

LINUXTM JOURNAL

Since 1994: The Original Magazine of the Linux Community

LUCI4HPC:
the Lightweight,
User-Friendly
Cluster
Management Tool

APRIL 2015 | ISSUE 252 | www.linuxjournal.com

HIGH-PERFORMANCE COMPUTING

JAILHOUSE

Real-Time
Security-Wise
Virtualization

PLUS

Explore the
Concepts of
STDIN, STDOUT
and STDERR

Write a Script
to Generate
Word Searches

HOW-TO:

Installing Libreboot
on an x60

CREATE

Dynamic Content
with Django Templates



WATCH:
ISSUE
OVERVIEW



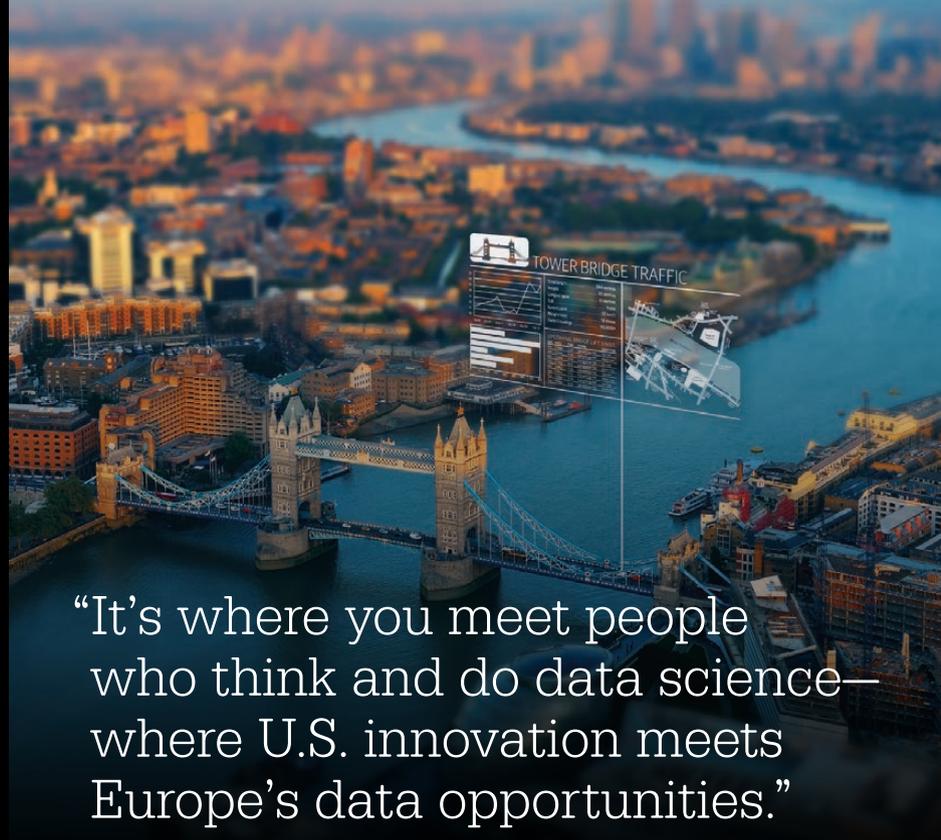
Strata+ Hadoop WORLD

PRESENTED BY

O'REILLY®

cloudera®

London, UK
5-7 May, 2015



“It’s where you meet people who think and do data science—where U.S. innovation meets Europe’s data opportunities.”

The Future Belongs to Those Who Use Data

Hear how companies like Google, Goldman Sachs, Ebay, CERN, BBC, and Siemens AG are using data—and how you can benefit from their experiences.

Conference Tracks include: Business & Industry, Privacy and Law, Data Science, Tools & Technology, and Internet of Things.



Don't miss what Forbes calls *"A mind-blowing big data experience."*
Register with code **BD20** to save 20% on passes.

strataconf.com/uk

The Advanced Foundation for Your Success

12

SUSE Linux Enterprise 12

- + Increase Uptime
- + Improve Operational Efficiency
- + Accelerate Innovation



www.suse.com/sle12



CONTENTS

APRIL 2015

ISSUE 252

HIGH-PERFORMANCE COMPUTING

FEATURES

68 LUCI4HPC

LUCI4HPC aims at providing an easy and convenient tool to set up and maintain an in-house high-performance computer cluster.

**Melanie Grandits,
Axel Sündermann
and Chris Oostenbrink**

78 Jailhouse

Explore a simple, compact, yet real-world Linux hypervisor striving to cover use cases that others don't quite fit.

Valentine Sinitsyn

ON THE COVER

- LUCI4HPC: the Lightweight, User-Friendly Cluster Management Tool, p. 68
- Jailhouse: Real-Time Security-Wise Virtualization, p. 78
- Explore the Concepts of STDIN, STDOUT and STDERR, p. 56
- Write a Script to Generate Word Searches, p. 42
- How-To: Installing Libreboot on an x60, p. 48
- Create Dynamic Content with Django Templates, p. 34

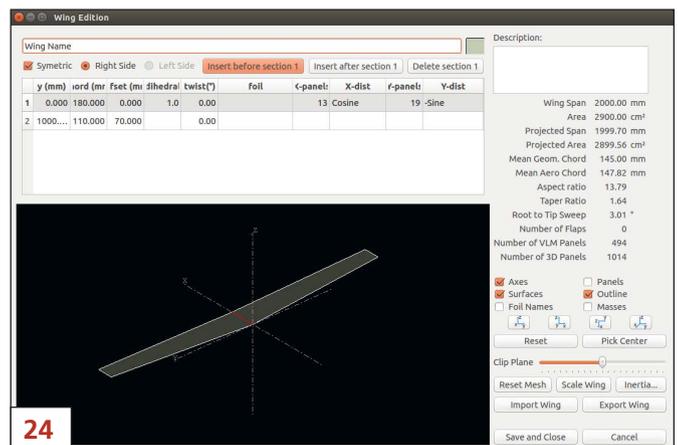
Cover Image: © Can Stock Photo Inc. / alexaldo

COLUMNS

- 34** **Reuven M. Lerner's
At the Forge**
Django Templates
- 42** **Dave Taylor's
Work the Shell**
Where's That Pesky Hidden Word?
- 48** **Kyle Rankin's
Hack and /**
Libreboot on an x60, Part II:
the Installation
- 56** **Shawn Powers'
The Open-Source Classroom**
Pipes and STDs
- 94** **Doc Searls' EOF**
Consent That Goes Both Ways

IN EVERY ISSUE

- 8** **Current_Issue.tar.gz**
- 10** **Letters**
- 16** **UPFRONT**
- 32** **Editors' Choice**
- 64** **New Products**
- 103** **Advertisers Index**



LUCI4HPC **CONTROL PANEL**

mmscluster

- ..:Overview:..
- ..:Manage:..
- ..:Scheduler:..
- ..:Options:..
- ..:License:..
- ..:Logout:..

Status

Nodes: 38 Logins: 2 Other: 7 Candidates: 0
Up: 40/40
Installed: 40/40
CPUs: 342/728 GPU: 6/22 Memory: 10/681 GB

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.

Bases loaded? Who's in your Lineup?



MINNOWBOARD MAX



Net:gate **Rookie**

Ht: 1 in. L: 4 in. W: 3 in. Wt. 5oz.

LEAGUE STATISTICS

CPU: Intel® Dual Core Atom™ E3825 1.33GHz
RAM: 2GB
Storage: MicroSD Slot
I/O: (1)Gb Ethernet, SATA2, (1)USB 3.0, (1)USB 2.0, micro HDMI, expansion headers
 Integrated Intel HD Graphics, 1920x1080 max
 Open Source hardware. Pick your OS: Windows®8.1, Android 4.4, FreeBSD 10, Linux, Yocto™Compatible



RCC-VE 4860



Net:gate **MVP**

Ht: 1.5 in. L: 6.8 in. W: 7 in. Wt. 17 oz.

LEAGUE STATISTICS

CPU: Intel Quad Core Atom C2558 "Rangeley" 2.4GHz
RAM: 8GB
Storage: 4GB eMMC onboard
I/O: (6)Gb Intel Ethernet, SATA2, mSATA, (2) mPCI sockets (one with microSIM), USB
 Fanless, low-power, NFV edge device
 Ships with CentOS 7



APU1D4



Net:gate **Southpaw**

Ht: 1 in. L: 6.3 in. W: 6.5 in. Wt. 16 oz.

LEAGUE STATISTICS

CPU: AMD® G Series Dual Core "Bobcat" T40E APU 1Ghz
RAM: 4 GB
Storage: mSATA slot, SD card slot
I/O: (3)Gb Ethernet, USB, (1)SATA, DB9 serial, (2) miniPCIe sockets (one with micro sim)
 Runs many flavors of BSD and Linux



RCC-VE 2440



Net:gate **Slugger**

Ht: 1.5 in. L: 6.8 in. W: 7 in. Wt. 17 oz.

LEAGUE STATISTICS

CPU: Intel Dual Core Atom C2358 "Rangeley" 1.7GHz
RAM: 4GB
Storage: 4GB eMMC onboard
I/O: (4)Gb Intel Ethernet, SATA2, mSATA, (2) mPCI sockets (one with microSIM), USB
 Fanless, low-power internet gateway system
 Ships with CentOS 7



C2758



Draft Pick
Photo Not Available

Net:gate **Heavy Hitter**

Size: 1U Rackmount Wt. TBA

LEAGUE STATISTICS

CPU: Intel Eight Core Atom C2758 "Rangeley" 2.4GHz
RAM: 8GB RAM
Storage: 64GB eMMC onboard. Yes, 64 GB!
I/O: (6)Gb Intel Ethernet, SATA2, mSATA, (2) mPCI sockets (one with microSIM), USB
 Flexible low-power networking platform
 Arrives with CentOS 7 and a low noise fan!



BEAGLEBONE BLACK



Net:gate **The Mascot**

Ht: <1in. L: 3.5in. W: 2.5in. Wt. 2oz.

LEAGUE STATISTICS

CPU: AM3358 1Ghz ARM®Cortex®-A8
RAM: 512MB
Storage: 4GB eMMC, microSD slot
I/O: (1)10/100 Ethernet, USB, HDMI, (2)46-pin headers, 3D graphics accel.
 100% Open Source. Runs FreeBSD, Android, Angstrom, Debian, Nintendo, Beaglenmt, etc!



7212 McNeil Drive Suite 204 Austin, TX 78729 +1.512.646.4100

Netgate is a registered trademark of Rubicon Communications, LLC.

Intel is a trademark of Intel Corporation in the U.S. and other countries. Microsoft Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

CentOS is a trademark of the CentOS project. AMD is a registered trademark of AMD. Atom is a trademark of Intel Corporation in the U.S. and other countries.

ARM and Cortex are registered trademarks of ARM Limited in the EU and elsewhere. Yocto Project™ is a trademark of the Linux Foundation.



SHAWN POWERS

High Performance: a Relative Term

My Pebble watch has several orders of magnitude more power than the mainframe computers used by NASA to land astronauts on the moon and then get them back safely. In fact, at the time, the six-megabyte program IBM developed to monitor the astronaut's biometric and environmental data was the most complex software ever written! Times certainly have changed, but our desire to push computing to its very limit surely hasn't. This month is the High-Performance Computing issue of *Linux Journal*, and whether you plan to land humans on the moon or analyze weather data over the Atlantic,

this issue should tickle your fancy.

We start out with Django templates, as Reuven M. Lerner builds on last month's column about the "atfproject" he started. Whether you want to develop with Django or not, Reuven's lessons always are beneficial to the new programmer and the seasoned developer alike. Dave Taylor follows with a topic that is half scripting and half brain teaser. How can you make a script to create the words in a word search? Is a brute-force, every possible iteration solution the best way? Dave starts the script this month and urges us to send him ideas.

Kyle Rankin continues last issue's article on Libreboot. We deal with the operating systems (specifically Linux!) all the time, but Kyle dives deeper and explains how to replace the system BIOS with an open alternative. My column isn't nearly



VIDEO:
Shawn Powers runs
through the latest issue.

With a friendly Web interface and high customization ability, LUCI4HPC might be perfect for your needs.

as low-level, but it addresses an often confusing concept for Linux users—namely, system I/O. If STDIN, STDOUT and STDERR sound scary, fear not. I explore I/O in all its glory and round things out with a clearer understanding of pipes and redirects too.

You will get a really clear look at LUCI4HPC this month, as Melanie Grandits, Axel Sündermann and Christ Oostenbrink explain the lightweight clustering system. Although clustering often is needed only in specialized circumstances, it doesn't mean the process needs to be difficult. With a friendly Web interface and high customization ability, LUCI4HPC might be perfect for your needs.

Valentine Sinitsyn describes how to use Jailhouse this month. Although it might seem like “just another virtualization platform”, Jailhouse is designed with real-time virtualization in mind. For folks in system automation, medical and telecommunications, real-time Linux is crucial. Sadly, most virtualization systems don't handle real-time

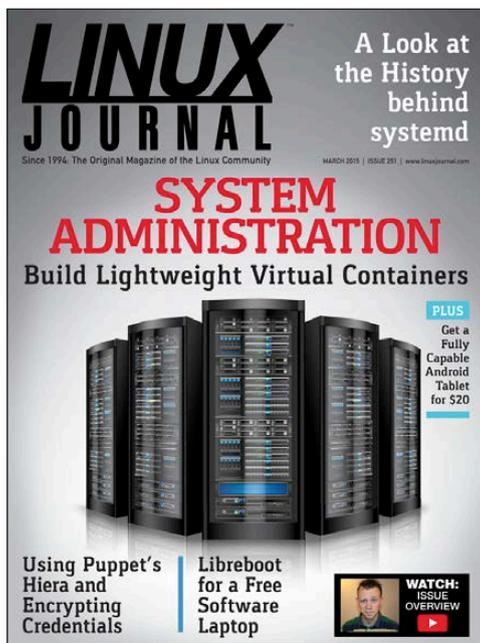
solutions very well. If you need real-time solutions, but want the cost benefits of virtualization, Jailhouse might be perfect.

Finally, Doc Searls tears open the “Do Not Track” concept and explores its meaning, intent, current state and future. If you, or your users, are concerned about who gets access to what data, Doc's column will be of particular interest this month.

In fact, as a Linux user, this entire issue is full of articles that most likely will pique your interest. Whether you want to rewrite the BIOS on your laptop or just want some free cash (be sure to read my UpFront article on ChangeTip), this issue aims to please. Add the product news, tech tips, kernel updates and so on, and you have on your screen a great way to spend April 1st—no fooling! ■

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 e-mail at shawn@linuxjournal.com. Or, swing by the #linuxjournal IRC channel on Freenode.net.

letters



Shuffling Cards—Dave Taylor

I love your column, although I think you concentrate a bit too much on the application and too little on interesting bash tricks. Now I know these are hard to find, or perhaps what is trivial for one is new to the other. Perhaps the same also applies with the shuffling cards algorithm from your February 2015 column. There is this “well” known algorithm from Fisher-Yates. See Wikipedia for the details. It is embarrassingly straightforward once you understand it and, therefore, so clever. And, it’s very easy to implement.

I’m looking forward to seeing

an implementation in your next great column.

—Hans

***Dave Taylor replies:** Thanks for your kind note, Hans! As you have no doubt figured out by now, I’m more intrigued by algorithms and implementations than I am by the weird corner-case tricks and shortcuts in the Bash shell. My logic is that obfuscated code might be neat, but it’s not elegant and therefore also isn’t maintainable. I hate reading someone else’s undocumented shell script that requires hours to figure out. That’s just not good coding. We’re not in the 1960s where every kilobyte counts, after all!*

I’ll check out the algorithm, but truth be told, I’m heading down a different path starting with the column I owe LJ today—one that’s tied to a project I promised my 11-year-old a while back, a program that can create word searches.

Shake Up the Content

I used to be an *LJ* subscriber for years and originally started reading your magazines somewhere in the mid-1990s. I used to enjoy reading and

learning from your magazine. Over the years, however, I started noticing a repeated pattern in the articles and writers for the mag. In my opinion, it started to lack originality, and my interest faded until I didn't re-subscribe. I received a "get a free copy of the February issue" e-mail this morning and decided to check it out, but was disappointed that the same pattern existed since roughly the year had passed since my subscription dropped.

I want you guys/gals to succeed, and that's why I'm writing this feedback. I'd also note that I have no experience running a magazine, so feel free to take my feedback with a grain of salt.

Shake up the usual authors:

- I remember when Dave Taylor's shell series began. I thought it was a good idea, but it has ran its course. While you may continue to learn bits from the article, the kind of problems being solved in the shell should be done in higher-level languages like Python/Ruby/etc. When I see the lack of appropriateness, I tend to just skip the entire article.
 - Try re-assigning the staff writers to new topics for a month or two. For example, kill Dave's shell series and ask him to start a new one with a new language with the same original premise that motivated the shell series: learning the basics of popular language X in this short 6-month series. (Don't drag it on for years.)
 - With a notable exception, I do think Kyle Rankin has a good variety of topics he covers.
- Look for new ways to engage your audience:*
- What if you had an *LJ* Docker account, and every article had an associated Docker image that you could immediately pull down and play with and/or follow along in the article?
 - Looking for new/repeated topics? What about the most votes on Linux topics from SO (<http://stackoverflow.com/questions/tagged/linux?sort=votes>)?
 - Some Linux themes, however, do need to be repeated every 2–3

years or so. I'm thinking of that perfect radio station that strikes the balance between introducing new songs (topics) and repeating old favorites but not every hour. The kind of stuff I'd expect here would be compiling your kernel; shell scripting basics (I was too hard on Dave, wasn't I?); and Emacs and Vim improvements (the topic never dies, does it?).

- Provide a "ransom for topic" feature where readers get to vote or even contribute \$\$ to see a particular topic thoroughly covered. I'm inspired by that SO link above in this case.
- Become "the" women's Linux magazine. I see a lot of inequality mentions in my Twitter feed about the disproportionate lack of women's involvement in tech. Imagine the kind of new readership you'd gain by actively seeking out new female authors? Find one to join the repeated, monthly contributors as well as have a variety of guest women authors. Heck, kick it off by having a special all-woman-authored issue?

Get crazy, generate some new, fresh ideas. I'll keep checking in periodically

to see how you all are doing. I hope to be back.

—Jon Miller

Jon, thanks for the great feedback. Trying to come up with relevant, interesting and entertaining content is the ultimate goal for us here. Without feedback, we're just making educated guesses with our own nerdy intuition. Thanks again, and hopefully we'll see you back in the future!—Shawn Powers

RE: the Awesome Program You Never Should Use

Regarding Shawn Powers' UpFront piece in the November 2014 issue: if one's shell is Bash and has the HISTCONTROL variable set to ignorespace, then one can "mitigate potential damage" by prefacing the command with a space so your user name/password doesn't appear in your .bash_history file.

Having said that, sshpass is a horrible tool for all the reasons outlined in the review. I'm not endorsing the method or the tool, I'd rather err on the side of caution and just not use it and find a better way to accomplish my goal.

But, if one doesn't mind feeding credentials on the command line

(ideally in a closed environment), then `sshpass` could be useful. And if people don't wish to have their user name/password appear in their history, at least there's a method for that.

—Eric Frost

Eric, great tip, and thank you! I pondered a long time about including `sshpass` in the magazine. Like you, I see just how horrible the idea of putting user/pass on the command line can be. Still, there are occasions when I find myself using it, so I decided I'd rather talk about it than try to "secure" it with obfuscation or by ignoring it. Still, it does creep me out every time I use it!—Shawn Powers

A Very Thorough Article on SQL Injection

Shea Nangle's article on `Drupageddon` in the February 2015 issue was well researched and thorough. It was very useful to see the comparison of the legitimate SQL queries vs. the malicious SQL queries, and the output of each. One thing missing was a suggestion of the "best" place to research security vulnerabilities in Drupal. The official Drupal Security Team site (<https://www.drupal.org/security>) is pretty limited if I want

to search for specifics. I don't see a way to "list only vulnerabilities for Drupal 7" or "show me only Highly Critical vulnerabilities". There must be something better than paging through ten entries at a time.

—Dan Stoner

Shea Nangle replies: *Thank you for your kind words regarding the article! In terms of the Drupal Security Team Web site, I am, unfortunately, not aware of any way to query the site in the fashion that you mention. That said, the Advanced Search functionality of the National Vulnerability Database (<https://web.nvd.nist.gov/view/vuln/search-advanced>) allows you to do the sort of querying that you're referring to, at least for any vulnerabilities for which CVEs have been assigned.*

Kernel Column

What happened to Zack Brown's `diff -u` kernel column in the February 2015 issue? This is first thing I read every issue! Do not ditch this!

—Stephen

Don't worry—Zack's kernel column is back this issue. He was just too busy during the holidays to write that month. Glad you like the column!

Readers' Choice Awards and Raspberry Pi

How about having a category called "Best application for on-line reading of *Linux Journal*"?

Have you bought your Raspberry Pi 2 yet? I looked on the Radio Spares (I am in UK) site today and already they are on back order.

—Roy Read

Thanks for the category suggestion for the next Readers' Choice Awards. We hope other readers will send ideas as well! Regarding your Raspberry Pi question: I have a few B+ models of the original Raspberry Pi, but I just don't have a need for the RPi 2 yet. I'm building an emulation machine for old console games, and I'll probably wish I had the faster model 2, but the B+ should be powerful enough. I love that the price is still \$35!!!—Shawn Powers



WRITE LJ A LETTER

We love hearing from our readers. Please send us your comments and feedback via <http://www.linuxjournal.com/contact>.

PHOTO OF THE MONTH

Remember, send your Linux-related photos to ljeditor@linuxjournal.com!

LINUX JOURNAL

At Your Service

SUBSCRIPTIONS: *Linux Journal* is available in a variety of digital formats, including PDF, .epub, .mobi and an on-line digital edition, as well as apps for iOS and Android devices. Renewing your subscription, changing your e-mail address for issue delivery, paying your invoice, viewing your account details or other subscription inquiries can be done instantly on-line: <http://www.linuxjournal.com/subs>. E-mail 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 on-line: <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.

Downtime isn't an option.

Towards Zero Downtime with SUSE.

You're under pressure to keep your systems available 24x7x365. Whether planned or unplanned—any downtime impacts your bottom line. So trust the 20 years of experience SUSE has been providing its customers reliable, cost-effective data center solutions that minimize server downtime.



www.suse.com/zerodowntime



diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

Recently there was some discussion about ways to ease the tired backs of kernel maintainers. Apparently the **merge windows** are times of great labor, and some folks wanted to alert contributors to some preferable code submission habits.

There were a variety of ideas, and **Kevin Cernekee** summarized them in patch form, but one key idea was that none of this advice really could be treated as etched into stone.

Linus Torvalds and **Theodore Ts'o**, in particular, pointed out that maintainers all have their own ways of doing things, and that no general rules could be relied on universally to produce repeatable results.

In general though, as Kevin posted, the merge window is not a great time to submit new patches. The merge window is the time after a new kernel version comes out and before the first -rc release. Developers either should avoid submitting patches at that time, or as was also discussed, they at least should not expect a reply to their patches, and they should avoid submitting any patch a second time

during that period, if the first one seems to go unaccepted.

Kevin also posted a very rough calculation of when developers might expect to see their code in an official kernel. If they submit code within the first four -rc releases, they could expect to see their code in the next official kernel release. If they submit code within the remaining four -rc releases, they could expect to see it in the second following official release. **Alan Cox** thought this calculation very valuable, though Linus cautioned that it was really quite a rough estimate and highly dependent on any given maintainer's particular patch acceptance habits.

Richard Weinberger has suggested a security improvement aimed at attackers who target forking servers, such as **httpd** and **sshd**. Apparently by creating lots of forks, the attacker could make guesses about code locations in each forked memory allocation. After enough tries, it potentially could find the location of key code and launch a root shell in the parent process. That would be bad.

Richard's idea was to identify if a child process dies due to a fatal error and cause future forks to wait 30 seconds before executing. This would cause the attack to take much more time, but would tend not to inconvenience regular users.

There was some support for this idea and some opposition. **Pavel Machek** came to believe that Richard's patch was only trying to slow the kernel down in random ways, in the hope that it might help. But

Kees Cook and **Andy Lutomirski** both felt that Richard's patch was highly targeted and would not unduly delay user code.

