EFI | OpenRISC | Mercurial | Netfilter | SQS | es

FREE TO SUBSCRIBERS

EPUB, Kindle, Android, iPhone & iPad editions

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community

DECEMBER 2011 | ISSUE 212 | www.linuxjournal.com

OPENRIS
AN INTRODUC

USE MERCURIAL FOR VERSION CONTROL

READERS' CHOICE AWARDS 2011

The Votes Are In!

+ SCALE WEB APPS WITH AMAZON'S SIMPLE QUEUE SERVICE

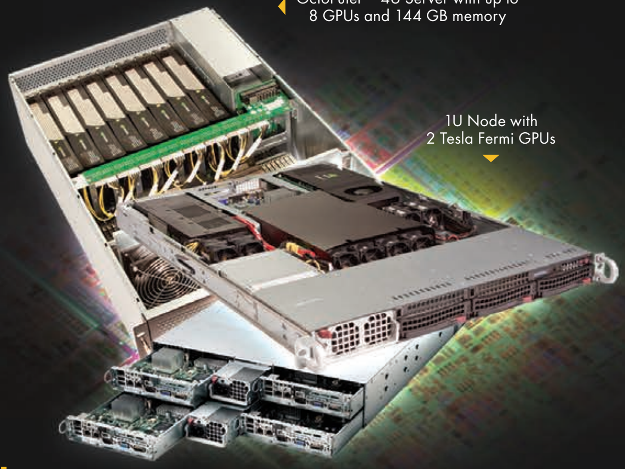
HOW TO:
Read *Linux Journal* on the Command Line

Best Practices for Designing Monitoring Systems

EFI Features and How They Impact Linux

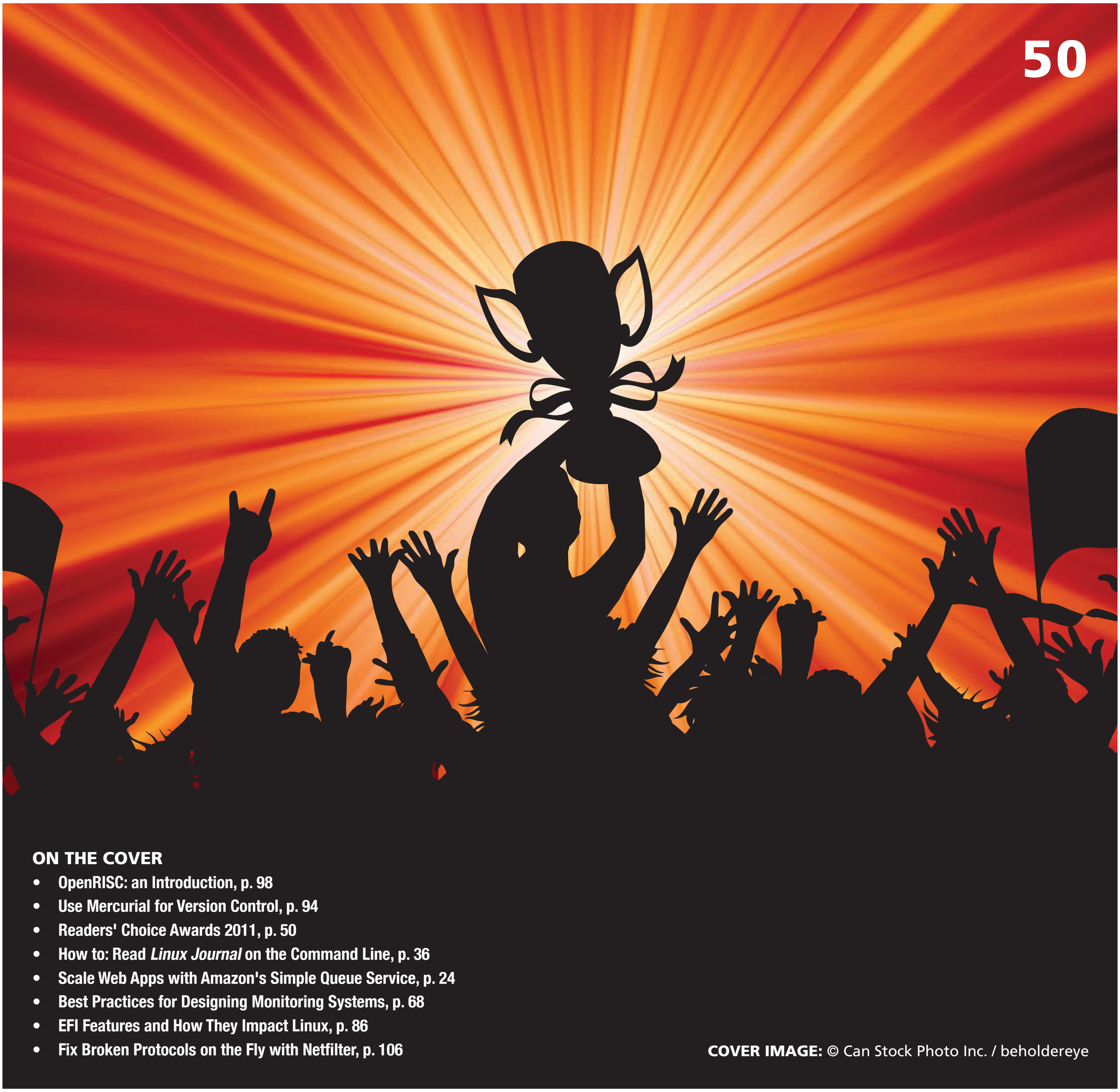
Fix Broken Protocols on the Fly with Netfilter

Understanding Drupal
Administering Drupal
Site Building with Drupal
Learning CCK for Drupal
Learning Views for Drupal
Theming Basics for Drupal
Advanced Theming for Drupal
Introduction to jQuery
jQuery & JavaScript in Drupal
Drupal Module Development



CONTENTS DECEMBER 2011 ISSUE 212

FEATURE



ON THE COVER

COVER IMAGE: © Can Stock Photo Inc. / beholdereye

COLUMNS

```
30 years. He's the author of the popular Wicked Cool Shell
quent column. Instead, I encourage you
Scripts and can be found on Twitter as @DaveTaylor and more
to grab the entire source code library
generally at www.DaveTaylorOnline.com.
RACKMOUNT SERVERS
RACKMOUNT
RACKMOUNT SERVERS
```

```
COL UMNS
HACK AND /
Remotely
KYLE RANKIN
Wipe a Server
What would you do if you had to erase all the files securely on a
server thousands of miles away?
In many ways, I feel sorry for people
simple request when the server is under
```

36 READ *LINUX JOURNAL* ON THE COMMAND LINE

IN EVERY ISSUE

INDEPTH

LINUX JOURNAL (ISSN 1075-3583) is published monthly by Belltown Media, Inc., 2121 Sage Road, Ste. 310, Houston, TX 77056 USA. Subscription rate is $29.50/year. Subscriptions start with the next issue.

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

Proofreader	Geri Gale

Publisher	Carlie Fairchild publisher@linuxjournal.com
Advertising Sales Manager	Rebecca Cassity rebecca@linuxjournal.com
Associate Publisher	Mark Irgang mark@linuxjournal.com
Webmistress	Katherine Druckman webmistress@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

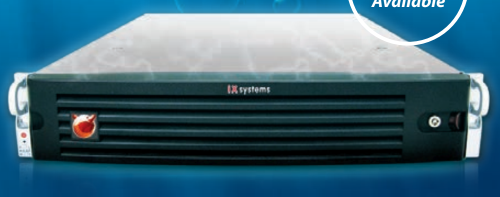
***Linux Journal* is published by, and is a registered trade name of,**
Belltown Media, Inc.
PO Box 980985, Houston, TX 77098 USA

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 16476, North Hollywood, CA 91615-9911 USA

TrueNAS™ 2U Appliance: You Are the Cloud

Expansion Shelves Available

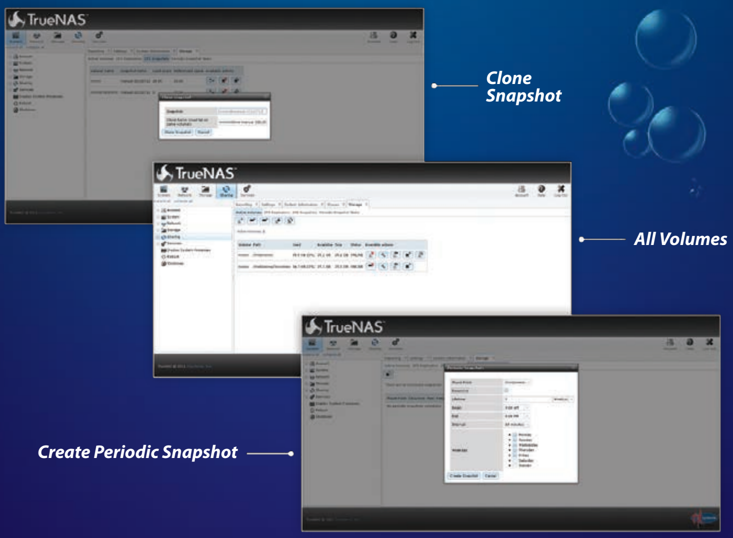


FreeNAS 8 POWERED

Storage. Speed. Stability.

With a rock-solid FreeBSD® base, Zettabyte File System (ZFS) support, and a powerful Web GUI, TrueNAS™ pairs easy-to-manage FreeNAS™ software with world-class hardware and support for an unbeatable storage solution. In order to achieve maximum performance, the TrueNAS™ 2U System, equipped with the Intel® Xeon® Processor 5600 Series, supports Fusion-io's Flash Memory Cards and 10 GbE Network Cards. Titan TrueNAS™ 2U Appliances are an excellent storage solution for video streaming, file hosting, virtualization, and more. Paired with optional JBOD expansion units, the TrueNAS™ System offers excellent capacity at an affordable price.

For more information on the **TrueNAS™ 2U System**, or to request a quote, visit: **http://www.iXsystems.com/TrueNAS**.



KEY FEATURES:

- Supports One or Two Quad-Core or Six-Core, Intel® Xeon® Processor 5600 Series
- 12 Hot-Swap Drive Bays - Up to 36TB of Data Storage Capacity*
- Periodic Snapshots Feature Allows You to Restore Data from a Previously Generated Snapshot
- Remote Replication Allows You to Copy a Snapshot to an Offsite Server, for Maximum Data Security
- Software RAID-Z with up to Triple Parity
- 2 x 1GbE Network interface (Onboard) + Up to 4 Additional 1GbE Ports or Single/ Dual Port 10 GbE Network Cards

JBOD expansion is available on the 2U System

* 2.5" drive options available; please consult with your Account Manager

iXsystems®

intel® Xeon® inside™

Powerful. Intelligent.

Call iXsystems toll free or visit our website today!

1-855-GREP-4-IX | www.iXsystems.com

Intel, the Intel logo, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation in the U.S. and/or other countries.

SHAWN POWERS

The Customer Is Always Right

This is the time of year when the *Linux Journal* staff turns to you, our readers, for insight on the best programs in the Linux world. I love this time of year. No, not because you all do most of the work, but rather because I get to see how my preferences compare to those of our readership. You get to do the same. Whether you're looking for validation with your software choices or hoping to fill a gap in your digital repertoire, this issue should please.

Along with the Readers' Choice winners, we have an issue full of "choice" articles we've picked to go along with this month's theme. Reuven M. Lerner shows us an easy way to scale Web applications with Amazon's Simple Queue Service (SQS). Amazon makes scaling services simple, and Web applications are no exception. Dave Taylor describes how to make a scale of our own for rating Twitter accounts. Using scripting (Dave's specialty), extracting data about a Twitter account is pretty simple. Come up with your own formulas for what makes tweets terrific, and you can make a script that is judge and jury all in one.

Our other command-line guru, Kyle Rankin, teaches us to laugh in the face of E Ink and scoff at the Kindles of Amazon. In the same way Kyle chats with Irssi, e-mails with Mutt and system-administers from an xterm, this month he shows how to read *Linux Journal* with his e-reader of choice: a terminal window. If you're a minimalist like Kyle or just like to out-geek the person next to you, you'll want to read Kyle's article. At the very least, it will make you thankful for your digital e-reader!

Michael Nugent addresses a problem this month that is near and dear to me. Every sysadmin should have a monitoring system, but what happens when that monitoring system is more annoying than helpful? I get daily e-mail messages from several of my systems with reports on their success or failure. After 20–30 days of "all normal", the messages tend to slip past my radar. Then one day when they stop arriving, their absence goes unnoticed. The opposite can be true though as well. How may times have you been woken up by your pager beeping incessantly over a false positive? At 3 o'clock in the morning? Michael discusses some best practices for making your monitoring system effective at doing its job while not driving you insane in the process.

If you're a software developer, you will want to check out Daniel Bartholomew's article on databases. Sure, databases aren't the most exciting things in the world, but if you're a programmer, interfacing with them is important. Add to that Joey Bernard's article on Mercurial for revision control, and it's like soup for the programmer's soul.

We realize not everyone is into programming though, and for you hardware hackers, we have a couple exciting articles as well. James Tandon shows us the open-source processor OpenRISC and teaches us some tricks for utilizing it. As a community that historically has struggled with working with proprietary hardware, the open-source hardware idea is very attractive. Roderick W. Smith helps us stay ahead of the hardware transition game this month too. He describes the new EFI boot mechanism that is slowly taking over the role of BIOS in computers. Since hardware manufacturers will be moving more and more toward EFI, it's important for us to understand. After all, "booting up" is a pretty important part of any operating system—even if it is only once every few years for Linux users.

Networking folks haven't been left out this month either. Paul Amaranth shows us a pretty neat method of fixing broken NAT protocols using NF_QUEUE. NAT works so well anymore, most of us take it for granted. Sometimes it doesn't work as magically as we expect, however, and Paul shows us how to do some magic of our own. Bill Childers and Kyle Rankin close off the issue with a scary, but educational, story about wiping out their data center—over and over. It's scary the damage we can do accidentally when we work on production servers. Bill and Kyle are two people I turn to when I have issues I can't solve, and as you'll read this month, they've learned much of their knowledge at the school of hard knocks.

We'd like to thank you, our readers, for making this issue fun for us. It's great to hear from you regarding what software and hardware you prefer. It not only helps us produce a magazine that will fit your needs, it also gives us a chance to learn from you. So sit back with your Kindle, prop up an iPad or flip through digital pages with your Android. This month, you get to see how you line up with other *Linux Journal* readers. We hope you enjoy.■

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

Mustache.js | ipset | Lustre | tcpdump | ENet | SSDs
LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
KEEP YOUR DATA SAFE ON PUBLIC NETWORKS
HACK YOUR HOME ROUTER
NETWORKING
ipset for Advanced Firewall Configuration
tcpdump the Powerful Network Analysis Utility
Deploy a Scalable and High-Performing Distributed Filesystem with Lustre
REVIEWED: INTEL'S 320 SSD LINE
October 2011 | ISSUE 210
www.linuxjournal.com

Home Router Hacking

I thoroughly enjoyed Kyle Rankin's article "Practice Hacking on Your Home Router" in the October 2011 issue. Usually system cracking gets a bit deep for me, but this example was both practical and easy to follow—nice! One minor nit: data encoded into a URL is actually GET data rather than POST data. (I'll give him a pass this time on "hacking" vs. "cracking".)

—Grant Root

Kyle Rankin replies: *Hey, precision is important, so thanks for the nitpick! You are right. The way I was submitting data to the router was GET data. GET or POST, the lesson is that if you accept input, sanitize it. If you are curious about why I use "hacker", I explain it in detail in my November 2010 column "Some Hacks from DEF CON".*

Return to Solid State

I enjoyed getting a little comparative data on SSD performance [see Kyle Rankin's "Return to Solid State" review in the October 2011 issue]. My own experience is with an under-powered Acer Aspire One Netbook. One note, to further improve disk IO time and SSD wear, I load all partitions with noatime and nodiratime options in fstab.

Acer Aspire 532h Fedora 14 i686:

OCZ 60GB:

- Cold boot to GNOME login: 27 seconds
- Log in to usable desktop: 2 seconds
- Open Firefox to cursor at URL bar: 2 seconds
- Open LibreOffice Write to blank doc: 2 seconds

Seagate ST9250827AS:

- Cold boot to GNOME login: 47 seconds
- Log in to usable desktop: 16 seconds
- Open Firefox to cursor at URL bar: 5 seconds
- Open LibreOffice Write to blank doc: 7 seconds

Thanks.

—rathomas

***Kyle Rankin replies:** Thanks for the feedback—a good point about atime. I can understand that it makes sense to disable atime completely on some filesystems for speed (such as mountpoints dedicated to database storage), but for me, I find such benefit in atime for forensics that if speed is a concern, I like to use the new relatime option in the fstab. That way, I still get atime; however, atime writes are cached so it gives much better performance.*

The Digital Subscription

I want to thank you for converting to all-digital download for *Linux Journal*. I see that in the past the magazine kept shrinking slowly over time, and now that you have gone 100% digital, it has grown in size! I think that is a plus for you. At first, I was worried about whether I would like the digital version, but after two months, I find I have begun to like it. Keep up the good work and the great articles!
—Jim Brown

You bring up a point I hadn't even considered in regards to the "thickness" of the magazine. We can be a little more flexible now than we could be with paper. Thanks for the kudos as well. In all the chaos involved with the digital transition, the one thing we wanted to maintain was content.—Ed.

Digital Format Surprise

I'm a longtime reader, first-time feedbacker. I just finished reading my first digital copy of *Linux Journal*, and although I thought I would hate the experience compared to reading in dead-tree format, I was pleasantly surprised. I spend all day staring at LCD pixels, and I rather disliked the idea of including my casual reading here. With the paper version of the journal, I often would read about something cool and say to myself, "Neat project I should check this out next time I sit at my computer." Only I would never remember to remember to check it out. I love the HTML links built in to the digital version of the journal. It is so much more of an interactive experience. And that got me thinking, take one more step by creating a wiki-like version of the journal with user comments that show up next to articles, videos and so on—just a thought. Keep up the good work.

—Dan

Cool idea Dan. As we learn to do this whole digital thing better and better every month, it will make new ideas feasible. Now that the entire production process is in-house (no more sending off to the printer), it makes experiments much easier as well. I totally agree regarding hyperlinks too. In the past, I've done my best to include short URLs in printed articles, but now we can use the real links and not worry about someone typing them in by hand. My favorite thing about digital though? Searching.—Ed.

epub

I tried out the epub version of the September *Linux Journal* on my Nook Color and enjoyed reading it on that device—until I came to the graphics. It seems that images, such as a Bash screen in an article, are too small on the Nook to be read. They don't magnify either. I tried turning the Nook on its side for landscape view, which it will do with books and Web pages, but the pages wouldn't landscape. Yuck!

So, please study the epub version for a way to fix landscape or magnify images (perhaps embed a clickable bigger picture). Perhaps the problem is really with the Nook and epubs. In general, the Nook has problems with PDFs also. Usually they will magnify, but the landscape mode tends to be busted or useless with it still in a smaller portrait mode while turned 90 degrees.

I guess for now I won't be able to read *LJ* on my Nook with much success. I'll read it on my computer instead.

—Dave

You are absolutely correct. Although we've been doing magazine layout forever, this whole "flowing text" thing is new for us. Trying to make epub files that work well on all devices is really challenging. We are working hard every *month to make the experience a little better, and I think we've succeeded a little bit each month. Another challenge is example code. Because devices are so varied in size, it's rough to make the code "perfect" on each reader.*

Like I said, we're working on honing our skills, so you should see improvement every month. Sending in comments like this really helps us determine what to focus on, so thank you again!—Ed.

Please Write about epub Tools

This month, I received *Linux Journal* in pure electronic form. I got it in PDF and mobi formats, and I can read it directly on my Kindle. I realized that since *Linux Journal* decided to deliver its content in e-formats, maybe it can write about it. For instance, do an overview of available software tools, techniques for text formatting (especially code formatting suitable for gadgets), tools for transforming from one format to another and so on. I think it would be great topic for readers.
—Valentin

I agree! I want to see some articles on epub creation myself, so hopefully we can find some experts on the topic. I know there are many conversion tools out there, but what I'm looking for (and I think you're looking for) are tools to create them or edit them. The epub format offers so many cool features like table of contents, chaptering, graphic placement and so on, and I want to learn how to make or tweak my own. Thanks for sharing my interest.—Ed.

WRITE *LJ* A LETTER We love hearing from our readers. Please send us your comments and feedback via **http://www.linuxjournal.com/contact**.

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 16476, North Hollywood, CA 91615-9911 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://ww.linuxjournal.com/advertising**. Contact us directly for further information: ads@linuxjournal.com or +1 713-344-1956 ext. 2.

diff -u
WHAT'S NEW IN KERNEL DEVELOPMENT

You may have heard of the recent security breach that took place on **kernel.org**. The attacker gained root access to the servers and modified a kernel source-tree release candidate, in the hope of infecting lots of users.

Since then, the kernel.org system administrators have been working like mad, cleaning out the servers and implementing security measures that might hopefully prevent another attack. One such measure involves restricting access to kernel.org itself. In the past, people maintaining a git tree on kernel.org could get a shell account on that system. Those days are gone. **H. Peter Anvin** announced that the **gitolite** tool would be used to update git trees from now on, and shell access would no longer be handed out as freely as it was.

The kernel.org folks also are instituting a cryptographic "web of trust", so that people maintaining a git tree will be able to establish their identity when doing updates. If you're a developer who hacks the kernel in your spare time or for your employer and typically submits patches via e-mail, you won't need to be part of the web of trust; in fact, your work flow can continue unchanged. Only folks involved in maintaining projects on kernel.org are affected by these new policies.

Linus Torvalds has expressed some doubt that cryptographic signatures are as important as others believe. He said, "Realistically, I checked a few signatures this time around due to the kernel.org issues, but at the same time, the thing that made me trust most of it was just looking at commits and the e-mail messages—the unconscious and non-cryptographic 'signature' of a person acting like you expect a person to act."

Andi Kleen has resubmitted his patch that makes **3.0 kernels** pretend to be 2.6 kernels, so binary-only software expecting to run on a 2.6 kernel still will run correctly under 3.0 kernels. It's an ugly pill to swallow. This time around, Linus Torvalds asked which binary-only software actually was breaking under 3.0, and a number of people replied, listing off several applications. Some **HP** management tools were among them. There also were a lot of nonbinary applications, including a number of Python scripts that performed an incorrect test for the current kernel version number.

Linus seems very reluctant to adopt this patch, especially considering that Andi has stated positively it's not just a short-term fix, but that 3.0 kernels would have to continue to masquerade as 2.6 kernels for the long term, in order to maintain compatibility with those binary-only tools.

—ZACK BROWN

Goodbye GNOME 2, Hello GNOME 2?

Many Linux users who have been GNOME fans for years find themselves in a sudden quandary. GNOME 3.0 has completely abandoned the desktop experience we've come to love during the years. That's not to say change is bad, it's just that many folks (even Linus Torvalds) don't really want to change.

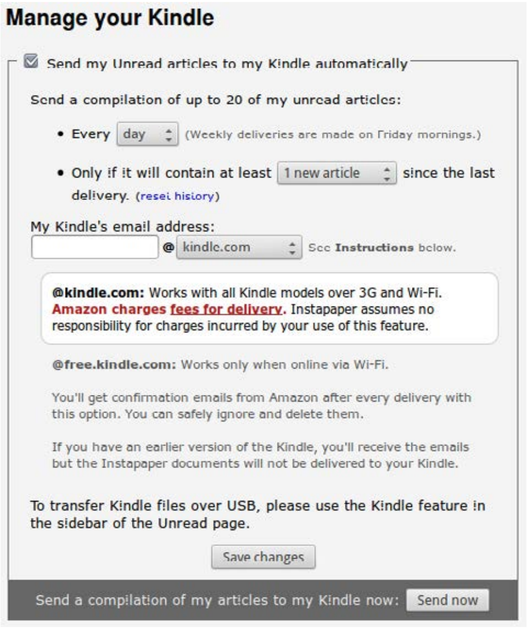
As an Ubuntu user for several years, I'm accustomed to how well Canonical makes Linux on the desktop "just work". Unfortunately, Ubuntu's alternative to the GNOME 3 switch is Unity. I want to like Unity. I've forced myself to use it to see if it might grow on me after a while. It hasn't. And, to make matters worse, version 11.10 won't have a classic GNOME option, which means I either need to bite the bullet and get used to Unity or go with an alternative.

Thankfully, XFCE has all the features I love about GNOME. No, XFCE isn't exactly like GNOME, but it feels more like GNOME 2 than GNOME 3 does! If you are like me and desperately want to have the old GNOME interface you know and love, I recommend checking out Xubuntu (the version of Ubuntu that uses XFCE). With minimal tweaking, it can look and feel like GNOME 2. Plus, XFCE has the ability to start GNOME (or KDE) services on login, which means GNOME-native apps usually "just work".

The time may come when we're forced to adopt a new desktop model. For the time being, however, alternatives like XFCE or even LDXE offer familiar and highly functional desktop experiences. If you fear GNOME 3 and Unity, try XFCE. Download Xubuntu and check it out: **http://www.xubuntu.org**.

—SHAWN POWERS

Get More from Your e-Reader: Instapaper



Manage your Kindle

☑ Send my Unread articles to my Kindle automatically

Send a compilation of up to 20 of my unread articles:

- Every day (Weekly deliveries are made on Friday mornings.)
- Only if it will contain at least 1 new article since the last delivery. (reset history)

My Kindle's email address:
@ kindle.com See **Instructions** below.

@kindle.com: Works with all Kindle models over 3G and Wi-Fi. Amazon charges fees for delivery. Instapaper assumes no responsibility for charges incurred by your use of this feature.

@free.kindle.com: Works only when online via Wi-Fi.

You'll get confirmation emails from Amazon after every delivery with this option. You can safely ignore and delete them.

If you have an earlier version of the Kindle, you'll receive the emails but the Instapaper documents will not be delivered to your Kindle.

To transfer Kindle files over USB, please use the Kindle feature in the sidebar of the Unread page.

Send a compilation of my articles to my Kindle now:

If you use a dedicated e-reader to read *Linux Journal* every month, chances are you want to read other material on it as well. Thanks to a free service called Instapaper, if you have an e-reader like the Linux-powered Kindle, you can take your favorite Web articles with you on the go, even if your destination doesn't have Internet access!

Instapaper works by giving you a bookmark to click anytime you want to view a Web page later on your e-reader. Then, from your Internet-enabled mobile device, you can read the articles at your leisure. For those folks with a Kindle, Instapaper provides a free delivery service that sends batches of articles to your Amazon account. It's important to realize Amazon will charge you for 3G-based deliveries, but if your Kindle supports Wi-Fi, delivery is free as well.

Instapaper is great if you don't like to do the bulk of your reading in a Web browser, but still want to read the latest news from the Internet. The service is currently free, but you can become a subscriber for $1 a month and support the company. Instapaper is available at **http://www.instapaper.com**.

—**SHAWN POWERS**

They Said It

If I'm not back in five minutes...just wait longer.—Ace Ventura, *Ace Ventura: Pet Detective*

This is space. It's sometimes called the final frontier. (Except that of course you can't have a final frontier, because there'd be nothing for it to be a frontier to, but as frontiers go, it's pretty penultimate...)
—Terry Pratchett

I can picture in my mind a world without war, a world without hate. And I can picture us attacking that world, because they'd never expect it.—"Deep Thoughts" with Jack Handey

Fantasy is the impossible made probable. Science fiction is the improbable made possible.—Rod Sterling, *The Twilight Zone*

There's that word again, "heavy". Why are things so heavy in the future? Is there a problem with the earth's gravitational pull?—Emmet Brown, *Back To The Future*

LinuxJournal.com

This month, we bring you the results of 2011's *Linux Journal* Readers' Choice Awards. Every year, the awards give our readers the opportunity to support their favorite open-source software, development tools, languages, hardware and more.

It also gives readers an opportunity to discover new and popular technologies they may have not yet explored. To that end, I encourage you to check out some of this year's winners at LinuxJournal.com.

You consistently demonstrate your love for Python. These articles are a great place to read more:

- "Python for Android": **http://www.linuxjournal.com/article/10940**.
- "Python Programming for Beginners": **http://www.linuxjournal.com/article/3946**.
- "Tech Tip: Really Simple HTTP Server with Python": **http://www.linuxjournal.com/content/tech-tip-really-simple-http-server-python**.

Interested in Git for version control? Try: "Git—Revision Control Perfected": **http://www.linuxjournal.com/content/git-revision-control-perfected**.

If you're into Puppet, "Automate System Administration Tasks with Puppet" is an oldie but goodie: **http://www.linuxjournal.com/magazine/automate-system-administration-tasks-puppet**.

If HTML5 is your thing (and why wouldn't it be?), we have several great articles for you. Start with "Augmented Reality with HTML5" at **http://www.linuxjournal.com/article/10920**, and then do a quick search for HTML5 to pull up several gems.

You seem far more enthusiastic about GNOME 3 than our on-line columnist Michael Reed, so please feel free to discuss it with him in the comments to his article "GNOME 3.2 More Evolution than Revolution" at **http://www.linuxjournal.com/content/gnome-32-more-evolution-revolution**. Be nice though. We like Michael!

—KATHERINE DRUCKMAN

Basic Chemistry on the GNOME Desktop

I've realized I've missed out on a huge area of computational science—chemistry. Many packages exist for doing chemistry on your desktop. This article looks at a general tool called avogadro. It can do computations of energy and gradient values. Additionally, it can do analysis of molecular systems, interface to GAMESS and import and export from and to several file formats. There also are lots of options for generating pretty pictures of your totally new molecule that you hope will revolutionize the chemical industry.

When you first start avogadro, it opens up a new project consisting of one tab for a view of the molecule. Within this window, you can create a molecule in several different ways. If you want, you can build your molecule step by step, one atom at a time. You can click on the pencil at the top or press F8 to select the draw tool. With this, you can click and drag to draw atoms and bonds. To select different elements or different bond types, click on the Tool Settings button to get a second panel with those options available. To draw, click with the left mouse button. To erase, click with the right mouse button.

You also have the option of building it up from molecule fragments. Under the Build menu item, select Insert→Fragment.

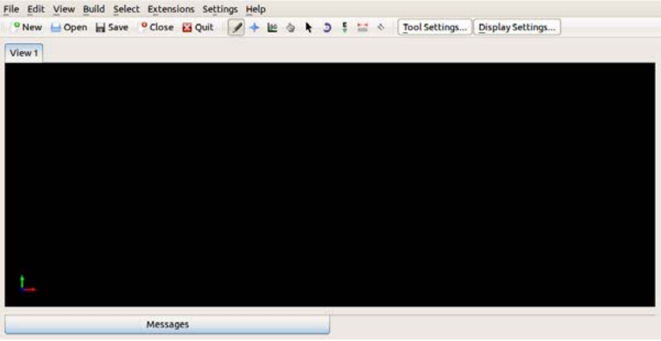


Figure 1. When you open avogadro, you get an empty view window in a new project.

