

LZO Compression | Vagrant | Raspberry Pi | Sam

FREE TO
SUBSCRIBERS
EPUB, Kindle, Android, iPhone & iPad editions

LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

RASPBERRY PI N900

SET UP
LOCAL MIRROR
TO SAVE TIME
AND BANDWIDTH

AUGUST 2012 | ISSUE 220 | www.linuxjournal.com

MANAGE
MULTIPLE
DEVELOPMENT
ENVIRONMENTS
WITH VAGRANT

USING LZO
COMPRESSION
IN HADOOP

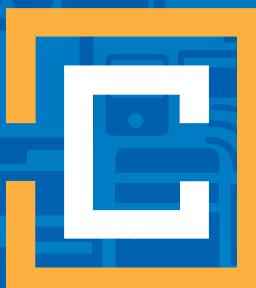
CREATE
VFS MODULES
TO EXTEND
SAMBA'S
FUNCTIONALITY

LEARN ABOUT
SHELL ALIASES
AND FUNCTIONS

+ AN EXCERPT FROM
DOC SEARLS'
NEW BOOK
*THE INTENTION ECONOMY:
WHEN CUSTOMERS TAKE CHARGE*

LINUX
AT WORK

TARSNAP: HIGH-SECURITY,
COMMAND-LINE-FRIENDLY
AUTOMATED BACKUPS



CloudOpen

A LINUX FOUNDATION EVENT

August 29 - 31, 2012
Sheraton Hotel & Marina, San Diego, CA

Introducing the first ever CloudOpen conference, **co-located with the 4th Annual LinuxCon North America!** CloudOpen is a conference celebrating and exploring the open source projects, technologies and companies who make up the cloud.

Attend Both Events For One Low Registration Fee!

Keynote Speakers

Chris Aniszczyk

*Open Source Manager
Twitter*

Rob Chandhok

*President
Qualcomm Innovation Center*

Phil McKinney

*Author of
"Creating Killer Innovations"*

More Keynote Speakers To Be Announced...

Additional Features

The popular **Linux Kernel Panel** with evening events sponsored by **Citrix, Intel and Qualcomm!**



LINUXCON
NORTH AMERICA 2012



THE
LINUX
FOUNDATION



**USE THIS DISCOUNT CODE
AND SAVE 15%: '12LCPR015'**

go.linuxfoundation.org/cloudopen



visit us at www.siliconmechanics.com or call us toll free at 888-352-1173

RACKMOUNT SERVERS STORAGE SOLUTIONS HIGH-PERFORMANCE COMPUTING

**"Just because
it's badass,
doesn't mean
it's a game."**

Pierre, our new Operations Manager, is always looking for the right tools to get more work done in less time. That's why he respects NVIDIA® Tesla® GPUs: he sees customers return again and again for more server products featuring hybrid CPU / GPU computing, like the Silicon Mechanics Hyperform HPCg R2504.v3.

We start with your choice of two state-of-the-art processors, for fast, reliable, energy-efficient processing. Then we add four NVIDIA® Tesla® GPUs, to dramatically accelerate parallel processing for applications like ray tracing and finite element analysis. Load it up with DDR3 memory, and you have herculean capabilities and an 80 PLUS Platinum Certified power supply, all in the space of a 4U server.



When you partner with Silicon Mechanics, you get more than stellar technology - you get an Expert like Pierre.

Expert included.

CONTENTS

AUGUST 2012
ISSUE 220

LINUX AT WORK

FEATURES

64 A Guide to Using LZO Compression in Hadoop

How to start implementing MapReduce programs that handle LZO compressed data.

Arun Viswanathan

76 Introducing Vagrant

Manage multiple development environments easily using Vagrant.

Jay Palat

86 Writing Samba VSF Modules

Extend Samba's functionality with VSF Modules

Richard Sharpe



ON THE COVER

- Raspberry Pi/N900 Hacks, p. 46
- Set Up a Local Mirror to Save Time and Bandwidth, p. 52
- Manage Multiple Development Environments with Vagrant, p. 76
- Using LZO Compression in Hadoop, p. 64
- Create VFS Modules to Extend Samba's Functionality, p. 86
- Learn about Shell Aliases and Functions, p. 42
- Tarsnap: High-Security, Command-Line-Friendly Automated Backups, p. 104
- An Excerpt from Doc Searls' New Book *The Intention Economy: When Customers Take Charge*, p. 117

Cover Image: © Can Stock Photo Inc. / NexusPlexus

COLUMNS

34 Reuven M. Lerner's At the Forge

Database Integrity and
Web Applications

42 Dave Taylor's Work the Shell

Scripting Lite: Shell Aliases
and Functions

46 Kyle Rankin's Hack and /

N900 with a Slice of Raspberry Pi

52 Shawn Powers' The Open-Source Classroom

Mirror, Mirror, Down the Hall

114 Doc Searls' EOF

Debugging Free Markets That
Still Aren't

117 "Your Choice of Captor" an Excerpt from Doc Searls' *The Intention Economy*

INDEPTH

104 Tarsnap: On-line Backups for the Truly Paranoid

Tarsnap's Linux-friendly encrypted
backup service makes keeping
your data in the cloud a snap.

