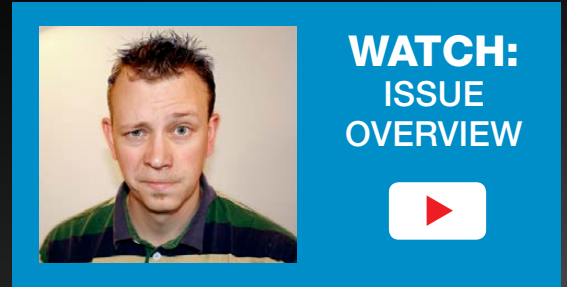


Manage Docker Images and Containers with Puppet

LINUXTM JOURNAL

Since 1994: The Original Magazine of the Linux Community



FEBRUARY 2017 | ISSUE 274
<http://www.linuxjournal.com>

Detect Man-in-the-Middle Cellular Attacks



BEST PRACTICES
for SysAdmin Alerts

POSTMORTEM:
What to Do
After an Attack

Secure Your Accounts
with Two-Factor Authentication

EOF:
Microsoft + Linux?

**Practical books
for the most technical
people on the planet.**

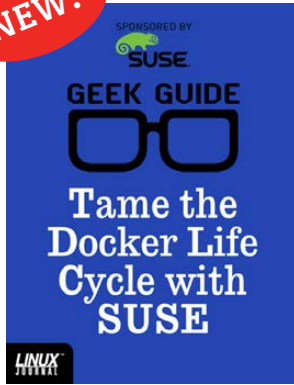
GEEK GUIDES



**Download books for free with a
simple one-time registration.**

<http://geekguide.linuxjournal.com>

NEW!



Tame the Docker Life Cycle with SUSE

Author: John S. Tonello
Sponsor: SUSE



SUSE Enterprise Storage 4

Author: Ted Schmidt
Sponsor: SUSE



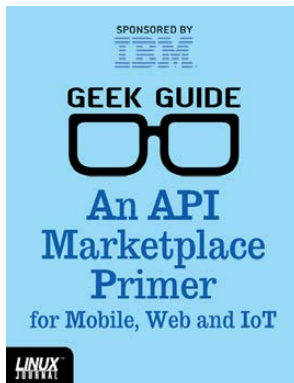
BotFactory: Automating the End of Cloud Sprawl

Author: John S. Tonello
Sponsor: BotFactory.io



Containers 101

Author: Sol Lederman
Sponsor: Puppet



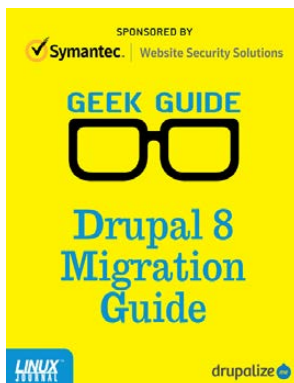
An API Marketplace Primer for Mobile, Web and IoT

Author: Ted Schmidt
Sponsor: IBM



Public Cloud Scalability for Enterprise Applications

Author: Petros Koutoupis
Sponsor: SUSE



Drupal 8 Migration Guide

Author: Drupalize.me
Sponsor: Symantec



Beyond Cron, Part II: Deploying a Modern Scheduling Alternative

Author: Mike Diehl
Sponsor: Skybot

CONTENTS

FEBRUARY 2017
ISSUE 274

FEATURES

74 Cellular Man-in-the-Middle Detection with SITCH

Build your own coordinated GSM anomaly detection system, using inexpensive, easy-to-source parts and open-source software.

Ash Wilson

92 Managing Docker Instances with Puppet

Leverage Puppet roles and profiles, and discover how to target specific Docker configurations on hundreds or even thousands of systems using simple hostname patterns.

Todd A. Jacobs



COLUMNS

32 Dave Taylor's
Work the Shell
Scissors, Paper or Rock?

38 Kyle Rankin's
Hack and /
Sysadmin 101: Alerting

48 Shawn Powers'
**The Open-Source
Classroom**
All Your Accounts Are
Belong to Us

58 Susan Sons'
Under the Sink
Postmortem

114 Doc Searls' EOF
From vs. to + for Microsoft
and Linux

IN EVERY ISSUE

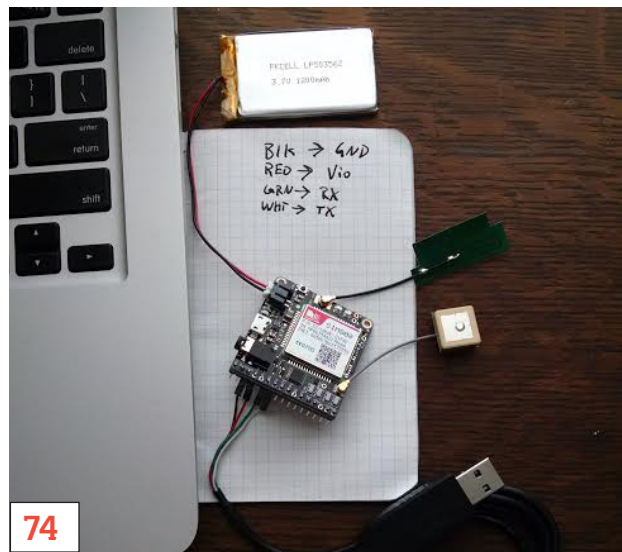
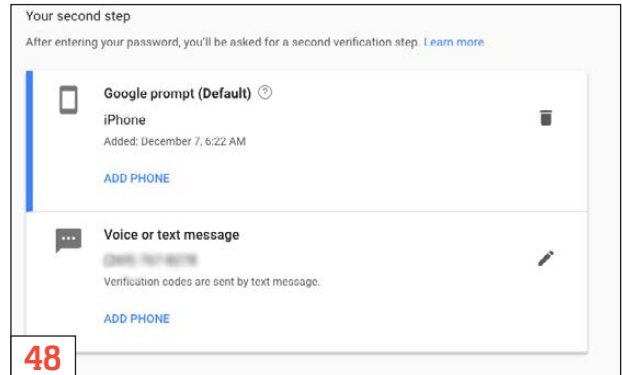
8 Current_Issue.tar.gz

10 UPFRONT

30 Editors' Choice

66 New Products

122 Advertisers Index



ON THE COVER

- Manage Docker Containers with Puppet, p. 92
- Detect Man-in-the-Middle Cellular Attacks, p. 74
- Secure Your Accounts with Two-Factor Authentication, p. 48
- Best Practices for SysAdmin Alerts, p. 38
- Postmortem: What to Do After an Attack, p. 58
- EOF: Microsoft + Linux?, p. 114

LINUX JOURNAL™

Subscribe to
Linux Journal
Digital Edition
for only
\$2.45 an issue.



ENJOY:

Timely delivery

Off-line reading

Easy navigation

Phrase search
and highlighting

Ability to save, clip
and share articles

Embedded videos

Android & iOS apps,
desktop and
e-Reader versions

SUBSCRIBE TODAY!

LINUX JOURNAL

Executive Editor	Jill Franklin jill@linuxjournal.com
Senior Editor	Doc Searls doc@linuxjournal.com
Associate Editor	Shawn Powers shawn@linuxjournal.com
Art Director	Garrick Antikajian garrick@linuxjournal.com
Products Editor	James Gray newproducts@linuxjournal.com
Editor Emeritus	Don Marti dmarti@linuxjournal.com
Technical Editor	Michael Baxter mab@cruzio.com
Senior Columnist	Reuven Lerner reuven@lerner.co.il
Security Editor	Mick Bauer mick@visi.com
Hack Editor	Kyle Rankin lj@greenfly.net
Virtual Editor	Bill Childers bill.childers@linuxjournal.com

Contributing Editors

Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti • Ludovic Marcotte
Paul Barry • Paul McKenney • Dave Taylor • Dirk Elmendorf • Justin Ryan • Adam Monsen

President Carlie Fairchild
publisher@linuxjournal.com

Publisher Mark Irgang
mark@linuxjournal.com

Associate Publisher John Grogan
john@linuxjournal.com

Director of Digital Experience Katherine Druckman
webmistress@linuxjournal.com

Accountant Candy Beauchamp
acct@linuxjournal.com

**Linux Journal is published by, and is a registered trade name of,
Belltown Media, Inc.**

PO Box 980985, Houston, TX 77098 USA

Editorial Advisory Panel

Nick Baronian
Kalyana Krishna Chadalavada
Brian Conner • Keir Davis
Michael Eager • Victor Gregorio
David A. Lane • Steve Marquez
Dave McAllister • Thomas Quinlan
Chris D. Stark • Patrick Swartz

Advertising

E-MAIL: ads@linuxjournal.com
URL: www.linuxjournal.com/advertising
PHONE: +1 713-344-1956 ext. 2

Subscriptions

E-MAIL: subs@linuxjournal.com
URL: www.linuxjournal.com/subscribe
MAIL: PO Box 980985, Houston, TX 77098 USA

LINUX is a registered trademark of Linus Torvalds.

**STORAGE
REDEFINED:**

**You
cannot
keep up
with data
explosion.**

Manage data expansion with SUSE Enterprise Storage.

SUSE Enterprise Storage, the leading open source storage solution, is highly scalable and resilient, enabling high-end functionality at a fraction of the cost.

suse.com/storage



Everything Is Data, Data Is Everything

It doesn't take more than a glance at the current headlines to see data security is a vital part of almost everything we do. Whether it's concern over election hacking or user accounts being publicized after a website compromise, our data integrity is more important than ever. Although there's little we can do individually to stop hackers from attacking websites we don't personally control, we always can be more conscious of how we manage our data and credentials for our own accounts. As is becoming more and more common, this month, we look at a lot of security issues.

Although not exactly security-related, Dave Taylor starts off on another scripting quest. We've been learning how to land on Mars, but this month, we look at how to play rock scissors paper with the command line. It sounds like a simple endeavor, but the programmatic side can become complicated quickly. As is always the case with Dave's column, the objective is fun, but the learning experience along the way is priceless.

This month, Kyle Rankin helps us all sleep a little better at night—not due to better security measures, but rather by helping us configure on-call alerts. Being woken up at 3am because a bird flew into the server room window is



**SHAWN
POWERS**

Shawn Powers is the Associate Editor for *Linux Journal*. He's also the Gadget Guy for LinuxJournal.com, and he has an interesting collection of vintage Garfield coffee mugs. Don't let his silly hairdo fool you, he's a pretty ordinary guy and can be reached via email at shawn@linuxjournal.com. Or, swing by the [#linuxjournal](https://www.freenode.net/channel/linuxjournal) IRC channel on [Freenode.net](https://www.freenode.net).



VIDEO:
Shawn Powers runs through the latest issue.

not a great way to catch 40 winks. Kyle shows how to avoid false positives, but also how to make more intelligent alerts in general. Because servers seldom misbehave during regular business hours, his column is invaluable.

A while back I wrote an article on how to pick smart passwords. I think it was only last year, but in IT time, that was eons ago. Thanks to a recent attempt at compromising my cell phone, security has been on the top of my list recently. Kyle Rankin helped me identify some ways to secure my identity, and I figured it was a good time to elaborate on some general tips on how to keep your credentials and accounts safe. Also, for the record, it's incredibly awesome to have Kyle as a personal friend—just saying.

Susan Sons teaches us to learn from our mistakes and avoid repeating unpleasant history. Specifically, she explains how to go about doing a postmortem on a security issue. Whether it's a practice run, a server-level compromise or even leaked account credentials, the lessons we learn from past problems are only as good as how detailed our postmortem procedures are. Thankfully, Susan is willing to share her expertise, and we can all benefit.

We go into a fairly scary world with Ash Wilson this month. It wasn't very long ago that cellular data services were rather difficult to attack. We've all been conditioned not to trust open Wi-Fi networks, but the cellular connection on our mobile devices isn't something most of us think about. Those times are changing, and Ash helps us learn to detect man-in-the-middle attacks on cellular networks. If you use a mobile device (and if you're reading *Linux Journal*, we all know you are), this article will both inform and scare you. I know it did me!

And finally, Todd A. Jacobs provides a great look into the current DevOps world with his article on managing Docker instances with Puppet. In one of those peanut-butter-in-my-chocolate situations, combining multiple DevOps tools tends to make something better than the sum of its parts. This article builds on Todd's December 2016 article about provisioning Docker with Puppet, and here he describes how to manage Docker images and containers.

This issue certainly has a lot of security-related content, which is great if you live in the current data-centric world. Thankfully, it also contains other tech tips, product announcements and insight on our current technology-rich world. Whether you're looking for a way to deploy a more secure application or just want to learn about the latest cool mobile game, this issue should do the trick. Enjoy!■



PREVIOUS
Current_Issue.tar.gz

NEXT
Editors' Choice



diff -u

What's New in Kernel Development

John Stultz wanted to allow specially privileged processes to migrate other processes between **cgroup** namespaces—essentially migrating processes from one virtual machine to another. This is risky, because one of the whole points of cgroups is to isolate a virtual system and prevent any potentially hostile processes within it from escaping.

John's patch, based on ideas from **Michael Kerrisk**, would allow this process migration if the controlling process had been granted **CAP_SYS_RESOURCE** capabilities.

John explained that this originally had been an **Android** feature created so that people wouldn't have to run their activity manager process with root privileges. John felt his approach was cleaner and more generic.

Kees Cook liked the patch, but **Andy Lutomirski** saw trouble up ahead. He explained:

Developments are afoot to make cgroups do more than resource control. For example, there's Landlock and there's Daniel's ingress/egress filter thing. Current cgroup controllers can mostly just DoS their controlled processes. These new controllers (or controller-like things) can exfiltrate data and change semantics.

At Your Service

Alexei Starovoitov asked if Andy knew a better approach, but Andy said he did not. He was only able to identify the problem, but had no solution to offer. He did, however, identify some constraints that any potential solution would need to adhere to. He said:

1. An insufficiently privileged process should not be able to move a victim into a dangerous cgroup.
2. An insufficiently privileged process should not be able to move itself into a dangerous cgroup and then use `execve` to gain privilege such that the `execve`'d program can be compromised.
3. An insufficiently privileged process should not be able to make an existing cgroup dangerous in a way that could compromise a victim in that cgroup.
4. An insufficiently privileged process should not be able to make a cgroup dangerous in a way that bypasses protections that would otherwise protect `execve()` as used by itself or some other process in that cgroup.

John didn't know where to go with those admonitions, and the project seemed to stall for a few weeks. Finally Andy suggested:

The cgroups interface is a bit unfortunate in that it doesn't really express the constraints. To safely migrate a task, ISTM you ought to have some form of privilege over the task *and* some form of privilege over the cgroup. cgroups only handles the latter.

SUBSCRIPTIONS: *Linux Journal* is available in a variety of digital formats, including PDF, .epub, .mobi and an online digital edition, as well as apps for iOS and Android devices. Renewing your subscription, changing your email address for issue delivery, paying your invoice, viewing your account details or other subscription inquiries can be done instantly online: <http://www.linuxjournal.com/subs>. Email us at subs@linuxjournal.com or reach us via postal mail at *Linux Journal*, PO Box 980985, Houston, TX 77098 USA. Please remember to include your complete name and address when contacting us.

ACCESSING THE DIGITAL ARCHIVE: Your monthly download notifications will have links to the various formats and to the digital archive. To access the digital archive at any time, log in at <http://www.linuxjournal.com/digital>.

LETTERS TO THE EDITOR: We welcome your letters and encourage you to submit them at <http://www.linuxjournal.com/contact> or mail them to *Linux Journal*, PO Box 980985, Houston, TX 77098 USA. Letters may be edited for space and clarity.

WRITING FOR US: We always are looking for contributed articles, tutorials and real-world stories for the magazine. An author's guide, a list of topics and due dates can be found online: <http://www.linuxjournal.com/author>.

FREE e-NEWSLETTERS: *Linux Journal* editors publish newsletters on both a weekly and monthly basis. Receive late-breaking news, technical tips and tricks, an inside look at upcoming issues and links to in-depth stories featured on <http://www.linuxjournal.com>. Subscribe for free today: <http://www.linuxjournal.com/enewsletters>.

ADVERTISING: *Linux Journal* is a great resource for readers and advertisers alike. Request a media kit, view our current editorial calendar and advertising due dates, or learn more about other advertising and marketing opportunities by visiting us on-line: <http://www.linuxjournal.com/advertising>. Contact us directly for further information: ads@linuxjournal.com or +1 713-344-1956 ext. 2.

CAP_CGROUP_MIGRATE ought to be okay. Or maybe cgroupfs needs to gain a concept of “dangerous” cgroups and further restrict them and CAP_SYS_RESOURCE should be fine for non-dangerous cgroups?

But, **Tejun Heo** objected that if CAP_SYS_RESOURCE was disqualified due to overlapping users, it would be better to use a different capability altogether. He suggested:

We can't do it properly on [cgroups] v1 because some controllers aren't properly hierarchical and delegation model isn't well defined. For example, nothing prevents a process from being pulled across different subtrees with the same delegation, but v2 can do it properly. All that's necessary is to make the CAP test OR'd to other perm checks instead of AND'ing so that the CAP just allows overriding restrictions expressed through delegation but it's normally possible to move processes around in one's own delegated subtree.

Tejun went on to explain:

Delegation is an explicit operation and reflected in the ownership of the subdirectories and cgroup interface files in them. The subhierarchy containment is achieved by requiring the user who's trying to migrate a process to have write perm on cgroup.procs on the common ancestor of the source and target in addition to the target.

In other words, it's a completely different approach from the one initially proposed by John.

The discussion ended inconclusively, with the main question remaining whether to use an existing capability or write a new one.

Typically, cgroup features are insane. There are often security issues affecting virtual systems that wouldn't affect the outer running system, forcing Linux to offer only a weird special-cased subset of normal features. And there are also bizarre use cases surrounding various feature enhancements, in which developers want to add functionality to cgroups that would not be desirable in regular Linux. It's all very here-be-dragons and full of magic. Migrating processes between virtual systems will

probably be a lot like that.

Serge E. Hallyn pointed out a security issue with cgroups. He said:

Root in a user [namespace] cannot be trusted to write a traditional security.capability xattr. If it were allowed to do so, then any unprivileged user on the host could map his own uid to root in a namespace, write the xattr, and execute the file with privilege on the host.

The problem was that in the outer system, a user might legitimately do something like that, while on a virtualized system, it was a security hole.

Serge posted a patch to do crazy madness in order to simulate proper behavior on the virtual machine. The patch, he said, “allows a simple setxattr to work, allows tar/untar to work, and allows us to tar in one namespace and untar in another while preserving the capability, without risking leaking privilege into a parent namespace.”

He explained:

When a task in a user ns (which is privileged with CAP_SETFCAP toward that user_ns) asks to write v2 security.capability, the kernel will transparently rewrite the xattr as a v3 with the appropriate rootid. Subsequently, any task executing the file that has the noted kuid as its root uid, or which is in a descendant user_ns of such a user_ns, will run the file with capabilities.

If a task writes a v3 security.capability, then it can provide a uid (valid within its own user namespace, over which it has CAP_SETFCAP) for the xattr. The kernel will translate that to the absolute uid and write that to disk. After this, a task in the writer’s namespace will not be able to use those capabilities, but a task in a namespace where the given uid is root will.

Eric W. Biederman gave a quick look and said the patch seemed strange but correct. He said he’d go over it thoroughly and report back. Meanwhile, Michael Kerrisk asked for some documentation, perhaps in the man pages for **user_namespaces(7)** or **capabilities(7)**, and Serge wrote some up.—Zack Brown

Non-Linux FOSS: a Clippy That Never Forgets

Flycut (Clipboard manager)

By Gennadiy Potapov

Open the Mac App Store to buy and download apps.



[View in Mac App Store](#)

Description

Flycut is a clean and simple clipboard manager for developers. It based on open source app called Jumpcut. Flycut is also open source: <http://github.com/TermiT/flycut>

[Gennadiy Potapov Web Site](#) [Flycut \(Clipboard manager\) Support](#)

[...More](#)

What's New in Version 1.5

- Sticky mode by default disabled
- clips and settings sync with Dropbox
- navigate in sticky mode through clips using J, K keys (vi style)

I hate it when I paste something into a window, only to realize I'd copied something new into the clipboard. I usually end up with eight paragraphs pasted into a login box. To quote my college-aged daughter, the struggle is real.

Thankfully, it's easy to integrate a clipboard manager into OS X. Several options are available, but my favorite happens to be open source. If you head over to <https://github.com/TermiT/Flycut>, you'll find Flycut, which is a clipboard manager that quietly records all your clippings and allows you to paste whichever one you want at any given time. By default, if you want to use Flycut instead of the system clipboard, you press Command-Shift-V instead of just Command-V. A screen overlay lets you scroll through previous clippings, and you double-click on the one you want to paste.

Flycut is a very simple tool, but all the best ones usually are. If you've ever accidentally overwritten your clipboard, you owe it to yourself to download Flycut either from the GitHub page or the Mac App store.—Shawn Powers

SUPERMICRO[®] MARKETPLACE

Powered by Silicon Mechanics



Broad
Selection



Zero
Defects



3-Year
Warranty

**Your Source for
Supermicro Platform Technology**

[Configure Now](#)



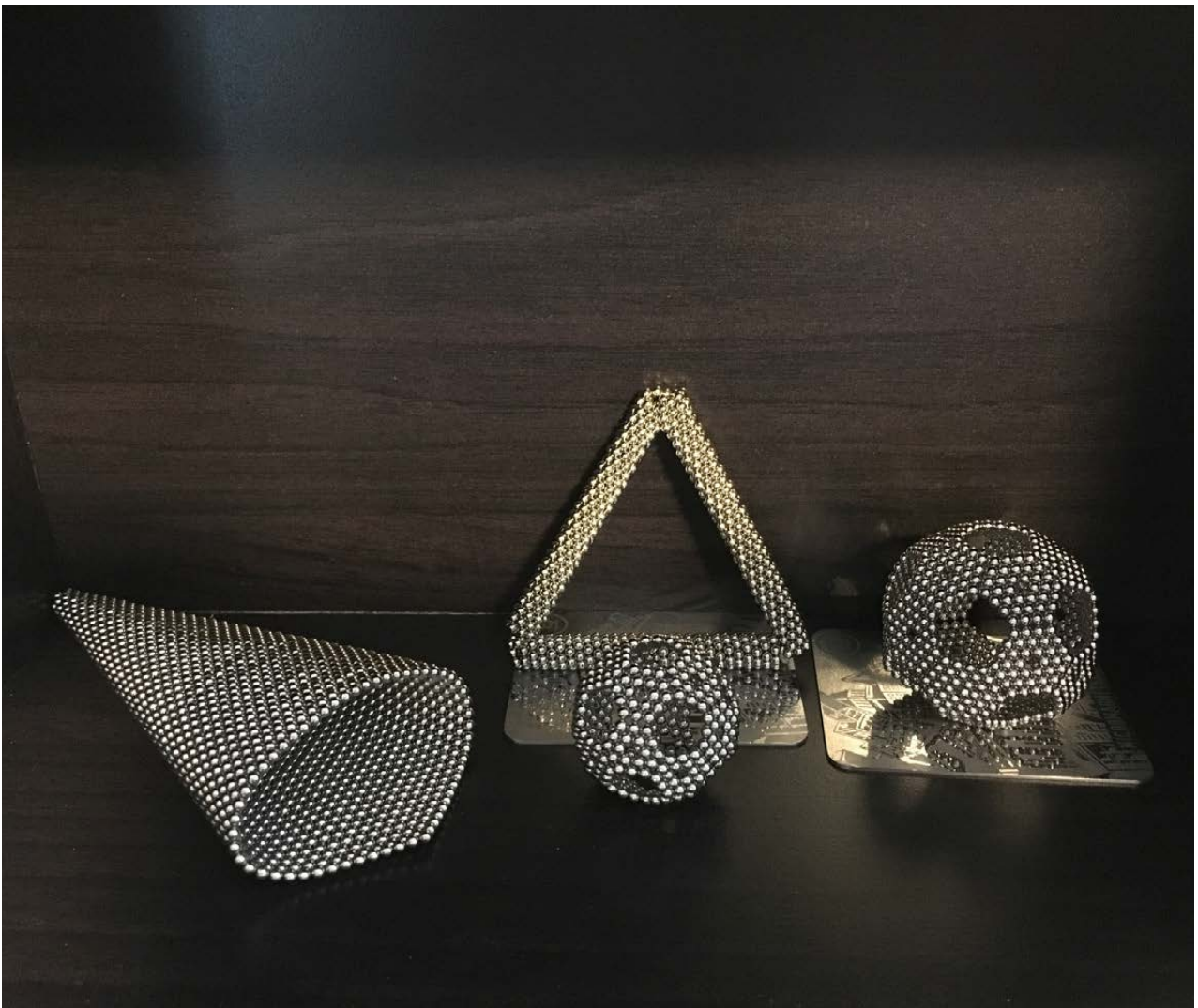
Talk to a Supermicro Expert!

866.352.1173

Getting Sticky with It

Although they might not be so good for credit cards or floppy disks, magnets are one of those things that always have fascinated me. For the past few years, I've wanted to get a set of the round Zen Magnets to play with—they're sort of like an extra science-y version of LEGOs. Unfortunately, before I was able to purchase any, the US government banned their sale!

Recently, the folks at Zen Magnets won their long legal battle

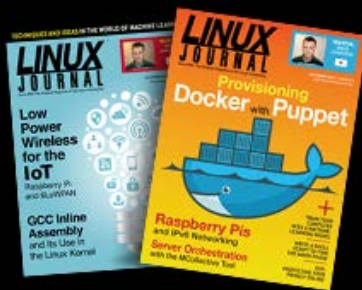


These are what I made last night with my new micromagnets. I can hardly wait for the full-size ones!

and are able to sell tiny, strong magnets again. The regular-size Zen Magnets aren't available yet, but thankfully, production once again can begin. In the meantime, I was able to order "micromagnets" from the same company. They work just like Zen Magnets, but are tinier. I decided to order a couple sets, because I'm impatient and also to support the company who fought the battle allowing magnets to be sold in the US once again.

To read about the legal battle, check out the blog here: <http://zenmagnets.com/magnet-ban-cleared-game-on>. And while you're there, feel free to pre-order some Zen Magnets. I sure did!
—Shawn Powers

LINUX
JOURNAL



LINUX JOURNAL ARCHIVES

Issues 1–272

The First 23 Years of *Linux Journal* (1994–2016)

Archive 1994–2016

NOW AVAILABLE!

SAVE \$10.00
by using
discount code
2017ARCH
at checkout.

*Coupon code
expires 3/28/2017*

www.linuxjournal.com/archive

Get a Haircut, Get a Real Job

I'm often asked about what the latest trends in IT will mean for job hunters. It's interesting for me, because although I haven't actively looked for a job in years, I do create training that helps people get hired every day. So, I figured a few tips for the current job market would be a great way for me to answer lots of emails in one fell swoop. Here it goes.

