and the only way others can use it in any way until the monopoly expires is with the express permission of the creator of the work, who is said to hold the copyright. Copyleft grants everyone receiving the work an unlimited license to use, improve, and share it, but only on the condition they grant the same conditional rights to every recipient. Copyleft thus makes more and more works freely usable as more and more people improve them.

Could we do the same thing to subvert patent law? It seems that's at least part of the motivation behind the use of a controversial combination of the BSD open source copyright license and a broad patent grant by Facebook. A few years ago, they silently standardized on releasing all their open source projects – including popular codebases like the RocksDB storage engine and the React.js user interface framework – under the venerable three-clause BSD license supplemented by a unilateral grant to any of Facebook's patents necessary to use the software.

While that initially sounds uncontroversial and generous, the patent grant also included an aggressive termination clause that removed any grant of patent rights in the event of patent litigation. That too may sound unremarkable – after all, the Apache and Mozilla licenses include termination provisions, as do most other modern open source licenses – but Facebook's version goes much further than any other.

The termination clause in the Apache license also removes patent grants in the event a company initiates patent litigation against the project, but the trigger for termination is limited to the code at hand – you have to sue an Apache contributor over the Apache project, and the patent grants you lose are those related to the project. Losing a license would mean the company you are attacking could then countersue for the infringement of the patents that were previously licensed to you. That sounds like a fair deal.

By contrast, Facebook's termination clause is triggered by patent action of any nature, related or unrelated to the project at hand (it originally triggered even on self-defensive patent litigation where Facebook was the aggressor, but that was removed in a revision a while back). It's also triggered not just by action against Facebook, but also action against its subsidiaries and partners, and also by action against members of the open source community working on the project at hand. It's not just you that can trigger the clause – any of your subsidiaries and agents can also make it happen.

So imagine you're a European company and your American distributor takes a patent potshot at another company who is intentionally using your distributor's patents without a license. If they happen to be a React.js community member or a Facebook partner, you will lose all your patent grants from Facebook, even though neither you nor they are involved in the action. You'll then risk patent litigation by Facebook.

Facebook defends their license combo by saying they have a need and a right to protect themselves from patents. They have little to say about the network effects of the license combo, but have clearly indicated they are an intentional byproduct. Facebook wants to use their massive presence in the market to chill use of patents in the whole market. Their competitors loudly say that's a bad thing. That's led to their employees at the Apache Software Foundation demonizing Facebook's license and banning it from use on any Apache project.

But what would happen if they were to generalize their license combo and submit it to OSI for approval as a new license? Then we could all release software that, as it spreads, makes the granting of a patent license conditional on not using patents out of a fear of some remote networked relationship in the community bringing down the fiery wrath of a giant corporation. I'd want to call such a license "PatentLeft" – a hack on patent law that renders patents useless in practice. Maybe that would be a good thing? ■■■

# The Hero We Need

BY ANDREW GREGORY

## Malwaretech stood up for us. Now we should stand up for him.

The security researcher Marcus Hutchins, known professionally as *malwaretech*, has been arrested in the US, accused of creating the Kronos banking trojan and conspiring to sell it for $3,000 on the dark web.

This seems unlikely to me, given that he had previously turned down a $10,000 reward for his work on stopping WannaCry. Yes, that same WannaCry ransomware that threatened several organizations in Europe (and would have targeted the US if Hutchins hadn't got to it first).

Ten thousand dollars is more than $3,000, so the motives don't add up for me. Hutchins may or may not have written some code, and that code may or may not have been used to commit a crime. Tech-literate people, such as the readers of *Linux Magazine*, understand the difference between creating a work and using it to commit a crime, but most of the media coverage – in the UK, at least – has been desperate to follow the paradigm of building a man up only to gleefully knock him down. Even his achievement of stopping WannaCry is decried as "accidental," a word full of self-deprecating charm when used by Hutchins, but which simply sounds malicious in the hands of the *Daily Mail* and *The Telegraph*.

I'm sure justice will prevail eventually, but in the meantime, there's a chap in a very sticky situation and a government department making itself look silly. Then there's the UK government, which, now that it's turned its back on the EU, is desperate not to upset the "special relationship" by standing up for its subjects.

I don't know whether he's done anything wrong, but I do know that there are no advocates standing up for the right to play with code. For all the purity of its intentions, the Free Software Foundation has had nothing to say about the four freedoms in this case. Any hacker at home who downloads some malware and examines the code, whether that's to look for weaknesses, to find ways to improve it, or just out of interest, is putting themselves at risk of arrest the next time they travel to the US, regardless of their intention. What's implied in this is that the government is allowed to decide who can study, learn from, copy, and share code, which is definitely not what the founding fathers of Free Software imagined.

If the tool Hutchins is alleged to have made was, for example, a gun, there would be no drama at all: the National Rifle Association is very clear that guns don't kill people; murderers kill people. Guns, the argument goes, are inanimate objects. Likewise malware created by security researchers doesn't shut down banks; it's black hats who deploy it that do the damage. And I don't know or care whether the alleged evil software was written on Linux or not. The logic of the prosecution defies the principles of Free Software, so it shouldn't matter what platform it was written for. Free Software campaigners aren't just fighting for bearded Trisquel users: We all need our representatives to step up and do better. ■■■

# Minetest

## Discover this open world, open source, Minecraft play-alike with giant maps and oodles of nifty features.

BY MIKE SAUNDERS

If you've never played Minecraft, you're missing out. Yes, the game was hyped to ridiculous levels during its glory days, and some players spent arguably an unhealthy amount of time in their fantasy worlds building enormous and highly intricate constructions, but it was still something very special. Minecraft was created on the fly using algorithms, creating a giant, procedurally generated world, in which you were plonked at a random point with no particular goals or equipment; you just had to explore and survive with what you could find.

For the first few minutes of Minecraft, you wonder what the point is. It feels empty and lonely, but then you start collecting items, crafting (combining things together to make other things), exploring the various landscapes (beaches, jungles, arctic tundras, and mountainous highlands), and finding non-player characters in villages. Before you know it, you've crafted a robust set of armor, developed some powerful weapons to protect yourself, and even built a house. Every change you make to the world is saved, so it starts to feel like a real, living and breathing place. Exploring caves full of nasty critters becomes a terrifying experience.


**Figure 1:** When you first fire up Minetest, you are prompted to create a new game world.

Now, Minecraft was (and still is) a great game, but it's not open source and free software – which is a bit of a downer for many FOSS purists. Fortunately, however, you have an alternative in the form of Minetest [1] (formerly Minetest-c55, named after the online handle of one of its lead developers). Minetest has been in development since 2010. Although its most recent version is 0.4.16, suggesting that it's still in the early stages of development, it's very much playable and loaded with cool features. On top of that, you can play with others online and even extend the game using modifications written in Lua. (For a text mode alternative, see the "Dwarf Fortress" box.)

Minecraft-like games can be a bit intimidating at first, so over the next few pages, I'll ease you into the world of Minetest, show you the most important tricks and techniques, and give you some pointers for further exploration.

### Initial Steps

If you visit the Downloads page on the Minetest website [2] and scroll down to the Linux section, you'll see packages for various Linux distros – some are more up to date than others. It's also worth checking your distro's package manager to see what's available. If you can't find anything suited to your particular flavor of Linux, you could try building the source code; instructions and links for this are provided at the bottom of the Downloads page.

Once you have the game installed and started, you'll be presented with a "Select World" screen (Figure 1). This isn't especially welcoming if you've never played a Minecraft-ish game before, but it basically means that you need to create a new environment to play in. In Minetest, you don't have a fixed world to explore that's identical to all other players – instead, you start a new one, based on a random "seed" number, which is then used to generate the rest of the world in a process known as procedural generation.

Therefore, all aspects of the world (hills, trees, lakes, etc.) are created using an algorithm as you head toward them. Using this technique, the game

doesn't need to store enormous amounts of data; instead, it only generates data when required. Additionally, if you really like the world you're in, you can save and share the short "seed" number, rather than having to transfer vast saved game files.

Clicking *New* in the dialog box lets you give your world a name (e.g., *My world*). In the Seed box, you'll see the random number mentioned above; type *1234* here, so that you're starting from the same point as the game in this article. Make sure to choose version *7* for Mapgen and *Voxelgarden* for the Game box; then, click *Create*. You'll return to the initial dialog box, with your newly created world selected – so click *Play* to start.

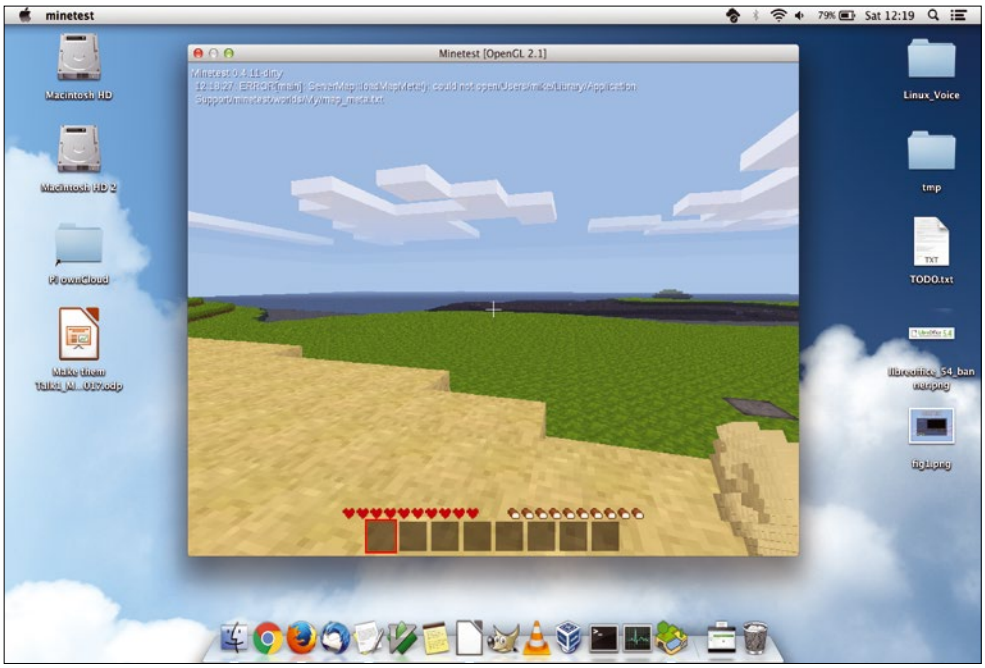



**Figure 2:** Here's where it all starts – on the edge of a desert, looking over a grassy plain.

You'll be plonked in the middle of the world (see Figure 2), with no indication of what to do next. Using the mouse, you can look around in various directions, and you'll see that the world is being built up all the time (using the aforementioned procedural generation). Note that for laptop users, a mouse is highly recommended for playing Minetest – if you don't have one, it's worth grabbing a cheap USB mouse just to experience this rather awesome game.

By moving the mouse, you can look around in your environment, but how do you move? Well, Minetest adopts the key bindings typically used in first-person shooters: WASD. Press *W* to walk forwards, *S* to walk backwards, and *A* and *D* to step left and right, respectively. By using these key bindings in conjunction with the mouse, you can now move around freely in your environment. If you hit a ledge or something else that you want to go over, tap the spacebar to jump.

Note that Minetest takes control of your mouse, so if you need to do something else in your Linux setup, hit Esc to grab the mouse back. This also brings up a menu that you can use to exit back to the main menu, exit the game completely, or change some settings. Note that you don't need to save the game explicitly– this happens automatically, and next time you run Minetest, you'll be dropped back in the same place with all the changes you have made to the environment.

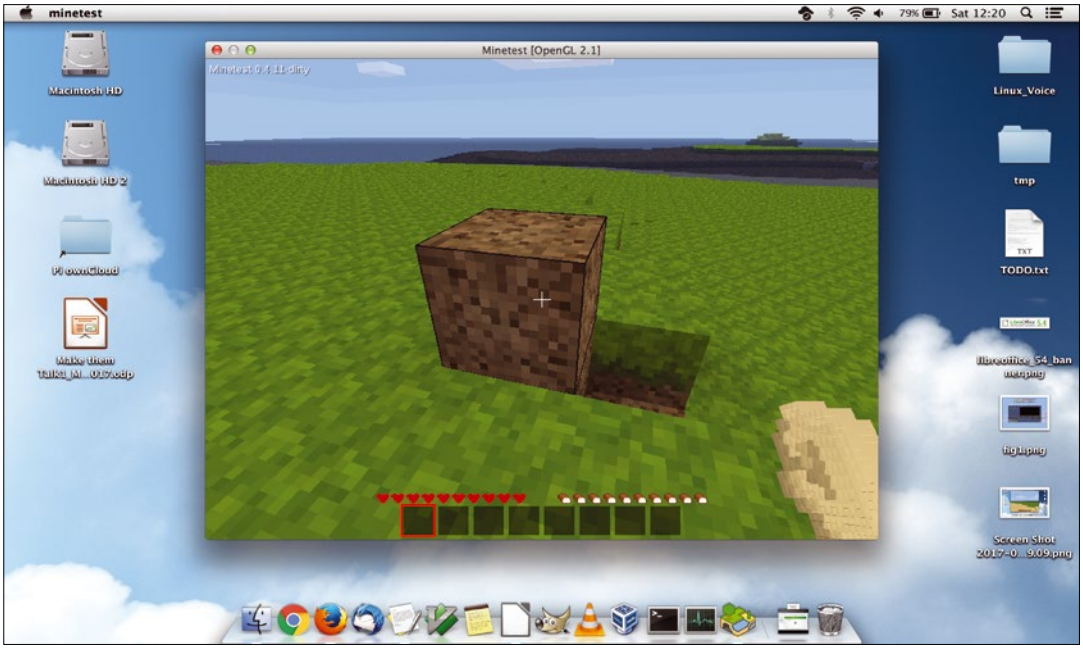



**Figure 3:** The basics of mining: digging up a block of dirt and putting it on the ground nearby.

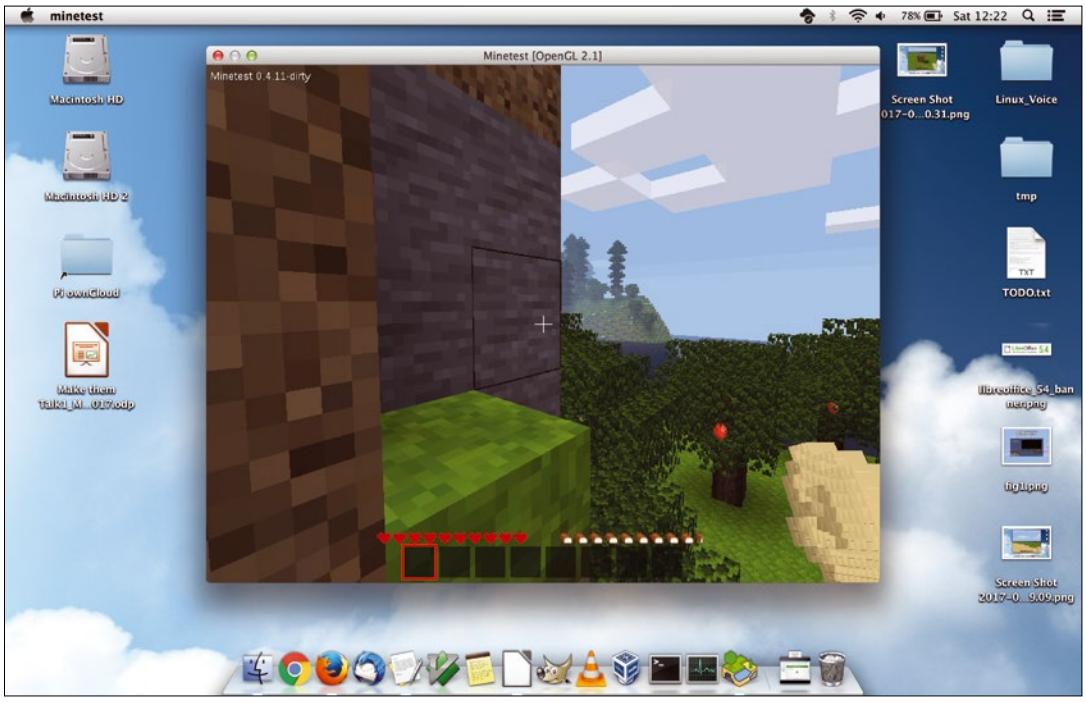



**Figure 4:** You've come across some stone, but you can't mine it with your bare hands – you need a wooden pickax!

## Build It Up

As the game's name suggests, Minetest involves a lot of "mining" – or to be more specific, digging up blocks, storing them, and using them to build other things. To see this action in the game, use the mouse to move the crosshair over a piece of land, such as desert or soil (you're on the border between a desert and some grasslands). That piece of land will have a black square around it. You have highlighted this square, so click and hold the left mouse button for a few seconds, and you'll see a digging animation. After a couple of moments, the block of dirt will be dug up and placed into your inventory (the row of icons at the bottom of the screen).