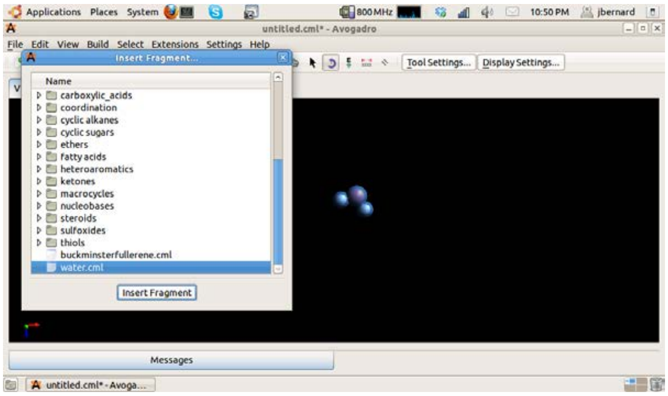


Figure 2. You can choose from a large list of chemical fragments in order to build your molecule.

This pops up a window where you can select from among molecule fragments, such as carboxylic acids, ketones, alkanes or amino acids. You even can build a buckminsterfullerene with a single click.

As you build up the molecule, it appears in the view window. You also can insert a fragment by entering a SMILES text string. SMILES is the Simplified

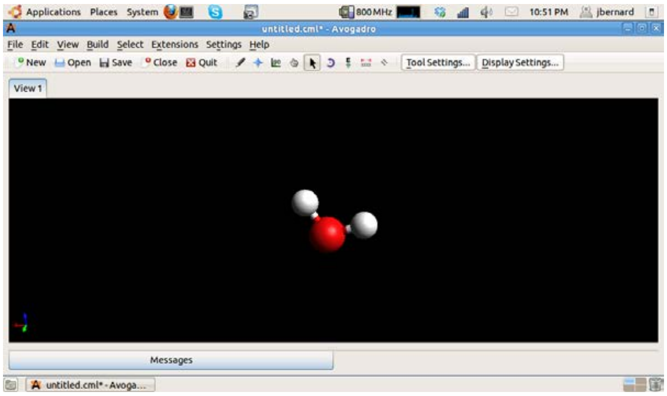


Figure 3. Here's what a water molecule would look like.

Molecular Input Line Entry Specification. Check the Wikipedia entry for more details on its format.

There is also a peptide builder under the menu entry Build→Insert→Peptide. This dialog window lets you build up a peptide by selecting the amino acids, the terminating fragments and the structure of the peptide. The available structures are straight chain, alpha helix and beta sheet.

You can make global adjustments to your molecule by adjusting the hydrogen bonds. You can do a global replacement of changing hydrogens to methyl groups. You also can change the number of hydrogen atoms based on the pH of the molecule.

You can save this molecule in several different formats, including CML, GAMESS Input, Gaussian Input, MDL SDfile, PDB, NWChem Input, Sybyl Mol2 and XYZ.

Once you are done building your molecule, one of the first things you may want to do is get an idea of what it actually looks like. As a first approximation, you can use the Optimize Geometry function, located under the Extensions menu item. This actually goes through your molecule and adjusts the bond lengths and the bond angles to optimize your molecule's physical geometry. You can select what force field is used to do this calculation. Under the menu item Extensions→Molecular Mechanics→Setup Force Field, a new window pops up letting you choose which force field you want to use, along with the number of steps to take, the algorithm to use (either Steepest Descent or Conjugate Gradients) and the convergence threshold. Depending on your molecule's complexity, some settings may work better than others. For example, the MMFF94(s) field is good for organic chemistry and drug-like molecules. If you try to optimize the geometry and get odd-looking results, checking these options should be your first step. You can see the calculation's progress by clicking on the Messages tab at the bottom of the window.

Once you have optimized your geometry, you can calculate your molecule's energy by selecting Extensions→Molecular Dynamics→Calculate Energy.

There is a limited amount that you can do directly in avogadro in terms of actual quantum-level calculations. But, that's okay, because lots of really

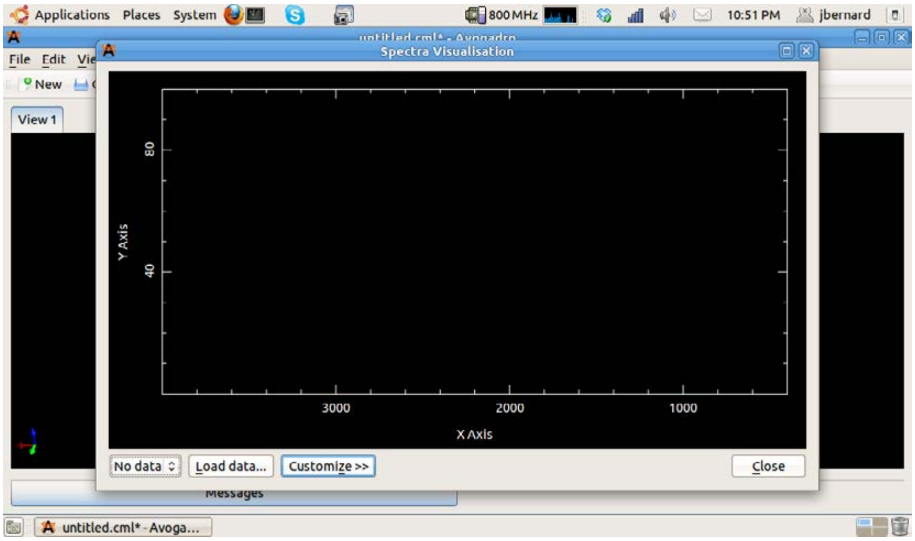


Figure 4. Here I'm ready to upload spectral data from actual experimental measurements.

good programs exist that already do that very well. You can use avogadro to output files that can be used by these other programs as input files to do such higher-level computations. Under the Extensions menu item, you will find entries to build input files for GAMESS, Gaussian, MOLPRO, MOPAC, NWChem and Q-Chem. Each of those entries pops open a new dialog window where you can select the extra options for the to-be-created input file. This includes things like the number of processors to use, the type of calculation to do or the theory to use, for example. A preview window shows you what this export file will look like, so you can be sure you're getting what you were expecting. Once you're happy, click on the generate button at the bottom of the window to generate the input file for the external program of interest.

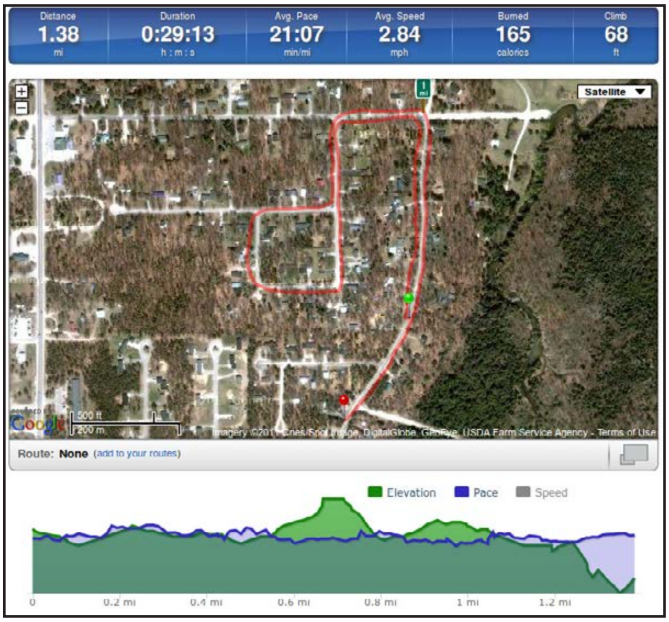
Once these calculations are done, you can import them back into avogadro to do some analysis. You can import trajectory files or vibration files. Once that data is imported, functions are available to graph this data in order to extract information and see what's happening. For files that contain molecular vibration information, you even can graph the results as a movie. You have some options in terms of the frame rate and so on, and once you are satisfied, you can save it as an AVI file.

You also can import data from experiments too. When studying chemicals, one common experiment is to look at the spectra of the chemical of interest. To start, click on the menu item Extensions→Spectra. This pops open a new window in which to do the analysis. At the bottom of the screen, you can click on Load Data. This lets you import data in two formats: either PWscf IR data or Turbomole IR data. You can select how this data is displayed, including producing publication-quality graphics.

Hopefully, this short introduction gives you some ideas for getting started. Lots of other chemistry programs exist that you can look at to offer more functions and calculations. Now you're ready to go out and do some quantum chemistry.

—JOEY BERNARD

Lowjack Your Body with RunKeeper



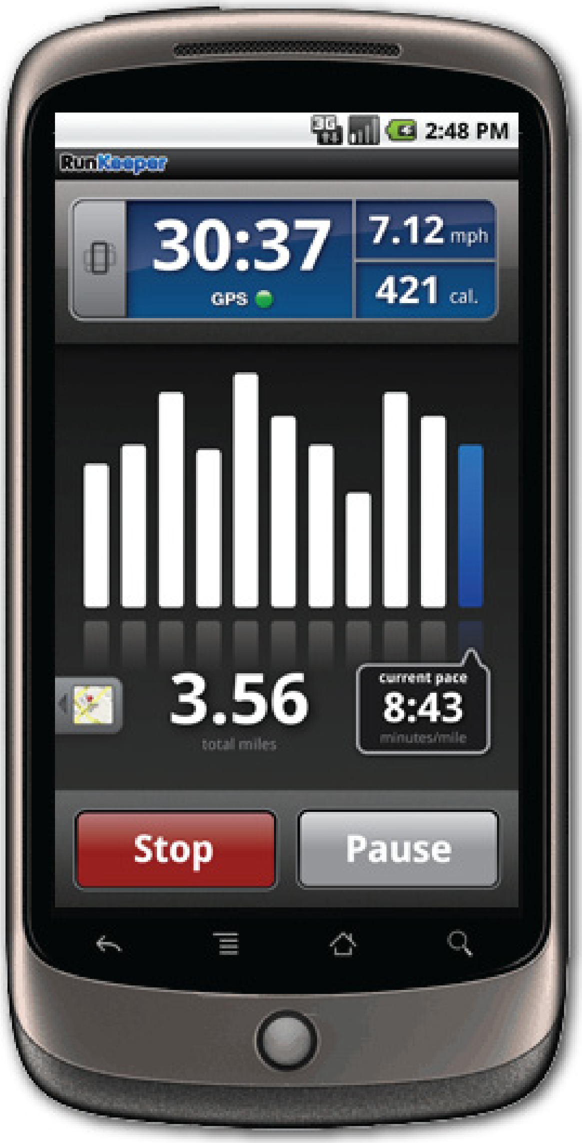
This past summer, I went to a beach resort in Mexico with my wife. It made sense to get into a little better shape so as not to cause any beached-whale rumors while I soaked in the rays. Typical geek that I am, I wanted to track my every move so I could see how much exercise I really was doing. And, I wanted to do that with technology.

Thankfully, RunKeeper is available for Android. RunKeeper is an exercise-tracking app that uses GPS to track your exercise. Thanks to geographical information over GPS, RunKeeper will track your distance, pace, time and even elevation. The free version provides lots of awesome features, and its social features also can help keep you accountable. (Although your Twitter followers might get tired of hearing about your daily walks to the park.)

Several other exercise apps are available, so if RunKeeper isn't your cup of tea, just search for "exercise" in the Marketplace, and you'll find a plethora of options. Keep in mind, however, that GPS-based exercise-tracking programs aren't much good in the northern winters, when running moves indoors to a treadmill.

Get the RunKeeper Android app at **http://www.runkeeper.com/android**.

—SHAWN POWERS



Non-Linux FOSS

We've covered the cross-platform video player VLC in past issues, but if you're an OS X user, it's often preferable to use QuickTime. Because it's built in to the operating system, QuickTime integrates with almost every aspect of the system. Unfortunately, QuickTime has limited playback support. Enter Perian. Perian is an open-source plugin for QuickTime that gives the native player the ability to play most popular video formats. The Perian Web site lists the following supported formats:

- File formats: AVI, DIVX, FLV, MKV, GVI, VP6 and VFW.
- Video types: MS-MPEG4 v1 and v2, DivX, 3ivx, H.264, Sorenson H.263, FLV/Sorenson Spark, FSV1, VP6, H263i, VP3, HuffYUV, FFVHuff, MPEG1 and MPEG2 video, Fraps, Snow, NuppelVideo, Techsmith Screen Capture and DosBox Capture.
- Audio types: Windows Media Audio v1 and v2, Flash ADPCM, Xiph Vorbis (in Matroska), MPEG Layer I and II Audio, True Audio, DTS Coherent Acoustics and Nellymoser ASAO.
- AVI support for AAC, AC3 Audio, H.264, MPEG4, VBR MP3 and more.
- Subtitle support for SSA/ASS, SRT and SAMI.

Is Perian better than VLC? No, it's not better or worse; it's different. If you want to play unsupported multimedia formats with Apple's QuickTime player, Perian can make that happen. If you'd rather use a completely open-source player application, VLC fits the bill. Either way, open source saves the day with the ability to play back just about any file you'd ever want to play. Check out Perian at **http://www.perian.org**.—SHAWN POWERS

REUVEN M. LERNER

Message Queues

Amazon's Simple Queue Service (SQS) provides an easy way to scale your Web applications.

This might come as a surprise, given that I have spent much of my professional career working with, writing on and teaching about the use of databases, but there was a period (mostly during and right after college) when I really didn't understand why people ever would need them. After all, I thought, you can just store and retrieve information in a file on your disk, no? My attitude back then demonstrated not only profound ignorance about databases themselves, but also about the types of problems people need to solve and the ways in which database technology had, even then, been developed to solve those problems.

Now I'm not quite as dismissive of technologies as I was back in my college days. But, it's true that although I've long heard of "message queues", it's been only in the last year or so, while working on a project, that I've come to realize just what a useful innovation they are. Sure enough, now that I understand how and why I would want to use them, I see uses for message queues everywhere.

In this article, I introduce the idea of message queues and give several examples of how you can install and use them. I also discuss why you might want to use a message queue, particularly on a Web application, which people typically think of as consisting only of an HTTP server and related software.

Message Queues

If you've been programming for any period of time, you know that a fundamental data structure is the queue, or FIFO (first in, first out). Like a queue at the post office, the first item stored also is the first item removed. Different queue implementations have different capabilities, but the general idea is that you put something in the queue and then retrieve it when it's ready. In Ruby (and many other languages, such as Python), you can implement a queue as follows:

```
class Queue
  def initialize
    @queue = [ ]
  end
  def enqueue(thing)
    @queue << thing
  end

  def dequeue
    @queue.shift
  end

  def to_s
    @queue.join(', ')
  end
end
```

You then can use it as follows:

```
>> require 'queue' #=> true
>> q = Queue.new #=>
>> q.enqueue('a') #=> ["a"]
>> q.enqueue('b') #=> ["a", "b"]
>> q.enqueue('c') #=> ["a", "b", "c"]
>> q.dequeue #=> "a"
```

Because queues in Ruby can hold any value, you don't need to worry about what will be on the queue. You just know that you can stick whatever you want on there, and that eventually you can retrieve it, in order.

(I should note that if you're actively programming in Ruby, I hope you're not really using a queue class like this, but that you're rather just using arrays, which support all the basic operations you're likely to need to work with simple queue data structures.)

Queues are great, but I'm sure you can already imagine all sorts of horrible scenarios if you were to use this simple one for something important, such as a list of bank transfers to execute. My banker's desk is full of piles of papers that she needs to handle, and she (presumably) works through them from top to bottom, dealing with each one in turn, but it would be pretty unforgivable for one or more of those papers to get lost. And, although it's easy to say that Ruby arrays are pretty stable, what happens if the power goes out? In such a case, the entire queue is lost, causing untold problems for the people who expected safe delivery.

The difference between the simple-minded queue I showed above and a true message queue is that the latter guarantees delivery of every message. This means that basically no matter what happens, you can be sure the message eventually will be delivered, despite power outages and other issues. But, message queues are even better than that. Not only do they guarantee delivery, but they also work quickly, allowing you to queue up a number of messages or actions that require attention, but for which you lack the resources to handle immediately.

For example, consider a Web application that is designed not to provide immediate feedback to users, but rather to receive and process information sent from other computers or mobile devices. This type of application typically doesn't require giving the user immediate feedback (other than an acknowledgement that data was received). All of the messages sent are of great importance (and should not be lost), but the number of messages can vary greatly from minute to minute, let alone from hour to hour. When the data is processed and eventually placed in the database, however, doesn't matter nearly as much.

There are more mundane examples as well. Consider a Web application that needs to send e-mail updates to people, such as from a calendaring application. If the application were to send e-mail each and every time an event were changed, the response time—or the number of server processes available to receive new incoming messages—might well suffer. Instead, the application can stick the mail-sending task on a message queue and then let a process on a separate computer retrieve the order and send the actual e-mail.

Offloading the retrieval of messages to a separate computer offers another performance advantage. It allows you to scale up the processing as necessary, by adding additional back-end computers. Given that a message queue is transactional (that is, all-or-nothing), you can have as many back-end machines retrieving from the queue as you want. You don't have to worry that the same message will be delivered twice or that two processes will have to fight over the retrieved data.

Amazon SQS

So, now that I've convinced you that you want to have a message queue, how do you go about using one? The first question is which one to choose. Many message queues exist, and each has its advantages and disadvantages. I've been using Amazon's Simple Queue Service on a project for the past number of months, and although it certainly has its downsides—it costs money, and it can take a bit of time for messages to percolate through the system—the advantages have been fairly clear, including Amazon's willingness to store messages for up to two weeks and its impressive uptime statistics. And, although I certainly could have set up my own message-queueing system, I've been working on other aspects of the project and appreciated that someone else, out there in "the cloud", was dealing with the various IT-related tasks associated with running a queue.

If you have used any of Amazon's previous cloud offerings, its queue service will not be a surprise. You need to have an Amazon account and sign up for a unique access key that will identify you to Amazon for identification and billing purposes. In addition to the access key, which you can think of as a user name, you also need a secret (akin to a password), which is sent to Amazon along with each request.

Once you have set yourself up with SQS, you need to connect to it, preferably (but not necessarily) using one of the many SQS client libraries that have been developed. Most of my work nowadays is in Ruby, and when I started my project, I found that the best-known Ruby gem for SQS access was from RightScale, in the "right_aws" package. I have been using this driver without any problems, but it's true that Amazon has since released its own drivers for Ruby. I hope to experiment with that driver in the near future and to compare it with the RightAws modules—although to be honest, I don't expect to see any significant differences.

If you're using another language, there almost certainly are libraries you can use as well. For the Python community, there are the boto packages. See Resources for more information.

By the way, it's true that SQS costs money. However, queueing systems exist to handle large quantities of data, which means they're going to charge you very little per message. How little? Well, according to the prices posted at the time of this writing, sending messages is absolutely free. Receiving messages is free for the first GB each month. After that, you pay nothing for the first GB, and then 12 US cents for each GB, up to 1TB. Now, Amazon does have a number of different server centers, and each might have its own pricing. Also, pricing is applicable only when going in or out of Amazon's server systems. This means if you're using a hosting solution, such as Heroku, which sits on Amazon servers, transfer to and from SQS is completely free. Actually, that isn't quite true—data transfers are free only if you stay within the same geographic server cluster. My point, however, is that for most people and projects, the pricing should not be an issue.

I'm using SQS for a Web application that is (by the time you read this, if all goes well) intended to receive JSON data from mobile devices, sent via an HTTP POST request. The JSON data then needs to be parsed and stuck into a relational database, but that doesn't need to happen right away. The architecture of the application, thus, consists of two separate parts. The main Web app receives the data and puts it onto the message queue with minimal parsing and validation. A separate Web app, running on a separate server, retrieves the JSON data, parses and validates it, and then puts it into the database. From the perspective of SQS, the fact that I'm using different servers really doesn't matter at all; as long as I connect to SQS with the right user name and password, and use the right queue name for sending and receiving, everything will be just fine.

Connecting to SQS

Before you can send to or receive from SQS, you first must connect to it. Since I'm using the right_aws gem for Ruby, I need to download and install that:

```
$ gem install right_aws
```

Note that because I'm using RVM, the Ruby version manager, I installed this gem as my own user. If I were installing it for the entire system, or if I were not using RVM, I would need to log in as root or use sudo to execute the command as root.

With the right_aws gem installed and in place, I now can use it to connect to the SQS server. Note that RightScale's gem provides access to several different APIs, including several different "generations" of SQS. I am using the second-generation API, via the RightAws::SqsGen2 class.

I've put my Amazon keys in a separate YAML-formatted configuration file, allowing me to change and update keys as necessary, as well as keep track of separate keys for different environments. I then read the configuration information into my program with the following line:

```
SQS_CONFIG = YAML.load_file("/Users/reuven/
➥Desktop/config.yml")['defaults']
```

The above takes each of the name-value pairs in the "defaults" section of my config.yml file and puts it into a hash named SQS_CONFIG. (Note that I've used all caps to indicate that this is a constant and should not be modified by other programmers unless they have a really, really good reason for doing so.)

I then can get a connection to SQS with the following code:

```
require 'right_aws'
sqs = RightAws::SqsGen2.new(SQS_CONFIG['aws_access_key_id'],
                            SQS_CONFIG['aws_secret_access_key'],
                            { :server => SQS_CONFIG['sqs_server'] }
                            )
```

As you can see from the above call, `Right::SqsGen2.new` takes three parameters: the AWS key, the AWS secret and a hash of options that help configure the queue object. The most important one to pass is the name of the SQS server you want to use. If you don't specify it, you'll get queue.amazonaws.com, but to be honest, I haven't really thought about it much since checking with Heroku (our hosting provider) about which server to use.

Once you're connected to SQS, you must create (or retrieve) a queue. You can think of a queue as a single array to which you can store or retrieve data, just as I did in my simple Queue class earlier in this article. The difference, of course, is that the actual data storage is happening across the network, on servers to which you have no direct access. You can have any number of queues, each with its own name, containing alphanumeric characters, hyphens and underscores. So, if you want to use a queue called "testq", just say:

```
sqs_queue = sqs.queue('testq')
```

This returns an instance of RightAws::SqsGen2::Queue, an object that represents an Amazon message queue. A number of methods are defined on this object, including creation (which I do via the above call, rather than directly), deletion (which will remove all of your data, so I really wouldn't suggest it unless you have to), and the sending and receiving of messages. You also can set the visibility timeout on this object, which tells Amazon how long a message should be invisible once it has been read, but before it has been deleted. You even can get the size of the message queue, using the `size` method.

Sending Messages

In my simple, non-distributed message queue example, you saw that new messages are added to the queue using an `enqueue` method, taking a single object as a parameter. The same is true in this case; if you want to send a message to the queue, you simply say:

```
my_message = 'hello!'
sqs_queue.send_message(my_message)
```

This will turn your string into an SQS message and send it to the queue. So long as the message is less than 64KB in size and is in text format (including JSON or XML), Amazon probably will accept it. (The RightScale gem claims to support messages only up to 8KB in size, just as Amazon used to do, but it's not clear to me whether the gem enforces these limits or if Amazon's updates are reflected by the gem's behavior.) Trying to send a message that's too long for Amazon's limits will result in an exception being thrown. There is an explicit list on the SQS FAQ page of which UTF-8 characters are acceptable in an SQS message.

One nice thing about SQS is that you can have any number of messages in a queue at a time; there is no defined limit. By default, messages are kept in a queue for four days, but you can configure that to be anywhere from one hour to two weeks.

Receiving Messages

So, you've sent a message to the message queue. How do you receive it? After going through the initial configuration, connection and queue creation/opening displayed above, you can retrieve the first waiting message on the queue with:

```
message = mothra_queue.receive
```

In RightScale's Ruby library, `message` is set to nil if no messages were available. Thus, before you can operate on the message, you must first check to ensure that it's non-nil.

Assuming that `message` is not nil, you can get contents by transforming the message into a string—in other words, by invoking `.to_s` on the message:

```
print message.to_s
```

When you retrieve a message, Amazon keeps a note of that and makes it invisible to other processes that might try to retrieve it. In other words, if you've queued a single message and then retrieve that message, other processes trying to retrieve from the queue will be told that no messages are available. However, this is true only for a short time. Once the visibility timeout has passed, the message is once again available to retrieving processes. So, in order to ensure that a message is not read twice, it must be deleted:

```
message.delete
```

Under most circumstances, you will want to retrieve and then delete a message almost right away.

Conclusion

If you're saying, "Well, that seems quite simple", you're right. Message queues are a dead-simple idea, particularly if you're familiar with queues as data structures. Distributed message queues can be quite difficult to get to work in a distributed and persistent way, but Amazon has done just that and makes its queue available for a very reasonable price, often ending up free for small organizations and sites.

The advantages that a distributed message queue can bring to the table are overwhelming though, particularly when you have tasks or pieces of data that are coming in too rapidly to handle, but which could be processed by a large number of back ends. It's easy to imagine a large number of back-end computers picking messages off and inserting them into a database, after parsing and checking them for validity. Indeed, that's what I'm doing on my current project, and it has been working like a charm.

Now, there are issues with Amazon's queues. For starters, they have longer latency than you would get with a local queue, and they also are sitting on third-party servers, which might not sit well with some companies. But for the most part, it has worked without a hitch and has become a core part of the infrastructure on my project. During the course of this work, I've started to find all sorts of uses for message queues, and I'm starting to incorporate them into other projects on which I work. The day may come when it's an exceptional project that doesn't use a message queue, rather than the other way around.■

Reuven M. Lerner is a longtime Web developer, architect and trainer. He is a PhD candidate in learning sciences at Northwestern University, researching the design and analysis of collaborative on-line communities. Reuven lives with his wife and three children in Modi'in, Israel.

Resources

The home page for Amazon's SQS is **http://aws.amazon.com/sqs**. This site, like all the other Amazon Web Services sites, has extensive documentation, tutorials and examples, as well as forums that let developers help one another. I've never needed to use the help, because the documentation always has been sufficient for my needs, but I've read through the forums on some occasions and have been impressed with the degree of both community involvement and official answers from Amazon's staff.

You can find, learn about and download RightScale's AWS-related gems from **http://rightaws.rubyforge.org**.

If you're a Python programmer, you can download code from the boto Project for AWS access from **http://code.google.com/p/boto**.

DON'T MULTITASK. MEGATASK.

Imagine what you can do with an expert workstation.

Take iteration, visualization, and simulation data to the next level of productivity with the supercharged performance of an expert workstation powered by the Intel® Xeon® processor 5600.

intel Xeon inside™

Powerful. Intelligent.

Deskside and portable workstations from Ace Computers feature the Intel® Xeon® processor 5600 series.

ACE
TECHNOLOGY WITH A VISION

www.acecomputers.com
Ace Computers
877-ACE-COMP (877-223-2667)

Inspiring creativity means using tools that don't disturb your train of thought. Ace presents the NEW LogiCAD™ 45525SQ workstation, which can handle the highest workloads possible, but does it at the ambient noise level of a regular desktop PC.

With the Ace® Raptor™ 6 portable workstation, engineers have the ability to adjust designs in the field directly. The Raptor™ 6 is equipped to meet the capabilities of professional desktop workstations but in a portable notebook-like form factor.

Ace® LogiCAD™ 45525SQ Super-quiet Workstation

- Dual Intel® Xeon® 5600 series processors with up to 384GB memory
- Super-quiet chassis—28dB maximum noise level—less than a desktop
- Up to 4 graphics cards for extreme CAD/graphical simulations

Starting at **$1,995.00**

Ace® Raptor™ 6 Portable Workstation with Built-in UPS

- Intel® Xeon® processor 5600 series
- Up to 4 hard drives with hardware RAID/up to 24GB DDR3 memory
- Up to 2 graphics cards for extreme CAD/graphical simulations

Starting at **$2,995.00**

DAVE TAYLOR

Playing with Twitter Stats

It's not easy to determine whether someone's worth following on Twitter, but Dave takes on the task with a shell script that extracts account stats for a given Twitter account, then calculates their follow value. He also explains the philosophy behind the project and finds that Twitter has some weirdnesses in its HTML that make parsing the results interesting.

So, you've been using Twitter since it was all about the fail whale and not about the corporate sponsorships and back-end analytics. Me too. The problem is, Twitter also has become even more crazy and hard to understand as it has gained its millions of followers and its utility ecosystem has expanded and contracted variously.

One thing that's always interested me though is whether there's a way to calculate a numeric value for given Twitter users based on both their visibility and engagement. How do you measure those? Visibility could be calculated simply by looking at how many followers someone has, but most Twitter users follow lots of random people, so that they can have lots of followers.

This behavior is based on what Dr Robert Cialdini calls the Principle of Reciprocity in his brilliant book *Influence*, wherein he observes that if someone does something for you, you feel an inherent obligation to return the favor. Think Hare Krishnas at the airport giving you a flower before they ask for a donation. Think of the self-appointed pundits and gurus telling you their rules of netiquette, or of your own reactions—"if this person's following me on Twitter, I should follow them back. It's only polite, after all."

The upside is that if you just look at how many followers someone has without also checking how many people they follow, you can be duped into thinking something along the lines of "25,000 followers? Impressive." without ever noticing that the person follows 30,000 people in turn.

One way to differentiate these different types of Twitter users, therefore, is to calculate the ratio of followers to following. That's half the calculation.

Engagement is trickier to calculate, but if you examine someone's Twitter stream, you can separate out broadcast messages from those that are either an at-reply (as in "@DaveTaylor nice column!") or a retweet.

It's another ratio. If the majority of tweets from someone are broadcast tweets, their level of engagement is low, whereas a Twitter user whose messages almost always are responses is high on the engagement scale.

One more criterion: gross numbers. How many followers does someone have overall? How many tweets has the user sent? An account with a high engagement but only seven tweets in the last six months is less interesting than one with lower engagement but an average of 20 tweets a day. Agreed?

So, how do we calculate these sorts of figures?

Understanding a Twitter Profile Page

Twitter offers up quite a bit of information for its public profiles (and just about every Twitter profile is public), including the key stats we want to start with: follower count and following count.

To get them, we don't even need to negotiate the OAUTH login. We can just use curl from the command line:

```
$ curl -s http://twitter.com/davetaylor |
    grep 'stats_count numeric'
  <span id="following_count" class="stats_count numeric">566 </span>
  <span id="follower_count" class="stats_count numeric">10,187 </span>
  <span id="lists_count" class="stats_count numeric">790 </span>
```

You can see that my Twitter account, @DaveTaylor, has 10,187 followers, while I'm following 566 people. The "list" figure suggests popularity too, but since most Twitter users I know eschew lists, let's just ignore that for now.