Andrew Fabbro

IN EVERY ISSUE

8 Current_Issue.tar.gz

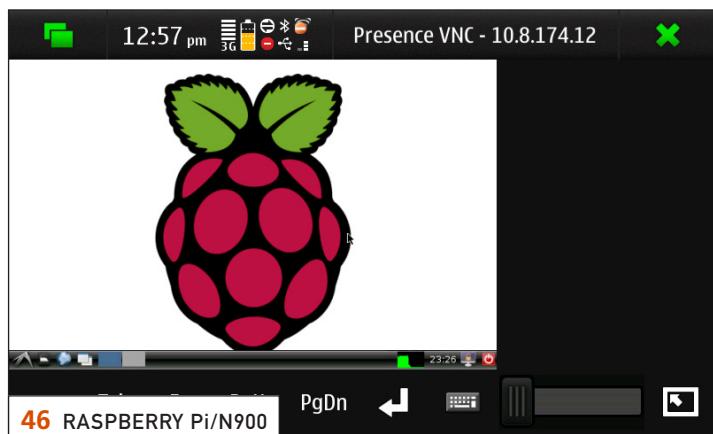
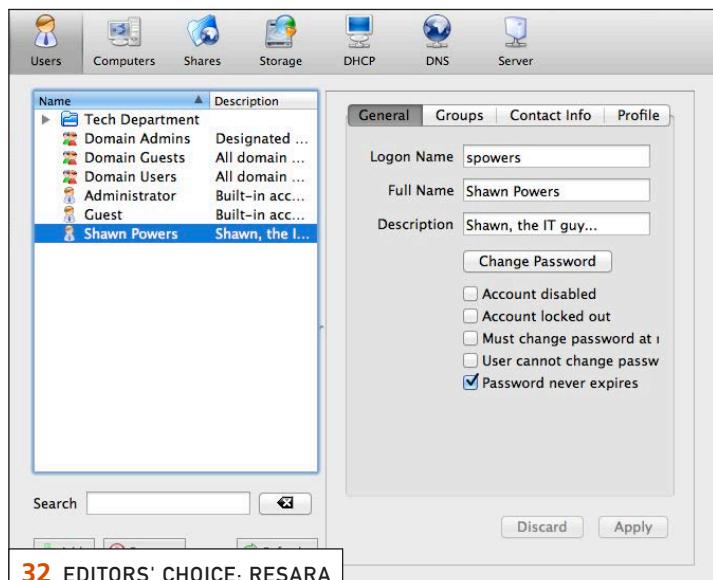
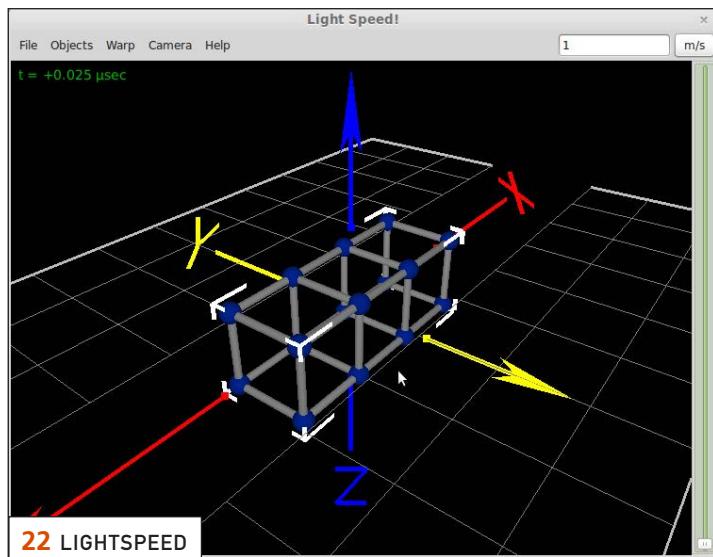
10 Letters

18 UPFRONT

32 Editors' Choice

60 New Products

123 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

| | |
|--------------------|-----------|
| Proofreader | Geri Gale |
|--------------------|-----------|

| | |
|------------------|--|
| Publisher | Carlie Fairchild publisher@linuxjournal.com |
|------------------|--|

| | |
|----------------------------------|---|
| Advertising Sales Manager | Rebecca Cassity rebecca@linuxjournal.com |
|----------------------------------|---|

| | |
|----------------------------|---|
| Associate Publisher | Mark Irgang mark@linuxjournal.com |
|----------------------------|---|

| | |
|--------------------|--|
| Webmistress | Katherine Druckman webmistress@linuxjournal.com |
|--------------------|--|

| | |
|-------------------|---|
| Accountant | Candy Beauchamp acct@linuxjournal.com |
|-------------------|---|

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

PO Box 980985, Houston, TX 77098 USA

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 Falty • Dennis Franklin Frey • Alicia Gibb
Victor Gregorio • Philip Jacob • Jay Kruizinga • 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.

TrueNAS™ Storage Appliances

Harness the Cloud

Unified. Scalable. Flexible. Storage.

As IT infrastructure becomes increasingly virtualized, effective storage has become a critical requirement. iXsystems' TrueNAS Storage appliances offer high-throughput, low-latency backing for popular virtualization programs such as Hyper-V, VMWare, and Xen. TrueNAS hybrid storage technology combines memory, NAND flash, and traditional hard disks to dramatically reduce the cost of operating a high performance storage infrastructure. Each TrueNAS appliance can also serve multiple types of clients simultaneously over both iSCSI and NFS, making TrueNAS a flexible solution for your enterprise needs.

For growing businesses that are consolidating infrastructure, the **TrueNAS Pro** is a powerful, flexible entry-level storage appliance. iXsystems also offers the **TrueNAS Enterprise**, which provides increased bandwidth, IOPS and storage capacity for resource-intensive applications.