Congratulations, you've "mined" your first block! Now you can do with it what you want. Select another piece of land with the crosshair, so that a block is selected with a black outline, and this time press the right mouse button to place the block from your inventory back into the game world (see Figure 3). (If you have trouble placing blocks, try walking back a bit, but keep the selected block where you want to place your item. The game prevents you from placing blocks directly underneath you.)

Now, you can dig up many blocks of dirt from the ground, and they will be stored in your inventory, along with a number that shows how many blocks of that type are in your possession. However, dirt isn't very useful – you want better blocks in your inventory! To find other things to dig up, you need to do some exploring. Head away from the desert toward the small islands with trees. Note that if you end up in water, you can press and hold the spacebar to swim. (You can let go to dive down under the water, but pay attention to the bubbles that appear above the inventory – if those run out, you'll drown!)

While you're exploring, you might notice that things start to get very dark after a while. Minetest's worlds have day and night cycles (albeit much faster than in the real world), which makes for exciting gameplay later on, but it's a bit of a hindrance at this early stage. Fortunately, you can fix this by bringing up a console and setting the time. Type

```
/time 8000
```

to reset the time to the morning, so you can carry on exploring with ease.

Keep heading toward the larger mountains until you come across some stone blocks (see Figure 4). If you highlight a block and try to mine it by holding down the left mouse button, however, you'll notice that nothing happens. What's going on here?

Well, although you can dig up soil and sand with your bare hands, for stone you need something more substantial. You need to create a pickaxe, using a technique known as crafting, and for this, you'll need some wood. Do some exploring until you come across some trees and then highlight a block in the trunk and "mine" it as usual. Do this for a few blocks of wood (e.g., 10 blocks) until you've built up a healthy inventory.

Now you're going to craft them into something else. Press the *i* key to bring up a "crafting" table, and drag your mined wood blocks anywhere into the 3x3 grid. On the right-hand side, you'll see that these have been converted into two units of wood planks – drag those back into your main inventory panel. Et voilà, you've just done some simple crafting, turning basic wood into something slightly more useful!

You can keep going further. Drag one of the planks to the middle bottom of the crafting table (Figure 5), and you'll see that you can make sticks, so drag the sticks back into your inventory. (Note that you can use the right mouse button when dragging bunches of items to split them into two.) Now use two sticks and three planks of wood, as

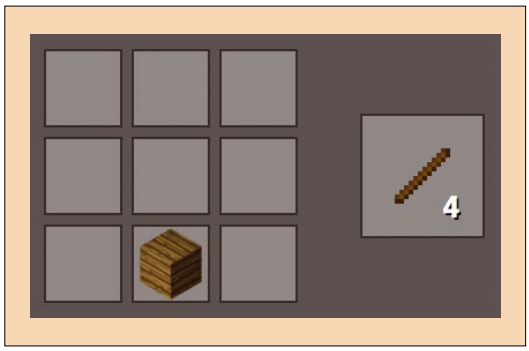



**Figure 5:** Time to start crafting, converting wooden planks into sticks.

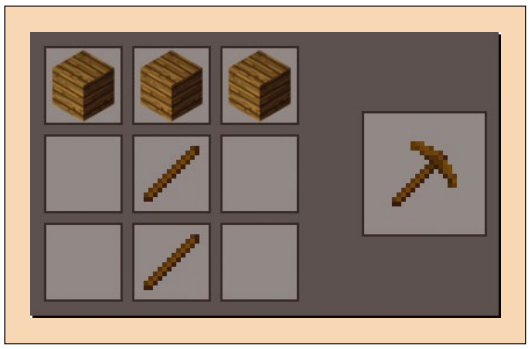


**Figure 6:** With wooden planks and sticks, you can create a pickax to start mining stone.

shown in Figure 6, to create a wooden pickaxe. You can now drag this axe into your inventory, press *i* to close the crafting screen, and then use the mouse wheel to switch between items in your inventory.

Equip yourself with the wooden pickaxe, and try mining stone with it – you'll see that it's now possible. You have one of the most basic tools in the game, and it'll break before long, but at least you have more than just your bare hands. Minetest is full of tools and weapons that you can craft. After a while, you'll feel like a proper warrior, capable of taking on any task – a far cry from the start of the game, when you were just abandoned on a grassy plain!

## Going Further

Now that you can start mining stone, you can build yourself a house. OK, it'll be a rather blocky house, but nonetheless, you can adorn it with a wooden roof and other features. It's possible to build enormous, intricate structures in Minetest and Minecraft; indeed, people have recreated famous landmarks and parts of cities on a block-by-block basis. Once you have your house set up and start exploring nearby, you might want to leave some kind of reminder to find your way back. A simple way to do this is to create small (two- or three-block high) stone pillars along the way. You could even attach torches to them – and the outside of your house – to help find your way back at night.

For more on crafting, including guides for creating the most important items in the game, see the Minetest wiki [3]. It shows you how to create the aforementioned torch, along with a chest to store items (a useful thing to have in your newly built house), as well as other useful bits and bobs. Scroll right down to the bottom of the page for information on smelting – setting up a furnace so that you can turn raw materials into food, gold, glass, and other items.

You might notice that the Minetest landscapes, while rich in vegetation and variety (e.g., deserts, oceans, grassy plains, mountains, and snowy hills) are otherwise rather empty. Because the game is still in the middle of development, the default version has no animals, but you can download add-on "mods" (modifications) from the Minetest site [4]. In particular, check out the Animals Modpack and Creatures Mob Engine. They make the game feel much more alive! ■■■

### Dwarf Fortress

If you're looking for something a bit more old school, but with similar gameplay mechanics to Minetest, check out Dwarf Fortress [5], a text mode alternative. Like Minetest, this takes place in a vast procedurally generated world, and there's an enormous amount to do: establishing a colony, fighting goblin invasions, and finding resources to build up wealth. Graphically it's much simpler, but it means you can run it on any old PC, and you're meant to use your imagination somewhat with the creatures that you encounter.

Dwarf Fortress influenced Minecraft and has a wiki with extensive guides [6]. Like the venerable NetHack, it can become strangely addictive and take up lots of your time – don't say I didn't warn you!

### Info

[1] Minetest: *http://www.minetest.net/*

[2] Download Minetest: *http://www.minetest.net/downloads/*

[3] Minetest wiki: *http://wiki.minetest.net/Crafting*

[4] Minetest add-ons: *http://www.minetest.net/customize/*

[5] Dwarf Fortress: *http://www.bay12games.com/dwarves/*

[6] Dwarf Fortress wiki: *http://dwarffortresswiki.org/index.php/Main_Page*

# FAQ Arduino

## Break out of the cyber confines of your computer, and program the real world.

BY BEN EVERARD

**Q Ard-You-I-Know. What does that mean?**

**A** It's pronounced Ard-wee-no, and it's an open source microcontroller board that's often used to control embedded systems and robotics (Figure 1).

**Q Ah, so it's like a Raspberry Pi?**

**A** Well, yes and no. There are certainly some similarities (it's a similar size, and both have programmable input and output pins that you can use to control external hardware), but there is one very big difference, and that's the processor at the heart of it.

The processor in a Raspberry Pi is a fully featured CPU capable of running a normal desktop operating system like Linux. It's not the most powerful CPU around, but it can do just about anything a more powerful chip can do, just a bit slower. The processor in the Arduino is much more limited. It doesn't have any graphics, or even a memory management unit (MMU), so it's not really capable of the sort of multithreading cleverness that's needed to run a modern operating system like Linux.

**Q So if it can't run Linux, what does it run?**

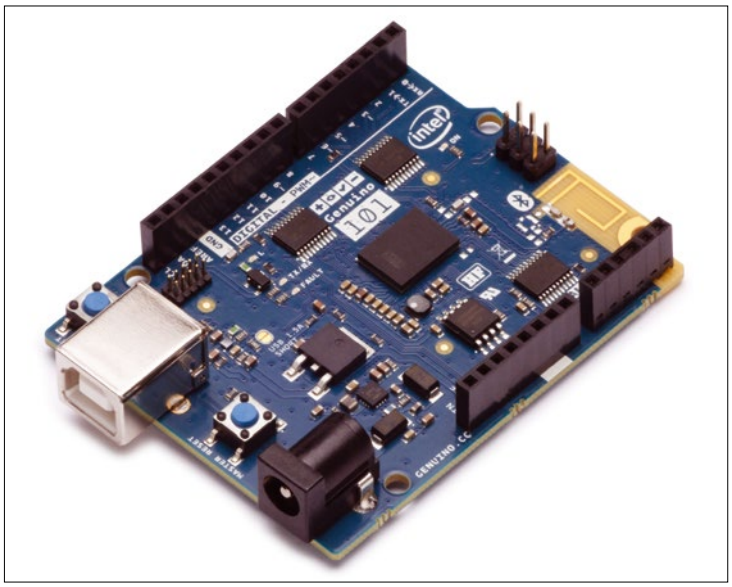


**Figure 1:** The 101 board is a great introduction to the world of physical computing.

**A** In a sense, it doesn't run any operating system. It comes with a bootloader already programmed into the chip, and this just runs the code that's stored in the chip's internal memory. The clever bit is that you can upload more code to it via USB. In many ways, it functions more as a peripheral than a computer, but purists would argue that it fulfills all the requirements of a computer in itself. (I would argue that so do many peripherals, and it's common for hard drives, WiFi adaptors, and more to come with far more capable computers than the Arduino has, but that's an argument for another day.)

The important bit to remember is that it's a programmable thing to which you can upload code from a computer but that doesn't itself come with any more capabilities than to connect over USB and listen for more code.

**Q If all I can do is connect it to my computer and run code on it, why should I bother? I can already run code; I don't need another place to run software.**

**A** The great thing about the Arduino is that while you need it to be connected to the computer to upload code, you don't need to be connected to run it. The code is stored in flash

memory that's on the Arduino, so once the code is uploaded, it will run whenever you provide power to it. Essentially, it's for creating embedded computing environments. This is particularly useful when you use the input and output pins to interact with the real world.

**Q Input and output pins? What are they?**

**A** They're connectors that you can turn on or off. They can connect to LEDs, buttons, motor drivers, or just about anything else to allow your programs to either understand or influence the world around it.

**Q I know nothing about electronics, so how would I go about using these pins?**

**A** Things called shields plug directly into the Arduino pins to provide some out-of-the-box functionality. Although, having said that, the sort of simple electronics needed to wire up things like LEDs or buttons is not complicated, and you could grasp it quite quickly with just a few pieces of inexpensive equipment.

**Q This Arduino thing sounds a lot like a Raspberry Pi, just without the ability to run Linux. Why exactly should I learn a whole new platform?**

**A** The Raspberry Pi and the Arduino do have a lot of features in common. If you're coming to the world of physical computing for the first time, you're probably going to end up using one of these two products. The main difference in our opinion is the operating system. The Raspberry Pi has a complete Linux system that gives you wonderful capabilities for doing everything from running powerful fully featured web and database servers to programming in a wide range of languages and editors. The downside is that this adds a lot of complexity to the system, with more to go wrong, and more to worry about. In an embedded situation, this can mean things like worrying about startup times and data robustness if users power off rather than shutting down. The Arduino, on the other hand, has much less to go wrong. All the code on it is code you've added, so it should start up instantly; the filesystem shouldn't be touched, so you shouldn't have problems with hard shutdowns; there should be far fewer security issues to worry about; and so on.

Both platforms have pros and cons, but my general advice is to stick to the Arduino unless you definitely need the extra power of the Raspberry Pi.

**Q How do I program the Arduino?**

**A** A special IDE contains a text editor, project manager, compiler, and uploader (Figure 2). As far as the Arduino is concerned, this does everything. It's open source, and you can get it from the Arduino website [1].

Once you've got that, you can browse examples and get to work. The code is written in a dialect of C++ similar to Processing. It's all fairly straightforward for anyone with any coding experience, and you'll find loads of examples, both included with the Arduino software and online.

**Q So how did the Arduino come about?**

**A** A very brief history is that it came out of the Interaction Design Institute Ivrea, a school in Italy. There, a group of supervisors and students created the boards to make it easier for students to work with interactive design. There's been a bit of turmoil since then, and the history of the various organizations looks like it might have been written for a soap opera, but the end result is that official Arduino hardware is now produced by Arduino LLC.