1) DevOps is no longer magic. For the past two years, if you could put "DevOps" on your résumé, you'd pretty much get hired on principle alone. Lately, DevOps has become a ubiquitous part of IT, and it isn't the special snowflake it used to be. Don't get me wrong; you still need to have DevOps skills on your résumé—just know that it won't get you hired on its own. Instead, mention what sorts of things you have done or can do utilizing DevOps.

2) Security is vital. If you love security, the future looks bright for you. But even if specializing in security isn't what you want to do as a career, it's important to approach every aspect of technology with a security mindset. Twenty years ago, we worried about firewalls, but rarely considered attacks coming from inside our own networks. That was a poor attitude 20 years ago, and now it's technology suicide. Security isn't something you add, it's a way you plan.

3) Developers, developers, developers. Steve Ballmer may have seemed like a crazy man when he shouted it on stage back in 2000, but now that DevOps is a part of everything we do, developer skills are as important as ever. Even the traditional system administrator or operations person will need to have at least rudimentary programming skills in order to function in our DevOps world. Plus, here's a secret: programming is actually kind of fun, especially when it can save you time on the job!

4) Don't forget your roots. In the Pixar movie *WALL-E*, civilization has advanced to the point that everything is automated. It means

life for people is extremely easy, but it also means they don't know how to do anything for themselves. With everything in the data center and the cloud being automated, it's easy to hire an entire team that knows nothing about the actual processes they're automating. That works great—until it doesn't. Make sure you're well versed in the underlying systems (almost always Linux), so when something goes wrong, you know how to fix it.

5) Be a softy! Soft skills (communication skills, cooperation skills and so on) are something we all too often overlook in IT. But not only do soft skills help you in the interviewing process, they also help you in the current IT landscape where various disciplines are working closer than ever. Again, DevOps is much to blame for this blurring of department lines. Any employee who is able to communicate cross-discipline, especially one who is able to communicate with non-IT folks, is going to be invaluable to any organization. Take some communication classes. You might be the only nerd in the room, but you'll also likely have the best job opportunities!—Shawn Powers

THEY SAID IT

Remember that nobody will ever get ahead of you as long as he is kicking you in the seat of the pants.

—Walter Winchell

The great thing about a computer notebook is that no matter how much you stuff into it, it doesn't get bigger or heavier.

—Bill Gates

Security is a kind of death.

—Tennessee Williams

Above all things, never be afraid. The enemy who forces you to retreat is himself afraid of you at that very moment.

—Andre Maurois

There's only one thing I hate more than lying: skim milk. Which is water that's lying about being milk.

—Ron Swanson

Gabedit: the Portal to Chemistry

Many chemistry software applications are available for doing scientific work on Linux. I've covered several here in previous issues of the magazine, and of them have their own peculiar specialties—areas where one may work better than another. So, depending on what your research entails, you may need to use multiple software packages to handle all of the work. This is where Gabedit will step in to help you out.

Gabedit provides a single unified interface to a multitude of chemistry packages available on your system. It should be available within the package management systems for most distributions. For example, on Debian-based systems, you can install it with the command:

```
sudo apt-get install gabedit
```

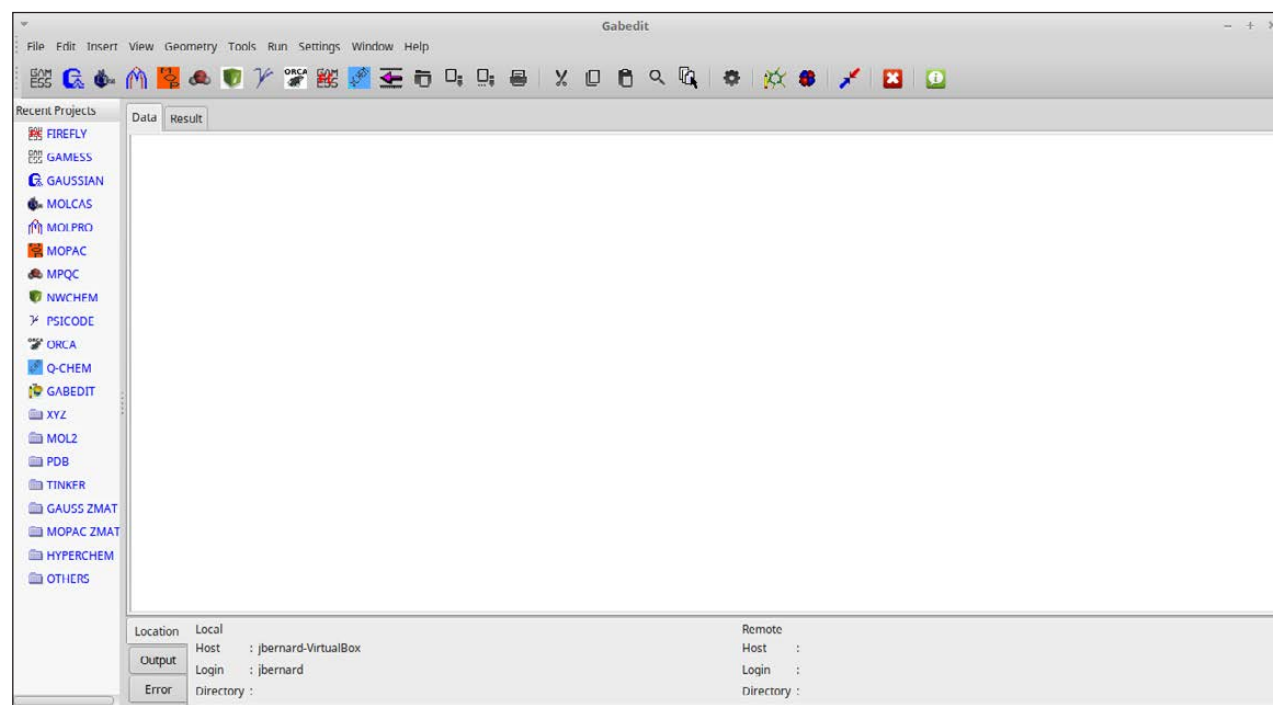


Figure 1. When you first start Gabedit, you'll get an empty project where you can begin your work.

Once it's installed, start it with the `gabedit` command. The very first time you start Gabedit, you'll see a series of windows describing all the data directories that need to be created in order for Gabedit to run. The pane on the left-hand side shows a listing of all the chemistry programs you could use for your work. The central pane provides two tabs, one for input and one for results.

To start working with Gabedit, you need to create a new input file for the software package you want to work with. The icon bar across the top of the window provide buttons for the various types of input files that Gabedit can use. Clicking on one of them will pop up a new window where you can enter parameters relevant for that type of input file. For example, clicking on the first button pops up a window where

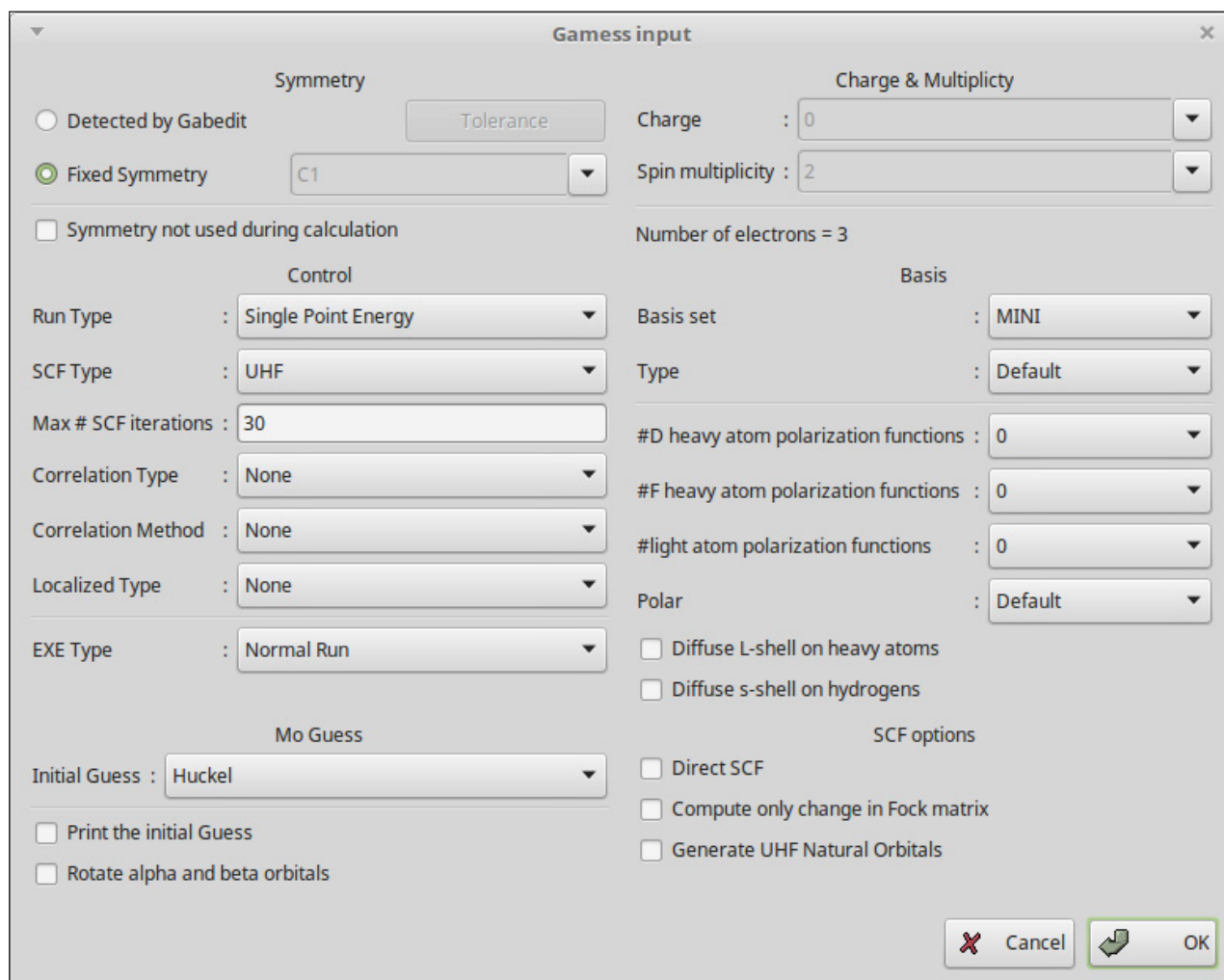


Figure 2. When you create a new input file, a new window pops up where you can enter the initial parameters.

you can create a new input file for GAMESS.

If you try to do this at the beginning of your work, you'll actually get an error. All of these programs depend on some set of atoms, defined as a geometry, in order to do their calculations, which means you need to create this geometry first. Clicking the Geometry menu entry will provide a list of different options for creating a new geometry. The first two are specialized options for Gaussian and Molpro. For this example, let's use the two options at the bottom of the list. The first option pops up a new window where you can select the type of geometry (XYZ, for example) and then create a table of atoms used within your geometry.

Right-clicking inside the table of the geometry editor provides a pop-up menu where you can add a new entry to the table. This allows you to select the element, location and charge for the new point in the geometry. This geometry exists within the memory space of the current project, which means it will be available for other functions within Gabedit.

The other available geometry function is the draw function. You

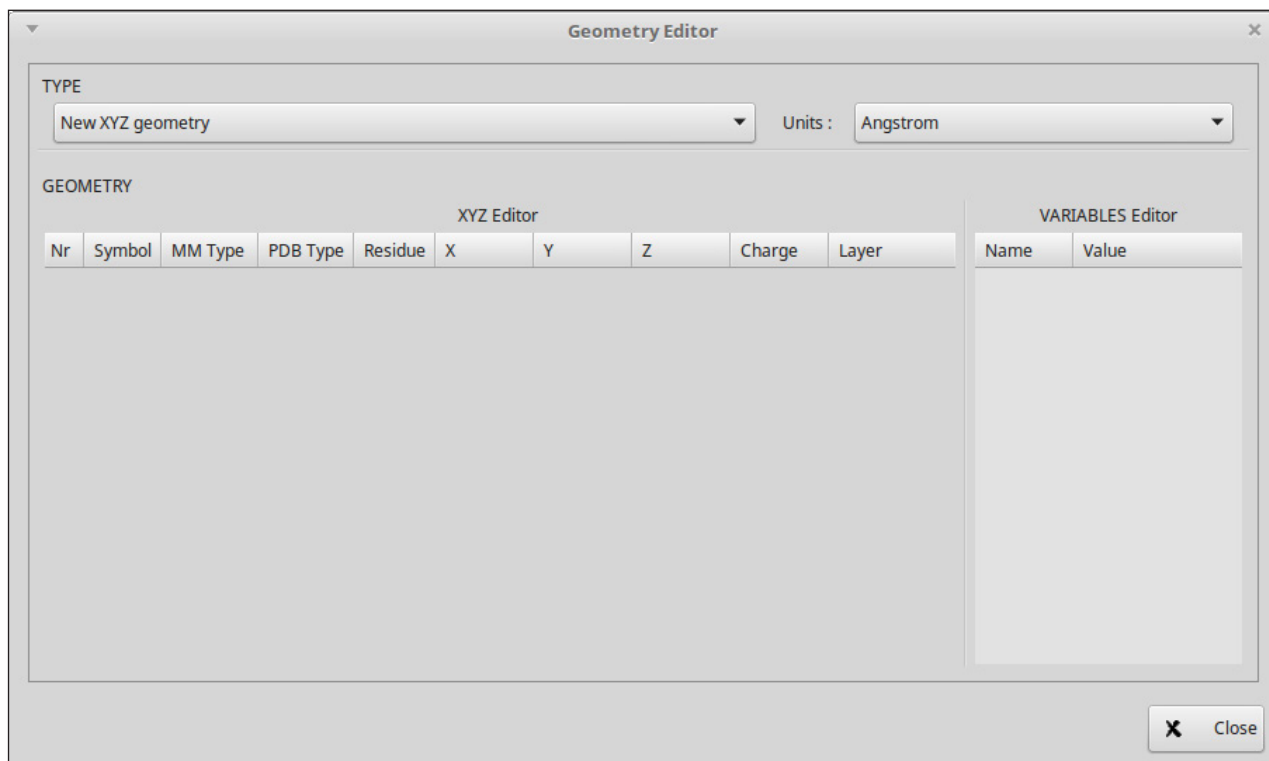


Figure 3. You need to create a new geometry that will be used in the calculations.

can access it via the Geometry→Draw menu item. This pops up a new window where you can visualize your molecule and manipulate it before doing any calculations.

New Center ✕

Atom Symbol :

MM Type : ▼

PDB Type : ▼

Residue Type :

X : ▼

Y : ▼

Z : ▼

Charge :

Layer : ▼

Figure 4. You can add individual elements, setting their chemical properties, to your geometry.

UPFRONT

Here, you can edit the existing geometry and move elements around, or you can add or remove elements to the molecule. You even can add entire functional units, such as benzene rings or alcohol groups.

Once you have an input file, you need to run it through the

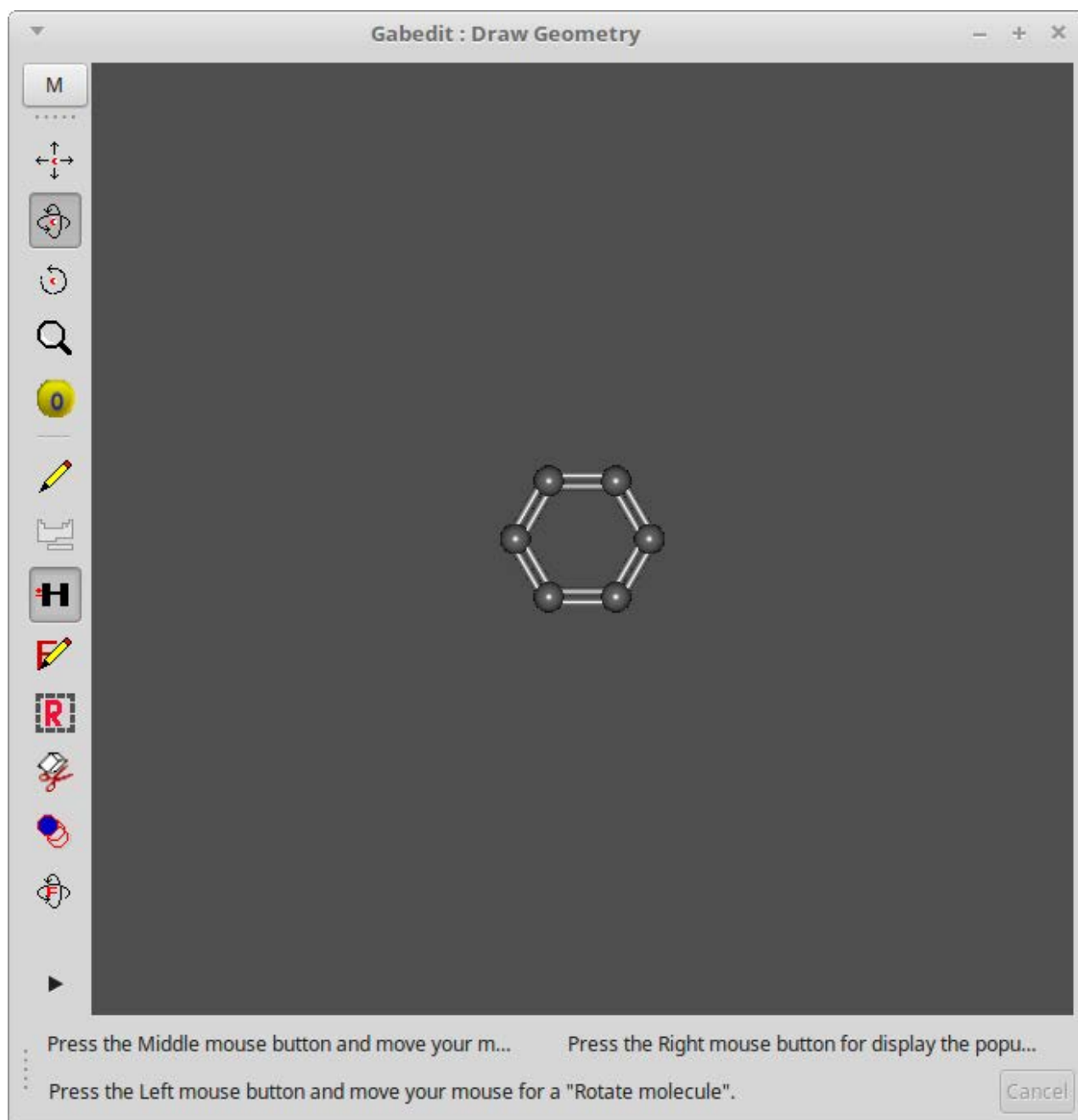


Figure 5. You can use the draw functionality to visualize the geometry of your collection of atoms.

appropriate software package in order to get results. If the programs you wish to use are installed on your local machine and are in your search path, it should just work out of the box. If they were installed in some other location, you need to tell Gabedit where they are. Clicking the Settings→Preferences menu item will bring up a new window where you can set the commands needed to run the relevant programs.

You then can run the program either by clicking the run button in the icon bar or clicking the Run→Run a Computation Chemistry program menu item. This will present a new window where you can set the parameters for this run.

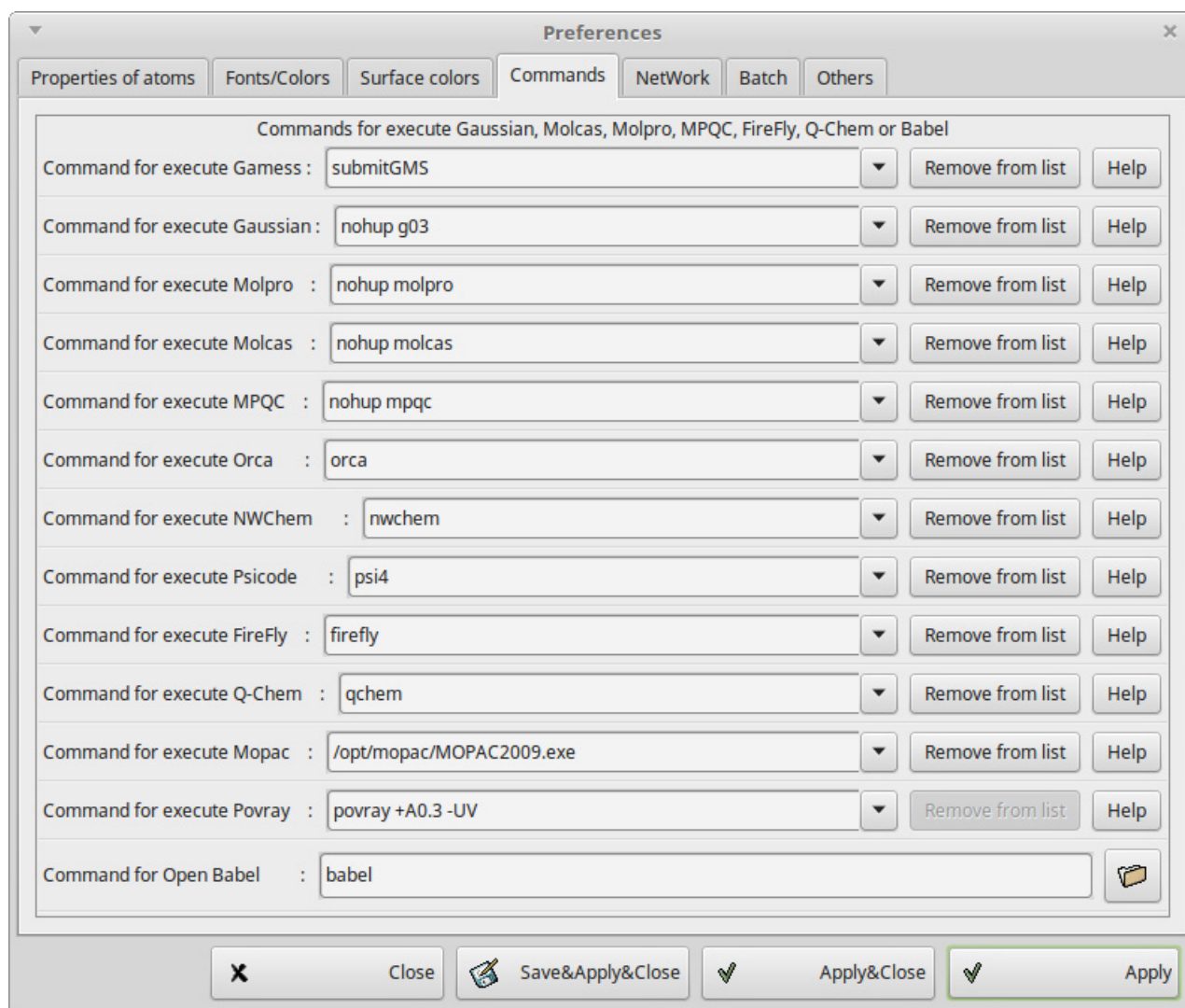


Figure 6. You can set the specific commands for each of the available chemistry packages.

For a local run, you can set parameters including which program to use, what folder to run in and the filenames and commands to execute. If you select "Remote host" instead, you can choose the protocol to communicate over and which host to communicate with. You also can set

Run

Program

Gamess Gaussian Molcas

Molpro Mopac MPQC

Orca NWChem FireFly

Q-Chem Psicode Other

Server

Local Remote host

Local

Folder:

Local/Remote

Save data in file : .inp

Command to execute :

NetWork protocols

FTP and rsh protocols ssh/scp protocols

Remote host

Host name :

Login :

Password :

Working Directory :

Figure 7. You can set the parameters for either a local run or a remote run within the same window.

the user name and password to use, along with the working directory on the remote machine. If you find that your initial choice of program isn't optimal, you can try another. By clicking the Tools→Open Babel menu item, you get a window that allows you to do a translation of input file from one file format to another. This way, you can reuse your previous work within a different software package.

Gabedit is not only useful in setting up a computational chemistry problem and running it, but it's also useful in analyzing the results afterward. The analysis functions are available under the Tools menu item. You can select to load a file for a basic XY-plot, and you can select the Tools→XY plotter menu item to bring up the plot window. Right-clicking the plot window brings up a menu where you can change the options of the plot as well as load data files to be plotted. There also is an option to do contour plots by clicking the Tools→Contours plotter menu item.

Additionally, there is a whole series of spectrum analyses that you can apply as well. You can do IR, Rahman, UV and ECD spectral analysis. For each of these options in the Tools menu, you can load an output file from a number of different file formats, including a special Gabedit file format.

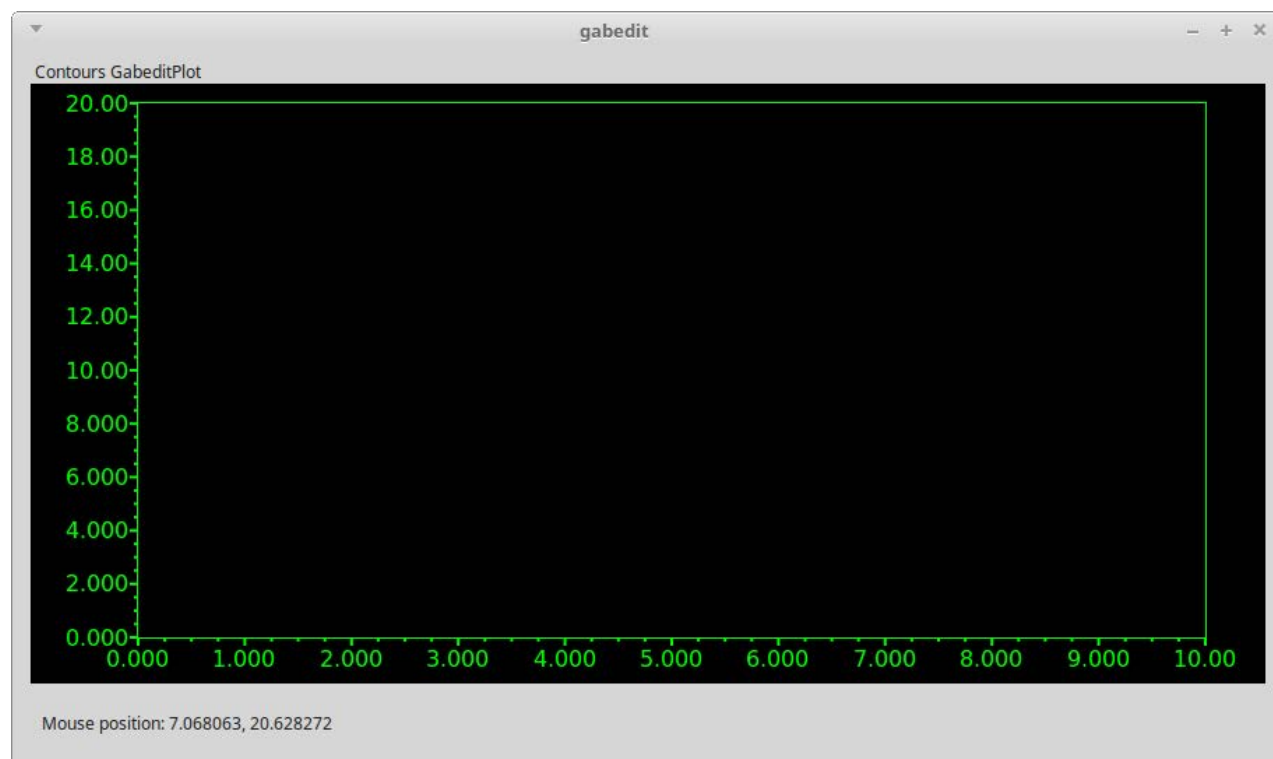


Figure 8. You can do contour plots of the results from a computation.