Call 1-855-GREP-4-IX, or go to www.ixsystems.com



- ✓ Supports iSCSI or NFS exports simultaneously
- ✓ Compatible with popular Virtualization programs such as Hyper-V, VMware, and Xen
- ✓ 128-bit ZFS file system with up to triple parity software RAID

TrueNAS Pro Features

- One Six-Core Intel® Xeon® Processor 5600 Series
- High Performance Write Cache
- Up to 480GB MLC SSD Cache
- Up to 320 TB SATA capacity
- Quad Gigabit Ethernet
- 48GB ECC Memory

TrueNAS Enterprise Features

- Two Six-Core Intel® Xeon® Processors 5600 Series
- Extreme Performance Write Cache
- Up to 640GB SLC or MLC High Performance ioMemory
- Up to 500TB SATA or 320TB SAS capacity
- Dual Ten Gigabit Ethernet
- 96GB ECC Memory





SHAWN POWERS

Water Coolers, Cubicles, Committee Meetings and a Penguin

One of these things doesn't belong in the workplace. If you ask most people in the business world, they'd say a penguin is a silly thing to keep at work. Those of us in the server room, however, just snicker at such foolishness. I'll take Linux over a committee meeting any day! This month, we get to see Linux at work.

You might have noticed the past few months we've chosen a product or program as "Editors' Choice" for the issue. It's a great way for us to highlight something we find particularly awesome. This month, I introduce Resara. Samba 4 has been in active development for years, but the folks at Resara package the current build into a very functional Microsoft Active Directory replacement. Their community version is full-featured and

free—it's definitely worth checking out.

Reuven M. Lerner talks about a particularly serious issue this month, as he discusses database integrity. Databases often are overlooked as boring, but we rely on them constantly. Having viable, accurate data is something we assume, but Reuven shows us how to make that assumption possible. Dave Taylor follows up with a tutorial on how to use shell aliases and functions. We often take the simple tweaks in our .bashrc file for granted, but Dave not only opens it up, he also shows how to add tweaks of our own.

The Raspberry Pi device is something most people in our circles are familiar with, even if we haven't been able to get our hands on one! Kyle Rankin, as usual, takes something awesome and makes it even more so. Combining his Nokia N900 and the

Raspberry Pi, Kyle is able to interface directly with the fancy new device. Whether you're interested in creating a super-efficient media center or just like playing with embedded devices, Kyle makes it simple.

My Open-Source Classroom column this month is all about smoke and mirrors—without the smoke. Although certainly not practical for everyone, sometimes having a local repository mirror is useful. I explain how to mirror CentOS and Ubuntu. Be sure to bring lots of storage space!

When we think about compression, it usually means archiving or resource conservation. Arun Viswanathan discusses LZO compression in Hadoop this month, which uses saved space to decrease disk read times. Since LZO can be decompressed very quickly, the advantages in data transfer times are significant. Arun has code samples and usage examples that really show off the benefits.

Jay Palat found a Vagrant he really likes. If you've ever deployed an application only to have it work on some systems but not others, you'll likely agree with him. Vagrant is a tool for providing a virtualized development environment so you can test applications on multiple systems without actually creating the entire test environment. It's certainly possible to virtualize different types of installation environments, but Vagrant allows you to test those environments with a single tool.

Linux has a great virtual filesystem layer that allows the seamless use of underlying filesystems. Most people don't know that Samba does the same thing for its

fileshares. Richard Sharpe shows how to set up Samba VFS Modules, giving us the advantages of different filesystem features via Samba shares. Richard guides us through an example that really shows how to leverage this little known feature of Samba.

Cloud computing is here to stay, and while the terminology and concepts are new, our needs as end users haven't changed very much. Andrew Fabbro teaches us to use Tarsnap, which uses familiar command-line commands to manipulate tar files in the cloud. Amazon S3 and EC2 are the de facto standard for cloud-based computing, and Tarsnap allows manipulation of Amazon services using the handy-dandy command line. If you manage cloud services, you'll want to check out Tarsnap.

Doc Searls ends our issue this month, which is usually how our magazine ends. This time, however, he shares a chapter from his new book *The Intention Economy*, which is a brilliant piece on free markets and the Internet. Are we slaves to Internet companies? Does the Internet work for us, or are we slaves of on-line captors? Be sure to read the excerpt, and be sure to enjoy this entire issue of *Linux Journal*. Whether you're interested in new products, coding examples, tech tips or cool projects, this issue aims to please. ■

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

letters



Digital Ergonomics

I'm considering renewing my subscription. My main problem with the magazine is that the digital format is not set

up for digital devices. Digital devices do not have letter page size and do not have 600dpi resolution, and many laptops do not have a portrait aspect ratio. What does this mean? Well, in order to view a page, you have to zoom in. Once zoomed in, you have to scroll up and down to view a single page. That is annoying. Compare it with the Web. You start at the top and scroll down because most Web pages are single-column text. They are digitally ergonomic, whereas your magazine's PDF is not. Please, I beg you, fix it by going to a single-column PDF or, better yet a, downloadable HTML .tgz. HTML would allow readers to set the single-column size to what works best for their devices.

—Mark Ramsell

The main PDF has changed slightly

since the digital transition. It "fits" on a tablet-size screen better. Laptops, I agree, don't lend themselves to reading a portrait document very well. In that case, I recommend using an EPUB reader, which allows for flowing text (a couple issues ago I wrote about a great Firefox plugin that does it). Otherwise, the interactive on-line version of the magazine formats as a two-page spread, which looks good on a computer screen. Hopefully, one of those formats will work well for you. The good news is they're all included with your subscription!—Ed.