**Q You say "official" Arduino hardware. Does that mean that there's unofficial stuff as well?**

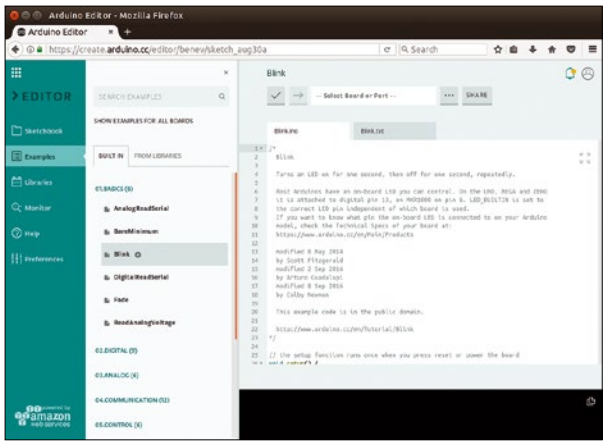



**Figure 2:** There's a web-based version of the Arduino IDE, so you don't even need to install software to get started.

**A** Yep, and in a variety of forms. Since the hardware and software is all open source, other people have been able to build off the platform. This varies from direct clones (often of poor quality) that are for sale on various websites, to boards that have added functionality to the original boards while still remaining compatible with software (and sometimes hardware, too).

**Q Are there different official versions of the Arduino board?**

**A** Quite a few, yep. There are two big things to consider when picking a board: 5 volts or 3.3, and the layout of the pins. Both of these things affect what hardware you can attach. Generally speaking, 5 volts is older and 3.3 volts is newer, so unless you happen to have a stack of old 5V hardware, you're probably better off going for a 3.3V board now. The second thing is the physical layout of the pins: The Uno-style layout (which used to be the standard) shows up on the Arduino Uno and Zero; other styles on other models are better suited to breadboarding. For now, more shields are in the Uno style, but if you're looking to develop your own hardware, one of the other form factors might be more useful.

**Fantastic. I'm off shopping for my first Arduino.** ■■■

## Info

[1] Arduino: *https://www.arduino.cc*

Valentine Sinitsyn works on a cloud infrastructure team and teaches students completely unrelated subjects. He also has a KDE Developer account he's never really used.

# CORE TECHNOLOGY

VM instances in the cloud are different beasts, even if they start off as a single image. Discover how they get their configuration in this month's Core Technologies. BY VALENTINE SINITSYN

## Bringing Up Clouds

First as a buzzword and then as a commodity, the cloud lives the typical life of an IT industry phenomenon. This means that running something (but usually Linux) in a cloud is a thing you now do more often than not. From a user perspective, it's simple: You click a button on the cloud provider's dashboard and get your virtual machine (VM) running within a minute.

This is drastically different from what you do on your desktop. Here, you insert the DVD or plug in a USB pen drive and spawn the installer. Be it an old-school, text-based or a slick GUI installer, it typically asks you some questions (Figure 1). Which locale do you want to use? What's your computer's hostname? What's your time zone? How do you want your user account named? Which password do you want to use? You may not even notice these questions, because installation takes a quarter of an hour or more, and you spend most of this time sipping coffee or chatting with friends. Yet these questions are essential for the system's operation. Without a password, you won't be able to log in. Or, even worse, everyone will be able to.

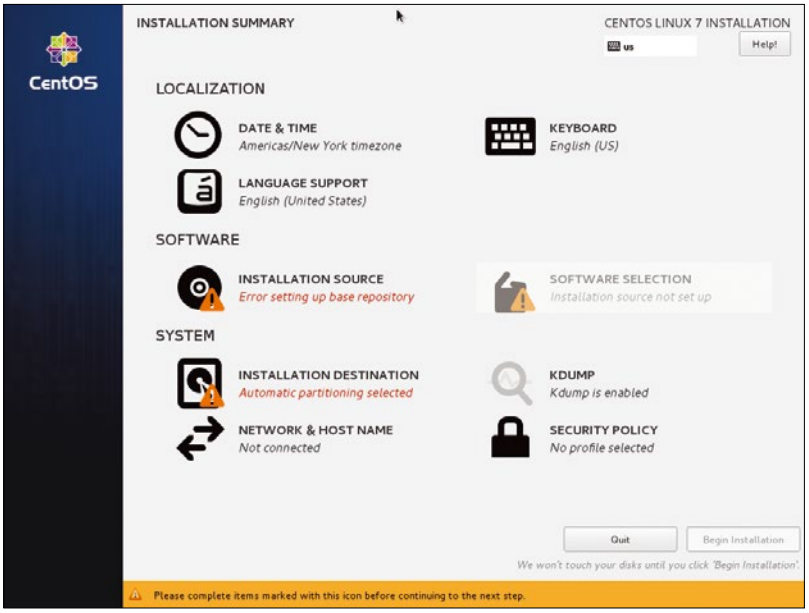



**Figure 1:** The Anaconda installer makes you answer some questions before you can install anything.

A VM in a cloud starts in seconds because it's not really installed. Clouds host prebuilt hard disk images that are effectively cloned when you need a new VM. Although it's faster, it also means there is no "configure" stage, when you can adjust settings to your liking. Cloud-based VMs need some other mechanism to make these changes on the fly so that many instances can start off of a single image. One of these mechanisms is `cloud-init` [1].

### cloud-init in a Nutshell

Initially an Ubuntu project, cloud-init becomes "the standard for customising cloud instances," as the homepage says (Figure 2). The initial idea was to make Ubuntu easier to consume in Amazon Elastic Compute Cloud (EC2) instances. Today, cloud-init integrates with many popular Linux varieties (and even FreeBSD) and runs across different clouds, from Microsoft Azure to your company's private OpenStack deployment. Needless to say, cloud-init is free (GPLv3), and you can get it through your package manager, as well as in Python sources.

As an end user, you don't think about cloud-init too much: It "just works." However, imagine that you want to build your own custom OS image for the cloud. Following the "least surprise" principle for your end user, you'd want to integrate cloud-init and make sure it is able to find the relevant settings and apply them.

You might think of cloud-init as an init [2] process replacement because of the name, but that's

not true. Linux already has many init implementations, and it makes little sense to write another one just because your system now runs in the cloud. Instead, cloud-init is designed to interoperate with an existing init, which calls it at well-known points during system startup.

Like it or not, in most Linux flavors today init is equal to systemd, so I'll look at cloud-init from this perspective. Otherwise, the stages are almost the same, but implementation varies per init, as you might expect.

## Up and Running

It all starts with the generator (see the "What on Earth Is a Generator?" box), which decides whether cloud-init needs to run at all. To disable cloud-init, you either create an `/etc/cloud/cloud-init.disabled` file or pass `cloud-init=disabled` as a kernel command-line option. The `KERNEL_CMDLINE="cloud-init=disabled"` environment variable takes precedence over the latter, which is useful if you run cloud-init in a container rather than a VM. So, you can temporarily disable cloud-init to run an image outside the cloud in your local virt-manager, for instance. This step is the only systemd-specific one, because no other init system has a notion of a generator.

If cloud-init is not disabled, the generator injects `cloud-init.target` into the `multi-user.target`. This target uses `WantedBy` to pull several units (`.service` files) that ultimately call `/usr/bin/cloud-init`, passing it different command-line options. Note that this executable is not a daemon: It does what was requested and then exits, and you won't find it running in your cloud instance. The wonderful cloud-init documentation [3] covers the boot process in a greater detail, but here is a quick summary for your convenience.

First, there is `cloud-init-local.service`, which runs as soon as the root filesystem becomes writable and before the network is configured. It translates to `/usr/bin/cloud-init init --local`, and the main idea is to render the required network settings on the first boot.

Second, `cloud-init.service` already has access to the network; that is, it can use non-local data sources (I'll revisit this later). This stage runs `/usr/bin/cloud-init init` to finalize the initialization process. It can configure SSH or provision CA certificates, for instance.

Third, cloud-init modules come into play with `/usr/bin/cloud-init modules`. This happens in two stages (or substages): the "config" stage (`cloud-config.service`) and the "final" stage (`cloud-final.service`). The config stage modules may install packages or configure NTP and the time zone. At the final stage, you upgrade packages and run configuration management agents, such as Salt Minions [4]. This is a proper place for your custom setup. Think of it as an `rc.local` equivalent in a traditional init.

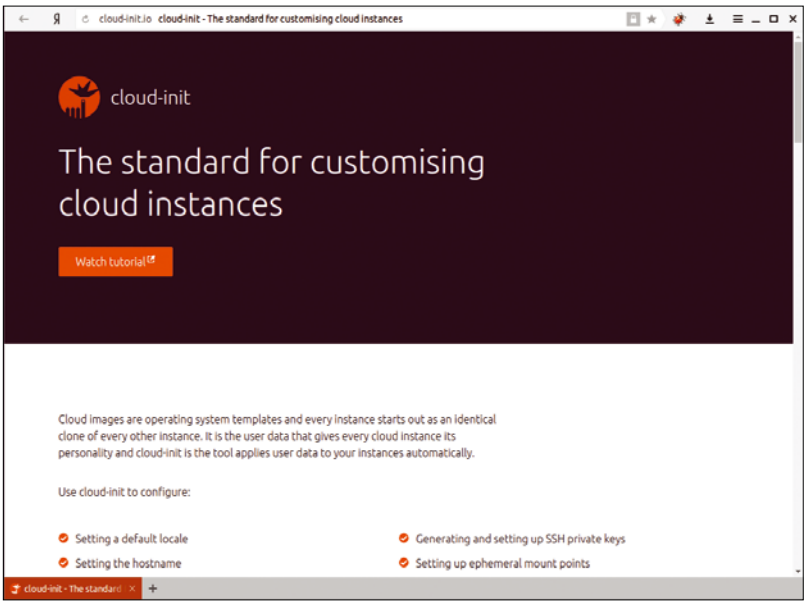



**Figure 2:** Cloud-init is no longer an Ubuntu-only thing, but the well-known palette readily reveals its origins.

## Modules Galore

The problem with computers is that they do what you say, not what you mean. Before cloud-init can apply any configuration, it must be told exactly what you want to apply. Internally, cloud-init stores all settings in a large `dict()`, and while it's not a Perl hash, there is more than one way to fill it.

For starters, cloud-init comes with a bare minimum of defaults. They enable data sources for the most popular cloud providers and do some other tweaks. The main configuration file resides in `/etc/cloud/cloud.cfg` by default, and it's an appropriate place for settings shared between all instances. For instance, you can configure your company's package repository or

### What on Earth Is a Generator?

A generator is a systemd concept. It's a small binary whose name ends with `-generator`. Generators run very early in the system startup and were designed to convert non-systemd configuration files into native units (hence the name). They should write unit files or create symbolic links, introducing new dependencies between them. This makes generators a viable means to customize the systemd boot process.

There are different places where a generator can reside in your system, each with its own priority (consult the `systemd.generator(7)` man page [5] for details). For instance, on my Ubuntu 16.04 LTS system, `cloud-init-generator` resides in `/lib/systemd/system-generators/`. It's a small shell script spanning a bit more than a hundred lines of code. This script performs the checks I discussed in the main text, and if cloud-init is not disabled and the data source is found, it creates a symlink to `cloud-init.target` in the appropriate `multi-user-target.wants` directory.

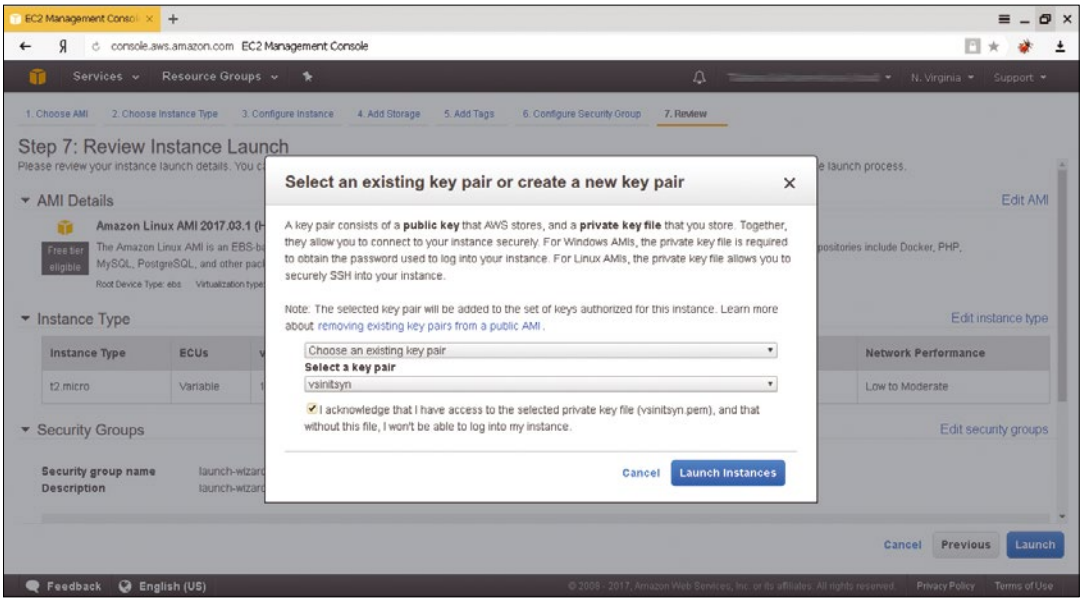



**Figure 3:** Supplying a valid key pair is essential in clouds: If you do it wrong, you have a hard time logging into your instance.

add an internal CA certificate here. Additional config files may come from `/run/cloud/cloud.cfg` (this is known as a "run-time config") or in fact any location you supply via a `CLOUD_CFG` environment variable. Again, the latter is mostly useful in containers.

Many data sources would allow you to supply configuration bits via some type of data: metadata, user data, or vendor data. This is the way to go for all instance-specific settings, but I will leave the details until the next section.

This may sound weird, but you can also configure cloud-init via the kernel command line! The format is:

```
cc: config bits end cc
```

Cloud-init translates a `\n` into a newline, so you can pass it a multiline text. Imagine you have spawned an instance and realized you've chosen the wrong SSH key. The warning shown in Figure 3 may look difficult, but if you can get to the VM's GRUB prompt, things are easy to fix. Just append

```
cc: ssh_import_id: gh:johndoe end_cc
```

to the kernel command line, and it will make cloud-init import your (actually, John Doe's) public key from GitHub. That's exactly what the `ssh_import_id` module is for.

`/etc/cloud/cloud.cfg` is actually a YAML file. This holds true for most other configuration means, such as the kernel command line above. Moreover, cloud-init supports putting configuration snippets in `/etc/cloud/cloud.d` to make it easier to extend configuration from the scripts. Those snippets must also be valid YAML, and they are merged together in lexical order as if they were forming a single file [6].

Now, I'll take a quick look at what's typically in `/etc/cloud/cloud.cfg`, which depends on the distribution and who built the image, so I opted for a generic CentOS 7 OpenStack image [7]. The config file is about 60 lines long, and you can find a heavily trimmed version in Listing 1.

First, you see some settings for the cloud-init modules. They prescribe creating a default user, disabling root, and SSHing password authentication, as recommended for networking setups. Three modules really take care of this: `users-groups`, `ssh`, and `set-passwords`, respectively. The first two come through `cloud_init_modules`, which lists modules to run at the init stage. The last one is from `cloud_config_modules`, so it runs at the `config` stage.

If you have a background in YAML, you may think options such as `disable_root` should be nested under `- ssh`, and you are probably right. Remember, however, that the same configuration bits may come through the kernel command line or other means. They have a different structure, and putting these settings at the top level makes handling module code more uniform.