We'd also like to grab the raw tweet count to see if it's an account that actually has sent some tweets or is dormant. Examining the HTML closely reveals that although the previous items are put into the class `stats_count`, the number of tweets sent is put in a similar, but not quite identical, class called `stat_count`. Typo? Maybe. Meanwhile, it forces us to tweak our regular expression:

```
$ curl -s http://twitter.com/davetaylor |
  grep -E '(stats_count|stat_count)'
  <span id="following_count" class="stats_count numeric">566 </span>
  <span id="follower_count" class="stats_count numeric">10,187 </span>
  <span id="lists_count" class="stats_count numeric">790 </span>
    <li id="profile_tab"><a href="/DaveTaylor" accesskey="u">
  <span id="update_count" class="stat_count">30,055</span>
  <span>Tweets</span></a></li>
```

It's a bit ugly, but it's not much work to extract and reformat the data in a script. The challenge really is just to strip away

The challenge really is just to strip away all the HTML junk, because once we’ve used it to select the lines in question, we don’t actually need it anymore.

all the HTML junk, because once we’ve used it to select the lines in question, we don’t actually need it anymore.

My first attempt is this:

```
$ echo "<test me>hello<test 2>" | sed 's/<.*>/-/g'
-
```

That didn’t work. We want “hello” as the result, because we don’t want to lose the non-HTML values. Here’s my second try:

```
$ echo "<test me>hello<test 2>" | sed 's/<[^>]*>/-/'
-hello<test 2>
```

Aha! That’s what we need—a regular expression that basically says “< followed by as many characters as are present other than the ’>’ character”.

To strip all the HTML, simply make it a global search and replace by appending a “g” to the sed statement:

```
$ echo "<test me>hello<test 2>" | sed 's/<[^>]*>/-/g'
hello
```

That’s great. Now we can turn the mess of results into something hopefully a bit more useful:

```
curl -s http://twitter.com/davetaylor | grep -E
➥'(stats_count|stat_count)' | sed 's/<[^>]*>/ /g'
   566
   10,187
   790
       30,055  Tweets
```

We still need to get rid of those pesky commas, but that’s a small addition to the sed statement, right? Let’s use this instead: `sed 's/<[^>]*>/ /g;s/,//g'`.

The results are ready to be parsed:

```
566
10187
790
      30055   Tweets
```

That can be done with one of my favorite scripting commands, `cut`. The wrinkle, however, is that when we drop this into a shell script, the results are a bit surprising if we look at my @FilmBuzz movie news Twitter profile. First, the script snippet:

```
stats="$(curl -s $twitterurl/$username | grep -E
➥'(stats_count|stat_count)' | sed 's/<[^>]*>/ /g;s/,//g')"
echo $stats
```

And, the results:

```
$ ./tstats.sh filmbuzz

#side .stats a:hover span.stats_count
 ➥#side .stats a span.stats_count 1701
4529 303 13034 Tweets
```

Although this just impacts the field number of the `cut` command, it turns out to be more tricky than it first seems. I run the same tstats script against @DaveTaylor and look what happens:

```
$ ./tstats.sh davetaylor

566 10187 790 30055 Tweets
```

Different output. Jeez—there's always something.

Let's stop here with this small dilemma. My next article will pick up the parsing challenge and then proceed to calculating some numeric scores for Twitter users. Stay tuned!■

Dave Taylor has been hacking shell scripts for a really long time, 30 years. He's the author of the popular *Wicked Cool Shell Scripts* and can be found on Twitter as @DaveTaylor and more generally at http://www.DaveTaylorOnline.com.

KYLE RANKIN

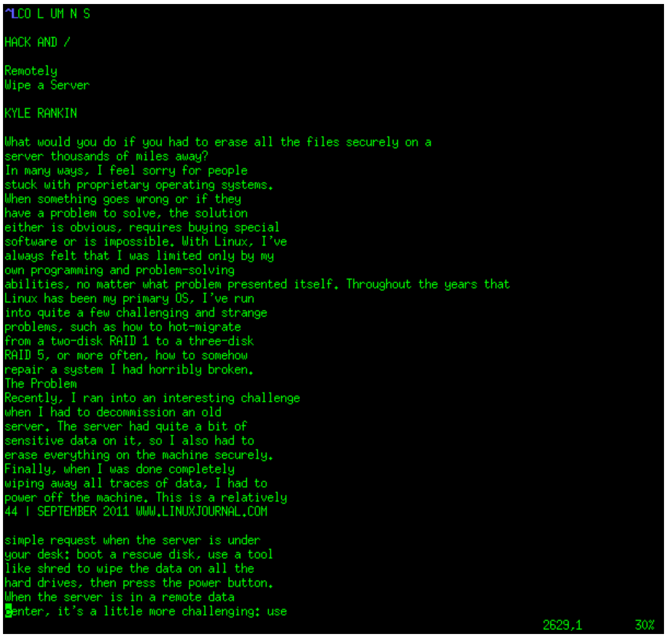
Read *Linux Journal* from the Command Line

Kindles? Nooks? It just takes your trusty command line and a few command-line tools to read *Linux Journal*.

In this day and age, there are more ways to read than ever before. Even though *Linux Journal* no longer publishes on paper, you still can read it with Web browsers, PDF software, e-book readers and cell phones. I don't have an e-book reader myself, but I think you could make the argument that the one true way to read *Linux Journal* is from the command line. After all, I read my e-mail, chat, check Twitter, do most of my day job and write my articles from the command line (okay, it's true I use gvim too; it frees up a terminal window), so why not read *Linux Journal* from the place where I spend most of my time?

The Text, the Whole Text and Nothing but the Text

The first format I'm going to cover is the Portable Document Format (PDF). Although PDFs are aimed at capturing a document so that it looks the same to everyone, it turns out you also can strip out the text and images from a PDF file. The first program I use for this is the aptly named pdftotext. This program is part of a group of PDF utilities that are packaged as the popper-utils package under Debian-based systems, but you should be able



```
^LCO L UM N S

HACK AND /

Remotely
Wipe a Server

KYLE RANKIN

What would you do if you had to erase all the files securely on a
server thousands of miles away?
In many ways, I feel sorry for people
stuck with proprietary operating systems.
When something goes wrong or if they
have a problem to solve, the solution
either is obvious, requires buying special
software or is impossible. With Linux, I've
always felt that I was limited only by my
own programming and problem-solving
abilities, no matter what problem presented itself. Throughout the years that
Linux has been my primary OS, I've run
into quite a few challenging and strange
problems, such as how to hot-migrate
from a two-disk RAID 1 to a three-disk
RAID 5, or more often, how to somehow
repair a system I had horribly broken.
The Problem
Recently, I ran into an interesting challenge
when I had to decommission an old
server. The server had quite a bit of
sensitive data on it, so I also had to
erase everything on the machine securely.
Finally, when I was done completely
wiping away all traces of data, I had to
power off the machine. This is a relatively
44 | SEPTEMBER 2011 WWW.LINUXJOURNAL.COM

simple request when the server is under
your desk: boot a rescue disk, use a tool
like shred to wipe the data on all the
hard drives, then press the power button.
When the server is in a remote data
center, it's a little more challenging: use
                                                                   2629,1        30%
```

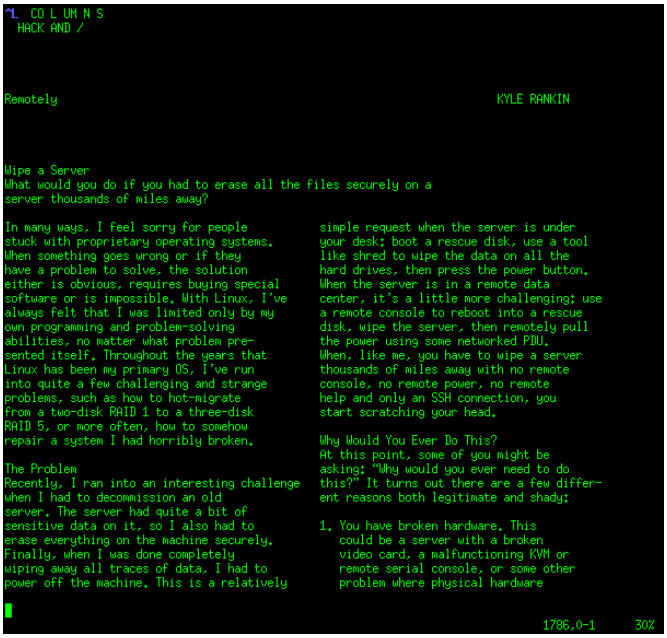
Figure 1. pdftotext's Default Output for My Column to find them under a similar name for your distributions. The most basic way to execute pdftotext is the following:

```
$ pdftotext input_document.pdf output_document.txt
```

By default, pdftotext does not attempt to preserve all the formatting of the document, which is nice because you don't have to scroll up and down multiple columns of a page. The downside is that it doesn't know to strip out all the extraneous text, headers, pull-quotes and other text you will find in a magazine article, so the result is a bit limited, as you can see in Figure 1.

Text Plus Columns

So although I suppose pdftotext's default output is readable, it's less than ideal. That's not to say I'm out of tricks though. Among its command-line options, it provides a `-layout` argument that attempts to preserve the original text layout. It's still not perfect, as you can see in Figure 2, but if you size your terminal so that it can fit a full page, it is rather readable.



```
^L  CO L UM N S
  HACK AND /



Remotely                                                                   KYLE RANKIN




Wipe a Server
What would you do if you had to erase all the files securely on a
server thousands of miles away?

In many ways, I feel sorry for people            simple request when the server is under
stuck with proprietary operating systems.        your desk: boot a rescue disk, use a tool
When something goes wrong or if they             like shred to wipe the data on all the
have a problem to solve, the solution            hard drives, then press the power button.
either is obvious, requires buying special       When the server is in a remote data
software or is impossible. With Linux, I've      center, it's a little more challenging: use
always felt that I was limited only by my        a remote console to reboot into a rescue
own programming and problem-solving              disk, wipe the server, then remotely pull
abilities, no matter what problem pre-           the power using some networked PDU.
sented itself. Throughout the years that         When, like me, you have to wipe a server
Linux has been my primary OS, I've run           thousands of miles away with no remote
into quite a few challenging and strange         console, no remote power, no remote
problems, such as how to hot-migrate             help and only an SSH connection, you
from a two-disk RAID 1 to a three-disk           start scratching your head.
RAID 5, or more often, how to somehow
repair a system I had horribly broken.           Why Would You Ever Do This?
                                                 At this point, some of you might be
The Problem                                      asking: "Why would you ever need to do
Recently, I ran into an interesting challenge    this?" It turns out there are a few differ-
when I had to decommission an old                ent reasons both legitimate and shady:
server. The server had quite a bit of
sensitive data on it, so I also had to           1. You have broken hardware. This
erase everything on the machine securely.           could be a server with a broken
Finally, when I was done completely                 video card, a malfunctioning KVM or
wiping away all traces of data, I had to            remote serial console, or some other
power off the machine. This is a relatively         problem where physical hardware

█
                                                                                  1786,0-1      30%
```

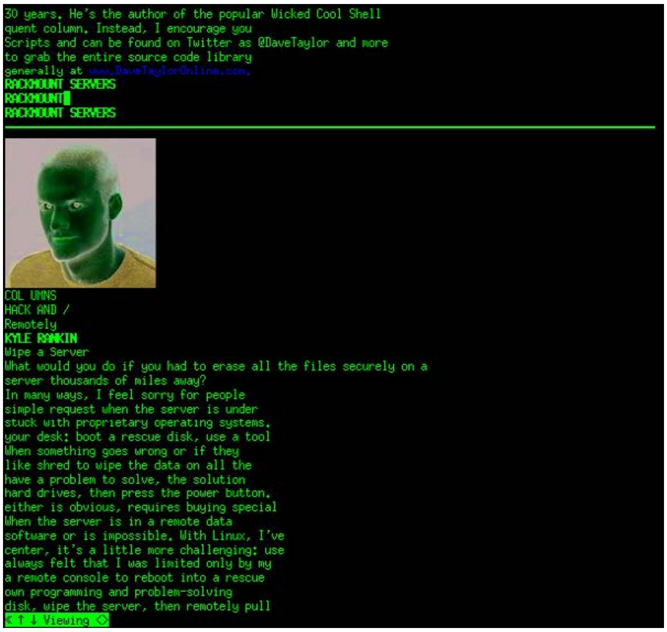
Figure 2. pdftotext with the Layout Preserved

Text Plus Images

There is a bit of a problem, you'll find, if you do read *Linux Journal* in text-only mode: there's no pictures! Although some articles still are educational in pure text, with others, it really helps to see a diagram, screenshot or some other graphical representation of what the writer is saying. You aren't without options, but this next choice is a bit of a hack. Because there are versions of the w3m command-line Web browser that can display images in a terminal (the w3m-img package on a Debian-based system provides it), what you can do is convert the PDF to HTML and then view the HTML with w3m. To do this, you use the pdftohtml program that came with the same package that provided pdftotext. This program creates a *lot* of files, so I recommend creating a new directory for your issue and `cd`-ing to it before you run the command. Here's an example of the steps to convert the September 2011 issue:

```
$ mkdir lj-2011-09
$ cd lj-2011-09
$ pdftohtml -noframes /path/to/linuxjournal201109-dl.pdf
 ➥lj-2011-09.html
```

Although it’s nice to see the images in a terminal, it would be better if everything was arranged so it made a bit more sense.



```
30 years. He’s the author of the popular Wicked Cool Shell
quent column. Instead, I encourage you
Scripts and can be found on Twitter as @DaveTaylor and more
to grab the entire source code library
generally at www.DaveTaylorOnline.com.
RACKMOUNT SERVERS
RACKMOUNT
RACKMOUNT SERVERS
```

```
COL UMNS
HACK AND /
Remotely
KYLE RANKIN
W1pe a Server
What would you do if you had to erase all the files securely on a
server thousands of miles away?
In many ways, I feel sorry for people
simple request when the server is under
stuck with proprietary operating systems.
your desk: boot a rescue disk, use a tool
When something goes wrong or if they
like shred to wipe the data on all the
have a problem to solve, the solution
hard drives, then press the power button.
either is obvious, requires buying special
When the server is in a remote data
software or is impossible. With Linux, I’ve
center, it’s a little more challenging: use
always felt that I was limited only by my
a remote console to reboot into a rescue
own programming and problem-solving
disk, wipe the server, then remotely pull
« ↑ ↓ Viewing <>
```

Figure 3. A More Negative Version of Me

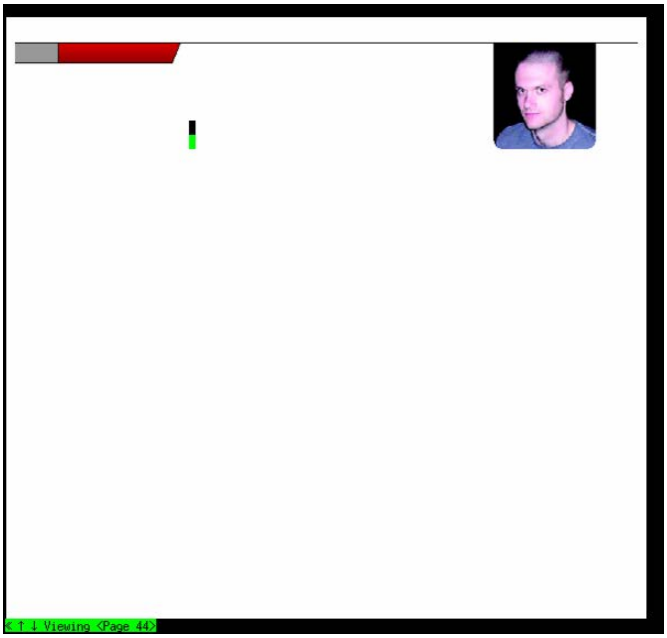
Once the command completes, you can run the w3m command against the lj-2011-09.html file, and if you have the special version that loads images, you will start to see the images load in the terminal. Now, by default, this output is much like the original output of pdftotext. There is no attempt to preserve formatting, so the output can be a bit of a mess to read. Also, as you can see in Figure 3, my headshot looks like a photo negative.

Text Plus Images Plus Columns

Although it’s nice to see the images in a terminal, it would be better if everything was arranged so it made a bit more sense. Like with pdftotext, pdftohtml has an option that attempts to preserve the layout. In the case of pdftohtml, you add the -c option:

```
$ mkdir lj-2011-09
$ cd lj-2011-09
$ pdftohtml -noframes -c /path/to/linuxjournal201109-dl.pdf
 ➥lj-2011-09.html
```

On the one hand, this command generates some really good-looking graphical pages. On the downside, it



```
« ↑ ↓ Viewing <Page 44>
```

Figure 4. It’s an improvement for image quality, but worse for readability.

looks like the images are displayed over the top of the text, and as you can see in Figure 4, there's a whole graphical section of my column with no text on it. As you scroll down the page, you still can read a good deal of the text, but it stands independent of the image. On the plus side, it no longer shows a negative headshot.

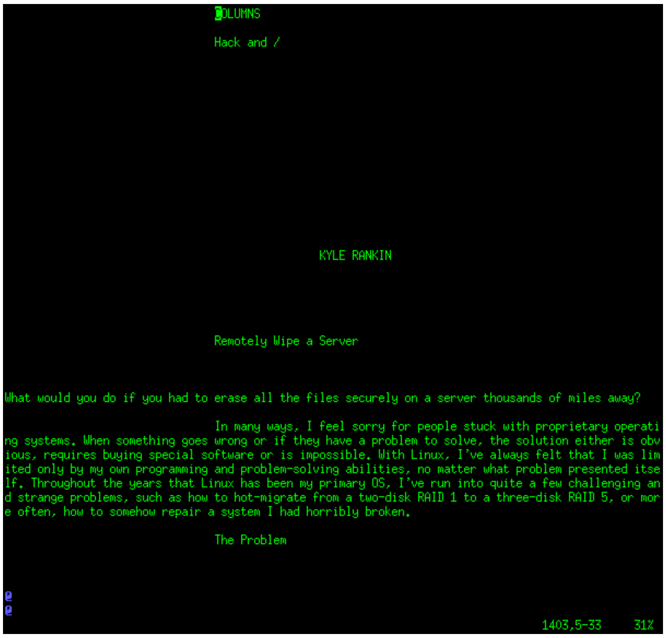
Go with the Reflow

So PDF conversion technically worked, but there definitely was room for improvement. As I thought about it, I realized that epub files work really well when it comes to reflowing text on a small screen. I figured that might be a better source file for my command-line output.

The tool I found best-suited to the job of converting epub files to text is Calibre. In my case, I just had to install a package of the same name, and I was provided with a suite of epub tools, including ebook-convert. Like with pdftotext, all you need to do is specify the input file and output file, and ebook-convert generates the output file in the format you want based on the file extension (.txt in this case). To create a basic text file, I would just type:

```
$ ebook-convert /path/to/LJ209-sept.epub LJ209-sept.txt
```

I found the resulting text file quite readable actually, although it did like to indent all of the headers and a lot of the rest of the text, so it started at the center of the terminal. That said, I would say that so far, it was the most readable of the output, as you can see in Figure 5.



```
                                   COLUMNS

                                   Hack and /







                                                    KYLE RANKIN





                                   Remotely Wipe a Server



What would you do if you had to erase all the files securely on a server thousands of miles away?

                                   In many ways, I feel sorry for people stuck with proprietary operati
ng systems. When something goes wrong or if they have a problem to solve, the solution either is obv
ious, requires buying special software or is impossible. With Linux, I've always felt that I was lim
ited only by my own programming and problem-solving abilities, no matter what problem presented itse
lf. Throughout the years that Linux has been my primary OS, I've run into quite a few challenging an
d strange problems, such as how to hot-migrate from a two-disk RAID 1 to a three-disk RAID 5, or mor
e often, how to somehow repair a system I had horribly broken.

                                   The Problem



@
@
                                                                                 1403,5-33      31%
```

Figure 5. Even with Indents, a Quite Readable *LJ* Article

So, with all of those different ways to read *Linux Journal* from the command line, two methods stand out to me right now. If you don't need images, I think the epub-to-text conversion works the best, with the pdftotext that preserves layout coming in second. If you do need to see images though, it seems like your main choice either is to convert from PDF to HTML and then use w3m, or just use w3m to browse the *Linux Journal* archives directly.■

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

Fluendo's Codec Pack

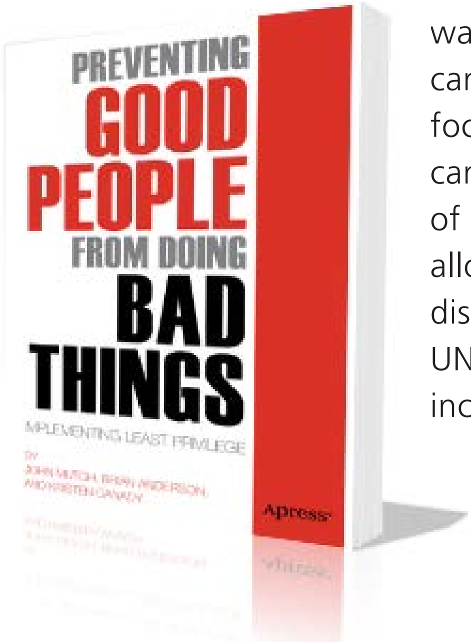
Fluendo, the folks whose mission is to make everything run on Linux, recently announced Codec Pack Release 15. In addition to the existing Windows Media, MPEG, DivX and other decoders, the new version 15 now offers support for AMD GPUs through the XvBA using OpenGL for efficient video rendering. This new addition follows on the heels of recently added support for VDPAU (NVIDIA) and VAAPI (Intel). Fluendo says its products are utilized by most OEMs for devices like desktops, laptops, thin clients, tablets, digital signage, set-top boxes, connected TV and media centers. Also, Fluendo solutions have been used by companies who've adopted Linux internally, allowing them to provide their employees with complete multimedia capabilities while remaining in full compliance with laws and patents.

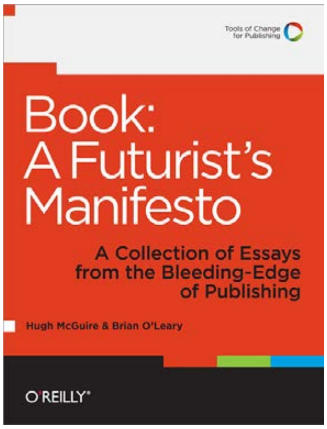
http://www.fluendo.com

Anderson and Mutch's *Preventing Good People From Doing Bad Things* (Apress)

"Good fences make good neighbors" is the old adage behind Brian Anderson and John Mutch's new security book, *Preventing Good People From Doing Bad Things: Implementing Least Privilege*. Many corporations have had to learn the hard way that they can have the slickest security software money can buy, while the greatest threats lurk from within. The main focus of the book, published by Apress, is to show how firms can remove these internal weaknesses by applying the concept of least privilege. The authors point out the implications of allowing users to run with unlimited administrator rights and discuss the implications when using Microsoft's Group Policy, UNIX and Linux servers, databases and other apps. Other topics include virtual environments, compliance and a cost-benefit analysis of least privilege. Auditors, geeks and suits will all find useful information in this book.

http://www.apress.com




Tools of Change for Publishing
Book: A Futurist's Manifesto
A Collection of Essays from the Bleeding-Edge of Publishing
Hugh McGuire & Brian O'Leary
O'REILLY

McGuire and O'Leary's *Book: A Futurist's Manifesto* (O'Reilly Media)

Linux Journal's own complete migration from the Gutenberg realm to the digital one is a fine case study on the powerful trends in publishing that are explored in a novel new project titled *Book: A Futurist's Manifesto* by Hugh McGuire and Brian O'Leary. The core content of this O'Reilly project is a collection of essays from thought leaders and practitioners on the developments occurring in the wake of the digital publishing shake-up brought on by the Kindle, iPhone and their kindred devices. The essays explore the new tools that are rapidly transforming how content is created, managed and distributed; the critical role that metadata plays in making book content discoverable in an era of abundance; the publishing projects that are at the bleeding edge of this digital revolution and how some digital books can evolve moment to moment, based on reader feedback. This particular project will do just that, incorporating reader feedback as the book is produced in hybrid digital-print format and determining in what ways the project will develop.

http://www.pressbooks.com and http://www.oreilly.com

Queplix's QueCloud FREEMIUM Version

Queplix recently announced the availability of the FREEMIUM version of QueCloud, a product the company bills as "the first data management cloud [that] enables companies to securely integrate cloud applications such as Netsuite, Google and SAP with unprecedented speed and simplicity". The benefits, says QueCloud, include an 80%+ TOC reduction vs. traditional ETL data management tools. QueCloud's core technology is Queplix's flagship Virtual Data Manager architecture, which enables the configuration of a series of intelligent application software blades that identify and extract key data and associated security information from many different target applications. The blades identify and extract key metadata and associated security information from the data stored within these applications, then bring it into the Queplix Engine to support data integration with other applications. The FREEMIUM version provides free access to users, developers and independent software vendors.

http://www.queplix.com

Arkeia's Backup Appliances

The new third generation of Arkeia Backup Appliances promises to "reduce costs with larger disk capacities, embedded SSD, and deduplication" says the well-known provider of backup and disaster recovery products. Deduplication delivers effective storage that is many times the capacity of internal drives. The company also touts new features, such as RAID-6, faster network connectivity, fully integrated and optimized Arkeia Network Backup v9 software, integrated bare-metal disaster recovery and support for VMware vSphere virtual environments. Target customers for the appliances are mid-sized companies or remote offices.

http://www.arkeia.com

OpenStack Diablo

The new Diablo (4th) release of the OpenStack cloud-computing platform recently went live, containing enhancements across the three existing projects: OpenStack Compute, Object Storage and Image Service. Key new features include new networking capabilities, such as "networking as a service" and unified authentication across all projects. Diablo extends existing API support and accelerates networking and scalability features, allowing enterprises and service providers to deploy and manage OpenStack clouds with larger performance standards and with ease. Two new projects for the next OpenStack software release, "Essex", have been initiated, currently code-named Quantum and Keystone.

http://www.openstack.org

eyeOS Professional Edition

eyeOS bills itself as "the most-advanced Web desktop in the world and the largest open-source software project in Spain". The new eyeOS Professional Edition takes the original edition to a new level, enabling private clouds that are accessible via any browser or device. All employees' and customers' workspaces can be virtualized in the cloud. The company says that the solution is highly scalable, easy to manage, does not require a large investment and includes all types of applications, including virtualized ones. Because eyeOS is developed in PHP and JavaScript and compiled in Hip-Hop, the system is not only high-performance, but also no software needs to be installed on the computer in order to work with it. Furthermore, the company says that eyeOS can integrate existing SaaS, virtualized legacy applications and other in-house apps served up as Web services.
http://www.eyeos.org

GrammaTech's CodeSonar

Kill programming bugs dead with CodeSonar, GrammaTech's static-analysis tool that performs a whole-program, interprocedural analysis on code and identifies complex programming bugs. The breakthrough feature in this release relates to the new program-analysis algorithms that identify data races and other serious concurrency defects. The process involves symbolic execution techniques to reason about many possible execution paths and interleavings simultaneously. The concurrency analysis can be applied to multithreaded software written for both single-core and multicore architectures. Another new feature, code-level metrics, is built on CodeSonar's existing code-analysis and reporting framework, which enables project managers to track popular metrics, such as cyclomatic complexity, or even define new metrics. Warnings can be generated automatically when metrics are outside an expected range. CodeSonar runs on Linux, Windows, Solaris and Mac OS and supports most compilers.
http://www.grammatech.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.

Fresh from the Labs

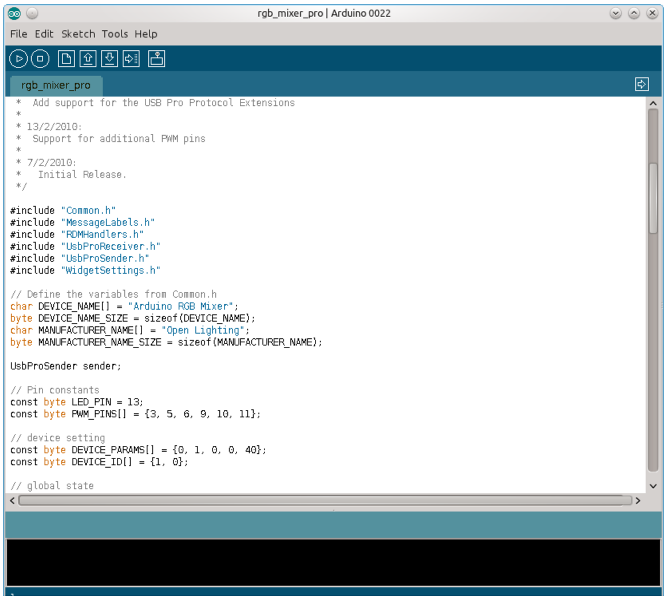
Arduino—Open Hardware and IDE Combo

http://arduino.cc/en

This article is a bit different from my usual column in two ways. First, it's starting with a hardware and software combo—something I've not done before. Second, the projects are linked to each other and come recommended to me by Perth LUG member, Simon Newton.

Given the mostly hardware-based information for the project, here are some carefully selected bits of information from the Web site:

> Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments.
>
> Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors and other actuators. The microcontroller on the board is programmed using the Arduino programming language and the Arduino development environment. Arduino projects can be standalone or they can communicate with software running on a computer.
>
> The open-source Arduino environment makes it easy to write code and upload it to the I/O board. It runs on Windows, Mac OS X and Linux. The environment is written in Java and based on Processing, avr-gcc and other open-source software.



```
 *  Add support for the USB Pro Protocol Extensions
 *
 * 13/2/2010:
 *  Support for additional PWM pins
 *
 * 7/2/2010:
 *   Initial Release.
 */

#include "Common.h"
#include "MessageLabels.h"
#include "RDMHandlers.h"
#include "UsbProReceiver.h"
#include "UsbProSender.h"
#include "WidgetSettings.h"

// Define the variables from Common.h
char DEVICE_NAME[] = "Arduino RGB Mixer";
byte DEVICE_NAME_SIZE = sizeof(DEVICE_NAME);
char MANUFACTURER_NAME[] = "Open Lighting";
byte MANUFACTURER_NAME_SIZE = sizeof(MANUFACTURER_NAME);

UsbProSender sender;

// Pin constants
const byte LED_PIN = 13;
const byte PWM_PINS[] = {3, 5, 6, 9, 10, 11};

// device setting
const byte DEVICE_PARAMS[] = {0, 1, 0, 0, 40};
const byte DEVICE_ID[] = {1, 0};

// global state
```

Arduino: this hardware/software combo allows you to program chips and test them on the fly—a real advantage of open hardware.

Installation For those who are happy with a binary, the Web site makes things very easy with 32- and 64-bit binary tarballs at the download page, and if you're lucky, the Arduino IDE may even be in your repository. If you're going with the binary tarball, just download the

One of the wacky creations possible with Arduino, this spider-like robot is made by Curtin University student, Phillip Lawrence.

latest from the Web site, extract it, and open a terminal in the folder. To run the program, enter the command:

```
$ ./arduino
```

If you're running from source instead, instructions are available on the Web site with a link from the Downloads page, although I don't have the space here to cover its somewhat unusual installer method. Nevertheless, it does recommend a series of packages that should help troubleshoot mishaps with both the source and binary tarballs. The Web site says you need the following: Sun Java SDK, avr-gcc, avr-g++, avr-libc, make, ant and git.