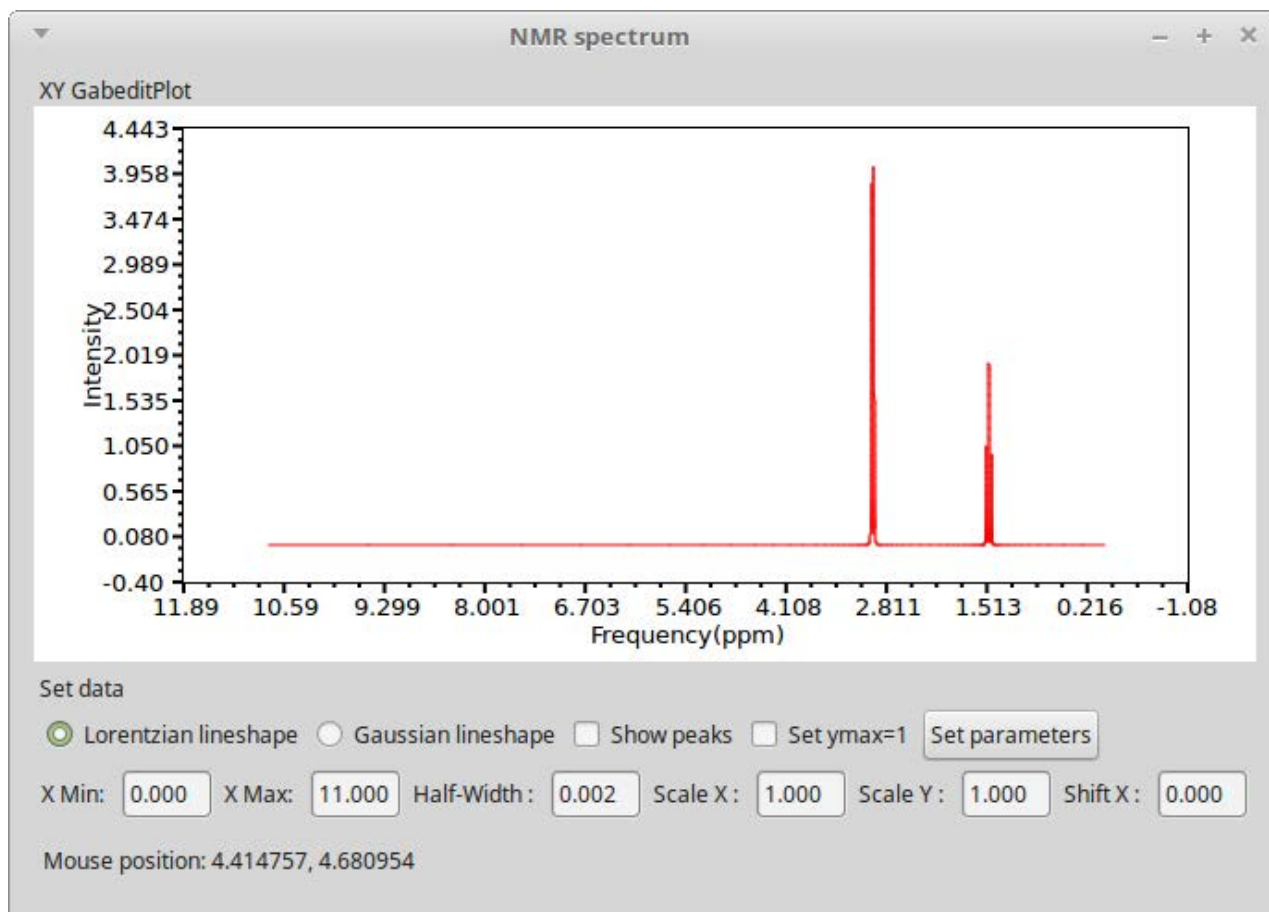


Figure 9. You can do NMR spectrum simulations for your molecule of choice.

Under the NMR spectrum entry of the Tools menu, you can select to load either a previously calculated results file or the NMR Spin-Spin Splitting Simulation.

Here you can set several options, such as the lineshape and the scaling. If you right-click the plot window, you have the same options as in the other plot windows. You also can add more data sets, change the plot details or the overall color theme.

With Gabedit, you can use quite a few of the available chemistry packages from a unified user interface. When doing more complicated research, or doing discovery work, being able to use multiple packages definitely will make everything easier to handle. You also can expand the options within Gabedit by adding your own functional units or altering the molecular mechanics parameters to be used in your work. Hopefully, Gabedit can help move your research into new areas.—Joey Bernard

The Fifteenth Annual
Southern California Linux Expo

SCALE 15x

March 2-5, 2017

Pasadena Convention Center
Pasadena, CA



<http://www.socallinuxexpo.org>
Use Promo Code LJ15X for a 30%
discount on admission to SCALE



PREVIOUS
UpFront

NEXT
Dave Taylor's
Work the Shell



Android Candy: Exploding Kittens!



I don't very often play games. I know that seems odd, because I do often write about gaming. Honestly though, I very rarely actually take the time to play video games. Recently, however, there has been an



exception to that rule.

One of my favorite online comics is *The Oatmeal*. The creator collaborated with another guy and came up with an incredibly fun card game called *Exploding Kittens*. I love the game. My teenage daughters love the game. Heck, I've even purchased another box so my college-aged daughter could play it with her roommates. Not only is the card game fun, but they also made a video game version that was on iOS-only for a long time.

Well, no more. Now you can get *Exploding Kittens* for \$1.99 at the Google Play store. It supports playing with random weirdos on the internet (I could be one of those weirdos) and playing with a group of friends. I won't describe the game itself other than to say it's silly, hilarious and fun. Plus, there are lots of awesome graphics drawn by *The Oatmeal*. In fact, this game is so much fun for such a reasonable price, I'm giving it this month's Editors' Choice award even though it's not open source. Because truly, it's an incredibly fun game that you can play in five minutes while you're doing whatever you might be doing that would facilitate five minutes of quiet time on your cell phone.

Search for *Exploding Kittens* at the Google Play store, and start playing now!—Shawn Powers

[RETURN TO CONTENTS](#)

Scissors, Paper or Rock?

I've spent a lot of time in this column looking at the sky—whether it was a Martian lander or a phase of the moon program, lots of math, lots of interesting code. Now let's land back on Earth and tackle a simple, straightforward challenge that has nothing to do with asteroids, gravitational anomalies or wormholes—well, hopefully not.



DAVE TAYLOR

Dave Taylor has been hacking shell scripts on UNIX and Linux systems for a really long time. He's the author of *Learning Unix for Mac OS X* and *Wicked Cool Shell Scripts*. You can find him on Twitter as @DaveTaylor, or reach him through his tech Q&A site: <http://www.AskDaveTaylor.com>.

◀ PREVIOUS
Editors' Choice

NEXT
Kyle Rankin's
Hack and /



IN THIS ARTICLE, I'm going to tackle a children's game that's extraordinarily complicated, with many variations, and the programming task is going to be quite tricky. Just kidding! Rock Paper Scissors (or RPS, as it's known) is pretty darn easy to simulate because there aren't really many variants or possible outcomes.

If you've never played it before, it's a one-vs.-one game where each person secretly chooses one of three possible options (rock, paper or, you guessed it, scissors). The players reveal their choices

WORK THE SHELL

simultaneously, and then there are rules about what beats what. For example, scissors beats paper because “scissors cut paper”, and rock beats scissors because “rock beats scissors”. If both players pick the same option, it’s a tie and the game proceeds.

Although you can play it as a one-off, it’s also generally played as a best of three to even things out slightly, although if everything’s completely random, you’ll win 33.33% of the time. For any given choice, there’s a 1/3 chance that you’ll have a tie, where both players pick the same thing, a 1/3 chance that you’ll win, and a 1/3 chance that you’ll lose.

The World Rock Paper Scissors Society

Except, in the real game, it turns out that there’s psychology involved too. In fact, according to the World Rock, Paper, Scissors Society (<http://worldrps.com>), rock is chosen 35.4%, paper 35% of the time and scissors only 29.6% of the time. Got it?

For the first version of the program, however, let’s stick with a completely random choice. The easy way to choose a random number between 1 and 3 in a Linux shell script is to use the variable `$RANDOM` like this:

```
compchoice=$(( ($RANDOM % 3) + 1 ))
```

The `%` is a modulus function and causes the random integer to be divided by 3, resulting in a 0..2 value. Add one, and you’ve got the 1..3 value. Easy enough.

With a simple shell array, you can add the name of the choice (remember, arrays start at index 0):

```
declare -a RPS; RPS=(nothing rock paper scissors)
```

Then the choice name is specified simply as:

```
choicename=${RPS[$compchoice]}
```

Those three lines are good enough for a tiny script where the computer

WORK THE SHELL

can choose randomly between rock, paper and scissors:

```
declare -a RPS; RPS=(nothing rock paper scissors)
compchoice=$(( ($RANDOM % 3) + 1 ))
echo "The computer chose ${RPS[$compchoice]}"
exit 0
```

Easy, but not very glamorous:

```
$ sh rps.sh
The computer chose rock
$
```

It's considerably more fun to have the computer prompt users for their selection, then "choose" its own and decide who won.

Making It into a Game

Interactivity is easily added by prompting users to choose whether they want rock, paper or scissors using a numeric value. Even better, you can prompt them using the same numeric values you're using internally:

```
echo -n "Please choose (1 = rock / 2 = paper / 3 = scissors): "
read choice
```

It's not a particularly onerous task to add interactivity, eh?

Now you need to compare answers and generate a result message. This is best done in a function, either standalone or by including an output string and tracking win/loss. I'll go for overkill (of course), so here's my function:

```
results() {
    # output results of the game, increment wins if appropriate
    echo ""
    if [ $choice = $compchoice ] ; then
        echo "You both chose $choicename. TIED!"
    fi
}
```

WORK THE SHELL

```
# rock beats scissors. paper beats rock. scissors beat paper.
# OR: 1 beats 3, 2 beats 1, and 3 beats 2.

elif [ $choice -eq 1 -a $compchoice -eq 3 ] ; then
    echo "Your rock beats the computer's scissors! Huzzah!!!"
    wins=$(( $wins + 1 ))
elif [ $choice -eq 2 -a $compchoice -eq 1 ] ; then
    echo "Your paper beats the computer's rock! Hurray!"
    wins=$(( $wins + 1 ))
elif [ $choice -eq 3 -a $compchoice -eq 2 ] ; then
    echo -n "Your scissors cut - and beat - the computer's "
    echo "paper! YAY!"
    wins=$(( $wins + 1 ))
elif [ $choice -eq 3 -a $compchoice -eq 1 ] ; then
    echo "The computer's rock beats your scissors! Boo."
elif [ $choice -eq 1 -a $compchoice -eq 2 ] ; then
    echo "The computer's paper beats your rock! Ptoi!"
elif [ $choice -eq 2 -a $compchoice -eq 3 ] ; then
    echo -n "The computer's scissors cut - and beat - "
    echo "your paper! Bummer."
else
    echo "Huh? choice=$choice and compchoice=$computer"
fi
}
```

It's straightforward, just a lot of typing. But really, that's 95% of the program. All you need is a looping mechanism so that you're "stuck" in the program until you get sick of the game—I mean ready to wrap things up.

Notice that the above code tracks wins, but not total games played; that'll have to be done in the main code, which, of course, is pretty straightforward because of how much of the code is pushed into the `results()` function:

```
echo "Rock, paper, scissors..."
echo "(quit by entering 'q' to see your results)"
```

WORK THE SHELL

```
while [ true ] ; do
  echo ""
  echo -n "Choose (1 = rock / 2 = paper / 3 = scissors): "
  read choice
  if [ "$choice" = "q" -o "$choice" = "quit" -o -z "$choice" ]
  then
    echo ""
    echo "Done. You played $games games, and won $wins of 'em."
    exit 0
  fi
  compchoice=$(( ($RANDOM % 3) + 1 ))
  choicename=${RPS[$compchoice]}
  games=$(( $games + 1 ))
  results
done
```

A quick run reveals that scissors isn't a bad strategy when the game is picking completely randomly:

```
$ sh rps.sh
Choose (1 = rock / 2 = paper / 3 = scissors): 3
Your scissors cut - and beat - the computer's paper! YAY!
$
```

When I tried it, I had a surprisingly longer-term result: an all-scissors strategy produced a 50% win rate (six games out of 12). Statistically that's unlikely if the computer really is picking randomly, but sometimes random is not so random.

Let's look at choosing paper:

```
$ sh rps.sh
Choose (1 = rock / 2 = paper / 3 = scissors): 2
The computer's scissors cut - and beat - your paper! Bummer.
$
```

In fact, playing all paper won only four of 14 games on a trial, and rock,

the most popular choice? That produces a win rate of three out of 14—worse than paper!

Matching Probabilities

The biggest change you could make to this program to match the “real” choice statistics is to stop picking randomly and instead reflect the percentages that the Rock Paper Scissors Society publishes: rock is chosen 35.4%, paper 35% and scissors only 29.6% of the time.

The easiest way to model that is to choose a random number between 1–1000 and then say that 1–354 is rock, 355–705 is paper, and 706–1000 is scissors. Instead of a single line where the number is being chosen, a function would be well written, and it’s pretty darn easy.

The other area you can expand this is to add a few more possibilities, and I bet most everyone reading this knows how to add “lizard” and “Spock” to the mix. Not sure? Here’s how a five-object RPS game works: <http://www.samkass.com/theories/RPSSL.html>.

So there you have it. Scientific? Not really. But, uh, rock, paper, scissors—come on! ■

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

Sysadmin 101: Alerting

Learn from my mistakes in this article covering on-call alert best practices.



KYLE RANKIN

Kyle Rankin is a Sr. Systems Administrator in the San Francisco Bay Area and the author of a number of books, including *The Official Ubuntu Server Book*, *Knoppix Hacks* and *Ubuntu Hacks*. He is currently the president of the North Bay Linux Users' Group.

PREVIOUS

◀ Dave Taylor's
Work the Shell

NEXT

Shawn Powers'
The Open-Source
Classroom ▶

THIS IS THE FIRST IN A SERIES OF ARTICLES ON SYSTEM ADMINISTRATOR FUNDAMENTALS.

These days, DevOps has made even the job title "system administrator" seem a bit archaic, much like the "systems analyst" title it replaced. These DevOps positions are rather different from typical sysadmin jobs in the past in that they have a much larger emphasis on software development far beyond basic shell scripting. As a result, they often are filled with people with software development backgrounds without much prior sysadmin experience. In the past, sysadmins would enter the role at a junior level and be mentored by a senior sysadmin on the team, but in many cases currently, companies go quite a while with cloud outsourcing before their first DevOps hire. As a result, DevOps engineers might be thrust into the role at a junior level

with no mentor around apart from search engines and Stack Overflow posts. In this series of articles, I'm going to expound on some of the lessons I've learned through the years that might be obvious to longtime sysadmins but may be news to someone just coming into this position.

In this first article, I cover on-call alerting. Like with any job title, the responsibilities given to sysadmins, DevOps and Site Reliability Engineers may differ, and in some cases, they may not involve any kind of 24x7 on-call duties, if you're lucky. For everyone else, though, there are many ways to organize on-call alerting, and there also are many ways to shoot yourself in the foot.

The main enemies of on-call alerting are false positives, with the main risks being ignoring alerts or burnout for members of your team. This article talks about some best practices you can apply to your alerting policies that hopefully will reduce burnout and make sure alerts aren't ignored.

Alert Thresholds

A common pitfall sysadmins run into when setting up monitoring systems is to alert on too many things. These days, it's simple to monitor just about any aspect of a server's health, so it's tempting to overload your monitoring system with all kinds of system checks. One of the main ongoing maintenance tasks for any monitoring system is setting appropriate alert thresholds to reduce false positives. This means the more checks you have in place, the higher the maintenance burden. As a result, I have a few different rules I apply to my monitoring checks when determining thresholds for notifications.

Critical alerts must be something I want to be woken up about at 3am. A common cause of sysadmin burnout is being woken up with alerts for systems that don't matter. If you don't have a 24x7 international development team, you probably don't care if the build server has a problem at 3am, or even if you do, you probably are going to wait until the morning to fix it. By restricting critical alerts to just those systems that must be online 24x7, you help reduce false positives and make sure that real problems are addressed quickly.

Critical alerts must be actionable. Some organizations send alerts when just about anything happens on a system. If I'm being woken up at 3am, I want to have a specific action plan associated with that alert so I

can fix it. Again, too many false positives will burn out a sysadmin that's on call, and nothing is more frustrating than getting woken up with an alert that you can't do anything about. Every critical alert should have an obvious action plan the sysadmin can follow to fix it.

Warning alerts tell me about problems that will be critical if I don't fix them. There are many problems on a system that I may want to know about and may want to investigate, but they aren't worth getting out of bed at 3am. Warning alerts don't trigger a pager, but they still send me a quieter notification. For instance, if load, used disk space or RAM grows to a certain point where the system is still healthy but if left unchecked may not be, I get a warning alert so I can investigate when I get a chance. On the other hand, if I got only a warning alert, but the system was no longer responding, that's an indication I may need to change my alert thresholds.

Repeat warning alerts periodically. I think of warning alerts like this thing nagging at you to look at it and fix it during the work day. If you send warning alerts too frequently, they just spam your inbox and are ignored, so I've found that spacing them out to alert every hour or so is enough to remind me of the problem but not so frequent that I ignore it completely.

Everything else is monitored, but doesn't send an alert. There are many things in my monitoring system that help provide overall context when I'm investigating a problem, but by themselves, they aren't actionable and aren't anything I want to get alerts about. In other cases, I want to collect metrics from my systems to build trending graphs later. I disable alerts altogether on those kinds of checks. They still show up in my monitoring system and provide a good audit trail when I'm investigating a problem, but they don't page me with useless notifications.

Kyle's rule. One final note about alert thresholds: I've developed a practice in my years as a sysadmin that I've found is important enough as a way to reduce burnout that I take it with me to every team I'm on. My rule is this:

If sysadmins were kept up during the night because of false alarms, they can clear their projects for the next day and spend time tuning alert thresholds so it doesn't happen again.

There is nothing worse than being kept up all night because of false positive alerts and knowing that the next night will be the same and that there's nothing you can do about it. If that kind of thing continues, it inevitably will lead either to burnout or to sysadmins silencing their pagers. Setting aside time for sysadmins to fix false alarms helps, because they get a chance to improve their night's sleep the next night. As a team lead or manager, sometimes this has meant that I've taken on a sysadmin's tickets for them during the day so they can fix alerts.

Paging

Sending an alert often is referred to as paging or being paged, because in the past, sysadmins, like doctors, carried pagers on them. Their monitoring systems were set to send a basic numerical alert to the pager when there was a problem, so that sysadmins could be alerted even when they weren't at a computer or when they were asleep. Although we still refer to it as paging, and some older-school teams still pass around an actual pager, these days, notifications more often are handled by alerts to mobile phones.

The first question you need to answer when you set up alerting is what method you will use for notifications. When you are deciding how to set up pager notifications, look for a few specific qualities.

Something that will alert you wherever you are geographically.

A number of cool office projects on the web exist where a broken software build triggers a big red flashing light in the office. That kind of notification is fine for office-hour alerts for non-critical systems, but it isn't appropriate as a pager notification even during the day, because a sysadmin who is in a meeting room or at lunch would not be notified. These days, this generally means some kind of notification needs to be sent to your phone.

An alert should stand out from other notifications. False alarms can be a big problem with paging systems, as sysadmins naturally will start ignoring alerts. Likewise, if you use the same ringtone for alerts that you use for any other email, your brain will start to tune alerts out. If you use email for alerts, use filtering rules so that on-call alerts generate a completely different and louder ringtone from regular emails and vibrate the phone as well, so you can be notified even if you silence your phone

After all, servers always seem to misbehave at around 3am.

or are in a loud room. In the past, when BlackBerries were popular, you could set rules such that certain emails generated a “Level One” alert that was different from regular email notifications.

The BlackBerry days are gone now, and currently, many organizations (in particular startups) use Google Apps for their corporate email. The Gmail Android application lets you set per-folder (called labels) notification rules so you can create a filter that moves all on-call alerts to a particular folder and then set that folder so that it generates a unique alert, vibrates and does so for every new email to that folder. If you don't have that option, most email software that supports multiple accounts will let you set different notifications for each account so you may need to resort to a separate email account just for alerts.

Something that will wake you up all hours of the night. Some sysadmins are deep sleepers, and whatever notification system you choose needs to be something that will wake them up in the middle of the night. After all, servers always seem to misbehave at around 3am. Pick a ringtone that is loud, possibly obnoxious if necessary, and also make sure to enable phone vibrations. Also configure your alert system to re-send notifications if an alert isn't acknowledged within a couple minutes. Sometimes the first alert isn't enough to wake people up completely, but it might move them from deep sleep to a lighter sleep so the follow-up alert will wake them up.

While ChatOps (using chat as a method of getting notifications and performing administration tasks) might be okay for general non-critical daytime notifications, they are not appropriate for pager alerts. Even if you have an application on your phone set to notify you about unread messages in chat, many chat applications default to a “quiet time” in the middle of the night. If you disable that, you risk being paged in the middle of the night just because someone sent you a message. Also, many third-party ChatOps systems aren't necessarily known for their

mission-critical reliability and have had outages that have spanned many hours. You don't want your critical alerts to rely on an unreliable system.

Something that is fast and reliable. Your notification system needs to be reliable and able to alert you quickly at all times. To me, this means alerting is done in-house, but many organizations opt for third parties to receive and escalate their notifications. Every additional layer you can add to your alerting is another layer of latency and another place where a notification may be dropped. Just make sure whatever method you choose is reliable and that you have some way of discovering when your monitoring system itself is offline.

In the next section, I cover how to set up escalations—meaning, how you alert other members of the team if the person on call isn't responding. Part of setting up escalations is picking a secondary, backup method of notification that relies on a different infrastructure from your primary one. So if you use your corporate Exchange server for primary notifications, you might select a personal Gmail account as a secondary. If you have a Google Apps account as your primary notification, you may pick SMS as your secondary alert.

Email servers have outages like anything else, and the goal here is to make sure that even if your primary method of notifications has an outage, you have some alternate way of finding out about it. I've had a number of occasions where my SMS secondary alert came in before my primary just due to latency with email syncing to my phone.

Create some means of alerting the whole team. In addition to having individual alerting rules that will page someone who is on call, it's useful to have some way of paging an entire team in the event of an "all hands on deck" crisis. This may be a particular email alias or a particular key word in an email subject. However you set it up, it's important that everyone knows that this is a "pull in case of fire" notification and shouldn't be abused with non-critical messages.

Alert Escalations

Once you have alerts set up, the next step is to configure alert escalations. Even the best-designed notification system alerting the most well intentioned sysadmin will fail from time to time either because a sysadmin's phone crashed, had no cell signal, or for whatever reason, the

Alert escalations are one of those areas that some monitoring systems do better than others.

sysadmin didn't notice the alert. When that happens, you want to make sure that others on the team (and the on-call person's second notification) is alerted so someone can address the alert.

Alert escalations are one of those areas that some monitoring systems do better than others. Although the configuration can be challenging compared to other systems, I've found Nagios to provide a rich set of escalation schedules. Other organizations may opt to use a third-party notification system specifically because their chosen monitoring solution doesn't have the ability to define strong escalation paths. A simple escalation system might look like the following:

- Initial alert goes to the on-call sysadmin and repeats every five minutes.
- If the on-call sysadmin doesn't acknowledge or fix the alert within 15 minutes, it escalates to the secondary alert and also to the rest of the team.
- These alerts repeat every five minutes until they are acknowledged or fixed.

The idea here is to give the on-call sysadmin time to address the alert so you aren't waking everyone up at 3am, yet also provide the rest of the team with a way to find out about the alert if the first sysadmin can't fix it in time or is unavailable. Depending on your particular SLAs, you may want to shorten or lengthen these time periods between escalations or make them more sophisticated with the addition of an on-call backup who is alerted before the full team. In general, organize your escalations so they strike the right balance between giving the on-call person a chance to respond before paging the entire team, yet not letting too much time pass in the event of an outage in case the person on call can't respond.

If you are part of a larger international team, you even may be able to set up escalations that follow the sun. In that case, you would select on-call administrators for each geographic region and set up the alerts so that they were aware of the different time periods and time of day in those regions, and then alert the appropriate on-call sysadmin first. Then you can have escalations page the rest of the team, regardless of geography, in the event that an alert isn't solved.

On-Call Rotation

During World War One, the horrors of being in the trenches at the front lines were such that they caused a new range of psychological problems (labeled shell shock) that, given time, affected even the most hardened soldiers. The steady barrage of explosions, gun fire, sleep deprivation and fear day in and out took its toll, and eventually both sides in the war realized the importance of rotating troops away from the front line to recuperate.

It's not fair to compare being on call with the horrors of war, but that said, it also takes a kind of psychological toll that if left unchecked, it will burn out your team. The responsibility of being on call is a burden even if you aren't alerted during a particular period. It usually means you must carry your laptop with you at all times, and in some organizations, it may affect whether you can go to the movies or on vacation. In some badly run organizations, being on call means a nightmare of alerts where you can expect to have a ruined weekend of firefighting every time. Because being on call can be stressful, in particular if you get a lot of nighttime alerts, it's important to rotate out sysadmins on call so they get a break.

The length of time for being on call will vary depending on the size of your team and how much of a burden being on call is. Generally speaking, a one- to four-week rotation is common, with two-week rotations often hitting the sweet spot. With a large enough team, a two-week rotation is short enough that any individual member of the team doesn't shoulder too much of the burden. But, even if you have only a three-person team, it means a sysadmin gets a full month without worrying about being on call.

Holiday on call. Holidays place a particular challenge on your on-call rotation, because it ends up being unfair for whichever sysadmin it lands

on. In particular, being on call in late December can disrupt all kinds of family time. If you have a professional, trustworthy team with good teamwork, what I've found works well is to share the on-call burden across the team during specific known holiday days, such as Thanksgiving, Christmas Eve, Christmas and New Year's Eve. In this model, alerts go out to every member of the team, and everyone responds to the alert and to each other based on their availability. After all, not everyone eats Thanksgiving dinner at the same time, so if one person is sitting down to eat, but another person has two more hours before dinner, when the alert goes out, the first person can reply "at dinner", but the next person can reply "on it", and that way, the burden is shared.