Richard had gotten his original idea while exploring the intricacies of the **offset2lib** weakness, which detailed a way for attacking code to identify the location of user libraries in memory. Once this location is known, there are relatively trivial ways to launch a root shell. Any technique by which an attacker could gain knowledge of the

Powerful: Rhino



- Rhino M4800/M6800**
- Dell Precision M6800 w/ Core i7 Quad (8 core)
 - 15.6"-17.3" QHD+ LED w/ X@3200x1800
 - NVidia Quadro K5100M
 - 750 GB - 1 TB hard drive
 - Up to 32 GB RAM (1866 MHz)
 - DVD±RW or Blu-ray
 - 802.11a/b/g/n
 - Starts at \$1375
 - E6230, E6330, E6440, E6540 also available

- High performance NVidia 3-D on an QHD+ RGB/LED
- High performance Core i7 Quad CPUs, 32 GB RAM
- Ultimate configurability — choose your laptop's features
- One year Linux tech support — phone and email
- Three year manufacturer's on-site warranty
- Choice of pre-installed Linux distribution:



Tablet: Raven



- Raven X240**
- ThinkPad X240 by Lenovo
 - 12.5" FHD LED w/ X@1920x1080
 - 2.6-2.9 GHz Core i7
 - Up to 16 GB RAM
 - 180-256 GB SSD
 - Starts at \$1910
 - W540, T440, T540 also available

Rugged: Tarantula



- Tarantula CF-31**
- Panasonic Toughbook CF-31
 - Fully rugged MIL-SPEC-810G tested: drops, dust, moisture & more
 - 13.1" XGA TouchScreen
 - 2.4-2.8 GHz Core i5
 - Up to 16 GB RAM
 - 320-750 GB hard drive / 512 GB SSD
 - CF-19, CF-52, CF-H2, FZ-G1 available



location of code in memory, therefore, must be considered a security hole and be fixed immediately. But, it's not always clear exactly how best to prevent that information from being seen.

The **Arm** and **Arm64** projects are experiencing a kind of growing pain—some incompatibilities between the `/proc/cpuinfo` files on both architectures that are causing some user programs to lose portability.

Part of the problem is that the Arm64 developers need to incorporate all APIs from Arm into their code if they want to maintain portability, although they really want to abandon those APIs in favor of better ones. In the current case, the `/proc/cpuinfo` files will have to be brought in line with each other, even if there's code out there that depends on their differences.

Russell King had a bit to say about the situation, in the form of a cautionary tale:

As ARM64 wants to be compatible with ARM32 (in that it wants to be able to run ARM32 applications), ARM64 *has* to offer a compatible user API for everything that is a user API.

That means you have to generate an ARM32 compatible `/proc/cpuinfo`,

ARM32 compatible `hwcap` information, ARM32 compatible signal structures, ARM32 compatible everything else. Which means you basically need to have a copy of the ARM32 core code in ARM64, even if you want a different native ARM64 user API.

This is *exactly* the reason why architectures like X86 decided it was silly having separated 32- and 64-bit, and why they went through a process of merging the two together. A lot of the code was identical, and a lot of the 32-bit-specific code was needed for 64-bit to provide the 32-bit API.

Right now, you're finding out this the hard way, and hitting these API problems in the process, and going "oh fsck" when you hit them—quite simply because you've backed yourselves into a corner over this. You have established a different ARM64 API because you didn't want the ARM32 legacy, but then you've found that you *do* need the ARM32 legacy. Now you're in the position of having to change the ARM64 API, possibly breaking ARM64 applications in the process.

—**ZACK BROWN**



Hosted By



LinuxFest Northwest

April 25th & 26th
Bellingham, WA

- All things Open Source
- 40+ Exhibitors
- 80+ Presentations
- 1500+ Attendees
- Prizes and after party
- FREE admission & parking
- Bring the whole family!



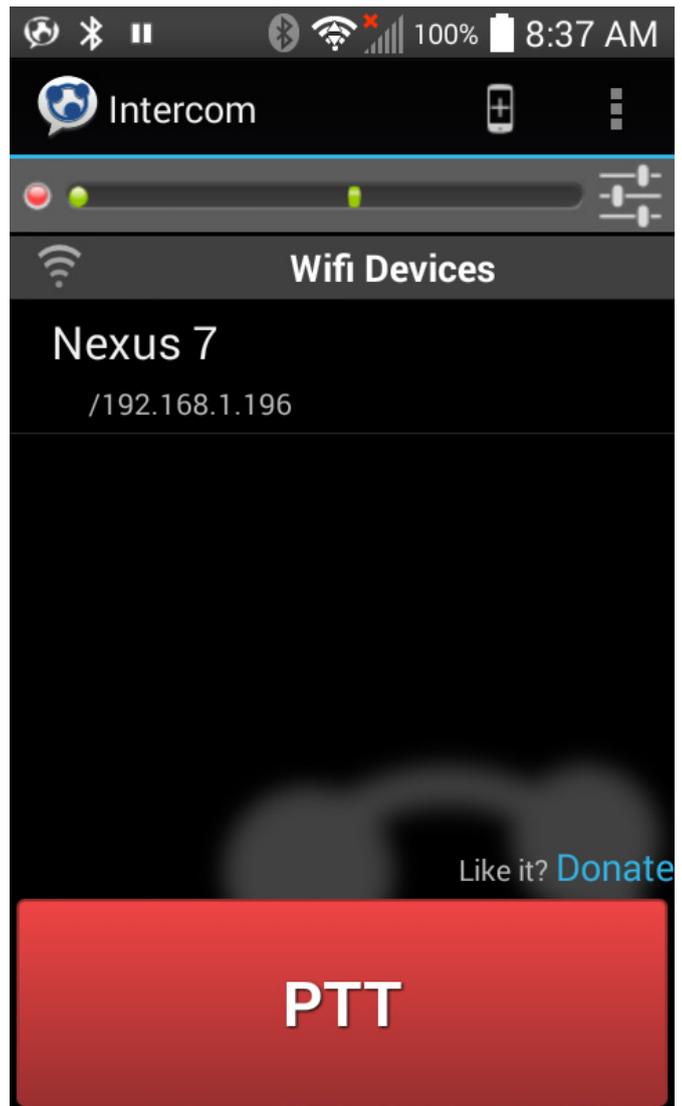
linuxfestnorthwest.org

Android Candy: Intercoms

Ever since my “tiny \$20 tablet” project (see my Open-Source Classroom column in the March 2015 issue), I’ve been looking for more and more cool things to do with cheap Android devices. Although the few obvious ones like XBMC or Plex remotes work well, I’ve recently found that having Android devices around the house means I can gain back an old-school ability that went out of style in the late 1980s—namely, an intercom system.

If you remember the big white boxes screwed to the wall in the garage and basement so you could talk to the person in the kitchen about making sandwiches, you know exactly what I mean. With multiple Android devices around the house, it means I can send an audible message quickly without the need to call or text.

There are several great intercom apps in the Google Play store, like “Intercom for Android”, “Intercom for Android” (yes, more than one!), “Tiki” and so on. Each has its own set of advantages and disadvantages. Some work well over great distance by using the Internet, and some work with an ad hoc Wi-Fi connection between rooms. Whatever



your instant communication needs, the intercom idea is one worth exploring, especially if you have multiple Android devices around your house. Download a few apps today, and let me know when that sandwich is done!

—**SHAWN POWERS**

POSSCON

JOIN THE LEADING

Open Source Experts

AND COMPANIES IN THE U.S.

April 14 & 15

COLUMBIA, SC

JUST A FEW OF THE FEATURED SPEAKERS:



ELIZABETH JOSEPH
Hewlett Packard, Ubuntu



CHRIS ANISZCZYK
Twitter, Head of Open Source



ANDY HUNT
Pragmatic Bookshelf, Programmer



CAROL SMITH
Google



JASON HIBBETS
Opensource.com



TIBOR VASS
Docker



ERICA STANLEY
Acire, Women Who Code ATL



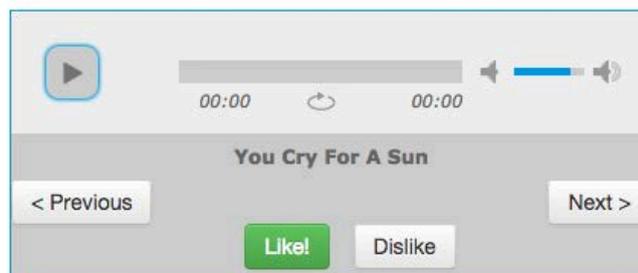
SEMMY PUREWAL
Netflix

www.posscon.org

Play for Me, Jarvis

Listen to unique, computer-generated music...

Computoser is an "artificial intelligence" algorithm that turns the computer into a music composer. Each track you hear is algorithmically generated.



Download: [Original .midi](#) | [.mp3](#) | [MusicXML](#) (licensed under Creative Commons)

Elon Musk is known to be particularly apprehensive about artificial intelligence (<https://twitter.com/elonmusk/status/495759307346952192>). Although many of us are both excited and worried about the potential future of AI, most don't need to fear computers taking over in the creative realms of society.

Or do we?

Heading over to <http://computoser.com> both delights and concerns me. Using nothing more than algorithms and preloaded data, the Web site will generate completely unique

and oddly pleasant electronic music. I expected the results to feel bland and single-dimensional, but honestly, some of the songs are incredible and seem to relay emotion that obviously was never there to begin with.

Although it might be the downfall of civilization and might mean the unemployability of creative folks like myself, you can taste the computer's creativity yourself. There's also an app in the Google Play store if you want some Skynet music in your pocket: <http://play.google.com/store/apps/details?id=com.computoser>.

—**SHAWN POWERS**

Non-Linux FOSS: .NET?

No, really! While on a normal day, the word “Microsoft” can be used as an antonym for “Open”, the world of .NET seems to be going legitimately open source. I have to confess that my limited development knowledge doesn’t give me a full appreciation of the significance of .NET and ASP.NET things being released into the open-source world the end of last year, but seeing actual GitHub repositories of the core technologies is encouraging.

Are you a Linux developer interested in branching into .NET programming now that it’s open source? Are you a .NET developer who wants to develop for non-Microsoft platforms now that it’s officially supported? Do you think Microsoft has done too little too late? Whatever your take, the .NET Foundation seems to be doing more than just releasing source code; the GitHub repositories are a significant step toward a real community. Check out the wide selection of Git repositories at <https://microsoft.github.io>.

—**SHAWN POWERS**

They Said It

If writers stopped writing about what happened to them, then there would be a lot of empty pages.

—*Elaine Liner*

The time to repair the roof is when the sun is shining.

—*John F. Kennedy*

The customer doesn’t expect everything will go right all the time; the big test is what you do when things go wrong.

—*Sir Colin Marshall*

If the universe is bigger and stranger than I can imagine, it’s best to meet it with an empty bladder.

—*John Scalzi*

Courage is the price that Life exacts for granting peace.

—*Amelia Earhart*

Designing Foils with XFLR5

For any object moving through a fluid, forces are applied to the object as the fluid moves around it. A fluid can be something like water, or even something like the air around us. When the object is specifically designed to maximize the forces that the fluid can apply, you can designate these designs as airfoils. A more common name that most people would use is a wing. The shape of a wing, or airfoil, determines the forces that are applied to it when it moves through a fluid or the air. These forces also depend on the speed of motion through the fluid and the direction of flow around the airfoil.

With all of these parameters, how can you design airfoils? How do you optimize airfoils for a particular use? You need some way of analyzing all of this information—specifically, you need software that can run the numbers and do the calculations. There are very complex pieces of software that can analyze hydrodynamic problems in the abstract. But, with airfoils, you can limit the problem to such a degree that the equations

are greatly simplified.

One of the software packages available to do these kinds of calculations is XFLR5 (<http://www.xflr5.com/xflr5.htm>). XFLR5 started as a fork of the much older xfoils program, but it has been extended with extra functionality.

Installation on Debian-based distributions can be done with the command:

```
sudo apt-get install xflr5
```

That command should install the XFLR5 documentation package as well.

When you start XFLR5 the first time, it is not very flashy. In fact, on my system, I end up with a plain black window.

Although you can design your own airfoil from scratch, doing so can be fairly tedious. It is much easier to take a previously designed airfoil as a starting point and make alterations. A good database of airfoil designs is located at the UIUC Airfoil Coordinates Database, containing nearly 1,600 airfoils

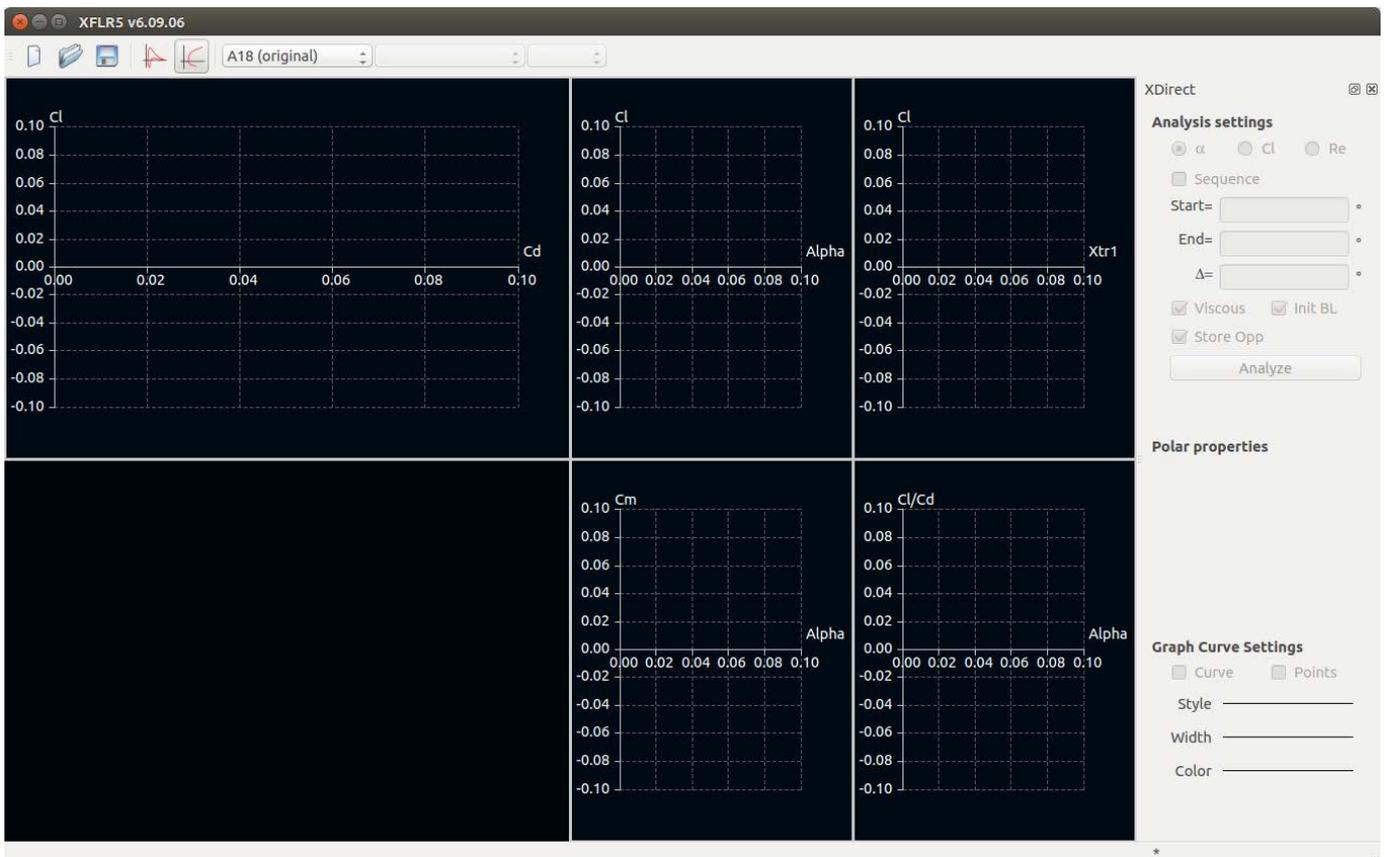


Figure 1. Opening a DAT file loads the data and switches to the polar view.

(http://m-selig.ae.illinois.edu/ads/coord_database.html). The database contains DAT files, which contain the information you need to use in XFLR5. They also have GIF files, allowing you to see what the airfoil looks like before downloading the DAT file. Once you choose one, download the related DAT file and open it in XFLR5 by clicking on the menu item File→Open.

You can change the view to the OpPoint View by clicking the menu item View→OpPoint View or by pressing the F5 key.

At the bottom of the window, you

can see airfoil characteristics, such as the thickness. Let's say that the first design change you need to make is to generate a thinner airfoil. You can do this by clicking the menu item Design→Scale camber and thickness or pressing the F9 key. This pops up a new window where you can change those characteristics.

When you make your changes and click OK, XFLR5 will ask you if you want to overwrite the current airfoil or if you want to create a new one. If you choose to create a new one, you will be able to switch between the

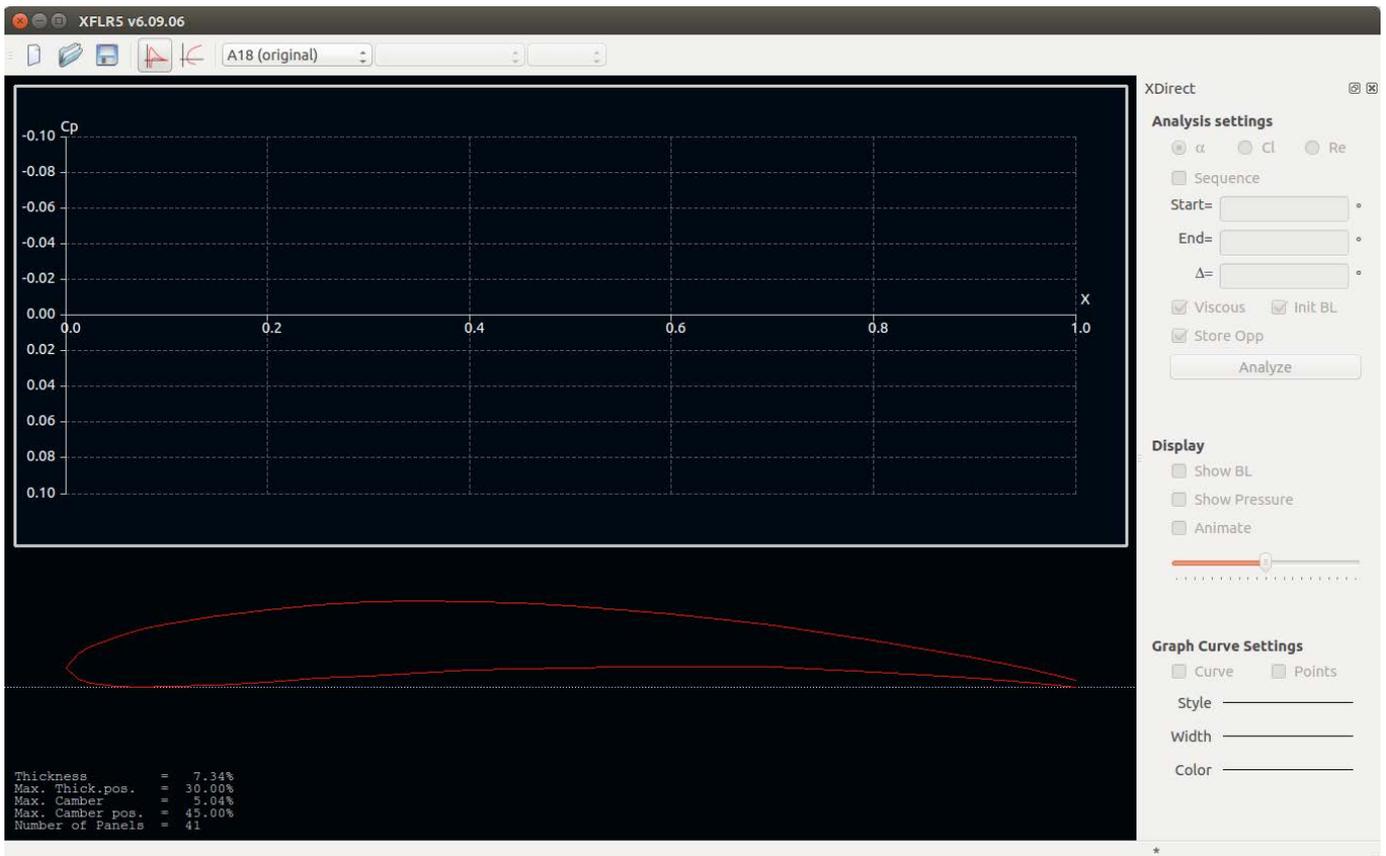


Figure 2. The OpPoint View gives you a traditional cross-section view of an airfoil.

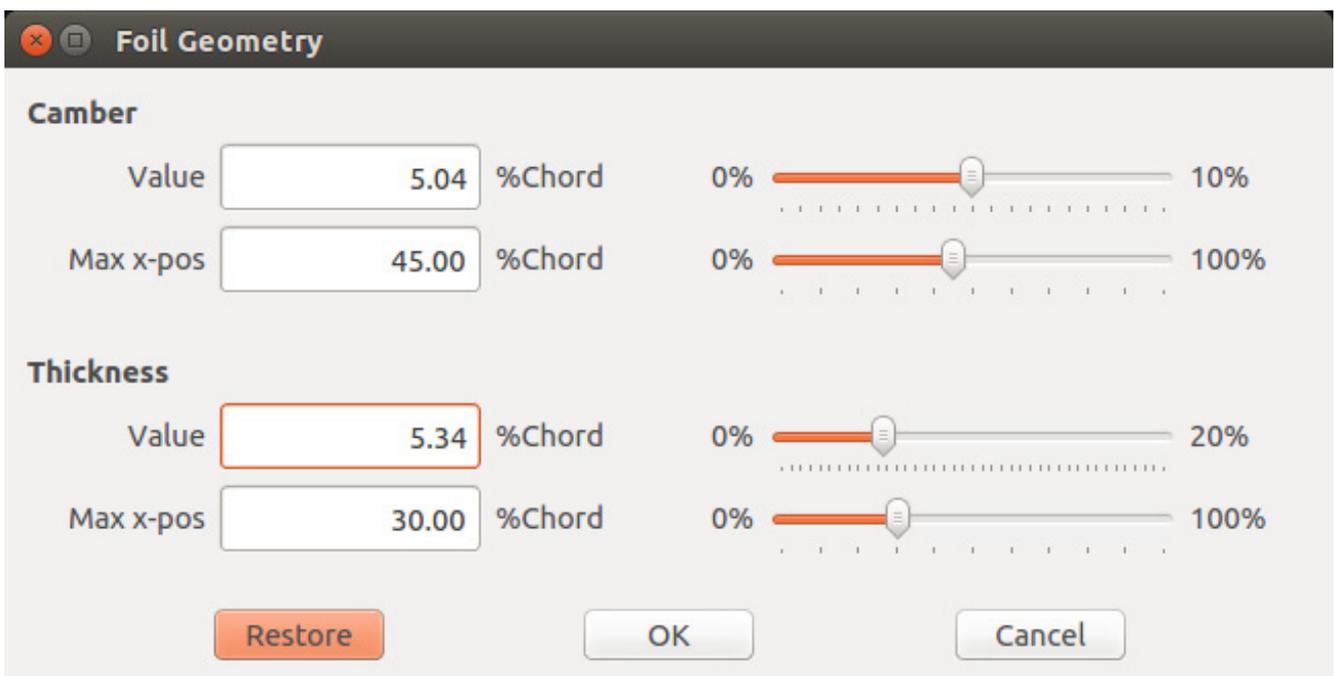


Figure 3. A new window lets you change the thickness and camber of your airfoil.

various loaded airfoils using the drop-down at the top of the window.

Now, let's generate the polars to do some analysis on this new airfoil that you created. The easiest way to do this is to click the menu item Analysis→Batch Analysis. If you have a multi-core or multi-CPU machine, you can select the Multi-threaded Batch Analysis menu item instead to get it done more quickly. This pops up a new window where you can select the range of Reynolds numbers to do the analysis over, and the step size for each

Reynolds number to use.