If you have a local repository version installed, chances are the program can be started with this command:

```
$ arduino
```

Under my Kubuntu installation, the Arduino IDE was available in the KDE menu under Applications→Electronics→Arduino IDE.

However, I must stop you here before actually running the program, and I apologize if I led you astray in the last few paragraphs (don't worry if you've already started it, you can close and re-open it with no worries). Obviously, before you can do anything with an Arduino board and the software, you first have to plug in your Arduino device. This will help in the configuration of your hardware, especially if you're using a USB connection.

Once that's out of the way, you now can start the program with any of the methods above.

Usage With the program running and the device plugged in, let's set it up. Inside the main window, click on the Tools menu and navigate your way to the Board menu. From there, choose your Arduino device (I had the Arduino Uno). Now you have to choose your serial port, which is under Tools→Serial Port. If you had a USB device and the program found it, a USB option should appear here (in my case, /dev/ttyUSB0).

With all of that boring stuff out of the way, I'm sure you're keen to sink your teeth into this hardware/software combo. The IDE makes things simple with a series of examples in an easy-access menu. Look under File→Examples, and check out the impressive list of examples from which to choose. I recommend starting with Blink under the 1.Basics menu.

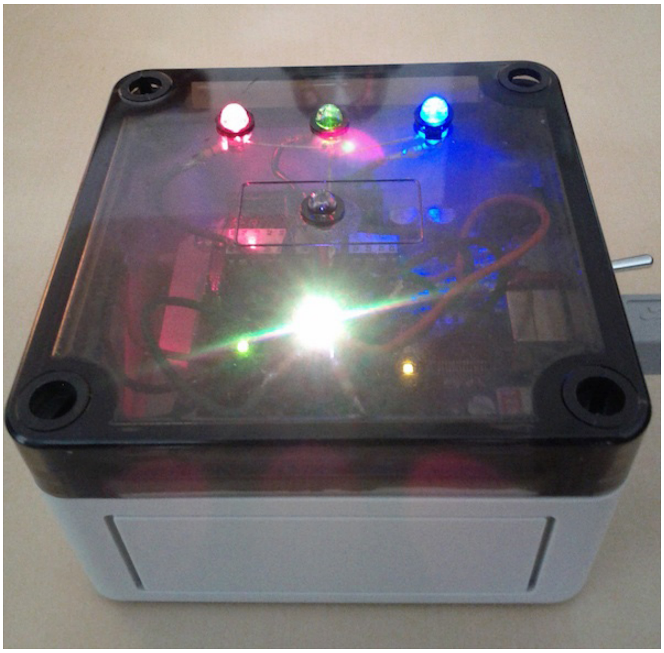
With Blink, you can start with the most basic of basics and come to grasp the syntax with well laid-out code including documentation for each line. To try out this code, click Upload, which is the sixth button along in the blue menu, with the right-facing arrow. If all goes well, you should see your device start blinking from an LED, perhaps with a board reset in the process.

If your board has an enabled reset facility like the Uno I was using, you should be able to make code changes by uploading them, watching the board next to you reset and start again with the new program. In fact, I recommend you try it now. Change one of the lines, perhaps one of the lines dealing with the delay time, and then upload it again. Now this may seem lame, but to a hardware "n00b" like myself, changing around the program and updating the running hardware in a visible way was quite a buzz!

If you want to check out your code before uploading it, the start and stop buttons are for verifying the code, with the stop button obviously allowing you to cancel any compiling partway through. Although I'm running out of space for the software side, I recommend checking out more of the examples in the code, where genuinely real-world uses are available. Some highlights include ChatServer, "a simple server that distributes any incoming messages to all connected clients"; a reader for barometric pressure sensors; and a program for demonstrating and controlling sprite animations.

However, I've been neglecting one of Arduino's real bonuses, and that is the ability to use a board to program any number of chips, remove them from the main Arduino board, and use them to run external devices. The nature of open hardware really makes this a robotic enthusiast's wet dream, with examples like my close mate Phil's robotic spider showing some of the cool things you can achieve with this suite.

Nevertheless, I do have one specific use of Arduino in mind to tie this column together, and that is Simon Newton's Arduino RGB Mixer: a six-channel color mixer that interfaces with OLA. Check out the following link for instructions on how to make this simple device that



Simon Newton's RGB Mixer is a great way to use both Arduino and OLA together.

shows off both of these projects at the same time: **http://www.opendmx.net/index.php/Arduino_RGB_Mixer**.

OLA—Open Lighting Architecture

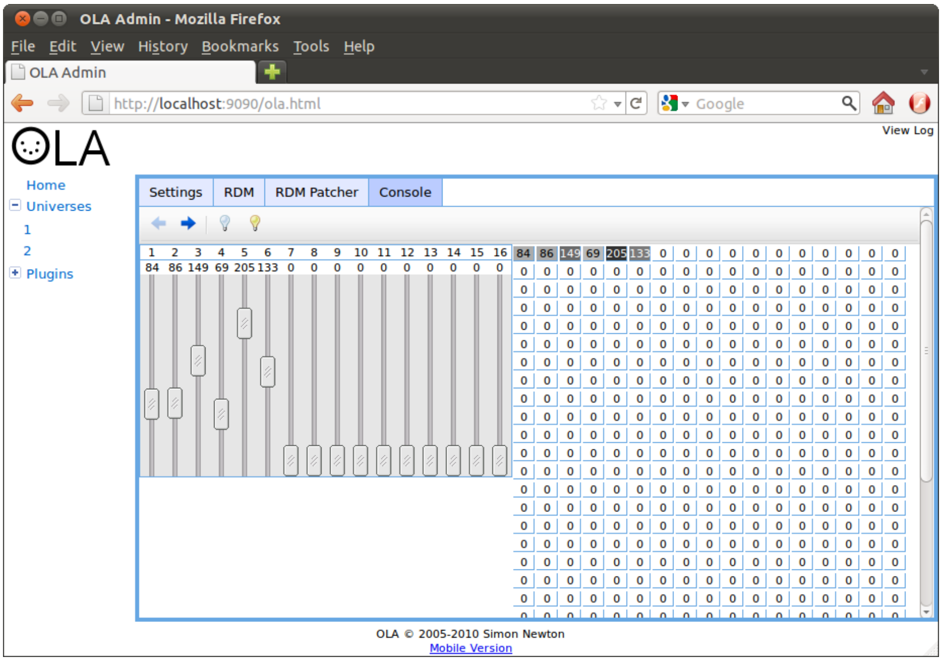
http://www.opendmx.net/index.php/OLA

Whether you're into concert lighting, flashy display stalls or Christmas lighting, or maybe you take mood lighting a little too seriously, you'll want to check out OLA. Combined with Simon's RGB Mixer, hopefully we can explore this project with relative ease.

According to the Web site:

> The Open Lighting Architecture (OLA) provides applications with a mechanism to send and receive DMX512 and RDM commands using hardware devices and DMX over IP protocols. This enables software lighting controllers to communicate with hardware either via Ethernet or traditional DMX512 networks.
>
> OLA also provides C++ and Python APIs so it's easy to build your own custom lighting control software. The Web site contains documentation on the APIs and there are code examples provided in the repo (./examples/ for C++ and ./python/examples/ for Python).

OLA supports some of the newest lighting control protocols, including Remote Device Management or RDM. Like the name suggests, this allows lighting devices to be remotely configured, and information like temperature, fan speeds, power consumption and so on to be fed back to the lighting controller. The Arduino RGB Mixer implements basic RDM functionality allowing the user to set the DMX Start address and invert the output signals in order to support different types of LEDs. With the addition of a temperature sensor, it can report this information back to OLA.



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
84	86	149	69	205	133	0	0	0	0	0	0	0	0	0	0

84	86	149	69	205	133	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OLA © 2005-2010 Simon Newton
Mobile Version

Inside the OLA Web interface, these sliders will let you control your lights in real time.

One of the jaw-dropping creations using OLA: the Nixie Mixie Matrix @ Gare St. Sauveur de Lille (FR) by Artist Boris Petrovsky (http://petrovsky.de). Photo: stereomorph.net (http://stereomorph.net).

Installation Before I continue, I must forewarn you of a deviation from the official documentation. The official docs recommend you use a program called QLC; however, I simply couldn't satisfy the library dependencies. Because of the library issues, I had to stray from the recommended applications in the Linux how-to and use some recommendations, amusingly, from a Windows how-to—a how-to that consisted of using VMware to run Linux under Windows (perhaps one of the more bizarre troubleshooting methods I've employed). And, may I also give a big thanks to the authors of this guide, most of which will be guiding the following process.

Anyway, this guide still uses Linux in the end, and it gave the following helpful command for installing the needed libraries:

```
$ sudo apt-get install libcppunit-dev libcppunit-1.12-1
➥uuid-dev pkg-config libncurses5-dev git libtool
➥autoconf automake g++ libmicrohttpd-dev libmicrohttpd5
➥protobuf-c-compiler libprotobuf-lite6 python-protobuf
➥libprotobuf-dev zlib1g-dev
```

Apologies for any "Ubuntuization" with that command, but the package names should point you in the right direction for your own distro, and if your distro doesn't use sudo, with any luck, you might be able to log in as root and simply drop the sudo from the start of the command. Depending on how your distro works, you may need to run `ldconfig` manually at this point (note, this requires root privileges to work properly).

From here, let's download the source code with git and compile it. To save time and space, I combine these steps into one stream of commands. From your console, enter the following:

```
$ git clone https://code.google.com/p/linux-lighting/ ola
$ cd ola
$ autoreconf -i
$ ./configure --enable-python-libs
$ make
$ make check
```

(Note: if you have errors after that last command, you still may be able to continue regardless.)

If your distro uses sudo:

```
$ sudo make install
$ sudo ldconfig
```

If your distro uses root:

```
$ su
(enter password)
# make install
# ldconfig
```

Although this obviously is a modest example of OLA's potential, if you look at the screenshots, there are impressive examples of using OLA on a much larger and dramatic scale.

Usage OLA uses a dæmon that various programs then can interact with, the easiest of which is a localized Web interface. To get started, most people should be able to get away with using this fairly simple command:

```
$ olad -l 3
```

If all goes well, your device should be named at the bottom of the text output (if not, see the basic documentation for more information on switches). Again, going with the best-case scenario, and what most users will be getting away with, let's now look at the Web interface for controlling the Arduino RGB Mixer.

In your favored Web browser, enter "localhost:9090" into your address bar. If all goes well, a local OLA Web page should appear with a redirected URL that resembles something like http://localhost:9090/ola.html.

To start interacting with your Arduino device, click the Add Universe button. This brings up a list of device ports, with "Open Lighting—RGB Mixer" most likely at the bottom. Check its box, and assign a number and name of your choosing for the Universe Id and Universe Name fields, respectively. And presto, you now can play with your device.

Click on the Console tab, and you'll see the vertical sliders. These turn the brightness of individual LEDs up and down in real time. Again, perhaps it's a bit lame, but it gave me the same thrill of interaction I experienced with the Arduino examples.

Although this obviously is a modest example of OLA's potential, if you look at the screenshots, there are impressive examples of using OLA on a much larger and dramatic scale. Hopefully, this is one of those projects that gains real development and maturity, becoming an underground hero. For a personal wish fulfillment, I'd like to see concert lighting become significantly cheaper, and if smaller independent bands can use this in particular to realize their artistic visions, OLA will make me a happy man.■

John Knight is a 27-year-old, drumming- and bass-obsessed maniac, studying Psychology at Edith Cowan University in Western Australia. He usually can be found playing a kick-drum far too much.

BREWING SOMETHING FRESH, INNOVATIVE OR MIND-BENDING? Send e-mail to newprojects@linuxjournal.com.

2011 Readers' Choice
LINUX JOURNAL

READERS' CHOICE AWARDS 2011

SEE HOW YOUR FAVORITES FARED IN THIS YEAR'S VOTE.

SHAWN POWERS

The votes are in, the tallies are counted, the hanging chads have been evaluated, and we have our winners. This year holds a few surprises, a couple dominant players and as much open source as you can handle. We don't encourage gambling here at *Linux Journal*, but if you had an office pool going for pizza money, it's officially too late to make your wager.

Best Linux Distribution

Ubuntu

Runner-up: Debian

apt, apt and more apt this year in the distribution category. Although it's no surprise that Ubuntu remains king of the distros, it's nice to see Debian, the "father" of Ubuntu gaining some ground. Whether it's because Linux Mint is making Debian more user-friendly or because folks are drawn to the appeal of Debian's stability, it got just about half the votes of all the Ubuntu variants combined. Way to go Debian! Oh, and of course, congratulations to the winner and still-champion, Ubuntu.

Best Distribution for Netbooks/Limited Hardware

Ubuntu Netbook Remix

Runner-up: Android and Debian (tie)

Although Ubuntu is streamlining its versions and making the desktop screen function similarly to the Netbook screen, Ubuntu Netbook Remix still garnered the most votes this year. Will the push to Unity make next year's Readers' Choice look a little different? The future awaits. Our runner-up last year was Android, and this year, Android is still our runner-up, but it shares the silver medal with Debian. Why is Debian getting so much attention this year? For the same reason soda-pop companies are releasing "throw-back" versions of their drinks with real sugar, sometimes the tried-and-true operating systems just taste a little sweeter.

Best Mobile OS

Android

Runner-up: MeeGo

With the death of Maemo, the abandonment of MeeGo and the discontinuation of webOS, the obvious winner this year is Android with 80% of the vote. MeeGo takes enough of the remaining vote to get our runner-up spot, but it's a bitter prize, as MeeGo's future looks pretty bleak. Will Android get another open-source competitor? Will the lack of open competition stifle Android innovation? Only time will tell. For current Linux-based handsets, however, Google's Android truly can say, "All your base are belong to us."

Best Desktop Environment

GNOME

Runner-up: KDE

Last year it was a tie. This year, our back-and-forth battle falls to GNOME as the best desktop environment. Due to the timing of the GNOME 3 release, it's hard to tell if the victory is because of version 3 or in spite of it. Nonetheless, GNOME ekes a victory with a 3% margin over KDE. The next-closest desktop environment is XFCE with less than one-third the votes of either of the big two. With such big contenders for first and second, however, that third-place spot is significant, and XFCE is gaining ground. We think it's one to keep an eye on next year.

Best Web Browser

Firefox

Runner-up: Chrome/Chromium

After its huge popularity spike last year, we thought the race for best browser would be neck and neck this year. The Firefox team stepped up its game, however, and this year made several major revisions. Although Chrome/Chromium is a major contender and even gained a few percentage points, Firefox still dominates with more than twice as many votes. As a Firefox user myself, it doesn't surprise me to see my favorite fiery fox on top, but it's hard to argue with Chrome/Chromium's lightning-fast response time. As more and more extensions are being ported to Google's browser, Firefox has some real competition. Hopefully, that competition will inspire greatness from both teams. As users, we can only benefit!

Best E-mail Client

Thunderbird

Runner-up: Gmail Web Client

Like its foxy-browser sibling, Thunderbird takes top spot again this year in the e-mail category. Now that Canonical has adopted Thunderbird as its default e-mail client in Ubuntu, we see the popularity rising for our blue-birdie friend. Still hanging on tightly to second place is Gmail. Is Gmail an app? We're not sure, but it certainly does get votes as the best e-mail client. Because Thunderbird can access Gmail servers easily, it's possible this category blurs the line a bit, as users simply can use both. When it comes to picking a favorite though, Thunderbird is the clear victor with more than twice as many votes as our runner-up.

Best IM Client

Pidgin

Runner-up: Skype

The results are similar, but the trend obviously is shifting in our Best IM Client category. Pidgin takes the number one spot with a full half of your votes, while Skype barely squeaks out a second-place win with 15%. Although its video chat is hard to beat, we think Skype lost some points due to its purchase by Microsoft. What does that new ownership mean for the future of Skype? No one knows for sure, but it has Linux users scrambling to find alternative video chat clients "just in case".

Best IRC Client

Pidgin

Runner-up: X-Chat

For years I've been touting the awesome IRC features Pidgin boasts. It's nice to see Pidgin take first place again as favorite IRC app. As my geek-fu has matured, so has my chatting preference, however, and I skipped right over our second-place IRC client X-Chat. I'm now using IRSSI for my textual communication needs. Although my preferences seldom represent those of the masses, I'll be shaking my IRSSI pom-poms next year for the awesome underdog. Credit where credit is due, however; it's hard to beat the flexibility of Pidgin and the huge feature set of X-Chat. It's clear why they are the Readers' Choice victors.

Best Microblogging Client

Gwibber

Runner-up: Choqok

Our top two microblogging clients from last year retain their status as class favorites. Gwibber and Choqok, GNOME- and KDE-native, respectively, garnered the most votes again this year. The ever-popular AIR application TweetDeck is right on their heels, however, and it's throwing in its cross-platform flexibility to make the contest interesting. Native clients (with odd names) still hold favor among readers, but those fancy Adobe AIR and HTML5 alternatives slowly are gaining ground. Who will win next year? We'll be sure to tweet the answer when the time comes.

Best Office Suite

LibreOffice

Runner-up: OpenOffice.org

The king has been dethroned! Well, sorta. Yes, technically the newcomer LibreOffice stomped on the former-champion OpenOffice.org. Because LibreOffice is a fork of OpenOffice.org, however, it seems like we should have an asterisk in there somewhere. Shortly after Oracle bought Sun Microsystems, OpenOffice.org was forked. The good news for users is that LibreOffice has a large dev crew, and updates and feature enhancements are coming out at a really nice rate. The king is dead; long live the king!

Best Single Office Program

OOWriter

Runner-up: AbiWord

We're not entirely clear if OOWriter is referring to the OpenOffice.org version or the LibreOffice version of the word-processing program. Basically, we're considering the winner, "the program that saves to .ODT by default", and we think that covers it. The runner-up again this year is not a member of the LibreOffice/OpenOffice.org suite, but rather the standalone AbiWord word processor. In fact, AbiWord is what this article is being typed on as we speak. Will the LibreOffice takeover change the favorite app category in the future? Based on voting this year, we guess not. It's nice to see our underdog-favorite AbiWord continue to get votes though.

Best Digital Photo Management Tool

digiKam

Runner-up: Picasa

The past few years have been an epic battle between these two programs. This year, we think the contestants might be a little tired, because although they still are clearly the top two choices, their popularity pretty much has leveled off. digiKam ekes out a victory by less than two percentage points, which means it's hard to go wrong when you pick one of these two. Whichever you choose, it's bound to be better than my solution: shoebox full of photos.

Best Graphics Design Tool

GIMP

Runner-up: Inkscape

GIMP kicks butt and takes names, as it scores two-thirds of the total votes this year. Inkscape remains in second place, but it's a very distant second. There certainly are other options available, but time and time again, we turn to GIMP for editing those photos. Although the learning curve for GIMP can be a bit steep, the same often is said for Linux itself. And like Linux, the reward is great.

Best Audio Tool

Audacity

Runner-up: Ardour

When I saw the Best Audio Tool category, my first instinct was to vote for speakers. I'm clearly in the minority, however, as 85% of you instantly thought of Audacity. Ardour is down another couple percentage points this year, but we still gave it runner-up status. We don't want Audacity to get too prideful after all.

Best Audio Player

Amarok

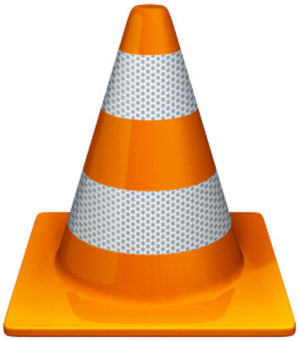
Runner-up: VLC

It's still clear readers love Amarok. In a surprisingly close second place this year is VLC. Although not normally thought of as an audio player, it does the job well enough to get just 7% fewer votes than Amarok. And last year's runner-up Rhythmbox? Sadly, it's far in the distance behind these two front-runners.

Best Media Player

VLC

Runner-up: MPlayer



Seeing VLC take runner-up in the audio player category makes this victory a no-brainer. VLC plays just about any sort of video you throw at it. I think if you shove a paper flip book into your floppy drive, VLC will animate it on-screen for you. VLC takes such a huge margin this year, we almost didn't include MPlayer as a runner-up. VLC is the favorite, without question. (Note: we don't actually recommend shoving a paper flip book into your floppy drive.)

Best Bookmark Sync Tool

Firefox Sync

Runner-up: Chrome Bookmarks

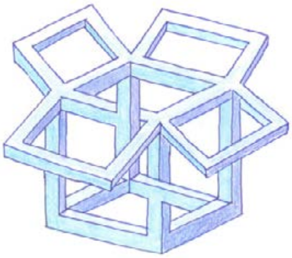
Our bookmark sync category completely rewrote history and gave us two brand-new winners. Firefox Sync, now built in to the browser, takes the victory handily with twice the votes of the runner-up, Chrome Bookmarks. This split makes absolute sense, because Firefox beat Chrome in the browser war by the same margin. In fact, if these numbers were different, it would cause our highly scientific voting process to look suspect. As it is, for Firefox users, we recommend Firefox Sync, and for Chrome users, we recommend Chrome Bookmarks. Feel free to call me Captain Obvious.

Best On-line Collaboration Tool

Google Docs

Runner-up: Wikis

We didn't title this "Most Popular Collaboration Tool", because as painful as it is, the majority of on-line collaboration tends to be e-mail messages with subjects like "RE:Fwd:Fwd:Re: This one Re: Final FWD: Final2" and ugly multi-fonted .doc files. Although popular, that's definitely not ideal. Google Docs takes the spoils of war again this year with its ever-improving feature set for on-line collaboration. It's even possible to watch as someone else edits a document. For everything else, wikis are still popular. Easy to edit and easy to maintain, wikis are a godsend for living documents.



Best Cloud-Based File Storage

Dropbox

Runner-up: Ubuntu One

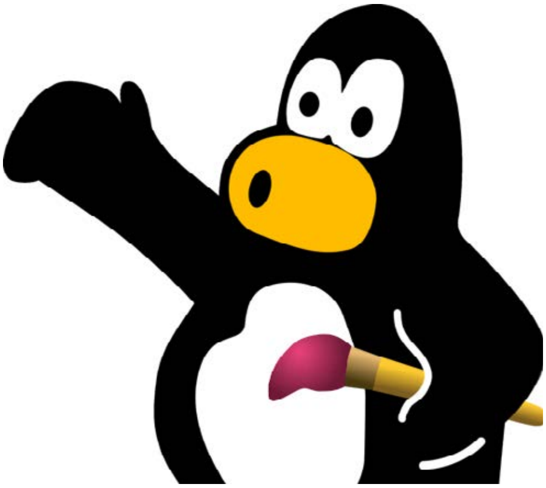
When it comes to cloud storage, it's hard to beat Dropbox. Although security is an often-touted concern with the cloud-storage behemoth, ease of use trumps those concerns. Ubuntu One is a distant second to the cross-platform, simple-to-use Dropbox. I'd put my Dropbox referral code here to get some free space, but I suspect our editor would frown on such a thing, plus you'd all likely flog me.

Best Kid-Friendly Application

Tux Paint

Runner-up: GCompris

Bill Kendrick's Tux Paint continues as the crowd favorite for 2011. Whether you're a kid of 5 or 50, it's hard not to smile when creating paintings with Bill's user-friendly application. GCompris is no slouch in the kid-friendly category either and quite nicely takes second place with its educational focus and lively graphics.



Best Game

World of Goo

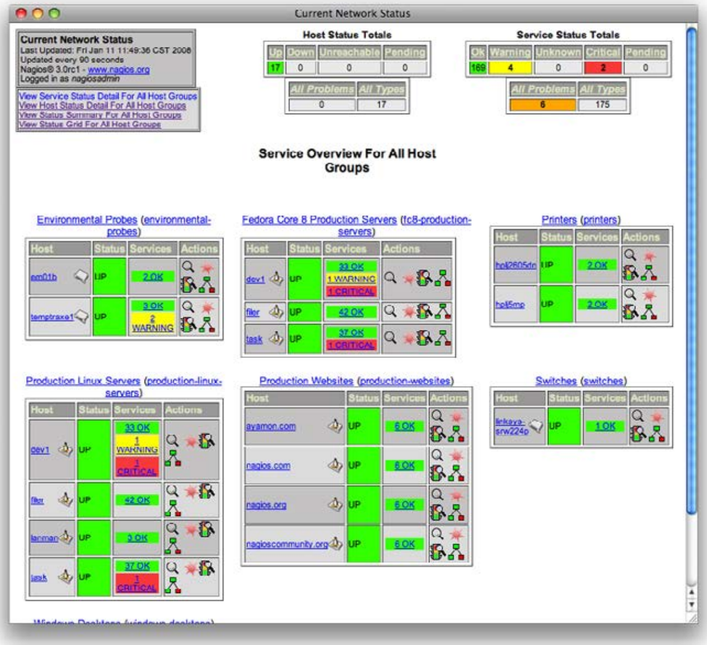
Runner-up: Battle for Wesnoth

For the first time in the history of histories, *Frozen Bubble* is not the most popular game! In fact, *Frozen Bubble* didn't even take second this year, as it lost to *Battle for Wesnoth* by half of a percentage point. (Normally we'd consider that a tie, but *Battle for Wesnoth* deserves recognition for bumping off the *Bubble*.) *World of Goo* is a game similar in addictiveness to *Frozen Bubble*, but with better graphics and more modern gameplay. If you're a casual gamer, check out *World of Goo*, it's really Goo-ood.

Best Monitoring Solution

Nagios

Runner-up: OpenNMS



Current Network Status

Current Network Status
Last Updated: Fri Jan 11 11:49:36 CST 2008
Updated every 90 seconds
Nagios® 3.0rc1 - www.nagios.org
Logged in as *nagiosadmin*

View Service Status Detail For All Host Groups
View Host Status Detail For All Host Groups
View Status Summary For All Host Groups
View Status Grid For All Host Groups

Host Status Totals

Up	Down	Unreachable	Pending
17	0	0	0

All Problems	*All Types*
0	17

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
169	4	0	2	0

All Problems	*All Types*
6	175

Service Overview For All Host Groups

Environmental Probes (environmental-probes)

Host	Status	Services	Actions
em01b	UP	2 OK	
temptraxe1	UP	3 OK 2 WARNING	

Fedora Core 8 Production Servers (fc8-production-servers)

Host	Status	Services	Actions
dev1	UP	33 OK 1 WARNING 1 CRITICAL	
filer	UP	42 OK	
task	UP	37 OK 1 CRITICAL	

Printers (printers)

Host	Status	Services	Actions
hplj2605dn	UP	2 OK	
hplj5mp	UP	2 OK	

Production Linux Servers (production-linux-servers)

Host	Status	Services	Actions
dev1	UP	33 OK 1 WARNING 1 CRITICAL	
filer	UP	42 OK	
lanman	UP	3 OK	
task	UP	37 OK 1 CRITICAL	

Production Websites (production-websites)

Host	Status	Services	Actions
ayamon.com	UP	6 OK	
nagios.com	UP	6 OK	
nagios.org	UP	6 OK	
nagioscommunity.org	UP	6 OK	

Switches (switches)

Host	Status	Services	Actions
linksys-srw224p	UP	1 OK	

Windows Desktops (windows-desktops)

I misspelled "monitoring" as I was typing this section, and AbiWord's recommendation for correction was "minotaur". Although I wouldn't argue a minotaur would be a wonderful monitoring solution for many circumstances, when it comes to computer hardware, Nagios is a little better, and far more popular. OpenNMS is a newcomer to our victory circle, and although it's far behind Nagios, it still scored quite well. "Minotaur", as it were, got very few votes.

Best Database

MySQL

Runner-up: PostgreSQL

It may not be the most-exciting topic around, but databases make the world go round. MySQL with its dolphin mascot takes first place again this year, with more than twice as many votes as its closest competition, PostgreSQL.

Best Backup Solution

rsync

Runner-up: tar

We geeks like our command line, and to back up stuff, nothing can beat rsync and tar. rsync is three times more popular than tar based on reader votes, but nothing else comes close when it comes to backup. For two years running, we don't need no stinkin' GUI!

Best Virtualization Solution

VirtualBox

Runner-up: VMware

Although the Oracle purchase certainly affected the OpenOffice.org popularity, VirtualBox's new ownership doesn't seem to bother anyone. VirtualBox is more than four times as popular as the distant runner-up, VMware. We even lumped the VMware options together, and they received only a cumulative 15% of the vote. VirtualBox beat *virtually* every other option hands down.

Best Revision Control System

Git

Runner-up: Subversion

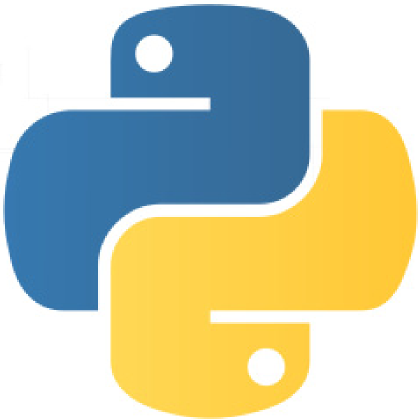
The Linus-Torvalds-created Git remains in the number one spot this year, as it widens the gap a bit more from the runner-up, Subversion. Either will do the job, but Subversion is becoming more and more the underdog. Perhaps having Linus on your side is an advantage in the Open Source world!

Best Open-Source Configuration Management Tool

Puppet

Runner-up: OpenQRM

In another repeat performance from last year, Puppet takes top spot for configuration management. If you administer greater than zero servers, you will benefit from using a tool like Puppet. If you're managing fewer than zero servers, well, we're not sure what that means. You probably need Puppet to manage your counting skills. Whatever your reason, configuration management is a hot topic, and Puppet is the hottest.



Best Programming Language

Python

Runner-up: C++

A three-time winner in our best programming category, Python continues to dominate. Close on its heels this year, however, is C++. In fact, a scant 6% separated the two. It's quite obvious, however, that our readers don't suffer from ophidiophobia in the least—*hiss*.

Best Scripting Language

Python

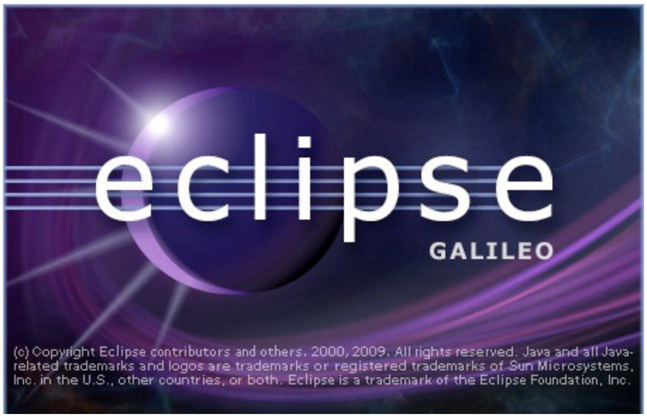
Runner-up: Bash

It hardly seems fair that Python gets both best programming *and* best scripting language, but I suppose excellence knows no bounds. A newcomer to our runner-up circle is Bash, the only language I can program with at all. Hats off to Python though, as it takes both categories again this year.