Digital Library

I have to admit, I am quite messy at keeping track of stuff, and things that have been added to my list of untrackable stuff are e-books. I have an old Android tablet, a new Android phone, a laptop with Windows 7/ OpenSUSE 12.1 and three more PCs that I usually work on. That's a lot of devices, and because of this, except for my work PC, I don't spend a lot of time at any given device. I think the PDF and .mobi formats are great for e-books like the magazine; however, it takes precious time to find the last page that I read. I am currently using Dropbox to share the files between all the computers, but I can't help wondering if there is a

better way to do it and keep the reading progress tracked.

I like to read on my regular PC because of the bigger monitor, but I rarely have time at work (I read when I need to wind down), but when I am waiting for something, I can read on my phone. Before going to bed, the tablet is great. At the moment, I am reading different things in each device, because I don't feel like wasting time searching for the last spot that I was on. I figure I am not the only one; other people must have encountered this problem. Keep up the good job!

—David

David, I have the same issue with e-books—not just Linux Journal, but my entire collection. For something I want to read cover to cover (and keep my place), I find the Kindle reader to do a nice job. You can upload the .mobi version of Linux Journal to your Kindle account as a personal document, and it will use WhisperSync to keep your spot on multiple devices. The only thing that frustrates me is that personal documents can't be read with the cloud reader, so reading on my screen isn't possible yet. Hopefully Amazon will lift that restriction!—Ed.

Downloading PDFs via Python

Thanks for the digital edition of *Linux Journal*! It's really nice to have full-text search on all the issues, thanks to our usual desktop search tools.

The other day I wanted to download all the back issues for my archives on a new PC, so got an old Python script and edited it to download them, as an alternative to wget's black magic. Also, it's easy to edit it to get only the latest issue and run in a cron job.

You can get it here, if you want; it's far from perfect, but it works:

<http://gerlos.altervista.org/files/dljget.py>

Obviously, it needs to be edited: you need to put your download page link and the download path in the script.

—gerlos

Thanks gerlos! This is exactly what many of us have been hoping for. The script works well, and it's easily modifiable for .epub or .mobi. gerlos is officially hero of the day! (Note: you need to make sure you have the python-bs4 package installed. Also, to find your variables, click through to your download page and look in the address field. You'll find your personal link information to put into the script.)—Ed.

Stop Cheating, Dave!

I'm finding all of Dave Taylor's shell scripts for cheating at *Scrabble*, *Words with Friends* and so on to be tiresome. Normally I'd be interested, but he's gone on about this ever since I renewed my subscription a few months ago!

I must admit, part of my blasé stems from the fact that I solved this problem more than five years ago for a different word game: *Jumble*, a syndicated newspaper word/riddle game I loved to play as a child. I was bored at my tech support job (with restricted Internet access), and someone gave me an entire illustrated book of *Jumble*. Rather than work the entire book manually, I wrote a series of Bourne shell scripts to solve it on a SCO UNIX system. All I had was SCO's man pages, aspell and a very limited vi. It took me a few weeks, but I was able to get it to the point where I could solve the entire book in about ten minutes.

So when reading Dave's column, I see that he's gone deeper than I did (granted, I had a much simpler, one-player game). In my case, I don't call it cheating. Scripting the problem means you never have to solve it again!

Will we see a different topic next month?

—Trey Blancher

Dave Taylor replies: ayup. I was ready to move on too and have done so (starting with my column in the July 2012 issue).

I will say this, however. If you have some great ideas about big, complex scripts I could work on, I'd be very interested in hearing about them. Sometimes I can rather get at a loss for a topic when my column's due, so if you've got ideas or problems you face, do share!

E-Reader Downloads

To Dave Heebner [see Dave's letter regarding downloading issues on his NOOK in the June 2012 issue]: after I download an issue of *LJ*, I open "My Stuff" in NOOK for PCs and use the "add new item" at the top of the page to copy the item from downloads.

—Ray Cleveland

Awesome! It's nice when things "just work" like they're supposed to.—Ed.

Esoteric...Hmmm...

Greg commented in the June 2012 issue that *Linux Journal* had become a bit esoteric with the subjects covered

recently. I'm a very recent two-year subscriber, and I must say that I echoed Greg's impression when I previewed a few past issues. I made a two-year subscription in spite of my initial impression because I was certain that it was a passing thing and that *LJ* eventually would cover more general and practical topics. A Bluetooth proximity device that locks and unlocks my computer when I leave and return is pretty incredibly cool, I admit, but not too practical. A wall-mounted, dual-monitor PC is also pretty cool. But practical? Don't get me wrong, there is nothing at all wrong with the esoteric, but let's not exclude the more common—something for the intermediate reader: an article on Udev and the many gotchas involved in manual configuration; Calibre, and what it can do; using notify-send to set up your own notifications; Flexget for automating torrent downloads. There is a trove of topics that would be beneficial to a very broad slice of your readership. I'd even be willing to write a few articles myself, if that would help. Love the magazine as a whole, just hoping for something a notch or two closer to home.

—Slurry

It's an admittedly strange game trying

to get just the right balance of articles to make everyone equally happy (or equally frustrated I suppose). We have a group of Linux Journal readers on our advisory board, and we run ideas past them every issue. We are always up for new writers, so if you'd like to send an article idea, we'd love to hear it. See <http://www.linuxjournal.com/author> for more information.

Substandard Working the Shell

Dave Taylor's Work the Shell column is far less useful than it could be. This is not *UNIX Journal*. This is *Linux Journal*. As such, the shell that people use is bash, not the Bourne shell. I just don't understand why every example is based on a 20-year-old shell that requires that every single thing can occur only by running external processes. Let's look at a few examples from the current issue:

```
if [ ! -z "$2" ] ...
```

vs.:

```
if [[ -n "$2" ]] ...
```

vs.:

```
[[ -z "$2" ]] || targetLength="$2"
```

[LETTERS]

For the record, I'd rather see people getting educated on what the difference is between use of single vs. double square brackets.