Next is the `system_info` section. You may think of it as a read-only piece of information regarding the system itself and the distribution. You specify the latter via the `distro` key, which is important because different distributions provide different tools for common tasks, like applying network configuration. This is not to mention that modules such as `yum_add_repo` are naturally per distribution. The aforementioned `ssh_import_id` is also a good example. It runs on Ubuntu and Debian

**Listing 1:** /etc/cloud/cloud.cfg (Abbreviated)

```
users:
 - default

disable_root: 1
ssh_pwauth:   0
...
cloud_init_modules:
 ...
 - users-groups
 - ssh
cloud_config_modules:
 ...
 - set-passwords
 ...
cloud_final_modules:
 ...
system_info:
  default_user:
    name: centos
    ...
  distro: rhel
  ...
```

**Listing 2:** Sample OpenStack Metadata (Abbreviated)

```
$ curl http://169.254.169.254/openstack/latest/meta_data.json | python -mjson.tool
{
    "availability_zone": "nova",
    "hostname": "my-instance",
    ...
    "random_seed": "...",
    "public_keys": {..},
    "uuid": "ec017a2a-b93d-4346-a1b3-db3d5e09a4af"
}
```

only, perhaps because the `ssh-import-id` command itself [8] is Ubuntu-specific.

The `default_user` key is another typical resident under `system_config`. It specifies the default user parameters that end up in `/etc/passwd`. The code in Listing 1 shows that you will be able to log in as *centos*, using the SSH key that you provided, thanks to the `ssh` module.

## Defining Settings

Now, the last missing bit is how you define settings that are specific to the instance, and that's where things start to get really interesting.

Cloud-init implements a data source concept that provides an abstraction for the data, which comes from the cloud content management system (CMS), such as OpenStack. Hostname, locale, or SSH keys are all valid examples. Moreover, a data source provides so-called user data and vendor data. For cloud-init to consume user data, it must come as a MIME multipart archive, which is the same format you use when sending email messages with attachments. Each part (or file) has an associated `Content-type` header and also begins with a signature: `#something`; cloud-init uses this information to decide how to handle the particular part.

User data is typically no more than 16KB. To overcome this limitation, the `#include` file can list additional URLs, one per line. They are retrieved and treated as if their contents were a part of the original user data. To make things even more compact, you can use Gzip compression. The `#cloud-config` file is in essence an `/etc/cloud/cloud.cfg` snippet embedded in user data. A few other types are supported out of the box, such as raw shell scripts, and it's possible to define your own content types and their respective handlers. See the *Formats* section on the cloud-init documentation page [3] for details.

Cloud-init comes with a variety of data source modules and enables most of them by default. You can redefine this via the `datasource` key in `/etc/cloud/cloud.cfg`. This makes sense, because you typically know which data sources are available in your cloud. Some data sources need network access, and some don't. When started, cloud-init probes enabled data sources one by one, and the first one that replies wins.

Perhaps the simplest data source is the OpenStack configuration drive, ConfigDrive. It's a tiny disk image (ISO 9660, or VFAT in earlier versions), and it is employed sometimes to provide networking settings. Cloud-init can then apply these settings and use a full-fledged data source, such as EC2 metadata, to do the rest of the configuration. In theory, ConfigDrive can provide you with all of the metadata, but a dedicated metadata service would work better.

What's metadata, you ask? It's data that describes the running VM instance. Typically, it's encoded in JSON and comes from an HTTP service listening on a link-local address (e.g., 169.254.169.254). In reality, this server is just a proxy that forwards requests to the CMS. The exact API is, of course, cloud-specific, but perhaps the two most popular are EC2 metadata [9] and OpenStack metadata, which is an extension to EC2. Listing 2 shows the anatomy of the latter.

The name assigned to this VM is `hostname`. The `set_hostname` module uses this bit of data to apply the setting in a distro-specific way. The `ssh` module installs `public_keys`, the SSH keys for the default user, and `random_seed` is a 512-byte base64-encoded blob of random data that usually comes from the host's `/dev/urandom` and serves as an external entropy supply to the VM. All VM instances start off of the same image and perform roughly the same initialization steps, so their kernels end up with similar entropy pools. This could be a security breach, as random numbers are often seen in cryptography, and they should be, well, random, not the same as on your neighbor's VM. Moreover, because VMs typically do not have so much hardware from which to collect the noise, their pools are easily exhausted, making tasks such as SSH key generation hang. The `seed_random` module is here to plug into these holes. ■■■

## Info


[1] Cloud-init homepage: *https://cloud-init.io/*

[2] `init(1)` man page: *http://man7.org/linux/man-pages/man1/init.1.html*

[3] cloud-init documentation: *http://cloudinit.readthedocs.io/*

[4] The Salt Open Source Software Project: *https://saltstack.com/community/*

[5] `systemd.generator(7)` man page: *https://www.freedesktop.org/software/systemd/man/systemd.generator.html*

[6] YAML cheat sheet: *https://learnxinyminutes.com/docs/yaml/*

[7] Generic OpenStack images: *https://docs.openstack.org/image-guide/obtain-images.html*

[8] `ssh-import-id` homepage: *https://launchpad.net/ssh-import-id*

[9] Amazon EC2 instance metadata and user data: *http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html*


# Command of the Month: **ec2-metadata**

While it's trivial to query cloud metadata with `curl`, it is certainly not the most convenient way. You get raw unformatted JSON that you feed to Python to make it barely readable; then, you look up the reference documentation to learn the meaning of all of the fields.

If you are on AWS or another cloud that serves EC2-compatible metadata, there is a better way. `ec2-metadata` is a command-line tool that you can use to query metadata in scripts or interactive sessions. It supports Linux and runs on top of `curl`, so make sure you have the latter installed.

To begin, download `ec2-metadata` from Amazon S3:

```
curl -o ec2-metadata ➥
   http://s3.amazonaws.com/➥
   ec2metadata/ec2-metadata
```

Next, make it executable (it's a Bash script), and type

```
./ec2-metadata --help
```

to see the options available. Note this won't work if EC2 metadata is unavailable, even if displaying the help page doesn't require metadata access. `ec2-metadata` dumps all information by default, but you can narrow down the output using one of the command-line switches. For example,

```
./ec2-metadata --public-keys --public-ipv4
public-keys:
...
public-ipv4: 34.226.202.171
```

lists public keys and the IPv4 address assigned to the instance (Figure 4).

To retrieve user data, enter

```
./ec2-metadata --user-data
user-data: Hello, EC2 user data!
```

(see also Figure 5).

Note that it doesn't have to be anything related to the cloud-init multipart data format. Although the instance metadata is not what you are interested in most of the time, tools such as `ec2-metadata` could aid troubleshooting in cases when cloud-init fails to initialize the instance properly – for whatever reason. ■■■

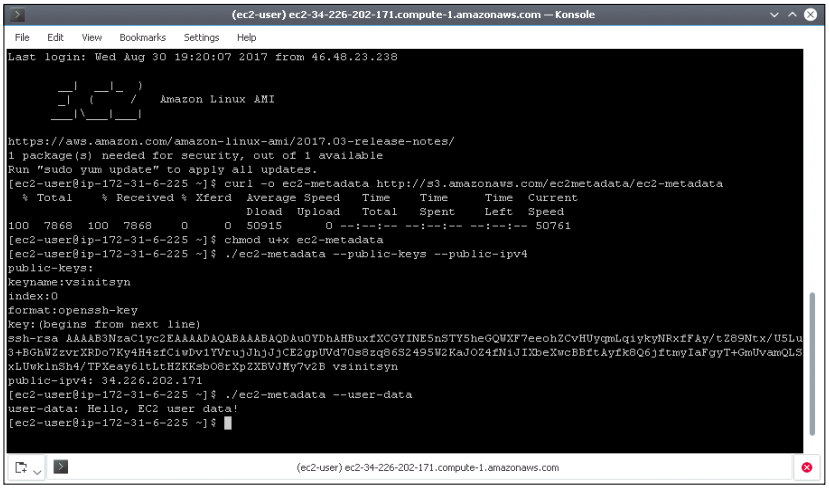



**Figure 4:** The `ec2_metadata` tool is really just a wrapper over `curl`, yet it makes our lives easier.

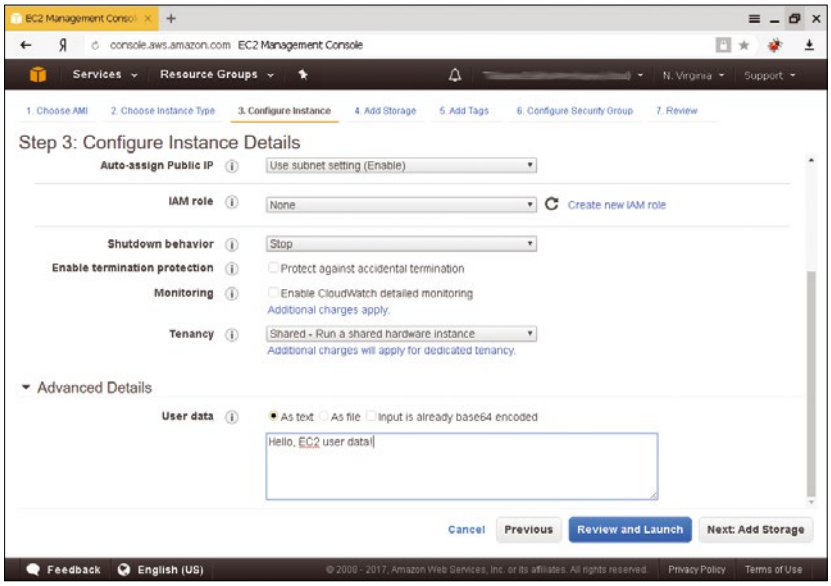
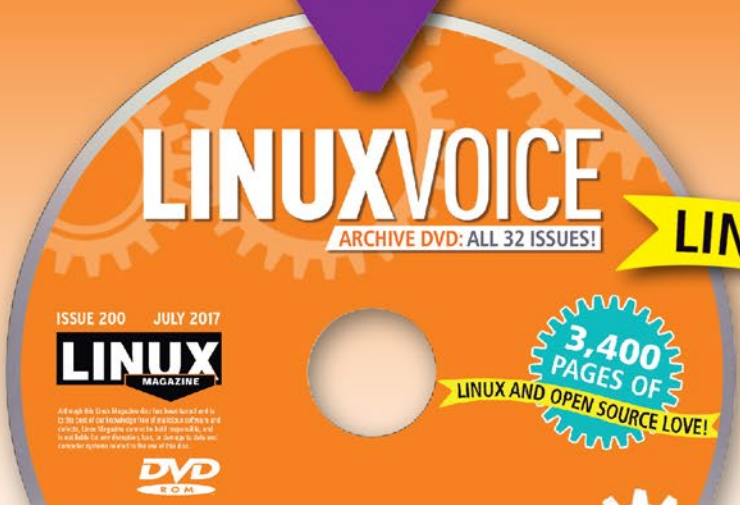



**Figure 5:** To provide custom user data for the EC2 instance, go to the Advanced Details section.

# THE COMPLETE LINUXVOICE

## ARCHIVE DVD: ALL 32 ISSUES!




3,400 PAGES OF LINUX AND OPEN SOURCE LOVE!


Since April 2014, Linux Voice has showcased the very best that Free Software has to offer. Now you can get it all on one searchable DVD.

**Includes all 32 issues in EPUB, PDF, and HTML format!**

# Order now!

**shop.linuxnewmedia.com**

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

# MADDOG'S DOGHOUSE

## Virtual private cloud environments.

BY JON "MADDOG" HALL

I have always felt that the best type of business was a "win-win-win" situation: The community of project and product producers should win, vendors of the projects and products should win, and, most of all, the end-user customers should win.

Many cloud solutions have central ownership by a large company that builds a server farm and sells hardware, software, or "platform" services to customers. Sometimes those sales are for money, sometimes for access to your information for marketing, and sometimes a combination of the two, but it is the end user who gives up something for access to the computer resources that they need. Almost always the end user has no control over where their data is stored, where their programs execute, or what programs they are executing. Because of a lack of control, the end user sometimes experiences charges that could be called "the end of the month surprise."

Without ownership of the resources, coupled with a lack of control, your data might fall under the laws of a country other than your own. Although your country's laws might protect you, another country's laws could expose your data for use, with no ability for you to change those laws by any means.

Recently I accepted the offer to become CEO at OptDyn [1], a company that offers that win-win-win solution through its open source, peer-to-peer, secure, and stable cloud solution called Subutai [2], as well as other "open" offerings.

Subutai forms virtual private cloud (VPC) environments for end users, who consume resources from authorized Subutai peers to grant resources for their applications. These peers mutually authenticate to create a virtual private network (VPN). Once the VPN is secured, peers contribute resources as Linux container hosts inside the VPN using a container-as-a-service (CaaS) cloud model. The end user can install any kind of service, application, or infrastructure software they wish on these hosts running in their cloud.

End-user cloud owners trade "goodwill" with peer owners in exchange for using their peer resources at an hourly goodwill rate. In the near future, an indelible blockchain-based ledger will be used to track goodwill and reputation while enabling smart contracts to implement service-level agreements (SLAs) between peer owners and cloud owners.

True to community values, Subutai awards "goodwill" for good deeds and habits that improve the system for everyone. Users inviting others, setting up peers, keeping peer uptimes high, reporting bugs, or even upgrading to new releases are awarded goodwill. You can gain even more goodwill by donating peer resources to open source projects. These projects can benefit from supporting infrastructure and by using resources for testing. In return, the open source projects may provide Subutai blueprints for their products to install and run on your private cloud at the press of a button. Everyone wins!

In my career, I have seen wasted resources in many settings, such as hospitals that have PCs running all the time but are idle 99 percent (or more) of the time, and universities with laboratories full of standalone PCs attached to a LAN, unable to reconfigure them easily to a high-performance cluster, while other parts of the organization suffer from a lack of resources. Subutai was designed to fix these problems.

Subutai can integrate the Internet of Things (IoT). Many of the IoT models I have seen adapt an application to the cloud and have the "Things" talk only to the application. Many "Things" in IoT can be presented as a resource, so these "resources" could be offered, bartered, or purchased by the entities that want them. A "virtual laboratory" could be set up using CaaS.

I know that "applications are everything." Subutai utilizes the Google App Engine APIs; all applications that run on that platform will run compatible binary on Subutai. On the other hand, other applications that are packaged into containers and use a "blueprint" to set up resources also can run with a small amount of application development work.

Readers who say, "This sounds really good. It is the integration, security, and the base functionality that I would like to have someday," do not have to wait. Subutai, a project originally funded by a government agency to be more efficient in sharing resources, was released as open source and is now in version 5.x, ready for use.

Development of Subutai is led by OptDyn Founder/CTO Alex Karasulu, who founded several projects at the Apache Software Foundation. He describes the Subutai engineering team, along with Director of Marketing and Media, Sally Khudairi, as "really, crazy good," and I agree with Alex.