Best IDE

Eclipse

Runner-up: vim



(c) Copyright Eclipse contributors and others. 2000, 2009. All rights reserved. Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S., other countries, or both. Eclipse is a trademark of the Eclipse Foundation, Inc.

Admit it, you weren't surprised at all to see Eclipse in the top seat. Seeing vim in the copilot seat, however, was a nostalgic treat for me. Eclipse is incredibly extensible and remarkably quick in most environments. Nothing can beat the simplicity of vim, however, and it took only 9% fewer votes than Eclipse. There is no denying it, we're geeks.

Best Platform for Developing Rich Internet Applications

HTML5

Runner-up: Adobe AIR

HTML5 is the new kid on the block this year, and it managed to take 80% of the vote! Adobe AIR gets an honorable mention, but only on principle. It was an entire order of magnitude less popular than HTML5.

Best Package Management Application

apt

Runner-up: Synaptic

It's no surprise that with Ubuntu and Debian in the top spots for distributions, apt would win handily for package management. Synaptic is a far-off second place, with dozens of others taking up the rear. But, our favorite response for this topic was `configure; make; make install`.

Best Content Management System

WordPress

Runner-up: Drupal

Our Webmistress, Katherine Druckman, is a die-hard Drupal fan—and for good reason, the entire *Linux Journal* site uses it and has for many years. Perhaps just to prove she didn't rig the voting, WordPress takes top spot again this year by a fairly narrow margin to Drupal. The great thing about open source is that it's hard to lose whichever route you take.

Best Linux-Friendly Web-Hosting Company

“Other”

Runner-up: 1&1 and GoDaddy.com (tie)

We’re taking these results as a good sign, in that perhaps so many Web-hosting companies are Linux-friendly, it was hard to pick one over the other. So in reality, GoDaddy and 1&1 took more votes than any other single Web-hosting company. Because their single-digit “victories” seemed a bit strange to celebrate, we gave them runner-up status to all the other options you sent in. Feel free to cry foul; it just seemed like the logical thing to do.

Best Linux Laptop Vendor

Dell

Runner-up: ASUS

Dell still grabs the top spot here, similar to last year. In a second-place upset, however, ASUS grabs the silver medal, and Lenovo (last year’s runner-up) didn’t even make the chart. This category is becoming less and less important every year, only because Linux is working on more and more laptops out of the box. We think that’s a great problem to have.

Best Linux Desktop Workstation Vendor

Dell

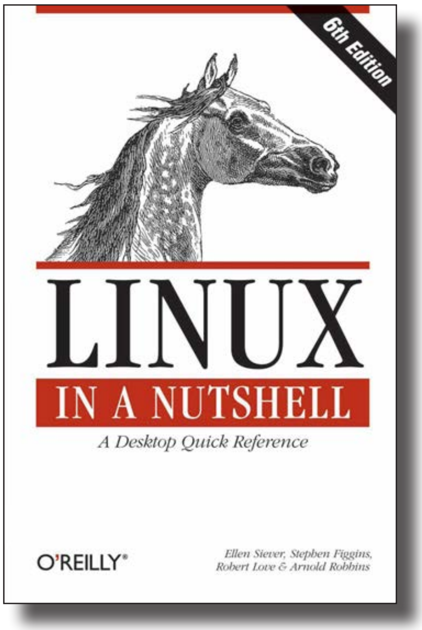
Dell is on everyone’s love-letter list this year and took the desktop workstation category by storm. In fact, the competition was so lopsided, we can’t even declare a runner-up. Dell gets all the penguin love.

Best Linux Server Vendor

IBM

Runner-up: Dell

Not to be outdone, IBM pulls through with a very narrow victory over the ever-popular Dell in our server category. When it comes to server racks, our readers trust Big Blue over anyone else (but just barely).


6th Edition
LINUX
IN A NUTSHELL
A Desktop Quick Reference
O'REILLY
Ellen Siever, Stephen Figgins,
Robert Love & Arnold Robbins

Best Linux Book

Linux in a Nutshell **by Ellen Siever et al.**

Runner-up: Just for Fun: The Story of an Accidental Revolutionary *by Linus Torvalds and David Diamond*

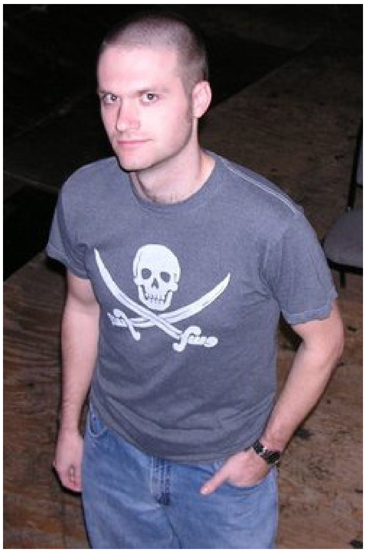
Linux in a Nutshell from O'Reilly remains your favorite book again this year. In fact, it took twice as many votes as the number two favorite, *Just for Fun*. We mentioned *Just for Fun* in last year's Readers' Choice awards, and apparently many of you took the hint and bought it.

Best *Linux Journal* Column

Hack and / by Kyle Rankin

Runner-up: Work the Shell by Dave Taylor

Kyle takes the rest of us to task again this year, stealing the number one spot for his Hack and / column. It's hard to hate Kyle, because he truly is a helpful, humble, easy-going guy. The second spot goes to an equally awesome individual, Dave Taylor. I know I'm biased, but picking a favorite *Linux Journal* column is like picking a favorite flavor of ice cream—it's hard to go wrong!



Best Brand of Video Chipset

NVIDIA

Runner-up: ATI/AMD

The big two in video chipsets are still NVIDIA and AMD. Intel is creeping up as a force in the embedded-video-chipset realm, but NVIDIA and AMD still are way ahead in popularity. NVIDIA takes more than twice as many votes as AMD, but both are wildly more liked than Intel, the distant third. No one likes the proprietary drivers, but we're all thankful to have working software and accelerated video.

Best Linux Smartphone Manufacturer

HTC

Runner-up: Samsung

This year, we tweaked the poll a little bit, and instead of looking for a specific model of smartphone, we asked for your favorite manufacturer. The race was tight between HTC's hackability and Samsung's wafer-thinness. HTC edged out the competition, making it your favorite Linux smartphone manufacturer—at least for now.

Best Linux Tablet Manufacturer

Samsung

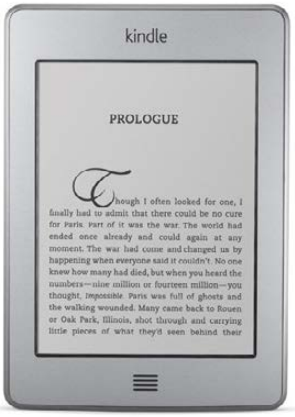
Runner-up: ASUS

In a newly created category, Samsung takes a clear and dominant victory over the other companies with its Galaxy line of tablets. This field is still very young, so it's hard to say what next year will bring, but for this year, nothing can touch Samsung. Our second-place company scored some big points with the ASUS Transformer. Its unique design allows a tablet computer to become an Android-powered laptop simply by clicking the tablet into the keyboard accessory. This will be an exciting category in the future, as competition is really starting to heat up.

Best Other Linux-Based Gadget

Amazon Kindle

Runner-up: TomTom Navigation System



Though I often looked for one, I finally had to admit that there could be no cure for Paris. Part of it was the war. The world had ended once already and could again at any moment. The war had come and changed us by happening when everyone said it couldn't. No one knew how many had died, but when you heard the numbers—nine million or fourteen million—you thought, Impossible. Paris was full of ghosts and the walking wounded. Many came back to Rouen or Oak Park, Illinois, shot through and carrying little pieces of what they'd seen behind their

The Kindle easily takes its place as your favorite Linux gadget this year. We may have to change our categories a bit next year, as the tablet/gadget/smartphone categories are starting to blend together. However you slice it, the Kindle wins this year. And if you need directions to the store in order to buy a Kindle? We recommend the TomTom Navigation System, also running Linux and also one of your favorites.

Best New Open-Source Project (released in 2010 or 2011)

LibreOffice

Although certainly standing on the shoulders of its progenitor, LibreOffice continues to progress at an impressive rate. Because the fork is technically a new project, your write-in votes were counted, and LibreOffice wins the coveted spot as best new open-source project.

Product of the Year

GNOME 3

And the moment you've all been waiting for...the winner is...GNOME 3! Although very controversial and barely edging out Android, GNOME 3 takes our product of the year title for 2011. GNOME 3 represents a drastic change in the way we compute on the desktop, and like its relative Unity, it has some people shaking their heads in frustration. You've proven, however, that change isn't always a bad thing, and GNOME 3 wins!

Be sure to follow our Web site this year, as we explore some of these winners, and perhaps watch the runners-up to see if they edge out the current incumbents. As for me, I'll be playing *World of Goo*. I hear it's rather good.

And finally, a big thanks to everyone for participating in the voting. If you have ideas for new categories you'd like us to include for Readers' Choice 2012, send e-mail to ljeditor@linuxjournal.com.■

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.

Complexity, Uptime and the End of the World

Poorly implemented monitoring systems can drive an administrator crazy. At best, they are distracting. At worst, they'll keep whoever is on pager duty up for nights at a time. This article discusses the best practices for designing systems that will keep your systems up and stay quiet when nothing is wrong.

MICHAEL NUGENT

After being in the computer industry for 20-odd years, I've come to realize there is a single thing everyone can agree on: no matter how new, how stable or how awesome any piece of technology is, it will break.

Fortunately, system administrators plan for these things. Whether it's a redundant server in the data center or a second availability zone in EC2, the first and best way to ensure uptime is to decrease the number of single points of failure across the network. There are drawbacks to this approach though. Increasing a Web cluster from one to ten boxes decreases the chance of hardware failure taking down the entire site by a factor of ten. Although this increases redundancy, it also dramatically increases the expense and complexity of the network. Instead of running a single server, there's now a series of boxes with a shared data store and load balancers. This complexity comes with drawbacks. It's ten times as likely that hardware failure will occur and a system administrator will wake up, and that only counts the actual Web servers. Whether you're in a data center or in the cloud, this kind of layering of services significantly increases the chances that a single device will go down and alert in the middle of the night.

Preventing this kind of thing is usually high on a system administrator's list of desires, even if it tends to be pushed lower on the priority list in practice. Waking up in the middle of the night to fix a server or piece of software is bad for productivity and bad for morale. You can take two steps to help make sure this doesn't happen. The first is to implement the necessary amount of redundancy without increasing the complexity of the system past what is required for it to run. The second step is to implement a monitoring system that will allow you to monitor exactly what you want as opposed to worrying about which individual box is using how much RAM.

The End of the World methodology is a thought experiment designed to help choose the level of redundancy and complexity required for the application. It helps determine acceptable scenarios for downtime. Often when you ask people when it's acceptable for their sites to be down, they'll say that it never is, but that's not exactly true. If an asteroid strikes Earth and destroys most of the human race, is it necessary for the site to stay up? If the application is NORAD, maybe it is necessary, but for Groupon, not so much. That kind of uptime requires massive infrastructure placed in strategic locations around the globe and the kind of capital investments and staffing to which only large governments usually have access.

Backing off step by step from this kind of over-the-top disaster, you can find where the acceptable level is. What if the disaster is localized to just the continent? Is it acceptable to be down at this time? If the site is focused on those customers, it may be. If the site is an international tool, such as Amazon or Google, possibly not. What if it's local to the data center or availability zone where your boxes are kept? Most shops would like to stay up even if a backhoe cuts the power to their data center.

When the problem is framed this way, it becomes obvious that there is an acceptable level of downtime. Administrators can

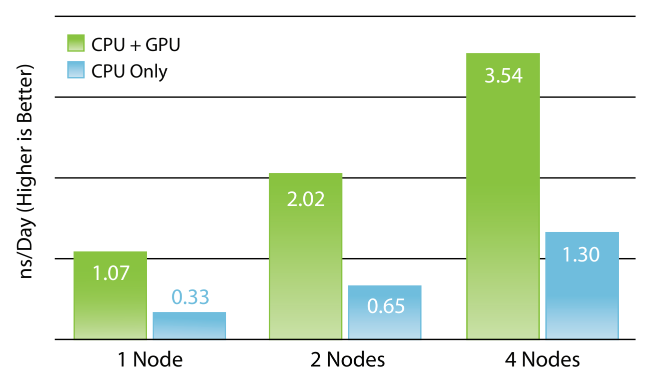
find the sweet spot between uptime and complexity. Finding the outer bounds of these requirements will uncover the requirements for monitoring the service as a whole. Notice that this is a service and not a server. Although it's easy to monitor whether a network interface is available, it's far more interesting to monitor the health of an entire cluster. In our ten-server cluster, if www6 goes down on a cluster that gets 40% utilization at night, it's probably not worth getting up for. If the entire Web service goes down, that's something that needs to be acted upon immediately.

A monitoring system is basically a scheduler and data collection tool that executes checks against a service and reports the results back to be presented on a common dashboard. It seems like one of those innocuous pieces of software that just runs in background, like network graphs or log analysis, but it has a hidden ability to hurt an entire engineering department. False positives can wake people up in the middle of the night and cause ongoing dread of going on pager duty. This results in people putting things in maintenance mode to quiet the false positives and can end up with an unnoticed failure of services.

Dealing with false positives often is

more of a policy problem than a design problem. Choosing what to monitor is far more important than choosing how to monitor it. Many companies have a history of monitoring things like CPU and RAM usage. They feel that sometimes spikes are precursors to crashes, so alerting on them is reasonable. The problem here is things that can cause the computer to use CPU and RAM, and most of them are within the normal bounds of an operating system. When the system administrator checks on the box, the resource is in use, but the application is functioning without a problem. Unless there is a clear documented link between RAM over a certain level and a crashing service, skipping on alerts for this kind of resource use leads to far fewer false positives. Monitors should be tied to a defined good or bad value with respect to a particular production service.

Another path that leads to a large number of false positives is using percentages in differently equipped boxes. For example, if a system has a 137G drive that's 95% full, it has only around 6G free. On sites with heavy traffic or sites with a lot of instrumentation in the code, 6G can go pretty quickly. Applying this monitor to the same Web server with a 2TB disk seems like less of an emergency. Leaving "only"



100G free on a system overnight is usually not a problem. If the average disk use for a day of work for a particular box is 5G, monitoring for 15G left and only allowing alerts for it during business hours will give three days notice. Alerts this far ahead of time let the system administrator plan downtime for the system if it is required, so that the server can be maintained without taking the supported service down.

The two most popular open-source monitoring systems are Zenoss and Nagios. Both of these systems offer similar monitoring capabilities. Zenoss provides more functionality and ease of use, incorporating some basic auto-discovery of nodes, built-in RRD graphing, syslog management and the ability to deduplicate events. Nagios provides a larger community and lighter install than Zenoss that allows administrators to use their own graphing solutions without duplicating software. The best part is that they have a common format for monitoring scripts—the processes that do the actual checking of services.

Although both systems come with basic templates for monitoring HTTP ports with similarly popular services, much of the power of these systems comes from the ability to write custom scripts. This is a great way to check not only that a Web server is up, but also that the application itself is working. The following is an example of a script that will monitor the success of Hudson jobs by calling its JSON API:

```
#!/usr/bin/env ruby
# Call as:
# check_hudson_job.rb ${jobname} ${hostname}

require 'rubygems'
require 'json'
require 'net/http'

jobname = ARGV[0]
hostname = ARGV[1]

url = URI.parse("http://#{hostname}/job/#{jobname}/
➥lastBuild/api/json")
res = JSON.parse(Net::HTTP.get_response(url).body)
lastResult = res["result"]

if lastResult == "SUCCESS"
 puts "OK|Status=0"
 exit(0)
else
 failurl = URI.parse("http://#{hostname}/job/
➥#{jobname}/api/json")
 failres = JSON.parse(Net::HTTP.get_response(failurl).body)
 health = failres["healthReport"][0]["description"]
 puts "Job #{jobname} broke: #{health}"
 exit(1)
end
```

The monitoring system calls the code with command-line parameters of the name of the job and the name of the host. The code then looks for the result from the Hudson server and checks for success. The return value and exit code are how the monitoring script replies to the monitoring system. A nonzero exit code indicates a failure, and the return value is a string

Hooking up the EC2 command-line programs to the monitoring service will allow new boxes to be launched if some are experiencing problems due to resource starvation, load or programs crashing on the box.

that the system displays as the reason for the failure. On Zenoss, this is also used in deduplication. On success, the monitoring script has an exit code of 0 with a string returned in a special form for the system to process (see code).

Using this structure, system administrators can work with developers to build custom URLs that the monitoring system can access to determine the health of the application without worrying about every system in the set.

It may seem hard to swallow that it's acceptable to leave a box down overnight. It may be the first in a cascading series of failures that cause multiple servers to go down, eventually resulting in a downed service, but this can be addressed directly from the load balancer or front-end appliance instead of indirectly looking at the boxes themselves. Using this method, the alert can be set to go off after a certain number of boxes fail at certain times of day, and there is no need to solve harder problems, such as requiring each box to know the state of the entire cluster.

So far, the design for the systems has been fairly agnostic as far as geographies and cloud footprint. For most applications, this doesn't make a lot of difference. Usually, with multiple geographies, each data center has its own instance of the monitoring system with each one monitoring its siblings in the other locations. Operating in the cloud offers greater flexibility. Although it still is necessary to monitor the monitoring system, this can be done easily using Amazon's great, but far less configurable system to monitor Nagios or Zenoss EC2 instances.

What really stands out about Amazon's cloud is that it's elastic. Hooking up the EC2 command-line programs to the monitoring service will allow new boxes to be launched if some are experiencing problems due to resource starvation, load or programs crashing on the box. Of course, this needs to be kept in check, or the number of instances could spiral out of control, but within reasonable bounds, launching new instances in place of crashing or overloaded ones from inside of a monitoring script is relatively easy.

Here is an example of a script that monitors the load of a Hadoop cluster and adds more boxes as the number of jobs running increases:

```
#!/bin/bash
# Call as:
# increase_amazon_set.sh ${threshold} ${AMI}

THRESHOLD=$1
AMI=$2

NUM_JOBS=`/opt/hadoop/current/bin/hadoop job -list |
 ➥head -1 | awk {'print $1'}`

if [[ $NUM_JOBS -gt $THRESHOLD ]] ; then
 echo "Warning: $NUM_JOBS running, increasing cluster size by 3"
 ec2-run-instances $AMI -n 3 --availability-zone us-east-1a
 exit 1;
else
 echo "OK|Status=0"
 exit 0;
fi
```

This follows the same format as the previous script, passing in variables from the command line and returning values to the monitoring system using the exit condition and returned strings. The big difference here is that you're not just monitoring a problem and passing it off to a system administrator to act on it. This script acts as an orchestrator, attempting to fix the problem it sees. Although care should be taken to place proper bounds on the way this works, and the computer should not be able to run amuck on the network, this kind of intelligent scheduler can be a powerful tool in automating tasks.

Although the idea of setting up a new monitoring system from scratch with great alerting rules and intelligent orchestration is a great idea, it's often just not possible. Most organizations have a monitoring system in place already, and often it's full of old alerts and boxes that have been placed in maintenance mode because they're more noisy than broken. If this is the case, it's time to cut out the cruft. Delete all the current alerts and take everything out of maintenance mode that isn't actually undergoing maintenance. Take the top ten noisy and badly behaved devices, and either stop monitoring the items that are provoking false positives or rewrite the scripts so they provide more meaningful data. When these first ten are under control, move to the next group. It may take a few iterations over a few days, but in the end, you'll care more about the messages coming from what could be a very powerful tool for you.

Monitoring systems often are overlooked as a required annoyance, but with a little bit of effort, they can be made to work for you. Monitoring for services, looking at clustered applications and alerting only on actual errors that can be handled provide real metrics to use for capacity planning and let system administrators sleep through the night so that they can be more proactive from day to day.

Michael Nugent has spent a good deal of his time designing large-scale solutions to fit into a tiny budget and leveraging Linux to fulfill the roles that typically would be filled by large commercial appliances. Recently, Michael has been working to design map-reduce clusters and elastic cloud systems for growing startups in the Silicon Valley area. When not building systems, he likes sailing, cooking and making things out of other things. Michael can be reached at michael@michaelnugent.org.

Security Threats 2012:

Secure & Empower Today's Enterprise

Protection in a Cloud, Collaboration, and Consumerization Environment

January 23, 2012 - Pre Conference Workshop
January 24-25, 2012 - Conference
Washington Plaza Hotel, Washington, DC

The consumerization of IT is in full tilt. The new application paradigm offers tremendous power – but challenges established security, risk, and compliance practices. Yesterday's solutions can't meet today's IT reality. Cloud computing, mobile apps, always–on connectivity, and social media force security professionals to develop new, more comprehensive solutions. Providing effective, unobtrusive security is the true modern day IT objective. Security Threats 2012 presents the best practices for tomorrow's security environment.

At this forum, leading-edge IT and security experts will discuss how they simultaneously protect and empower their businesses. There are few unbiased IT/security discussions in the marketplace, however, at this intimate forum you'll have the opportunity to learn from thought-leaders making these daily decisions.

Sponsorship and Exhibiting Opportunities
If you are interested in sponsoring, speaking or exhibiting at this event,
please call **212-532-9898** or email **info@opalevents.org**

Register
To register, visit us online at **www.opalevents.org**
or email us at **marketing@opalevents.org**

REF CODE: SETEA1203

Opal Events
Your Source For Superior Events

MariaDB/MySQL, PostgreSQL and SQLite3: Comparing Command-Line Interfaces

Don't be afraid of using your chosen database's command-line client. DANIEL BARTHOLOMEW

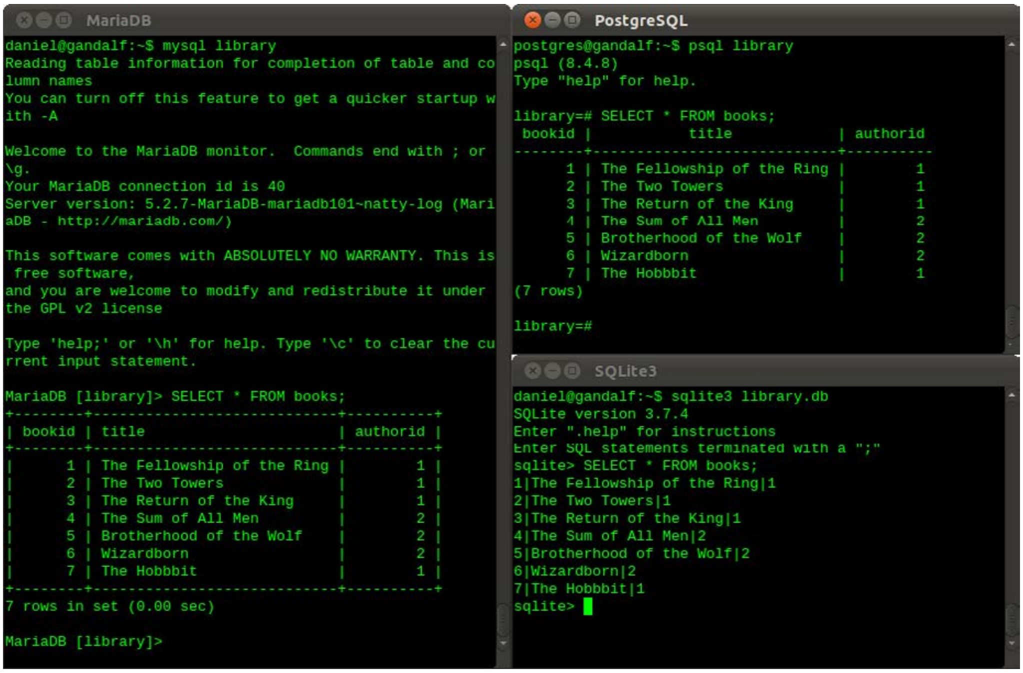
I might as well say this up front: I don't like using GUI (aka non-command-line or graphical) tools with my databases. This is likely because when I first learned it was with command-line tools, but even so, I think command-line database tools often are the best way to interact with a database manually.

Two of the most popular databases in use on Linux are MySQL and PostgreSQL. Each of them have very useful, if slightly different, command-line clients. If you ever need

To Serve...or Not

PostgreSQL and MariaDB have what is known as a client/server architecture. Clients connect to the server, and although client and server often are installed together and you may think of them as a single entity, they actually are not. The client does not need to be run on the same machine as the server. The MariaDB server is called mysqld, and it always is running while the server is up. Likewise, the PostgreSQL server is called postgres.

SQLite does not have a client/server architecture. There is just the database you are using, which is a local file, and client programs, which can interact with it.



```
daniel@gandalf:~$ mysql library
Reading table information for completion of table and co
lumn names
You can turn off this feature to get a quicker startup w
ith -A

Welcome to the MariaDB monitor.  Commands end with ; or
\g.
Your MariaDB connection id is 40
Server version: 5.2.7-MariaDB-mariadb101~natty-log (Mari
aDB - http://mariadb.com/)

This software comes with ABSOLUTELY NO WARRANTY. This is
 free software,
and you are welcome to modify and redistribute it under
the GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the cu
rrent input statement.

MariaDB [library]> SELECT * FROM books;
+--------+-----------------------------+----------+
| bookid | title                       | authorid |
+--------+-----------------------------+----------+
|      1 | The Fellowship of the Ring  |        1 |
|      2 | The Two Towers              |        1 |
|      3 | The Return of the King      |        1 |
|      4 | The Sum of All Men          |        2 |
|      5 | Brotherhood of the Wolf     |        2 |
|      6 | Wizardborn                  |        2 |
|      7 | The Hobbbit                 |        1 |
+--------+-----------------------------+----------+
7 rows in set (0.00 sec)

MariaDB [library]>
```

```
postgres@gandalf:~$ psql library
psql (8.4.8)
Type "help" for help.

library=# SELECT * FROM books;
 bookid |           title            | authorid
--------+----------------------------+----------
      1 | The Fellowship of the Ring |        1
      2 | The Two Towers             |        1
      3 | The Return of the King     |        1
      4 | The Sum of All Men         |        2
      5 | Brotherhood of the Wolf    |        2
      6 | Wizardborn                 |        2
      7 | The Hobbbit                |        1
(7 rows)

library=#
```

```
daniel@gandalf:~$ sqlite3 library.db
SQLite version 3.7.4
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> SELECT * FROM books;
1|The Fellowship of the Ring|1
2|The Two Towers|1
3|The Return of the King|1
4|The Sum of All Men|2
5|Brotherhood of the Wolf|2
6|Wizardborn|2
7|The Hobbbit|1
sqlite> 
```

Figure 1. The MariaDB, PostgreSQL and SQLite3 Clients in Action

to move between these two databases, or if you're new to databases in general, a comparison of the two is helpful.

But, because a two-horse race isn't as thrilling as a three-horse one, I wanted to include a third database command-line client in my comparison. I chose SQLite, because it is arguably the most popular database in the world. You probably have several SQLite databases on your local computer right now. The command-line client is nice too.

Also, I use MariaDB instead of MySQL in my examples, because that's what I have installed, and because I like the improvements MariaDB includes in both the command-line client and in the database server. MariaDB and MySQL are very compatible, and my examples are simple, so whenever I mention MariaDB, you can assume it applies to MySQL as well.

Installation

I won't go into how to install MariaDB, MySQL, PostgreSQL or SQLite3 here. Most distributions have packages for them, and in the case of MariaDB, there are packages for Debian, Ubuntu, Red Hat and a generic Linux binary available from its download page. See the documentation for each and your distribution's documentation for instructions.

On Ubuntu, you can install all three with the following:

```
sudo apt-get install mariadb-server postgresql sqlite3
```

Other Linux distributions are just as easy for the most part. (You need to have added the appropriate MariaDB Ubuntu repository for the above to work. Instructions are on the MariaDB downloads page.)

Basic Client Commands

The client programs for MariaDB, PostgreSQL and SQLite3 are mysql, psql and sqlite3, respectively. I've listed several useful commands for each client in Table 1. The first entry shows the basic command used to connect to a database; however, each client has several options.

Table 1. MariaDB/MySQL, PostgreSQL and SQLite Client Cheat Sheet

Task	MariaDB/MySQL	PostgreSQL	SQLite
Connect to a database	mysql <dbname>	psql <dbname>	sqlite3 <filename>
Client help	help contents	\?	.help
SQL help	help contents	\h	n/a
List databases	SHOW DATABASES;	\l	.databases
Change database	USE <dbname>	\c <dbname>	n/a
List tables	SHOW TABLES;	\dt	.tables
Show table info	DESCRIBE <tablename>;	\d <tablename>	.schema <tablename>
Load data	LOAD DATA INFILE '<file>'	\i <file>	.import <file> <table>
Export data	SELECT ... INTO OUTFILE '<file>'	\o <file>	.dump <table>
Exit the client	quit (or exit)	\q	.exit

These include (in the case of MariaDB and PostgreSQL) options for specifying the user, password and database host server. You will need these often, so refer to the man pages for the clients for what they are and how to use them. Some of the commands listed in Table 1 have extended options; refer to the documentation for details.

The first time you connect to a newly installed MariaDB or PostgreSQL database, you need to connect as the database superuser because you likely have not set up any other users.

To launch a freshly installed MariaDB mysql client, do the following:

```
mysql -u root -p
```

You will be prompted for the password you entered during the package install process.

To launch a freshly installed PostgreSQL psql client, do the following:

```
sudo su - postgres
psql
```

Creating and Deleting a Database

Just installing database clients and/or servers does not automatically give you a database to work with. For MariaDB and PostgreSQL, a database can be created either with the client or with an external utility.

In MariaDB and PostgreSQL, to create a database called library, the command is:

```
CREATE DATABASE library;
```

To connect to this newly created database in MariaDB, do:

```
USE library
```

In PostgreSQL, do:

```
\c library
```

To delete the newly created library database, drop it with:

```
DROP DATABASE library;
```

I shouldn't have to say this, but be careful with the previous command. If you just dropped the library database, create it again. You'll need it later to follow along with the examples in this article.

In SQLite3, there is no database server, and databases are just regular files, often with a .db extension. To create a database, name it on the command line when you launch the client, and if it doesn't exist, the client will create it, like so:

```
sqlite3 library.db
```

To remove an SQLite3 database, just remove it like you would any other file (with `rm` or via your file manager).