If you are new to on-call alerting, I hope you have found this list of practices useful. You will find a lot of these practices in place in many larger organizations with seasoned sysadmins, because over time, everyone runs into the same kinds of problems with monitoring and alerting. Most of these policies should apply whether you are in a large organization or a small one, and even if you are the only DevOps engineer on staff, all that means is that you have an advantage at creating an alerting policy that will avoid some common pitfalls and overall burnout. ■

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

The **Free Software Foundation**, GNU Project and MIT's SIPB invite you to



LIBRE PLANET 2017

"Roots of Freedom"

A conference for everyone who loves free software

Doctorow

Stallman



- + Hackers
- + Activists
- + Beginners
- + Users
- + Writers
- + Artists

March 25 & 26 at MIT. Students and FSF members attend gratis.

libreplanet.org/conference

All Your Accounts Are Belong to Us

Make your accounts more secure with two-factor authentication!



**SHAWN
POWERS**

Shawn Powers is the Associate Editor for *Linux Journal*. He's also the Gadget Guy for LinuxJournal.com, and he has an interesting collection of vintage Garfield coffee mugs. Don't let his silly hairdo fool you, he's a pretty ordinary guy and can be reached via email at shawn@linuxjournal.com. Or, swing by the [#linuxjournal](https://freenode.net) IRC channel on [Freenode.net](https://freenode.net).

PREVIOUS

◀ Kyle Rankin's
Hack and /

NEXT

Susan Sons'
Under the Sink ▶

LAST WEEKEND MY WORK PHONE SUDDENLY STOPPED WORKING. Not the phone itself, but rather all service stopped. I first noticed (of course) due to an inability to load any web pages. Then I tried calling someone and realized my phone was disconnected. In fact, when someone tried to call me, it said the line was no longer in service. It was Sunday, and my phone is a company device, so I had to wait until Monday to get things sorted.

It turns out someone called in to Verizon claiming to be me. The individual claimed his phone (my phone) had been stolen, and he wanted to transfer service to another device. He had enough information about me to pass whatever verification Verizon required,

and if he'd been a little smoother on the phone, he'd have likely gotten my number. It turned out that the Verizon employee felt the call was suspicious and disabled the account instead of transferring service. (I know that only because the employee made a note on the account.) After a stressful day of back and forth, the company I work for was able to get my phone turned back on, and I still have the same phone number I've always had—thank goodness.

Kyle Rankin saw me tweet about my phone issues, and he immediately responded that I should check my online accounts, especially those with two-factor authentication. If other people *had* been able to get my phone number, they could use that as “proof” of their identity and reset many of my passwords. It hadn't occurred to me just how much we depend on our cell-phone companies for security, even on our personal bank accounts. That doesn't mean two-factor authentication (2FA) isn't important, it just means we need to consider our phones as a viable vector for attack. So in this article, I want to talk about securing your online accounts.

Call Your Mobile Provider

Before I talk about securing online accounts, I urge you to contact your cell-phone company. I use several providers myself, and after my experience with the company phone, I realized just how important it is to contact the provider and set up security. By default, your cell-phone company might have a few security questions for you to answer. It also might just ask for your date of birth in order to access account information. It's important to call and ask what sort of security you can add to the account to make sure a third party can't pretend to be you. What that security looks like will be different for every company, but really, call them. Anyone on Facebook can look up my birthday, and if that's all you need to make changes to an account...well, yikes.

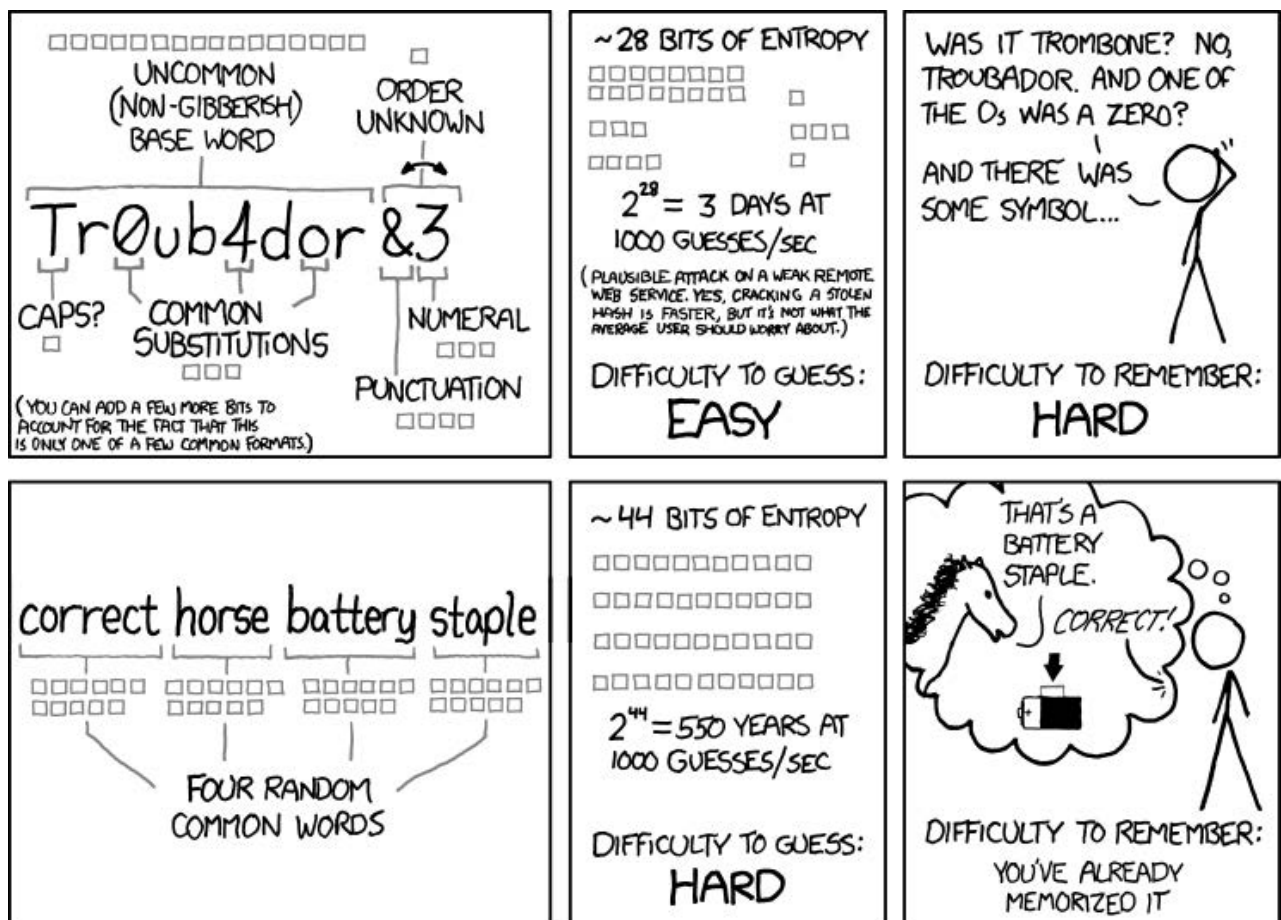
Once you're confident that your phone isn't easily compromised, it's time to start looking at your online accounts. Not all businesses provide two-factor authentication, but more and more are adding the service every day. Even if your banks, email accounts and Spotify stations don't have extra layers of protection, having a good password is crucial.

My Name Is My Passport, Verify Me

I've written in the past about creating "good" passwords. Some of what I recommended is valid, and some was shortsighted. I was in good company with my shortsightedness, because tons of companies still require "complex" passwords. The problem is, password complexity generates passwords that are hard for humans to remember and easy for computers to guess. The famous *xkcd* comic explains the problem much better than I can (Figure 1).

(Note: Randall Munroe from <http://xkcd.com> made it fairly clear that occasionally reprinting his comics is okay as long as he is attributed. I'll go so far as to say not only is his work awesome, but you should go buy things from his store. Seriously, he's awesome.)

The problem with truly "good" passwords is that they rarely meet the



THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Figure 1. This comic titled "Password Strength" from *xkcd* is so true it hurts (<https://xkcd.com/936>).

THE OPEN-SOURCE CLASSROOM

requirements for complexity that most websites demand. It seems like companies are perfectly fine with an eight-character password, as long as there's a capital letter, punctuation, a number and no common words. Basically, they demand we have crappy, hard-to-remember passwords. It's very frustrating.

If you're not using a password manager that generates random passwords, the best I can recommend is that you make your password as long as possible. My method for making a password is to string together words (like `correcthorsebatterystaple`, which I didn't even have to look up, because I totally remembered it), and then add the weird complexity requirements at the end. That still doesn't help with password reuse, however, which is an even bigger problem than using strong passwords. Again, Randall illustrates the problem perfectly at <https://xkcd.com/792>. (I'm just giving a link this time; I don't want to push my luck.)

Basically, if you use the same password everywhere, if one system is compromised, all your accounts are vulnerable. I addressed that problem in my last article about setting up good passwords, but unfortunately, any pattern you might use to create passwords can be figured out. Here's what I mean. Let's say you use this pattern for generating passwords:

```
word1 word2 sitename word3 word4 complexity_junk
```

On the surface, this seems brilliant. You can remember four words, have a standard "complexity" ending for meeting dumb password requirements, and you can add the name of the website in the middle. That means every password will be different. The problem is, it's still a pattern. Let's say an attacker discovers that your Facebook password is this:

```
hampergranitefacebookcoffeeostrich53BLT!
```

That's a nice, long, unique password. The problem is, now the attacker knows your Amazon password is this:

```
hampergraniteamazoncoffeeostrich53BLT!
```

Truly, the best method I know of is to have a password manager that will store and potentially generate passwords for you. I prefer passwords

I don't have to copy/paste in order to use, so I usually generate long passwords using words. That way, I can glance at the password and type it out quickly. The point of this whole section is to make you think about passwords. Consider passwords that are truly strong, but also remember that it's extremely important not to reuse passwords on multiple sites.

Adding Another Factor

Two-factor authentication comes in many flavors. For cell phones, the trend is to use fingerprints. Granted, fingerprints aren't the most secure authentication method, but when used in addition to passwords, it does add significant security. (I once heard Kyle Rankin say fingerprints are terrible passwords, because you can change your "password" only ten times, and you leave them written everywhere you touch.)

The cell-phone number itself is one of the most common forms of 2FA. Like my original example demonstrated, many websites utilize SMS messages sent to a phone number as verification of identity. There are many issues with an SMS being the sole form of authentication, but as a required second factor, it's not bad. What I mean by that is, many companies allow you to use your cell phone for 2FA, but they also allow you to recover your password by simply proving who you are by entering a code sent via SMS. That completely eliminates the security of 2FA!

My personal favorite 2FA method is provided by Google. The implementation is fairly robust, and in function, it's very easy to use. Basically, you authenticate your phone, and rather than having a code texted to you, which you have to type into a web form, the Google authenticator just pops up on your phone asking if you're currently trying to log in (with information on where you're trying). You simply click "yes", and the 2FA is successful. I like it not only for simplicity, but also because my phone number being hijacked doesn't automatically give the thief the ability to provide 2FA.

There certainly are other methods for attaining multiple authentication factors. Yubi is a company that has provided hardware-based USB authentication for years. The problem I usually have is not everywhere supports multiple forms of 2FA. However, if a website allows you to log in with your Google account, Google handles the 2FA, thus securing the site without any custom-2FA code on the particular site at all.

If You Use Google, Beef It Up

Part of me dislikes recommending Google as your go-to source for 2FA. Google is a commercial company, and using its proprietary system as a form of authentication is a little unsettling. But here's the deal: I'd rather everyone trust the integrity of Google than trust the integrity of random hackers on the internet. Google's 2FA is easy to set up, has proven to be reliable, and at the very least, it's better than not using 2FA at all. So if you're interested in continuing down the Google rabbit hole, I highly recommend you go through its security wizards to make sure your account is yours.

Head over to <https://accounts.google.com> (Figure 2). On the left, you'll see sign in and security options. This page is also where you can configure

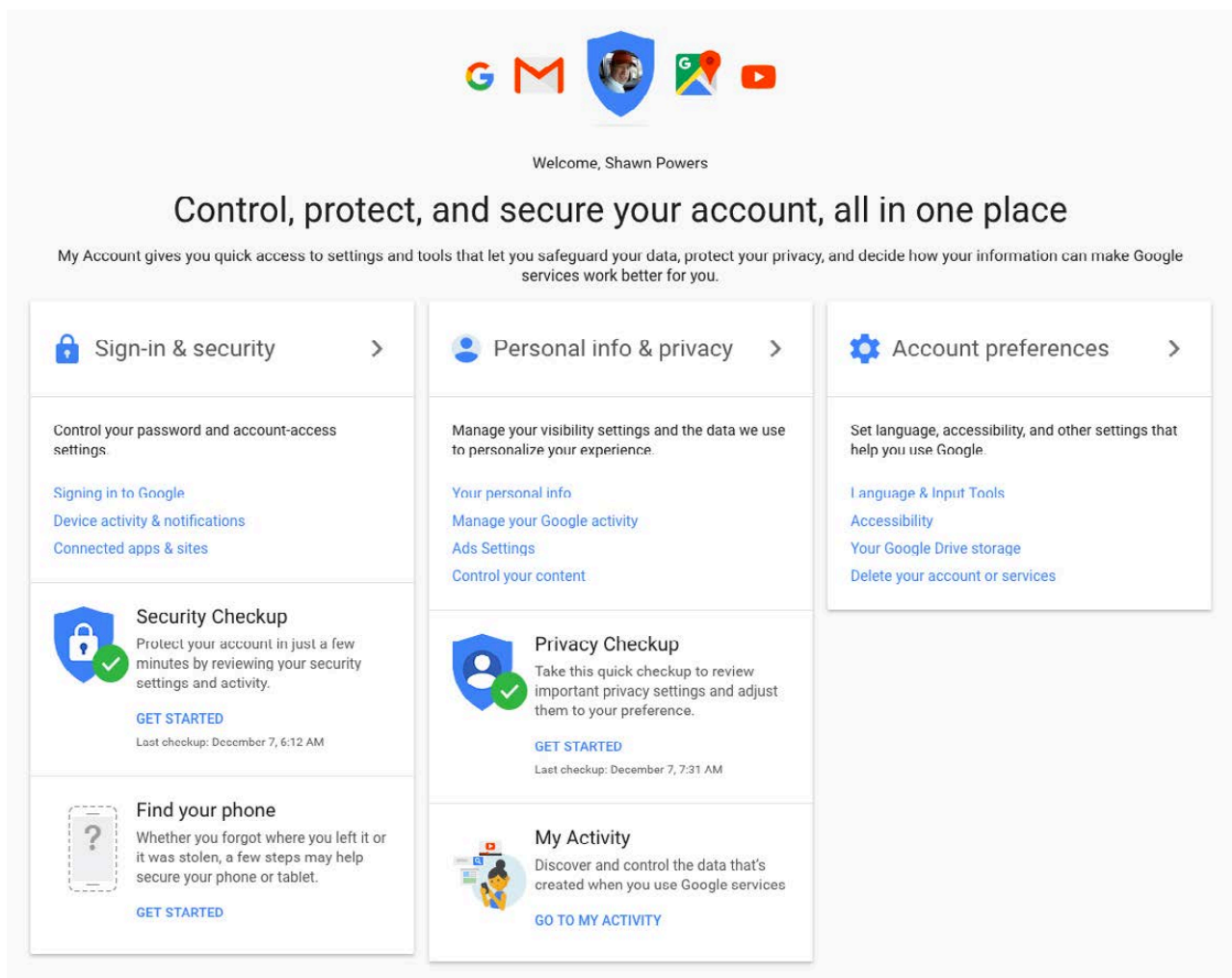


Figure 2. Follow all these links. The checkups are very useful, and it's better to over-prepare than under-prepare.

THE OPEN-SOURCE CLASSROOM

your privacy settings and recent activity. But for this article, I'm focusing on the sign-in and security page. Figure 3 shows when your password was last changed, whether or not 2FA (Google calls it two-step verification) is turned on, and whether you have any app passwords. You also can set up your account recovery information on this page, providing alternate email, phone number and secret questions.

When you turn on 2-step verification (Figure 4), you're able to configure multiple 2FA options and set a default. I use the Google Prompt (described

Figure 3. Please turn on 2FA. It's painless and so much more secure than a password alone.

Password & sign-in method

Your password protects your account. You can also add a second layer of protection with 2-Step Verification, which sends a single-use code to your phone for you to enter when you sign in. So even if somebody manages to steal your password, it is not enough to get into your account.

Note: To change these settings, you will need to confirm your password.

Password	Last changed: December 7, 6:10 AM	>
2-Step Verification	On since: December 7, 6:16 AM	>
App passwords	None	>

Account recovery options

If you forget your password or cannot access your account, we will use this information to help you get back in.

Recovery email	*****@gmail.com	>
Recovery phone	(202) 741-8278	>
Secret question	"What is your pet's name?"	>

THE OPEN-SOURCE CLASSROOM

previously) as my default method, but I also have my phone number as an option. Plus, Google allows you to configure a number of alternates like a USB hardware key, printable offline codes and an authenticator app that will generate 2FA codes even while your phone is offline. Truly, it's the variety of options that makes me love Google for my 2FA needs.

Ultimately, I urge you to set up 2FA on as many sites as support it. Most sites still require you to use SMS as the second factor, so be sure

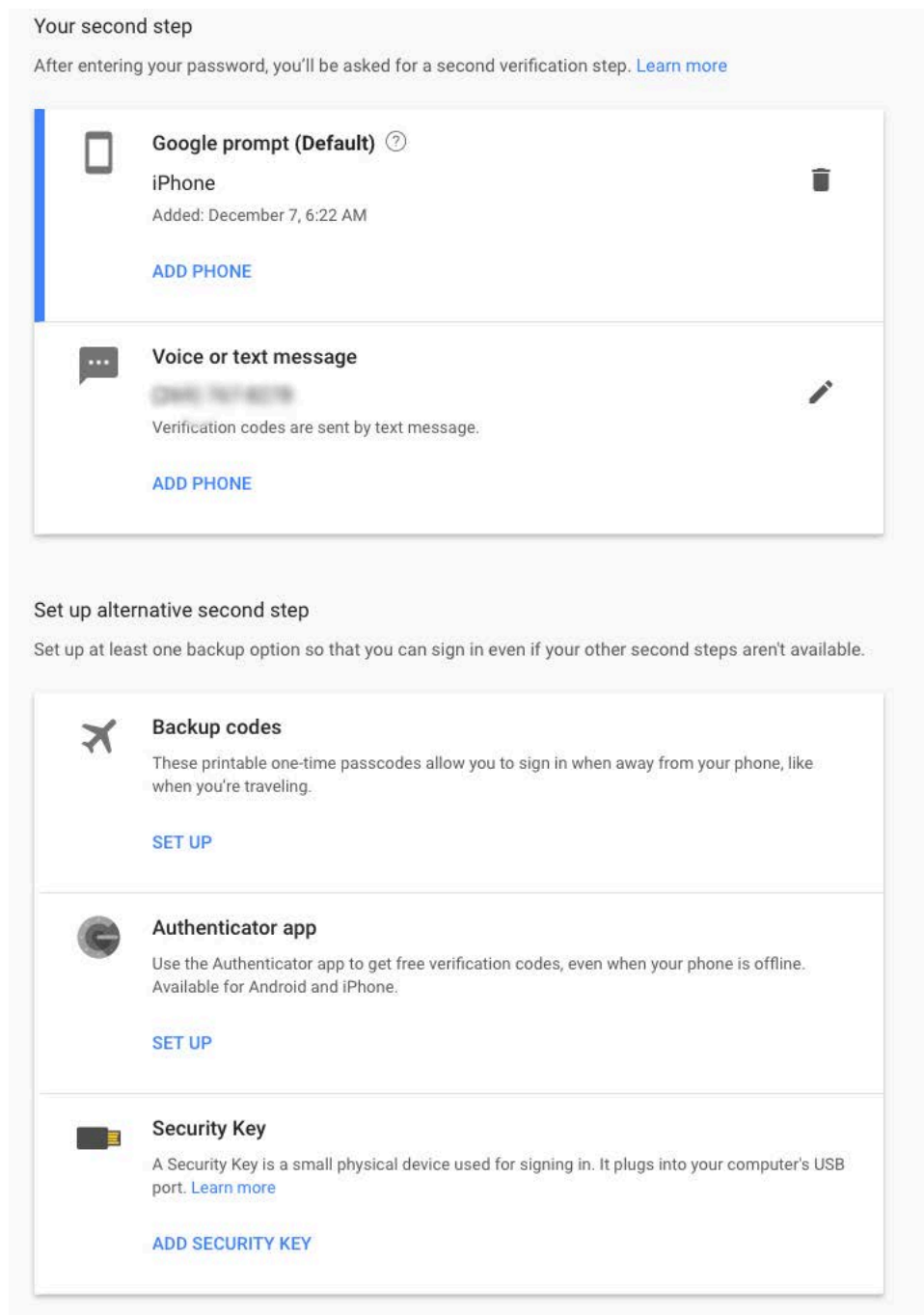


Figure 4.
Google's 2FA is really well done.

your phone number is secure (remember to contact your cell-phone provider). If websites support Google for 2FA instead of SMS, I personally recommend it. It's simpler, and that means you're actually more likely to use it. But whatever method you choose, 2FA is a good thing.

Password Management

I use a password manager. I've used several through the years, but I do find having a secure database of passwords is helpful. If I'm being completely honest, none of the password managers I've tried are perfect. It's often cumbersome to get the password (especially hard-to-type passwords) from the manager to the website where you need it. Plus, going between desktops and mobile devices is always a challenge. I use LastPass, but it's not a perfect solution, and it's not free for mobile devices. There are open-source password management tools like KeePass, Padlock and Passbolt, but I've yet to find the perfect solution. If you have a password management system that works across platforms and devices in a convenient, yet secure way, please let me know. I'd love to write about it.

So, the moral of the story is to make sure your phone is secure, and then make sure your accounts are secure too—preferably with multiple factors of authentication. At the absolute least, please don't use the same password for multiple websites!■

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

drupalize.me

Instant Access to Premium Online Drupal Training

- ✓ *Instant access to hundreds of hours of Drupal training with new videos added every week!*
- ✓ *Learn from industry experts with real world experience building high profile sites*
- ✓ *Learn on the go wherever you are with apps for iOS, Android & Roku*
- ✓ *We also offer group accounts. Give your whole team access at a discounted rate!*

Learn about our latest video releases and offers first by following us on Facebook and Twitter (@drupalizeme)!

Go to <http://drupalize.me> and get Drupalized today!



Postmortem

What to do after a security incident.

PREVIOUS

◀ Shawn Powers' The Open-Source Classroom

NEXT ▶
New Products

INCIDENTS HAPPEN. Vulnerabilities happen. The quality of your response can make the difference between a bad day and a disaster. What happens after the response can make the difference between endless firefighting and becoming stronger with every battle. A quality postmortem analysis is free ammunition. Every incident is someone or some event showing where a system's weaknesses are, if only one is willing to listen.

This is how a good information security officer, or an engineer who's a true information security evangelist, can make a difference:

1. Something happens. It may be an exercise, or a real incident.
2. You now have real information to go on. You are in a very different position from when you were working from the theoretical.
3. *If* you know how to understand that information, and what information you need, you may have a new



SUSAN SONS

Susan Sons serves as a Senior Systems Analyst at Indiana University's Center for Applied Cybersecurity Research (<http://cacr.iu.edu>), where she divides her time between helping NSF-funded science and infrastructure projects improve their security, helping secure a DHS-funded static analysis project, and various attempts to save the world from poor information security practices in general. Susan also volunteers as Director of the Internet Civil Engineering Institute (<http://icei.org>), a nonprofit dedicated to supporting and securing the common software infrastructure on which we all depend. In her free time, she raises an amazing mini-hacker, writes, codes, researches, practices martial arts, lifts heavy things and volunteers as a search-and-rescue and disaster relief worker.

Postmortem mistakes can have long-term implications, but they also can take a long time to identify. A bad postmortem feels just as satisfying as a good postmortem to someone who doesn't know the difference.

understanding of the project or organization's security needs. Even if this is only confirmation of what you knew before, it is important because...

4. This information and analysis, if communicated effectively, especially in the aftermath of an incident, can be a powerful tool for fixing problems.
5. Next time around, the organization will be a little more on its game, and another set of weaknesses can be shored up. Every iteration makes the organization stronger.

How to Sabotage Your Postmortem

Postmortem mistakes can have long-term implications, but they also can take a long time to identify. A bad postmortem feels just as satisfying as a good postmortem to someone who doesn't know the difference. Unfortunately, it fills a team—or a whole organization—with false beliefs, missed opportunities and bad data, eroding its ability to mature its security. These erosions are small individually, but like water lapping up against a beach, they eventually aggregate. Learn these anti-patterns and be certain to recognize them.

Play the blame game. Yes, some incidents are clearly one person's fault. Most of the time though, there's plenty of blame to go around. Blame is an out that makes it too easy to ignore systemic problems, and looking for someone to punish makes valuable sources of information go silent. Deal with personnel issues separately from incident postmortem, except in cases of actual malicious insider attacks.

Stop at the vulnerability. Calling it quits once you've found

something to patch or a configuration to change is perhaps the most common mistake. In the best of cases, looking deeper can confirm what's working and what isn't. In the majority of cases, there is more than one cause to be found. Don't stop looking once you've found something; poke in all the corners first.

Stop at the forensics. Another common mistake is to look for signs of technological vulnerability or compromise, such as incorrect firewall configurations, software bugs, rootkits and so on without looking at the bigger picture of what may be causing those things to happen. Poor software engineering practice or inadequate tools for engineers will raise the incidence of bugs. So will overwork and poor morale. Similarly, a lack of configuration management for systems can cause mistakes due to forcing administrators to repeat processes by rote many times.

What Actually Works

See failures as information. Every failure, including not having enough information to do a proper postmortem, is itself information. Do not lose sight of this. If you find yourself at a loss in a postmortem, start looking at what you would have needed to do a postmortem that you don't have. That is your first lesson learned.

Treat "root cause" as an adjective. There's never only one root cause, because if there is only one root cause, the other root cause is "we failed to practice fault tolerance by implementing defense in depth". Root cause analysis is the act of finding root causes, plural, not the search for a single root cause.

Go back to first principles. In my day job at Indiana University's Center for Applied Cybersecurity Research (<http://cacr.iu.edu>), we've been working on a set of seven principles from which cybersecurity in general can be derived (<http://cacr.iu.edu/principles>). First principles work in reverse as well: they are not only a tool for performing information security, but also for figuring out how information security failed.

■ **Comprehensivity.** Was there a system no one knew about? Was a risk being ignored? Comprehensivity failures tend to be failures of scope.

■ **Opportunity.** Did something go unmaintained because the burden

UNDER THE SINK

was placed on under-resourced in-house staff instead of using well maintained common tools? Were staff under-trained so that they didn't recognize something they should have? Was no one staying abreast of current threats?