You also can select whether to do this for only the current foil, or you can do the analysis for a list of foils. Once you have all of the parameters set, you can click on the Analyze button at the bottom of the window. For each step, you will see an output message in the top right-hand pane telling you how many iterations were needed for convergence, and in the bottom left-hand pane, you will see the actual plotted values for each iteration of each step. Once it finishes, you can close this analysis

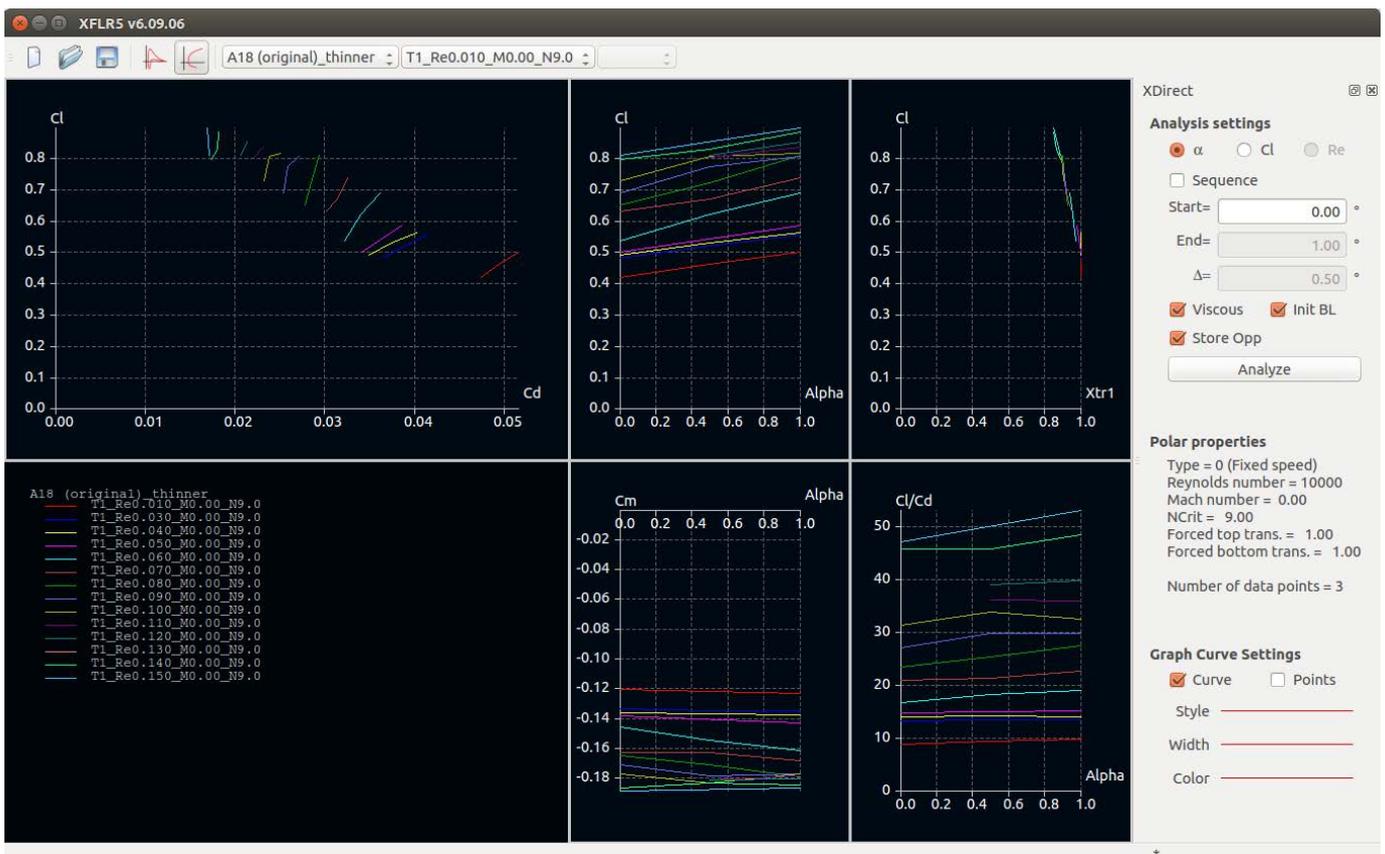


Figure 4. There are several polar graphs showing you the results of your analysis.

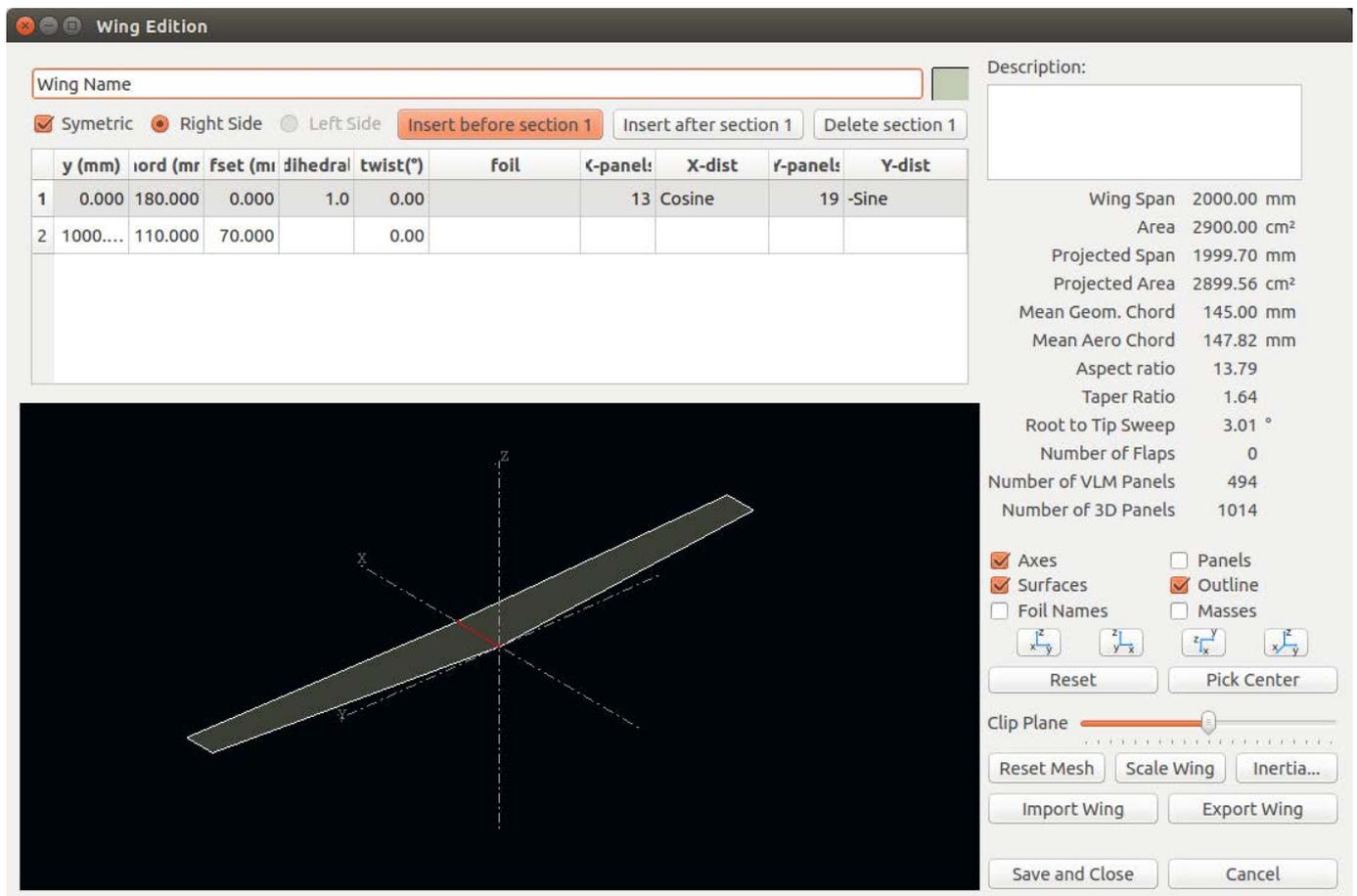


Figure 5. Once you analyze your airfoil, you can start designing a full wing.

window and go back to the main window. The polar view will be opened automatically, showing you all of the polar plots. You can select a single polar plot of interest by clicking on the menu item Polars→Polar Graphs, and then selecting the graph you want to see.

Now that you have a foil and its polars calculated, you can move on to three-dimensional analysis and look at a full wing design. Clicking on the menu item File→Wing and Plane Design will pop up a new

view where you can design a new full wing. Within this new view, you will need to click on the menu item Wing-Plane→Define a New Wing to open up a new window to create your new wing.

You can give it a name and description, and set all kinds of characteristics. You also can select sections of your wing and use the airfoils that you designed in the earlier step to provide the cross-sections of the wing along its length. Once you are happy, you can click on the Save

Analysis Definition

Polar Name
Wing Name 1
 Auto Analysis Name
T1-10.0 m/s-LLT

Polar Type
 Type 1 (Fixed Speed)
 Type 2 (Fixed Lift)
 Type 4 (Fixed aoa)

Plane and Flight Data
Free Stream Speed = 10.00 m/s
 α = 0.00 °
 β = 0.00 °

Flight Characteristics
Wing Loading = 0.000kg/cm²
Tip Re = 73 000
Root Re = 119 000

Inertia properties
 Use plane inertia
Plane Mass = 0.000 kg
X_CoG = 0.000 mm
Z_CoG = 0.000 mm

Aerodynamic Data
Unit International Imperial
 ρ = 1.225 kg/m³
 ν = 0.000 m²/s

Wing analysis methods
 LLT
 VLM
 3D Panels

Ground Effect
 Ground Effect
Height = 0.00 mm

Options
 Viscous
 Tilt. Geom.
 Ignore Body Panels

Reference Area and Span for Aero Coefficients
 Wing Planform
 Wing Planform projected on xy plane

OK Cancel

Figure 6. You can do an analysis of the entire wing as a whole.

and Close button and see your new wing displayed in the main window.

You now need to test your wing and

analyze how it will behave when it starts moving through the air. Clicking on the menu item Analysis→Analysis

Definition will bring up a parameter window where you can set up the details of your analysis.

Once everything is set, click on the OK button to get back to the main window. Depending on what you are trying to do, you may need to set some advanced settings by clicking the menu item Analysis→Advanced Settings. Here, you can change items like the maximum number of iterations, the relaxation factor or the panel boundary conditions.

The final step is to set the number of sequential steps in the right-hand pane, under the analysis settings section. Clicking the Analyze button in the right-hand pane starts off the whole process.

You also can design an entire plane, which is made up of one or more wings as well as a tail and fins. You can do this by clicking on the menu item Wing-Plane→Define a New Plane. In this part of XFLR5, you can define your entire plane and see how it behaves as a complete object.

With XFLR5, you now can design your very own aircraft wings. This tool should be helpful for anyone, but especially for hobbyists who are designing their own RC aircraft. Cost is no longer a barrier for letting your creativity run wild. You might come up with a totally new, awesome wing design.

You can find more documentation at the main XFLR5 Web site, which can help you do more complex analysis. Some of it was written based on older versions, however, so the location of certain functions within XFLR5 has moved, and you may need to do some investigative work to figure out how to do similar tasks. But, it is definitely worth the minor amount of work involved.

—**JOEY BERNARD**

LINUX JOURNAL on your **Android** device

Download the app now on the **Google Play Store**



www.linuxjournal.com/android



PERCONA
LIVE

115+

MySQL experts

13

tutorials

120

breakout sessions

MySQL™ Conference & Expo **2015**

April 13-16 • Santa Clara, CA

[> Check it out](#)

Here, Have Some Money...



I love Bitcoin. It's not a secret; I've written about Bitcoin mining and cryptocurrency in the past. I'm the first to admit, however, that we're at the very beginning of the cryptocurrency age. Although it's becoming easier and easier to use Bitcoin (see <http://www.coinbase.com>, for example), the limited use cases combined with the wild volatility of price make Bitcoin the wild wild west of the financial world.

There are a few awesome ideas, however, that are brilliant in their simplicity. Certainly things like the Humble Bundle folks integrating Bitcoin purchasing and Overstock.com allowing Bitcoin purchases are great first steps. Those are really just re-inventing the wheel, however, because we already can buy things using existing forms of currency.

Enter ChangeTip.

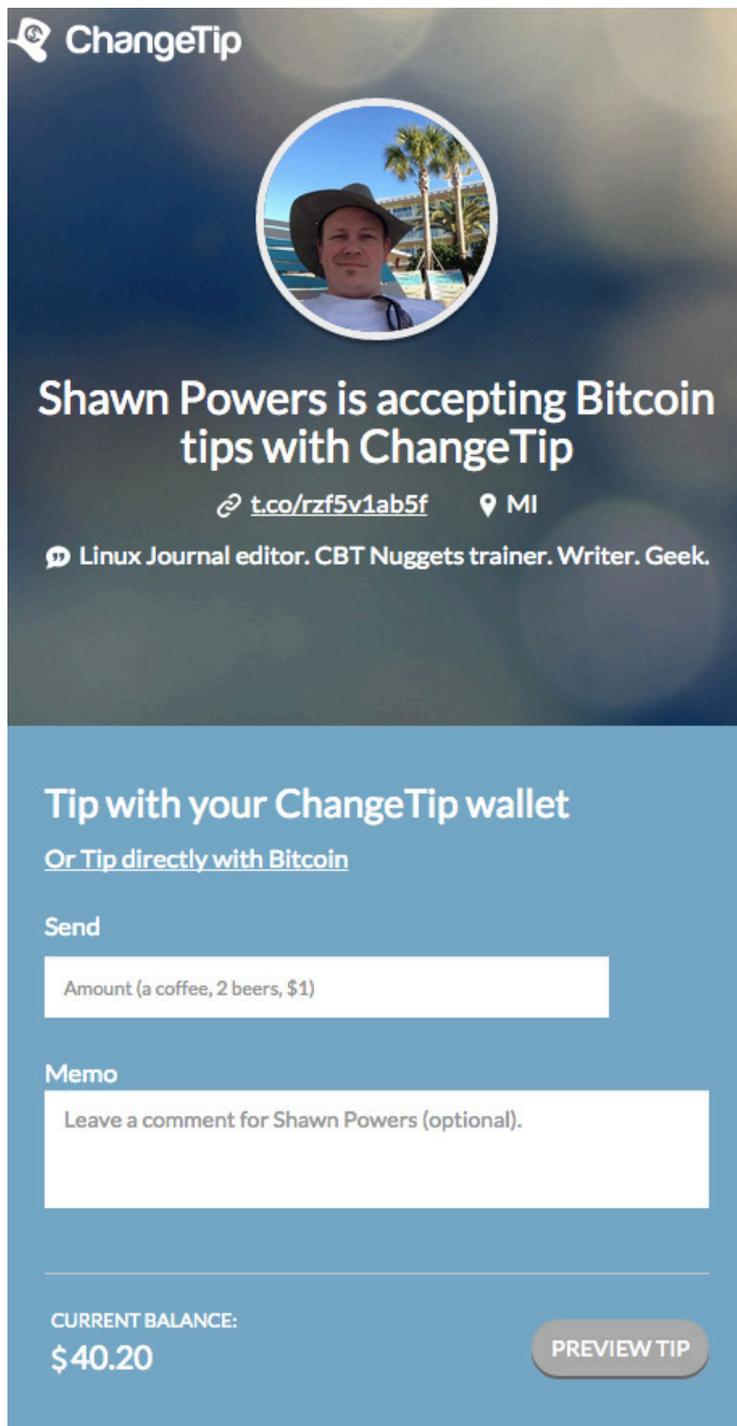
Sending someone a tip or donation on the Internet generally has been done with something like PayPal. That's all well and good, but it's fairly cumbersome. The folks at <http://www.changetip.com> have

made sending money over the Internet extremely simple, and fun!

With its integration into Twitter, Facebook, GitHub, Google+, YouTube, Reddit and more, ChangeTip makes sending money as simple as sending a Tweet. Thanks to the world of OAUTH, you don't even need to create an account on ChangeTip.com to claim or send funds. If you send money to people who don't have accounts, they simply sign in to ChangeTip via the social-media account from which you sent it to them, and the money will be there waiting for them. Oh, and the money? It's Bitcoin!

With its seamless integration to Coinbase, ChangeTip makes actual financial transactions secure, simple, and did I mention simple? Check it out today at <http://changetip.com>, or visit my personal page at <http://shawnpowers.tip.me>. And, if you want incentive to try it out, I originally planned to include a bunch of "one-time links" in this article that could be claimed for \$1 each. It turns out that the one-time links expire after a week. So although it might have been

On April 1st, 2015, watch my personal Twitter account (@shawnpowers), and I'll tweet out some ChangeTip URLs worth actual money.



The screenshot shows a ChangeTip profile for Shawn Powers. At the top left is the ChangeTip logo. Below it is a circular profile picture of Shawn Powers wearing a hat and sunglasses. The main text reads: "Shawn Powers is accepting Bitcoin tips with ChangeTip". Below this is a link "t.co/rzf5v1ab5f" and a location pin "MI". A bio line says "Linux Journal editor. CBT Nuggets trainer. Writer. Geek." The bottom section is titled "Tip with your ChangeTip wallet" and includes a sub-link "Or Tip directly with Bitcoin". There is a "Send" button, a text input field for the amount (pre-filled with "Amount (a coffee, 2 beers, \$1)"), and a "Memo" field with the placeholder text "Leave a comment for Shawn Powers (optional)". At the bottom left, it shows "CURRENT BALANCE: \$40.20" and a "PREVIEW TIP" button.

a great April Fool's joke, I really want to give everyone a chance to claim some tips, so keep reading!

On April 1st, 2015, watch my personal Twitter account (@shawnpowers), and I'll tweet out some ChangeTip URLs worth actual money. Be the first to click the link, and you will be the proud owner of \$1 from yours truly. I'll try to spread out the tweets throughout the day, so don't worry if you're reading this after work. It probably won't be too late!

Due to its awesome use of cryptocurrency and social media, along with the ease of use and ability to give money to folks who read my article, this month's Editors' Choice award goes to ChangeTip. Let's change the world!

(Note: I'm not asking for tips! I know many of you are really kind and generous, so I want to make it perfectly clear that posting the link to my ChangeTip page isn't my way of asking for tips. It's just so you can see how simple ChangeTip is to use!)—**SHAWN POWERS**



REUVEN M.
LERNER

Django Templates

Displaying dynamic data with Django's template language.

In my last article (February 2015), I explained how to create a simple Django project ("atfproject") and inside that, create a simple application (atfapp). The application worked in that if you went to the URL `http://localhost:8000/hello/Reuven`, you got the text "hello, Reuven".

This was thanks to a combination of the view function:

```
def hello(request, name):  
    return HttpResponse("hello, {}".format(name))
```

and the URL pattern that I created:

```
urlpatterns = patterns('',  
    url(r'^hello/(?P<name>\w+)$', hello),  
    url(r'^admin/', include(admin.site.urls)),  
)
```

Notice that in the first URL pattern, I define a named group, "name", as a string of alphanumeric characters (`\w+`). The captured characters then

are passed to the view function's "name" parameter, which allows the view function to accept and display the values within the view function.

Now, this does work, but if you're thinking that this is a pretty awkward way to display text, as a string within a view function, you're not alone. Indeed, aside from producing extremely small pieces of text, you're likely never going to return HTML directly from a view function. Instead, you'll use a template.

This shouldn't come as a surprise to anyone who has been doing Web development for any length of time. Every Web framework I have used has some form of templates. Unfortunately, every Web framework uses a unique type of template, with a new and different way to integrate HTML and the sorts of dynamic content that you need to present.

So in this article, I describe how Django's templates work, allowing

you to generate dynamic content for your users.

Invoking Templates

It's important to remember that Django's templates are HTML files with a bit of extra code thrown in. And even saying that there is "code" in Django templates probably is exaggerating things a bit. The template syntax is designed explicitly to reduce the amount of code that you have to write, but also to reduce the amount of code that is executed in the template. By removing code from the template and putting it into your view methods (and models), you make your application more modular, more flexible and more testable.

To start with Django templates, you don't need to know anything special. That's because a plain-old HTML file is a fine Django template. Inside the "atfapp" application, let's create a new subdirectory called templates. This is where Django will look for your templates. You always can configure this differently by setting the `TEMPLATE_DIRS` variable inside the application's settings.

Here is a simple template that I created and then put inside `atfapp/templates/hello.html`:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <h1>Hello!</h1>
    <p>Hello out there!</p>
  </body>
</html>
```

In order to get Django to display this template, you need to change your "hello" view function (defined in your application's `views.py`) such that it renders its response using the template. The easiest way to do that is to use the `render_to_response` function, defined in the `django.shortcuts` package. Thus, change `views.py` to read as follows:

```
from django.shortcuts import render
from django.http import HttpResponseRedirect
from django.shortcuts import render_to_response

def hello(request, name):
    return render_to_response('hello.html')
```

Notice that it isn't enough to invoke `render_to_response`. As with all functions in Python, you must explicitly return the object that `render_to_response` returned to you.

With the template in place and

the view function updated, you now can reload the application at `http://localhost:8000/hello/Reuven`. And...well, you'll probably see a debugging screen from Django, telling you that the template wasn't found. The problem here is that when you use `render_to_response`, it looks in the template directories of all of the registered Django applications within the project. But wait, you never registered your application! Thus, although you can invoke view functions from within `atfapp`, Django won't look in the `atfapp/templates` directory, because it's not a registered app.

The simple solution is to go to `settings.py` in the main project's configuration directory (`atfproject/atfproject/settings.py`, in this case), find the definition of `INSTALLED_APPS`, and then add the following line to the end:

```
'atfapp'
```

Note that you'll have to add a comma to the end of the previous line.

With this in place, Django's template system will find your template. Going to `/hello/Reuven` (or any other URL under `/hello/`) now will display your template.

Passing Variables

Of course, this basic "hello" template isn't really demonstrating the power of a Web application. In order for that to happen, you're going to need to pass values to the template, which then will mix your values with the HTML and display the result to the user.

So, you need to do two things. First, you need to change your invocation of `render_to_response`, such that it passes one or more variable values. If you are at all familiar with Python, you won't be surprised to discover that you will do this by passing a dictionary to `render_to_response`, in which the keys are the variable names you want to pass. For example:

```
def hello(request, name):  
    return render_to_response('hello.html', {'name':name})
```

In this example, you take the parameter "name", which was assigned via the URL, and pass it as the value in your dictionary. The key is called "name", which I admit can be a tiny bit confusing, but it still makes the most sense here.

In your template, Django looks for double curly braces: `{{` and `}}`. Django will look inside those braces and look for the name in the dictionary

that it was passed:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <h1>Hello!</h1>
    <p>Hello, {{name}}!</p>
  </body>
</html>
```

With just these two changes in place—and without even having to restart the server—the contents of your URL now affect the template’s output.

You can pass any number of name-value pairs in the dictionary defined in `render_to_response`. The passed values might come from arguments passed to the view function, from the database or from a remote server. From the template’s perspective, it has access only to the data that was passed to it and doesn’t really care where the rest of the data came from.

Of course, there are times when you might want to have text appear conditionally. This also is possible with Django templates. Instead of using `{{ and }}` around variable names, you can use `{% and %}` around commands. Now, these are not Python commands, so don’t expect the syntax, names or

behavior to be identical. Also, because you don’t have Python’s indented block syntax, you must end your “if” condition (for example) with an “endif” tag.

Given that information, you probably can figure out what happens in this template:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <h1>Hello!</h1>
    {% if name == 'Reuven' %}
    <p>Hello, master {{name}}!</p>
    {% else %}
    <p>Hello, {{name}}!</p>
    {% endif %}
  </body>
</html>
```

The template gets a parameter “name”, which it then compares with the string “Reuven”. If I’m the named person, it prints one message. Otherwise, it prints another message.

Loops and Filters

The previous example shows what it looks like when you take a value from a URL and then want to pass it to a template for display. Parameters to view functions always are going to be passed as strings.

However, there is no reason why you can't pass another data structure, such as a list, tuple or dict. If you do this (or, to be honest, if you pass any iterable), you use the template's looping function, which operates identically to Python's "for" operator, but with the addition of a closing "endfor" tag.

Let's change the view function to work as follows:

```
def hello(request, name):
    return render_to_response('hello.html', {'name':name,
                                             'children': ['Atara', 'Shikma', 'Amotz']})
```

As you can see, you're now going to pass a section variable to your template, containing my children's first names. Inside your template, you can iterate over this variable, almost precisely as you would within a non-Django, non-template Python program:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <h1>Hello!</h1>
    {% if name == 'Reuven' %}
    <p>Hello, master {{name}}!</p>
    {% else %}
    <p>Hello, {{name}}!</p>
    {% endif %}
```

```
<h2>Children</h2>
<ol>
  {% for child in children %}
  <li>{{child}}</li>
  {% endfor %}
</ol>

</body>
</html>
```

In this example, you have combined HTML's "ol" tag to provide an enumerated list, along with a "for" loop in Django's templates. Because "child" is defined as a variable within the loop, you can use the {{child}} syntax to indicate where you want the child's name to appear.

Now, if you're printing a list of names, it's possible that the strings have become all lowercase. Let's say you would like to ensure that the names follow English-language rules, in which the first character is capitalized. Now, if you were using straight Python, you could use the str.upper method, as in:

```
<li>{{child|capfirst}}</li>
```

But, if you change the children's names to lowercase and then change the template to look as it does above...well, let's just say that it won't work. This is part of Django's philosophy of keeping

REGISTER TODAY!



2015 USENIX Annual Technical Conference

JULY 8-10, 2015 • SANTA CLARA, CA
www.usenix.org/atc15

USENIX ATC '15 brings leading systems researchers together for cutting-edge systems research and unlimited opportunities to gain insight into a variety of must-know topics, including virtualization, system administration, cloud computing, security, and networking.

Co-located with USENIX ATC '15:

HotCloud '15

7th USENIX Workshop on Hot Topics in Cloud Computing

JULY 6-7, 2015
www.usenix.org/hotcloud15

Researchers and practitioners at HotCloud '15 share their perspectives, report on recent developments, discuss research in progress, and identify new/emerging "hot" trends in cloud computing technologies.