OptDyn wants to grow the Subutai network, giving more access to resources around the world, and we invite you to download the code and join the community.

Win-win-win. ■■■

## INFO

[1] OptDyn: *http://optdyn.com/*

[2] Subutai: *http://subutai.io/*

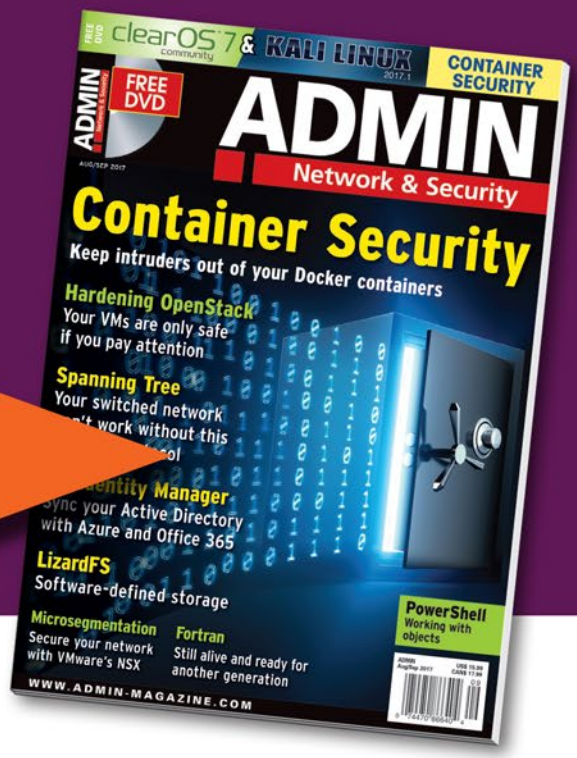
# What?!

## Archives

come with my

digital subscription?


# Archives + Current!




Sign up for a digital subscription to get the latest issues of ADMIN, **PLUS access to archive articles.**

shop.linuxnewmedia.com

**That's a lot of articles!**

# FOSSPicks

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

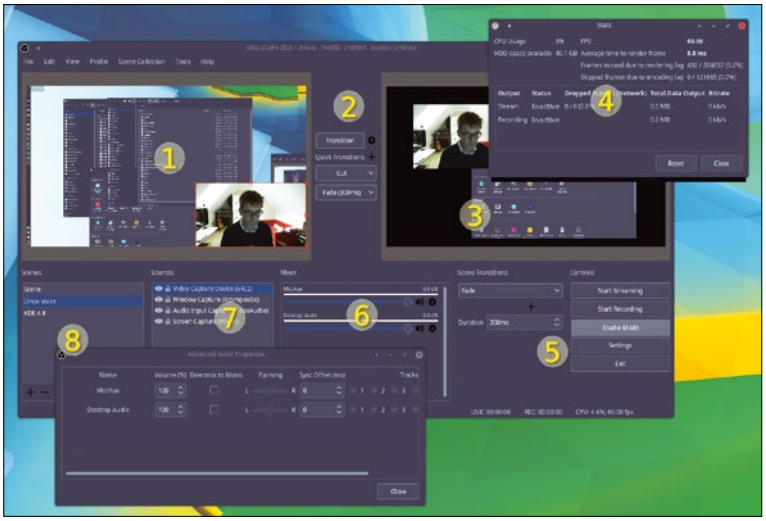
Graham tears himself away from updating Arch Linux to search for the best new free software. BY GRAHAM MORRISON

Screen streaming studio

## OBS Studio 20

Screen recorders have become almost mainstream with so many people sharing their thoughts and screencasts on the Internet. Which may explain why we're looking at two this month – Green Recorder 3.0 on the next page and this, Open Broadcaster Software (OBS) Studio 20. OBS Studio has considerably more ambitions than simply turning your mouse wiggling into a file, as its name implies, and consequently does far more than simply stream your screen to the Internet. Alongside being completely cross-platform, it offers the kind of functionality you're more likely to find in a nonlinear video editor; you can capture, composite, edit, encode, and stream video content, directly from your Linux desktop.

When the application starts, the size of your canvas can be different from the size of your screen. The canvas size is important because you can add lots of different sources to this canvas, all with their own resolutions, and the final output will always be scaled to your overall canvas resolution. You can even set a different base and output resolution for the canvas, effectively oversampling your output for higher quality, which is an excellent option if you have the system to handle it.




**1 Preview.** In Studio Mode, the left panel allows you to build a scene with a real-time preview. **2 Transition.** Change the way a recording or stream shows the scene transition. **3 Broadcast view.** Work live and on a hidden scene at the same time. **4 Monitoring.** Alongside a huge range of transcoding options, there's decent system performance monitoring. **5 Studio Mode.** Switch between normal capture and streaming, or Studio Mode with transitions. **6 Monitoring.** See audio input as it's captured. **7 Sources.** Create a scene from lots of different audio, video, and image sources. **8 Scenes.** Save all inputs as a scene for quick retrieval.

The background organization of a recording or stream is mirrored in the application window layout, especially in the lower panes. Sources, in the bottom left, are the containers for each recording setup, just like presets, allowing you to switch between them easily and save configurations for how you want to use the application. This involves a set of sources, which are listed in the pane to the right. This Sources pane can behave like a clip manager in a video editor, containing image, text, and video files – but, crucially, sources can also be real-time inputs. You can choose different audio inputs and outputs, for example, alongside different video inputs and output, and not just video capture devices – a window or entire screen can be used as an input source, alongside images for overlays and text. In this way you could create a webcam overlay of yourself alongside the screencast, for instance. As with Gimp, these elements are layered, so you'd put the webcam source at the top if you wanted this to sit over a window capture. As you add sources, they appear in the preview window above, where you can rearrange them and resize them for the capture or stream.

However, the real standout feature in OBS Studio is Studio Mode. With this enabled, you can transition between one scene and another, broadcasting or only recording the scene on the right of the window. This is brilliant in live situations as it allows you to line up sections in one scene while performing in the other, switching over seamlessly when ready – exactly as you might in a television studio, and it takes open source screen casting to a whole new level of capability.

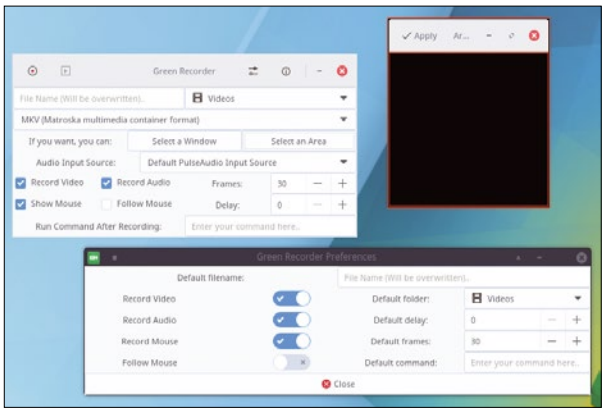
**Project Website**
https://obsproject.com/

Screen recorder

# Green Recorder 3.0

Like waiting for a bus, screen recorders are now appearing everywhere. They may not seem the most important kind of application you can install, but for those of us who occasionally share screen sessions, they're the difference between wanting to do it again or giving up entirely. This is because recorders deal with huge amounts of data – grabbing the contents of the display, maybe 4K or ultrawide, tens of times a second; synchronizing that with an audio source or two; and then saving the output as something useful, rather than a huge data dump. It has to do all of this without adding system load or getting in the way of whatever it is you want to record. This is why there are so many options and why so many of us opt to use SimpleScreenRecorder simply because it has proven itself to work well.

But Green Recorder, especially with this huge update, is a serious challenger to SimpleScreenRecorder. Of particular note is that it now supports the GIF format, and the image size has been optimized for the process – with claims of this conversion being 10 times better than the same process with FFmpeg. There's also a new Preferences window, and the overall design and simplicity of the user interface is one of Green Recorder's best features. You can easily see which audio input from PulseAudio to grab, for example, or change the desired frame rate, the file format, and whether the recorder follows your mouse. It's simple to use and capable of excellent results, and it caused no issues when recording from a huge composited Nvidia-driven display, complete with wobbly windows and spinning cube. If it can keep up this reliability and usability, Green Recorder is likely to replace SimpleScreenRecorder for those of us with secret YouTube channels. Plus, having a choice is always a good thing.




Turn yourself into a YouTube megastar with regular KDE screencasts, thanks to the latest Green Recorder.

**Project Website**
https://github.com/foss-project/green-recorder

Android terminal

# Termux

Termux is an open source terminal emulator for Android. Now, admittedly, Android has many different terminal emulators, but none come close to offering the power, functionality, and ease of use of Termux. Two parts of Termux make it so powerful. The terminal emulator itself works brilliantly. It's quick, it uses colors of your choice, and it adds neat options such as shortcut scripts that can be run directly from a widget on your phone. It also has one of the best input mechanisms I've seen, augmenting your default keyboard with a layer of keys essential for Linux: Esc, Ctrl, Alt, Tab, -, /, and |. Added to this, you can hold down the Volume Up key to access a whole host of shortcuts. Hold Volume Up+W, A, S, or D, for example, for cursor control. Volume Up and a number is a function key, and it's absolutely the best interface I've come across on Android for Linux terminals.

But the best thing about Termux is that it includes its own package manager, and you don't need Android root to add essential Linux utilities to its special `chroot`-like environment. These packages, which can all be found on GitHub, are specially built, and there are hundreds of them, from SSH to Vim; just type

```
pkg install <package-name>
```

and they'll be added to your Termux environment. You can even give Termux access to your Android storage, without root, so you can read and write to user-accessible parts of the Android filesystem. It all works perfectly, and you can even add a widget that hovers over any other Android screen for instant terminal action from whatever app you're using. It may sound like a gimmick, but after you create a few shortcuts for your servers and start editing configuration files with Vim (really!), Termux becomes about the most useful app you can install.


Termux is available on both F-Droid and Google Play, and I'd highly recommend dropping the developer a few coins for this wonderful app.

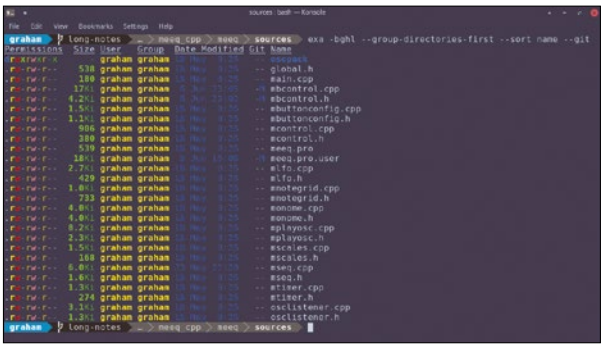
**Project Website**
https://termux.com/

File listing

# Exa 0.7.0

On the terminal, `ls` is surely a contender for being one of the most used commands. Typing `ls` is often muscle memory for those of us who often navigate the filesystem from Bash, checking the contents of directories or the sizes of files. It's quick to type and often just as quick to deliver results, and there's never been a really good reason to replace it. However, this command – `exa` – offers enough upgrades over vanilla `ls` that it might be worth the alias in your `bashrc`. However, there are several big differences when using `exa`, and the first you'll notice is the output, especially long `-l` output. The output from `exa` is always full of color – and not just color for its own sake, but functional color. Each permission has a different hue, for example, as does the Date, owner, and Name columns. You can also view extended attributes and sizes, making the additional colors both easier on your eyes and a valuable addition when surfing through hundreds of files.

This profusion of color is not the only advantage `exa` has over `ls`; `exa` includes a tree view by default, which is still the best way of getting a good overview of your system if you can't access a GUI. You can even see which files are staged and unstaged when navigating a Git repository. Git integration may seem niche, but it's very useful if you don't use something like Powerline, as you often navigate projects from the command line while executing `git status` to see what's staged. Because of the way the arguments are written and because of some of the extended



One of `exa`'s best features is its ability to show which files in a Git repository have been staged, even from the tree view.

options, `exa` can't directly replace `ls`, and typing `exa` is certainly more laborious than the two finger rapidity of `ls`, but I've found it useful enough to replace `ls` in most circumstances, especially with the

```
exa -bghl ➥
--group-directories-first ➥
--sort name
```

set of arguments, which can easily be used as an alias for `ls -al`.

**Project Website**
https://the.exa.website/

System monitor

# gtop

In the future, there will be no desktop environment, only tmux full of lots of command-line utilities doing everything more efficiently than their GUI counterparts without the distraction of YouTube. In this future, `gtop` might become your task manager. It's like `top` only better looking. Each core and CPU is charted for usage across the largest panel at the top, along with a numeric percentage alongside. Beneath this pane, memory and swap is documented with both a percentage chart and a pie chart showing usage – the latter being very useful because a quick glance tells you how close to maximum you are. The lower half of `gtop` is then split into a great little network monitor, showing input and output bandwidth usage for your machine, a process list just like the original `top`, and a disk usage pie chart.

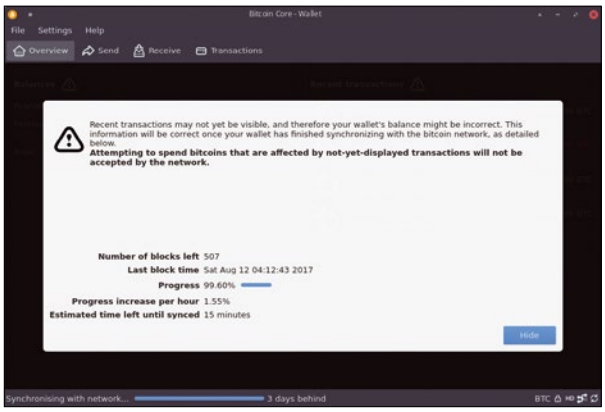
Of course, all of this information is only ever a command away in Linux, but seeing it so well presented and accessible is what makes `gtop` so useful. If you run remote servers, for example, you could create a tmux session split into `gtop` running on many different machines at once, and a quick look over their status would tell you how those machines were running, without necessarily looking into the details unless something appeared wrong. It would also look rather awesome if anyone happened to be looking over your shoulder. Despite the superficial nature of its good looks (and the `npm` install requirement), `gtop` is a great little tool that can help you keep on top of your machine's resources, whether that machine is local or remote, and you don't have to remember all those keyboard shortcuts, as you do with `top` or even `htop`.


Make your system usage look like something a Hollywood hacker would use with gtop.

**Project Website**
https://github.com/aksakalli/gtop

Bitcoin wallet

# Bitcoin Core v0.14.2




Forget quantitative easing. There will only ever be 20,999,999.9769 bitcoins.