- **Rigor.** Was the organization caught out by assumptions that weren't being verified? Did monitoring fail? Was something not specified clearly enough to ensure that everyone was on the same page? Was automation not put in place to ensure that repetitive tasks were not done precisely and consistently across time and space?
- **Minimization.** Was something a bigger target than it needed to be? Were there more ways in, or more moving parts, than there needed to be? Could something become easier to protect by eliminating or shrinking some part of it?
- **Compartmentation.** Did someone or something have access that it didn't absolutely need? Did isolation fail? Was cryptography not implemented appropriately? Were monolithic systems and processes used when things could have been segmented from one another? Were interfaces between systems or components of systems unclear or overly complex?
- **Fault tolerance.** Was there a single point of failure? Was there a credential that wasn't cheap and easy enough to revoke, so it wasn't replaced when it should have been? Was something built or configured with the assumption that bad things wouldn't happen to it?
- **Proportionality.** Was security, or any systems or software decision, made in isolation, without considering the environment as a whole? This one can be a killer—when security interferes with getting the job done, people will circumvent it. When security is too expensive, no one will implement it. When a business case hasn't been made relative to other risks, the organization won't know what security to invest in and may invest in none at all because doing *all* information security controls is untenable.

It is extremely hard to advocate for resources to be put into security in any organization, because resources are always limited and prevention is impossible to quantify.

It takes time to work with and learn to use the principles for analysis, but it's worth doing so. They are invaluable in flexing one's brain around whatever problem comes along, instead of learning types of problems one at a time. Each principle has much more to it than these brief examples, but the examples here should provide a starting point for how they may crop up in an incident postmortem.

Lessons Learned

Here are a few things I've learned through years of doing postmortem analyses.

There will be more bugs. There always will be more bugs. Sometimes the right answer really is "patch it and move on". However, one should not move on without asking whether one can become more robust. Could patching happen faster in order to prevent future compromises? Is there a secondary control that could be put in place so that a vulnerability in one component doesn't equate to a vulnerability in the system as a whole? How can fault tolerance be increased? Is there adequate monitoring for appropriate response? If the bug is in software you maintain, is the bug just a bug, or is it the result of engineering practices that are holding back your engineers or deeper architectural problems that should be cleaned up in a refactor?

Getting a patch out is nice, but eliminating classes of problems, rather than the one problem that came to your attention, is how a system or organization becomes more mature and more secure in a meaningful way over time.

An incident is proof, and proof is leverage. It is extremely hard to advocate for resources to be put into security in any organization, because resources are always limited and prevention is impossible to quantify. When there is an incident, you have something concrete in your

hands, *if* you know how to use it effectively.

Do not fall prey to the temptation to make every incident into a moral panic. Overblown scare tactics just serve to deafen others to the security team's constant cries of disaster. Save that for when the sky really is falling. Instead, look at what the incident cost to mitigate or what specifically was headed off. Look at what underlying problems were revealed, and what the aggregate cost would be of more incidents like this if the underlying problems are not fixed. Speak in risk vs. reward, and have dollar figures and time estimates at hand, even if they are a bit rough. Think about other organizational costs and benefits too, including changes in time to market, personnel turnover, reputation, liability and so on.

Do not provide a laundry list. If you ask for more than the decision-makers can take in, you have lost them. Do not drown them in minutia. They want to hear about the details of the build system's server components about as much as you want to hear about the components of the next shareholder meeting report. Perhaps less. There are entire books on clear communication and working with management, so I won't try to reproduce them here. Please go read one or two of them.

Keep track of change over time. Security professionals, and technologists in general, tend to be problem-solvers by nature. We focus on things that are broken. Not only can this make us seem negative to others, but it can cause us to appear like we're treading water rather than truly making progress, sometimes even to ourselves. Each postmortem—whether of an exercise or actual vulnerability/incident—can be leveraged to spur incremental improvement. Getting the most out of this aggregate requires knowing what the aggregate is.

Concretely demonstrating how security has improved year over year will improve team morale, protect security resourcing and autonomy, help leadership see security as a concrete, tractable problem rather than an infinite and nebulous one, and help the person pushing the security envelope stay sane. Also, if that year-over-year picture isn't a solid improvement in maturity, it's better to know that so you can triage the problem. Whether it's a lack of support, lack of training or something else, find a way to do better. No security is perfect. Strive to do better than your past self.

A Final Note on Forensics

Readers may note that, apart from assuming that some data about the nature of an incident is available, this article didn't talk about doing forensics. The truth is that digital forensics is expensive and requires specialized skills. It's also useless in an organization that doesn't know how to use the resulting information. Master postmortem analysis based on whatever information you have available, and you will soon know when you need more information, and when digital forensics techniques make sense for your budget and the incident at hand.

Don't get blindsided by the shiny sound of forensic techniques before you know whether more rudimentary analysis will get you where you need to go. My life-critical technology projects will engage in digital forensics when it's called for; a \$30k/year nonprofit project never will. I have yet other projects that may not have forensic resources themselves, but may cooperate with CERTs or ISACs as appropriate to help understand threats that are relevant to more than one organization.

Remember, the goal of a postmortem is to improve your defenses, not to answer every question. Real life is not a Sherlock Holmes novel. You don't always get a neat resolution with all loose ends neatly tied up. In fact, that almost never happens. ■

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

The Best SharePoint 2016 and Office 365 Training!



April 2-5, 2017 • AUSTIN, TEXAS



SPTechCon offers classes and tutorials for IT professionals, business decision makers, information workers, developers and software and information architects. Each presenter at SPTechCon is a true SharePoint expert, with many drawn from Microsoft's tech teams or holding Microsoft MVP status.

Whether you're looking to upgrade to a more current version, making a move to the cloud, or simply need answers to those daunting problems you've been unable to overcome, SPTechCon is the place for you! Come join us!

- Choose from more than 80 classes and panel sessions
- Improve your skills and broaden your knowledge of Microsoft's collaboration and productivity software
- Learn about SharePoint 2016, the latest on-premises server release from Microsoft
- Tips and tricks for working with SharePoint 2013 and 2010, and Office 365
- Practical information you can put to use on the job right away!
- The most knowledgeable instructors working in SharePoint today

www.sptechcon.com

NEW PRODUCTS

 PREVIOUS Susan Sons' Under the Sink	NEXT Feature: Cellular Man-in-the-Middle Detection with SITCH	
---	--	---



Nventify's Imagizer Cloud Engine

Organizations that rely on compelling imagery to help clients make informed decisions face challenges presenting it appropriately across devices. To assist, Nventify launched Imagizer Cloud Engine, a new cloud-based image manipulation platform that removes the complexities of dynamically delivering best-sized images to end users. The new platform enables customers to deploy image transformation services for their products in five minutes or less. Key features of the platform include dynamically responsive image adjustment based on screen layout, automatic format selection (such as WebP) by detecting browser and device types, reduction of image storage due to transcoding of master images on the fly and client-side SDKs. Libraries for the latter include compatibility with popular languages, such as JavaScript, Java, PHP, Ruby, Python and Swift.

<http://imagizer.nventify.com>



Server Technology's HDOT Alt-Phase Switched POPS PDU

Server Technology says that although the engineering challenge was significant—adding per outlet power measurement into its popular High-Density Outlet Technology (HDOT) power distribution unit (PDU) family—product quality and manufacturability were not sacrificed. The new HDOT Switched POPS (Per Outlet Power Sensing) PDU, bolstered by the addition of device-level monitoring, is the new ultimate solution for density, capacity planning and remote power management for the modern data center, states Server Technology. At each outlet, the novel POPS technology provides $\pm 1\%$ billable-grade accuracy for energy consumption for typical data-center equipment loads as well as current, voltage, active power, apparent power, power factor and crest factor. POPS complements the existing HDOT concept, “the smallest form factor PDU” that significantly increases real estate in the back of the rack by fitting as many as 42 C13s in a 42U-high network-managed PDU device. Finally, Alternating Phase outlets in Server Technology's PDUs distribute phases on a per receptacle basis, resulting in better load balancing and cable management.

<http://servertech.com>



UNIVERSAL DESKTOP CONVERTER

IGEL Universal Desktop Converter

IGEL Technology's next-generation Universal Desktop Converter 3 (UDC3) is a powerful and universally deployable managed thin-client solution, a low-cost alternative to desktop hardware solutions. UDC3 allows businesses to reduce their desktop replacement costs dramatically, eliminating the need to invest in new hardware to support their virtualized infrastructures. Converting PCs, laptops and thin clients from other manufacturers into IGEL Linux 10 OS-based thin clients also enables IT organizations to administer all of their endpoint devices from a centralized management console securely. The IGEL UDC3 can be installed as the operating system on any device having an x86-based 64-bit processor, 2GB of RAM and 2GB of storage. Using the IGEL UDC3 on these end-user computing devices converts them into an IGEL thin client running the IGEL Linux 10 OS. The updated IGEL Linux 10 OS now features support for the Unified Extensible Firmware Interface (UEFI) to extend the IGEL's UDC3 target platforms to the latest end-user devices including many laptop computers and desktop PCs, thin clients and compute sticks. Plus, with its enhanced 64-bit OS compatibility, the new version can address more than 4GB of RAM in next-generation devices.

<http://igel.com/us>



Natalie Rusk's Scratch Coding Cards (No Starch Press)

The phrase “Learn to Program One Card at a Time” plays the role of subtitle and friendly invitation from *Scratch Coding Cards*, a colorful collection of activities that introduce children to creative coding. Developed by Natalie Rusk, research scientist in the Lifelong Kindergarten Group at the MIT Media Lab, the

resource consists of illustrated activity cards that provide a playful entry point into Scratch, the graphical programming language used by millions of children around the world. The cards make it easy for kids to learn how to create a variety of interactive projects, such as a racing game, an animated interactive story, a virtual pet and much more. Each card features step-by-step instructions for beginners to start coding. The front of the card shows an activity kids can do with Scratch, such as animating a character or keeping score in a game. The back shows how to snap together blocks of code to make the projects come to life. Along the way, kids learn key coding concepts, such as sequencing, conditionals and variables. Publisher No Starch Press recommends the coding activity cards for sharing among small groups in homes, schools and after-school programs.

<http://nostarch.com>



Ensono M.O.

Application performance in hybrid IT environments typically has been a function of simple infrastructure provisioning. This limited approach cannot manage complex resources in real time nor ensure optimal, dynamic application performance, asserts hybrid IT services provider Ensono. To address this complexity, the company released Ensono M.O., a hybrid IT service platform that provides a comprehensive view of clients' managed solutions on any platform, anywhere. Ensono M.O. helps manage complex client solutions regardless of data center or cloud infrastructure and location. The platform ensures exceptionally high service levels by creating scalable, robust and transparent service delivery modes directly to clients. Ensono M.O. further enhances client service and accountability by codifying IT best practices and ensuring optimal real-time integration of technology, people and processes. Additional Ensono M.O. features include an integrated toolset that provides a single, comprehensive view of the client infrastructure in real time, efficient development of best-practice IT service management process in one integrated toolset, automation of working practices to help drive significant staffing efficiencies, business-critical transaction and application infrastructure monitoring and management, high levels of automation and scalability, "best bet" triage capability and 24/7 Global Operations Centers.

<http://ensono.com>

SmoothWall Express 3.0

Control About Services Networking VPN Logs Tools Maintenance

shutdown | help

system web proxy firewall ids instant messages email

Check logs for attempted access to your network from outside hosts. Connections listed here **have** been blocked.

Settings:
 Month: Day:

Log:

Time	In > Out		Source	Src Port	Destination	Dst Port
12:27:36	eth0 > -	UDP	<input type="checkbox"/> 202.97.238.202	49987	<input type="checkbox"/> 82.69.176.148	1027
12:29:47	eth0 > -	TCP	<input type="checkbox"/> 82.9.212.160	3706	<input type="checkbox"/> 82.69.176.151	2967
12:29:50	eth0 > -	TCP	<input type="checkbox"/> 82.9.212.160	3706	<input type="checkbox"/> 82.69.176.151	2967
12:32:55	eth0 > -	UDP	<input type="checkbox"/> 221.208.208.90	36877	<input type="checkbox"/> 82.69.176.148	1026
12:32:55	eth0 > -	UDP	<input type="checkbox"/> 221.208.208.90	36877	<input type="checkbox"/> 82.69.176.148	1027
12:32:55	eth0 > -	UDP	<input type="checkbox"/> 221.208.208.90	36877	<input type="checkbox"/> 82.69.176.151	1026
12:32:55	eth0 > -	UDP	<input type="checkbox"/> 221.208.208.90	36877	<input type="checkbox"/> 82.69.176.151	1027
12:37:54	eth0 > -	UDP	<input type="checkbox"/> 212.23.6.163	53(DOMAIN)	<input type="checkbox"/> 82.69.176.148	32768
12:43:38	eth0 > -	UDP	<input type="checkbox"/> 164.210.120.191	30593	<input type="checkbox"/> 82.69.176.148	1026
12:43:38	eth0 > -	UDP	<input type="checkbox"/> 164.210.120.191	30596	<input type="checkbox"/> 82.69.176.151	1026
12:49:15	eth0 > -	TCP	<input type="checkbox"/> 82.246.144.203	2381	<input type="checkbox"/> 82.69.176.148	445(MICROSOFT-DS)
12:49:18	eth0 > -	TCP	<input type="checkbox"/> 82.246.144.203	2381	<input type="checkbox"/> 82.69.176.148	445(MICROSOFT-DS)
12:51:44	eth0 > -	TCP	<input type="checkbox"/> 82.240.62.74	2032	<input type="checkbox"/> 82.69.176.148	445(MICROSOFT-DS)
12:51:47	eth0 > -	TCP	<input type="checkbox"/> 82.240.62.74	2032	<input type="checkbox"/> 82.69.176.148	445(MICROSOFT-DS)
12:55:30	eth1 > eth0	UDP	<input type="checkbox"/> 192.168.72.16	137(NETBIOS-NS)	<input type="checkbox"/> 192.168.110.110	137(NETBIOS-NS)
12:55:36	eth1 > eth0	UDP	<input type="checkbox"/> 192.168.72.16	137(NETBIOS-NS)	<input type="checkbox"/> 192.168.110.110	137(NETBIOS-NS)
12:55:41	eth1 > eth0	UDP	<input type="checkbox"/> 192.168.72.16	137(NETBIOS-NS)	<input type="checkbox"/> 192.168.110.110	137(NETBIOS-NS)
12:55:46	eth1 > eth0	UDP	<input type="checkbox"/> 192.168.72.16	137(NETBIOS-NS)	<input type="checkbox"/> 192.168.110.110	137(NETBIOS-NS)

SmoothWall Express 3.0-polar-i386
 SmoothWall™ is a trademark of SmoothWall Limited

© 2000 - 2007 The SmoothWall Team
 Credits - Portions © original authors

Smoothwall Express

The award-winning Smoothwall Express open-source firewall—designed specifically to be installed and administered by non-experts—continues its forward development march with a new 3.1 release. Smoothwall Express runs on hardware from early 32-bit Pentiums for those who need a basic firewall to recent 64-bit multi-core systems with gigabytes of RAM for those who need VPNs, HTTP/HTTPS caching and filtering, Snort intrusion detection and ClamAV protections. Addressing a number of problems and deficiencies as well as some housekeeping work, the new version 3.1 refreshed the Linux, iptables, xtables-addons, OpenSSL, Snort and Squid packages, among others.

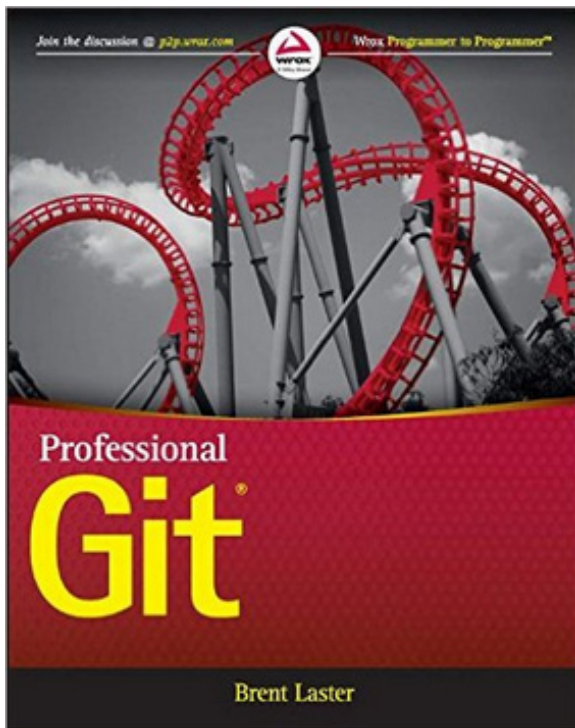
<http://smoothwall.org>



SSH Communications Security's Universal SSH Key Manager

Today's IAM solutions, warns enterprise cybersecurity expert SSH Communications Security, fail to address fully the requirements of trusted access. Organizations lack an efficient way to manage and govern trusted access credentials and have no visibility into the activities that occur within the secure channels that are created for trusted access operations. Leading the charge to fix the issue once and for all, SSH Communications Security announced significant enhancements to its Universal SSH Key Manager (UKM) solution. UKM helps organizations more effectively manage SSH user key-based and encrypted access, control privileged access and enforce defined compliance policies. In addition, UKM helps customers discover, monitor, lockdown, remediate and automate the lifecycle of SSH user key-based access for interactive and machine-to-machine trusts without disrupting existing processes or the need to deploy agents. Updates include application-level policy management, status and compliance reporting and a new SSH Risk Assessment Tool.

<http://ssh.com>



Brent Laster's *Professional Git* (Wrox)

More than 40% of software developers use the massively popular software development tool Git as their primary source control tool. Those new to the Git fold who are looking for a professional, up-to-date guide to get them rolling have a new resource in Brent Laster's new book *Professional Git*. Laster's

Wrox-published title is more than just a development manual: it gets users into the "Git mindset". The book offers extensive discussion of corollaries to traditional systems as well as considerations unique to Git to help one draw upon existing skills while looking out—and planning for—the differences. Connected labs and exercises are interspersed at key points to reinforce important concepts and deepen understanding, while a focus on the practical goes beyond technical tutorials to help users integrate the Git model into real-world workflows. This book instructs users how to harness the power and flexibility of Git to streamline the development cycle. <http://wrox.com>

Please send information about releases of Linux-related products to newproducts@linuxjournal.com or New Products c/o Linux Journal, PO Box 980985, Houston, TX 77098. Submissions are edited for length and content.

[RETURN TO CONTENTS](#)

Cellular Man-in- the-Middle Detection with SITCH



PREVIOUS
New Products

NEXT
Feature:
Managing Docker
Instances with Puppet



Use a Raspberry Pi and inexpensive components to detect cellular man-in-the-middle attacks.

ASH WILSON



The technical and financial barriers for entry into the world of cell-phone interception technologies seem to be on a race to the bottom. Building a device for intercepting cell-phone calls is something one can accomplish without a deep knowledge of GSM networks for around \$600.

Sometimes labeled as IMSI (International Mobile Subscriber Identity) catchers, Cell Site Simulators or Evil BTS (Base Transceiver Station), these devices can be used to identify an individual cell-phone account holder by fooling the phone into transmitting the IMSI, which is a unique ID burned into your phone's SIM card. Moreover, these devices can be used to record cell-phone conversations and SMS messages. To complicate the issue, cell phones, by default, don't have a method for ascertaining the trustworthiness of the BTS before association (this is specifically a problem with 2.5G GSM networks). The implications are serious for everyone from foreign media correspondents to CFOs, where an intercepted conversation could compromise the safety of a source or lead to insider trading or even market manipulation.

Without a doubt, the use of such a device is quite illegal. Detection, however, proves challenging. Without installing software onto your smartphone to interrogate the cell radio—see AIMSICD (<https://github.com/CellularPrivacy/Android-IMSI-Catcher-Detector>) and Femto Catcher (<https://github.com/iSECPartners/femtocatcher>)—you don't have a great way to know if your phone is associating with a known good BTS. Established open-source detection methods often have revolved around the use of Software-Defined Radios (SDRs) in a dedicated piece of hardware or installing software on the phone itself to put the handset into airplane mode in the event that a questionable BTS association occurs.

Rogue BTS detection has found its way into commercial offerings with PwnieExpress and Bastille Networks, but you're not here for a commercial product pitch, and the author doesn't work for either of those companies. The author is pitching SITCH, which stands for Situational Information from Telemetry and Correlated Heuristics. SITCH is open-source, the cost per-sensor is around \$150, and you easily can source the parts from your favorite maker-oriented electronics vendor.

Current Detection Methods

Before looking at how SITCH works and how to set it up, let's have a closer look at currently available methods for solving the problem of detection. One of the projects that inspired the early design of SITCH is Pedro Cabrera's FakeBTS (<http://fakebts.com/en>). It uses a Bash script to coordinate and analyze output from Airprobe and Wireshark to track nearby BTSes. This is an SDR-centric approach, which lends itself to using inexpensive hardware and takes more of an objective approach to the detection of rogue BTSes. Other methods, like the Android IMSI Catcher Detector (AIMSICD), involve interrogating the phone's cell radio and, therefore, produce a more subjective analysis, based on the radio's preference of nearby BTSes for association. These represent two methods: SDR-based scanning and GSM radio interrogation. Both of these methods are incorporated in SITCH.

Solution Proposition

SITCH Overview Now, let's consider how these methods come together in SITCH. SITCH uses an SDR device for tracking the observed power of GSM channels. The SDR USB dongle used in development is the RTL-SDR-based NESDR XTR from NooElec. The open-source software tool used to operate the SDR dongle and process the signal is called Kalibrate. Kalibrate typically is used for determining the frequency offset for an SDR device. This is necessary because the tuner components in software-defined radios are notorious for drifting high or low, sometimes just because of a variation in ambient temperature. I'm not using Kalibrate for determining frequency offset here though.

Kalibrate produces a number representing the power of the signal for each channel it detects. For the remainder of this article, I refer to this channel as ARFCN, which stands for Absolute Radio Frequency Channel Number. Within each ARFCN, there is a frequency correction channel. This is what GSM radios use to calibrate themselves. Think of a musician using a tuning fork as a reference pitch for tuning a horn. The Frequency Correction Channel (FCCH) is what Kalibrate uses to produce a list of ARFCNs.

The SDR approach takes around seven minutes to scan an entire GSM band, and it can positively detect when a femtocell goes live nearby. Femtocells are the range-extender devices that your cell-phone provider will

sell you if you have bad reception indoors. Although these devices often are legitimate, they haven't proven invulnerable. Live hacking of a femtocell has been demonstrated (<https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2013/august/femtocell-presentation-slides-videos-and-app>), and it's just as effective as an evil BTS at capturing communications traffic. For a more subjective reading, SITCH interrogates a GSM radio to determine the BTSes it prefers to associate with, which takes into account more than just signal strength.

The use of a GSM radio can get results in seconds, which is far better than waiting the seven minutes required for the SDR scan, and with more detailed information than you get using Kalibrate. Where SITCH's SDR approach is lacking is in producing information you can use to identify a specific provider's network, like the Mobile Country Code (MCC) and Mobile Network Code (MNC), which are used to identify a specific cellular network service provider. The information provided by the GSM radio goes even further by providing MCC and MNC along with Location Area Code (LAC) and CellID (CID), and when these network identifiers are combined (MCC+MNC+LAC+CID), you get the Cell Global ID (CGI). You never should see the same CGI in two different locations.

The actual detection process happens in two stages. The first part occurs within the SITCH sensor itself. The information gathered is compared against two data feeds. One feed is derived from the FCC license database, which tells what frequencies are licensed to each provider and the geo-location of the tower permitted to operate on that frequency. The second data feed is the OpenCellID database (<http://opencellid.org>). This is a crowd-sourced feed of observed BTSes. Using these two feeds with the information you collect, you can determine the following:

- Is the observed ARFCN licensed to operate in this area? (ARFCN comes from both SDR and GSM radio observations.)
- Is the observation of this CGI in this area corroborated by the OpenCellID feed (comparing GSM findings and OpenCellID database)?
- Has there been a change in preferred BTS (tracking the GSM radio's preferred BTS)?

- Has an ARFCN been observed over the site threshold I set (able to set a per-sensor ARFCN power threshold)?

In addition to tracking cellular network information, functionality recently has been added to detect GPS spoofing (<http://www.rtl-sdr.com/spoofing-gps-locations-with-low-cost-tx-sdrs>), using GeolIP and a GPS dongle. GPS spoofing has a great potential for mischief, especially if used to defeat geolocation-based phone unlocking like Google's Trusted Places (<http://www.androidcentral.com/how-add-trusted-place-android-50-lollipop>).

The second method of detection happens in the service side of SITCH. I'm using a time-series database to track measurements over time, and I can use this to find anomalies. This is especially useful for tracking ARFCN power as reported by Kalibrate.

SITCH System Details SITCH was designed so that once you have the back-end services set up, it is as simple as plugging components into a Raspberry Pi 2, imaging and installing an SD card, and providing power and connectivity to the device. Device updates are managed by a service called Resin.io, so ideally, you never have to touch the device again, except to decommission it. No more SD card re-imaging to update the software—it's all delivered automatically, to all of your sensors, within minutes of building the new version of the software. All the telemetry information flows up to the service (which you host with your favorite cloud provider). Alerts generated by the system are delivered through Slack, and you optionally can forward the collected information to the log aggregation or SIEM system of your choice, provided there's a Logstash output plugin that will facilitate the information delivery for you.

The service side of the SITCH system is composed of a few components. Resin.io is used for managing the device software and runtime variables. Elasticsearch-Logstash-Kibana (ELK stack) is used for aggregation and storage. Graphite and InfluxDB are interchangeable in the SITCH service. However, testing uncovered the hazard of using Graphite/Whisper, which allocates files for the entire lifecycle of a metric as soon as it's first observed, in an environment where the metric namespace can rapidly expand. Slack is used for alerting. Vault (<https://www.vaultproject.io>) is used for the secure distribution of certificates and keys to sensors.