Here's another:

```
for word in $(cat $possibilities)
do
    length=$(echo $word | wc --c)
    length="${(( $length - 1 ))}" # Why the dbl quotes?
```

vs.:

```
for word in $(< $possibilities)
do
    length=$(( ${#word} - 1 ))
```

A few processes were saved. Who cares? The people who go into a job interview and demonstrate how little they know and how sloppy they are at not bothering to learn *properly* the language in which they are being paid to write. Besides that, there are lots of bad >10K LOC scripts out there that get buried under the sheer weight of the excess processes.

One last one:

```
uword=$(echo $word | tr '[:lower:]' '[:upper:]')
```

Sure, using a bash4 feature might be

pushing my luck because of all those people out there who are still(?) using an antique copy of bash. Maybe they can't use:

```
uword="\${word^^}"
```

But, at the very least, they could use:

```
uword=$(tr '[:lower:]' '[:upper:]' <<<$word)
```

I won't even go into why a while loop is being used that has a start, a termination value and an increment, when a for loop would be perfectly lovely.

I do hope that this problem can be rectified since I see no reason why it shouldn't.

—**Steven W. Orr**

Dave Taylor replies: Thanks for your interesting note, Steve. I have old dog/new tricks syndrome with the scripting I work off, so I have to say that I haven't seen the [[]] notation before. I'm curious. I tend to work off the reference materials I have on hand and the man pages: where are some of these newer scripting capabilities documented so I can catch up on things?

I'll definitely hit on these topics in my next column, due July 1, so stay tuned

for it, and thanks again for taking the time to write.

And, Steven replies to Dave: You have to read the bash man page about 200 times.

There used to be a program in /bin or /usr/bin called test. For a while, that program used to link to a file called [. After that, the test became built in, and you'd have to go out of your way to invoke the external test program. The [[]] construct is relatively new, around 20 years old or so. It started with ksh88, and bash has had it for a very long time.

The differences are quite important:

```
if [ -z "$foo" ] && [ $(xxx; echo $?) -eq 0 ] ||
➥[ $(yyy; echo $?) -eq 0 ]
```

will not do what you expect. It will test to see if \$foo is null, and if it is, it will then run the xxx test and not the yyy test. If the foo test is false, it will run the yyy test and not the xxx test.

The correct approach is:

```
if [[ -z "$foo" ]] && {
    xxx || yyy
}
```

Note that double square brackets are not a built-in; they are a keyword.

Another example that will demonstrate things:

```
if [[ t1 && t2 && t3 ]]
```

is not the same as:

```
if [ t1 -a t2 -a t3 ]
```

Multithreaded Simulation and OpenGL Visualization Demos

I am a long-term subscriber to *Linux Journal*, and I have recently posted the source code of two of my projects on GitHub. One is a lattice simulation and the other an N-body simulation:

<https://github.com/fhstoica/AbelianHiggs> and https://github.com/fhstoica/N_Body_Simulation.

The code gives a good real-world example of highly parallelizable computations making use of barrier synchronization and CPU affinity (Linux-specific) to improve performance. The visualization code implements the rendering of an isosurface using the Mesa and GLUT libraries.

I would have liked to see such examples when I was learning multithreaded and

[LETTERS]

OpenGL programming, so I am trying to make them available to anyone interested.

—Horace Stoica

Photo of the Month

Here's a picture of myself as Tux for Halloween.

—Joey Bartlett



Joey Bartlett, 13 Years Old

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

LINUX JOURNAL

At Your Service

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

SAVE THE DATE!



10th USENIX Symposium
on Operating Systems Design
and Implementation

October 8-10, 2012, Hollywood, CA

The 10th OSDI seeks to present innovative, exciting research in computer systems. OSDI brings together professionals from academic and industrial backgrounds in what has become a premier forum for discussing the design, implementation, and implications of systems software.

The full program and registration info will be available in August 2012.

www.usenix.org/osdi12

USENIX

STAY CONNECTED ON



www.facebook.org/usenixassociation
www.twitter.com/usenix #osdi12

Don't Miss the Co-Located Workshops

- HotDep '12: Eighth Workshop on Hot Topics in System Dependability, October 7
- HotPower '12: 2012 Workshop on Power Aware Computing and Systems, October 7
- MAD '12: 2012 Workshop on Managing Systems Automatically and Dynamically, October 7



diff -u WHAT'S NEW IN KERNEL DEVELOPMENT

Recently, some folks identified a bad patch that had gotten into the 3.3.1 kernel in the stable tree. It caused a certain set of hardware to fail to wake up again, once it'd been suspended. The fix was simply to revert the bad patch, after which the affected systems would work fine again. But, **Greg Kroah-Hartman** refused to make the change in his upcoming 3.3.2 release until **Linus Torvalds** made the change in the upstream git tree.

The result was an interesting debate between **Filipe Contreras** and a bunch of other developers over whether that was a good idea. Filipe's argument was that the 3.3.x kernels represented the "stable series", and had the goal of eliminating bugs and giving the smoothest ride possible. Therefore, he argued, it made no sense to wait around for bug fixes to be incorporated into the upstream git tree first. The upstream git tree was for regular development, not stabilization, although the bug fix in question would result in real systems being able to do real work, instead of sitting there in a hung state.

There were a number of responses to that argument from various developers,

including Linus. One argument made by **Willy Tarreau** was that if bug fixes didn't get into the upstream tree first, the bugs potentially would reappear in the very next development release. And, because each stable series was derived from a development release, it was only by incorporating stable series fixes upstream that the developers could ensure that fixes to one stable series would make it into the next stable series.