HotStorage '15

7th USENIX Workshop on Hot Topics in Storage and File Systems

JULY 6-7, 2015
www.usenix.org/hotstorage15

HotStorage '15 is an ideal forum for leading storage systems researchers to exchange ideas and discuss the design, implementation, management, and evaluation of these systems.



Stay Connected...



twitter.com/usenix



www.usenix.org/facebook



www.usenix.org/youtube



www.usenix.org/linkedin



www.usenix.org/gplus



www.usenix.org/blog

GO AHEAD. BE A HERO.



Drupal is increasingly the CMS of choice for large scale websites.

Join us at DrupalCon Los Angeles to learn how to scale Drupal for the enterprise and optimize for top performance. Who knows what heroic feats you will accomplish with your new skills? You may just save the day at your organization.

Become a hero at events.drupal.org. Join us in Los Angeles this May 11 - 15, 2015.



Brought to you by the Drupal Association.

Image credit to Jimmy Baikovicius (jikat) on Flickr. Some rights reserved.



DAVE TAYLOR

Where's That Pesky Hidden Word?

Word search—Dave tackles the complex task of writing a script to generate word searches from a list of words. Doable? We'll see.

I've been promising my 11-year-old for a long time now that I'd write a program that would let you build custom word searches based on a list of words given by the user. I wrote one years and years ago in C, but since I can't find that code any more and wanted to tackle another interesting project for this column, that's what I'm going to look at herein.

There aren't any well-established algorithms around word searches, so it's also a good opportunity to come up with something from scratch, which means that as with the most interesting projects, we'll actually need to think about the program before we start coding. It's good discipline!

I also should admit that I've loved solving word search puzzles since

I was a young pup, so there's an element of enjoyment for me in this project too, not just the pleasure of doing something for my daughter!

In case you've never seen a word search, it's typically a grid of letters, and within that is a set of listed



Figure 1. Word Search Example

And this brings up one interesting characteristic of word searches: there always are incidental words that can appear, so one important task for the resultant program is to ensure that no obscenities appear.

words that can be found running horizontally, vertically or diagonally, spelled forward or backward. Figure 1 shows a tiny example.

Looking Figure 1, how many words can you find? I can find CAT, DOG, GOD (which is, of course, DOG backward), TIC, COP and, more nerdy, ROM and ARG too. And this brings up one interesting characteristic of word searches: there always are incidental words that can appear, so one important task for the resultant program is to ensure that no obscenities appear.

Upon first glance at a word search, it seems like the way to do it is to populate the grid randomly, then flip letters to make the words fit. But, it seems to me that a better strategy is essentially to make a crossword puzzle, then fill in the empty holes with random letters.

So that's going to be our first strategy for building the word

search. Each word also will be randomly oriented horizontal, vertical or diagonal, as well as forward or backward. For now, let's just worry about forward or backward, meaning the initial word orientation code will look like this:

```
orient()
{
    # randomly pick an orientation and
    # shift the word to match
    local direction;
    word=$1 # to make things neat and tidy
    if [ $(( RANDOM % 2 )) -eq 1 ] ; then
        # we need to reverse the value of $word
        word="$(echo $word | rev )"
    fi
}
```

Arrays are created by initializing them in the Bash shell, in a format that looks like this:

```
arrayname=( value1, value2, ... valueN )
```

And since this is going to be a lot about arrays, let's start by loading up the wordlist as an array, orienting words randomly as we proceed.

First, here's a really easy way to read a file in as an array of word values:

```
wordlist=( $(cat $1) )
```

Easy enough, but now let's step through the array word by word to reverse any that are randomly selected

by the orient() function:

```
count=0
while [ ! -z "${wordlist[count]}" ]
do
    orient ${wordlist[count]};
    wordlist[$count]=$word
    echo "word $count = ${wordlist[count]}"
    count=$(( $count + 1 ))
done
```

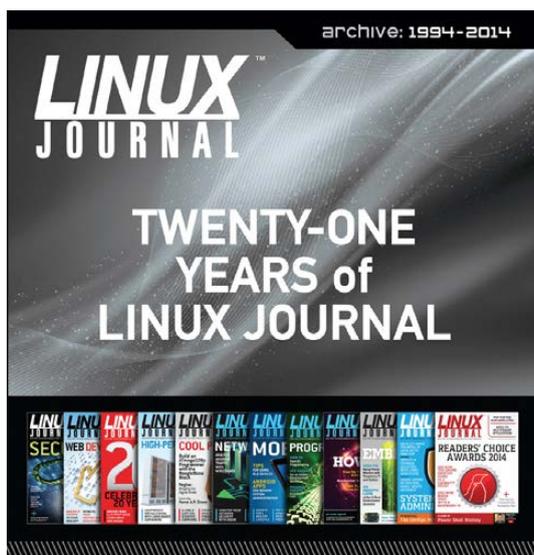
With just this snippet and a word file that contains "cat", "dog" and "car", a single invocation looks like this:

```
$ sh wordsearch.sh wordlist.txt
word 0 = cat
word 1 = god
word 2 = rac
```

That's a reasonable enough start. We now can read in a wordlist file and randomly reverse individual words as we go. Now, let's create a grid array and try inserting the words one by one.

And here's a wrinkle associated with the Bash shell: although it supports arrays, it doesn't support multidimensional arrays, which is rather a pain in the booty. So to have a 5x5 grid, we'll need five arrays of five elements. To start, let's initialize them at the beginning of the script:

LINUX JOURNAL ARCHIVE DVD



NOW AVAILABLE

www.linuxjournal.com/dvd

```
row1=( "-" "-" "-" "-" "-" )
row2=( "-" "-" "-" "-" "-" )
row3=( "-" "-" "-" "-" "-" )
row4=( "-" "-" "-" "-" "-" )
row5=( "-" "-" "-" "-" "-" )
```

Then, further down, a simple function will allow us to print out the grid in an attractive format:

```
showgrid()
{
    echo "${row1[0]} ${row1[1]} ${row1[2]} ${row1[3]}
        ${row1[4]}"
    echo "${row2[0]} ${row2[1]} ${row2[2]} ${row2[3]}
        ${row2[4]}"
    echo "${row3[0]} ${row3[1]} ${row3[2]} ${row3[3]}
        ${row3[4]}"
    echo "${row4[0]} ${row4[1]} ${row4[2]} ${row4[3]}
        ${row4[4]}"
    echo "${row5[0]} ${row5[1]} ${row5[2]} ${row5[3]}
        ${row5[4]}"
}
```

We'll end up rewriting this function down the road to make it more flexible for an N x M size grid, but for now, let's just proceed with 5x5 so we can get into the algorithm itself.

Now the actual work of the script: inserting words into the grid.

Initially, of course, it's easy because we're pretty much guaranteed the word will fit if it's

less than five letters long, but as more and more words are put into the grid, it becomes harder to fit each one.

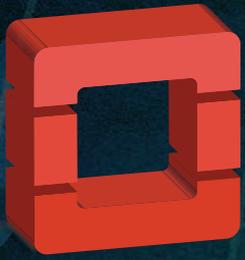
To simplify things, we're going to look at inserting words only horizontally or vertically to start. It turns out that diagonal insertions are a bit more nuanced. That's okay, we'll circle back and add it once we get the basics working.

To start, the function `fitword()`, given a word (that might already be reversed), randomly picks an orientation and starting location that'll fit, then hands it to a horizontal or vertical insertion function for actual placement testing:

```
fitword()
{
    # fit word "$1" into the grid with a random orientation
    success=0

    wordlength=$( echo $1 | wc -c )    # always +1
    wordlength=$(( $wordlength - 1 )) # and now it's fixed
    case $(( $RANDOM % 2 )) in
        0 ) # horizontal
            until [ $success -eq 1 ] ; do
                startpoint=$(( $cols - $wordlength ))
                col=$(( $RANDOM % $startpoint ))
                row=$(( $RANDOM % 5 ))
                Hinsert $1 $col $row

                success=$?                # what does Hinsert return?
            done
        ;;
    ;;
```

openstack™
LIVE

26

Openstack experts

6

tutorials

20

breakout sessions

Openstack™ Conference & Expo 2015

April 13-16 • Santa Clara, CA

➤ Check it out



KYLE RANKIN

Libreboot on an x60, Part II: the Installation

If you weren't scared away by my column last month, you must be ready to flash your BIOS.

In my last article, I introduced the Libreboot project: a free software distribution of coreboot, which is itself an open-source BIOS replacement. I also talked about some of the reasons you may want to run a free software BIOS and discussed some of the associated risks. If you made it through all of that and are ready to flash your BIOS, this article will walk you through the process.

Get Libreboot

Libreboot is available via binary distributions that make it easy to install (which is what I cover below) as well as source code distributions at <http://libreboot.org/#releases>. To get the latest binary release, go to <http://libreboot.org/docs/release.html>, and be sure to download both the .xz as well as the corresponding .xv.sig

file, such as:

- http://libreboot.org/release/20150208/libreboot_bin.tar.xz
- http://libreboot.org/release/20150208/libreboot_bin.tar.xz.sig

Once you download the files, use `gpg --verify` to validate that the signature matches:

```
$ gpg --verify libreboot_bin.tar.xz.sig libreboot_bin.tar.xz
gpg: Signature made Tue 14 Oct 2014 09:07:32 PM PDT using
↳RSA key ID 656F212E
gpg: Good signature from "Libreboot Releases (signing key)"
↳<releases@libreboot.org>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs
↳to the owner.
Primary key fingerprint: C923 4BA3 200C F688 9CC0 764D
↳6E97 D575 656F 212E
```

Note that since I haven't added the Libreboot GPG key to my keyring and trusted it, all it can do here is validate that the signature matches whatever key generated the .sig, not that it's the official Libreboot key. To do that, I would have to go to more effort to download and validate the Libreboot GPG key.

Now that it has been validated, I can use tar to extract it and cd to the libreboot_bin directory:

```
$ tar xvf libreboot_bin.tar.xz
$ cd libreboot_bin
```

Pull Down Software Dependencies

There are a number of different libraries and software that this binary release needs on your system to work. Inside the libreboot_bin directory, you will see a deps-trisquel and deps-parabola script to be run as root. If you use a Debian-based distribution, run `deps-trisquel`, and if you use an Arch Linux-based distribution, run `deps-parabola`. For other distributions, unfortunately, you will need to use those scripts as a guide for what sorts of libraries and packages you will need to download. In my case, I was running from a Debian-based distribution (inside Tails in fact), so I ran:

```
$ sudo ./deps-trisquel
```

Once you have the package dependencies, you need to build flashrom and bucts on your system. Libreboot has provided two scripts to automate this process as well, called `builddeps-bucts` and `builddeps-flashrom`, so the next step is to run those:

```
$ sudo ./builddeps-flashrom
$ sudo ./builddeps-bucts
```

This should create a `./flashrom/flashrom` and a `./bucts/bucts` binary that subsequent scripts will use.

Choose Your ROM

Once you have all of the software downloaded or compiled, the next step is to identify which ROM you want to use. To ease the process and help ensure that you don't brick your laptop, you can choose from a number of pre-compiled BIOS ROMs that Libreboot provides. Under the `./bin/` directory are a few different directories named after the different laptops Libreboot currently supports:

```
$ ls bin/
macbook21  t60  x60  x60t
```

In my case, I'm flashing an x60, so I want to choose a ROM from

that directory:

```
$ ls bin/x60
libreboot_frazerty_txtmode.rom
libreboot_frazerty_vesafb.rom
libreboot_itqwerty_txtmode.rom
libreboot_itqwerty_vesafb.rom
libreboot_svenska_txtmode.rom
libreboot_svenska_vesafb.rom
libreboot_ukdvorak_txtmode.rom
libreboot_ukdvorak_vesafb.rom
libreboot_ukqwerty_txtmode.rom
libreboot_ukqwerty_vesafb.rom
libreboot_usdvorak_txtmode.rom
libreboot_usdvorak_vesafb.rom
libreboot_usqwerty_txtmode.rom
libreboot_usqwerty_vesafb.rom
```

As you can see, there are a number of different ROMs for different languages and keyboard layouts, and within each category, there also are txtmode and vesafb options depending on whether you want your BIOS to display a graphical GRUB screen in VESA mode or just rely on text mode. In my case, I selected bin/x60/libreboot_usqwerty_vesafb.rom.

Back up the Old BIOS

You still are not yet at the point where you risk bricking anything, but you are close, so it's time to back up the old BIOS, so you have a chance of recovering this laptop in case

something goes wrong. When I first tried to flash an x60 with coreboot, the main challenge was due to the fact that the laptop series had two different potential BIOS chipsets, and each required a special patch to flashrom. This meant physically inspecting the motherboard with a magnifying glass and reading the tiny print on the BIOS chip. The Libreboot project has greatly simplified this by creating both flashing tools ahead of time and realizing that one will work, and the other will fail safely.

So to back up your BIOS, cd to the flashrom directory and run two different commands:

```
$ cd flashrom
$ sudo ./flashrom_lenovobios_sst -p internal -r factory.bin
flashrom v0.9.7-unknown on Linux 3.16.0-4-586 (i686)
flashrom is free software, get the source code at
↳http://www.flashrom.org

Calibrating delay loop... OK.

Found chipset "Intel ICH7M". Enabling flash write... WARNING:
↳SPI Configuration Lockdown activated.

OK.

No EEPROM/flash device found.

Note: flashrom can never write if the flash chip isn't
↳found automatically.

$ sudo ./flashrom_lenovobios_macronix -p internal -r factory.bin
flashrom v0.9.7-unknown on Linux 3.16.0-4-586 (i686)
flashrom is free software, get the source code at
```

Now, this command is going to output some incredibly frightening error messages.

→<http://www.flashrom.org>

Calibrating delay loop... OK.

Found chipset "Intel ICH7M". Enabling flash write...

→WARNING: SPI Configuration Lockdown activated.

OK.

Found Macronix flash chip "MX25L1605D/MX25L1608D/MX25L1673E"

→(2048 kB, SPI) mapped at physical address 0xffe00000.

Reading flash... done.

In this case, it turns out I had a Macronix BIOS chip, so the first script failed and the second script worked. The important thing is that at the end, you should have a `factory.bin` file in this directory. **Back this file up!** Because often the BIOS image has customizations that apply to that particular laptop, and because I have a number of different BIOS images I need to back up, I like to label my BIOSes based on the serial number, such as `x60-BIOS-LV-A4332.bin` (not a real serial number).

Perform the First Libreboot Flash

Warning: if you run any of the commands after this point in the column incorrectly, you risk bricking

your laptop! If you aren't willing to take that risk, do not proceed! If you decide to proceed, read each example carefully and check all of your commands for correctness before you press Enter.

The BIOS flashing process occurs in two stages. The first stage is easily reversible (if you use a provided Libreboot ROM at least) and flips a particular setting in your BIOS and changes part but not all of the BIOS firmware. In the root directory where you unpacked the Libreboot tarball, you will see two scripts: `lenovobios_firstflash` and `lenovobios_secondflash`. Run the `lenovobios_firstflash` command as root and pass it the path to the Libreboot ROM you identified earlier.

Now, this command is going to output some incredibly frightening error messages. This is because it's using a general-purpose flashrom tool that in this first phase cannot completely reflash your BIOS. Instead, it is going to set `BUC.TS=1` (a flag that will let you completely rewrite the BIOS after a complete shutdown) as well as set up

a basic BIOS bootloader, but otherwise will fail, as it doesn't yet have the ability to rewrite all of the flash:

```
$ sudo ./lenovobios_firstflash bin/x60/libreboot_usqwerty-vesafb.rom
Don't panic. See docs/index.html for an explanation of what BUC.TS is.
MAKE SURE THAT YOU SEE 'Updated BUC.TS=1' IF NOT CHECK #libreboot
  ↳ON FREENODE
bucts utility version '4'
Using LPC bridge 8086:27b9 at 0000:1f.00
Current BUC.TS=0 - 128kb address range 0xFFFFE0000-0xFFFFFFFF is
  ↳untranslated
Updated BUC.TS=1 - 64kb address ranges at 0xFFFFE0000 and 0xFFFFF0000
  ↳are swapped
READ THE BIG WARNING ABOVE!
MAKE SURE THAT YOU SEE 'DO NOT SHUT DOWN OR REBOOT' (YOU WANT TO
  ↳SEE THAT. MEANS IT WORKED). IF NOT CHECK #libreboot
  ↳ON FREENODE
If (when) you see 'DO NOT SHUTDOWN OR REBOOT' do not panic.
  ↳That is normal, expected and very good. And you will
  ↳ignore what it says.
flashrom v0.9.7-unknown on Linux 3.16.0-4-586 (i686)
flashrom is free software, get the source code at
  ↳http://www.flashrom.org
Calibrating delay loop... OK.
Found chipset "Intel ICH7M". Enabling flash write... WARNING:
  ↳SPI Configuration Lockdown activated.
OK.
No EEPROM/flash device found.
Note: flashrom can never write if the flash chip isn't
  ↳found automatically.
flashrom v0.9.7-unknown on Linux 3.16.0-4-586 (i686)
flashrom is free software, get the source code at
  ↳http://www.flashrom.org
```

```
Calibrating delay loop... OK.
Found chipset "Intel ICH7M". Enabling flash write... WARNING:
  ↳SPI Configuration Lockdown activated.
OK.
Found Macronix flash chip "MX25L1605D/MX25L1608D/MX25L1673E"
  ↳(2048 kB, SPI) mapped at physical address 0xffe00000.
Reading old flash chip contents... done.
Erasing and writing flash chip... spi_block_erase_20 failed
  ↳during command execution at address 0x0
Reading current flash chip contents... done. Looking for another
  ↳erase function.
Transaction error!
spi_block_erase_d8 failed during command execution at address
  ↳0x1f0000
Reading current flash chip contents... done. Looking for another
  ↳erase function.
spi_chip_erase_60 failed during command execution
Reading current flash chip contents... done. Looking for another
  ↳erase function.
spi_chip_erase_c7 failed during command execution
Looking for another erase function.
No usable erase functions left.
FAILED!
Uh oh. Erase/write failed. Checking if anything has changed.
Reading current flash chip contents... done.
Apparently at least some data has changed.
Your flash chip is in an unknown state.
Get help on IRC at chat.freenode.net (channel #flashrom) or
mail flashrom@flashrom.org with the subject "FAILED:
  ↳<your board name>"!
-----
DO NOT REBOOT OR POWEROFF!
READ THE BIG WARNING ABOVE!
Now you will SHUT DOWN (ignore the flashrom warning) but
```

➔first keep in mind before you then boot:

Use 'Search for GRUB configuration on local storage' if

➔the normal menus don't work, or check docs/index.html

➔or #libreboot on freenode.

SHUT DOWN NOW!!!! WAIT A FEW SECS!!!! THEN BOOT.

DON'T PANIC.

With all of this output, there are a few specific things that you want to see. The first is this:

Current BUC.TS=0 - 128kb address range 0xFFFE0000-0xFFFFFFFF

➔is untranslated

Updated BUC.TS=1 - 64kb address ranges at 0xFFFE0000 and

➔0xFFFF0000 are swapped

If you don't see Updated BUC.TS=1, don't reboot, but instead, attempt to run the command again. The second kind of output you want to look for is something like this:

Reading old flash chip contents... done.

Erasing and writing flash chip... spi_block_erase_20 failed

➔during command execution at address 0x0

Reading current flash chip contents... done. Looking for

➔another erase function.

Transaction error!

spi_block_erase_d8 failed during command execution at

➔address 0x1f0000

Reading current flash chip contents... done. Looking for

➔another erase function.

spi_chip_erase_60 failed during command execution

Reading current flash chip contents... done. Looking for

➔another erase function.

spi_chip_erase_c7 failed during command execution

Looking for another erase function.

No usable erase functions left.

FAILED!

Uh oh. Erase/write failed. Checking if anything has changed.

Reading current flash chip contents... done.

Apparently at least some data has changed.

Your flash chip is in an unknown state.

Yes, that seems like a scary error, but it's apparently the kind of scary error that you want to see. What's happening is that flashrom was able to write part of the flash chip but not all of it, so it's erroring. If you see some sort of radically different scary error from the above, don't reboot or shut down your machine. Instead, use the flashrom tool to re-install your original BIOS.

Otherwise, if you see similar output to mine, completely shut down your machine, wait a few seconds, and then boot up again. You should see the Libreboot boot screen with a GRUB menu presenting a few options. You can attempt to use the normal menu options to boot from the local hard drive, or if that fails, select Search for GRUB configuration on local storage.

If First Flash Fails

If after the first flash you don't see anything when you power on, the simplest explanation may be that your laptop backlight reset, so use the

Fn-Home key combination to increase the brightness. Otherwise, if you see no boot screen, but the laptop itself doesn't make any sounds, you still can revert to the old BIOS. Just remove the keyboard and disconnect the CMOS battery for five to ten seconds, then plug it back in. You should be able to boot back in to your original BIOS. Otherwise, if you hear three beeps when you power it on, the laptop unfortunately has been bricked, and you will have to resort to a hardware flash to restore it.

Perform the Second Libreboot Flash

Once you boot back in to your system on the new Libreboot BIOS, it's time to perform the second flash. This flash will permanently replace the original BIOS with Libreboot. Go back to your Libreboot binary directory, and run the `lenovobios_secondflash` utility as root with the same ROM you chose before as an argument:

```
$ sudo ./lenovobios_secondflash bin/x60/
↳libreboot_usqwerty-vesafb.rom

Don't panic. See docs/index.html for an explanation
↳of what BUC.TS is.

MAKE SURE THAT YOU SEE 'VERIFIED' AT THE END (YOU WANT TO SEE
↳THAT. MEANS IT WORKED).

flashrom v0.9.7-unknown on Linux 3.16.0-4-586 (i686)

flashrom is free software, get the source code at
↳http://www.flashrom.org
```

```
Calibrating delay loop... OK.

coreboot table found at 0x7f6bd000.

Found chipset "Intel ICH7M". Enabling flash write... OK.

Found Macronix flash chip "MX25L1605D/MX25L1608D/MX25L1673E"
↳(2048 kB, SPI) mapped at physical address 0xffe00000.

Reading old flash chip contents... done.

Erasing and writing flash chip... Erase/write done.

Verifying flash... VERIFIED.

READ THE BIG WARNING ABOVE!

MAKE SURE THAT YOU SEE 'Updated BUC.TS=0' IF NOT CHECK
↳#libreboot ON FREENODE

bucts utility version '4'

Using LPC bridge 8086:27b9 at 0000:1f.00

Current BUC.TS=1 - 64kb address ranges at 0xFFFE0000 and
↳0xFFFF0000 are swapped

Updated BUC.TS=0 - 128kb address range 0xFFFE0000-0xFFFFFFFF
↳is untranslated

Not writing BUC register since TS is already correct.

READ THE BIG WARNING ABOVE!

If the above 2 conditions are met, then shut down now. If not,
↳then run: sudo ./bucts/bucts 1

DON'T PANIC.
```

I don't know, there's something about seeing the words "don't panic" in all caps that makes you want to panic. Okay, as you can see in this output, there shouldn't be any scary errors. Instead, I was able to read the old flash contents and erase and write the new one:

```
Reading old flash chip contents... done.

Erasing and writing flash chip... Erase/write done.

Verifying flash... VERIFIED.
```




SHAWN POWERS

Pipes and STDs

Standard input, output and error are confusing—until now.

Punny title aside, the concepts of STDIN (standard input), STDOUT (standard output) and STDERR (standard error) can be very confusing, especially to folks new to Linux. Once you understand how data gets into and out of applications, however, Linux allows you to string commands together in awesome and powerful ways. In this article, I want to clear up how things work, so you can make the command line work much more efficiently.

Processes and Their Data

At a basic level, when a process is run on the command line, it has three “data ports” where it can send and/or receive data. Figure 1 shows my depiction of an application’s I/O design.

Here are some definitions:

- **STDIN:** this is where an application receives input. If you run a program that asks you to enter your age, it receives that info via its STDIN mechanism, using the keyboard as

the input device.

- **STDOUT:** this is where the results come out of the program. If you type `ls`, the file listings are sent to STDOUT, which by default displays on the screen.
- **STDERR:** if something goes wrong, this is the error message. It can be a little confusing, because like STDOUT, STDERR is displayed on the screen by default as well. If you type `ls mycooldoc`, but there’s no such file as “mycooldoc”, you’ll get an error message on the screen. Even though it appears on the screen in the same place STDOUT appears, it’s important to understand that it came out of a different place. That’s important, because STDOUT and STDERR can be directed separately and to different places.