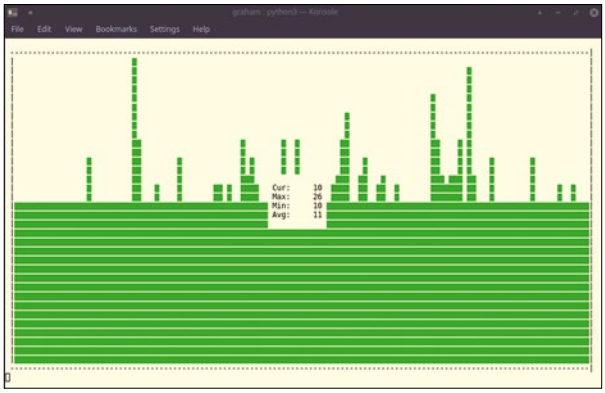
It's been a tumultuous month for bitcoin prices, with the currency passing the $4,000 dollar mark after remarkable gains. While it's impossible to predict what the price might be as you read this, this decentralized flux is part of what makes bitcoin and other digital currencies so remarkable. With lack of control and "no governance," the central tenets of the currency (along with a certain kind of transparency surrounding the transactions that add and reduce its value) are analogous to open source itself, and it's difficult to see (or trust in) bitcoin's value without the supporting tools that help the currency itself being open source. Bitcoin Core is one such tool.

Bitcoin Core is essentially an open source wallet or client for managing your own bitcoins, or more likely your satoshi (one hundred millionth of a single bitcoin). The Qt-based version 0.14.0 was a major update, including performance improvements, a network activity toggle, and a useful Hierarchical Deterministic (HD) indicator. But what makes Bitcoin Core important is that it's considered the reference client that sets the standard, both in protocol and implementation, locally and across the network. Thanks to the way it downloads every bitcoin transaction to validate the entire blockchain locally, you don't need to rely on an external exchange or entity to send and receive bitcoin. This obviously comes with a huge caveat: If you lose your local wallet, you absolutely irrefutably lose your bitcoin. Bitcoin Core mitigates this by allowing you to encrypt your wallet for safe backup, which you should do to a trusted source, but it's always something to consider. However, outside of the ethics of bitcoin, the idea behind blockchains and digital currencies is fascinating and helps make Bitcoin Core the perfect playground for a little experimentation.

**Project Website**
https://bitcoin.org/en/bitcoin-core/

Network monitoring

# gping




Not to be confused with a Windows tool of the same name, our `gping` sticks to the terminal.

Keeping with the theme of testing out command-line tools starting with the letter "g" that replace ancient command-line stalwarts (see `gtop`), `gping` is an attempt to usurp `ping`. Many of us must use `ping` all the time – it sends a signal to a remote computer and waits for an echo request packet to be sent back. Its name comes from the sound of a sonar pulse, and it was originally written by Mike Muuss in December 1983 as a simple network troubleshooting tool, but it's just as useful in the age of flaky WiFi and occluded 4G. Fire up a simple `ping` command, and you can see not only whether the remote site is online, but whether it really is taking an age for a page to load, as well as whether your DNS is working and whether the response time changes under different network conditions.

`gping` does the same thing, but it adds a very useful graph to its output, so you can see the change in latency over time. While the original output from `ping` could be scanned and scrolled, it is often difficult to get a real feeling for how erratic a `ping` value might be from a big list of milliseconds. With `gping`, you can see whether the fluctuations are large or small and easily see whether something else may be affecting the network latency between you and the server you are testing. It could be that some huge download is sucking up every bit of bandwidth, for example, and you'll be able to see how this affects latency on your network. You can also see the current latency, alongside the maximum, minimum, and average for the time you run `gping`, all of which helps when detecting problems.

> gping adds a very useful graph to its output, so you can see the change in latency over time.

**Project Website**
https://github.com/orf/gping

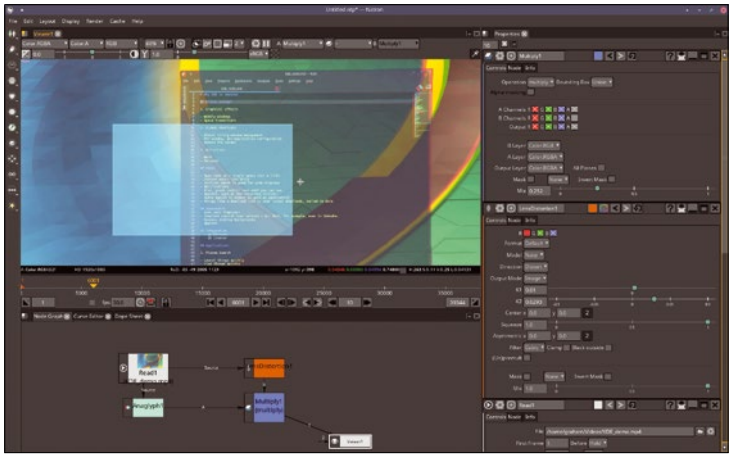
Video processing

# Natron

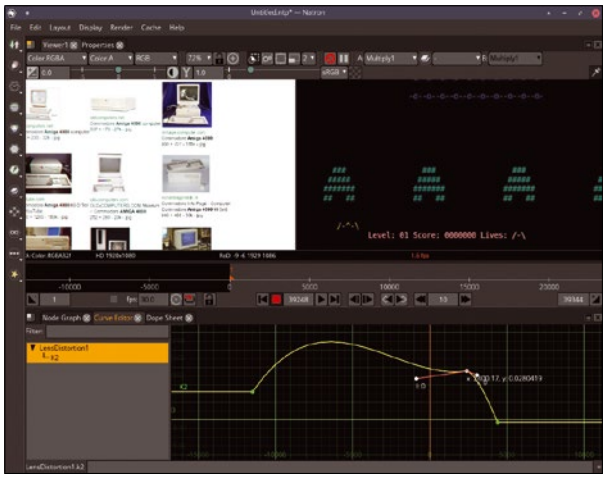
Natron, like Blender and Krita, is one of those applications that has been in development for a long period of time and has become a significant cornerstone for an entire industry. Yet those of us without a specific interest in animation or video might not realize just how powerful and capable Natron has become. Natron performs the graphical equivalent of audio mastering, letting you process and mix video and image data from multiple sources to superimpose other video or rendering layers, improve the output, and add postproduction effects. In the world of video and image, this is known as compositing – the combination of these various sources into a single output. It's a little like using Gimp to fix an old image, add color and improve the contrast, and superimpose a UFO hovering in the background, only Natron does this with moving images, dynamically, and, crucially, using a node-based user interface rather than a layer-based one.

Applications like Gimp, Adobe Photoshop, and, more similarly, Adobe After Effects (an application that serves the same purpose as Natron) implement these processes through layers, where each layer is normally reserved for a separate process or effect. A node-based user interface, on the other hand, replaces layers with separate processes (nodes) that you manually choose how to link together – just as you would modules in a modular synth; this gives you more control. The simplest example uses a Join Views node linked to two views with a single output, the contents of which is a mix of the two inputs; however, nodes can be used for anything, from color correction and lens distortion to rotoscoping with masks. Natron even has commercial nodes that can be used alongside the huge number of open source nodes that are bundled with the default package.

After you get the idea of nodes into your head, Natron is actually straightforward to use. Drag image and video sources into the node area and start connecting nodes between these sources and the viewer showing the output. Above this node area is a preview of the processed output, complete with navigation controls, and it's from here that you also step through frames and keyframes. The keyframe mechanic is particularly



Like Blender with shaders and materials, Natron uses nodes to process streams of image data.



The transition between keyframes is handled by the powerful Curve Editor.

powerful; this is used to change elements within each node over time. If you want to change a color value at 30 seconds, for example, and get this to merge into another value at 35 seconds, you'd right-click on the value within the node parameter pane, add a keyframe at both the first and the second point, and then use the Curve Editor to edit the various spline and straight line options for the transition. It's easy to see what's happening and get the curves to make any transitions to feel as natural as possible – just as you might automate audio effects in a piece of audio software, to keep the audio analogy.

Natron is a wonderful piece of software. If you've ever played with any kind of video editing to create effects, it's going to provide you with all kinds of potential, and there's enough fine-grained control to produce professional results. That a project like this is open source is fantastic, and hopefully, the developer can earn enough from selling support that it remains at the cutting edge of video processing throughout the entire 3.x release cycle.

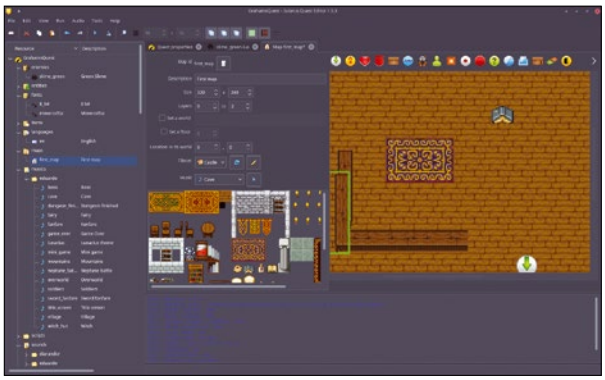
**Project Website**
https://natron.fr

ARPG game engine

# Solarus

When you take a look at Solarus, it's clear where its inspiration comes from – Nintendo's classic *Legend of Zelda: A Link to the Past*. This amazing game has shaped so much of what came after, it's difficult to know where to start. Its oblique 2D graphics, storyline, combat, and overall aesthetic are still the blueprint for many games, even when they've moved to 3D game engines and added social networking features. But there's a considerable following for the original style, too, with even new games being produced that stick close to that pixilated template. This is obviously an area ripe for community involvement. Solarus is both a game engine and an editor that enables anyone to create games like these, and that's exactly what its community of users have been doing.

Before you get started, though, you'll need some assets. The original Zelda assets obviously can't be redistributed, but attempts have been made to fill in the missing files (the data files that Solarus expects to find in a specific location), and you can always draw your own or hook a few artistic friends into the project. The editor itself is really easy to use, combining media browsing with map editing and the Lua code required to bind everything together. There are several examples you can use to help with your own project, and creating a game like this is very effective, especially if you enjoy playing these kinds of games as it opens an entirely fun and new aspect to the genre. The project includes a graphical app that's used to manage your installed games, the quest editor, and the `run` command for launching quests, and the games created by Solarus can be made to run on anything from Android to Amiga!



A complete game, *The Legend of Zelda: Mystery of Solarus DX*, has already been created and made real with Solarus.

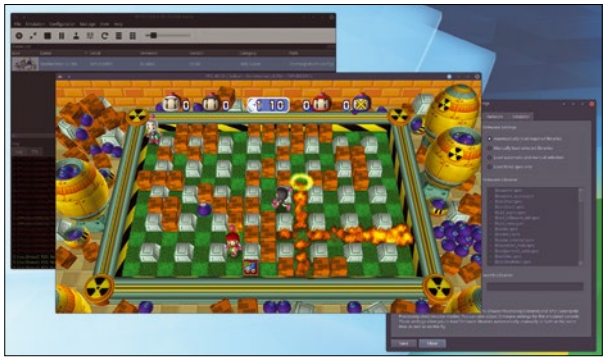
**Project Website**
http://www.solarus-games.org/

PS3 emulator

# RPCS3

Emulators are amazing. They keep old consoles and computers alive when the companies that built them have either mismanaged themselves into oblivion or moved on to "freemium" titles for smartphones. Although it's difficult to believe that Sony's remarkable PlayStation 3 (PS3) is now over a decade old, it's not so difficult to guess why its hardware has yet to be adequately emulated. The venerable PS3 uses an esoteric CPU, consisting of one PowerPC-based core alongside eight Synergistic Processing Elements (SPEs) and its GPU. This architecture makes the PS3 a particular challenge to emulate, especially on x86 hardware, but it's beginning to look like it may finally be possible.

RPCS3 is still very much under development, despite that development starting in 2011, but it's getting tantalizingly close to being usable. In fact, it's getting so close that its developers claim just over 12 percent of all PS3 games can now be played through on the emulator! Because the current download is a `.AppImage`, it can be run in place, and you'll also need to download Sony's firmware separately. Fortunately, you're spared having to rip this from your console. Just point the emulator at one of Sony's official updates, and it installs the firmware itself. After that, you'll need to use your own games to get up and running. RPCS3 runs remarkably well – well enough to actually play some of your old favorite games, and even some newer ones. Thanks to Vulkan support for GPU drivers and a huge plethora of recent performance improvements – notably rewriting the vertex GPU processing – many games can run smoothly on a modern system, and there are plenty of options to squeeze the most out of your hardware. You can also configure your keyboard to act as a games controller if you don't have one handy. RPCS3 is incredibly clever and fast; more fundamentally, it works.



Emulating the PlayStation 3 was long considered impossible, but RPCS3 proves the doubters wrong with 90fps!

**Project Website**
https://rpcs3.net/blog/

# Expand Your Command Line with moreutils

## Upgrade your Bash sessions with extra features and power.

BY BEN EVERARD

Way back in the distant past (the 1970s), a group of programmers at Bell Labs created the first version of Unix. This operating system came with a set of utilities to help use the shell-driven interface. Those utilities proved incredibly useful, and we still have them today. Things like `ls`, `rm`, and `cat` are all descendants of the first pieces of Unix software, and they've changed remarkably little over the years. The GNU versions found on most Linux systems have more features than their ancestors, but the basic functionality remains the same.

These utilities have remained fairly static, because they stick to the basic Unix philosophy of "do one thing well." When you do one thing, there's far less to change or optimize. For this to work well, though, you need enough different tools that each does one thing well. While there's new Linux software being created all the time, there's surprisingly little of the sort of utility software that makes it easy to build powerful commands. In this tutorial, I'm going to look at the work of one project looking to change that: *moreutils*. Essentially, this project is just looking to expand the basic set of utilities. You should find it in your package manager (probably in a package called *moreutils*), or you can download it directly from the project website [1].

The aim of each of the utilities is to do just one thing well, so none of them are particularly complicated to use, and each utility has a well-written man page for guidance (Figure 1). The first of the commands I'll look at is `combine`, which takes two sets of input and combines them using a single logic rule to form the output.

For example,

```
combine file1 and file2
```

outputs every line in `file1` that is also in `file2`, whereas

```
combine file1 not file2
```

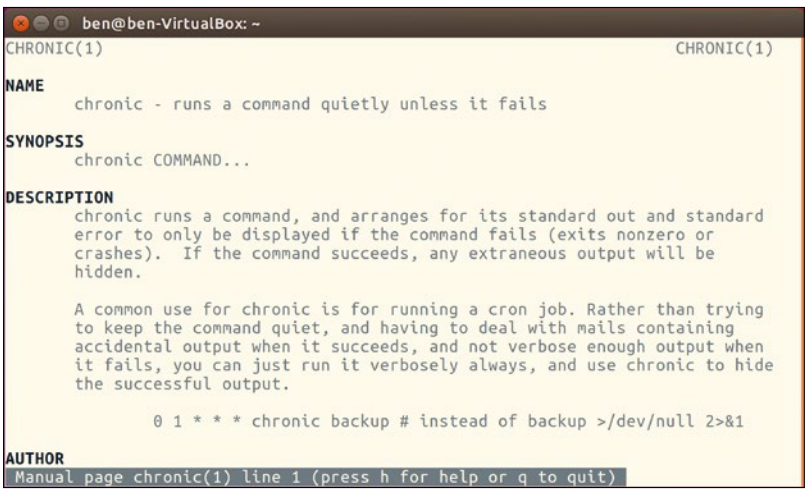
outputs every line that's in `file1` but not in `file2`. Other options are `or` and `exor`. You can also replace a filename with a "`-`" to get input from stdin, which makes it particularly useful for whitelisting (or blacklisting) output from a particularly verbose command. For example, run the command once and send the contents to a file called `file1`. Run it a second time with

```
| combine -- not file1
```

and you'll just get the output that's different from the first time you ran it.