Linus also pointed out that there had been a time when this particular rule didn't exist, and the problem of fixes going into the stable series and then being left out of the development tree happened "all the time".

Adrian Chadd offered up a bit of history from the **Squid Project**. Apparently, it had a stable and a development branch that it allowed to diverge, with the result that users wanted features and bug fixes for the older tree that were incompatible with the newer one, and mayhem ensued. So, Adrian argued, having a rule requiring changes to the stable tree to be incorporated into the development tree was a clean way to avoid that.

The really interesting thing about the

whole discussion is that it illustrates the way kernel development itself develops. Traditionally, there's always been some aspect of the process that would bog down or become in some way unworkable—whether it was the struggle to find a good version control system, or identifying which people were the right ones to send patches directly to Linus, or figuring out how to do development while also producing

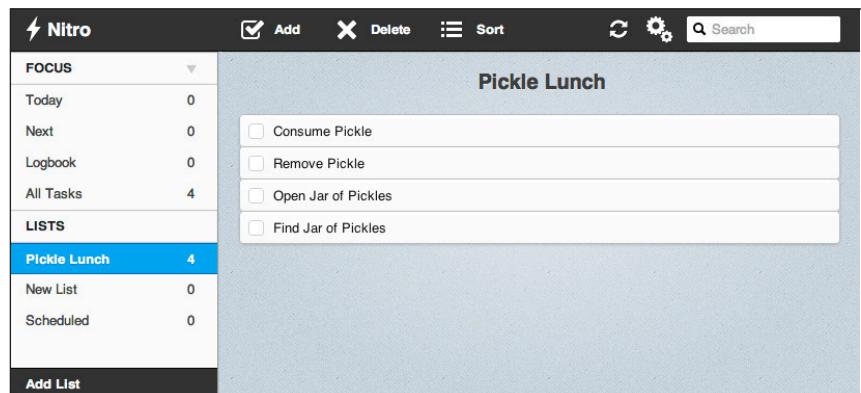
a stable kernel for distributions. Eventually something breaks down and reveals some set of nuances that no one had noticed before. At that point, Linus and various other folks typically take an algorithmic approach to smoothing things out again. Sometimes the solutions break down later in new ways, and the succession of adjustments is fascinating to watch over time.—**ZACK BROWN**

Linux-Native Task Management? Check.

Task management programs are commonplace in our busy lives, but it seems that every system lacks something. Google Tasks is nice, but it lacks the features of a more robust task management system.

Remember the Milk is nice, but it charges for some of its features. Many standalone task management programs are great, but they don't sync between devices.

Thankfully, Nitrotasks is a Linux-based task management program that not only syncs between computers, but also has some organization features reminiscent of the "Get



Things Done" method. When you add the Web-based version and syncing compatibility with Dropbox or Ubuntu One, Nitrotasks is worth checking out. It's simple, but its Linux-native roots and Web-based access make it compelling. Be sure to put "check out <http://www.nitrotasks.com>" on your to-do list!—**SHAWN POWERS**

Visit LinuxJournal.com and Help Take Over the World

Linux really means business. It's not just an issue focus, but also a reality. *Linux Journal* has been around since before Linux was cool. We've seen it grow from a system praised by hobbyists and hardcore geeks, through its not-so-awkward adolescence during the tech boom of the late 1990s, and blossom into an operating system so elegant that it has penetrated the mainstream and directly touches so many people in such a way that few even are aware of its presence. As we all know, few people who participate in the digital world in which we currently live are not touched by Linux in some way. That said, if you are reading this issue, you more than likely are one of the informed few who helps power the open-source ecosystem that in turn powers the lives of so many. To that end, we offer you LinuxJournal.com as a resource to aid you in your pursuit of technical greatness. With articles focused on programming, security, desktop, system administration and more, we think you'll find it a handy reference for staying up to date and discovering new skills that will help you make the world a better place.

—KATHERINE DRUCKMAN

They Said It

**Most certainly,
some planets are
not inhabited, but
others are, and
among these there
must exist life
under all conditions
and phases of
development.**

—Nikola Tesla

**Life is and will
ever remain an
equation incapable
of solution, but it
contains certain
known factors.**

—Nikola Tesla

**The opinion of the
world does not
affect me. I have
placed as the real
values in my life
what follows when I
am dead.**

—Nikola Tesla

**I predict that
very shortly the
old-fashioned
incandescent lamp,
having a filament
heated to brightness
by the passage of
electric current
through it, will
entirely disappear.**

—Nikola Tesla (in April, 1930!)

**The desire that
guides me in all I
do is the desire to
harness the forces of
nature to the service
of mankind.**

—Nikola Tesla

Ubuntu's New DNS: Unknown Host

If you're the type of person who installs Ubuntu's server edition, you're also likely the sort of person who knows how to configure network settings. For most distributions, especially those based on Debian, the process is a bit strange, but familiar.

To configure the various interfaces, you edit /etc/network/interfaces and add the appropriate IP information, along with the gateway address. That doesn't complete the process, however, because if you manually configure the network interfaces, you need to add the DNS servers manually to the /etc/resolv.conf file. That's the way it's always been, and I never put much thought into it—until Canonical changed the way the resolv.conf file works.

I'll admit, my initial reaction was one of frustration, but once I got over myself, I have to say it makes much more sense to add the DNS configuration right into the /etc/network/interfaces file as well. My only complaint is that there aren't comments in either the /etc/resolv.conf file or the /etc/network/interfaces file on actually how to do it! (Thankfully, there is a note in /etc/

resolv.conf warning that any changes will be overwritten, but no hints on how to make changes properly.)