It’s also important to realize that I/O is different from command-line

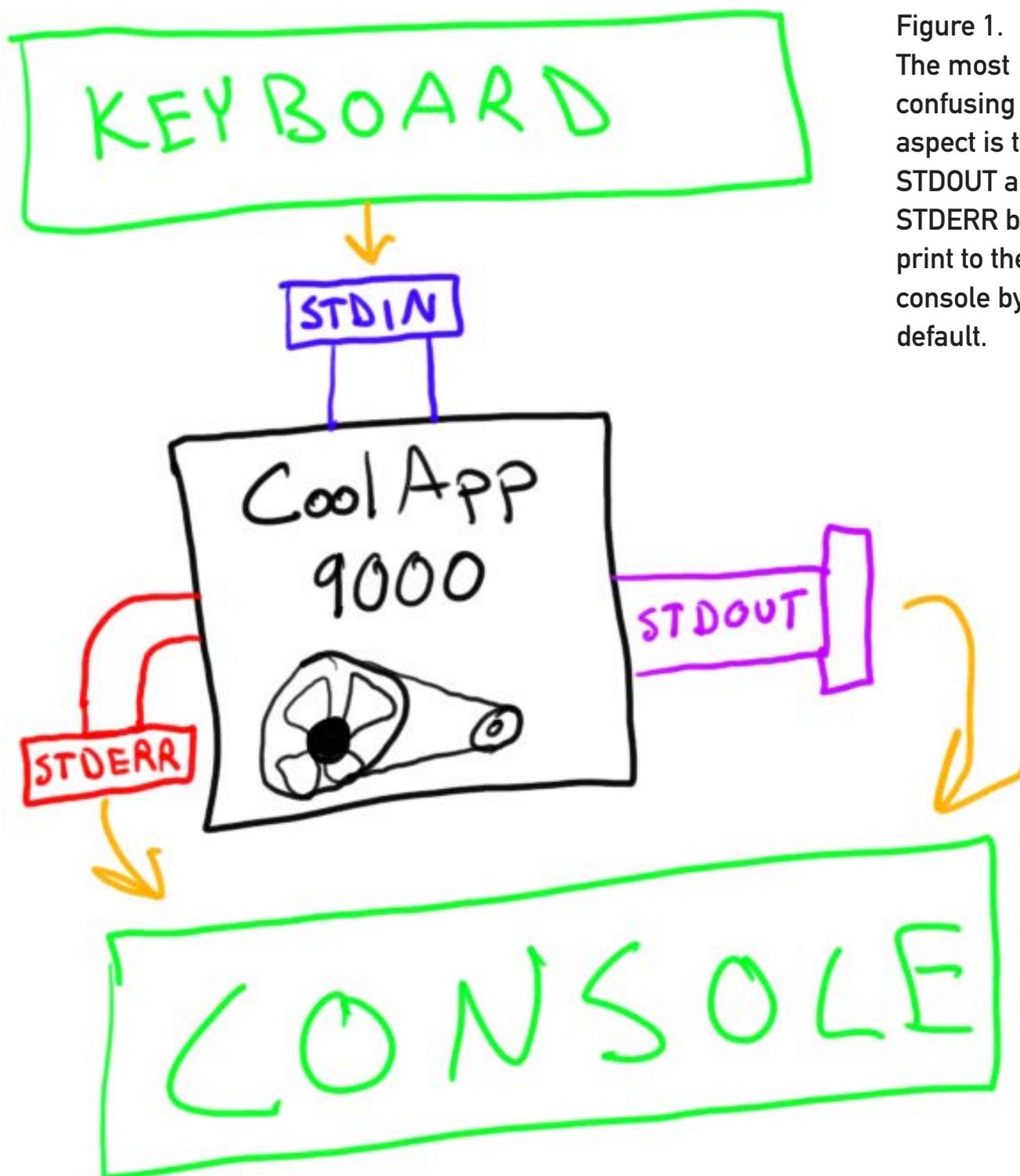


Figure 1. The most confusing aspect is that STDOUT and STDERR both print to the console by default.

arguments or flags. Input, for example, is data the process gets from some external source. When you run a command with arguments, those

arguments just tell the process how to run. Typing `ls -l`, for instance, just tells the `ls` program how to execute. The `STDIN/OUT/ERR` are used only

once the program is running as a way to send or receive data.

STDIN Example

By default, STDIN is read from the keyboard. So, this little script prompts for input via the keyboard. When you enter the information and press enter, it's fed into the application's STDIN. Then that information is processed, and the result is dumped out of STDOUT, which by default is displayed on the command line:

```
#!/bin/bash
echo "What is your favorite number?"
read favnum
echo "My favorite number is $favnum too!"
```

If you look closely, the initial "What's your favorite number?" text is also sent out STDOUT, and since it defaults to the screen, you see it as a prompt before the script uses the read command to get data into STDIN.

Redirecting STDOUT and STDERR

You've seen that STDOUT and STDERR both default to displaying on the screen. It's often more desirable to have one or both get saved to a file instead of displayed. To redirect the output, use the "greater-than" symbol. For example, typing:

```
ls > results.txt
```

will save the directory listing to a file called results.txt instead of displaying it on the screen. That example, however, redirects only the STDOUT, not the STDERR. So if something goes wrong, the error message displays on the screen instead of getting saved to a file. So in this example:

```
ls filename > results.txt
```

if there is not file called "filename", you'll see an error on the screen even though you redirected STDOUT into a file. You'll see something like:

```
ls filename > results.txt
ls: cannot access filename: No such
file or directory
```

There is a way to redirect the STDERR, which is similar to redirecting STDOUT, and without first understanding the difference between the two output "ports", it can be confusing. But to redirect STDERR instead of STDOUT, you'd type this:

```
ls 2> errors.txt
```

Which, when typed, simply would print the file listing on the screen. Using the 2> structure, you are only redirecting errors to the file. So in this case:

```
ls filename 2> errors.txt
```

As you can imagine, redirecting output is very useful when running scripts or programs that are executed in the background; otherwise, you'd never see the output!

if there isn't a file named "filename", the error message would get saved to the file `errors.txt`, and nothing would display on the screen. It's possible to do both at once too. So you could type:

```
ls > results.txt 2> errors.txt
```

and you'd see the file listing in `results.txt`, while any error messages would go into `errors.txt`. You've probably seen something similar in `crontab`, where the desire is to have both `STDOUT` and `STDERR` go into a file. Usually, the desire is to have them both get redirected into the same file, so you'll see something like this:

```
ls > stuff.txt 2>&1
```

That looks really confusing, but it's not as bad as it seems. The first part should make sense. Redirecting `STDOUT` into a file called `stuff.txt` is clear. The second part, however, is just redirecting `STDERR` into `STDOUT`.

The reason you can't just type `2>1` is because Bash would interpret that as "I want to save the `STDERR` into a file named `1`", so the ampersand preceding the `1` tells Bash you want to redirect the `STDERR` into `STDOUT`.

One last trick regarding the redirection of `STDOUT` and `STDIN` is the double greater-than symbol. When you redirect output into a file using the `>` symbol, it overwrites whatever is in the file. So if you have an `errors.txt` file, it will overwrite what's already in there and just show the single error. With a `>>` symbol, it will append the results instead of overwriting. This is really useful if you're trying to make a log file. For example, typing:

```
ls >> files.txt  
ls -l >> files.txt
```

will create a file called "files.txt" that has a list of the files, then a long directory listing of the same files. Thankfully, if the file doesn't exist, using a double greater-than symbol will create

the file just like a single greater-than symbol will do. As you can imagine, redirecting output is very useful when running scripts or programs that are executed in the background; otherwise, you'd never see the output!

Redirecting STDIN

This concept is a little bit harder to understand, but once you "get" the whole concept of I/O, it's not too bad. It's important to know that not all applications listen on their STDIN port, so for some programs, redirecting STDIN does nothing. One common command that does listen on STDIN, however, is `grep`. If you type:

```
grep chicken menu.txt
```

the `grep` command will search through the `menu.txt` file for any lines containing the string "chicken", and print those lines on the screen (via STDOUT, which should make sense now). `grep` also will accept input via STDIN instead of via filename, however, so you could do this:

```
cat menu.txt | grep chicken
```

and the exact same results will be shown. If that seems confusing, just walk through the process with me. When you type `cat menu.txt`, the

`cat` program displays the contents of `menu.txt` to the screen, via STDOUT. If you used a `>` symbol, you could redirect that STDOUT into a new file, but if you use the pipe symbol (`|`), you can redirect the STDOUT data into another program's STDIN. That's what's happening in this example. It's as if the `cat` program's purple STDOUT tube in Figure 1 is bolted directly onto `grep`'s blue STDIN tube. Since `grep` is listening on its STDIN port for data, it then executes its search for the word `chicken` on that data that is coming into STDIN rather than reading from a file.

This example above might seem frivolous, and honestly it is. Where redirecting with a pipe symbol comes in handy is when you string together multiple actions. So this, for example:

```
grep chicken menu.txt | grep pasta
```

will return a list of all of the lines in `menu.txt` that have the word "chicken" in them *and* have the word "pasta" in them. You could do the same thing by typing this:

```
grep chicken menu.txt > chickenlist.txt  
grep pasta chickenlist.txt
```

But, that takes two lines, and then you have a fairly useless file on your system called `chickenlist.txt`, when all

Once you get used to piping STDOUT from one command into STDIN for another, you'll find yourself becoming a grep ninja in no time.

you wanted was a list of items that contain both chicken and pasta.

Once you get used to piping STDOUT from one command into STDIN for another, you'll find yourself becoming a grep ninja in no time. Granted, there are many other applications that listen on STDIN for information, but grep is one that is very commonly used. For example:

```
ls -l /etc | grep apache
```

is a way to look for any files or directories in the /etc folder that contain the string "apache" in their name. Or:

```
cat /var/log/syslog | grep USB
```

is a great way to look for any log entries in the syslog that mention USB devices. You even could go further and type:

```
cat /var/log/syslog | grep USB > usbresults.txt
```

and you'd have a text file containing any lines in /var/log/syslog that mention USB. Perhaps you're troubleshooting

an issue, and you need to send those lines to a tech support person.

Redirecting STDOUT and STDERR into a file, or piping them into another process' STDIN, is an important concept to understand. It's important to know the difference between what a >, >> and | do so that you get the results you want. Sometimes redirecting STDOUT, STDERR and STDIN aren't enough, however, because not all applications listen for data on STDIN. That's where xargs comes into play.

xargs: Making Apps Play Nice

Sometimes you want to use the STDOUT from one command to interact with an application that doesn't support getting data piped into STDIN. In this case, you can use the simple and powerful xargs command. Here's a scenario: your hard drive is filling up, so you want to delete all the .mp3 files in all the folders in the entire system. You can get a list of all of those files by typing:

```
find / -name "*.mp3"
```




A P A C H E C O N

NORTH AMERICA

April 13 - 16, 2015

Austin, Texas

ApacheCon hosts development and collaboration on some of today's hottest open source projects, including Apache projects like Cassandra, CloudStack, Cordova, CouchDB, Geronimo, Hadoop, Hive, HTTP Server, Lucene, OpenOffice, Struts, Subversion and Tomcat, among many others.

Attendees come to ApacheCon to learn about the latest developments across Apache projects and to collaborate with the people advancing the work that is defining the future of technology and that represents a new generation of software development.

Co-located events for ApacheCon North America include: Apache Traffic Server Summit, CloudStack Days and an Apache Ignite Training Session.

Register Today
go.linuxfoundation.org/apachecon-na2015

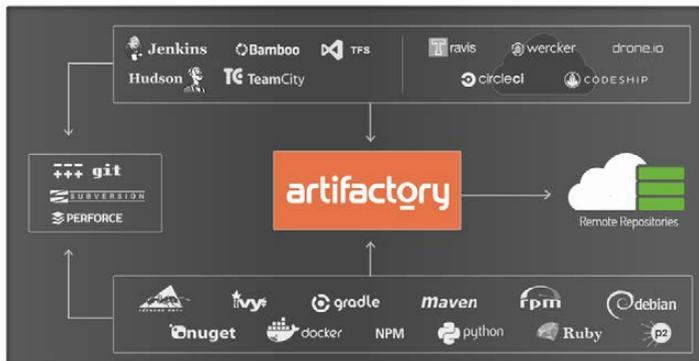
Useful Network Video Wall

Digital-display software provider Useful boasts that its new Network Video Wall, which is able to support up to 25 displays from a single Core i7 PC, will be the new benchmark for flexibility, affordability and simplicity in the sector.

Useful also calls this the first solution with the capability to run multiple video walls throughout a building from a single PC or server. The

Userful Network Video Wall can be arranged in artistic, angled designs or in a standard grid configuration. Typical installations include lobbies, museums, restaurants, stadiums, transportation hubs, retail stores, college campuses, control rooms, meeting rooms and briefing and broadcast centers.

<http://userful.com>

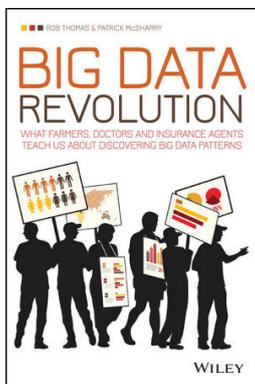


JFrog's Artifactory Binary Repository Management Solution

Because today's software development and distribution is a wholly continuous process that demands speed and agility, the need to make flexible, precise and

lightning-fast queries across the entire development environment continues to grow. To enable this high-performance search capability, JFrog announced a significantly enhanced Artifactory Binary Repository Management Solution version 3.5, which features the company's new Artifactory Query Language (AQL). AQL, proffers JFrog, is the industry's first and only search tool to help developers, DevOps and QA teams locate binaries based on any set of criteria, independent of repository type or packaging format. Examples of AQL's "exceedingly specific searches" include things like find all Docker images marked as deployed in production, get the latest Web application binaries produced by a certain build branch, retrieve all artifacts that have been downloaded more than 1,000 times but have a newer improved version and so on.

<http://www.jfrog.com>



Rob Thomas and Patrick McSharry's *Big Data Revolution* (Wiley)

The new Wiley book *Big Data Revolution* is a guide to improving performance, making better decisions and transforming business through the effective use of Big Data. In this collaborative work by Rob Thomas, IBM Vice President of Big Data Products, and Patrick McSharry, an Oxford Research Fellow, this book presents inside stories from varied industries

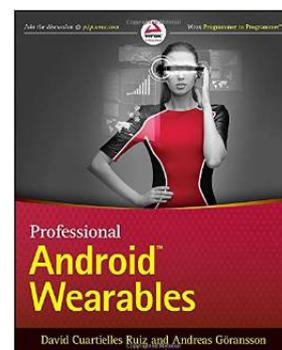
that demonstrate the power and potential of Big Data within the business realm. As implied by its subtitle, *What farmers, doctors and insurance agents teach us about discovering big data patterns*, the book focuses on how to uncover patterns and insights. Readers are guided through tried-and-true methodologies for getting more out of data and using it to the utmost advantage. This book describes the major trends emerging in the field, the pitfalls and triumphs being experienced, and the many considerations surrounding Big Data, all while guiding readers toward better decision making from the perspective of a data scientist. Companies are generating data faster than ever before, and managing that data has become a major challenge. With the right strategy, Big Data can be a powerful tool for creating effective business solutions, but deep understanding is key when applying it to individual business needs.

<http://www.wiley.com>

David Cuartielles Ruiz and Andreas Goransson's *Professional Android Wearables* (Wrox)

The the next transformative wave of smart mobile devices will be wearables. To help Android developers surf into the Pope's living room without getting axed with wearables is the new Wrox book *Professional Android Wearables* by David Cuartielles Ruiz and Andreas Goransson. The veteran developers demonstrate how to use the Android Wear platform and other techniques to build real-world apps for a variety of wearables including smart bands, smart watches and smart glasses. In no time, readers will grasp how wearables can connect them to the Internet in more pervasive ways than with PCs, tablets or mobile devices; how to build code using Google's Wear SDK for Android-enabled hardware devices; how Android Wear and other Android development techniques are capable of building several presented example projects; and much more.

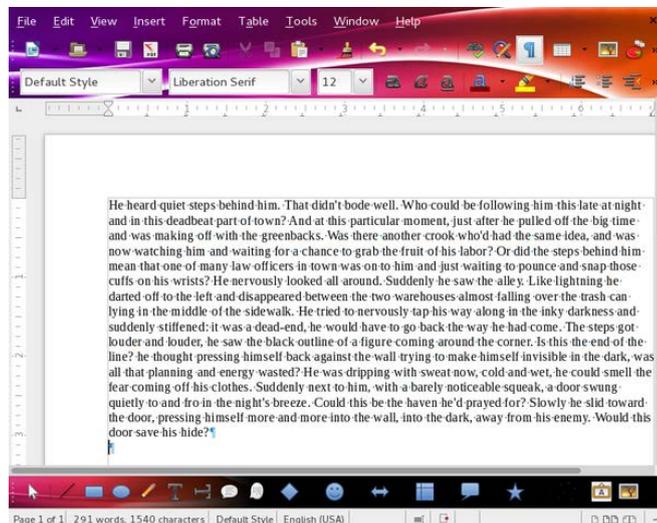
<http://www.wrox.com>



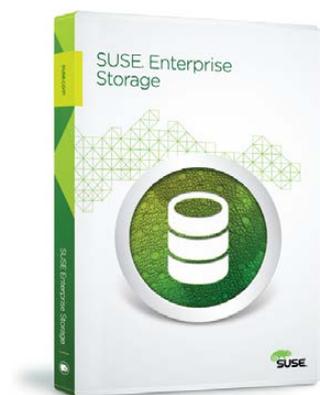
The Document Foundation's LibreOffice

The Document Foundation is proud of the ninth major iteration of LibreOffice, describing the new 4.4 release of its office suite as possessing “a lot of UX and design love” and representing its “most beautiful” version to date. Most notably, the user interface has been improved significantly, and interoperability with OOXML file formats has been extended. A sampling of other notable improvements includes support of OpenGL transitions in Windows and improved implementation of the new OpenGL, digital signing of PDF files during the export process, installation of myriad free fonts, several new default templates, visual editing of Impress master pages, better Track Changes function in Writer, improved import filters and greatly expanded support for media capabilities on each platform.

<http://libreoffice.org>



SUSE Enterprise Storage



The move from expensive, proprietary systems to more affordable, open-source solutions is a well-worn path in our disruptive sector of the IT world. The disruption team at SUSE widens that path nicely with SUSE Enterprise Storage, a self-managing, self-healing, distributed software-based storage solution for enterprise customers. Powered by the Ceph open-source distributed storage solution, SUSE Enterprise Storage leverages commodity, off-the-shelf servers and disk drives to build highly scalable storage at a drastically reduced cost per unit. Based on the Firefly version of Ceph, the fully featured SUSE Enterprise Storage is well suited for object, archival and bulk storage, with features including cache tiering, thin provisioning, copy-on-write cloning and erasure coding. SUSE's solution is available as an option with SUSE OpenStack Cloud or as a standalone storage solution.

<http://www.suse.com/storage>

Zato Source s.r.o.'s Zato

Zato Source self-describes its open-source ESB (Enterprise Service Bus) and application server Zato 2.0 as written “by pragmatists for pragmatists”. Written in Python to guarantee usability and productivity and “not yet another system quickly stitched

```

21
22 # stdlib
23 from contextlib import closing
24 from traceback import format_exc
25
26 # Zato
27 from zato.common.broker_message import MESSAGE_TYPE, CHANNEL
28 from zato.common.odb.model import ChannelZMQ, Cluster, Service
29 from zato.common.odb.query import channel_zmq_list
30 from zato.server.connection.zmq_channel import start_connector
31 from zato.server.service.internal import AdminService, AdminSIO
32
33 class GetList(AdminService):
34     """ Returns a list of ZeroMQ channels.
35
36     class SimpleIO(AdminSIO):
37         request_elem = 'zato_channel_zmq_get_list_request'
38         response_elem = 'zato_channel_zmq_get_list_response'
39         input_required = ('cluster_id',)
40         output_required = ('id', 'name', 'is_active', 'address', 'socket_type', 'sub_key', 'service_name')
41
42     def get_data(self, session):
43         return channel_zmq_list(session, self.request.input.cluster_id, False)
44
45     def handle(self):
46         with closing(self.odb.session()) as session:
47             self.response.payload[:] = self.get_data(session)

```

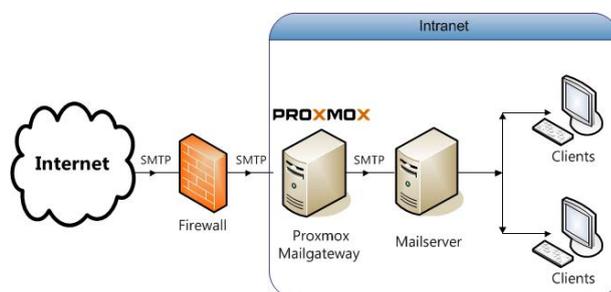
together by a vendor on the wave of ESB/SOA hype”, Zato can be used for building middleware and back-end systems. Zato facilitates intercommunication across applications and data sources spanning an organization’s business or technical boundaries and beyond, enabling users to access, design, develop or discover new opportunities and processes. The new Zato 2.0 adds dozens of new features, a few of which include a Dockerfile to install a fully operational cluster of two servers, load-balancer and Web-admin in ten minutes; a wealth of new connection types; Redis-based REST publish/subscribe; and new security mechanisms. Commercial support and training for Zato are available.

<http://zato.io>

Proxmox Server Solutions GmbH's Proxmox Mail Gateway

One reason the Proxmox Mail Gateway from Proxmox Server Solutions GmbH has experienced great success over its ten-year history is the application of two—and an optional three—antivirus weapons. These engines include ClamAV, Cyren’s Zero-Hour Virus Outbreak Protection and the optional Avira. Proxmox Mail Gateway 4.0 is the new version of Proxmox’s e-mail security system that, either on bare-metal or as a virtual appliance, protects e-mail servers from spam, viruses, trojans and phishing e-mails and is managed through an intuitive, Web-based interface. Version 4.0 features a complete package update and is now based on Debian Wheezy 7.8.

www.proxmox.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.

LUCI4HPC

**LUCI4HPC is a lightweight,
user-friendly high-performance
computer cluster installation
and management tool.**

**It offers a graphical
Web-based control panel
and is fully customizable.**

Melanie Grandits, Axel Sündermann and Chris Oostenbrink

Today's computational needs in diverse fields cannot be met by a single computer. Such areas include weather forecasting, astronomy, aerodynamics simulations for cars, material sciences and computational drug design. This makes it necessary to combine multiple computers into one system, a so-called computer cluster, to obtain the required computational power.

The software described in this article is designed for a Beowulf-style cluster. Such a cluster commonly

between them, such as InfiniBand.

This rather complex setup requires special software, which offers tools to install and manage such a system easily. The software presented in this article—LUCI4HPC, an acronym for lightweight user-friendly cluster installer for high performance computing—is such a tool.

The aim is to facilitate the maintenance of small in-house clusters, mainly used by research institutions, in order to lower the dependency on shared external

THE AIM IS TO FACILITATE THE MAINTENANCE OF SMALL IN-HOUSE CLUSTERS, MAINLY USED BY RESEARCH INSTITUTIONS, IN ORDER TO LOWER THE DEPENDENCY ON SHARED EXTERNAL SYSTEMS.

consists of consumer-grade machines and allows for parallel high-performance computing. The system is managed by a head node and accessed via a login node. The actual work is performed by multiple compute nodes. The individual nodes are connected through an internal network. The head and login node need an additional external network connection, while the compute nodes often use an additional high-throughput, low-latency connection

systems. The main focus of LUCI4HPC is to be lightweight in terms of resource usage to leave as much of the computational power as possible for the actual calculations and to be user-friendly, which is achieved by a graphical Web-based control panel for the management of the system.