FEATURE: Cellular Man-in-the-Middle Detection with SITCH

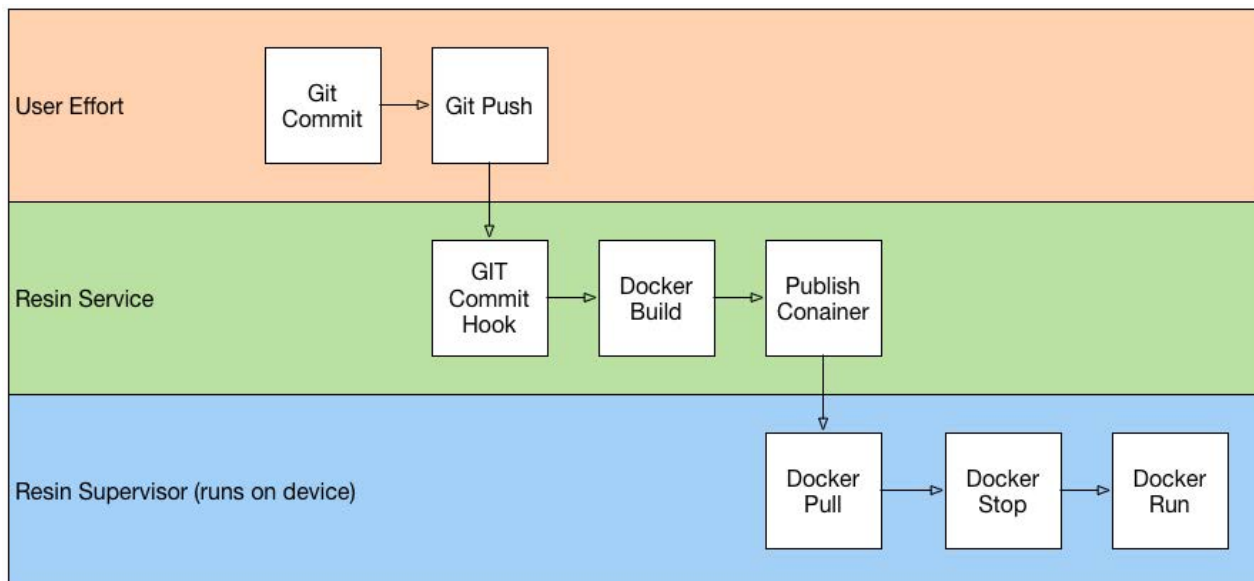


Figure 1. One code push to Resin causes all sensors to update, hands-free.

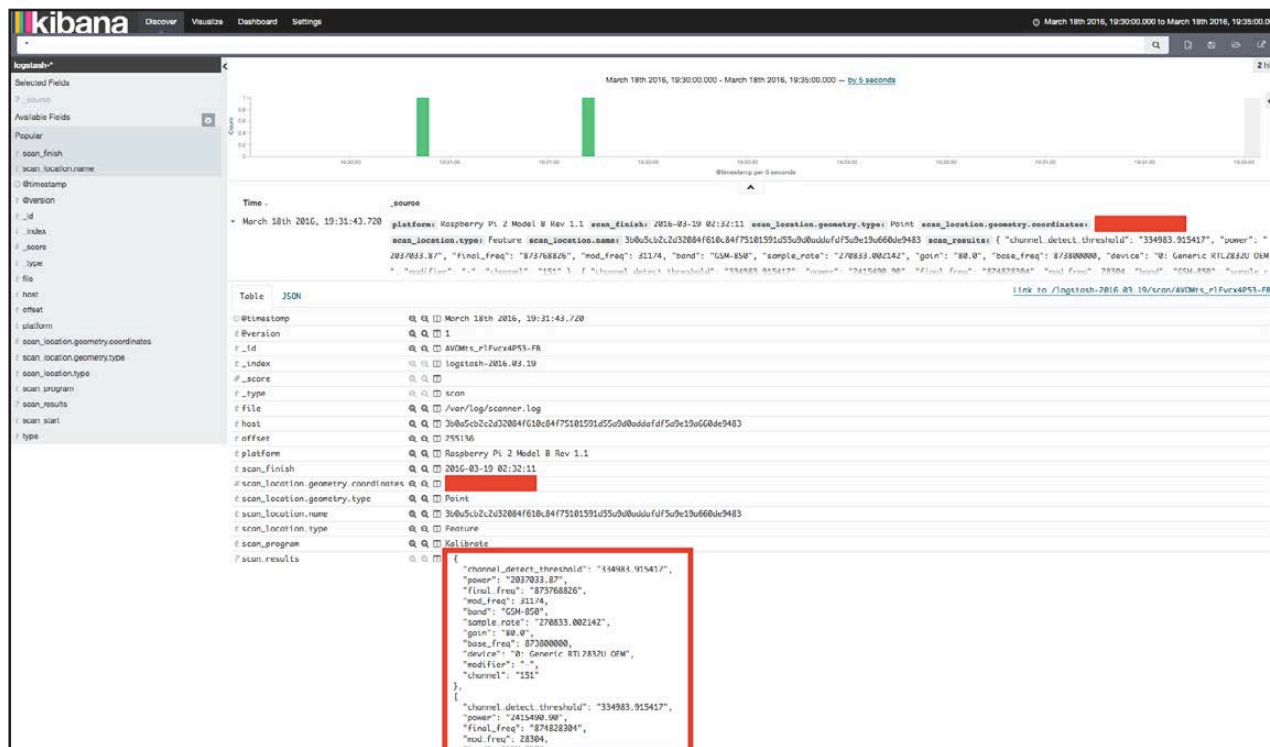


Figure 2. Kalibrate Scan Results Viewed in Elasticsearch

Although it sounds like a lot to manage, much of this has been containerized and automated to get you up and running rapidly.

The SITCH sensor itself is based on the Raspberry Pi 2 platform. SDR

functionality is provided by a USB RTL-SDR device (<http://www.nooelec.com/store/sdr/sdr-receivers/nesdr-xtr-rtl2832u-e4000.html>). The GSM modem used in testing is a SIM808, but the AT command set used for interacting with the modem is general enough that many GSM modems will work.

Putting It All Together

Here's the sensor parts list:

- 16GB MicroSD card (faster is better).
- NooElec NESDR XTR (NESDR Mini will work too, but only for the lower-frequency 850–900MHz bands).
- SIM808 GSM breakout board.
- GSM antenna (frequently sold with the SIM808).
- Some SIM808 modules require a lithium-ion battery.
- ND-105C GlobalSat USB GPS dongle.
- USB-to-serial RPi console adapter (consider Adafruit product 954).
- Power supply for the Raspberry Pi ($\geq 2A$).
- USB cable for relocating the SDR dongle (due to its size, it can block other USB ports).
- USB cable for providing power to the GSM modem.
- Ethernet cable or Wi-Fi adapter.

Services you'll need logins for:

- Choose a cloud provider. Nothing here is provider-specific. You just need to be able to instantiate Linux instances.

- GitHub: set up multi-factor authentication (MFA).
- Resin.io: <https://resin.io> (use MFA here as well).
- OpenCellID: <http://opencellid.org>.
- Slack.
- Twilio (API credentials).
- Your favorite domain registrar (as long as it provides DNS too).
- Docker Hub: <https://hub.docker.com> (if you plan on modifying any of the base images).

Setting Up the SITCH Service Before getting started, a few caveats. This walk-through is going to provide you with a demo-grade service. You're urged to consider using Kubernetes, Mesos + Marathon, or another more resilient platform to get the benefit of a more self-healing application. That being said, the components are all containerized, so restarting pieces in the event things get weird is trivial. For the sake of brevity, some common administrative tasks are not covered in detail. You can find more documentation and troubleshooting information at <http://sitch.io>.

Instance Creation Create one Linux instance with at least 4GB of RAM and 8GB of disk space on the root volume, and add a second volume with at least 40GB of space. This demo relies on Docker, not any specific Linux distribution. Allocate a static IP to the instance and give it a DNS name. Initially, you need only SSH access. Make sure that your instance is only reachable via SSH from your current IP address. Once the instance is alive, ssh in, format the 40GB volume with XFS, and mount the 40GB volume under /opt/shared.

Obtaining Certificates I use EFF's Certbot to obtain certificates for the web server portion of the service (<http://letsencrypt.readthedocs.io/en/latest/install.html?highlight=docker#running-with-docker>). Open up TCP ports 443 and 80 for inbound access so that the Let's Encrypt service can

verify your control of your server's DNS name. Next, run this command:

```
docker run -it --rm \
-p 443:443 -p 80:80 \
--name certbot \
-v "/etc/letsencrypt:/etc/letsencrypt" \
-v "/var/lib/letsencrypt:/var/lib/letsencrypt" \
quay.io/letsencrypt/letsencrypt:latest \
certonly
```

This runs the certbot container image, which will walk you through the process of obtaining a certificate for your environment. Close TCP port 80. You won't need it again until you renew the certificates. You also should consider only leaving TCP port 443 open to IPs where your sensors will live.

Setting Up Your Own Vault I use Vault by Hashicorp to store the crypto material for securing the sensor-to-service communication. Start up Vault, mounting in the certificates created in the prior step:

```
docker run -d \
--cap-add=IPC_LOCK \
-p 8200:8200 \
-v /etc/letsencrypt/:/etc/letsencrypt/ \
-e 'VAULT_LOCAL_CONFIG={"backend": {"file": {"path":
  ➔"/vault/file"}}, "listener":{"tcp":{"address":"0.0.0.0:8200"
➔,"tls_cert_file": "/etc/letsencrypt/live/YOUR_DOMAIN_NAME_HERE/
➔fullchain.pem", "tls_key_file":"/etc/letsencrypt/live/
➔YOUR_DOMAIN_NAME_HERE/privkey.pem"}}, "default_lease_ttl":
  ➔"7200h", "max_lease_ttl": "7200h"}' \
--name sitch_vault \
vault server
```

Replace `YOUR_DOMAIN_NAME_HERE` in the above command with the DNS name of your server, which is the same name that you used in the Certbot wizard, above. Running `docker ps` should confirm that the vault service is now up and running. Next, you need to unseal the vault


```

root@ip-172-31-25-115:~# docker exec sitch_vault vault init --tls-skip-verify
Unseal Key 1: 2g5                JQB
Unseal Key 2: +Qt                hoC
Unseal Key 3: J8t                2UD
Unseal Key 4: ln1                y4E
Unseal Key 5: SL1                lEF
Initial Root Token: 43          58

Vault initialized with 5 keys and a key threshold of 3. Please
securely distribute the above keys. When the Vault is re-sealed,
restarted, or stopped, you must provide at least 3 of these keys
to unseal it again.

Vault does not store the master key. Without at least 3 keys,
your Vault will remain permanently sealed.
root@ip-172-31-25-115:~# █

```

Figure 3. Output from `docker exec sitch_vault vault init --tls-skip-verify`

and obtain a root token.

To unseal the vault, start with this: `docker exec sitch_vault vault init --tls-skip-verify` You'll see something like Figure 3.

To unseal the vault, run this command:

```
docker exec -it sitch_vault vault unseal --tls-skip-verify
```

That will result in a prompt requesting a key. Copy/paste one from above. Do this three times total, using a different unseal key each time, and the vault will unseal. You should see output from the final command that reads: `Sealed: false`. Record your Initial Root Token in your password manager.

Populating Vault with Keys Log delivery uses Filebeat and Logstash. These require certificates for operation. Fortunately, the process for generating and uploading it has been automated. First, you'll need to open TCP port 8200 from the world, into your server. Next, you'll run the SITCH

Self-Signed Seeder (https://hub.docker.com/r/sitch/self_signed_seeder):

```
docker run -it \
-e VAULT_URL=$VAULT_URL \
-e VAULT_TOKEN=$VAULT_TOKEN \
-e LS_CLIENTNAME=$LS_CLIENTNAME \
-e LS_SERVERNAME=$LS_SERVERNAME \
docker.io/sitch/self_signed_seeder
```

This will cause the Vault to be populated with certs and keys for sensor and service. Make sure that your LS_SERVERNAME is set to the same hostname as included in the VAULT_URL, because these containers are running on the same host. There are two tokens mentioned in the (quite verbose) output at the end: Client token and Server token. Look under each section and grab the token labeled client-token (Figure 4).

Your Vault is now seeded with self-signed certs and keys for Logstash.

```
Status: Downloaded newer image for sitch/self_signed_seeder:latest
Generating a 2048 bit RSA private key
.....+++
writing new private key to '/usr/share/easy-rsa/pki/private/ca.key.XXXXoPAFJF'
-----
Generating a 2048 bit RSA private key
.....+++
writing new private key to '/usr/share/easy-rsa/pki/private/cabron.sitch.io.key.XXXXhnFBGg'
-----
Using configuration from /usr/share/easy-rsa/openssl-1.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName      :ASN.1 12:'sitch.io'
Certificate is to be certified until Nov  5 22:32:15 2017 GMT (365 days)

Write out database with 1 new entries
Data Base Updated
Generating a 2048 bit RSA private key
.....+++
writing new private key to '/usr/share/easy-rsa/pki/private/sensor.sitch.io.key.XXXXciMMDg'
-----
Using configuration from /usr/share/easy-rsa/openssl-1.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName      :ASN.1 12:'sensor.sitch.io'
Certificate is to be certified until Nov  5 22:32:15 2017 GMT (365 days)

Write out database with 1 new entries
Data Base Updated
Vault url: https://          :8200/
Setting server key material...

Client token is:
{u'lease_id': u'', u'warnings': None, u'wrap_info': None, u'auth': {u'lease_duration': 12960000, u'policies': [u'client', u'default'], u'cli
ent_token': u'67f64          28b', u'accessor': u'07e          0c5ca', u'renewable': True, u'metadata':
None}, u'lease_duration': 0, u'request_id': u'179/abf          1e5', u'data': None, u'renewable': False}

Server token is:
{u'lease_id': u'', u'warnings': None, u'wrap_info': None, u'auth': {u'lease_duration': 12960000, u'policies': [u'default', u'server'], u'cli
ent_token': u'f0bf64          54b', u'accessor': u'6c3          aad75', u'renewable': True, u'metadata':
None}, u'lease_duration': 0, u'request_id': u'cdcf3bee-          19', u'data': None, u'renewable': False}
```

Figure 4. Grabbing the Token

Configuring Storage for Scans Set up the Elasticsearch and Kibana portions of the ELK stack in whatever manner makes the most sense for your environment. If you're using AWS, you can accelerate this by using the AWS ElasticSearch Service. Use ElasticSearch version 2.3 or greater. Retain the URLs for accessing Kibana and Elasticsearch.

Configuring Logstash for Ingestion Logstash is used for ingestion of telemetry from the sensors. There's a SITCH spin of the Logstash container—follow the instructions in the README (found at <https://hub.docker.com/r/sitch/logstash>) to set your environment variables for running the container. Before you complete this step, you'll need access to Slack to create a webhook for notification. GRAPHITE_HOST is the name of your server, and GRAPHITE_PORT will be 2003. The Graphite line protocol is used for delivering time-series information, which is understood by InfluxDB. Finally, open port 5001 so that the Filebeat log shipper can connect to Logstash.

Building the SITCH Data Feed Now let's build the SITCH feed, which

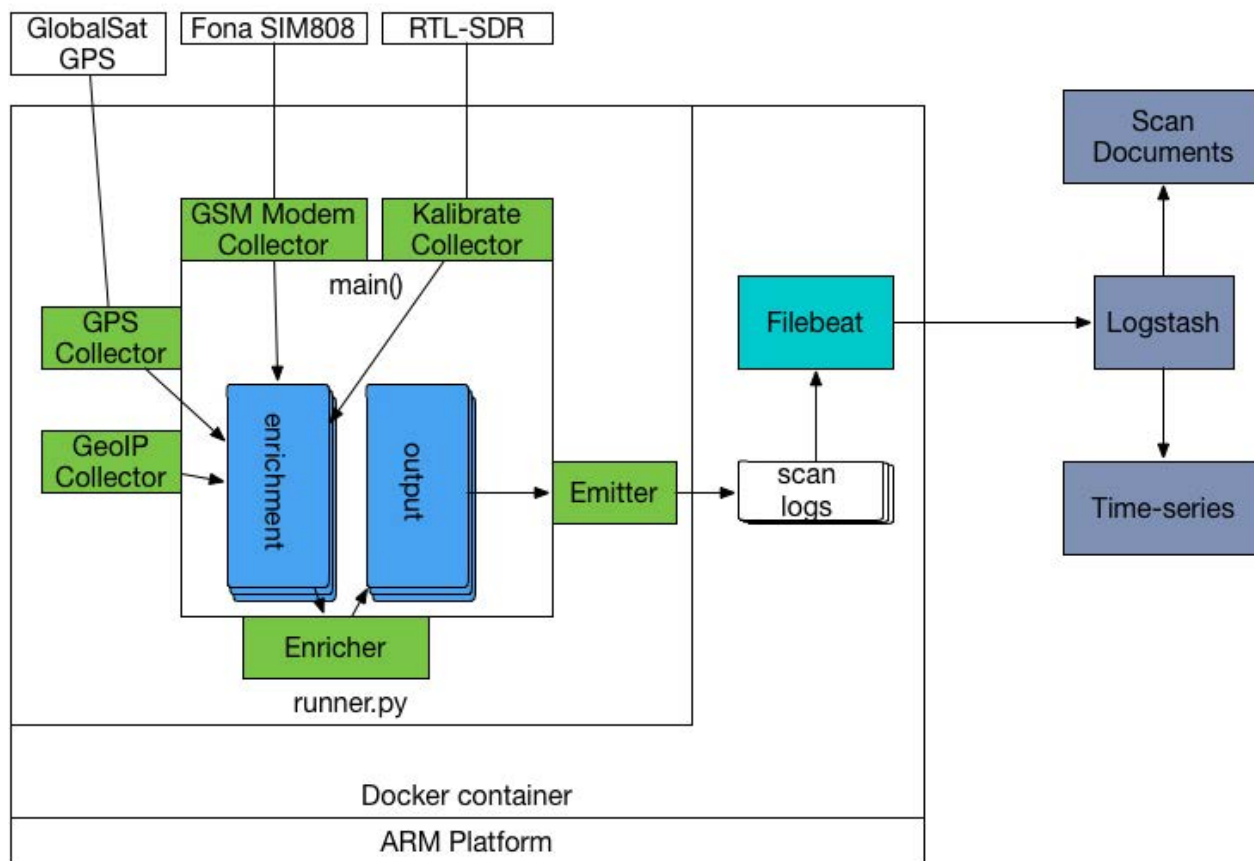


Figure 5. Diagram of Sensor Software and Enrichment Information Flow

is composed of the OpenCellID database, enriched with information from the Twilio API and the FCC license database. Locate your OpenCellID API key and your Twilio SID and token for API access. Run the container according to the README at https://hub.docker.com/r/sitch/feed_builder. This job will take quite a while to run. If your curiosity demands to see progress, run `docker logs -f CONTAINER_NAME` to see the feed builder's progress. Don't stop it mid-job, or you may have to wait until tomorrow to try again. The OpenCellID database can be retrieved only once daily, per API key. So let it roll until it's done.

Configuring the Time-Series Database Any time-series database that supports the Graphite line protocol should work with SITCH. For the purposes of this demo, I'm using InfluxDB. Make sure that TCP ports 2003, 8083, 8086 and 1000 are accessible from the server itself, using its own public IP address. Start InfluxDB with this command:

```
docker run -d \  
--name sitch_influx \  
-p 8083:8083 \  
-p 8086:8086 \  
-p 2003:2003 \  
-e INFLUXDB_GRAPHITE_ENABLED=true \  
-v /opt/shared/influxdb:/var/lib/influxdb \  
influxdb
```

The last step in preparing the persistence layer is Chronograf. I'm using Chronograf to visualize the information stored in InfluxDB. Start it like this:

```
docker run -d \  
-p 10000:10000 \  
--name sitch_chronograf \  
chronograf
```

Instructions for running the SITCH front-end web server container are at <https://hub.docker.com/r/sitch/web>. Follow the instructions there and confirm that you can download `https://YOUR_SERVER_NAME/310.csv.gz`. This will confirm that your feed is built and available for your sensors. Now

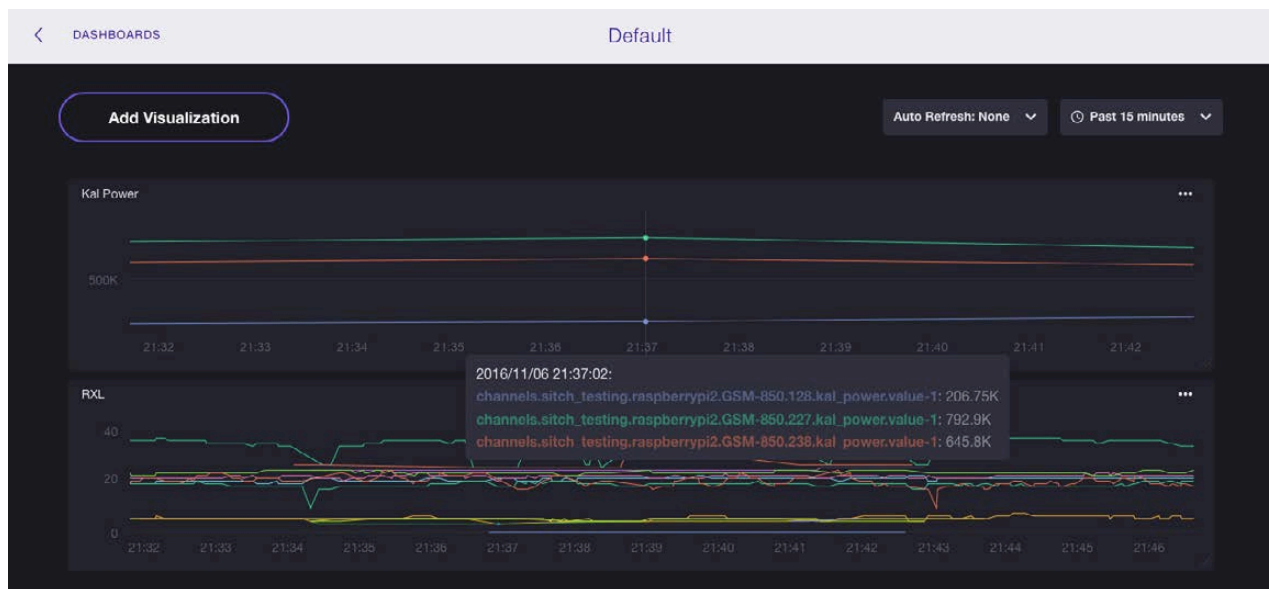


Figure 6. Time-Series Data from Kalibrate and GSM Modem Graphed in Chronograf

is a great time to make sure that the ports mapped in for the web container are accessible to you, from your IP address. It's not an awful idea to take it a step further and employ authentication in the web container, or a VPN for accessing it, but that's outside the scope of this demo.

Building the SITCH Sensor Log in to <https://resin.io> and create your first project. Name it whatever you like. Click "Download ResinOS" to download the image for your Raspberry Pi. Follow the directions on-screen to image your MicroSD card. Insert the card into the Raspberry Pi 2, and plug in the GPS, SDR and Ethernet cable. Use the USB console cable to attach the SIM808 module to the Pi. Black goes to ground, red to vio, green to rx and white to tx. Give the Pi power and, in a few minutes, verify that the device has registered with your application. Next, set the following environment variables in your Resin project:

- `FEED_URL_BASE` — this should be `https://YOUR_SERVER_NAME/`.
- `GSM_MODEM_BAND` — try `GSM850_MODE`.
- `KAL_BAND` — try `GSM850`.
- `KAL_GAIN` — if you're indoors and have bad reception, try 60 or 80.

- `KAL_THRESHOLD` — threshold for Kalibrate power alarm, try 1000000 for starters.
- `LOCATION_NAME` — text string, no spaces.
- `LOG_HOST` — hostname and port, colon-separated. The port for the Logstash instance created earlier is 5001.
- `MCC_LIST` — this is a comma-separated list of MCCs. USA should be set to "310,311,316".
- `MODE` — set this to "clutch" to enter a holding state before starting services. It's for debugging. Set it to anything else to run normally.
- `STATE_LIST` — comma-separated list of states to load FCC feed data for. California and Texas would be "CA,TX".
- `VAULT_TOKEN` — this is the client's `client_token` you retained from seeding the Vault.
- `VAULT_URL` — this is the same URL you used when you ran the seeder.
- `VAULT_PATH` — set this to "secret/client".

Clone the sensor repository locally with `git clone https://github.com/sitch-io/sensor`. Descend into the `sensor/` directory and add your Resin application as a remote Git repository using the `git remote add...` command in the upper-right corner of the screen when viewing the Resin application page in your browser.

Push the sensor software to Resin with `git push resin master`. You'll notice that it attempts to build the sensor software before accepting the push. After a few minutes, you will see an ASCII art unicorn in your terminal. Within a couple minutes, the application will begin to download to the sensor. Depending on the model of the GSM modem you're using, you may need to locate and press the GSM modem's power button.

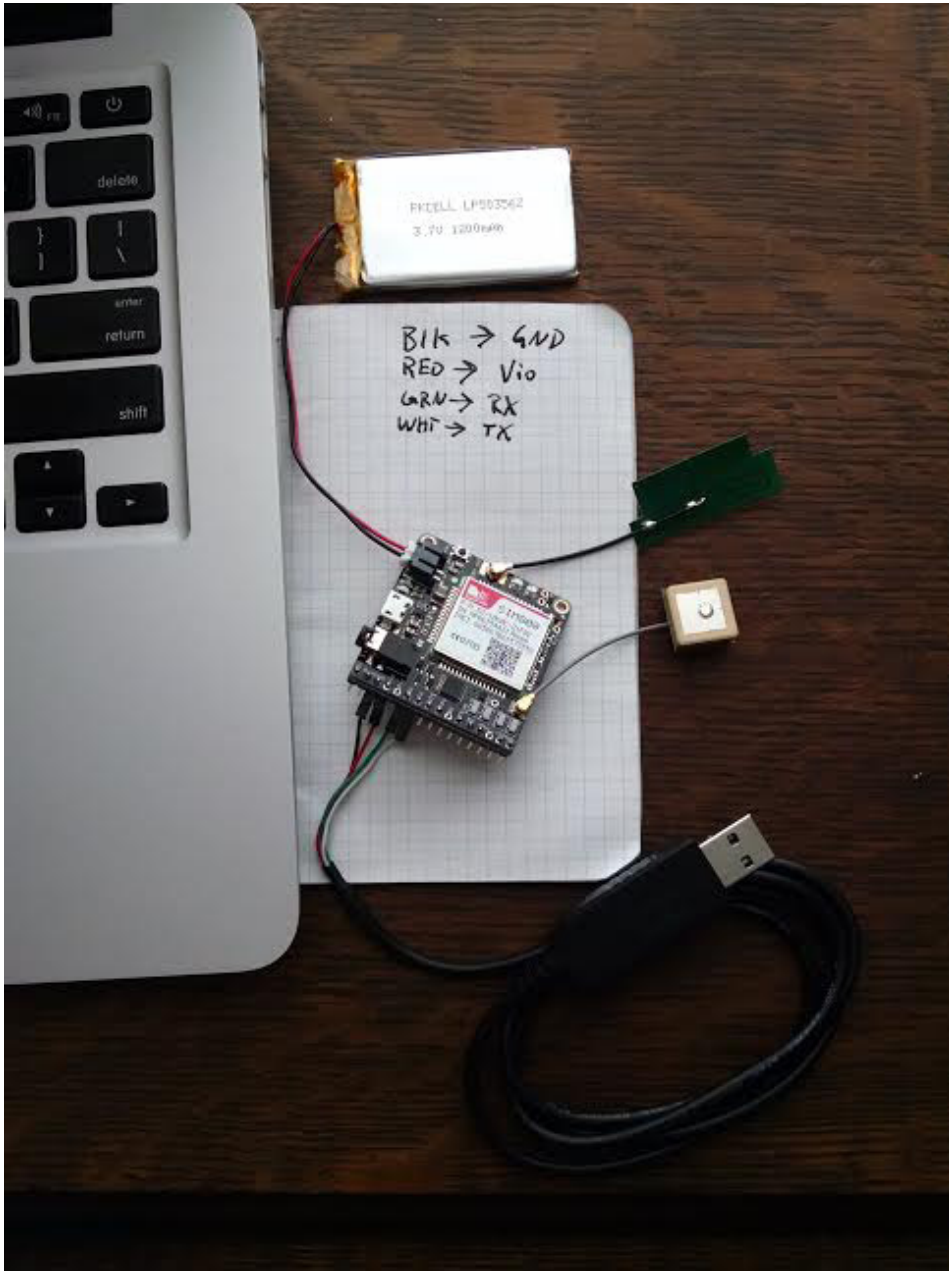


Figure 7. SIM808 Module with USB Console Cable Attached

As the sensor is powering up, you'll see a lot of information scroll by. Most important, you need to see that the device detector has picked up your GPS and GSM modem. If you're not using a SIM808 module, the README for the sensor repository has instructions for adding the proper init string so that your modem can be recognized (<https://github.com/sitch-io/sensor>).