Managing Users and Permissions

There isn't space to go into the details of how to create and manage the permissions of database users here. Refer to the MariaDB and PostgreSQL documentation for details. I will continue to use the default superuser accounts for the examples here.

There is no internal database user or user permissions management with SQLite3. If local users have write access to the database file, they can do anything they want.

Common SQL Operations

This article is about the command-line clients for MariaDB, PostgreSQL and SQLite, but one of the main things you do when using such clients is write SQL statements. So let's look at some of the basic SQL-related similarities and differences between the three.

The most common SQL statements are selects, inserts, updates and deletes. As a computer language, SQL is one of the more popular ones, and there is an official standard, ANSI SQL, which has gone through various revisions through the years. Most relational database management systems (RDBMSes) use SQL as their query language, but they differ in how closely they adhere to ANSI SQL. Of the three I'm exploring here, PostgreSQL sticks closest to the standard. MariaDB drifts from the standard in places to make it easier to use. SQLite3 doesn't pretend to support every feature of ANSI SQL. Instead, it supports only a subset. After all, it's supposed to be "Lite".

Some people would like to see SQL die and never be used again. I am not one of those people. SQL has issues, but so do most computer languages. I find SQL easy to read, flexible and well worth the time it takes to learn it. The examples below are simple, and I gloss over a lot of the complexity of SQL. I also don't explain every part of every statement. My goal here is to give you a taste of what SQL looks like in practice and to point out some of the similarities and differences between the three databases. The on-line documentation for each of these databases (and the in-client help for MariaDB and PostgreSQL) includes extensive information on SQL syntax. I found the SQLite syntax diagrams to be especially helpful for that database.

SQL statements can be written on a single line, or they can be broken up across many lines to make it easier to read. In the examples

The SERIAL Datatype

A datatype is how you tell the database what type of data is in a column. Common datatypes include integer, text, varchar and date. The SERIAL datatype is a special one. In MariaDB, the SERIAL datatype is an alias for the following:

```
BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE
```

That's quite a mouthful, but it does the job of creating a column suitable for use as a PRIMARY KEY. BIGINT is a large integer; UNSIGNED means no negative values; NOT NULL means it can't be empty; AUTO_INCREMENT means that if a specific value is not specified when a row is inserted, the value should be "the current highest value + 1"; and UNIQUE means that no other row in that table is allowed to have the same value in that column.

In PostgreSQL, the SERIAL datatype is an alias for this:

```
INTEGER NOT NULL DEFAULT nextval('tablename_colname_seq')
```

The odd `nextval('tablename_colname_seq')` bit is referring to an "ALTER SEQUENCE", specifically:

```
ALTER SEQUENCE tablename_colname_seq OWNED BY tablename.colname;
```

This is just PostgreSQL's way of creating an auto-incrementing column. Thankfully, when you create a column with type SERIAL, PostgreSQL creates the ALTER SEQUENCE for you. This column also is suitable for use as a PRIMARY KEY.

below, I do the latter. SQL statements usually end with a semicolon (;).

The CREATE TABLE Statement

You won't get very far in your database adventures without some tables. If you're not familiar with databases, think of database tables as spreadsheet sheets, without all the fonts and border styles.

Returning to our library example, the most common things in a library are books, so let's create a books table:

```
CREATE TABLE books (
  bookid serial PRIMARY KEY,
  title varchar(100) NOT NULL,
  seriesid integer,
  authorid integer
);
```

The above works for both MariaDB and PostgreSQL, but it doesn't work for SQLite3, because of the use of the SERIAL datatype, which often is used as the datatype for a PRIMARY KEY. See the "The SERIAL Datatype" sidebar for more information.

A common feature of many database tables is a PRIMARY KEY. This key uniquely refers to a single row of a table. The PRIMARY KEY can be a combination of two or more columns in a row (as long as the combination is guaranteed to be unique in that database table), but most commonly, there is a specific, auto-incrementing column that is used as the PRIMARY KEY.

Every row in an SQLite3 table automatically has a PRIMARY KEY column (SQLite calls it the RowID) created when you create the table. However, it is hidden unless you specify a column with a type of `integer PRIMARY KEY`. So for SQLite, change the bookid line in the CREATE TABLE statement above to this:

```
bookid integer PRIMARY KEY,
```

And, SQLite3 will create a table with equivalent settings to MariaDB and PostgreSQL.

The INSERT Statement

Now that you have a table, it's time to enter (or INSERT) some information. Inserting data between the three databases is very similar, but there is one important difference. Both MariaDB and PostgreSQL allow you to insert multiple rows of information in one statement. SQLite3, on the other hand, lets you insert only a single row at a time.

For example, to insert some data into the books table you created earlier, use this SQL statement for both MariaDB and PostgreSQL:

```
INSERT INTO books (title, seriesid, authorid) VALUES
  ('The Fellowship of the Ring', 1, 1),
  ('The Two Towers', 1, 1),
  ('The Return of the King', 1, 1),
  ('The Sum of All Men', 2, 2),
  ('Brotherhood of the Wolf', 2, 2),
  ('Wizardborn', 2, 2),
  ('The Hobbbit', NULL, 1);
```

You may have noticed a typo in the last line. I did it on purpose so you would have something to fix later.

For SQLite3, each row that you are inserting needs to be done separately, like so:

```
INSERT INTO books (title, seriesid, authorid) VALUES
  ('The Fellowship of the Ring', 1, 1);
INSERT INTO books (title, seriesid, authorid) VALUES
  ('The Two Towers', 1, 1);
```

...and so on.

In the SQL statements above, I don't specify the bookid in the column names section. I do this because that column is set up as the PRIMARY KEY, and it is filled automatically by the database with the correct value.

The SELECT Statement

SELECT is the most common database operation. The only reason I didn't talk about this first is because until the table was CREATE-ed and had data INSERT-ed into it, as you did in the previous sections, there was nothing to SELECT.

On all three of the databases, SELECT statements work pretty much the same. Basic SELECT statements, such as the following, will work on all three:

```
SELECT * FROM books;
SELECT title, authorid FROM books WHERE authorid = 1;
SELECT * FROM books ORDER BY authorid;
```

Joins also work very well across all three. Joins are where you combine information from two or more tables together. For example, here is a join that matches author names to their books based on the authorid number:

```
SELECT title AS "Book Title",
givenname, surname
  FROM books INNER JOIN authors
USING (authorid)
ORDER BY surname;
```

The above SELECT statement presupposes the creation of an authors table and the insertion into it of at least a couple rows of data, like so:

On MariaDB and PostgreSQL:

```
CREATE TABLE authors (
  authorid serial PRIMARY KEY,
  surname varchar(100),
  givenname varchar(100),
  birthdate date
);
```

On SQLite3, change the authorid line to the following, and the CREATE TABLE statement will work properly:

```
authorid integer PRIMARY KEY,
```

Here is some data for the table, formatted to work on all three:

```
INSERT INTO authors (surname, givenname) VALUES
  ('Tolkien', 'J.R.R.');
INSERT INTO authors (surname, givenname) VALUES
  ('Farland', 'David');
```

Now, you can run the `SELECT ... JOIN` statement.

The UPDATE Statement

Remember that typo? Well, it's time to fix it. This UPDATE line works for all three:

```
UPDATE books SET title = 'The Hobbit' WHERE title = 'The Hobbbit';
```

The DELETE Statement

Deleting rows also is the same across all three:

```
DELETE FROM books WHERE bookid = 7;
```

The above will delete the row in the books table that has a bookid of 8. If you've been following along, there should not be an entry with that bookid, so nothing will happen.

The ALTER Statement

Suppose I decide to remove the seriesid column from the books table. In MariaDB and PostgreSQL, the following statement will do it:

```
ALTER TABLE books DROP seriesid;
```

SQLite3, on the other hand, does not support the removal of columns from tables. You can add columns to a table, or modify columns, but the only way to remove a

SQLite Output

When trying the SQL examples, you will notice the SQLite output is not nearly as pretty as the output from MariaDB/MySQL or PostgreSQL. By default, SQLite doesn't print column names or try to pad columns so that they line up nice and fancy like the others do. To make SQLite do so for the `SELECT ... JOIN` statement, enter the the following commands before the statement:

```
.explain ON
.mode column
.width 30 10 10
```

The `.explain` command instructs SQLite to display column headers; `.mode` sets the output to display in columns, and the `.width` command sets the width of the columns. The only issue with doing this is that it will mess up the output of future queries (unless they happen to look fine with the `.width` values you specified). To reset things back to the default, set the output mode back to the default "list" with `.mode list`. Doing this also turns off `explain` and resets the column widths back to their defaults.

column is to create a new table without a seriesid column, transfer the data from the old table to the new table, drop the old table, and then rename the new table to the original name. It's not as annoying as you might think, thanks to some SQL INSERT trickery (well, I thought it was tricky the first time I saw it in action). The basic idea is to use the output of a SELECT statement as the input to an INSERT statement, like so:

```
CREATE TABLE books2 (
  bookid integer PRIMARY KEY NOT NULL,
  title varchar(100) NOT NULL,
  authorid integer
);
INSERT INTO books2 (bookid, title, authorid)
  SELECT bookid, title, authorid FROM books;
DROP TABLE books;
ALTER TABLE books2 RENAME TO books;
```

The above trick also works as written in MariaDB and PostgreSQL as long as you change the bookid line of the CREATE TABLE statement to the following:

```
bookid serial PRIMARY KEY,
```

But, that's an awful lot of work if you just want to drop a column from a table.

These examples should be enough SQL to give you a picture of how the three compare to each other.

Conclusion

It is not hard to interact with databases on

Single vs. Double Quotes

In the SQL examples I use single quotes (') for most things and double quotes (") sparingly. MariaDB and SQLite allow you to use single or double quotes interchangeably for most quoted text in queries. PostgreSQL is pickier, because it tries to stay closer to the ANSI SQL standard, which says single quotes should be used for values (for example: `title = 'The Hobbbit'`), and double quotes should be used for system identifiers (field names, table names and so on—for example: `SELECT title AS "Book Title"...`). You can force MariaDB to obey the standard—and reject double-quoted values—with the command `SET sql_mode='ANSI_QUOTES'`.

the command line. In my opinion, doing the tasks listed above is much easier on the command line than through a graphical database program.

Of course, manipulating your database by hand, whether on the command line or with a graphical program, probably should be avoided in many cases in favor of using an automated front end—for example, a PHP content management front end for the database that contains the content for your company Web site. However, for those times when you do need to dive in and tweak something manually, or for small projects that don't justify the time or expense of a custom front end, there is no need to be afraid of using the command-line client of your chosen database.■

Daniel Bartholomew works for Monty Program (http://montyprogram.com) as a technical writer and system administrator. He lives with his wife and children in North Carolina, and he often can be found hanging out in #maria on Freenode IRC (he occasionally pokes his head into #linuxjournal too).

Resources

MariaDB Web Site: **http://mariadb.org**

MariaDB Documentation: **http://kb.askmonty.org**

MariaDB Downloads:
http://downloads.askmonty.org

PostgreSQL Web Site: **http://www.postgresql.org**

PostgreSQL Documentation:
http://www.postgresql.org/docs

PostgreSQL Downloads:
http://www.postgresql.org/download

SQLite Web Site: **http://www.sqlite.org**

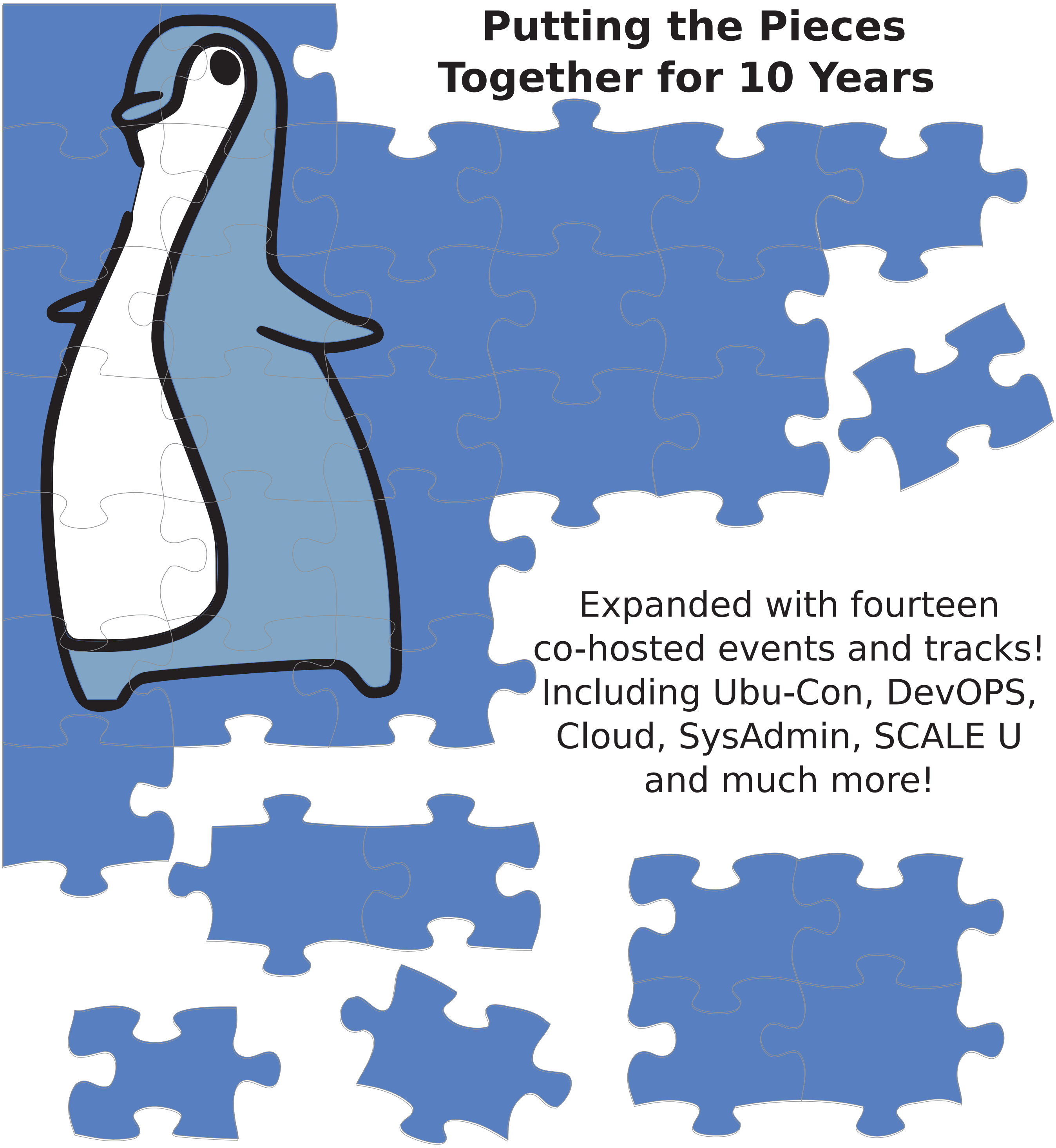
SQLite Documentation:
http://www.sqlite.org/docs.html

SQLite Downloads:
http://www.sqlite.org/download.html

SQLite SQL Syntax Diagrams:
http://www.sqlite.org/syntaxdiagrams.html

Wikipedia Article on SQL:
http://en.wikipedia.org/wiki/SQL

Wikibooks Article on Moving between MySQL and PostgreSQL: **http://en.wikibooks.org/wiki/Converting_MySQL_to_PostgreSQL**



Using Linux with EFI

EFI features and how they impact Linux. RODERICK W. SMITH

A seismic shift is under way in the computer world, and many people are unaware of the trembling beneath their feet. Since the IBM PC's introduction in 1981, most x86 and x86-64 computers have used the Basic Input/Output System (BIOS) firmware. The BIOS, however, is antiquated and limiting. The industry is, therefore, shifting from it to a new system, the Extensible Firmware Interface (EFI) and its even newer variant, the Unified EFI (UEFI). Although most computer features aren't affected by this change, it does greatly affect how the computer boots and how you must prepare your hard disk for OS installation. This article is the first in a series of four that describes these changes and helps you get Linux up and running on the new EFI-based computers. (I use "EFI" to refer to both the original EFI 1.x and the new UEFI, which is essentially EFI 2.x.)

This article describes the overall features and principles of EFI, including why you might want to use it, how EFI boots and what types of bootloaders you might use with it to enable Linux to boot on an EFI computer. The next three parts of this series will describe how to prepare to install Linux on an EFI computer, how to perform the installation and how to manage the computer once it's up and running.

Why Use EFI?

Chances are you're using BIOS-based computers today. They work fine, and they boot fine, so why should you switch to EFI? In practice, you won't have much choice, because manufacturers are phasing BIOS out. Beyond this very pragmatic reason for switching, EFI has certain advantages over BIOS. To understand these new EFI features though, you first must understand what firmware in general does and the ways in which BIOS is deficient.

Firmware is software that's built in to a device, stored in nonvolatile memory, such as electrically erasable programmable read-only memory (EEPROM) chips. Motherboards, plug-in cards and many external devices all use firmware. Both BIOS and EFI firmware run on a computer's motherboard and perform several important tasks. Most important, the firmware contains the first code that the computer runs when it starts up. This code includes hardware check functions and

Modern hardware makes larger firmware more practical than it was in 1981, so EFI implementations can be more complex than older BIOS designs.

functions that read and execute programs from the hard disk.

The IBM PC was introduced in 1981, so its BIOS was simple by today's standards. In particular, to begin the boot process, the BIOS loads the first sector of the boot device and executes it. On a hard disk, the first sector often is called the Master Boot Record (MBR), and it has several limitations that have been causing problems for 30 years:

- The MBR bootloader is tiny. Typically, it chainloads additional code in a partition's boot sector or in some other location. The methods used to locate this extra code are usually simple, because neither the BIOS nor the MBR bootloader understands filesystems.

- The computer's boot process is vulnerable to changes caused by writing new code to the MBR. OS installations sometimes render other OSes unbootable because they overwrite the MBR code, and viruses that embed themselves in the MBR have wreaked havoc over the years too.

- Getting multiple OSes or OS installations to coexist can be difficult.

- Because the BIOS design dates back 30 years, it uses ancient 16-bit 8086 operating modes. A 32-bit or 64-bit computer is unlikely to need these operating modes at all except for the boot process, but CPU manufacturers must continue to build those modes into their products just to support the BIOS.

EFI aims to overcome some of these BIOS limitations. Modern hardware makes larger firmware more practical than it was in 1981, so EFI implementations can be more complex than older BIOS designs. This added complexity enables EFI to perform tasks that BIOS implementation's can't handle. The key features of EFI include the following:

- EFI can parse partition tables and filesystems, which enables bootloader code to be stored in files in a partition. Bootloaders, therefore, can be complex, and you can store as many of them as you like on your computer.

- EFI implementations usually provide some means of selecting which bootloader to use and, therefore, which OS to boot, at boot time. In practice, these user interfaces still are usually pretty limited, so you may want to use another bootloader as the selector. (The upcoming section, "Choosing an EFI-Capable Bootloader", describes some EFI bootloader options for Linux.)

- If your OSes are well behaved, they won't overwrite each other's bootloaders. Unfortunately, bugs can and do cause problems sometimes. Also, one bootloader must be designated as the primary one, and an OS might change the primary bootloader when it installs itself.

- EFI supports drivers for filesystems and hardware, enabling you to boot from devices on plug-in boards even if those devices lack firmware of their own.

- EFI implementations typically provide a simple command-line shell and a scripting language, enabling you to write boot-time scripts that can perform various tasks before any OS boots. You can use tools, such as text editors and partitioning utilities, to adjust your system if you run into boot problems.

- The EFI specification describes a new partition table type, the GUID Partition Table (GPT). The old MBR partition system is limited to 232 sectors, which works out to 2 TiB on disks with 512-byte sectors. GPT uses 64-bit pointers, so its limit is 264 sectors, or 8 ZiB (zebibytes). Although you can use GPT on BIOS-based computers, Windows refuses to boot from GPT on BIOS-based computers. Because Windows boots fine from GPT on UEFI-based computers, UEFI is a practical necessity to boot Windows from a GPT disk.

EFI User Interfaces

One of UEFI's selling points for the public is a prettier GUI-based interface to the firmware's setup utility. This "eye candy" can be nice, but it doesn't fundamentally alter the firmware's capabilities, much less how any OS boots.

In fact, many motherboards with plain text-mode user interfaces to their firmware use UEFI. In combination with the BIOS emulation mode, this can make the computer act just like a BIOS-based model, so you may not even realize that you're using a UEFI PC!

- Most modern EFI implementations include a BIOS emulation mode. This is a stopgap measure that enables you to install an OS with no or poor EFI support even on an EFI-based computer. Intel-based Macintoshes use this feature to boot Windows using Apple's Boot Camp software.

- EFI designs can boot a computer more

quickly than can BIOS designs. In my tests, the results typically are about 20 or 30 seconds faster when using EFI boot mode rather than BIOS boot mode.

EFI has its drawbacks too, of course. The most important of these is the fact that it's new. This means that old bootloaders don't work with it and users are unfamiliar with it. One more significant problem is that the EFI boot process assumes the OS will run in the same bit depth as the EFI. Because all UEFI-based PCs and most EFI-based Macs use 64-bit firmware, this means that 64-bit OSes work best with these computers. (The earliest Intel-based Macs used 32-bit EFIs though.) Installing a 32-bit version of Linux on a computer with a 64-bit EFI is possible, but you'll give up runtime EFI interfaces. This makes bootloader maintenance harder, since the efibootmgr utility (which I'll describe later in this series of articles) relies on such interfaces. For this reason, I recommend installing a 64-bit distribution if your firmware is 64-bit.

Overall, EFI's feature set provides a great deal of flexibility. In theory, it should enable easier coexistence between different OSes on multiboot computers and easier maintenance of the boot process even on computers that boot just one OS. In practice, EFI booting still is new enough that it's sometimes awkward simply because the tools are new and small in number. Lack of familiarity also can make EFI awkward to those who know all the BIOS booting tricks.

EFI's Boot Model

Recall that the BIOS begins the boot process by reading a single sector (the MBR) from the hard disk. EFI is more complex, however, so it can read a bootloader from a file in a filesystem. To do this though, the EFI requires its own partition, just as OSes usually require their own partitions. The EFI's dedicated partition is known as the EFI System Partition (ESP). Because the EFI's main job is to boot the computer, you're likely to find OS-specific bootloaders on the ESP.

The EFI specification states that the ESP should use the FAT-32 filesystem, but in practice, any filesystem that the EFI supports will work. This normally means FAT-12, FAT-16 and FAT-32. Macintoshes also can use HFS+. Some versions of Windows refuse to accept an ESP with anything but FAT-32, so I strongly recommend using FAT-32 on your ESP.

The ESP is identified by a specific type code in the partition table. On a GPT disk, you can set this type code in various ways:

- If you use GPT fdisk (gdisk, cgdisk or sgdisk) to partition a GPT disk, you should give your ESP a type code of 0xEF00.

- If you use a libparted-based utility, such as parted or GParted, you should set the "boot flag" on the disk. Note that this "boot flag" is not equivalent to a "boot flag" on an MBR disk, and on a GPT disk, you should set it *only* on the ESP.

Linux installations normally mount the ESP at /boot/efi. The EFI directory holds subdirectories, each of which holds bootloaders and their support files. For instance, EFI/Microsoft holds the Windows bootloader files, and EFI/ubuntu holds Ubuntu's bootloader. In Linux, these directories would be /boot/efi/EFI/Microsoft and /boot/efi/EFI/ubuntu. The EFI/BOOT directory holds a default bootloader file, should no other bootloader be installed. If you install an EFI bootloader independently of your OS installations, you probably will either place it in the EFI/BOOT directory as the default bootloader or create a new subdirectory named after the bootloader itself.

EFI programs, including bootloaders, have .efi filename extensions. You can use any name you like, although the default bootloader in the EFI/BOOT directory has a special name: BOOTX64.EFI on x86-64 systems.

You can store a startup script in the startup.nsh file in the ESP's root directory (that is, /boot/efi/startup.nsh in Linux). You can use this file to launch a bootloader or to provide user-selectable preboot options, but I don't describe that in detail in this series.

The EFI specification doesn't provide much guidance on the size of the ESP. The Microsoft Windows installer creates a 100 MiB ESP by default; Mac OS X creates a 200 MiB ESP, and Linux distributions create ESPs of various sizes. I recommend creating an ESP that's in the 200–300 MiB range, particularly if you use ELILO (described shortly).

EFI implementations should provide a boot manager that enables you to select which OS to boot. EFI maintains a list of bootloaders in Flash storage on the motherboard, and you normally can enter a boot manager utility at system startup time by pressing a special key, such as F10 or F12. Sometimes you can use this boot manager or the firmware's more complete setup utility to add or remove items from the boot manager's menu. If you can't find such options, you can use Linux's efibootmgr utility (described later in this series of articles) to manage your boot options.

Because the EFI boot manager user interface varies so much from one implementation to another, you should consult your motherboard's or computer's documentation to learn more. If the documentation is unclear, you may need to experiment with it.

Choosing an EFI-Capable Bootloader

The universe of EFI bootloaders is quite limited compared to the range available for BIOS. Nonetheless, several bootloaders for Linux exist. Table 1 summarizes their features. The bootloaders include:

- ELILO: in my experience, ELILO is the most reliable Linux bootloader on UEFI-based PCs; however, I've had little luck with it on a 32-bit Mac Mini. It can load Linux kernels from the ESP, but not from other locations. This means

Table 1. Bootloader Features

Bootloader	Load Linux	Kernel Location	Chainload
ELILO	Y	ESP	N
GRUB Legacy	Y	any partition	Y
GRUB 2	Y	any partition or LVM	Y
rEFIt	N	N/A	Y
The Linux kernel	Y	ESP	N

that your ESP must be big enough to hold as many Linux kernels and initial RAM disks as you care to install. ELILO can't chainload another bootloader, so if you want to multiboot with other OSes, you'll need to use your firmware's boot manager or another bootloader in addition to or instead of ELILO. It reads its configuration file, elilo.conf, from the same directory in which its .efi file resides.

- GRUB Legacy: the official version of GRUB Legacy doesn't support EFI booting; however, Fedora has created a heavily patched version that does support EFI. This version of GRUB supports booting a Linux kernel or chainloading to another EFI bootloader. Thus, you may be able to use GRUB Legacy as your primary bootloader in a multiboot environment. It can read kernels from any common Linux filesystem, but not from an LVM or RAID configuration. In my experience, it's pretty reliable. It reads its configuration file from the same directory as its binary .efi file. The configuration file is named after the binary file—normally grub.conf.

- GRUB 2: GRUB 2 officially supports both BIOS and EFI booting; however, you must install an EFI-capable package, such as grub2-efi under Debian or Ubuntu. GRUB 2 can load a kernel from any Linux filesystem on a partition, LVM or RAID configuration. It also can chainload to another EFI bootloader. The main problem with GRUB 2 is its complexity, which makes its installations delicate. Distribution configuration scripts sometimes get details wrong, at least for EFI installations, which can render your computer unbootable.

- rEFIt: unlike ELILO, GRUB Legacy and GRUB 2, rEFIt isn't capable of directly booting a Linux kernel. Instead, it presents a menu of bootloader options. When you select a bootloader, rEFIt chainloads it. This makes rEFIt a useful replacement for an EFI implementation's bootloader, if that bootloader is limited or awkward. By default, rEFIt presents a graphical menu. The most common rEFIt binaries use a "fat" 32-/64-bit format that's usable only on Macs. If you have a UEFI-based PC, you must track down a pure 64-bit version of the program. (Debian and Ubuntu both ship with such packages; see the Resources section for another source for such binaries.)

- The Linux kernel: work is under way to embed an EFI bootloader in the Linux kernel itself. When this work is done, you will be able to launch Linux directly, without using ELILO, GRUB Legacy or GRUB 2. You'll have to store the kernel and its initial RAM disk on the ESP or some other partition that the EFI can read though. This code is not yet available in any publicly released kernel, as of the 3.1-rc7 kernel, but see the Resources section for a set of patches.

Overall, my preference for an EFI-capable Linux bootloader is either ELILO or Fedora's patched GRUB Legacy. When multibooting with a non-Linux OS, ELILO works best when paired with rEFIt. GRUB 2 is just too finicky and unreliable. It might eventually be possible to boot the Linux kernel directly using no other bootloader, but this support is still extremely new and is not yet available in released kernels.

Next Time

Part two of this series covers preparatory steps for installing Linux on an EFI computer—disk partitioning and understanding the features and limitations of some common Linux distributions with respect to EFI.■

Roderick W. Smith is a Linux consultant, writer and open-source programmer living in Woonsocket, Rhode Island. He is the author of more than 20 books on Linux and other open-source technologies, as well as of the GPT fdisk (gdisk, cgdisk and sgdisk) family of partitioning software.

Resources

Official UEFI documentation can be obtained at the UEFI home page: **http://www.uefi.org**.

ELILO is based at **http://elilo.sourceforge.net**.

GRUB is headquartered at **http://www.gnu.org/software/grub**. This page is mostly dedicated to GRUB 2, although older GRUB Legacy documentation is still available.

You can learn more about rEFIt at **http://refit.sourceforge.net**.

Pure 32- and 64-bit builds of rEFIt that include patches to eliminate some video glitches on UEFI systems are available from **http://www.rodsbooks.com/efi-bootloaders/refit.html**.