LUCI4HPC focuses only on essential features in order not to burden the user with many unnecessary options so that the system can be made operational

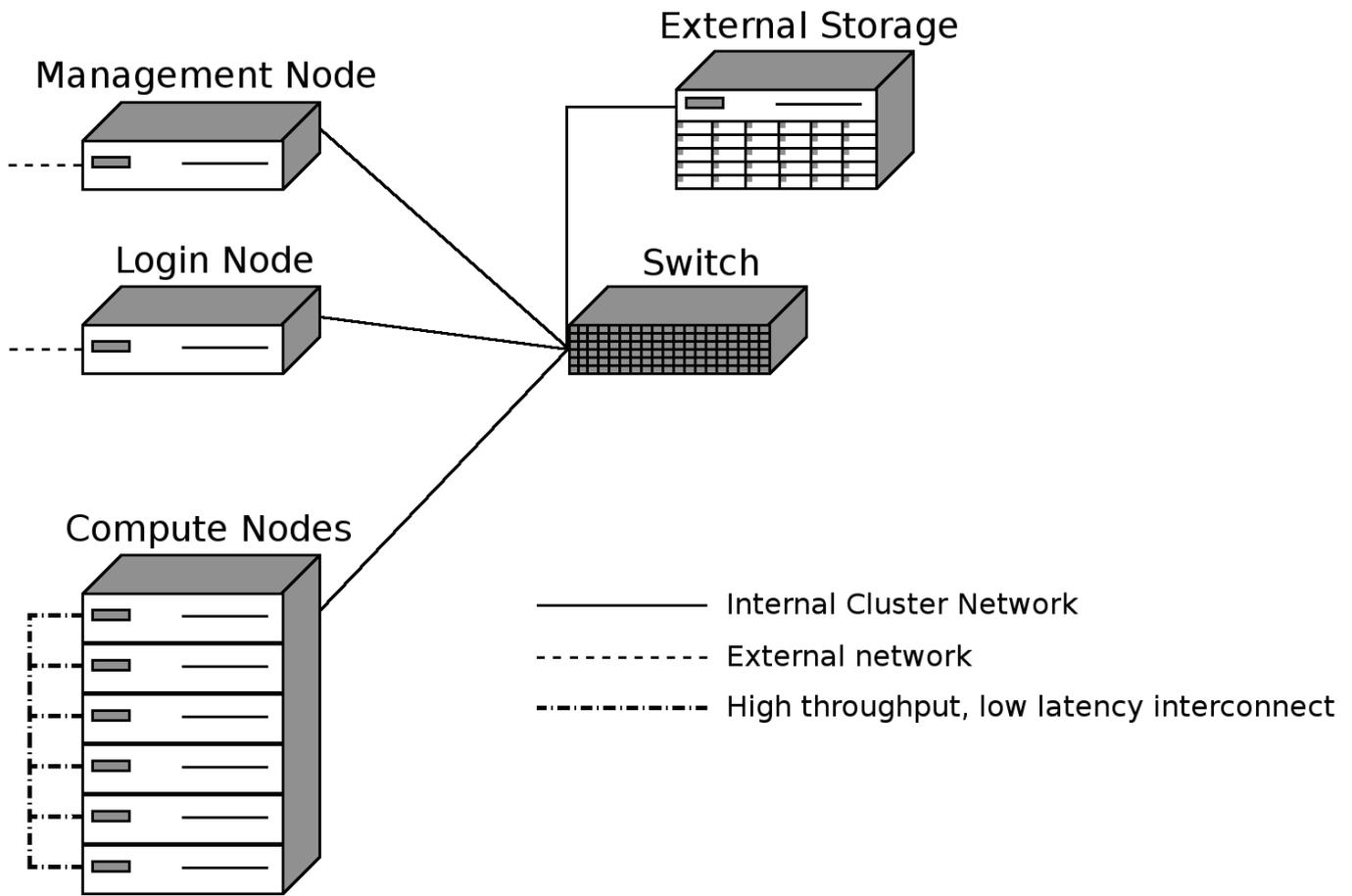


Figure 1. Recommended Hardware Setup for a Cluster Running LUCI4HPC

quickly with just a few clicks.

In this article, we provide an overview of the LUCI4HPC software as well as briefly explain the installation and use. You can find a more detailed installation and usage guide in the manual on the LUCI4HPC Web site (see Resources). Figure 1 shows an overview of the recommended hardware setup.

The current beta version of LUCI4HPC comes in a self-extracting binary package and supports Ubuntu Linux. Execute the binary

on the head node, with an already installed operating system, to trigger the installation process. During the installation process, you have to answer a series of questions concerning the setup and configuration of the cluster. These questions include the external and internal IP addresses of the head node, including the IP range for the internal network, the name of the cluster as well as the desired time zone and keyboard layout for the installation of the other nodes.

The installation script offers predefined default values extracted from the operating system for most of these configuration options. The install script performs all necessary steps in order to have a fully functional head node. After the installation, you need to acquire a free-of-charge license on the LUCI4HPC Web site and place it in the license folder. After that, the cluster is ready, and you can add login and compute nodes.

on the node.

Currently, the software distinguishes three types of nodes: namely login, compute and other. A login node is a computer with an internal and an external connection, and it allows the users to access the cluster. This is separated from the head node in order to prevent user errors from interfering with the cluster system. Because scripts that use up all the memory or processing time may affect the LUCI4HPC programs, a compute

THE CANDIDATE SYSTEM HAS THE ADVANTAGE THAT MANY NODES CAN BE TURNED ON AT THE SAME TIME AND THAT YOU CAN LATER DECIDE FROM THE COMFORT OF YOUR OFFICE ON THE TYPE OF EACH NODE.

It is very easy to add a node. Connect the node to the internal network of the cluster and set it to boot over this network connection. All subsequent steps can be performed via the Web-based control panel. The node is recognized as a candidate and is visible in the control panel. There you can define the type (login, compute, other) and name of the node. Click on the Save button to start the automatic installation of Ubuntu Linux and the client program

node performs the actual calculation and is therefore composed out of potent hardware. The type "other" is a special case, which designates a node with an assigned internal IP address but where the LUCI4HPC software does not automatically install an operating system. This is useful when you want to connect, for example, a storage server to the cluster, where an internal connection is preferred for performance reasons, but that already has an operating

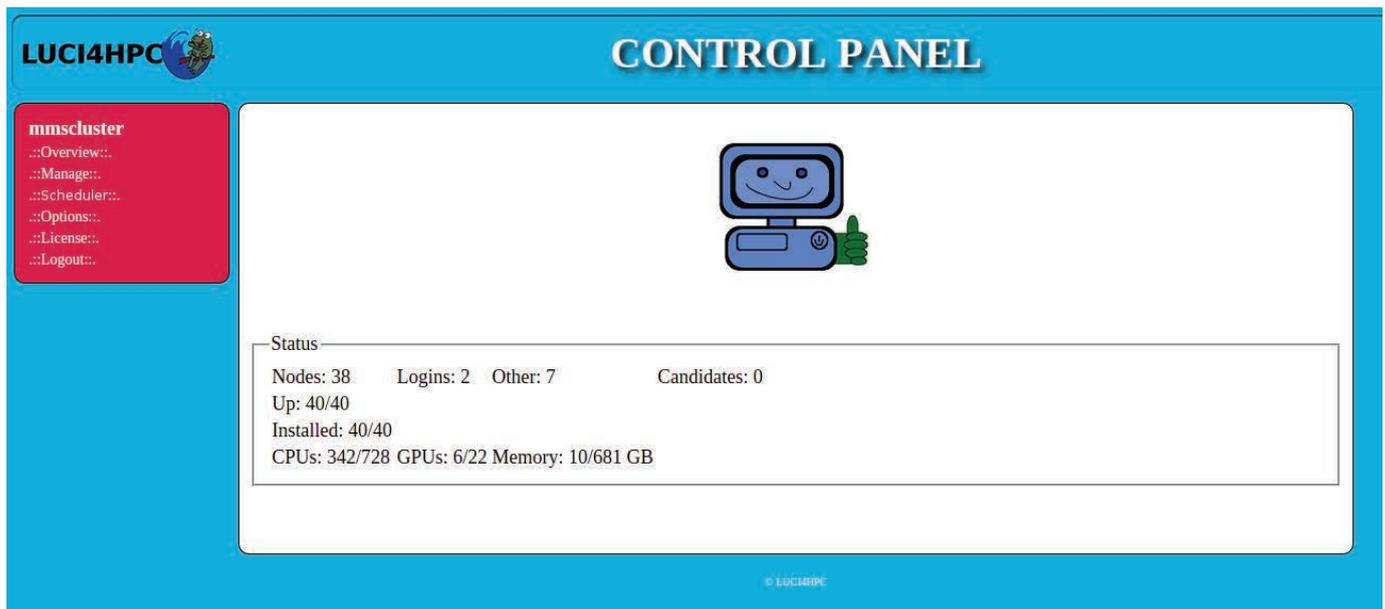


Figure 2. LUCI4HPC Web-Based Control Panel, Cluster Overview Page

system installed. The candidate system has the advantage that many nodes can be turned on at the same time and that you can later decide from the comfort of your office on the type of each node.

An important part of a cluster software is the scheduler, which manages the assignment of the resources and the execution of the job on the various nodes. LUCI4HPC comes with a fully integrated job scheduler, which also is configurable via the Web-based control panel.

The control panel uses HTTPS, and you can log in with the user name and password of the user that has the user ID 1000. It is, therefore, very easy and convenient to change the login credentials—just change the

credentials of that user on the head node. After login, you'll see a cluster overview on the first page. Figure 2 shows a screenshot of this overview.

This overview features the friendly computer icon called Clusterboy, which shows a thumbs up if everything is working properly and a thumbs down if there is a problem within the cluster, such as a failed node. This allows you to assess the status of the cluster immediately. Furthermore, the overview shows how many nodes of each type are in the cluster, how many of them are operational and installed, as well as the total and currently used amount of CPUs, GPUs and memory. The information on the currently used amount of resources is directly taken

from the scheduler.

The navigation menu on the right-hand side of the control panel is used to access the different pages. The management page shows a list of all nodes with their corresponding MAC and IP addresses as well as the hostname separated into categories depending on their type. The top category shows the nodes that are marked as down, which means that they have not sent a heartbeat in the last two minutes. Click on the “details” link next to a node to access the configuration page. The uptime and the load as well as the used and total amount of resources are listed there. Additionally, some configuration options can be changed, such as the hostname, the IP address and the type of the node, and it also can be marked for re-installation. Changing the IP address requires a reboot of the node in order to take effect, which is not done automatically.

The scheduler page displays a list of all current jobs in the cluster, as well as whether they are running or queuing. Here you have the option of deleting jobs.

The queue tab allows you to define new queues. Nodes can be added to a queue very easily. Click on the “details” link next to a queue to get a

list of nodes assigned to it as well as a list of currently unassigned nodes. Unassigned nodes can be assigned to a queue, and nodes assigned to a queue can be removed from it to become an unassigned node. Additionally, a queue can have a fair use limit; it can be restricted to a specific group ID, and you can choose between three different scheduling methods. These methods are “fill”, which fills up the nodes one after another; “spread”, which assigns a new job to the least-used node and thus performs a simple load balancing; and finally, “full”, which assigns a job to an empty node. This method is used when several jobs cannot coexist on the same node.

There also is a VIP system. This system gives temporary priority access to a user when, for example, a deadline has to be met. VIP users always are on the top of the queue, and their job is executed as soon as the necessary resources become available. Normally, the scheduler assigns a weight to each job based on the amount of requested resources and the submission time. This weight determines the queuing order.

Finally, the options page allows you to change configuration options of the cluster system, determined during the installation. In general, everything

that can be done in the control panel also can be done by modifying the configuration scripts and issuing a reload command.

With the current beta version, a few tasks cannot be done with the control panel. These include adding new users and packages as well as customizing the installation scripts. In order to add a user to the cluster, add the user to the head node as you normally would add a user under

added to the `additional_packages` file in the LUCI4HPC configuration folder. During the startup or installation process, or after a reload command, the nodes install all packages listed in this file automatically.

The installation process of LUCI4HPC is handled with a preseed file for the Ubuntu installer as well as pre- and post-installation shell scripts. These shell scripts, as well as the preseed file, are customizable. They

BECAUSE OF THE POSSIBILITY TO CHANGE THE INSTALLATION SHELL SCRIPTS AND TO USE CONFIGURATION OPTIONS DIRECTLY FROM THE CLUSTER SYSTEM IN THESE SCRIPTS, YOU CAN VERY EASILY ADAPT THE INSTALLATION TO YOUR SPECIFIC NEEDS.

Linux. Issue a reload command to the nodes via the LUCI4HPC command-line tool, and then the nodes will synchronize the user and group files from the head node. Thus, the user becomes known to the entire cluster.

Installing new packages on the nodes is equally easy. As the current version supports Ubuntu Linux, it also supports the Ubuntu package management system. In order to install a package on all nodes as well as all future nodes, a package name is

support so-called LUCI4HPC variables defined by a `#`. The variables allow the scripts to access the cluster options, such as the IP of the head node or the IP and hostname of the node where the script is executed. Therefore, it is possible to write a generic script that uses the IP address of the node it runs on through these variables without defining it for each node separately.

There are special installation scripts for GPU and InfiniBand drivers that are executed only when the

appropriate hardware is found on the node. The installation procedures for these hardware components should be placed in these files.

Because of the possibility to change the installation shell scripts and to use configuration options directly from the cluster system in these scripts, you can very easily adapt the installation to your specific needs. This can be used, for example, for the automated installation of drivers for specific hardware or the automatic setup of specific software packages needed for your work.

For the users, most of this is hidden. As a user, you log in to the login node and use the programs `lqsub` to submit a job to the cluster, `lqdel` to remove one of your jobs and `lqstat` to view your current jobs and their status.

The following gives a more technical overview of how LUCI4HPC works in the background.

LUCI4HPC consists of a main program, which runs on the head node, as well as client programs, one for each node type, which run on the nodes. The main program starts multiple processes that represent the LUCI4HPC services. These services communicate via shared memory. Some services can use multiple threads in order to increase their throughput. The services are

responsible for managing the cluster, and they provide basic network functionality, such as DHCP and DNS. All parts of LUCI4HPC were written from scratch in C/C++. The only third-party library used is OpenSSL. Besides a DNS and a DHCP service, there also is a TFTP service that is required for the PXE boot process.

A heartbeat service is used to monitor the nodes and check whether they are up or down as well as to gather information, such as the current load. The previously described scheduler also is realized through a service, which means that it can access the information directly from other services, such as the heartbeat in the shared memory. This prevents it from sending jobs to nodes that are down. Additionally, other services, such as the control panel, can access information easily on the current jobs.

A package cache is available, which minimizes the use of the external network connection. If a package is requested by one node, it is downloaded from the Ubuntu repository and placed in the cache such that subsequent requests from other nodes can download the package directly from it. The synchronization of the user files is handled by a

Put JavaScript, HTML5, CSS, and the latest web tools to work.

Fluent is for everyone who has a hand in web development, from front-end to back-end and everything in between. Get practical training on the latest in HTML5, CSS3, JavaScript, and the frameworks that build on those technologies.

- Interface and experience design
- HTML5 and CSS3
- Pure Code and JavaScript
- Application architectures
- Graphics and visualization
- Development tools
- Cross-platforming

fluentconf.com

[@fluentconf](https://twitter.com/fluentconf)

Save 20% on your ticket
Use code LINUXJ



O'REILLY®

Fluent

CONFERENCE

THE WEB PLATFORM

APRIL 20-22, 2015 • SAN FRANCISCO, CA

Jailhouse

**A new approach to real-time
security-wise virtualization in Linux.**

Valentine Sinitsyn

Because you're a reader of *Linux Journal*, you probably already know that Linux has a rich virtualization ecosystem. KVM is the de facto standard, and VirtualBox is widely used for desktop virtualization. Veterans should remember Xen (it's still in a good shape, by the way), and there is also VMware (which isn't free but runs on Linux as well). Plus, there are many lesser-known hypervisors like the educational lguest or hobbyist Xvisor. In such a crowded landscape, is there a place for a newcomer?

There likely is not much sense in creating yet another Linux-based "versatile" hypervisor (other than doing it just for fun, you know). But, there are some specific use cases that general-purpose solutions just don't address quite well. One such area is real-time virtualization, which is frequently used in industrial automation, medicine, telecommunications and high-performance computing. In these applications, dedicating a whole CPU or its core to the software that runs bare metal (with no underlying OS) is a way to meet strict deadline requirements. Although it is possible to pin a KVM instance to the processor core and pass through PCI devices to guests, tests show the worst-case latency may be above some

realistic requirements (see Resources).

As usual with free software, the situation is getting better with time, but there is one other thing—security. Sensitive software systems go through rigorous certifications (like Common Criteria) or even formal verification procedures. If you want them to run virtualized (say, for consolidation purposes), the hypervisor must isolate them from non-certifiable workloads. This implies that the hypervisor itself must be small enough; otherwise, it may end up being larger (and more "suspicious") than the software it segregates, thus devastating the whole idea of isolation.

So, it looks like there is some room for a lightweight (for the real-time camp), small and simple (for security folks) open-source Linux-friendly hypervisor for real-time and certifiable workloads. That's where Jailhouse comes into play.

New Guy on the Block

Jailhouse was born at Siemens and has been developed as a free software project (GPLv2) since November 2013. Last August, Jailhouse 0.1 was released to the general public. Jailhouse is rather young and more of a research project than a ready-to-use tool at this point, but now is a good time to become acquainted it and be

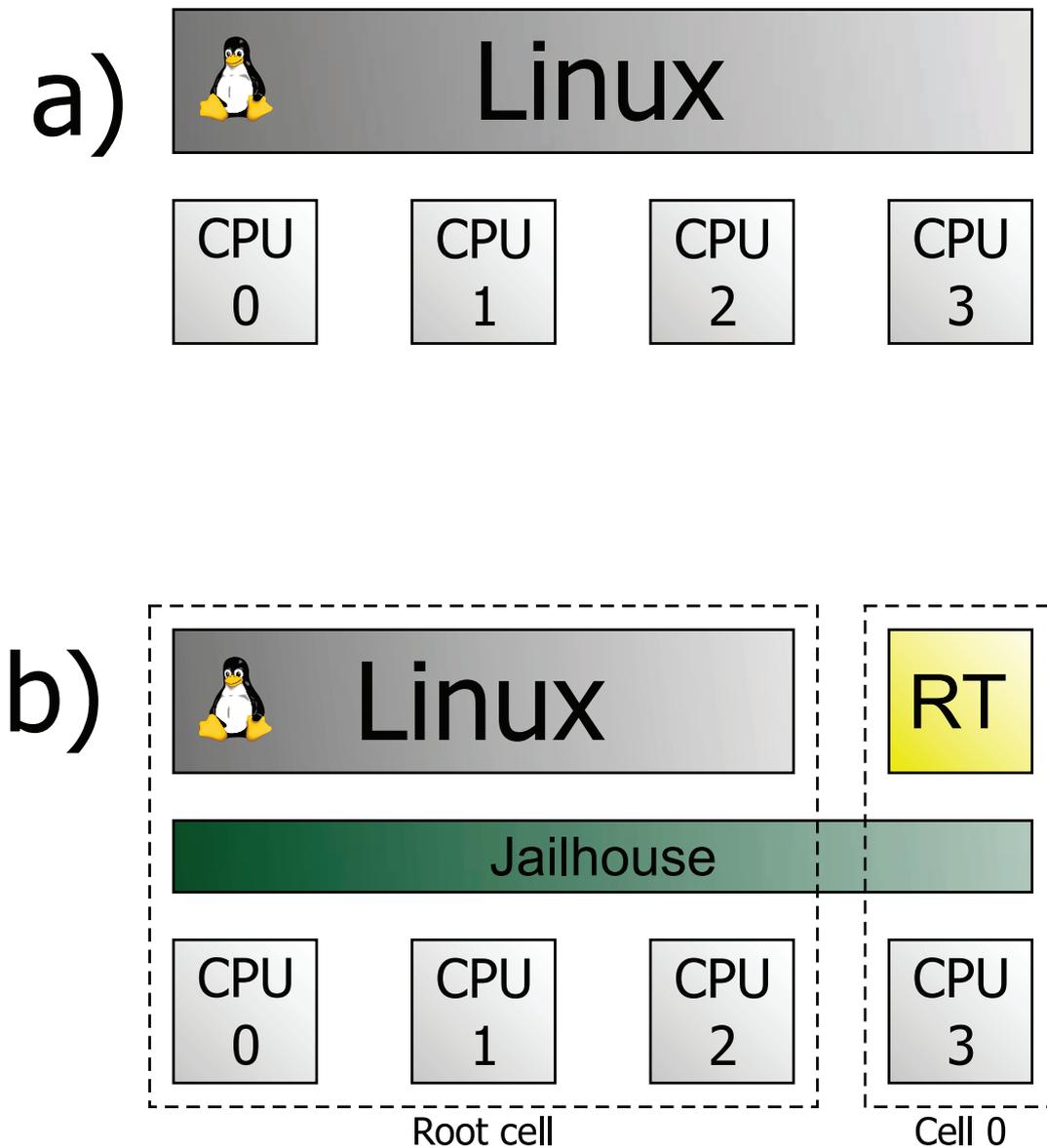


Figure 1. A visualization of Linux running-bare metal (a) and under the Jailhouse hypervisor (b) alongside a real-time application. (Image from Yulia Sinitsyna; Tux image from Larry Ewing.)

prepared to meet it in production.

From the technical point of view, Jailhouse is a static partitioning hypervisor that runs bare metal but cooperates closely with Linux. This means Jailhouse doesn't emulate resources you don't have. It just splits your hardware into isolated compartments called "cells" that are

wholly dedicated to guest software programs called "inmates". One of these cells runs the Linux OS and is known as the "root cell". Other cells borrow CPUs and devices from the root cell as they are created (Figure 1).

Besides Linux, Jailhouse supports bare-metal applications, but it can't run general-purpose OSes (like

Windows or FreeBSD) unmodified. As mentioned, there are plenty of other options if you need that. One day Jailhouse also may support running KVM in the root cell, thus delivering the best of both worlds.

As mentioned previously, Jailhouse cooperates closely with Linux and relies on it for hardware

bootstrapping, hypervisor launch and doing management tasks (like creating new cells). Bootstrapping is really essential here, as it is a rather complex task for modern computers, and implementing it within Jailhouse would make it much more complex. That being said, Jailhouse doesn't meld with the kernel as KVM (which is

Getting Up to Date

Sometimes you may need the very latest KVM and QEMU to give Jailhouse a try. KVM is part of the kernel, and updating the critical system component just to try some new software probably seems like overkill. Luckily, there is another way.

`kvm-kmod` is a tool to take KVM modules from one kernel and compile them for another, and it usually is used to build the latest KVM for your current kernel. The build process is detailed in the README, but in a nutshell, you clone the repository, initialize a submodule (it's the source for KVM), and run the configure script followed by `make`. When the modules are ready, just `insmod` them instead of what your distribution provides (don't forget to unload those first). If you want the change to be permanent, run `make modules_install`. `kvm-kmod` can take the KVM sources from wherever you point to, but the defaults are usually sufficient.

Compiling QEMU is easier but more time consuming. It follows the usual `configure && make` procedure, and it doesn't need to be installed system-wide (which is package manager-friendly). Just put `/path/to/qemu/x86_64-softmmu/qemu-system-x86_64` instead of plain `qemu-system-x86_64` in the text's examples.

a kernel module) does. It is loaded as a firmware image (the same way Wi-Fi adapters load their firmware blobs) and resides in a dedicated memory region that you should reserve at Linux boot time. Jailhouse's kernel module (`jailhouse.ko`, also called "driver") loads the firmware and creates `/dev/jailhouse` device, which the Jailhouse userspace tool uses, but it doesn't contain any hypervisor logic.

Jailhouse is an example of Asynchronous Multiprocessing (AMP) architecture. Compared to traditional Symmetric Multiprocessing (SMP) systems, CPU cores in Jailhouse are not treated equally. Cores 0 and 1 may run Linux and have access to a SATA hard drive, while core 2 runs a bare-metal application that has access only to a serial port. As most computers Jailhouse can run on have shared L2/L3 caches, this means there is a possibility for cache thrashing. To understand why this happens, consider that Jailhouse maps the same guest physical memory address (GPA) to a different host (or real) physical address for different inmates. If two inmates occasionally have the same GPA (naturally containing diverse data) in the same L2/L3 cache line due to cache associativity, they will interfere with each other's work and degrade the performance. This effect

is yet to be measured, and Jailhouse currently has no dedicated means to mitigate it. However, there is a hope that for many applications, this performance loss won't be crucial.

Now that you have enough background to understand what Jailhouse is (and what it isn't), I hope you are interested in learning more. Let's see how to install and run it on your system.

Building Jailhouse

Despite having a 0.1 release now, Jailhouse still is a young project that is being developed at a quick pace. You are unlikely to find it in your distribution's repositories for the same reasons, so the preferred way to get Jailhouse is to build it from Git.

To run Jailhouse, you'll need a recent multicore VT-x-enabled Intel x86 64-bit CPU and a motherboard with VT-d support. By the time you read this article, 64-bit AMD CPUs and even ARM (v7 or better) could be supported as well. The code is already here (see Resources), but it's not integrated into the mainline yet. At least 1GB of RAM is recommended, and even more is needed for the nested setup I discuss below. On the software side, you'll need the usual developer tools (`make`, `GCC`, `Git`) and headers for your Linux kernel.

Running Jailhouse on real hardware isn't straightforward at this time, so if you just want to play with it, there is a better alternative. Given that you meet CPU requirements, the hypervisor should run well under KVM/QEMU. This is known as a nested setup. Jailhouse relies on some bleeding-edge features, so you'll need at least Linux 3.17 and QEMU 2.1 for everything to work smoothly. Unless you are on a rolling release distribution, this could be a problem, so you may want to compile these tools yourself. See the Getting Up to Date sidebar for more information, and I suggest you have a look at it even if you are lucky enough to have the required versions pre-packaged. Jailhouse evolves and may need yet unreleased features and fixes by the time you read this.

Make sure you have nested mode enabled in KVM. Both `kvm-intel` and `kvm-amd` kernel modules accept the `nested=1` parameter, which is responsible just for that. You can set it manually, on the `modprobe` command line (don't forget to unload the previous module's instance first). Alternatively, add options `kvm-intel nested=1` (or the similar `kvm-amd` line) to a new file under `/etc/modprobe.d`.