As the system is currently configured, you should be able to receive alerts in Slack based on alarms fired from within the sensor itself. The higher-level correlation, for instance, ARFCN power trends, can be

accomplished by configuring Chronograf (<https://www.influxdata.com/get-started/visualizing-data-with-chronograf>) and Kapacitor (<https://www.influxdata.com/get-started/configuring-alerts-with-kapacitor>) to visualize and alert on the information stored in the InfluxDB time-series database. Start with monitoring `kal_power` readings over time, and go from there.

Postscript

Finally, this is where “ease of use” begins. In order to add another sensor, you need to assemble only one, and install the Resin OS, just like you did with the first one. Tweak your sensor metadata in Resin if you need to. You’ll likely want to change the `site_name` environment variable to facilitate distinction between sensors in the data you’re amassing. Go ahead and turn up one at every office. Just make sure you have the storage provisioned to support it. If you need to integrate this with your log management system, use the image found at <https://hub.docker.com/r/sitch/logstash> as a base, and adapt the configuration to accommodate your log management system. This is very much beta-grade software and feedback is greatly appreciated. Feel free to file an issue against the appropriate GitHub project, all of which are accessible via <https://github.com/sitch-io>. ■

Ash Wilson is a native of Apison, Tennessee, and currently resides in San Francisco, California. He entered the security domain through systems and network engineering, spent a number of years in network security tooling and integration, and he currently works in R&D for CloudPassage.

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

MANAGING DOCKER INSTANCES with PUPPET

Docker provides a powerful tool for creating lightweight images and containerized services, while Puppet provides the means to deploy and manage those same images and containers as a standard part of the configuration management lifecycle. Whether you're working in the cloud or the data center, this one-two punch is a real knockout!

In this in-depth article, you'll learn how to use Puppet roles and profiles to assign Docker images and containers to an unlimited number of nodes based on standardized naming conventions. If you're not careful, you also might learn a few tips and tricks about Vagrant, Linux hostnames and SSH along the way.

TODD A. JACOBS

PREVIOUS



Feature: Cellular
Man-in-the-Middle
Detection with SITCH

NEXT
Doc Searls' EOF



My previous article, “Provisioning Docker with Puppet” in the December 2016 issue, covered one of the ways you can install the Docker service onto a new system with Puppet. By contrast, this article focuses on how to manage Docker images and containers with Puppet.

Container management with Puppet allows you to do a number of things that become ever more important as an organization scales up its systems, including the following:

1. Leveraging the organization’s existing configuration management framework, rather than using a completely separate process just to manage Docker containers.
2. Treating Docker containers as “just another resource” to converge in the configuration management package/file/service lifecycle.

Reasons for Integrating Docker with Puppet

There are three core use cases for integrating Docker with Puppet or with another configuration management tool, such as Chef or Ansible:

1. Using configuration management to provision the Docker service on a host, so that it is available to manage Docker instances.
2. Adding or removing specific Docker instances, such as a containerized web server, on managed hosts.
3. Managing complex or dynamic configurations inside Docker containers using configuration management tools (for example, Puppet agent) baked into the Docker image.

“Provisioning Docker with Puppet”, in the December 2016 issue of *LJ*, covered the first use case. This article is primarily concerned with the second.

3. Installing Docker containers automatically based on hostname, node classification or node-specific facts.
4. Orchestrating commands inside Docker containers on multiple hosts.

Although there certainly are other ways to achieve those goals (see the Picking a Toolchain sidebar), it takes very little work to extend your existing Puppet infrastructure to handle containers as part of a node's role

Picking a Toolchain

Why focus on container management with Puppet? There certainly are other ways to manage Docker instances, containers and clusters, including some native to Docker itself. As with any other IT endeavor, your chosen toolchain both provides and limits your capabilities. For a home system, your choice of toolchain is largely a matter of taste, but in the data center, it's often better to leverage existing tools and in-house expertise whenever possible.

Puppet was chosen for this series of articles because it is a strong enterprise-class solution that has been widely deployed for more than a decade. However, you could do much the same thing with Chef or Ansible if you choose.

Puppet also was selected over other container orchestration tools because many large organizations already make use of at least one configuration management tool. In many cases, it's advantageous to include container management within the existing toolchain rather than climbing the learning curve of a more specialized tool, such as Kubernetes.

If you already use Puppet, Chef or Ansible in your data center, getting started with container management by extending your current toolset is probably smart money. However, if you find yourself bumping up against the limitations of your configuration management tool, you may want to evaluate other enterprise-class solutions, such as Apache Mesos, Kubernetes or DC/OS.

or profile. That's the focus for this article.

Creating a Test Environment

To follow along with the code listings and examples in the remainder of this article, ensure that Vagrant and VirtualBox are already installed. Next, you'll prepare a set of provisioning scripts to configure a test environment on an Ubuntu virtual machine.

Preparing Your Provisioning Scripts Create a directory to work in, such as ~/Documents/puppet-docker. Place the Vagrantfile and docker.pp manifest within this directory (see Listings 1 and 2).

The Vagrantfile is a Ruby-based configuration file that Vagrant uses to drive one or more "providers". Vagrant supports VirtualBox, Hyper-V

Listing 1. Vagrantfile

```
Vagrant.configure(2) do |config|
  # Install the official Ubuntu 16.04 Vagrant guest.
  config.vm.box = 'ubuntu/xenial64'

  # Forward port 8080 on the Ubuntu guest to port
  # 8080 on the VirtualBox host. Set the host value
  # to another unused port if 8080 is already in
  # use.
  config.vm.network 'forwarded_port',
                    guest: 8080,
                    host: 8080

  # Install the puppet agent whenever Vagrant
  # provisions the guest. Note that subsequent
  # releases have renamed the agent package from
  # "puppet" to "puppet-agent".
  config.vm.provision 'shell', inline: <<-SHELL
    export DEBIAN_FRONTEND=noninteractive
    apt-get -y install puppet
  SHELL
end
```


and Docker by default, but it also supports many other providers, such as VMware Fusion, DigitalOcean, Amazon AWS and more. Because the goal is to simulate the management of the Docker `dæmon`, images and containers on a full-fledged OS, let's focus on the cross-platform VirtualBox provider.

Note that this particular Vagrantfile (Listing 1) installs Puppet 3.8.5, which is the currently supported version for Ubuntu 16.04.1 LTS. Different versions are available as Puppet Enterprise packages or Ruby gems, but this article focuses on the version provided by Ubuntu for its current long-term support release.

Docker.pp is a Puppet manifest that uses a declarative syntax to bring a node into a defined state. The `docker.pp` manifest (Listing 2) makes use of an officially supported Puppet Forge module that takes care of a great deal of low-level work for you, making the installation and management of the Docker `dæmon`, images and containers easier than rolling your own.

Listing 2. `docker.pp`

```
# Most Vagrant boxes use 'vagrant' rather than
# 'ubuntu' as the default username, but the Xenial
# Xerus image uses the latter.
class { 'docker':
  package_name => 'docker.io',
  docker_users => ['ubuntu'],
}

# Install an Apache2 image based on Alpine Linux.
# Use port forwarding to map port 8080 on the
# Docker host to port 80 inside the container.
docker::run { 'apache2':
  image    => 'httpd:alpine',
  ports    => ['8080:80'],
  require => Class['docker'],
}
```

On some systems, a Puppet manifest can enable a basic Docker setup with only a simple `include 'docker'` statement. However, for this article, you will override specific settings, such as your user name on the guest OS, so that the right user is added to the group that can communicate with the Docker `dæmon`. You also will override the name of the Docker package to install, as you want to use the Ubuntu-specific “`docker.io`” package rather than the upstream “`docker-engine`” package the Puppet module uses by default.

When placing `docker.pp` into the same directory as the Vagrantfile, Vagrant will make the Puppet manifest available inside the virtual machine automatically using its synced folder feature. As you will see shortly, this seemingly minor step can pay automation-friendly dividends when provisioning the guest OS.

Provisioning with Puppet Apply With the Vagrantfile and `docker.pp` stored in your working directory, you’re ready to launch and configure the test environment. Since this article is all about automation, let’s go ahead and script those activities too.

Create the shell script shown in Listing 3 in the same directory as the Vagrantfile. You can name it anything you like, but a sensible name, such as `vagrant_provisioning.sh`, makes it clear what the script does.

Make the script executable with `chmod 755 vagrant_provisioning.sh`, then run it with `./vagrant_provisioning.sh`. This will start and configure the virtual machine, but it may take several minutes (and a great deal of screen output) before you’re returned to the command prompt. Depending on the horsepower of your computer and the speed of your internet connection, you may want to go make yourself a cup of coffee at this point.

When you’re back with coffee in hand, you may see a number of deprecation warnings caused by the Puppet Forge module, but those can be safely ignored for your purposes here. As long as the `docker.pp` manifest applies with warnings and not errors, you’re ready to validate the configuration of both the guest OS and the Docker container you just provisioned.

Believe it or not, with the `docker.pp` Puppet manifest applied by the Puppet agent, you’re already done! You now have a Docker container running Apache, and serving up the default “It works!” document. You

Listing 3. vagrant_provisioning.sh

```
#!/usr/bin/env bash

# Provision an Ubuntu guest using VirtualBox.
vagrant up --provider virtualbox

# Install the officially-supported Docker module
# from the Puppet Forge as a non-root user.
vagrant ssh -c \
    'puppet module install \
    puppetlabs-docker_platform --version 2.1.0'

# Apply our local Docker manifest using the Puppet
# agent. No Puppet Master required!
#
# Note that the modulepath puppet installs to can
# vary on different Ubuntu releases, but this one is
# valid for the image defined in our Vagrantfile.
vagrant ssh -c \
    'sudo puppet apply \
    --modulepath ~/.puppet/modules \
    /vagrant/docker.pp'

# After adding the "ubuntu" user as a member of the
# "docker" group to enable non-root communications
# with the Docker daemon, we deliberately close the
# SSH control connection to avoid unhelpful Docker
# errors such as "Cannot connect to the Docker
# daemon. Is the docker daemon running on this
# host?" on subsequent connection attempts.
vagrant ssh -- -O exit
```

can test this easily on your top-level host with `curl localhost:8080` or at `http://localhost:8080/` in your desktop browser.

Don't be fooled. Although you haven't really done anything yet that couldn't be done with a few lines at the command prompt, you've *automated* it in a consistent and repeatable way. Consistency and

Applying Local Manifests with Puppet Agent

By using `puppet apply` as shown here, you're able to perform the same process that you'd employ in a more traditional client/server Puppet configuration, but without the need to do the following:

1. Configure a Puppet Master first.
2. Manage SSL client certificates.
3. Install server-side modules into the correct Puppet environment.
4. Specify a Puppet environment for the node you want to manage.
5. Define roles, profiles or nodes that will use the manifest.

This is actually one of the key techniques for Puppet testing and an essential skill for running a masterless Puppet infrastructure. Although a discussion of the pros and cons of masterless Puppet is well outside the scope of this article, it's important to know that Puppet does not actually *require* a Puppet Master to function.

repeatability are the bedrock of automation and really can make magic once you extend the process with roles and profiles.

Controlling Docker with Puppet Roles and Profiles

It may seem like a lot of work to automate the configuration of a single machine. However, even when dealing with only a single machine, the consistency and repeatability of a managed configuration is a big win. In addition, this work lays the foundation for automating an unlimited number of machines, which is essential for scaling configuration management to hundreds or thousands of servers. Puppet makes this possible through the "roles and profiles" workflow.

In the Puppet world, roles and profiles are just special cases of Puppet manifests. It's a way to express the desired configuration through composition, where profiles are composed of component modules and

Listing 4. roles_and_profiles.pp

```
#####
# Profiles
#####
# The "dockerd" profile uses a forge module to
# install and manage the Docker daemon. The only
# difference between this and the "docker" class
# from the earlier docker.pp example is that we're
# wrapping it inside a profile.
class profile::dockerd {
  class { 'docker':
    package_name => 'docker.io',
    docker_users => ['ubuntu'],
  }
}

# The "alpine33" profile manages the presence or
# absence of the Alpine 3.3 Docker image using a
# parameterized class. By default, it will remove
# the image.
class profile::alpine33 ($status = 'absent') {
  docker::image { 'alpine_33':
    image      => 'alpine',
    image_tag  => '3.3',
    ensure     => $status,
  }
}

# The "alpine34" profile manages the presence or
# absence of the Alpine 3.4 Docker image. By
# default, it will remove the image.
class profile::alpine34 ($status = 'absent') {
  docker::image { 'alpine_34':
    image      => 'alpine',
    image_tag  => '3.4',
    ensure     => $status,
  }
}

#####
# Roles
#####
# This role combines two profiles, passing
# parameters to add or remove the specified images.
# This particular profile ensures the Alpine 3.3
# image is installed, and removes Alpine 3.4 if
# present.
class role::alpine33 {
  class { 'profile::alpine33':
    status => 'present',
  }

  class { 'profile::alpine34':
    status => 'absent',
  }
}

# This role is the inverse of role::alpine33. It
# calls the same parameterized profiles, but
# installs Alpine 3.4 and removes Alpine 3.3.
class role::alpine34 {
  class { 'profile::alpine33':
    status => 'absent',
  }

  class { 'profile::alpine34':
    status => 'present',
  }
}

#####
# Nodes
#####
# Apply role::alpine33 to any host with "alpine33"
# in its hostname.
node /alpine33/ {
  include ::role::alpine33
}

# Apply role::alpine34 to any host with "alpine34"
# in its hostname.
node /alpine34/ {
  include ::role::alpine34
}

```

then one or more profiles comprise a role. Roles are then assigned to nodes dynamically or statically, often through a `site.pp` file or an External Node Classifier (ENC).

Let's walk through a simplified example of what a roles-and-profiles workflow looks like. First, you'll create a new manifest in the same directory as your Vagrantfile named `roles_and_profiles.pp`. Listing 4 shows a useful example.

Note that all the profiles, roles and nodes are placed into a single Puppet manifest. On a production system, those should all be separate manifests located in appropriate locations on the Puppet Master. Although this example is illustrative and extremely useful for working with masterless Puppet, be aware that a few rules are broken here for the sake of convenience.

Let me briefly discuss each section of the manifest. Profiles are the reusable building blocks of a well organized Puppet environment. Each profile should have exactly one responsibility, although you can allow the profile to take optional arguments that make it more flexible. In this case, the Alpine profiles allow you to add or remove a given Docker image depending on the value of the `$status` variable you pass in as an argument.

A role is the "reason for being" that you're assigning to a node. A node can have more than one role at a time, but each role should describe a singular purpose regardless of how many component parts are needed to *implement* that purpose. In the wild, some common roles assigned to a node might include:

- `role::ruby_on_rails`
- `role::jenkins_ci`
- `role::monitored_host`
- `role::bastion_host`

Each role is composed of one or more profiles, which together describe the purpose or function of the node as a whole. For this example, you

define the `alpine34` role as the presence of the Docker `dæmon` with Alpine 3.4 *and* the absence of an Alpine 3.3 image, but you could just as easily have described a more complex role composed of profiles for NTP, SSH, Ruby on Rails, Java and a Splunk forwarder.

This separation of concerns is borrowed from object-oriented programming, where you try to define nodes through composition in order to isolate the implementation details from the user-visible behavior. A less programmatic way to think of this is that profiles generally describe the *features* of a node, such as its packages, files or services, while roles describe the node's *function* within your data center.

Nodes, which are generally defined in a Puppet Master's `site.pp` file or an external node classifier, are where roles are statically or dynamically assigned to each node. This is where the real scaling power of Puppet becomes obvious. In this example, you define two different types of nodes. Each node definition uses a string or regular expression that is matched against the hostname (or `certname` in a client/server configuration) to determine what roles should be applied to that node.

In the node section of the example manifest, you tell Puppet to assign `role::alpine33` to any node that includes "alpine33" as part of its hostname. Likewise, any node that includes "alpine34" in the hostname gets `role::alpine34` instead. Using pattern-matching in this way means that you could have any number of hosts in your data center, and each will pick up the correct configuration based on the hostname that it's been assigned. For example, say you have five hosts with the following names:

1. foo-alpine33
2. bar-alpine33
3. baz-alpine33
4. abc-alpine34
5. xyz-alpine34

Then, the first three will pick up the Alpine 3.3 role when they contact the Puppet Master, and the last two will pick up the Alpine 3.4 role instead. This is almost magical in its simplicity. Let's see how this type of dynamic role assignment works in practice.

Dynamic Role Assignments Assuming that you've already placed `roles_and_profiles.pp` into the directory containing your Vagrantfile, you're able to access the manifest within the Ubuntu virtual machine. Let's log in to the VM and test it out (Listing 5).

Next, run the `roles_and_profiles.pp` Puppet manifest to see what happens. Hint: it's going to fail, and then you're going to explore why that's a *good* thing. Here's what happens:

```
ubuntu@ubuntu-xenial:~$ sudo puppet apply --modulepath
↳ ~/.puppet/modules /vagrant/roles_and_profiles.pp
Error: Could not find default node or by name with
↳ 'ubuntu-xenial.localdomain, ubuntu-xenial' on node
↳ ubuntu-xenial.localdomain
Error: Could not find default node or by name with
↳ 'ubuntu-xenial.localdomain, ubuntu-xenial' on node
↳ ubuntu-xenial.localdomain
```

Why did the manifest fail to apply? There are actually several

Listing 5. Logging in to the Ubuntu Virtual Machine

```
# Ensure we're in the right directory on our Vagrant
# host.
cd ~/Documents/puppet-docker

# Ensure that the virtual machine is active. There's
# no harm in running this command multiple times,
# even if the machine is already up.
vagrant up

# Login to the Ubuntu guest.
vagrant ssh
```

reasons for this. The first reason is that you did not define any nodes that matched the current hostname of "ubuntu-xenial". The second reason is that you did not define a default to be applied when no other match is found. Puppet allows you to define a default, but in many cases, it's better to raise an error than to get a configuration you weren't expecting.

In this test environment, you want to show that Puppet is able to assign roles dynamically based on the hostname of the node where the Puppet agent is running. With that in mind, let's modify the hostname of the Ubuntu guest to see how a site manifest can be used to configure large clusters of machines appropriately based solely on each machine's hostname.

Changing a Linux Hostname

When changing the hostname on a Linux system, it's important to understand that the `sudo` utility will complain loudly and often if a number of information sources don't agree on the the current hostname. In particular, on an Ubuntu system, the following should all agree:

1. The hostname stored in `/etc/hostname`.
2. The hostname defined for `127.0.1.1` in `/etc/hosts`.
3. The hostname reported by `/bin/hostname`.

If they don't all match, you may see errors such as:

```
> sudo: unable to resolve host quux
```

And in extreme cases, you even may lose the ability to run the `sudo` command. It's best to avoid the situation by ensuring that you update all three data sources to the same value when changing your hostname.

In order to avoid errors with the `sudo` command, you actually need to change the hostname of your virtual machine in several places. In addition, the hostname reported by the `PS1` prompt will not be updated until you start a new shell. The following commands, when run inside the Ubuntu guest, will make the necessary changes:

```
# Must be exported to use in sudo's environment.
export new_hostname="foo-alpine33"

# Preserve the environment or sudo will lose the
# exported variable. Also, we must explicitly
# execute on localhost rather than relying on
# whatever sudo thinks the current hostname is to
# avoid "sudo: unable to resolve host" errors.
sudo \
  --preserve-env \
  --host=localhost \
  -- \
  sed --in-place \
    "s/${HOSTNAME}/${new_hostname}/g" \
    /etc/hostname /etc/hosts
sudo \
  --preserve-env \
  --host=localhost \
  -- \
  hostname "$new_hostname"

# Replace the current shell in order to pick up the
# new hostname in the PS1 prompt.
exec "$SHELL"
```

Your prompt now should show that the hostname has changed. When you re-run the Puppet manifest, it will match the node list because you've defined a rule for hosts that include "alpine33" in the hostname. Puppet then will apply `role::alpine33` for you,

simply because the hostname matches the node definition! For example:

```
# Apply the manifest from inside the Ubuntu guest.
sudo puppet apply \
  --modulepath ~/.puppet/modules \
  /vagrant/roles_and_profiles.pp

# Verify that the role has been correctly applied.
docker images alpine
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
alpine	3.3	6c2aa2137d97	7 weeks ago	4.805MB

To apply this role to an entire cluster of machines, all you need to

Ignore “update_docker_image.sh” Errors

When running the Puppet manifest in the example, you may see several errors that contain the following substring:

```
> update_docker_image.sh alpine:3.4 returned 3 instead of one of [0,1]
```

These errors currently are caused by upstream bugs in the Puppet Docker modules used in the examples. Bugs have been filed upstream, but can safely be ignored for the immediate purposes of this article. Despite the reported error, the Docker images actually still are being properly installed, which you can verify yourself inside the virtual machine with `docker images alpine`.

If you want to track the progress of these bugs, please see:

- <https://github.com/garethr/garethr-docker/issues/607>
- <https://github.com/garethr/garethr-docker/issues/608>

do is ensure they have hostnames that match your defined criteria. For example, say you have five hosts with the following names:

1. foo-alpine33
2. bar-alpine33
3. baz-alpine33
4. abc-alpine33
5. xyz-alpine33

Then, the single node definition for `/alpine33/` would apply to all of them, because the regular expression matches each of their hostnames. By assigning roles to *patterns* of hostnames, you can configure large segments of your data center simply by setting the proper hostnames! What could be easier?

Reassigning Roles at Runtime Well, now you have a way to assign a role to thousands of boxes at a time. That's impressive all by itself, but the magic doesn't stop there. What if you need to reassign a system to a different role?

Imagine that you have a box with the Alpine 3.3 image installed, and you want to upgrade that box so it hosts the Alpine 3.4 image instead. In reality, hosting multiple images isn't a problem, and these images aren't mutually exclusive. However, it's illustrative to show how you can use Puppet to add, remove, update and replace images and containers.

Given the existing node definitions, all you need to do is update the hostname to include "alpine34" and let Puppet pick up the new role:

```
# Define a new hostname that includes "alpine34"
# instead of "alpine33".
export new_hostname="foo-alpine34"

sudo \
  --preserve-env \
  --host=localhost \
```



```

-- \
sed --in-place \
    "s/${HOSTNAME}/${new_hostname}/g" \
    /etc/hostname /etc/hosts
sudo \
    --preserve-env \
    --host=localhost \
    -- \
    hostname "$new_hostname"
exec "$SHELL"

# Rerun the manifest using the new node name.
sudo puppet apply \
    --modulepath ~/.puppet/modules \
    /vagrant/roles_and_profiles.pp