The process, as it turns out, is rather simple. In /etc/network/interfaces, simply add a couple lines to the end of the stanza for a particular interface—for example:

```
# The primary network interface
auto eth0
iface eth0 inet static
    address 192.168.1.200
    netmask 255.255.255.0
    gateway 192.168.1.1
    dns-nameservers 192.168.1.1
    dns-domain example.com
    dns-search example.com
```

You'll notice the last three lines contain a few unfamiliar directives. They are fairly self-explanatory, but important to know. Once added to the /etc/network/interfaces file like above, the /etc/resolv.conf file is populated with the proper information when the network is activated. It's fairly simple, and it makes sense to have all the settings in one spot—but frustrating if you don't know about the changes!

—SHAWN POWERS

Lightspeed on Your Desktop

One area of physics that is hard to wrap your head around is relativity. Basically, relativity breaks down into general and special relativity. General relativity deals with large masses and high energies, and it describes how space-time is warped by these. Special relativity deals with what happens during high velocities. Many odd and counter-intuitive effects happen when speeds get close to the speed of light, or c . The problem is that these types of conditions are quite far outside normal experience, so people don't have any frame of reference as to what these effects would look like—enter Lightspeed.

Lightspeed is an OpenGL program that shows what an object would look like as it travels closer and closer to the speed of light. Lightspeed actually models four different effects that occur when you get close to the speed of light. The first effect is called Lorentz contraction. This effect causes objects to appear to shrink in the direction of travel. This is scaled by a factor called gamma. Gamma is calculated by:

$$\gamma = \frac{1}{\sqrt{1 - v^2/c^2}}.$$

So, as you can see, as your velocity (v) gets closer to the speed of light (c), the value gamma increases toward infinity. The length contraction is calculated by

$$l' = l / \gamma.$$

This means the length of an object in the direction of travel (l) decreases toward zero as the velocity increases toward the speed of light.

The second effect that Lightspeed models is Doppler shifting. You probably have noticed the sound version of Doppler shift when a fire truck drives by with its siren on. You can hear the shift in sound frequency from high to low as it goes by. The same thing happens with light too. As a light source comes toward you, it shifts in color toward blue. As it goes away, the color shifts toward red.

The third effect is something called the Headlight effect. When you have a light source that is moving, the amount of light being emitted isn't the same in all directions if that light source is moving. If the light is coming toward you, it will appear brighter. If it is moving away from you, it will appear darker.

The last effect that Lightspeed models is something called optical aberration. Because light travels at a finite speed, the light from different parts of the object come to you at different times. The end effect is that as an object comes toward you, it looks stretched out. And, when it travels away from you, it looks squashed. This is actually different from Lorentz contraction.

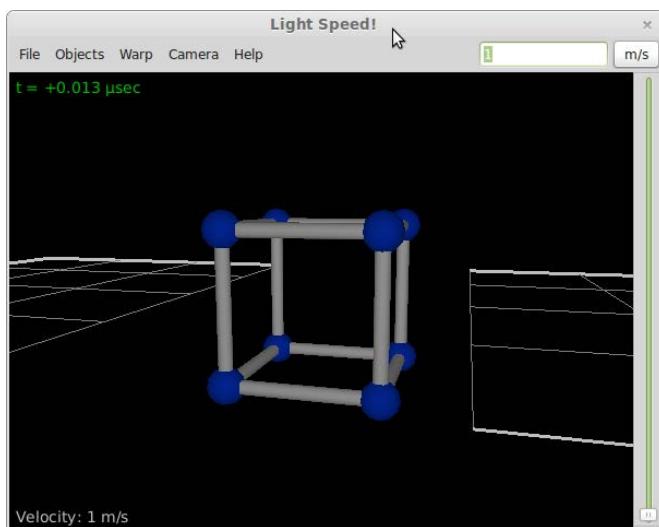


Figure 1. Initial Scene on Starting Lightspeed

So, how can you see what an object would look like, taking all of these effects into account? This is where Lightspeed comes in. Most distributions should have a package available that you can install using their package management system. If you are interested in building from source, it is hosted at SourceForge. When you first start it up, it opens with a cube set in the middle of a field made up of rods for the edges and balls for the vertices (Figure 1). The beginning velocity is set as 1m/s. Since this is an OpenGL program, you simply can grab the cube with your mouse and spin it around, or up and down, in order to change the view of the object. At the far right, you can set the velocity that the object has relative to you, going from 1m/s all the way up to 299,792,457m/s (the speed of light is 299,792,458m/s).