You also should reserve memory for

Jailhouse and the inmates. To do this, simply add `memmap=66M$0x3b000000` to the kernel command line. For one-time usage, do this from the GRUB menu (press `e`, edit the command line and then press `F10`). To make the change persistent, edit the `GRUB_CMDLINE_LINUX` variable in `/etc/default/grub` on the QEMU guest side and regenerate the configuration with `grub-mkconfig`.

Now, make a JeOS edition of your favorite distribution. You can produce one with SUSE Studio, `ubuntu-vm-builder` and similar, or just install a minimal system the ordinary way yourself. It is recommended to have the same kernel on the host and inside QEMU. Now, run the virtual machine as (Intel CPU assumed):

```
qemu-system-x86_64 -machine q35 -m 1G -enable-kvm -smp 4
    ↪-cpu kvm64,-kvm_pv_eoi,-kvm_steal_time,-kvm_asyncpf,
    ↪-kvmclock,+vmx,+x2apic -drive
    ↪file=LinuxInstallation.img,id=disk,if=none
    ↪-virtfs local,path=/path/to/jailhouse,
    ↪security_model=passthrough,mount_tag=host
    ↪-device ide-hd,drive=disk -serial stdio
    ↪-serial file:com2.txt
```

Note, I enabled `9p` (`-virtfs`) to access the host filesystem from the QEMU guest side; `/path/to/jailhouse` is where you are going to compile Jailhouse now. `cd` to this directory

and run:

```
git clone git@github.com:siemens/jailhouse.git jailhouse
cd jailhouse
make
```

Now, switch to the guest and mount the 9p filesystem (for example, with `mount -t 9p host /mnt`). Then, `cd` to `/mnt/jailhouse` and execute:

```
sudo make firmware_install
sudo insmod jailhouse.ko
```

This copies the Jailhouse binary image you've built to `/lib/firmware` and inserts the Jailhouse driver

module. Now you can enable Jailhouse with:

```
sudo tools/jailhouse enable configs/qemu-vm.cell
```

As the command returns, type `dmesg | tail`. If you see "The Jailhouse is opening." message, you've successfully launched the hypervisor, and your Linux guest now runs under Jailhouse (which itself runs under KVM/QEMU). If you get an error, it is an indication that your CPU is missing some required feature. If the guest hangs, this is most likely because your host kernel or QEMU are not up to date enough for Jailhouse, or

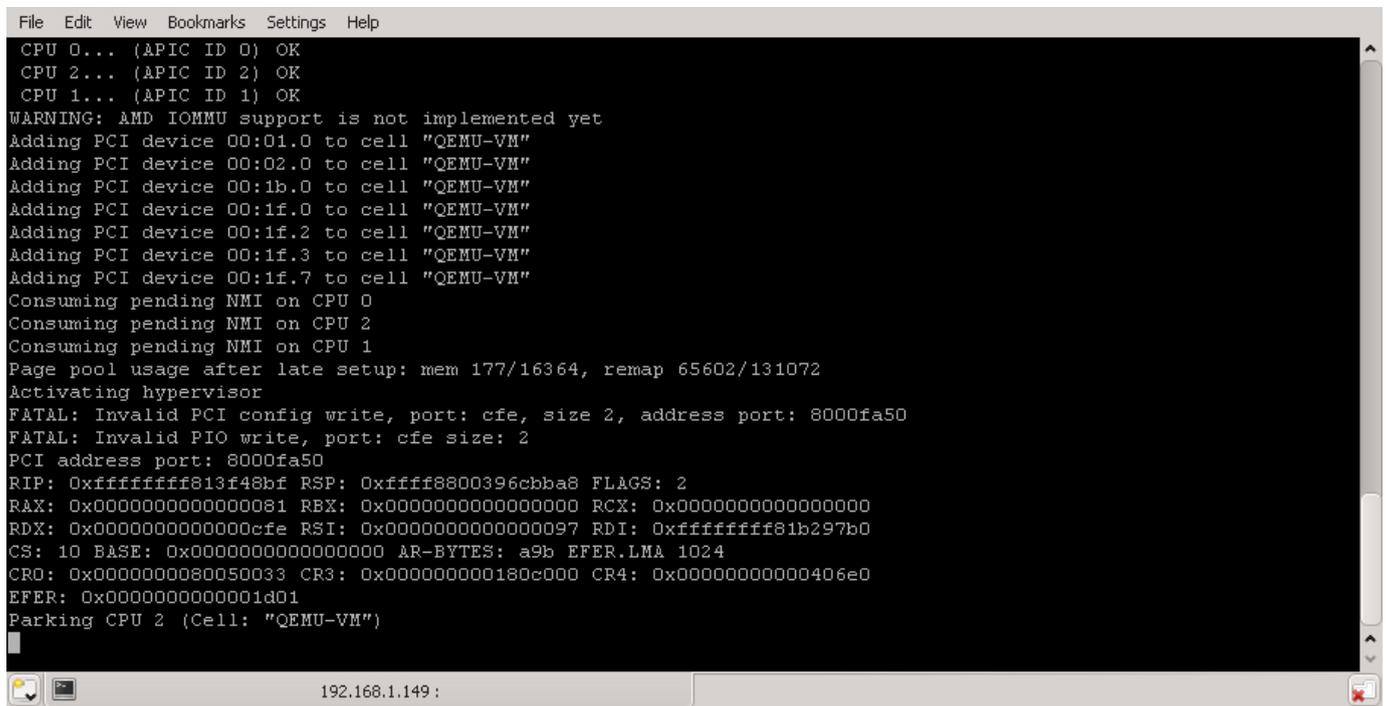


Figure 2. A typical configuration issue: Jailhouse traps "prohibited" operation from the root cell.

something is wrong with `qemu-vm` cell config. Jailhouse sends all its messages to the serial port, and QEMU simply prints them to the terminal where it was started (Figure 2). Look at the messages to see what resource (I/O port, memory and so on) caused the problem, and read on for the details of Jailhouse configuration.

Configs and Inmates

Creating Jailhouse configuration files isn't straightforward. As the code base must be kept small, most of the logic that takes place automatically in other hypervisors must be done manually here (albeit with some help from the tools that come with Jailhouse). Compared to `libvirt` or `VirtualBox` XML, Jailhouse configuration files are very detailed and rather low-level. The configuration currently is expressed in the form of plain C files (found under `configs/` in the sources) compiled into raw binaries; however, another format (like `DeviceTree`) could be used in future versions.

Most of the time, you wouldn't need to create a cell config from scratch, unless you authored a whole new inmate or want the hypervisor to run on your specific hardware (see the Jailhouse for Real sidebar).

Cell configuration files contain

information like hypervisor base address (it should be within the area you reserved with `memmap=` earlier), a mask of CPUs assigned to the cell (for root cells, it's `0xff` or all CPUs in the system), the list of memory regions and the permissions this cell has to them, I/O ports bitmap (0 marks a port as cell-accessible) and the list of PCI devices.

Each Jailhouse cell has its own config file, so you'll have one config for the root cell describing the platform Jailhouse executes on (like `qemu-vm.c`, as you saw above) and several others for each running cell. It's possible for inmates to share one config file (and thus one cell), but then only one of these inmates will be active at a given time.

In order to launch an inmate, you need to create its cell first:

```
sudo tools/jailhouse cell create configs/apic-demo.cell
```

`apic-demo.cell` is the cell configuration file that comes with Jailhouse (I also assume you still use the QEMU setup described earlier). This cell doesn't use any PCI devices, but in more complex cases, it is recommended to unload Linux drivers before moving devices to the cell with this command.

Now, the inmate image can be

Jailhouse treats all inmates as opaque binaries, and although it provides a small framework to develop them faster, the only thing it needs to know about the inmate image is its base address.

loaded into memory:

```
sudo tools/jailhouse cell load apic-demo
↳ inmates/demos/x86/apic-demo.bin -a 0xf0000
```

Jailhouse treats all inmates as opaque binaries, and although it provides a small framework to develop them faster, the only thing it needs to know about the inmate image is its base address. Jailhouse expects an inmate entry point at 0xffff0 (which is different from the x86 reset vector). `apic-demo.bin` is a standard demo inmate that comes with Jailhouse, and the inmate's framework linker script ensures that if the binary is mapped at 0xf0000, the entry point will be at the right address. `apic-demo` is just a name; it can be almost anything you want.

Finally, start the cell with:

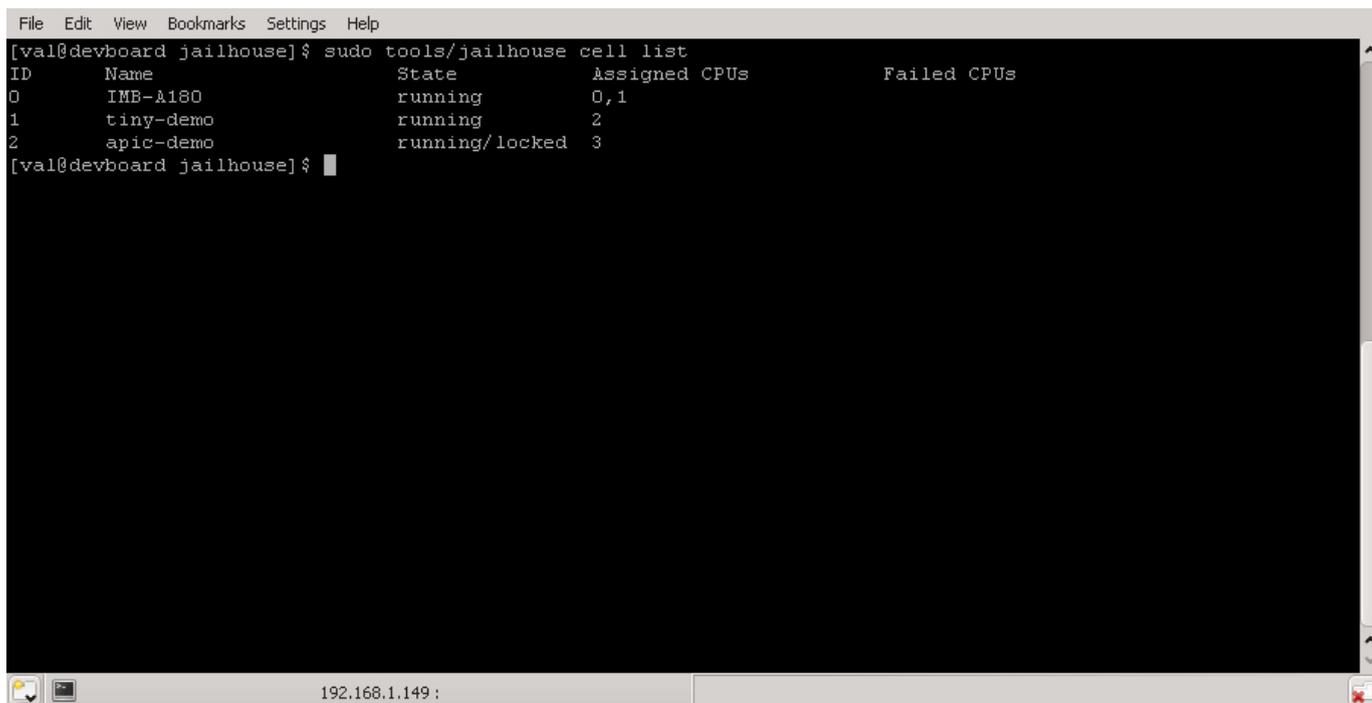
```
sudo tools/jailhouse cell start apic-demo
```

Now, switch back to the terminal from which you run QEMU. You'll see that lines like this are being sent to the serial port:

```
Calibrated APIC frequency: 1000008 kHz
Timer fired, jitter: 38400 ns, min: 38400 ns, max: 38400 ns
...
```

`apic-demo` is purely a demonstrational inmate. It programs the APIC timer (found on each contemporary CPU's core) to fire at 10Hz and measures the actual time between the events happening. Jitter is the difference between the expected and actual time (the latency), and the smaller it is, the less visible (in terms of performance) the hypervisor is. Although this test isn't quite comprehensive, it is important, as Jailhouse targets real-time inmates and needs to be as lightweight as possible.

Jailhouse also provides some

A terminal window with a menu bar (File, Edit, View, Bookmarks, Settings, Help) and a title bar (192.168.1.149). The terminal shows the command `sudo tools/jailhouse cell list` and its output:

```
[val@devboard jailhouse]$ sudo tools/jailhouse cell list
ID      Name           State           Assigned CPUs   Failed CPUs
0       IMB-A180      running        0,1
1       tiny-demo     running        2
2       apic-demo     running/locked 3
[val@devboard jailhouse]$
```

Figure 3. Jailhouse cell listing—the same information is available through the sysfs interface.

means for getting cell statistics. At the most basic level, there is the sysfs interface under `/sys/devices/jailhouse`. Several tools exist that pretty-print this data. For instance, you can list cells currently on the system with:

```
sudo tools/jailhouse cell list
```

The result is shown in Figure 3. “IMB-A180” is the root cell’s name. Other cells also are listed, along with their current states and CPUs assigned. The “Failed CPUs” column contains CPU cores that triggered some fatal error (like accessing

an unavailable port or unassigned memory region) and were stopped.

For more detailed statistics, run:

```
sudo tools/jailhouse cell stat apic-demo
```

You’ll see something akin to Figure 4. The data is updated periodically (as with the `top` utility) and contains various low-level counters like the number of hypercalls issued or I/O port accesses emulated. The lifetime total and per-second values are given for each entry. It’s mainly for developers, but higher numbers mean the inmate causes hypervisor

COUNTER	SUM	PER SEC
vmexits_total	1885	20
vmexits_msr	1880	20
vmexits_management	3	0
vmexits_cr	2	0
vmexits_cpuid	0	0
vmexits_hypercall	0	0
vmexits_mmio	0	0
vmexits_pio	0	0
vmexits_xapic	0	0
vmexits_xsetbv	0	0

Figure 4. Jailhouse cell statistics give an insight into how cells communicate with the hypervisor.

involvement more often, thus degrading the performance. Ideally, these should be close to zero, as jitter in apic-demo. To exit the tool, press Q.

Tearing It Down

Jailhouse comes with several demo inmates, not only apic-demo. Let's try something different. Stop the inmate with:

```
sudo tools/jailhouse cell destroy apic-demo
JAILHOUSE_CELL_DESTROY: Operation not permitted
```

What's the reason for this?

Remember the apic-demo cell had the "running/locked" state in the cell list. Jailhouse introduces a locked state to prevent changes to the configuration. A cell that locks the hypervisor is essentially more important than the root one (think of it as doing some critical job at a power plant while Linux is mostly for management purposes on that system). Luckily, apic-demo is a toy inmate, and it unlocks Jailhouse after the first shutdown attempt, so the second one should succeed. Execute the above command one more time, and apic-demo should

Jailhouse for Real

QEMU is great for giving Jailhouse a try, but it's also possible to test it on real hardware. However, you never should do this on your PC. With a low-level tool like Jailhouse, you easily can hang your root cell where Linux runs, which may result in filesystem and data corruption.

Jailhouse comes with a helper tool to generate cell configs, but usually you still need to tweak the resultant file. The tool depends on Python; if you don't have it on your testing board, Jailhouse lets you collect required data and generate the configuration on your main Linux PC (it's safe):

```
sudo tools/jailhouse config collect data.tar
# Copy data.tar to your PC or notebook and untar
tools/jailhouse config create -r path/to/untarred/data
↳configs/myboard.c
```

The configuration tool reads many files under /proc and /sys (either collected or directly), analyzes them and generates memory regions, a PCI devices list and other things required for Jailhouse to run.

Post-processing the generated config is mostly a trial-and-error process. You enable Jailhouse and try to do something. If the system locks up, you analyze the serial output and decide if you need to grant access. If you are trying to run Jailhouse on a memory-constrained system (less than 1GB of RAM), be careful



FIGURE A. A must-have toolkit to run Jailhouse bare metal: serial-to-USB converter, null modem cable (attached) and mountable COM port. (Image from Yulia Sinitsyna.)

with the hypervisor memory area, as the configuration tool currently can get it wrong. Don't forget to reserve memory for Jailhouse via the kernel command line the same way you did in QEMU. On some AMD-based systems, you may need to adjust the Memory Mapped I/O (MMIO) regions, because Jailhouse doesn't support AMD IOMMU technology yet, although the configuration tool implies it does.

To capture Jailhouse serial output, you'll likely need a serial-to-USB adapter and null modem cable. Many modern motherboards come with no COM ports, but they have headers you can connect a socket to (the cabling is shown in Figure a). Once you connect your board to the main Linux PC, run minicom or similar to see the output (remember to set the port's baud rate to 115200 in the program's settings).

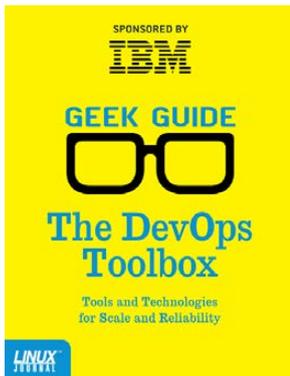
NEW!

Linux Journal eBook Series

GEEK GUIDES

FREE
Download
NOW!

The DevOps Toolbox: Tools and Technologies for Scale and Reliability



By Bill Childers

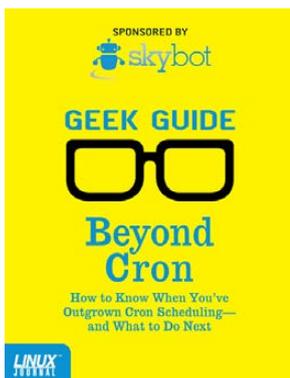
Introducing *The DevOps Toolbox: Tools and Technologies for Scale and Reliability* by Linux Journal Virtual Editor Bill Childers.

When I was growing up, my father always said, “Work smarter, not harder.” Now that I’m an adult, I’ve found that to be a core concept in my career as a DevOps engineer and manager. In order to work smarter, you’ve got to have good tools and technology in your corner doing a lot of the repetitive work, so you and your team can handle any exceptions that occur. More important, your tools need to have the ability to evolve and grow over time according to the changing needs of your business and organization.

In this eBook, I discuss a few of the most important tools in the DevOps toolbox, the benefits of using them and some examples of each tool. It’s important to not consider this a review of each tool, but rather a guide to foster thinking about what’s appropriate for your own organization’s needs.

Register today to receive your complimentary copy of The DevOps Toolbox:
<http://linuxjournal.com/devops-toolbox-guide>

Beyond Cron How to Know When You’ve Outgrown Cron Scheduling— and What to Do Next



By Mike Diehl

If you’ve spent any time around Unix, you’ve no doubt learned to use and appreciate cron, the ubiquitous Unix job scheduler. Cron is simple and easy to use, and most importantly, it just works. It sure beats having to remember to run your backups by hand, for example.

But cron has its limits. Today’s enterprises are larger, more interdependent, and more interconnected than ever before, and cron just hasn’t kept up. These days, we have virtual servers that spring into existence on demand. We’ve got accounting jobs that have to run after billing jobs have completed, but before the backups run. And we’ve got enterprises that connect web servers, databases, and file servers. These enterprises may be in one server room, or they may span several data centers.

This GeekGuide will help you figure out when you’ve outgrown Cron, and offers solutions for what’s next; Beyond Cron.

Register today to receive your complimentary copy of Beyond Cron:
<http://linuxjournal.com/beyond-cron-guide>

<http://linuxjournal.com/geekguides>

WEBCASTS



Learn the 5 Critical Success Factors to Accelerate IT Service Delivery in a Cloud-Enabled Data Center

Today's organizations face an unparalleled rate of change. Cloud-enabled data centers are increasingly seen as a way to accelerate IT service delivery and increase utilization of resources while reducing operating expenses. Building a cloud starts with virtualizing your IT environment, but an end-to-end cloud orchestration solution is key to optimizing the cloud to drive real productivity gains.

> <http://lnxjr.nl/IBM5factors>



Modernizing SAP Environments with Minimum Risk—a Path to Big Data

Sponsor: **SAP** | Topic: **Big Data**

Is the data explosion in today's world a liability or a competitive advantage for your business? Exploiting massive amounts of data to make sound business decisions is a business imperative for success and a high priority for many firms. With rapid advances in x86 processing power and storage, enterprise application and database workloads are increasingly being moved from UNIX to Linux as part of IT modernization efforts. Modernizing application environments has numerous TCO and ROI benefits but the transformation needs to be managed carefully and performed with minimal downtime. Join this webinar to hear from top IDC analyst, Richard Villars, about the path you can start taking now to enable your organization to get the benefits of turning data into actionable insights with exciting x86 technology.

> <http://lnxjr.nl/modsap>

WHITE PAPERS



White Paper: JBoss Enterprise Application Platform for OpenShift Enterprise

Sponsor: **DLT Solutions**

Red Hat's® JBoss Enterprise Application Platform for OpenShift Enterprise offering provides IT organizations with a simple and straightforward way to deploy and manage Java applications. This optional OpenShift Enterprise component further extends the developer and manageability benefits inherent in JBoss Enterprise Application Platform for on-premise cloud environments.

Unlike other multi-product offerings, this is not a bundling of two separate products. JBoss Enterprise Middleware has been hosted on the OpenShift public offering for more than 18 months. And many capabilities and features of JBoss Enterprise Application Platform 6 and JBoss Developer Studio 5 (which is also included in this offering) are based upon that experience.

This real-world understanding of how application servers operate and function in cloud environments is now available in this single on-premise offering, JBoss Enterprise Application Platform for OpenShift Enterprise, for enterprises looking for cloud benefits within their own datacenters.

> <http://lnxjr.nl/jbossapp>

WHITE PAPERS



Linux Management with Red Hat Satellite: Measuring Business Impact and ROI

Sponsor: **Red Hat** | Topic: **Linux Management**

Linux has become a key foundation for supporting today's rapidly growing IT environments. Linux is being used to deploy business applications and databases, trading on its reputation as a low-cost operating environment. For many IT organizations, Linux is a mainstay for deploying Web servers and has evolved from handling basic file, print, and utility workloads to running mission-critical applications and databases, physically, virtually, and in the cloud. As Linux grows in importance in terms of value to the business, managing Linux environments to high standards of service quality — availability, security, and performance — becomes an essential requirement for business success.

> <http://lnxjr.nl/RHS-ROI>



Standardized Operating Environments for IT Efficiency

Sponsor: **Red Hat**

The Red Hat® Standard Operating Environment SOE helps you define, deploy, and maintain Red Hat Enterprise Linux® and third-party applications as an SOE. The SOE is fully aligned with your requirements as an effective and managed process, and fully integrated with your IT environment and processes.

Benefits of an SOE:

SOE is a specification for a tested, standard selection of computer hardware, software, and their configuration for use on computers within an organization. The modular nature of the Red Hat SOE lets you select the most appropriate solutions to address your business' IT needs.

SOE leads to:

- Dramatically reduced deployment time.
- Software deployed and configured in a standardized manner.
- Simplified maintenance due to standardization.
- Increased stability and reduced support and management costs.
- There are many benefits to having an SOE within larger environments, such as:
 - Less total cost of ownership (TCO) for the IT environment.
 - More effective support.
 - Faster deployment times.
 - Standardization.

> <http://lnxjr.nl/RH-SOE>



DOC SEARLS

Consent That Goes Both Ways

Until now we've been consenting to what Web sites and apps want, but soon it'll also be the other way around.

Whatever your opinions about Do Not Track, set them aside for a minute and just look at what the words say and who says them. Individuals—the people we call “users” (you know, like with drugs)—are the ones saying it. In grammatical terms, “do not track”

don't get it.

It's easy to lay the blame on lack of agreement about what Do Not Track does, or should do, and how. But the real problem is deeper: in the power asymmetry of client-server, which we might also call calf-cow (Figure 1).

In client-server, we're calves and

Having Do Not Track in the world has done nothing to change the power asymmetry of client-server.

is spoken in the first person. In legal terms, it's spoken by the first party. The site is the second person and the second party. The unwanted tracking is mostly by a third person *them*: third parties the first one doesn't want following him or her around. In both the grammatical and the legal senses, individuals want *consent* to the Do Not Track request. And mostly they

sites are cows. We go to sites to suckle “content” and get lots of little unwanted files, most of which are meant to train advertising crosshairs on us. Having Do Not Track in the world has done nothing to change the power asymmetry of client-server. But it's not the only tool, nor is it finished. In fact, the client-side revolution in this space has barely started.