# Show the Alpine images installed.
docker images alpine

```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
alpine	3.4	baa5d63471ea	7 weeks ago	4.803MB

As you can see from the output, Puppet has removed the Alpine 3.3 image, and installed Alpine 3.4 instead! How did this happen? Let's break it down into steps:

1. You renamed the host to include the substring "alpine34" in the hostname.
2. Puppet matched the substring using a regular expression in its node definition list.
3. Puppet applied the Alpine 3.4 role (`role::alpine34`) assigned to nodes that matched the "alpine34" substring.
4. The Alpine 3.4 role called its component profiles (which are actually parameterized classes) using "present" and "absent" arguments to

declare the intended state of each image.

5. Puppet applied the image management declarations inside the Alpine 3.3 and Alpine 3.4 profiles (`profile::alpine33` and `profile::alpine34`, respectively) to install or remove each image.

Other Puppet Options for Node Assignment

Puppet can assign roles, profiles and classes to nodes in a number of ways, including the following:

- Classifying nodes with the Puppet Enterprise Console.
- Defining nodes in the main site manifest—for example, `site.pp`.
- Implementing an External Node Classifier (ENC), which is an external tool that replaces or supplements the main site manifest.
- Storing hierarchical data in a Hiera YAML configuration file.
- Using Puppet Lookup, which merges Hiera information with environment and module data.
- Crafting conditional configurations based on facts known to the server or client at runtime.

Each option represents a set of trade-offs in expressive power, hierarchical inheritance and maintainability. A thorough discussion of these trade-offs is outside the scope of this article. Nevertheless, it's important to understand that Puppet gives you a great deal of flexibility in how you classify and manage nodes at scale. This article focuses on the common use case of name-based classification, but there are certainly other valid approaches.

Although hostname-based role assignment is just one of the many ways to manage the configuration of multiple systems, it's a very powerful one, and certainly one of the easiest to demonstrate. Puppet supports a large number of ways to specify what configurations should apply to a given host. The ability to configure systems dynamically based on discoverable criteria makes Puppet a wonderful complement to Docker's versioned images and containerization.

Conclusion

In this article, I took a close look at managing Docker images and containers with `docker::image` and `docker::run`, but the Puppet Docker module supports a lot more features that I didn't have room to cover this time around. Some of those additional features include:

- Building images from a Dockerfile with the `docker::image` class.
- Managing Docker networks with the `docker::networks` class.
- Using Docker Compose with the `docker::compose` class.
- Implementing private image registries using the `docker::registry` class.
- Running arbitrary commands inside containers with the `docker::exec` class.

When taken together, this powerful collection of features allows you to compose extremely powerful roles and profiles for managing Docker instances across infrastructure of almost any scale. In addition, by leveraging Puppet's declarative syntax and its ability to automate role assignment, it's possible to add, remove and modify your Docker instances on multiple hosts without having to manage each instance directly, which is typically a huge win in enterprise automation. And finally, the standardization and repeatability of Puppet-driven container management makes systems more reliable when compared to hand-tuned, hand-crafted nodes that can "drift" from the ideal state over time.

In short, Docker provides a powerful tool for creating lightweight golden images and containerized services, while Puppet provides the

means to orchestrate those images and containers in the cloud or data center. Like strawberries and chocolate, neither is “better” than the other; combine them though, and you get something greater than the sum of its parts. ■

Todd A. Jacobs is a frequent contributor to *Linux Journal*, a Stack Exchange enthusiast, and an industry leader in DevOps transformations that incorporate automated security and IT governance. He currently lives in Baltimore with his beautiful wife, toddler-aged son and two geriatric but lovable dogs.

RESOURCES

Key Files from This Article, Available on GitHub: <https://github.com/CodeGnome/MDIWP-Examples>

Docker: <https://www.docker.com>

Puppet Home Page (Docs and Commercial Versions): <https://puppet.com>

Puppet Ruby Gem (Open-Source Version): <https://rubygems.org/gems/puppet>

Puppet Labs docker_platform Module: https://forge.puppet.com/puppetlabs/docker_platform

The garethr-docker Module Wrapped by docker_platform: <https://github.com/garethr/garethr-docker>

Official Apache HTTP Server Docker Images: https://hub.docker.com/_/httpd

Oracle VirtualBox: <https://www.virtualbox.org>

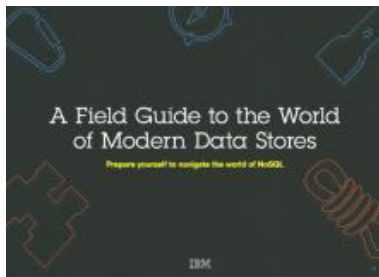
Vagrant by HashiCorp: <https://www.vagrantup.com>

Ubuntu Images on HashiCorp Atlas: <https://atlas.hashicorp.com/ubuntu>

Puppet Documentation on the “Roles and Profiles” Pattern:
https://docs.puppet.com/pe/2016.4/r_n_p_intro.html

Send comments or feedback via
<http://www.linuxjournal.com/contact>
or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)



A Field Guide to the World of Modern Data Stores

There are many types of databases and data analysis tools to choose from when building your application. Should you use a relational database? How about a key-value store? Maybe a document database? Is a graph database the right fit? What about polyglot persistence and the need for advanced analytics?

If you feel a bit overwhelmed, don't worry. This guide lays out the various database options and analytic solutions available to meet your app's unique needs.

You'll see how data can move across databases and development languages, so you can work in your favorite environment without the friction and productivity loss of the past.

Sponsor: IBM

> <https://geekguide.linuxjournal.com/content/field-guide-world-modern-data-stores>



Why NoSQL? Your database options in the new non-relational world

The continual increase in web, mobile and IoT applications, alongside emerging trends shifting online consumer behavior and new classes of data, is causing developers to reevaluate how their data is stored and managed. Today's applications require a database that is capable of providing a scalable, flexible solution to efficiently and safely manage the massive flow of data to and from a global user base.

Developers and IT alike are finding it difficult, and sometimes even impossible, to quickly incorporate all of this data into the relational model while dynamically scaling to maintain the performance levels users demand. This is causing many to look at NoSQL databases for the flexibility they offer, and is a big reason why the global NoSQL market is forecasted to nearly double and reach USD3.4 billion in 2020.

Sponsor: IBM

> <https://geekguide.linuxjournal.com/content/why-nosql-your-database-options-new-non-relational-world>



RunKeeper Case Study

Boston-based fitness start-up RunKeeper was struggling with its database, and could not keep pace with the company's expansion. With new users joining every day, this limitation threatened to halt the company's operations. With a database of 30 million users and growing fast, scaling up also became an issue.

RunKeeper's initial database PostgreSQL failed to provide the required speed and scale. Partnering with IBM, RunKeeper transformed using IBM Cloudant's Dedicated Cluster as its new data layer.

"We were impressed by the wealth of experience that the IBM team was able to draw on to adapt the solution to meet our business needs," says Joe Bondi, CTO and Co-founder of RunKeeper.

Sponsor: IBM

> <https://geekguide.linuxjournal.com/content/run-keeper-case-study>



The 2016 State of DBaaS Report: How managed services are transforming database administration

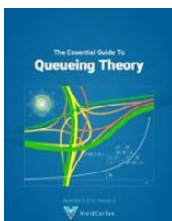
If you didn't have to manage your database, what would you do with your free time? All those hours you previously spent micromanaging your data

layer—ensuring it keeps your application running 24/7 and is able to scale up or down based on demand— would suddenly reappear in your day. You could spend more time building your applications, from adding key features to improving the experience of your users, and you would even get some hours back in your personal life.

The 2016 State of DBaaS Report, commissioned by IBM, assessed the business and technical impact of database-as-a-service (DBaaS), as identified by 680 executive and technical enterprise users, and found that developers are saving a substantial amount of time after adopting DBaaS. All of those surveyed were using a managed, NoSQL database service across a variety of industries, including insurance, healthcare, gaming, retail and finance.

Sponsor: IBM

> <https://geekguide.linuxjournal.com/content/2016-state-dbaas-report-how-managed-services-are-transforming-database-administration>



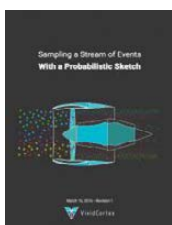
The Essential Guide To Queueing Theory

Whether you're an entrepreneur, engineer, or manager, learning about queueing theory is a great way to be more effective. Queueing theory is fundamental to getting good return on your efforts. That's because the results your systems and teams produce are heavily influenced by how much waiting takes place, and waiting is waste. Minimizing this waste is extremely important. It's one of the biggest levers you will find for improving the cost and performance of your teams and systems.

Author: Baron Schwartz

Sponsor: VividCortex

> <https://geekguide.linuxjournal.com/content/essential-guide-queueing-theory>



Sampling a Stream of Events With a Probabilistic Sketch

Stream processing is a hot topic today. As modern Big Data processing systems have evolved, stream processing has become recognized as a first-class citizen in the toolbox. That's because when you take away the how of Big Data and look at the underlying goals and end results, deriving real-time insights from huge, high-velocity, high-variety streams of data is a fundamental, core use case. This explains the

explosive popularity of systems such as Apache Kafka, Apache Spark, Apache Samza, Apache Storm, and Apache Apex—to name just a few!

Author: Baron Schwartz

Sponsor: VividCortex

> <https://geekguide.linuxjournal.com/content/sampling-stream-events-probabilistic-sketch>

From vs. to + for Microsoft and Linux

Microsoft is now in the foundation for Linux.
What does that mean, if anything?



DOC SEARLS

Doc Searls is Senior Editor of *Linux Journal*. He is also a fellow with the Berkman Center for Internet and Society at Harvard University and the Center for Information Technology and Society at UC Santa Barbara.

PREVIOUS

◀ Feature: Managing Docker
Instances with Puppet

In November 2016, Microsoft became a platinum member of the Linux Foundation, the primary sponsor of top-drawer Linux talent (including Linus), as well as a leading organizer of Linux conferences and source of Linux news (<https://www.linuxfoundation.org/announcements/microsoft-fortifies-commitment-to-open-source-becomes-linux-foundation-platinum>).

Does it matter that Microsoft has a long history of fighting Linux with patent claims? Seems it should. Run a Google search for “microsoft linux patents”, and you’ll get almost a half-million results, most of which raise questions. Is Microsoft now ready to settle

or drop claims? Is this about keeping your friends close and your enemies closer? Is it just a seat at a table it can't hurt Microsoft to sit at?

Maybe it will help to look at patents in general, rather than any of the ones you'll find in contention (or potential contention) at that last link.

The history of patents, at least in the US, is thick with ironies, such as the one we see here, starting with Thomas Jefferson's famous letter to Isaac McPherson in 1813 (http://press-pubs.uchicago.edu/founders/documents/a1_8_8s12.html). Here's the relevant excerpt:

Stable ownership is the gift of social law, and is given late in the progress of society. It would be curious then, if an idea, the fugitive fermentation of an individual brain, could, of natural right, be claimed in exclusive and stable property. If nature has made any one thing less susceptible than all others of exclusive property, it is the action of the thinking power called an idea, which an individual may exclusively possess as long as he keeps it to himself; but the moment it is divulged, it forces itself into the possession of every one, and the receiver cannot dispossess himself of it. Its peculiar character, too, is that no one possesses the less, because every other possesses the whole of it. He who receives an idea from me, receives instruction himself without lessening mine; as he who lights his taper at mine, receives light without darkening me. That ideas should freely spread from one to another over the globe, for the moral and mutual instruction of man, and improvement of his condition, seems to have been peculiarly and benevolently designed by nature, when she made them, like fire, expansible over all space, without lessening their density in any point, and like the air in which we breathe, move, and have our physical being, incapable of confinement or exclusive appropriation. Inventions then cannot, in nature, be a subject of property. Society may give an exclusive right to the profits arising from them, as an encouragement to men to pursue ideas which may produce utility, but this may or may not be done, according to the will and convenience of the society, without claim or complaint from anybody.

Perspective: in 1987, Microsoft had one patent. By 2005, it had 3,955.

That may be the first solid case for both free software and open source. Yet it would be a mistake to hold Jefferson, himself an inventor, to that single statement, since his positions on patents and attitudes toward them changed over the years (<https://www.monticello.org/site/research-and-collections/patents>).

Same with Bill Gates (http://en.swpat.org/wiki/Changes_in_company_policy_over_time). In 1991, Bill wrote (<http://c4sif.org/2011/06/bill-gates-1991-comments-on-patents>):

PATENTS: If people had understood how patents would be granted when most of today's ideas were invented, and had taken out patents, the industry would be at a complete standstill today. I feel certain that some large company will patent some obvious thing related to interface, object orientation, algorithm, application extension or other crucial technique. If we assume this company has no need of any of our patents then [they] have a 17-year right to take as much of our profits as they want. The solution to this is patent exchanges with large companies and patenting as much as we can.

Which they've done. Perspective: in 1987, Microsoft had one patent. By 2005, it had 3,955. In January 2015, the company bragged (<http://blogs.microsoft.com/on-the-issues/2016/01/15/our-growing-patent-portfolio/#sm.00d71dm415slflq11b7168e1z6rrq>):

Microsoft and its employees have worked hard to build and maintain a world-class patent portfolio (<http://spectrum.ieee.org/at-work/innovation/patent-power-2015-social-media-and-smartphones-score-big>), which now includes more than 59,000 US and international patents, and over 36,000 pending patent applications. Drafting and prosecuting high-quality patent applications is key to that success.

Our patenting strategy, in terms of how, when and where we choose to protect those innovations, is closely aligned to Microsoft's overall business strategy (<http://www.geekwire.com/2015/exclusive-satya-nadella-reveals-microsofts-new-mission-statement-sees-more-tough-choices-ahead>). Microsoft is committed to transparency of patent ownership and all patents owned directly by either Microsoft or MTL or through subsidiaries are publicly available via the Open Register of Patent Ownership (<http://oropo.net>).

Microsoft will continue to be one of the top patent filers in the world, which reflects our commitment to the tremendous amount of R&D and innovation that goes into creating products for our partners and our customers.

A friend at Microsoft many years ago explained to me that the main reason for a large company to hold a patent portfolio was not to license or cross-license, but instead to participate in what he called "nuclear arms dealing", most of which consists of "trades": private agreements that open business opportunities unimpeded by patent-based threats. Kind of like, "I'll let you work on my land if you let me work on yours."

There are also moves that don't involve any second parties. The first party simply opens market opportunities by telling the world that a patented invention is free for the taking. Perhaps the best example of that is Ethernet (<https://en.wikipedia.org/wiki/Ethernet>), which won in the market over IBM's Token Ring (https://en.wikipedia.org/wiki/Token_ring) and General Motor's Token Bus (https://en.wikipedia.org/wiki/Token_bus_network) because Ethernet's patent holders—Xerox, Digital Equipment Corp. and Intel (nicknamed "DIX")—declared Ethernet an open standard (<http://standards.ieee.org/events/ethernet/history.html>). This was at the urging of Bob Metcalfe, Ethernet's primary inventor, an inveterate critic (<https://web.archive.org/web/20070316025237/http://www.infoworld.com/articles/op/xml/99/06/21/990621opmetcalfe.html>) of open source whose own mind changed over the years (<https://www.linux.com/news/bob-metcalfe-re-evaluates-open-source>).

To put patents in perspective, Figure 1 shows the Burton Matrix, which

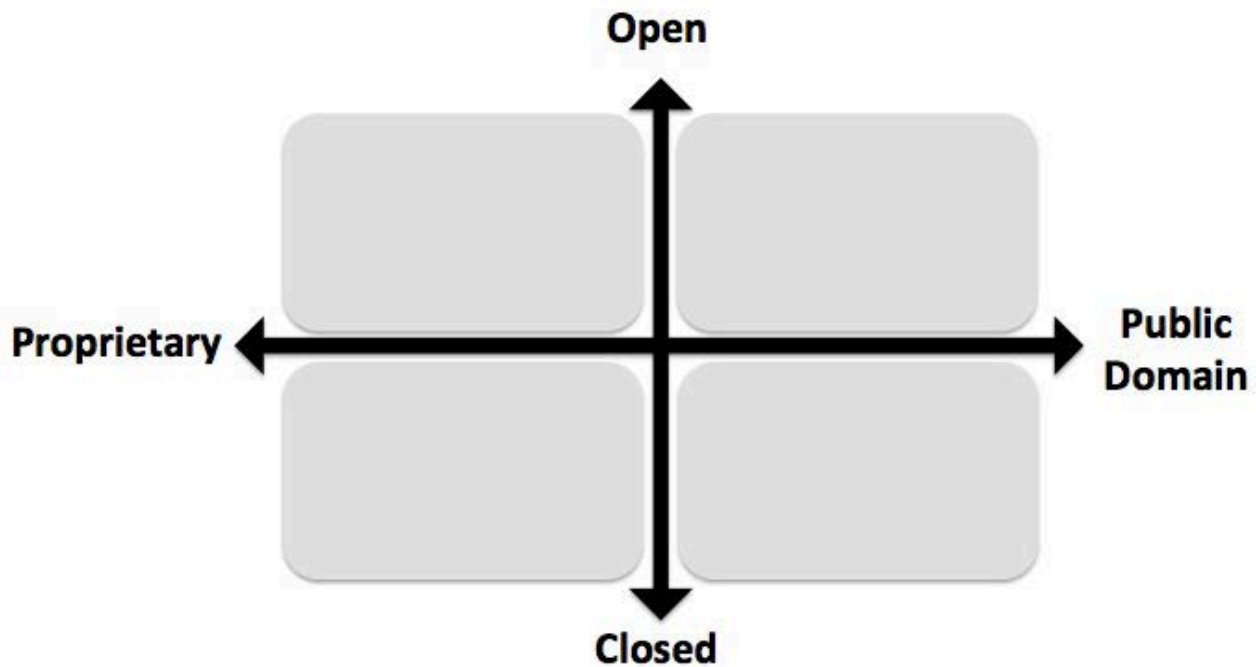


Figure 1. The Burton Matrix

Craig Burton created originally to make clear that the opposite of open was not proprietary, but closed. And that the opposite of proprietary was public domain.

Patents that aren't encumbered by property rights threats would go in the lower-right quadrant, while Linux and other forms of open-source code would go in the upper-right. Could it be that Microsoft would like to follow DIX's Ethernet move by pushing some of its patents in a rightward direction here? I guess we'll see.

Meanwhile, it should help to be mindful of where patents, and their corporate parents, dwell in civilization. To help with that, let's borrow The Long Now's "Layers of Time" graphic (Figure 2).

Here's how I explained the graphic in O'Reilly's *Open Sources 2.0*, published in 2005 (http://programmer.97things.oreilly.com/wiki/index.php/Open_Sources_2.0/Beyond_Open_Source:_Collaboration_and_Community/Making_a_New_World):

At the bottom we find the end-to-end nature of the Net. It's also where we find Richard M. Stallman, the GNU project, the Free Software Foundation (FSF) and hackers whose interests are anchored in the nature

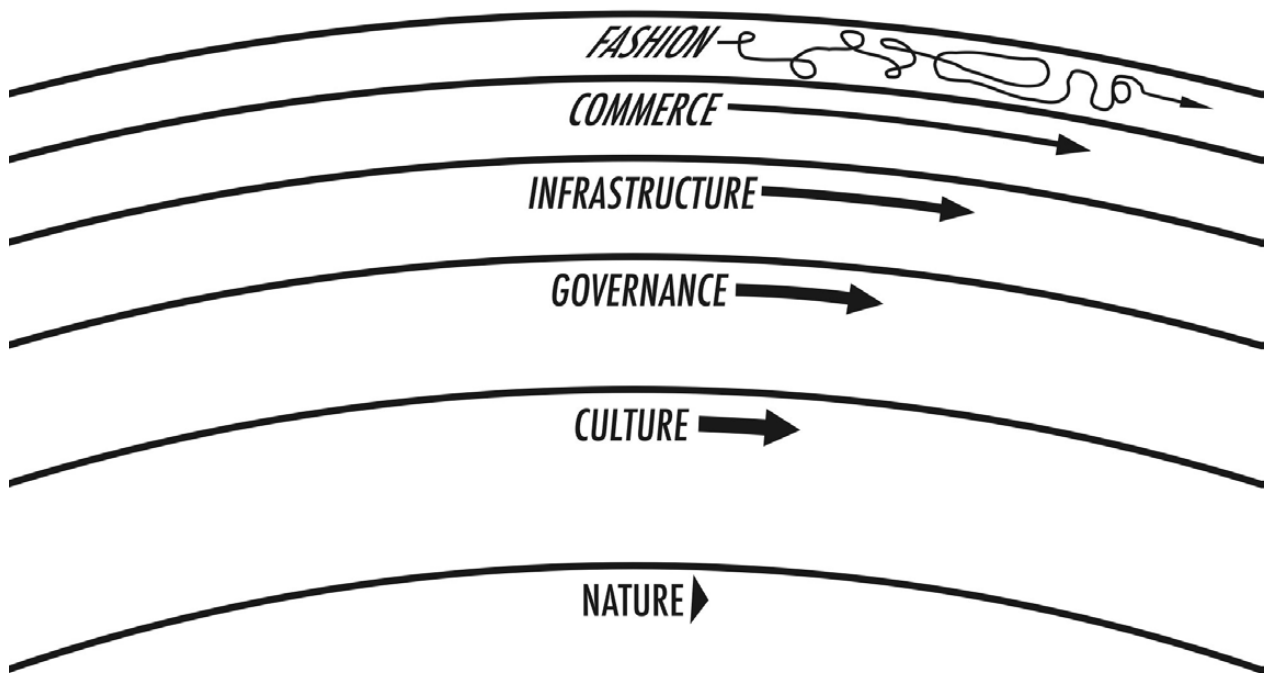


Figure 2. The Long Now's "Layers of Time"

of software, which they understand fundamentally to be free.

When Richard M. Stallman writes "everyone will be able to obtain good system software free, just like air", he's operating at the Nature level. He doesn't just believe software *ought* to be free; he believes its nature is to be free. The unbending constancy of his beliefs has anchored free software, and then open source development, since the 1980s. That's when the GNU tools and components, along with the Internet, began to grow and flourish.

The open source movement, which grew on top of the free software movement, is most at home one layer up, in Culture. Since Culture supports the Governance, the open source community devotes a lot of energy and thought to the subject of licensing. In fact, the Open Source Initiative (OSI) serves a kind of governance function, carefully approving open source licenses that fit its definition of open source. While Richard and the FSF, sitting down there at the Nature level, strongly advocate one license (the GPL or General Public License), the OSI has approved around 50 of them. Many of those licenses are authored by commercial entities with an interest in the governance that supports the infrastructure they put to use.

In fact, it was an interest in supporting business that caused the open source movement to break off of the free software movement. That break took place on February 8, 1998, when Eric S. Raymond wrote “Goodbye, ‘free software’; hello, ‘open source’” (<http://www.catb.org/%7eesr/open-source.html>). Here is where the Culture layer can clearly be seen moving faster, and breaking from, the Nature layer...

Not coincidentally, the Culture on which this new world depends is hacker culture, about which Eric, a founder of the OSI, has written extensively (he edited both editions of *The Hacker’s Dictionary*). Both he and Bruce Perens, another leading open source figure, have purposefully advocated open source to business for many years.

And although open source hackers tend to be more interested in business than free software hackers, both want Governance and Infrastructure that *support* business but are not *determined* by business—except when business works with the hacker community. Hence OSI’s license-approval process. While the number of open source licenses has been a source of some debate (almost everybody would rather see fewer licenses), it is important to note that the relationship between these layers is not the issue. The last thing anybody in the free software or open source movements wants is for anybody at the Commerce level to reach down into Governance to control or restrict Infrastructure that everybody relies upon. Even though that’s exactly why large companies, and whole industries, hire lobbyists. More about that issue shortly.

Changing corporate culture to adapt to open source development methods is not easy. Dan Frye, who runs IBM’s Linux development program, recently told me that IBM has worked hard to make its internal development efforts coordinate smoothly with Linux’s. That way, when IBM “scratches its itches”, the kernel patches that result have a high likelihood of acceptance. IBM has faith that its accepted patches are ones that are most likely to work for everybody and not just for IBM. This is a natural and positive way for infrastructure to grow.

And grow it has. The selection of commodity open source building

materials is now so complete that most businesses have no choice but to use those components—or, in many cases, to recognize that IT personnel in their enterprises have been building their own open source “solutions” for some time.

That realization can come as a shock. Open source infrastructure inside companies often (perhaps usually—it’s hard to tell) gets built without IT brass knowing about it. In many cases, internal open source development and use has had conditional approval by CIOs and CTOs. Whatever the course of open source growth, at a certain point a threshold is crossed, and companies suddenly know that open source is no longer the exception, but the rule.

Now we find ourselves living in a time of extreme dependence on the commerce layer, living like serfs in the feudal castles of Google, Apple, Facebook and Amazon: a combined entity commonly called GAFAM in Europe. Microsoft wouldn’t mind having an M in that acronym, I suppose. Is that what joining the Linux Foundation is about?

Or is Microsoft finally taking the advice I gave here two years ago, in a column titled “A Cool Project for Microsoft: Adopt Linux” (<http://www.linuxjournal.com/content/cool-project-microsoft-adopt-linux>)? There I wrote this:

Today the reality of Linux is of a piece with the reality of the Internet. Neither is going away. Both are co-evolving in the minds of every geek adding value to them. Both transcend the interests of every company contributing to them, including Google. If Satya Nadella (https://en.wikipedia.org/wiki/Satya_Nadella) looks at reality with the same clear eyes Bill Gates cast on the Internet in 1995, he might see the wisdom of embracing Linux with the same enthusiasm and commitment.

How would Microsoft do that, exactly? It could start by joining the Linux Foundation (<https://www.linuxfoundation.org>)....

Mother Jones’ original tagline was “You trust your mother. But you cut the cards.” That may be the best attitude for the Linux Foundation

to have toward Microsoft. In “Microsoft and Linux Patents and Tweets” (<https://meshedinsights.com/2016/11/22/microsoft-linux-patents-tweets>), Simon Phipps has two excellent recommendations I’ll leave us with here: What could Microsoft and the Linux Foundation do?

- The Linux Foundation should include in its membership agreement a good-faith commitment not to initiate any patent litigation relating to the Linux platform against anyone and exclude those who break it. A trade association should not permit its members to fight among themselves.
- Microsoft should declare that no part of the company will in future initiate software patent claims against the Linux platform and as a sign of its good faith join the Open Invention Network. That’s not of itself magical—Oracle and Google are both OIN members and still litigating Android patents—but the combination of gestures could make a tremendous difference to community trust. ■

Send comments or feedback via <http://www.linuxjournal.com/contact> or to ljeditor@linuxjournal.com.

[RETURN TO CONTENTS](#)

ADVERTISER INDEX

Thank you as always for supporting our advertisers by buying their products!

ADVERTISER	URL	PAGE #
Drupalize.me	http://drupalize.me	57
Libre Planet 2017	http://libreplanet.org/conference	47
Peer 1 Hosting	http://go.peer1.com/linux	124
SCALE 15x	http://www.socallinuxexpo.org	29
Silicon Mechanics	http://www.siliconmechanics.com	15
SPTechCon	http://www.sptechcon.com	65
SUSE	http://suse.com/storage	7

ATTENTION ADVERTISERS

The *Linux Journal* brand’s following has grown to a monthly readership nearly one million strong. Encompassing the magazine, Web site, newsletters and much more, *Linux Journal* offers the ideal content environment to help you reach your marketing objectives. For more information, please visit <http://www.linuxjournal.com/advertising>



GEEK GUIDE

Tame the Docker Life Cycle with SUSE

By John S. Tonello

It's no accident or mere passing fad that containers are revolutionizing how IT shops of all sizes do their work. Whether you're looking to make better use of existing data-center resources or improve portability to the cloud, Docker and the new-found freedom it offers to use virtual environments for everything from development to enterprise applications holds a lot of promise.

The challenge is figuring out how best to move beyond a standard Docker install to an enterprise-worthy solution that's secure, easy to manage and scalable. It's also important to find ways to manage all your containers easily as well as the images you modify and plan to reuse. After all, containers are only part of any enterprise, which is now a healthy mix of bare-metal boxes, virtual machines, containers and on- and off-premises clouds. Tools that can help provide a common framework—and familiar interfaces—are critical.

With SUSE Enterprise Linux Server 12 and the tools it offers, you and your team can begin to solve real-world problems, tame the Docker life cycle, and create, run and maintain containers at nearly any scale.

The Container Revolution

Anyone managing hardware—from a

few blades to full data centers—knows that bare-metal server deployments are costly, time-consuming and not very efficient. Even if you could still afford it, the idea of running one or two services on a single physical server—maybe a database here, a website there—is just not practical. Even if you're the best system administrator out there, you can really make only educated guesses about the maximum amount of CPU, memory and storage a particular service will need over time. Once you do the math and purchase the hardware, you know there surely will be hours, days and weeks when your physical server's capacity is idle and of no use to you.

Virtual machines changed all that by enabling more efficient use of that same physical server's resources by sharing them across separate instances of Linux and Windows servers. With the advent of VMware and Hyper-V and open-source KVM and Xen, suddenly you could place multiple servers on a single physical box, quickly move them between clusters, more easily run backups and restores, clone them and manage them all from a single interface.

To continue reading, download the complete eBook for FREE at <http://geekguide.linuxjournal.com>.



Where every interaction matters.

break down your innovation barriers

power your business to its full potential

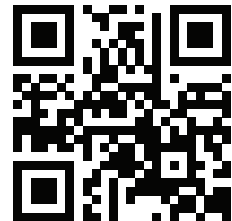
When you're presented with new opportunities, you want to focus on turning them into successes, not whether your IT solution can support them.

Peer 1 Hosting powers your business with our wholly owned FastFiber Network™, global footprint, and offers professionally managed public and private cloud solutions that are secure, scalable, and customized for your business.

Unsurpassed performance and reliability help build your business foundation to be rock-solid, ready for high growth, and deliver the fast user experience your customers expect.

Want more on cloud?

Call: 844.855.6655 | go.peer1.com/linux | [View Cloud Webinar:](#)



Public and Private Cloud | Managed Hosting | Dedicated Hosting | Colocation