The next command I'll look at is `pee`. OK, take a moment to snigger at the name, and then I'll move on to what it does. The name comes from the fact that it works a little like `tee`, but for processes; since the `tee` command isn't that common, I'll just ignore explaining that for now. Basically, `pee` takes stdin and sends it to more than

**Figure 1:** All the utilities in *moreutils* have well-written man pages to help you out if you forget how to use them.



one command. A ridiculously simple example is this command:

```
echo "hello" | ↵
   pee cat cat cat
```

The result is *hello* printing three times. Note that all this output is in stdout, so the following (wholly useless command) only outputs *hello* once:

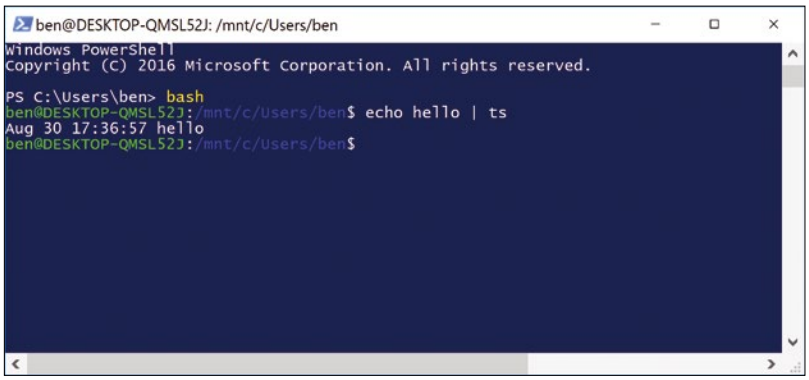



**Figure 2:** If you can't install Linux on your machine, Windows Subsystem for Linux lets you use *moreutils* (and, indeed, any other Linux command-line tool).

```
echo "hello" | pee cat cat cat | uniq
```

Have you ever left a long-running command only to come back to it and had no idea when the last line of output printed? Or made some tweaks to some settings and wanted to know what effect they had on the time it took between two lines of output? Well, thanks to `ts`, that's easy! This command does one very little thing that proves to be surprisingly useful – it appends a timestamp to every line in stdin. For the above examples, all you would need to do is pipe the commands to `ts`, and you'll be able to see exactly when each line of output reached `ts`. It's like logging for lazy people. The `-i` option outputs the time since the previous line of stdin; this is useful for profiling changes to settings in software.

Most standard Unix commands work with text files. Some have the additional ability to work with zipped text files, but not all do. In fact, why should command-line tools come with the ability to work with zipped files? That is, after all, against the basic principle of doing one thing well. `zrun` is the solution. It's a tool that does just one thing: Unzips a file to a temporary file and then runs a command with that temporary file. That all sounds a little more complicated than it actually is, so I'll look at an example. If you have a zipped file, `hello.txt.zip`, you can `cat` it with the command:

```
zrun cat hello.txt.zip
```

With `zrun`, you can run any Linux command with zipped files.

There are some commands that you run all the time, and most of the time they work. You want them to just get on with their job and not spam you with information about what's going on. I'm thinking of things that go in cron jobs or systemd timer units. However, every once in awhile, they'll break, and then you want to know everything that happened. The old fashioned way of dealing with this situation was either to direct all the output to `/dev/null` and cross your fingers or to write the output to a logfile and just deal with the fact that most of the data there was pointless. The `chronic` utility solves this problem. By default, it'll just run a command and drop all of the output. However, if the command fails, it will send everything to stdout. In this way, you can run commands in such a way that they output all the details that might be useful, without having to worry about it clogging up logfiles, but still have the information available if you need it. Just run it like this:

```
chronic <command>
```

I've taken a look at just some of my favorite utilities in *moreutils*, but I haven't covered all of them (see the "Even More Utils" box). In fact, the team behind this software is still on the lookout for more commands to help bolster their collection of tools that set out to make our lives easier. ■■■

## Even More Utils

*moreutils* isn't the only source of new command-line tools. Here are a few more of my favorites:

- `jq`: works as a complicated, but very powerful command-line JSON parser
- `pv`: views the progress of data through piped commands
- `autossh`: automatically reconnects `ssh` connections and tunnels after network disruptions or timeouts
- `tmux`: runs multiple terminal sessions inside a single window

If you occasionally have to work on a Windows machine, you can use any of these commands with Windows Subsystem for Linux [2] (Figure 2).

## Info

[1] moreutils: *https://joeyh.name/code/moreutils/*

[2] Windows Subsystem for Linux: *https://msdn.microsoft.com/en-us/commandline/wsl/install_guide*

# COBOL: The language that Refuses to Die

## Despite being more than half a century old, COBOL is still in use. Explore this fascinating old-school language and see how it ticks.

BY MIKE SAUNDERS

New programming languages pop up all the time. In recent years, there's been plenty of hype around Rust, Go, Swift, Clojure, and many others – and often for good reason. These languages have their own plus points and useful features, and many of them are maturing well. Despite hyperbolic claims from certain overzealous fans, however, none of these languages are going to replace C, C++, or Java completely any time soon. Sure, those languages are old and have their limitations, but they're extremely well established, and rewriting large codebases in the current hot language du jour is a mammoth task.

Although C dates back to the early 1970s, there's an even older language that's still in use – albeit to a much lesser extent. COBOL, the common business-oriented language, was created in 1959 by the US Department of Defense (Figure 1) as a portable language that could be used to process data across many different machines and architectures. The language was standardized a few years later, although there are many different dialects. The most recent update to the language specification was COBOL 2014.

Why is such an ancient language still in use today? COBOL doesn't look pretty, but it's quite easy for mere mortals to write (as you'll see in a moment) and is especially suited for data and transaction processing. Indeed, many banks still use COBOL extensively – not on 1960s or 1970s hardware, of course, but on modern mainframes that are purposely built for the task. IBM still makes plenty of bucks selling big-iron hardware running COBOL [1] (Figure 2).

Sure, some companies are slowly migrating from COBOL to other languages, but recent surveys suggest many billions of lines of COBOL are still being used in production. COBOL coders can earn good money, too; even though jobs based around this language aren't widespread, experts can be sure of long-term work prospects maintaining old (but functioning) codebases. (See the "Does COBOL Have a Future" box.)

In this tutorial, I'll explore some of the features of COBOL and show you how the language works. Even if you never intend to write another line of COBOL again, it's well worth exploring the language to see how it influenced other languages. Plus, you get extra geek bragging

**Figure 1:** US Navy Rear Admiral Grace Hopper had a huge influence on modern computing and helped create COBOL.


**Figure 2:** Many companies are still using thousands (or even millions) of lines of COBOL, often on big-iron mainframes.

rights and can tell those darn kids with their fancy new meme languages to get off your lawn. Let's go!

## Installing COBOL

I mentioned IBM COBOL before, and if you look around the net for other commercial COBOL implementations, you may become light headed when you see the prices. Usually the COBOL vendors don't even list them, preferring you to contact them for a sales quote, but they're not cheap: You're looking at minimum $500 for a single developer license and many thousands of dollars for enterprise-level solutions. That's not especially surprising, given that these tools are targeted at banks and other large organizations, but it's not ideal for us.

Fortunately, there's an open source implementation: GnuCOBOL (formerly known as OpenCOBOL) [2]. In some distros, it's still provided under the name OpenCOBOL, so in Debian and Ubuntu systems you can get it with this command in a terminal window:

```
sudo apt-get install open-cobol
```

For other distros, open your package manager and search for COBOL, and you should find it. Once installed, it's time to test it! To write your very first COBOL program, create a file called `test.cbl` in a plain text editor (Figure 4) with the following content:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TEST-PROGRAM.

PROCEDURE DIVISION.
    DISPLAY 'Ciao!'.
    STOP RUN.
```

Not the prettiest language ever, is it? Actually, you can switch the uppercase characters here to lowercase to make it slightly friendlier to the eyes; however, I'm sticking with the traditional formatting, as you'll see in code written decades ago.

This program doesn't need much explanation; you can see that it prints "Ciao!" on the screen and then ends. COBOL programs are split into divisions, as you can see here, with the `IDENTIFICATION` division providing some information about the program, and the `PROCEDURE` division containing the code itself. You'll also note that each line ends with a full stop (period). This looks rather bizarre, and obviously this approach wasn't adopted by other programming languages, but you can see how the language designers thought it'd make the code look more like natural human language.

Compile your `test.cbl` file into an executable binary like so:

## Does COBOL Have a Future?

The answer is yes. Despite being an ancient language, COBOL hasn't stayed completely still (Figure 4). Developers are working on new features and syntax for the language, and we expect to see some more standards updates over the next 10 or 20 years. That sounds like a very long time, indeed, but bear in mind that lots of COBOL code written in the 1970s and 1980s is still being used in production today.

Large organizations are pretty conservative when it comes to major changes, and while devices and interfaces change rapidly (think of the big switch to mobile devices and web-based interfaces in recent years), COBOL will still be grinding away in the background, churning through data for many years to come. I think there's something quite reassuring about that in this hectic world.


**Figure 3:** Have a search on Amazon, and you can see that books are still being written about COBOL, such as this one from Apress in 2014!

```
cobc -free -x -o test test.cbl
```

The `-free` flag tells GnuCOBOL to be more lenient when parsing code (rather than requiring extremely precise formatting), and `-x` tells GnuCOBOL to generate an executable file. Afterward, you have a binary called `test` that you can execute with:

```
./test
```

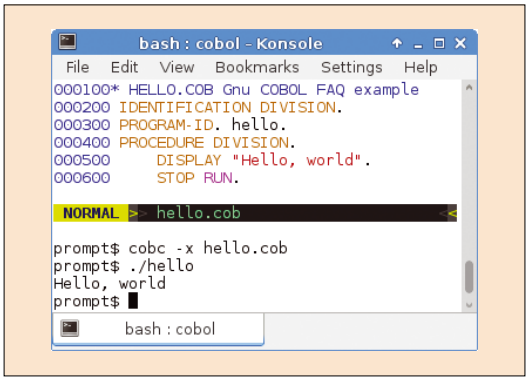



**Figure 4:** The Vim editor has a syntax highlighting scheme for COBOL code to make it easier on the eyes.

There you have it! Your COBOL coding career has begun.

### Working with Data

Next, you can create a slightly more complex program, with some variables. In COBOL, you have to declare variables in the `DATA` division and be very specific about their size and capabilities (e.g., Listing 1).

Think about what happens here – if you've done some programming before, you've probably guessed that it prints *Ciao* followed by the number *123*. Well, that's almost right, but there's quite a bit going on here.

First of all, the `DATA` division sets up a subsection for "working storage" – that is, for the variables. You define two of these, starting each time with the level number – in both cases `01`. You can use levels to split up and organize data, but I won't go into that now. This program is pretty simple, so you can just use the top level for your variables.

**Listing 1:** COBOL Program with Variables

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TEST-PROGRAM.

DATA DIVISION.
    WORKING-STORAGE SECTION.
    01 WS-NUM1 PIC 9(3).
    01 WS-NAME1 PIC X(10) VALUE 'Ciao'.

PROCEDURE DIVISION.
    DISPLAY WS-NAME1.
    SET WS-NUM1 TO 123.
    DISPLAY WS-NUM1.
    STOP RUN.
```

**Listing 2:** COBOL Example Using Paragraphs

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TEST-PROGRAM.

DATA DIVISION.
    WORKING-STORAGE SECTION.
    01 WS-NAME1 PIC X(20).
    01 WS-NUM1 PIC 9(2).

PROCEDURE DIVISION.
    A-PARA.
    DISPLAY 'Enter your name:'.
    ACCEPT WS-NAME1.
    PERFORM B-PARA UNTIL WS-NUM1=10.
    STOP RUN.

    B-PARA.
    DISPLAY 'Hello 'WS-NAME1.
    ADD 1 to WS-NUM1.
```

Next, you provide names for the variables: `WS-NUM1` and `WS-NAME1`. You can use other names here, providing they don't clash with existing COBOL keywords, but it's common practice to precede them with `WS-`. Next, the `PIC` provides a picture of the storage required for the variable. In the first case, `9(3)` means numeric data of a maximum of three bytes, and in the second variable, `X(10)`, means alphanumeric data with a maximum of 10 bytes (i.e., a string variable, effectively). Additionally, the value for the second variable is set immediately.

Yes, it may look ugly and fiddly – especially when compared with more modern programming languages – but bear in mind the extremely limited specifications of computers running COBOL back in the 1960s and 1970s. Every byte was important, and it was essential to be very specific about the type of data that could be stored to avoid problems later on. The `PROCEDURE` division displays the contents of `WS-NAME1` and then sets the value of `WS-NUM1` and displays it as well.

So far so good – now for a look at input, paragraphs, and loops. In COBOL, paragraphs let you split your code into smaller chunks, much like functions or subroutines in other languages. These paragraphs live in the `PROCEDURE` division; Listing 2 shows an example of these paragraphs in use in a program that gets the user's name and then prints a greeting message 10 times.

In Listing 2, the `DATA` division contains two variables, as in the previous example. Then, the `PROCEDURE` division begins with a new paragraph called `A-PARA`. Using COBOL's `ACCEPT` keyword, the program gets a string from the user and stores it in the `WS-NAME1` variable.

Then some magic starts a loop by telling COBOL to `PERFORM` (execute) the code in the `B-PARA` paragraph underneath, again and again, until the `WS-NUM1` variable contains `10`. After that, the program stops. Now look in `B-PARA`: You can see that it prints *Hello* followed by the contents of the string variable, and then it increments the `WS-NUM1` variable by one. Once execution of `B-PARA` is complete, execution continues back in the `PERFORM` line in the previous paragraph, so `B-PARA` is executed 10 times.

### Strings, Conditionals, and Modules

As you'd expect from a language designed for data processing, COBOL has plenty of ways to handle strings, and as you've probably come to expect from the previous code examples, they're

**Listing 3:** Counting Characters

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TEST-PROGRAM.

DATA DIVISION.
    WORKING-STORAGE SECTION.
    01 WS-NUM1 PIC 9(2).
    01 WS-STRING1 PIC X(256).

PROCEDURE DIVISION.
    DISPLAY 'Enter some text:'.
    ACCEPT WS-STRING1.
    INSPECT WS-STRING1 TALLYING WS-NUM1 FOR
       ALL 'a'.

    IF WS-NUM1 = 0 THEN
        DISPLAY 'No letters a in the string.'
    ELSE
        DISPLAY 'Number of a letters: 'WS-NUM1
    END-IF.

    STOP RUN.
```

not always the prettiest – but they work! So now I'll look at how to count the number of specific characters in a string and then use conditionals to print a message. This program asks the user to input some text and then counts the number of "a" characters in the string. If there are no characters, it prints one message – but if some "a" characters can be found, it displays the exact count (Listing 3).