The patches to turn the Linux kernel into its own EFI bootloader can be found at **https://groups.google.com/group/linux.kernel/browse_thread/thread/9aac8bf3b646bf62/f0963b50a956f3d9?lnk=gst&q=x86+EFI+boot+stub#f0963b50a956f3d9**. Be aware that only those familiar with software patching and kernel compilation should attempt to use this feature at the moment.

usenix

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

2012 CONFERENCES

In Cooperation: 19th Annual Network & Distributed System Security Symposium (NDSS 2012)

SPONSORED BY THE INTERNET SOCIETY IN COOPERATION WITH USENIX

February 5–8, 2012, San Diego, CA, USA
http://www.isoc.org/isoc/conferences/ndss/12

10th USENIX Conference on File and Storage Technologies (FAST '12)

SPONSORED BY USENIX IN COOPERATION WITH ACM SIGOPS

February 14–17, 2012, San Jose, CA, USA
http://www.usenix.org/fast12

In Cooperation: EuroSys 2012

SPONSORED BY ACM SIGOPS IN COOPERATION WITH USENIX

April 10–13, 2012, Bern, Switzerland
http://eurosys2012.unibe.ch

9th USENIX Symposium on Networked Systems Design and Implementation (NSDI '12)

SPONSORED BY USENIX IN COOPERATION WITH ACM SIGCOMM AND ACM SIGOPS

April 25–27, 2012, San Jose, CA, USA
http://www.usenix.org/nsdi12

Workshops co-located with NSDI '12 include:

2nd Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE '12)

April 24, 2012, San Jose, CA, USA
http://www.usenix.org/hotice12
Paper registration due: January 6, 2012

5th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '12)

April 24, 2012, San Jose, CA, USA
http://www.usenix.org/leet12
Submissions due: February 13, 2012

In Cooperation: 5th Annual International Systems and Storage Conference (SYSTOR 2012)

SPONSORED BY IBM IN COOPERATION WITH USENIX

June 4–6, 2012, Haifa, Israel
http://www.research.ibm.com/haifa/conferences/systor2012

4th USENIX Workshop on Hot Topics in Parallelism (HotPar '12)

SPONSORED BY USENIX IN COOPERATION WITH ACM SIGMETRICS, ACM SIGSOFT, ACM SIGOPS, ACM SIGARCH, AND ACM SIGPLAN

June 7–8, 2012, Berkeley, CA, USA
http://www.usenix.org/hotpar12
Paper registration due: January 24, 2012

2012 USENIX Federated Conferences Week

June 12–15, 2012, Boston, MA, USA

2012 USENIX Annual Technical Conference (USENIX ATC '12)

June 13–15, 2012
http://www.usenix.org/atc12
Paper titles and abstracts due: January 10, 2012

3rd USENIX Conference on Web Application Development (WebApps '12)

June 13–14, 2012
http://www.usenix.org/webapps12
Submissions due: January 23, 2012

4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '12)

4th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage '12)

21st USENIX Security Symposium (USENIX Security '12)

August 6–10, 2012, Bellevue, WA, USA

10th USENIX Symposium on Operating Systems Design and Implementation (OSDI '12)

October 8–10, 2012, Hollywood, CA, USA
http://www.usenix.org/osdi12
Submissions due: May 3, 2012

26th Large Installation System Administration Conference (LISA '12)

December 9–14, 2012, San Diego, CA, USA

Stay Connected... http://www.usenix.org/facebook http://twitter.com/usenix

USENIX: The Advanced Computing Systems Association

FOR A COMPLETE LIST OF USENIX AND USENIX CO-SPONSORED EVENTS, SEE HTTP://WWW.USENIX.ORG/EVENTS

Mercurial— Revision Control Approximated

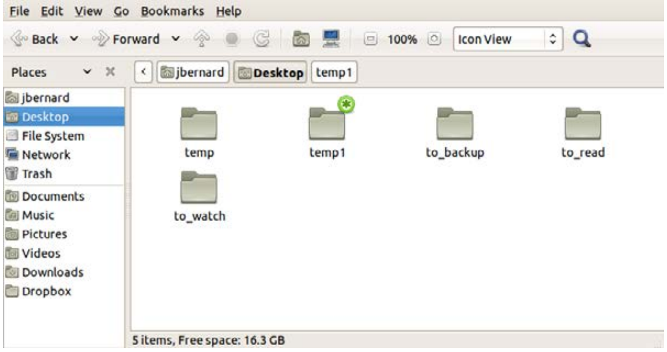
Mercurial provides a Git-like repository with the flexibility of a plugin architecture. JOEY BERNARD

A short while ago, an article appeared in *Linux Journal* implying Git was the be-all and end-all of source code revision control systems ("Git—Revision Control Perfected" by Henry Van Styn, August 2011). I would like to challenge that assumption and declare to the world that the *real* perfect version control system is here, and its name is Mercurial.

In case you didn't notice it, my tongue was firmly in my cheek in that last paragraph. I think version control systems are like editors. They are all different and fit people and their work habits differently. There is no one perfect system to rule them all. Git may be the perfect fit for some people, and RCS may fit someone else better. This article describes another option to add to the mix. Mercurial provides some of the features of systems like Git, and some of the features of systems like CVS or Subversion. Hopefully, after reading this article, you'll have enough information to make a rational choice as to what is best for you.

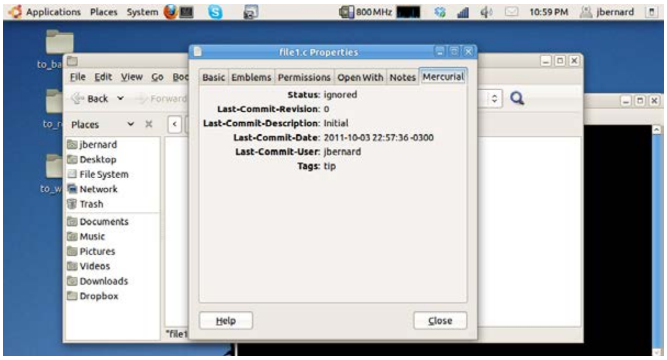
The main Mercurial site contains lots of documentation for end users and developers alike. Several tutorials are available, and they even include a series of work flows that cover how end users can use Mercurial for their development projects. Using those, you can see how you could use Mercurial as a solo developer or as one of a group of developers, or how to work with a central repository like CVS. These work flows are great starting points for you to create your own.

First, let's look at what makes up Mercurial. A Mercurial repository consists of a working directory, which is paired with a store. The store contains the history of the repository. Every working directory is paired with its own copy of the store. This means that Mercurial has a distributed system, much like Git. When you commit a series of file changes, a single changeset is created, encapsulating these changes. Each changeset gets a sequential number, called the revision number. But, remember that



5 items, Free space: 16.3 GB

Figure 1. Here you see that Mercurial repositories are tagged for easy finding.



Status: ignored
Last-Commit-Revision: 0
Last-Commit-Description: Initial
Last-Commit-Date: 2011-10-03 22:57:36 -0300
Last-Commit-User: jbernard
Tags: tip

Figure 2. Right-clicking a file and pulling up the properties gives you lots of Mercurial information.

each working directory gets its own copy of the store, so these revision numbers may not actually match up. For this reason, each revision also gets a 40-digit hexadecimal globally unique ID.

So, what happens when two users are doing parallel development? Assuming they are starting with equal repositories, any committed changes by user one creates a new branch, and any committed changes by user two also creates a new branch. User one then pulls in any changes from user two's repository. This creates two branches in user one's repository: one branch for user one's changes and one branch for user two's changes. User one then needs to merge these two branches together in order to incorporate all the changes since the last synchronization of repositories. User two would need to do the same thing (pull and merge) in order to synchronize the repositories. Changes also can be pushed to another repository.

One of Mercurial's strengths is its use of extensions. Several extensions are available from the project, and you always can go ahead and write your own. Extensions are written in Python, so hone your scripting skills. You can enable these extensions by adding them to the `[extensions]` section of your configuration file.

So, how do you actually use Mercurial? You probably will want to set some basic configuration options in the .hgrc file. Mercurial needs a user name for recording commits. You can set this option in the configuration file with:

```
[ui]
username = John Doe <john.doe@company.com>
```

The first thing to do is to create your local repository. If you are working off a copy from someone else, you would make a clone. The format of the clone command is:

```
hg clone [OPTIONS...] SOURCE [DEST]
```

The source option can take several different forms. If the repository you are cloning is on the same machine, you simply can provide the filesystem path to the source repository. Mercurial includes a Web server that can be used to provide access to a repository over HTTP. If you are cloning such a repository, the command simply would be:

```
hg clone http://[user[:pass]@]somemachine.com[:port]/[path][#revision]
```

You also can do this over HTTPS. At my work, we keep backup copies of repositories on a machine that is accessible only over SSH. And, that's fine, because Mercurial is perfectly happy cloning over SSH. You can use the following to do so:

```
hg clone ssh://user@host[:port]/[path][#revision]
```

You need to have a valid login on the remote machine, of course. The path is relative to your home directory, so if you want to use a full path, you need to start it with two forward slashes:

```
hg clone ssh://user@host//full/path/to/repo
```

Creating a new repository is even easier. All you need to do is create a directory to house all of the files going into your repository. Then, you can `cd` to that directory and execute the following:

```
hg init
```

This command creates a subdirectory named .hg, containing all of the store files for your new repository.

Changing your repository's contents is done through the `add` and `remove` commands. There also is a `rename` command you can use to change the name of a file within your repository. You can use that command to move files around within your repository as well. Let's say you want to move a file to subdirectory dir1. You would execute this:

```
hg rename file1.c dir1
```

You can get the current state of a file with the `status` command. This will tell you whether a file has been modified, added, removed and so on. The `diff` command shows the differences in a file from the current version and the last committed version. If you decide to toss away all of these changes, you can use the `revert` command to reset the file to the last committed version. Once you are happy with your edits, you can commit any changes with the `commit` command.

At the level of the repository as a whole, a lot of commands are available. When you have done a lot of editing and committed all your changes to your local copy of the repository, you can send the changes out to another repository with the `push` command. The destination for the `push` command can have any of the forms shown above in the clone command examples. If the changes of interest were made by another user at a remote repository, you can use the `pull` command to grab them and put them into your local repository.

You may want to check what is going to happen before merging these changes. Before pushing changes out, you can use the `outgoing` command to see what changesets would have been sent had you actually issued a `push` command. For pulls, you can use the `incoming` command to see what changesets would be brought in had you issued a `pull` command. Once this is done, these changes sit in a separate branch. You then need to merge this branch back in to the main one in order to incorporate the changes.

But, what if you don't really have any kind of direct access over the network? You can use the `bundle` command to generate a compressed file containing the changeset. This can then be transferred, either by e-mail or SneakerNet, to the remote repository. Once it is there, you can use the `unbundle` command to import the changeset into the remote repository. Again, you can use the `incoming` and `outgoing` commands, with the `--bundle filename` option, to check out the changesets and see what they will do before actually running the real commands.

As I mentioned earlier, Mercurial includes a Web server that can provide access to your repository over HTTP. It is not appropriate to provide public full-time access to a repository, because it doesn't provide any type of user authentication. In those cases, you would use a real Web server, like Apache, to serve up the repository. But, if you simply want to throw up the server for quick temporary access, or if you are just offering up access internally on a local network and don't need to worry too much about security, this gives you really quick access. You simply need to run:

```
hg serve [OPTIONS...]
```

Some of the more common options include `-d` or `--daemon`. This drops the Mercurial Web server into the background.

You may want to set the port that it is listening on with the option `-p` or `--port`. The default port is 8000. You can push and pull from such a Web server. If you want to serve over HTTPS rather than HTTP, you can use the option `--certificate` to set the SSL certificate file to use.

Several clients are available for working with Mercurial repositories. For GNOME users, there is a handy one called tortoise. The really great part of this client is that it integrates nicely with Nautilus. This means you can interact with your repository, commit changes, clone it, synchronize it with a remote repository and much more. You also get informational icons within Nautilus, letting you see immediately which files are outdated, changed or whatever their status may be. All of the tools are simply a right-click away. Some great standalone clients also are available, so look around and see what you like.

Hopefully, this introduction gives you some ideas on what you can get done with Mercurial. Now you don't have any excuses for not putting your source code under version control.■

Joey Bernard spends his days helping university researchers do scientific computing. But by night, he is a masked crusader fighting crime—at least, once he gets the kids to sleep and can hit the sack himself.

The OpenRISC Processor: Open Hardware and Linux

How you can use open-source hardware in your next embedded system project. JAMES TANDON

Linux has become a very mature operating system with support for a wide variety of devices and processors. There have been several bumps on the way though—a common problem for kernel developers is that manufacturers sometimes want to keep hardware proprietary by releasing binary-only device drivers. This issue has caused considerable consternation to both the manufacturers and the Open Source Software community. Open-source hardware resulted partially from this friction.

A sizable community develops open hardware so that hobbyists and professional FPGA and ASIC developers can implement advanced hardware functions in their systems. The fruits of their labor have matured into the OpenRISC soft processor core. The GNU C cross-compiler project team for OpenRISC worked in tandem with the processor project team, and now it runs our favorite operating system, Linux! My aim in this article is to explain how embedded system engineers now use open hardware for system design, and how Linux can run on an OpenRISC 1200 chip. The article explores what people have done so far with OpenRISC, then introduces you to several options for developing with Linux and OpenRISC yourself.

What Is Open-Source Hardware?

The principle behind open-source hardware is the same as with software, except that they generally use different design languages. The common languages used today for hardware are Verilog and VHDL. Just as you can download the C language source code to Linux, you can download the Verilog code to the

OpenRISC 1200 processor. Do you want to implement a multicore OpenRISC? No problem! You can instantiate as many cores as you want. Do you want to implement a network processor? Just download an Ethernet MAC core and connect it. Do you want hardware acceleration for your MPEG codec in your embedded Linux system? You can implement this in Verilog and access it directly from your processor with custom processor instructions. This is the beauty of open hardware. You can mix and match hardware components (commonly called cores) to create your own unique processor. Sound, graphics, networking, robotic control—anything is possible.

The tricky part is implementation. Very few people have a budget to pay a chip foundry like IBM or TSMC to manufacture chips using the fanciest chip technology. For a small manufacturing run, you can expect to pay tens of thousands of dollars to build a test chip. Also, there is no guarantee it will work the first time. Because of this, only companies with millions of dollars, or government organizations, can afford it. However, hobbyists are not left out completely. The field programmable gate array (FPGA) is a special kind of chip that can run "synthesized" Verilog code. This is very similar to running compiled C code. The penalty for using an FPGA is that the circuit runs about two to three times slower while using more power. However, you can purchase a prebuilt FPGA board with an Ethernet PHY, RS232, VGA or similar devices for as little as $200. It is a much more reasonable way to implement complex hardware.

What Is the OpenRISC?

The OpenRISC architecture is a 32-bit instruction, 32-/64-bit data processor. It is a specification that allows chip developers to implement the processor so that it is optimized for high speed, reduced power or minimized cost. It includes an optional cache and memory management unit (MMU) as well, which makes porting Linux possible with only minimal changes to the primary codebase. Floating-point instructions also are an option. The processor uses a special on-chip bus architecture called Wishbone to connect to other on-chip devices like an Ethernet, VGA or SDRAM controller. Also, because the processor is open-sourced, it is possible for you to extend the processor with your own instructions and registers. This is very useful for hardware acceleration. And, one key point in its favor: OpenRISC is licensed under the Lesser GNU Public License (LGLP), so it has very wide appeal.

The OpenRISC 1200 is an implementation of the OpenRISC architecture in Verilog that is known as a "soft core". Several soft-core processors are available for chip designers to purchase today. One you may know is the ARM architecture. Apple licensed an ARM soft core for the iPhone, then customized

Time, money and red tape prevent individuals from downloading and improving on proprietary cores, whereas OpenRISC is freely available to anybody.

the processor for implementation in mobile applications. Nintendo used two separate ARM processors in the Nintendo DS handheld video game system.

The OpenRISC uses a very different instruction set from ARM. However, many of the processing capabilities are the same. If you want to use the ARM soft core for integrating your project, you need the backing of a large company or research institution to license the core. Time, money and red tape prevent individuals from downloading and improving on proprietary cores, whereas OpenRISC is freely available to anybody. You can download the source and simulate it now in an open-source Verilog simulator like Icarus Verilog or Verilator.

An open-source processor is very beneficial to the community, but this raises a question: what can it do? Rather than talk about possibilities, let's look at a sample of what the community has accomplished so far with the OpenRISC:

- A full processor architecture specification with an extendible instruction set.
- Implementation in the Verilog HDL (OpenRISC 1200).
- Simulation in Icarus Verilog, an open-source simulator.
- VGA and PS2 keyboard interface implemented for usage like a traditional desktop.
- Verified OR1200 to run at 50MHz or better in FPGA.
- ASIC implementation of OR1200 runs at 150MHz in 0.18um technology (possible to run much faster in the latest 28nm technology).
- Implemented as control processor for robotic control.
- Full integration with Ethernet to implement an embedded Web server.
- Play a digital music file.
- Other devices that you can implement include USB, UART, I2C, SPI, SDRAM, SD and many more.
- GNU C Compiler 4.5.1 cross-compiler works.
- Working implementations of both

uClibc and newlib: two standard C library implementations for embedded systems.

- OpenRISC support included in the Linux 3.1 code base!
- A software simulator so programmers can write software without purchasing hardware.

If you decide to use the OpenRISC processor for your project, you can proceed with the knowledge that your base system has been proven in hardware multiple times.

Getting Started with OpenRISC

Now that you know something about OpenRISC, you might be ready to try downloading the soft core and associated development tools yourself. Several smaller projects compose the OpenRISC Project that you should look at:

- OpenRISC 1000/2000 specifications—provides a full listing of the processor architecture and instruction opcodes.
- Wishbone bus specification—standard interface for connecting devices to OpenRISC.
- OpenRISC 1200 processor source code (Verilog)—synthesizable soft-core implementation.
- ORPSoC—synthesizable embedded processor with Ethernet, SPI, SDRAM, VGA and other peripherals.
- OR1KSim—the software simulator for people developing software for OpenRISC processors.
- GNU C cross compiler for OpenRISC (version 4.5.1).
- GNU binutils for OpenRISC (version 2.20.1).
- uClibC for OpenRISC (version 0.9.0).
- Linux for OpenRISC (version 2.6 modified or version 3.1 integrated).

The OpenRISC 1200 is a bare core that does not include any peripherals—not even RAM. It is just a processor with a bare, addressable bus interface. If you want a more-complete processor with a bootloader and basic I/O interfaces, such as a serial port, VGA controller or keyboard interface, you need to start with the more-complete ORPSoC package. You can check out the latest version of the OpenRISC source tree using Subversion. Visit the Web site **http://opencores.org/or1k/Main_Page** to get started. Follow the links describing how to check out a copy of the OpenRISC repository with Subversion. This will give the the latest version of the processor with all relevant bug fixes. If you prefer to download tarballs, visit

http://opencores.org/download,or1k instead.

The OpenRISC trunk line does not include all the software, however. You probably will want to download the latest versions of Linux and GCC that are patched for OpenRISC as well. These are stored in git repositories and no registration is required. Just follow the instructions at **http://www.openrisc.net/toolchain-build.html** to get started.

These development versions may be unstable, so obtaining a release copy of the Linux 3.1 kernel or later from **ftp://ftp.kernel.org** may be a safer approach.

This will get you the latest versions of the OpenRISC software development tools. With these packages, you now have everything you need to perform software-only and hardware-software simulation on OpenRISC with Linux. A lot of information is available for installing and running these tools, but if compiling and simulating with these packages seems intimidating, don't worry. All software is built using familiar configure scripts and make files. For the hardware hackers, the Web site **http://www.opencores.org** has a large community dedicated to OpenRISC and a number of peripherals that can connect directly to it.

Chances are, if the extensive on-line help section does not answer your questions, somebody in the forums will have encountered your problem before.

If you are more of a software person with a knack for programming embedded systems, take a look at **http://www.openrisc.net** and the mailing lists listed there instead.

Experimenting with OpenRISC

Now that you have the software and hardware source code, your designs are limited only by your creativity. You can take three separate development paths when developing with the current version of OpenRISC: embedded software development, custom digital circuit implementation in FPGA-based systems or creation of your own custom processor (ASIC).

If you do not have the time or inclination to understand and develop an FPGA or ASIC but want to develop software, you will want to use the emulator, or1ksim. This high-level software emulator allows you to test your programs for correctness without having to purchase a prebuilt system. For instance, say you created a Web server control panel, but you want to test it on OpenRISC. The simulator has an Ethernet device option that your software can access and control. If you wished to simulate a handheld game system, there is emulation for general-purpose I/O (for push-buttons) and a VGA display. Here's another possibility: you already have an open-source project, but would like to test that it compiles and runs on OpenRISC/Linux. The simulator can help you see if your software package compiles and runs properly. The project is still young, so do not expect real-time 3-D rendering (yet). However, if you want to test basic functionality of your open-source project to OpenRISC, or1ksim is the way to go.

Programming Embedded Processors

So, you are ready to bootstrap your embedded Linux system on a brand-new OpenRISC implementation. How do you do this? With a cross compiler, of course. The GNU C compiler project can generate binary executables for 32-bit i386 processors or x86-64 64-bit processors on your basic GNU/Linux installation. However, if you want to develop for OpenRISC, you have to recompile the GNU C compiler so it can target OpenRISC. This is a rather involved process, which may require a special implementation of the standard C library—this means you need to write code so that printf()and scanf() know how to read and write characters! The instructions at **http://www.openrisc.net** are concise and quick. If you follow them, you can get a full working copy of the GNU toolchain for OpenRISC very quickly.

If you want to understand how to build the OpenRISC toolchain from scratch, or make modifications for your custom implementation, I recommend reading **http://www.openrisc.net/toolchain-build.html** by Gene Sally. He does an excellent job of introducing the details of cross compilers.

The hardware hackers reading this article mostly will develop with FPGAs. Altera and Xilinx produce excellent FPGA options for running the OpenRISC processor. Both companies have embedded processor cores; however, neither is open-sourced. If you develop for the Xilinx Microblaze processor, for instance, you are locked in to using Xilinx exclusively. OpenRISC gives you freedom to choose the best FPGA for speed, power or area optimization.

The fastest tested processor speed I have observed in FPGA tests is 50MHz, although I suspect that the fastest may go above 150MHz if the source Verilog and implementation scripts are fine-tuned properly for one of the newest FPGAs like the Stratix V or the Virtex-6. Altera claims that digital circuits can reach 550MHz on its latest FPGA architecture, although this is a best-case scenario. The OpenRISC processor is fully capable of controlling a robotic arm, and this application has been demonstrated (video at **http://www.youtube.com/watch?v=Lv1Gow7WZxM**).

The most extreme development path you might take is integrated circuit (ASIC) development. This work tends to be left to professionals, but open-source tools exist for layout and simulation of integrated circuits. Compiling, also known as synthesizing, Verilog and VHDL to a net list consisting of only simple logic gates is possible, though very rudimentary without proprietary tools. Also, open-source tools for placement and routing of logic gates in integrated circuits are virtually nonexistent. The best place to look for source code that compiles a net list, does place and route, or extracts parasitics is to search the Web pages of university research projects.

If you have the budget for proprietary chip-design software (think millions of US dollars), it is possible to design for the latest manufacturing process available. Apple, Inc., has licensed the ARM processor and developed the A6 processor in a 28nm technology node for the iPhone, and at the time of this writing, it is currently in testing. OpenMoko and Android are both Linux-based distributions for smartphones. A company developing a smartphone processor that is targeted for use with these distributions might consider using the OpenRISC target. General chip design is beyond the scope of this article, but if you are really interested, **http://opencircuitdesign.com** or my personal Web site **http://www.jamestandon.com** will get you started.

One group of people who have benefited tremendously from the OpenRISC are university researchers who explore new technologies in digital and mixed-signal development of integrated circuits. If a graduate student needs to research how custom hardware interacts with software quickly, it is possible to simulate and fabricate a processor without spending years developing a processor core with a corresponding C compiler. A researcher can test the system in FPGA, then submit a custom chip for fabrication within six months.

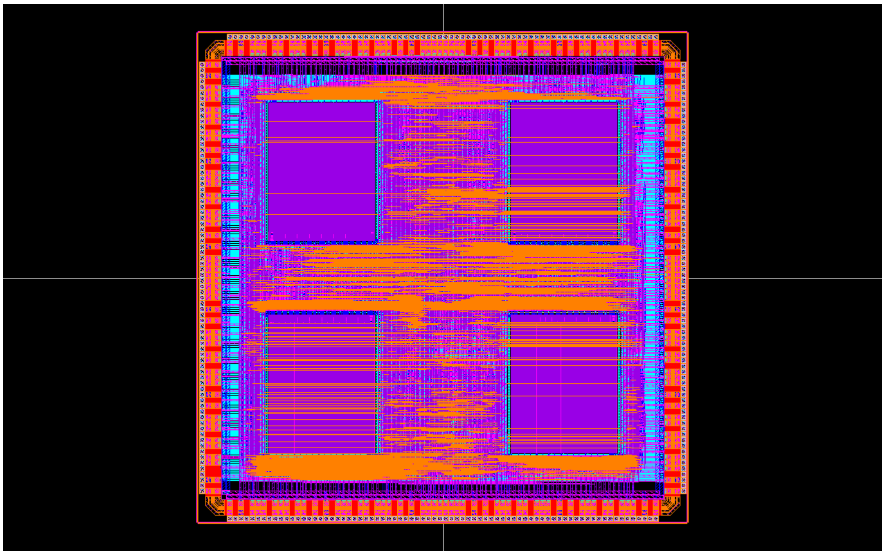


Figure 1. Layout of an OpenRISC 1200 processor in 0.18um technology implemented at the University of Tokyo. It has a serial port, Ethernet, I2C, 128kB of on-chip memory and a custom serial communication interface.

The Future

OpenRISC has a large following in the integrated circuit research community and the hardware hacker community. The OpenRISC/Linux development team recently submitted their patch for inclusion in the main Linux distribution. If all goes well, you will soon see the OpenRISC build target when you download version 3.1 of the Linux kernel. Work on the latest 3.1 OpenRISC kernel patch continues. If you feel inclined to help, a lot of testing is needed for the latest development version.

Will it be possible to purchase an OpenRISC ASIC in the future? Quite possibly. The team at **http://opencores.org** is soliciting donations for a custom ASIC implementation. Hardware hackers who decide to commercialize their FPGA projects may want a faster, lower-power version of OpenRISC when selling their applications. Another possibility is that open-source integrated circuit design will become available as well, making it possible for small groups of people, or even individuals, without millions of dollars, to submit custom chip designs to a manufacturer. The fabless semiconductor company has become a much more viable business model in the past 15 years.

The place where OpenRISC truly shines right now is in FPGA design. Anybody can purchase a single FPGA for as little as a few dollars or a prebuilt FPGA board for as little as $200 USD. This means that anybody with a little extra cash can build a custom embedded system with the OpenRISC. While processor cores that manufacturers provide confine you to their architecture, the OpenRISC allows you to choose the best FPGA for your project. Because Linux can run effectively on OpenRISC, you can include any number of open-source projects on your custom hardware. If this article has whet your interest for more, check the list of Resources for this article. Also, if you implement a project with Linux and OpenRISC, send me an e-mail to let me know! ■

James Tandon is a post-doctoral researcher at the VLSI Design & Education Center of Tokyo University, Japan. He has fabricated numerous digital and mixed-signal integrated circuits in technology nodes from 0.35um down to 65nm. His personal Web address is http://www.jamestandon.com.

Resources

Home Page for the Linux and GCC Development Projects: **http://www.openrisc.net**

Large Repository of Open-Source Hardware Cores: **http://www.opencores.org**

Home Page of the OpenRISC 1000 Project: **http://opencores.org/or1k/Main_Page**

Open-Source Verilog Simulator (good for simulating before implementation and very stable): **http://iverilog.icarus.com**

Verilog Programming Tutorial: **http://www.asic-world.com/verilog/veritut.html**

Demonstration of OpenRISC Controlling a Robot: **http://www.youtube.com/watch?v=Lv1Gow7WZxM**

Fixing Broken Protocols with NF_QUEUE

Broken protocols can be fixed on the fly with Netfilter's ability to direct packets to userspace programs. PAUL AMARANTH

Recently, one of my clients was experiencing problems with remote print servers. These print servers were on an internally NATted network connected to a central records system located across the state through the Internet. The print servers would not stay connected, dropping the connection after a few minutes. Investigation finally tracked this problem down to the keep-alive protocol used between the central system and the remote print servers. The keep-alive protocol employed a UDP packet with the source and destination IP addresses contained within the data. Normally, this would just mirror the addresses in the UDP header and would seem to be redundant. In this case, the server ignored the UDP header addresses and used only the internal addresses. When the packets went through NAT translation, the internal addresses were sent through unchanged, and the central server was attempting to reply to a nonroutable 10.xxx.xxx.xxx address.

Working with my client, we identified the problem and located documentation that specifically stated the protocol would not work with print servers behind NAT. For a number of reasons, my client was unable to move the print servers to a non-NATted environment, which left the problem of fixing the protocol.

Because all the keep-alive packets already were passing through the Linux firewall, that seemed to be the logical place to fix them. The NF_QUEUE facility in Netfilter turned out to be the perfect solution.

Netfilter and NF_QUEUE

Netfilter is part of the packet filtering framework within the Linux 2.4.x and 2.6.x kernels. As stated on the Netfilter home page: "Netfilter provides a set of hooks inside the Linux kernel that allows kernel modules to register callback functions with the network stack. A registered callback function is then called back for every packet that traverses the respective hook within the network stack."

The NF_QUEUE facility extends this ability to userspace, allowing packets to be directed using iptables rules to a userspace program. The program then can look at the packet and take action based on the packet content. The program might decide to accept or reject the packet, for example, allowing the firewall to filter packets based on content. The program also might decide to modify the packet and return it to Netfilter for further processing. It is this latter ability that allows broken protocols to be fixed on the fly.

The QUEUE facility initially was introduced into the 2.3 kernel and allowed for a single queue. This was changed to NF_QUEUE in the 2.6.14 and later kernels to allow for multiple queues with a 16-bit queue identifier, so it is possible to have up to 65,535 queues. If the queue number is left off the iptables rule, it will default to queue 0 and the behavior is equivalent to the older QUEUE behavior.

An important point to remember is this is a queue. The kernel queues up packets for processing in userspace, and there is finite space to buffer the packets. If it takes too long to process the packet, the queue will fill up and packets will be lost.

Although my situation and this example use IPv4, the NF_QUEUE facility is also available in the IPv6 Netfilter code and the ip6tables command. The details of mangling the packet change to reflect the protocol and headers that are involved are different, and there are slight differences in the ip6tables chain traversal, but the overall process remains substantially the same.

Because the packet processing takes place in userspace, you are not limited to writing the program in C. You can use any language you want, as long as there is a binding to the NF_QUEUE facility. At the time of this writing, in addition to C and C++, you can use Perl and Python to write your packet handler (see Resources).

In my case, I chose to write my routines in C. Because the firewall in question also serves as a gigabit router between two internal networks, supports a VPN gateway as well as handling almost a thousand iptables rules, I was interested in keeping overhead low. C was the natural choice.

Before using NF_QUEUE, it must be enabled in the kernel. If your kernel supports the config.gz option, you can use the following:

```
gzcat /proc/config.gz|grep -E "NETLINK|NFQUEUE"
```

and see if the configuration options listed below are set. If you do not have gzcat, it's just a hard link to gzip; many distributions seem to leave that out.

If NF_QUEUE is not configured, you'll have to configure and rebuild the kernel. The configuration parameters that you will need to set are the following:

```
CONFIG_NETFILTER_NETLINK=y
CONFIG_NETFILTER_NETLINK_QUEUE=y
CONFIG_NETFILTER_NETLINK_LOG=y
CONFIG_NETFILTER_XT_TARGET_NFQUEUE=y
```

Use your favorite kernel configuration tool, rebuild, install and reboot.

If not present, you also need to build and install libnfnetlink and libnetfilter_queue (see Resources).