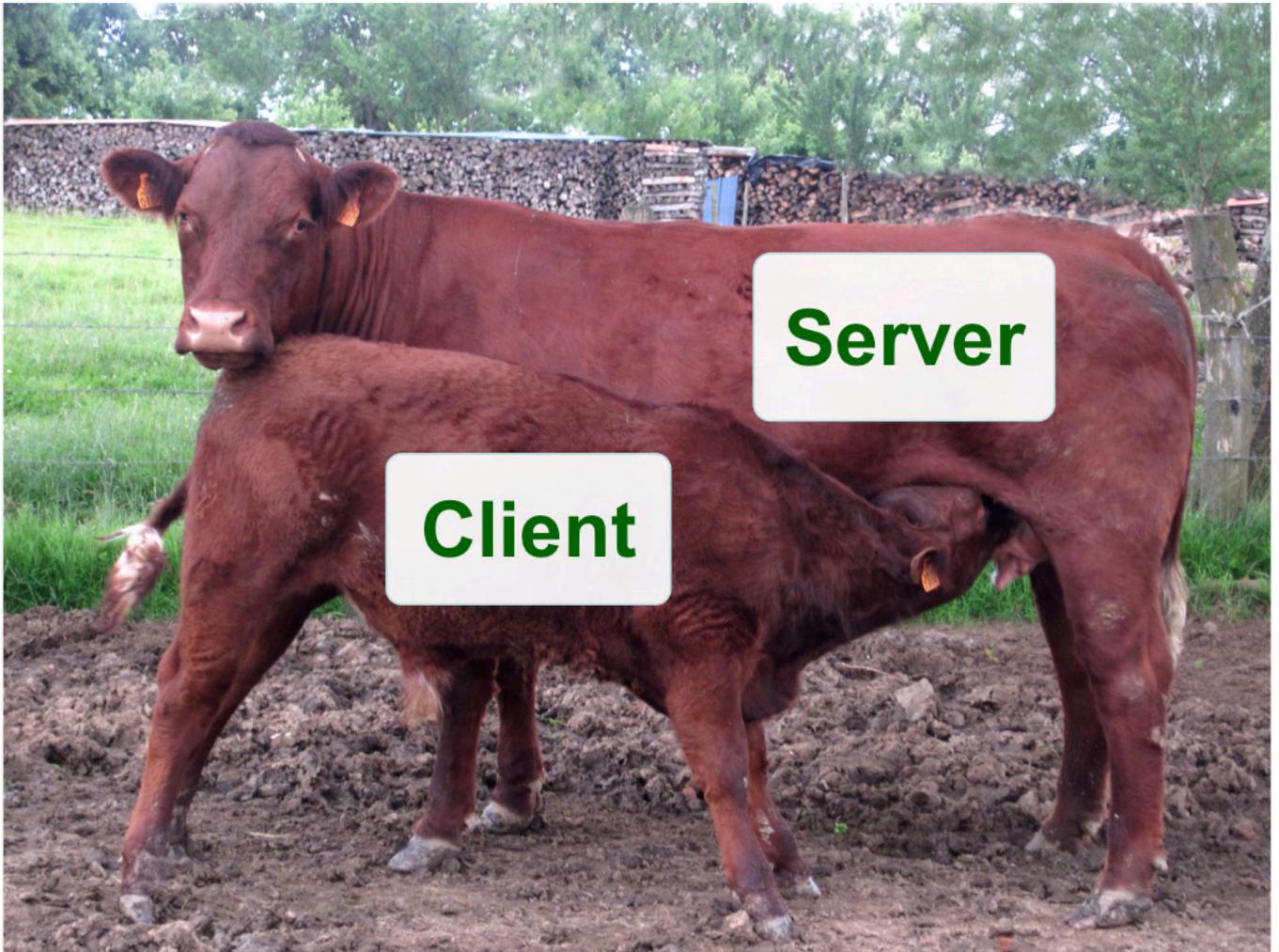


Figure 1. Client-Server or Calf-Cow

I am writing this to prepare for a talk I'll give (as a *Linux Journal* editor) at the Workshop on Meaningful Consent in the Digital Economy 2015. It's at the University of Southampton in the UK on February 26, which means it will have happened by the time you read this. It will be interesting to see how coverage differs from what I'm planning to say, and you're about to read.

By "meaningful consent", they

mean "issues related to giving and obtaining user consent online, with special emphasis on privacy and data protection". I'm focusing on the giving side, because that's the frontier. Very few commercial sites give consent to users of any meaningful kind—except, perhaps, as legal butt-covering. ("Here's our consent: go look at our privacy policy.") And there are few ways for individuals to express the desire for consent,

especially around privacy. (Do Not Track is just one of them.) Basically we travel the Web naked, unless we're wizards (such as *Linux Journal* readers) who know how to secure their on-line homes, wear the right protective clothing and customize their own vehicles. But ways are being developed for the muggles of the world, and I want to run a few of those down. Here they are:

1. Do Not Track.
2. Ad and tracking blockers.
3. Privacy icons.
4. UMA (User Managed Access).
5. IDESG's Internet Ecosystem Steering Group.
6. Open Notice and Consent Receipts.
7. Respect Trust Framework.
8. Customer Commons' user-submitted terms.
9. CommonAccord's Digital Law Commons.

1. Do Not Track is an HTTP header (DNT) that asks a site or an app to

disable unwanted tracking. There is disagreement about what kinds of tracking should be disabled and how various things work. But all the browser makers enable it (in different ways), so that's progress. And, regardless of whatever becomes of DNT, the step is still in the right direction, because it carries a signal of intent from the individual. Consent comes back the other way—or should. (I first heard of DNT from Chris Soghoian at a Berkman Center meeting in the late 2000s. Chris, Sid Stamm and Dan Kaminsky are regarded as DNT's original authors. In recent years, the W3C has carried the DNT ball, through many internal and external disagreements—especially with the IAB and the DAA. (Chris also published a detailed history of DNT in January 2011.)

2. Ad and tracking blockers

selectively throttle tracking, advertising or both. Here's a partial list of the ones I have installed on my own browsers:

- Adblock Plus: <https://adblockplus.org>.
- AVG PrivacyFix: <http://www.avg.com/us-en/privacyfix>.
- Customer Commons Web Pal: <http://customercommons.org/about-web-pal>.

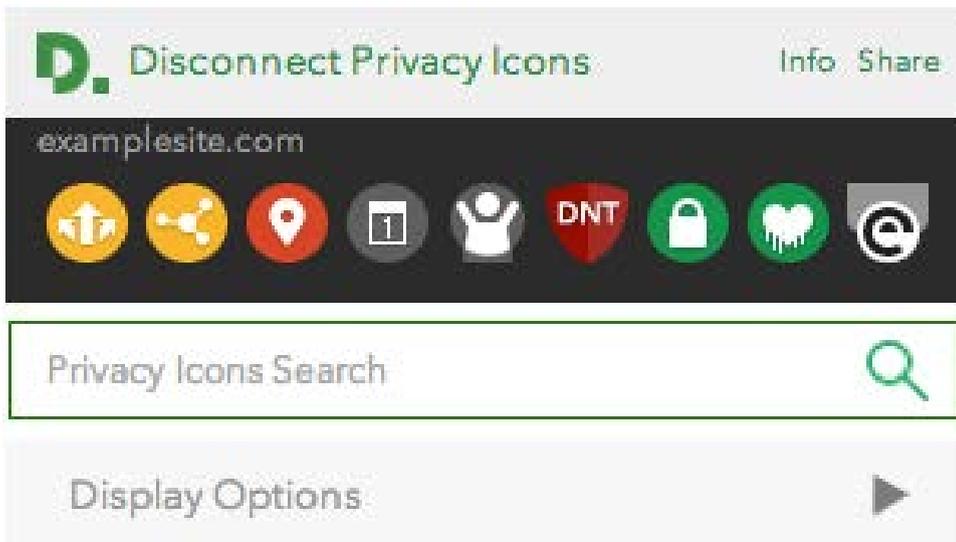


Figure 2.
Privacy Icons

- Disconnect: <https://disconnect.me>.
- Ghostery: <https://www.ghostery.com/en>.
- Privacy Badger: <https://www.eff.org/privacybadger>.
- PrivownyBar: <https://privowny.com>.

Each has advantages, none of which I'll visit here. As for consent, they all fail to signal much if anything. Their main work is prophylactic.

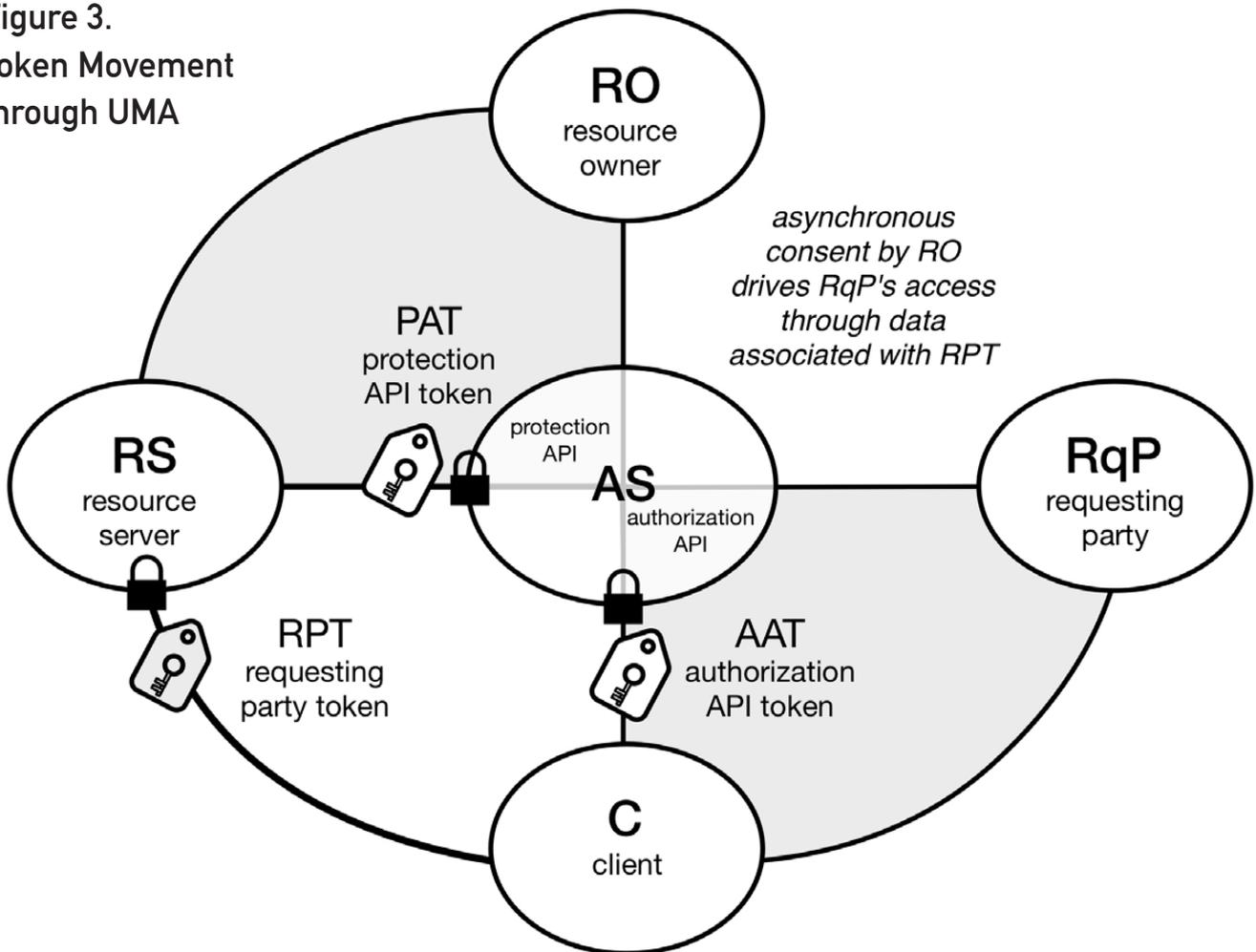
3. Privacy icons are visual signals. Disconnect's can "read and change all your data on the websites you visit". They look like Figure 2.

An earlier effort is Aza Raskin's, which became Mozilla's. The following is a list of what each symbol says. (The images are gone from Aza's original post, but are available at his Flickr site

with a Creative Commons Attribution-NonCommercial 2.0 Generic license. Since *Linux Journal* is commercial, we'll leave seeing them up to you—see Resources.)

- Your data is only for the intended use.
- Your data may be used for purposes you do not intend.
- Your data is never bartered or sold.
- Your data may be bartered or sold.
- Your data is never given to advertisers.
- Site gives your data to advertisers.
- Your data is given to law enforcement only when legal process is followed.

Figure 3.
Token Movement
through UMA



- Data may be given to law enforcement even when legal process is not followed.
- Your data may be kept for less than one month (or three, or six, or eighteen months).
- Your data may be kept indefinitely.

Those would be handy. Alas, the commercial Web site publishing business has shown little interest in

them and will continue so as long as all the signaling is left up to them.

4. UMA—User Managed Access is the brainchild of Eve Maler (currently an analyst with Forrester Research), and its home is the User Managed Access WG (Working Group) at Kantara. UMA's charter is "to develop specs that let an individual control the authorization of data sharing and service access made between online services on the individual's behalf, and to facilitate interoperable

implementations of the specs". It's an OAuth-based protocol. Token movement through UMA currently looks like Figure 3.

5. IDESG's Identity Ecosystem Steering Group is part of NSTIC: the National Strategy for Trusted Identities in Cyberspace, which was launched by the White House in 2011 and is meant to create an implementation road map that will reside within the Department of Commerce. The IDESG is working toward "secure, user-friendly ways to give individuals and organizations confidence in their online interactions". Here's the wiki: https://www.idecosystem.org/wiki/Main_Page. And here is the User Experience Committee: <https://www.idecosystem.org/group/user-experience-committee>, which is taking the lead on this thing.

6. Open Notice and Consent Receipts is an OpenNotice.org project, "a group of people and projects that are innovating to address the broken notice and consent infrastructure to enable greater control of personal data". Its main focus is on the consent receipt project. Work here lives at the Consent & Information Sharing Work Group (CISWG) at Kantara. The purpose is to specify receipts of consent exchanged between first

and second parties. Everything else I can tell you about it is beyond complicated. What matters is that it's being worked on by highly committed people who not only grok it, but also are working to simplify it.

7. Respect Trust Framework is one of five frameworks created by the Open Identity Exchange. This one requires that parties to the framework promise to "respect each others' digital boundaries". This past year many individuals, companies and development projects (me included) joined with the Respect Network ("the first global ecosystem for trusted personal information exchange") to at least agree to the Framework. Respect Network as a company ran out of runway, but it did succeed in getting the Framework agreed to by a pile of parties, which is an accomplishment by itself.

8. Customer Commons user-submitted terms is intended to do for individual terms what Creative Commons did for copyright licenses. Figure 4 shows one straw man proposal, drawn originally on a whiteboard at VRM Day in October 2014.

It derives from EmanciTerms, by ProjectVRM, and which I described like this in *The*

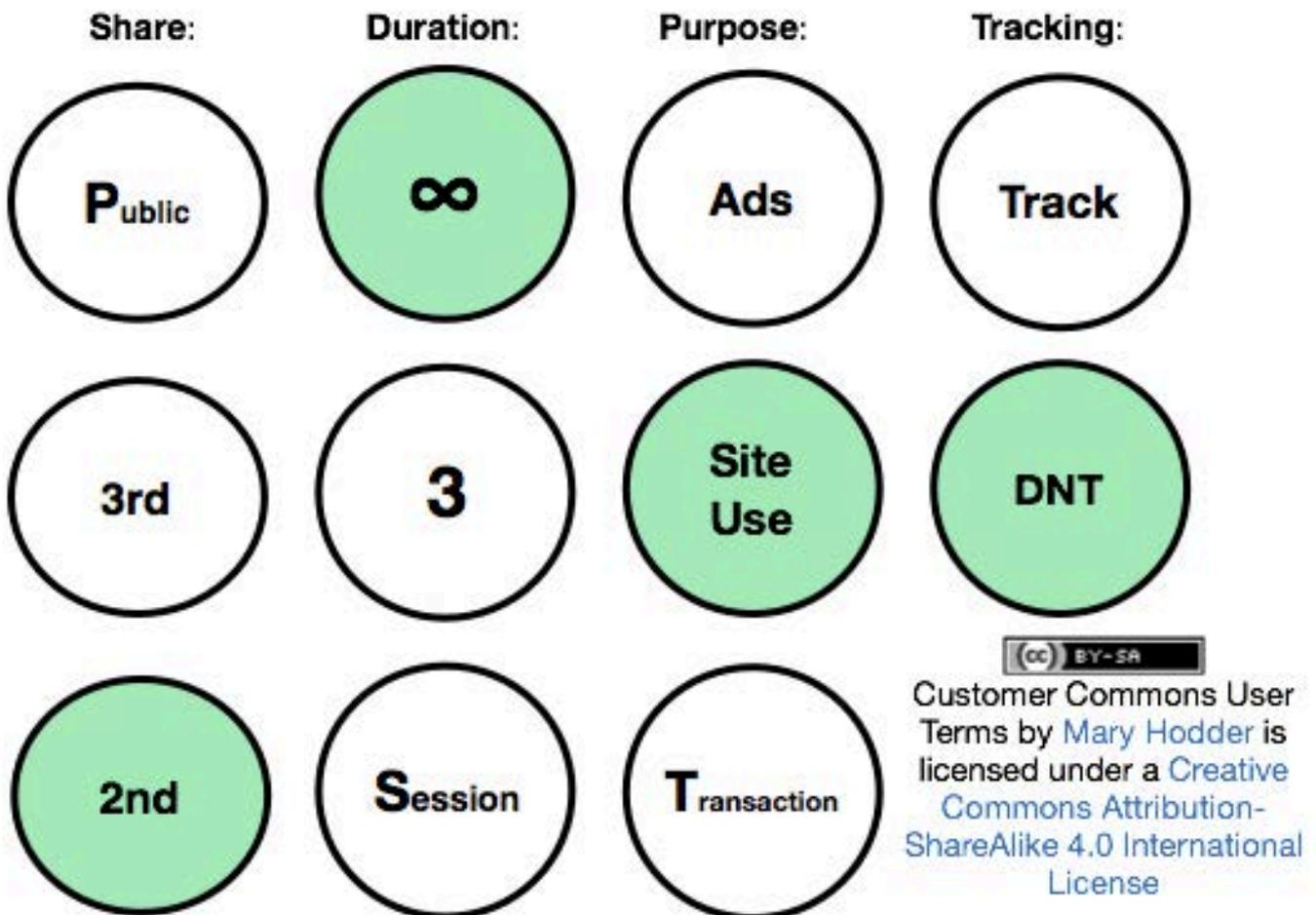
Intention Economy:

With full agency...an individual can say, in the first-person voice, "I own my data, I control who gets access to it, and I specify what I wish to happen under what conditions." In the latter category, those wishes

might include:

- Don't track my activities outside of this site.
- Don't put cookies in my browser for anything other than helping us remember each other and where we were.

MY TERMS: Icon format and structure



NOTE: I'm the first party. My terms are: 2nd-∞-SU-DNT

Figure 4. Customer Commons Straw Man Proposal from VRM Day in October 2014

- Make data collected about me available in a standard, open format.
- Please meet my fourth-party agent, Personal.com (or whomever).

These are EmanciTerms, and there will be corresponding ones on the vendor's side. Once they are made simple and straightforward enough, they should become normative to the point where they serve as *de facto* standards, in practice.

Since the terms should be agreeable and can be expressed in text that code can parse, the process of arriving at agreements can be automated.

The mission of Customer Commons (a nonprofit spin-off of ProjectVRM), is "to restore the balance of power, respect and trust between individuals and the organizations that serve them". Toward making that happen through EmanciTerms, Customer Commons has engaged the Cyberlaw Clinic at the Berkman Center and Harvard Law School, which "provides high-quality, pro-bono legal services to appropriate clients on issues relating to the Internet,

new technology, and intellectual property". It's still early in that process, but the end result will be terms that you and I can assert, and others will need to accept.

9. CommonAccord's Digital Law Commons is an end state when you have what Jim Hazard and Primavera De Filippi call "a way of rendering a document from snippets of text organized as key-values in lists—a 'graph'. It's applicable to a lot of knowledge management tasks, but especially useful for codifying legal docs". In slightly more technical terms, the Common Accord site explains, "We have created a modular template system of text cards that relies on {expansion} of strings and [expansion] of cards. Period. People can program their relationships. Lawyers can codify boilerplate. Management can have a data picture of the enterprise's relationships, situation and activities. Smart contracts can be both technical 'dry' code (i.e. self-contained code snippets) and legal 'wet' code. With Common Accord, contract and legal text becomes cards of text, interoperating, in git, shared, forked, tested and improved." Here it is on Github: <https://github.com/CommonAccord/Org/tree/master/Doc>.

I expect many of these efforts to merge, support each other and cross-fertilize.

I expect many of these efforts to merge, support each other and cross-fertilize. There is a lot of convergence already.

Why should I be optimistic about results? Four reasons.

First is tech. Code is law, Professor Lessig taught us, and the code required for asserting and agreeing to terms won't be terribly complicated.

Second is publicity. We—Customer Commons (on the board of which I sit) and friends—will make a Big Thing out of the term, once we have them, just like Creative Commons made a Big Thing out of its licenses when those came out. Sites and services that don't listen to what users and customers want will be exposed and shamed. Simple as that.

Third is pickup. It won't be hard for organizations like Consumer Reports—as well as everybody in the lazyweb's long tail—to give thumbs-up and thumbs-down to sites and services that agree to simple and reasonable terms submitted by customers and users.

Fourth is performance. As I put it in *The Intention Economy*:

Rather than guessing what might get the attention of consumers—or what might “drive” them like cattle—vendors will respond to *actual intentions of customers*. Once customers' expressions of intent become abundant and clear, the range of economic interplay between supply and demand will widen, and its sum will increase. The result we will call the *Intention Economy*.

This new economy will outperform the *Attention Economy* that has shaped marketing and sales since the dawn of advertising. Customer intentions, well expressed and understood, will improve marketing and sales, because both will work with better information, and both will be spared the cost and effort wasted on guesses about what customers might want, flooding media with messages that miss their marks. Advertising will also improve.

The volume, variety, and relevance of information coming from customers in the Intention Economy will strip the gears

Resources

Do Not Track: https://en.wikipedia.org/wiki/Do_Not_Track

Workshop on Meaningful Consent in the Digital Economy 2015: http://www.meaningfulconsent.org/blog/?page_id=24

University of Southampton: <http://www.southampton.ac.uk>

List of HTTP Header Fields: https://en.wikipedia.org/wiki/List_of_HTTP_header_fields

Christopher Soghoian: https://en.wikipedia.org/wiki/Christopher_Soghoian

Dan Kaminsky: https://en.wikipedia.org/wiki/Dan_Kaminsky

World Wide Web Consortium: https://en.wikipedia.org/wiki/World_Wide_Web_Consortium

IAB: <http://www.iab.net>

DAA: <http://www.digitaladvertisingalliance.org>

“The History of the Do Not Track Header” by Christopher Soghoian: <http://paranoia.dubfire.net/2011/01/history-of-do-not-track-header.html>

Aza Raskin: <http://www.azarask.in/blog>

Mozilla Wiki: <https://wiki.mozilla.org/Drumbeat/MoJo>

Aza Raskin’s Post on Privacy Icons: <http://www.azarask.in/blog/post/privacy-icons>

Asa Raskin’s Flickr Site: <https://www.flickr.com/photos/azaraskin/5304502420>

User-Managed Access (UMA): https://en.wikipedia.org/wiki/User-Managed_Access

Eve Maler’s Blog: <http://blogs.forrester.com/blog/21486>

Forrester Research: <https://www.forrester.com/home>

User Managed Access Working Group: <http://kantarainitiative.org/confluence/display/uma/Home>

Kantara Initiative: <https://kantarainitiative.org>

Identity Ecosystem Steering Group: <https://www.idecosystem.org>

National Strategy for Trusted Identities in Cyberspace PDF: http://www.whitehouse.gov/sites/default/files/rss_viewer/NSTICstrategy_041511.pdf

National Strategy for Trusted Identities in Cyberspace (NSTIC): <https://www.idecosystem.org/page/national-strategy-trusted-identities-cyberspace-nstic>

IDESG Wiki: https://www.idecosystem.org/wiki/Main_Page

OpenNotice.org: <http://opennotice.org>

Consent Receipt Project: <http://opennotice.org/a-quick-history-the-consent-receipt>

Consent & Information Sharing Work Group: <https://kantarainitiative.org/confluence/display/infosharing/Home>

Trust Frameworks: <http://openidentityexchange.org/resources/trust-frameworks>

Open Identity Exchange: <http://openidentityexchange.org>

Customer Commons and User Submitted Terms: <http://customercommons.org/2014/10/27/customer-commons-and-user-submitted-terms>

Respect Network: <https://www.respectnetwork.com>

EmanciTerm: <http://cyber.law.harvard.edu/projectvrm/EmanciTerm>

ProjectVRM: <http://blogs.law.harvard.edu/vrm>

Cyberlaw Clinic: <http://cyber.law.harvard.edu/node/1321>

commonaccord.org: <http://commonaccord.org>

Code and Other Laws of Cyberspace by Lawrence Lessig: <http://code-is-law.org>

Professor Lessig: <http://www.lessig.org>

“Code Is Law” by Lawrence Lessig: <http://harvardmagazine.com/2000/01/code-is-law.html>



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

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!