In this code example, I create a numeric and a string variable in the `DATA` division, as before. The `PROCEDURE` part uses `ACCEPT` to get the user to input some text and then uses the `INSPECT` command to analyze the string. You can do various things with `INSPECT`; in this case, `TALLYING` counts all instances of the letter "a" that are stored in the variable. Then, this number is placed into the `WS-NUM1` variable.

Next up is a conditional block. If you've ever programmed in BASIC, this structure will be instantly recognizable to you – the `IF` does a comparison, and if that's true, the code after `THEN` is executed. However, if the comparison doesn't match, the `ELSE` code is executed instead. In this program, if `WS-NUM1` contains zero, it prints one string; if it contains a larger number, however, it prints something different. Note that these `IF` blocks have to be terminated with an `END-IF` statement.

### Listing 4: test.cbl

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TEST-PROGRAM.

DATA DIVISION.
    WORKING-STORAGE SECTION.
    01 WS-NUM1 PIC 9(2) VALUE 10.

PROCEDURE DIVISION.
    DISPLAY 'In test program!'.
    DISPLAY 'Value of WS-NUM1: 'WS-NUM1
    DISPLAY 'Now calling extra...'.
    CALL 'extra'.
    DISPLAY 'Value of WS-NUM1: 'WS-NUM1
    STOP RUN.
```

### Listing 5: extra.cbl

```
IDENTIFICATION DIVISION.
PROGRAM-ID. EXTRA.

DATA DIVISION.
    LINKAGE SECTION.
    01 WS-NUM1 PIC 9(2).

PROCEDURE DIVISION.
    DISPLAY 'Hello from extra!'.
    MOVE 20 TO WS-NUM1.
    EXIT PROGRAM.
```

Finally, I'll take a look at modules. As in most programming languages, when you're working on a large project in COBOL, the code will be split up into many different files, and you don't want to recompile the whole caboodle every single time you make a change – just the file that has been changed. To see this in action, first create a file named `test.cbl` (Listing 4), and then a file called `extra.cbl` (Listing 5).

First, compile `extra.cbl`:

```
cobc -free -m extra.cbl
```

Note the `-m` flag here, which tells the COBOL compiler that you want this file to be turned into a module, rather than a standalone executable. Second, compile `test.cbl` (Listing 4) as you usually would. Finally, run `test`; you'll see from the output that `WS-NUM1` is initially set to `10`, but after the program has jumped into the `EXTRA` module, it is set to `20`. If you have problems running this code, try the following beforehand:

```
EXPORT COB_LIBRARY_PATH=↵
  <path to where your COBOL source files are>
```

How does this program work? Most of `test.cbl` is easy to understand by now, but the magic takes place in the `CALL` line. This hands over control to the compiled `EXTRA` module. Note the `LINKAGE SECTION` line in the `DATA` division of `extra.cbl`. This line tells COBOL that you don't want to create brand new variables here, but instead use the ones from the calling program – that is, the ones defined in `test.cbl`.

In the `PROCEDURE` division, set the value of `WS-NUM1` to 20, instead of 10 as it was originally, and you'll also see that the module ends with `EXIT PROGRAM` rather than `STOP RUN`, because the latter halts execution completely. You don't want to do that, of course, you just want to hand control back to the calling program!

So those are the basics of COBOL. There's much more to the language, of course, so check out the GnuCOBOL Guides page [3] for more reading material. Oh, and if you end up getting a lucrative job in a bank, making megabucks from maintaining old COBOL codebases, don't forget about the tutorial that got you started in this career. At least send us a beer. Thanks. ■■■

### Info


[1] IBM and COBOL: *https://www.ibm.com/us-en/marketplace/ibm-cobol*

[2] GnuCOBOL: *https://sourceforge.net/projects/open-cobol/*

[3] GnuCOBOL Guides: *https://open-cobol.sourceforge.io*

# 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*.

## Linux Kernel Summit

**Date:** October 24–27, 2017

**Location:** Prague, Czech Republic

**Website:** *http://events.linuxfoundation.org/events/linux-kernel-summit*

The annual Linux Kernel Summit brings together core kernel developers to discuss the state of the existing kernel and plan the next development cycle. New in 2017 are four days of sessions and workshops opened to a larger group of developers, along with the half-day, invitation-only Maintainer Summit.

## LISA17

**Date:** October 29–November 3, 2017

**Location:** San Francisco, California

**Website:** *https://www.usenix.org/conference/lisa17*

LISA17, "where systems engineering and operations professionals share real-world knowledge about designing, building, and maintaining the critical systems of our interconnected world," addresses the overlap and differences between traditional and modern IT operations and engineering.

## W-JAX

**Date:** November 6–10, 2017

**Location:** Munich, Germany

**Website:** *https://jax.de/en/*

The W-JAX conference bills itself as "the conference for Java, architecture, and innovation." Join the experts, as they share their professional experiences in sessions and power workshops focusing on Core and enterprise Java, big data, containers, JavaScript, continuous delivery, DevOps, agile planning, security, and more.

## EVENTS

| | | | |
|---|---|---|---|
| **Sylius Hackathon Nürnberg** | October 13–15 | Nuremberg, Germany | https://www.xing.com/events/sylius-hackathon-nurnberg-1838167 |
| **Ubucon 2017** | October 13–15 | Wolfsburg, Germany | https://www.ubucon.de/2017/ |
| **Heise Cloud Conference** | October 17 | Cologne, Germany | https://www.heise-events.de/cloudkonf |
| **DevOpsDays Berlin** | October 18–19 | Munich, Germany | https://www.devopsdays.org/events/2017-berlin/welcome/ |
| **OSAD - Open Source Automation Day** | Ocotber 19 | Munich, Germany | http://www.osad-munich.org/ |
| **All Systems Go!** | October 21–22 | Berlin, Germany | https://all-systems-go.io/ |
| **All Things Open** | October 23–24 | Raleigh, North Carolina | https://allthingsopen.org/ |
| **Open Source Summit Europe** | October 23–25 | Prague, Czech Republic | http://events.linuxfoundation.org/events/open-source-summit-europe |
| **Embedded Linux Conference Europe** | October 23–25 | Prague, Czech Republic | http://events.linuxfoundation.org/events/embedded-linux-conference-europe |
| **WebTech Conference** | October 23–27 | Munich, Germany | https://webtechcon.de/ |
| **heise devSec** | October 24–26 | Heidelberg, Germany | https://www.heise-devsec.de/ |
| **EclipseCon Europe** | October 24–26 | Ludwigsburg, Germany | https://www.eclipsecon.org/europe2017/ |
| **Linux Kernel Summit** | October 24–27 | Prague, Czech Republic | http://events.linuxfoundation.org/events/linux-kernel-summit |
| **Mesoscon Europe** | October 25–27 | Prague, Czech Republic | http://events.linuxfoundation.org/events/mesoscon-europe |
| **LISA17** | Oct. 29–Nov. 03 | San Francsco, California | https://www.usenix.org/conference/lisa17 |
| **API Strategy & Practice** | Oct. 31–Nov. 02 | Portland, Oregon | http://events.linuxfoundation.org/events/apistrat |
| **OpenRheinRuhr** | November 4–5 | Oberhausen, Germany | http://openrheinruhr.de/ |
| **W-JAX** | November 6–10 | Munich, Germany | https://jax.de/ |
| **SPTechCon** | November 12–17 | Washington, DC | http://www.sptechcon.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.*

**NOW PRINTED ON** recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

## AUTHORS



## CONTACT INFO


**Editor in Chief**
Joe Casad, jcasad@linux-magazine.com

**Managing Editor**
Rita L Sooby, rsooby@linux-magazine.com

**Localization & Translation**
Ian Travis

**News Editor**
Swapnil Bhartiya

**Copy Editor**
Amy Pettle

**Layout**
Dena Friesen, Lori White

**Cover Design**
Lori White

**Cover Image**
© Dima Sobko, 123RF.com

**Advertising – North America**
Ann Jesse, ajesse@linuxnewmedia.com
phone +1 785 841 8834

**Advertising – Europe**
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 89 99 34 11 48

**Publisher**
Brian Osborn, bosborn@linuxnewmedia.com

**Marketing Communications**
Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
616 Kentucky St.
Lawrence, KS 66044 USA

**Customer Service / Subscription**
For USA and Canada:
Email: cs@linuxpromagazine.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)
Fax: 1-785-856-3084

For all other countries:
Email: subs@linux-magazine.com
Phone: +49 89 99 34 11 67

www.linuxpromagazine.com – North America

www.linux-magazine.com – Worldwide




Printed in Nuremberg, Germany, by hofmann infocom GmbH on recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Distributed by Seymour Distribution Ltd, United Kingdom.

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 616 Kentucky St., Lawrence, KS, 66044, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 616 Kentucky St., Lawrence, KS 66044, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Zieblandstr. 1, 80799 Munich, Germany.

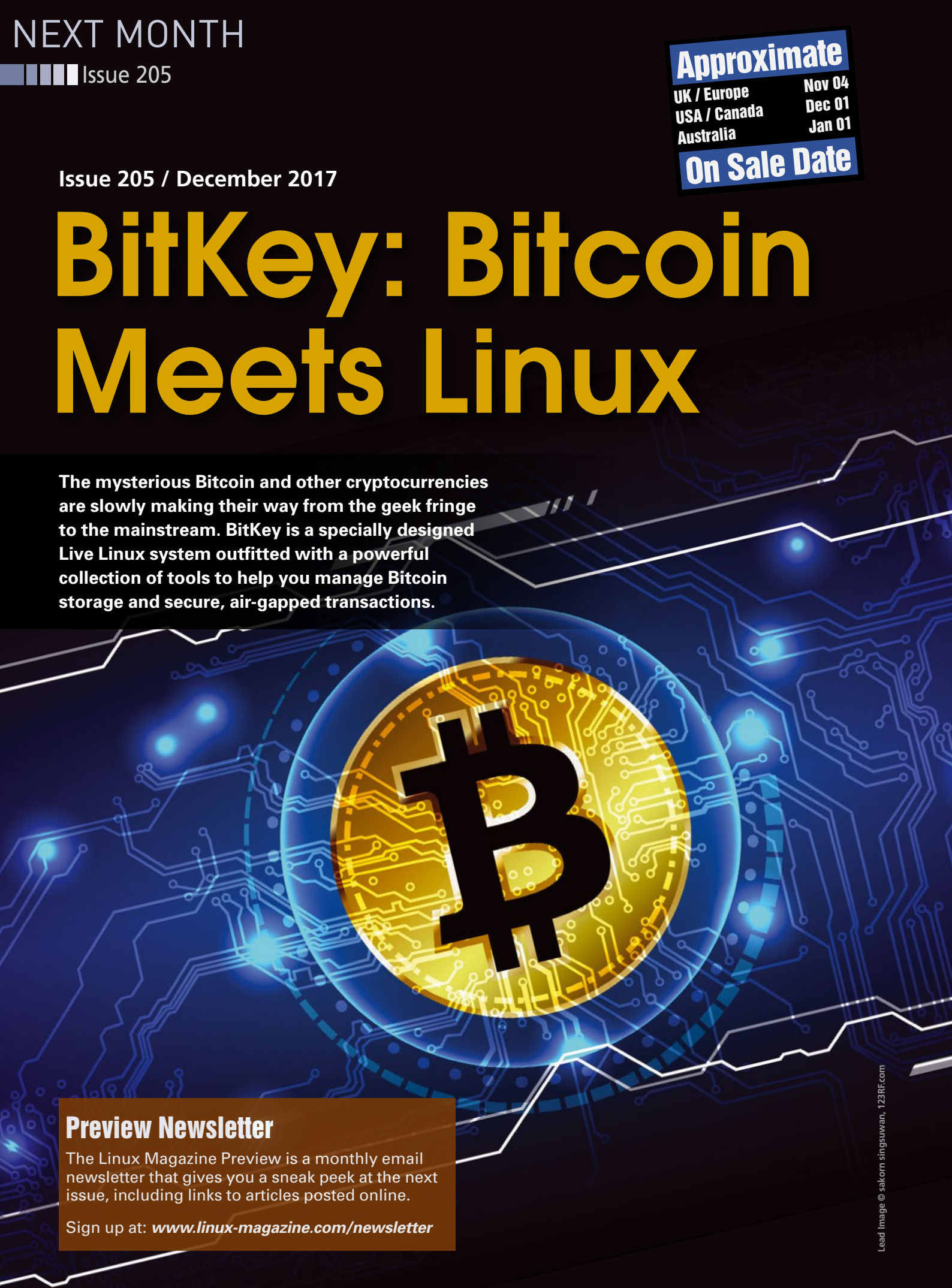
**Approximate On Sale Date**

| | |
|---|---|
| UK / Europe | Nov 04 |
| USA / Canada | Dec 01 |
| Australia | Jan 01 |

Issue 205 / December 2017

# BitKey: Bitcoin Meets Linux

**The mysterious Bitcoin and other cryptocurrencies are slowly making their way from the geek fringe to the mainstream. BitKey is a specially designed Live Linux system outfitted with a powerful collection of tools to help you manage Bitcoin storage and secure, air-gapped transactions.**



Lead Image © sakorn singsuwan, 123RF.com

## Preview Newsletter

The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

Sign up at: ***www.linux-magazine.com/newsletter***

# OPEN SOURCE MONITORING CONFERENCE

November 21 – 24
2017 | Nuremberg

## ADD-ONS

Workshops & Hackathon

## FEATURED SPEAKERS

**Alba Ferri Fitó** | Vodafone

**Caleb Hailey** | Sensu

**James Shubin** | RedHat

**www.osmc.de**

# SUPERMICRO®

# MicroBlade®

Highest-Density • High-Performance • High-Efficiency • Cost-Effective
Enterprise, Data Center, Web Applications, HPC and Cloud Computing Solutions

6U MicroBlade

3U MicroBlade

- Supports Intel® Xeon® Scalable Processors
- Best density and power efficiency with up to 56 UP or 28 DP server nodes
- M.2 NVMe SSD and HW SAS3 RAID support
- Ethernet switches supporting 1G, 2.5G and 10G downlinks with 96% fewer cables
- Redundant 2200W Titanium Level (96% efficiency) digital power supplies
- Battery Backup Power (BBP) modules with integrated power supply and cooling

intel XEON PLATINUM inside

We Power the Cloud

SUPERMICRO X11 SERVER PLATFORMS

TITANIUM

nvm EXPRESS

SUPERMICRO®

*Intel Inside®. Powerful Productivity Outside.*

***Learn more at supermicro.com/X11***