Listing 1. NF_QUEUE Boilerplate Code in C

```
struct nfnl_handle *nh;
struct nfq_handle *h;
int ec, fd, rv;
char buf[1500];

// 1) Open library handle. For space reasons, the
//    error checking is not shown.

h = nfq_open();

// 2) Unbind existing nf_queue handler for AF_INET.
//    Ignore return code for 2.6.23 kernel.
//    See Resources for link.

ec = nfq_unbind_pf(h, AF_INET);

// 3) Bind the queue connection handle.

ec =  nfq_bind_pf(h, AF_INET);

// 4) Create queue, bind to queue 0 since that is
//    what the default QUEUE target in iptables
//    expects, specify callback function.

qh = nfq_create_queue(h, 0, &nfqueue_cb, NULL);

// 5) Set the amount of data to be copied to
// userspace for each packet sent to the queue.

nfq_set_mode(qh, NFQNL_COPY_PACKET, 0xffff);

// Get the netlink handle associated with the queue
// connection handle.

nh = nfq_nfnlh(h);

// Get a file descriptor for the netlink handle.

fd = nfnl_fd(nh);

// Packet loop.

while ((rv = recv(fd, buf, sizeof(buf), 0)) &&
        rv >= 0) {
  nfq_handle_packet(h, buf, rv);
  }
```

Listing 1 shows the standard boilerplate code for an NF_QUEUE packet handler. Steps 1–3 are basic setup. Step 4 creates a specified queue and binds the program to it. The queue identifier must match the queue number used in the iptables rules. This call also registers the callback function, which is where the packet actually is processed. Step 5 tells NF_QUEUE how much data is to be sent to the userspace program. The choice is none, all or just the packet metadata (information from NF_QUEUE, but no packet data).

This is pretty standard, except that under 2.6.23 kernels, step 2 will return an error, which may be safely ignored. The packet is not actually read using the recv() function; it is accessed by a callback function invoked by the nfq_handle_packet() function. In addition to the packet data, this allows access to additional NF_QUEUE metadata and permits re-injection of the packet as well as ACCEPT and DROP decisions to be made. The return code from the recv() call may be used to determine if the queue has filled and packets are being dropped.

Before the program exits, it should close gracefully by unbinding the queue with a call to nfq_destroy_queue(), followed by a call to nfq_close(). In my implementation, I elected to include a signal handler that closed any log files and unbound the queue on receipt of a SIGINT or SIGHUP signal.

Packet Mangling

The callback function is where the real action is. Here you have access to the entire packet,

Listing 2. Sample Callback Function That Will Dump a Packet in Hex

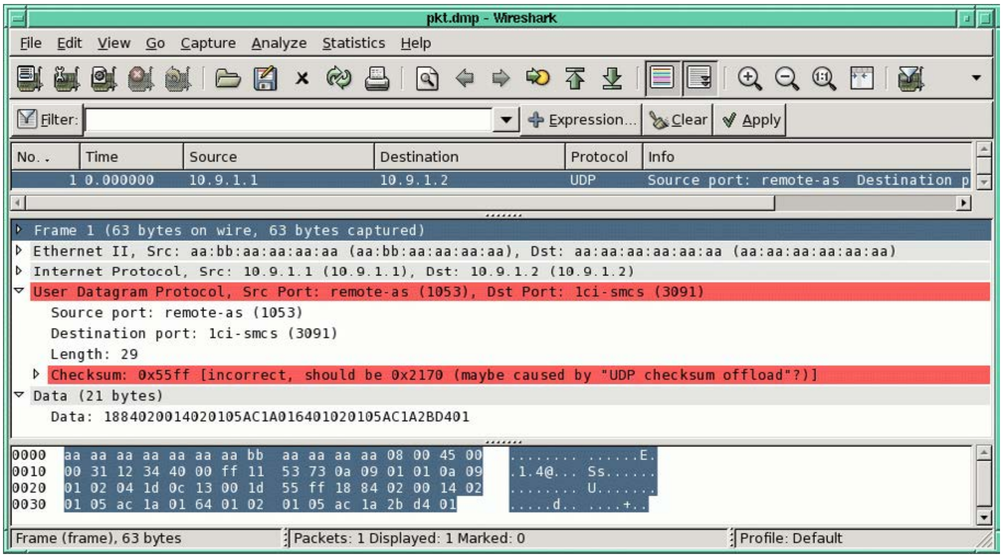
```
// Sample NF_QUEUE callback function.
static int nfqueue_cb(struct nfq_q_handle *qh,
                      struct nfgenmsg *nfmsg,
                      struct nfq_data *nfa,
                      void *data) {

    struct nfqnl_msg_packet_hdr *ph;
    int id = 0;
    int size = 0;
    int i;
    unsigned char *full_packet;
    unsigned char * c;
    struct in_addr ipa;
    char src_ip_str[20];
    char dst_ip_str[20];

ph = nfq_get_msg_packet_hdr(nfa);

if (ph) {

  // Print out metatdata.
  id = ntohl(ph->packet_id);
  fprintf(stdout,
        "hw_protocol = 0x%04x hook = %u id = %u\n",
         ntohs(ph->hw_protocol), ph->hook, id);

  // Retrieve packet payload.
  size = nfq_get_payload(nfa, &full_packet);

  // Get IP addresses in char form.
  ip = (struct iphdr *) full_packet;
  ipa.s_addr=ip->saddr
  strcpy (src_ip_str, inet_ntoa(ipa));
  ipa.s_addr=ip->daddr;
  strcpy (dst_ip_str, inet_ntoa(ipa));

  fprintf(stdout,
         "Source IP: %s   Destination IP: %s\n",
          src_ip_str, dst_ip_str);

  // Print out packet in hex.
  c = (unsigned char *)payload;
  for (i=0; i<size; ++i,++c) {
    fprintf (stdout, "%02x", (unsigned int)*c);
  }
  fprintf (stdout, "\n");

  // Done with packet, accept it.
  nfq_set_verdict(qh, id, NF_ACCEPT, 0, NULL);
  }
}
```

including headers and NF_QUEUE metadata. Listing 2 shows a simple callback routine that prints the source and destination IP addresses and dumps the packet contents in hex.

When you have finished processing the packet, the routine must return a verdict using the nfq_set_verdict() function. This can be NF_ACCEPT or NF_DROP (other verdicts are possible, but these two cover 99% of the cases). If the packet is unchanged, a zero value length parameter and null pointer can be returned. If the packet has changed, these parameters should point to the new packet. In any case, a call to nfq_set_verdict() must be made for every packet retrieved so the kernel can remove the packet and associated information from its internal data structures.

With access to the packet, combined with an understanding of the protocol, you can mangle the packet to fix almost any conceivable problem. In my case, I used a small lookup table to translate the internal IP addresses to an external address on packets being sent to the central system and the external IP address to the internal address on packets being returned to the print server.

A very important point with packet mangling is that any change to the packet contents will, almost certainly, change the packet checksum. The IP header checksum



```
No. .  Time       Source      Destination   Protocol  Info
     1 0.000000   10.9.1.1    10.9.1.2      UDP       Source port: remote-as  Destination p
```

```
▷ Frame 1 (63 bytes on wire, 63 bytes captured)
▷ Ethernet II, Src: aa:bb:aa:aa:aa:aa (aa:bb:aa:aa:aa:aa), Dst: aa:aa:aa:aa:aa:aa (aa:aa:aa:aa:aa:aa)
▷ Internet Protocol, Src: 10.9.1.1 (10.9.1.1), Dst: 10.9.1.2 (10.9.1.2)
▽ User Datagram Protocol, Src Port: remote-as (1053), Dst Port: 1ci-smcs (3091)
    Source port: remote-as (1053)
    Destination port: 1ci-smcs (3091)
    Length: 29
  ▷ Checksum: 0x55ff [incorrect, should be 0x2170 (maybe caused by "UDP checksum offload"?)]
▽ Data (21 bytes)
    Data: 188402001402010 5AC1A016401020105AC1A2BD401
```

```
0000  aa aa aa aa aa aa aa bb  aa aa aa aa 08 00 45 00   ........ ......E.
0010  00 31 12 34 40 00 ff 11  53 73 0a 09 01 01 0a 09   .1.4@... Ss......
0020  01 02 04 1d 0c 13 00 1d  55 ff 18 84 02 00 14 02   ........ U.......
0030  01 05 ac 1a 01 64 01 02  01 05 ac 1a 2b d4 01      .....d.. ....+..
```

```
Frame (frame), 63 bytes    Packets: 1 Displayed: 1 Marked: 0    Profile: Default
```

Figure 1. Not only does Wireshark tell you the checksum is wrong, it also tells you what it should be.

will be unchanged unless you modify the source or destination IP addresses. However, any change to the packet contents will change the checksum for UDP or TCP packets. When the kernel NATs the packet, it does not recompute the checksums. Instead, it optimizes the process by modifying only the current checksums with the difference between the old and new addresses. If the kernel does not use NAT, the checksum is not even inspected. This means the checksums must be correct when returning the packet to the kernel. Actually, with UDP packets, you can cheat by using a 0 checksum. The UDP protocol specification states the checksum is optional, and a 0 value indicates that you have not calculated it. That is not a recommended practice, particularly when traversing external networks, but you can get away with it. With TCP packets, this is not an option; the TCP header must contain a correct checksum. Any packet with an incorrect checksum will be dropped by the next network device it hits.

There are a number of ways to determine if your checksum is correct. The easiest is to look at the packet using a sniffer like Wireshark (Figure 1). Unlike tcpdump, which will print only the packet contents, Wireshark will verify the packet checksums and even tell you what they should be if they're not correct.

The UDP checksum is fairly easy to calculate, although it does involve constructing a pseudo-header containing the source IP, destination IP, UDP data length and the UDP protocol number. I was in a hurry and grabbed two different routines off the Net and found that both calculated incorrect checksums on a 64-bit platform. I finally had to rewrite one of the routines to generate correct checksums (see Resources for a download link). During development, you can use the checksum calculator URL in Resources to paste in a hex dump of your packet to verify your results.

NF_QUEUE Hooks

When NF_QUEUE activates a callback, along with the requested data is a parameter containing the hook that invoked it. The possible values of the hook are defined in netfilter_ipv4.h as:

```
NF_IP_PRE_ROUTING       0
NF_IP_LOCAL_IN          1
NF_IP_FORWARD           2
NF_IP_LOCAL_OUT         3
NF_IP_POST_ROUTING      4
```

The value of the hook tells you which iptables chain was employed to direct the packet to the callback. You can use different iptables commands with different chains to change the behavior of your program by checking the value of the hook parameter. You might direct packets from your internal Net to the PREROUTING chain and packets from the external Net to the POSTROUTING chain, for example.

Understanding how iptables behave is essential in picking the right hook. The URL for faqs.org listed in Resources has one of the clearest explanations I have found. Figure 2, adapted from this reference, illustrates the packet path through iptables. The top label in the ovals is the table name while the lower label identifies the chain.

In this case of mangling packets transiting the firewall, the LOCAL_IN and LOCAL_OUT hooks will not be used (they apply only to packets originating from or destined to the local host). That leaves three choices: PRE_ROUTING, FORWARD or POST_ROUTING (note that older kernels had more limited choices for the mangle table).

In this case where the IP header and actual source and destination addresses are not changed, any of the three choices would work. This might not be the case if you modify the source or destination addresses, which might affect subsequent routing decisions. If the destination address were changed to the local system, for example, you would be limited to the PREROUTING chain. If you want to modify the packet after all filtering has been done and intercept any locally generated packets as well, you would use the POSTROUTING chain. The FORWARD chain is useful for packets transiting the system.

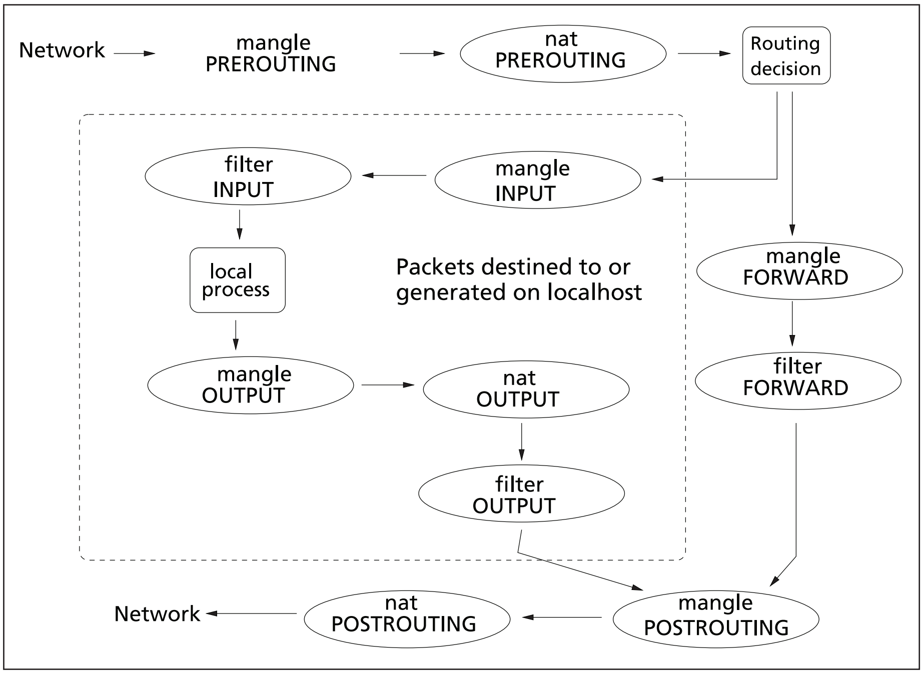


Figure 2. Packet flow through iptables tables and chains. Table names are in lowercase, chain names in uppercase.

iptables

At this point, you have a kernel with NF_QUEUE enabled, the nfqueue and nfnetlink libraries are installed, you have a packet sniffer ready to go, and your program is compiled and ready to test. How do you connect it to iptables?

The iptables target QUEUE will send any matching packets to a program that has registered itself for that queue, which defaults to Queue 0. An optional `--queue-num` parameter may be used to specify a nondefault queue. It is also possible to use a `--queue-balance` parameter with recent kernels that specifies a range of queues. This allows multiple instances of a userspace program on multicore architectures to improve throughput. If no program has registered itself for the queue, the QUEUE target is equivalent to DROP.

iptables has four built-in tables: filter, nat, mangle and raw. Each table supports different chains (Figure 2). The filter table, which is the default for the iptables command, is useful if your program is making an accept or deny decision on the packet, thus allowing firewall filtering based on content. The nat table is used for address translation, and the raw table, a recent addition, is used only for setting marks on packets that should not be handled by the conntrack system. The table to use when altering packets is, as the name implies, the mangle table. Listing 3 illustrates a few iptables commands that will set up NF_QUEUE forwarding for UDP packets destined for port 1331. In practice, this can become more complicated if you limit the source and destination addresses using additional iptables commands or include other selection criteria.

Listing 3. iptables Commands to Route UDP Packets on Port 1331 to NFQUEUE

```
// Set up a new chain in the mangle table.
 iptables -t mangle -N PktMangle
// In the mangle FORWARD chain, route UDP packets
// to the new chain.
 iptables -t mangle -A FORWARD -p udp -m udp \
    --dport 1331 -j PktMangle
// Log the packet and invoke the queue facility.
 iptables -t mangle -A PktMangle -j LOG \
    --log-level info  --log-prefix "PktMangle rule"
 iptables -t mangle -A PktMangle -j QUEUE
```

Testing

The NF_QUEUE application must run as root, at least when setting up the queue. Otherwise, you will get a -1 return from the unbind or bind calls.

In my case, the print server generated a ready supply of keep-alive packets, so I had no need of a packet generator. In the general case, you will need some way of generating test packets to verify that your system is operating correctly. A plethora of packet generators are available. One example is PackETH, which is a GUI-based packet generator that is quite easy to use, although still a little unfinished.

Another necessary requirement is the ability to capture packets both before and after processing to verify the output packet is correct. This can be easily done using tcpdump or Wireshark to view packets on the input and output interfaces of the test system. Wireshark may be used directly if you have X libraries available on the test system. In my case, since I was running the packets through a production system, I used tcpdump and then viewed the packet dump files off-line with Wireshark.

Conclusions

The project turned out to be very successful. Fixing the keep-alive packets as they traversed the firewall resolved the problem without requiring any configuration changes on either endpoint. It was a completely transparent solution and my client was very happy with the result.

After being in place for a while, a crisis ensued when the remote system IP was changed. The printers stopped, because the remote IP was hard-configured into the print servers, and local personnel were unavailable to reconfigure them. With the addition of a DNAT firewall rule and a tweak to the protocol dæmon to fix the server address within the packet (all done remotely), the printers came back on-line in time to run payroll.

TCP protocols also can be fixed using this approach. Because TCP is a connection-oriented protocol rather than a datagram, the program will need to keep some state information as it processes the data stream. This is a little more complicated than a UDP protocol, but not unreasonable in practice.

I've posted a small sample NF_QUEUE packet sniffer on my Web site containing complete build and execution directions. It's fairly basic, but it allows you to get a hex dump of UDP or TCP streams determined by iptables rules and can serve as a basic framework if you're building an NF_QUEUE handler.■

Paul Amaranth (paul@auroragrp.com) is a Principal Consultant at Aurora Group, Inc., where he builds secure systems (including Linux firewalls), does software development and handles the odd bit of system administration. He's been involved with computers much longer than he cares to remember.

Resources

Netfilter Hacking HOWTO: **http://netfilter.org/documentation/HOWTO/netfilter-hacking-HOWTO.html**

Netfilter Error Return in 2.6.23 Kernels: **http://www.spinics.net/lists/netfilter/msg42063.html**

libnfnetlink: **http://www.netfilter.org/projects/libnfnetlink/index.html**

libnetfilter_queue: **http://www.netfilter.org/projects/libnetfilter_queue/index.html**

libnetfilter_queue man Page: **http://code.google.com/p/nattt/wiki/libnetfilter_queue**

NF_QUEUE Language Bindings: **http://www.nufw.org/projects/nfqueue-bindings/wiki**

Correct UDP Checksum Routine: **http://www.auroragrp.com/downloads**

Sample nfq_sniff Application to Dump Packets: **http://www.auroragrp.com/downloads**

tcpdump: **http://www.tcpdump.org**

Wireshark: **http://www.wireshark.org**

PackETH: **http://packeth.sourceforge.net**

On-line Checksum Generator: **http://moat.nlanr.net/Software/HEC/hexhec.html**

iptables Packet Flow: **http://www.faqs.org/docs/iptables/traversingoftables.html**

KYLE RANKIN

BILL CHILDERS

Zoning Out

This month, Bill describes that incredible sinking feeling.

Sometimes events and equipment conspire against you and your team to cause a problem. Occasionally, however, it's lack of understanding or foresight that can turn around and bite you. Unfortunately, this is a tale of where we failed to spot all the possible things that might go wrong.

Flashback...

It was 2006, and we were just getting our feet wet with piloting a new server architecture for our company. We'd just received our first fully populated Hewlett-Packard blade chassis (a P-Class chassis with eight dual-core blades, for those of you who're savvy with that type of gear), a new EMC Storage Area Network (SAN) and three VMware ESX licenses. We had just finished converting a fair amount of the development network over to the VMware environment using a Physical-to-Virtual (P2V) migration, and things were going quite well. Matter of fact, many of the people in the company didn't quite understand exactly the improvements we were making to the systems, but they did notice the performance boost of going from machines that were something like single-processor Pentium 4-class servers with IDE disks to a dual-core Opteron where the storage was backed by the speed of the Fibre Channel SAN. In all, things were going quite well, and the feedback we'd received to date fueled a rather rapid switch from the aging physical architecture to a much faster virtual machine architecture.

Background

Before we dive into the story, a couple bits of background information will become very important later on. As I said, we'd received eight dual-core blades, but only three of them at that time were set aside for VMware hosts. The rest were slated to become powerful physical machines—Oracle servers and the like. All these new blades were configured identically: they each had 16GB of RAM, two dual-core

Opteron processors, two 300GB disks and Fibre Channel cards connected to the shiny new EMC SAN. With respect to the SAN, since we were devoting this SAN strictly to the blade servers, the decision was made not to add the complexity of zoning the SAN switch. (Zoning a SAN switch means that it is set up to allow only certain hosts to access certain disks.) The last tidbit relates to kickstart.

Both Kyle and I have written a few articles on the topic of kickstarting and automated installation, so by now you're probably aware that we're fans of that. However, this was 2006, and we both were getting our feet wet with that technology. We'd inherited a half-set-up kickstart server from the previous IT administration, and we slowly were making adjustments to it as we grew more knowledgeable about the tech and what we wanted it to do.

[Kyle: Yes, the kickstart environment technically worked, but it required that you physically walk up to each machine with a Red Hat install CD, boot from it, and manually type in the full HTTP path to the kickstart file. I liked the idea of kicking a machine without getting up from our desks, so the environment quickly changed to PXE booting among a number of other improvements. That was convenient, because those blades didn't have a CD-ROM drive.]

Getting back to the story...we'd moved a fair amount of the development and corporate infrastructure over to the VMware environment, but we still had a demand for high-powered physical machines. We'd gotten a request for a new Oracle database machine, and since they were the most powerful boxes in the company at the time, with connections to the Storage Area Network, we elected to make one of the new blades an Oracle box.

As my imperfect memory recalls, Kyle fired up the lights-out management on what was to be the new Oracle machine and started the kickstart process, while I was doing something else—it could have been anything from surfing Slashdot to filling out some stupid management paperwork. I don't remember, and it's not critical to the story, as about 20 minutes after Kyle kickstarted the new Oracle blade, both of our BlackBerries started beeping incessantly.

[Kyle: Those of you who worked (or lived) with us during that period might say, "Weren't your BlackBerries always *beeping incessantly?" Yes, that's true, but this time it was different: one, we were awake, and two, we actually were in the office.]*

Trouble in Paradise

We both looked at our BlackBerries as we started getting "host down" alerts from most of the machines in the development environment. About that time, muttering could be heard from other cubicles too: "Is the network down? Hey, I can't get anywhere." I started getting that sinking feeling in the pit of my stomach as Kyle and I started digging into the issue.

Sure enough, as we started looking, we realized just about everything was down. Kyle fired up the VMware console and tried restarting a couple virtual machines, but his efforts were met with "file not found" errors from the console upon restart. File not found? That sinking feeling just accelerated into free-fall. I started looking along with Kyle and realized that all the LUNs (disks where the virtual machines reside) just flat-out stopped being available to each VM host.

[Kyle: It's hard to describe the sinking feeling. I was relatively new to SAN at the time and was just realizing how broad a subject it is in its own right. SAN troubleshooting at a deep level was not something I felt ready for so soon, yet it looked like unless we could figure something out, we had a large number of servers that were gone for good.]

I jumped on the phone and called VMware while Kyle continued troubleshooting. After a few minutes on the line, the problem was apparent. The LUNs containing the virtual machines had their partition tables wiped out. We luckily could re-create them, and after a quick reboot of each VM host, we were back in business, although we were very worried and confused about the issue.

[Kyle: So that's why that sinking feeling felt familiar. It was the same one I had the first time I accidentally nuked the partition table on my own computer with a bad dd command.]

Our worry and concern jumped to near-panic when the issue reared its head a second time, however, under similar circumstances. A physical machine kickstart wound up nuking the partition table on the SAN LUNs that carried the virtual machine files. We placed another call to VMware, and after some log mining, they determined that it wasn't a bug in their software, but something on our end that was erasing the partition table.

A Light Dawns

Kyle and I started to piece together the chain of events and realized that each time this occurred, it was preceded by a kickstart of a blade server. That led us to look at the actual kickstart control file we were using, and it turned out there was one line in there that caused the whole problem. The directive `clearpart --all --initlabel` would erase the partition table on all disks attached to a particular host, which made sense if the server in question had local disks, but these blades were attached to the SAN, and the SAN didn't have any zoning in place to protect against this. As it turns out, the system did exactly what it was set up to do. If we had placed the LUNs in zones, this wouldn't have happened, or if we'd have audited the kickstart control file and thought about it in advance, the problem wouldn't have happened either.

[Kyle: Who would have thought that kickstart would become yet another one of those UNIX genie-like commands like dd that do exactly *what you say. We not*

only placed the LUNs in zones, but we also made sure that the clearpart directive was very specific to clear out only the disks we wanted—lucky for us, those HP RAID controllers show up as /dev/cciss/ devices, so it was easy to write the restriction.]

Lessons Learned

We learned a couple things that day. First was the importance of zoning your SAN correctly. The assumption we were operating under—that these boxes would all want to access the SAN and, therefore, zones were unnecessary—was flat-out wrong. Second, was the importance of auditing and understanding work that other sysadmins had done prior and understanding how that work would affect the new stuff we were implementing. Needless to say, our SAN always was zoned properly after that.■

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

Bill Childers is an IT Manager in Silicon Valley, where he lives with his wife and two children. He enjoys Linux far too much, and he probably should get more sun from time to time. In his spare time, he does work with the Gilroy Garlic Festival, but he does not smell like garlic.

OLPC | Arduino | Trim-Slice | vDSO | Elmer | MPD
LINUX JOURNAL
ONE LAPTOP PER CHILD STATUS REPORT
USE YOUR PHONE TO LOCK YOUR SCREEN AUTOMATICALLY
HACK THIS
HOW TO: ADD A CUSTOM vDSO TO A LINUX KERNEL
Get Started Programming the Arduino
Create a Unified Boot Environment on a USB Drive
NETWORKING

DOC SEARLS

Reality Fidelity Field

We have to die. Code doesn't.

I'm writing this the day after Steve Jobs died, and the man's famous "reality distortion field" has surely survived him. It will be months or years before anybody can get a good-enough handle on what the guy did, and meant. Meanwhile his death looms, larger than life. As it should, because death and life both matter, and both need each other.

Life, far as we can tell, exists only on the surface of one planet of one star among a hundred billion in our galaxy, which is one of a hundred billion other galaxies. The chance of life happening elsewhere exceeds zero, but not the chance of finding out for sure in our lifetimes. Our deathtimes are another matter. We are all dead for most of eternity.

But if life exists elsewhere, it depends on death no less than does our own. That's because death is more than the end or the absence of life. It is a condition required by life. The living eat the dead. The living also heat, forge, manufacture, build and destroy with the dead. Except for wind, water, sun and radioactive elements, we produce all our electricity with products and by-products of death. Oil, coal, wood and gas come straight from death. So do asphalt, concrete, plastics and all natural and artificial fabrics. Without death, we would not have handy forms of geology known only on Earth: limestone, marble, travertine, chert, diatomite. None of the world's most beautiful caves would have formed, and there would be no stalactites or stalagmites. Without death, no great pyramids, no Notre Dame, no Pantheon, no Parthenon.

Death uses us to make more of itself. It wants us out of the way. Sooner or later, we are obliged to cease living, and to burn or flush our remains into death's system. Steve Jobs knew that, years before he departed. Here's what he said in a commencement address to graduating students at Stanford in 2005:

> No one wants to die. Even people who want to go to heaven don't

> want to die to get there. And yet death is the destination we all share. No one has ever escaped it. And that is as it should be, because Death is very likely the single best invention of Life. It is Life's change agent. It clears out the old to make way for the new. Right now the new is you, but someday not too long from now, you will gradually become the old and be cleared away. Sorry to be so dramatic, but it is quite true.
>
> Your time is limited, so don't waste it living someone else's life. Don't be trapped by dogma—which is living with the results of other people's thinking. Don't let the noise of others' opinions drown out your own inner voice. And most important, have the courage to follow your heart and intuition. They somehow already know what you truly want to become. Everything else is secondary.

As I write this, much is being said about how the tech world will miss Steve's creative muse, his leadership, his taste and the rest of it. Which it will. But "what would Steve do" is exactly the kind of dogma trap the man warned us about. So is being like Steve, or like anybody other than ourselves.

Like lots of other people, I appreciated a lot of what Steve Jobs created, even as

Advertiser Index

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

ADVERTISER	URL	PAGE #
AceComputer	http://www.acecomputers.com	67
Emac, Inc.	http://www.emacinc.com	23
iXsystems	http://www.ixsystems.com	7
JC Technology	http://www.jc-technology.com	31
Logic Supply	http://www.logicsupply.com	35, 69
Lullabot	http://store.lullabot.com	2
Microway	http://www.microway.com	3, 70, 71
Nagios	http://www.nagios.com	58, 59
Opal Events	http://www.opalevents.org	75
RackMountPro	http://www.rackmountpro.com	11
SCALE	http://www.socallinuxexpo.org/scale10x	85
SharePoint Technology Conference	http://www.sptechcon.com	121
Silicon Mechanics	http://www.siliconmechanics.com	21
USENIX Lisa	http://www.usenix.org/lisa11/lj	93

Most of us in the Linux community have long made a point of working as far as possible outside of Steve Jobs' shadow, as well as those of Bill Gates and other industrial giants of the computing and networking worlds.

his control freakiness also drove me nuts. But he did what only he could do, and now he's gone. His shadow remains long. Yet we have to step out of it.

Most of us in the Linux community have long made a point of working as far as possible outside of Steve Jobs' shadow, as well as those of Bill Gates and other industrial giants of the computing and networking worlds. What I'm wondering, now that Steve's dead and Bill has left the building, is what more we can do with Linux, free software and open source, than we would if they were still around.

There are huge opportunities, especially with mobile devices. It's been fun to see Samsung rolling out kernel code (**http://www.androidpolice.com/2011/09/21/att-samsung-galaxy-s-ii-kernel-source-code-released**) for its Android devices, even as Google closes doors on Android 3.0 "Honeycomb" (**http://www.zdnet.com/blog/google/google-android-30-honeycomb-open-source-no-more/2845**).

I even take heart in the weirdness of Nokia once again moving toward Linux, this time for its low-end phones, through something called Meltremi (**http://thenextweb.com/mobile/2011/09/29/nokias-meltemi-project-tipped-to-bring-new-low-end-linux-os-to-the-next-billion**). (What was wrong with Maemo?)

But a strength of Linux has always been its non-corporate nature. Android might belong to Google, but Linux doesn't belong to anybody. Linux's sole purpose is to be useful. What makes us keep improving it is a deeply felt need to maximize that usefulness. That usefulness outlives us, but not because it's dead.

See, free and open code is a kind of living building material. It's like wood that's still quick. And nothing needs to die for it to keep improving. It's a product of life that lives to support more life. All it needs is to be used, patched and re-used. As creators, the roles are reversed: code's gods are mortal, but code doesn't have to be.■

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

The Best SharePoint Training

Register Early and SAVE!

SPTechCon

The SharePoint Technology Conference

Feb. 26-29, 2012 → San Francisco

Downtown Location!
San Francisco Hilton

Check out more than 55 exhibiting companies!

Choose from over 90 Classes & Workshops!

"Great content and speakers."
—Dan Stolts, IT Pro Evangelist, Microsoft

"I really enjoyed SPTechCon. It was intense, like being in a parallel universe. It's a great place to find answers to problems, issues and concerns; everyone really wants to help! "
—Bisi Adebesin, Business Analys, Actionet

"Great place to get a lot of knowledge in a short period of time."
—Lola Flippo, Sr. Business Solutions Architect, Medseek

Follow us at twitter.com/SPTechCon
A BZ Media Event

www.sptechcon.com