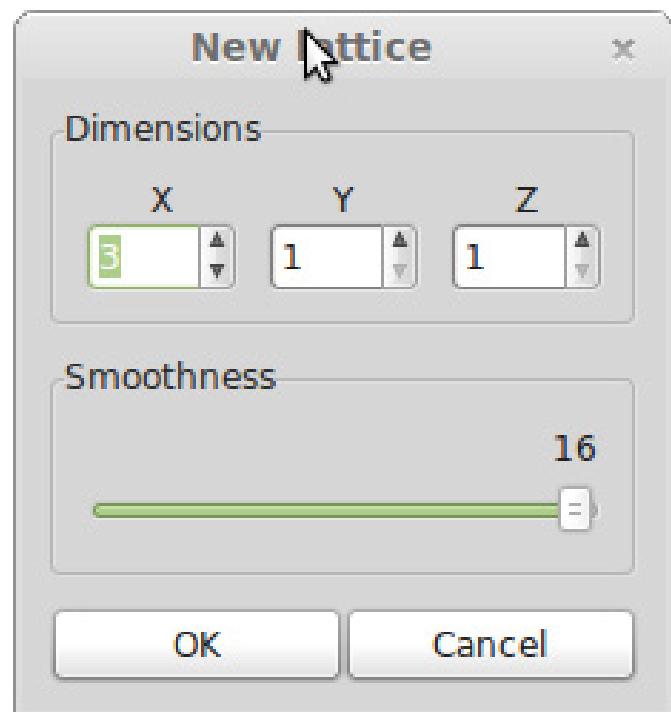


Figure 2. Setting the Number of Vertices in Your Object

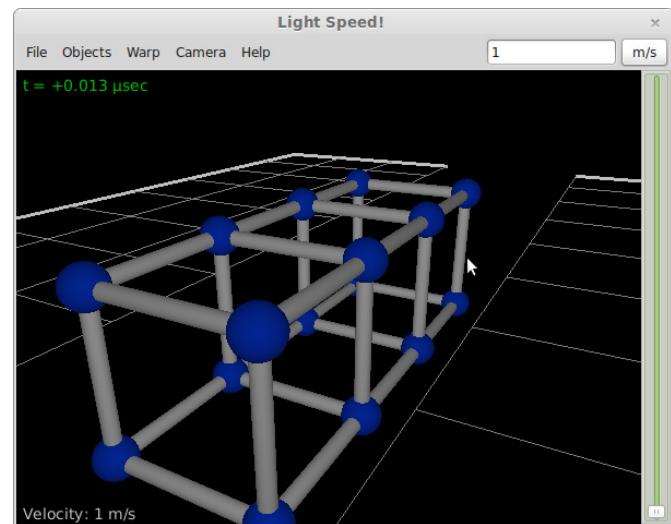


Figure 3. Altering the Smoothness of the Rendering

You can change the dimensions of the default cube object by selecting the menu option File→New lattice (Figure 2). You

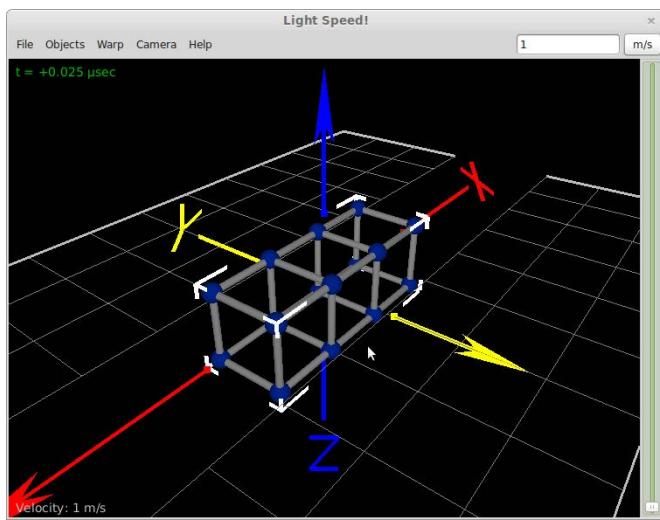


Figure 4. Displaying Axes and a Bounding Box around Your Object

can set how many balls will be drawn in each of the three dimensions. You also can change the smoothness factor when the object is rendered on the display. In this case, it will look like what is shown in Figure 3.

You also have the option of loading your own 3-D object, either in 3D Studio format (*.3DS or *.PRJ) or in LightWave format (*.LWO). This new object is what will be rendered, and the optical effects will be applied to this. You can save a snapshot of a particular object at a particular speed in either PNG or TIFF format. You also can export to an SRS file format (Special Relativity Scene). This is a format used by the program BACKLIGHT, which is a specialized raytracer used to illustrate relativistic effects. This lets you generate much higher resolution images of the

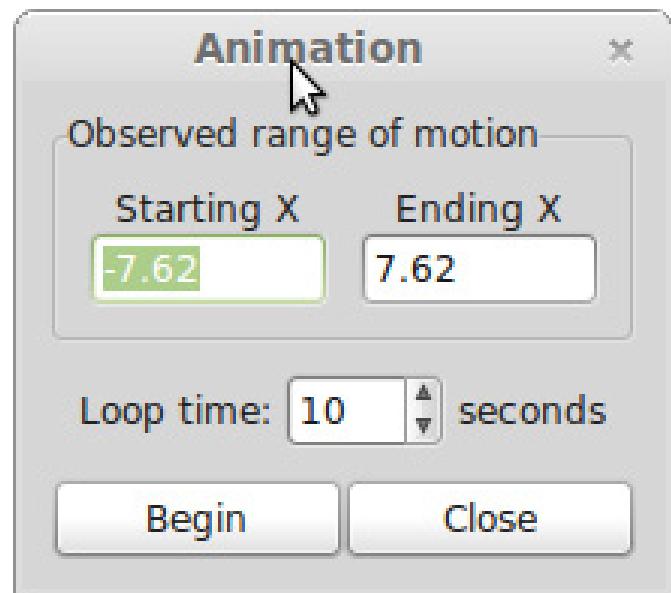


Figure 5. Setting Options for an Animation of Your Object

affected object.

In the Objects menu item, there are options for drawing supplementary objects. By default, the floating grid is selected. You also can select coordinate axes, identifying the x, y and z axes. You can select the display of a bounding box for your object as well (Figure 4).

Additionally, you can animate the scene by selecting Objects→Animation in the menu. This pops up a dialog box where you can select the starting and ending points on the x axis and the loop time in seconds (Figure 5). The scene then will be animated until you click stop.

Above, I looked at the four effects Lightspeed can model and apply to your object. By default, all of the

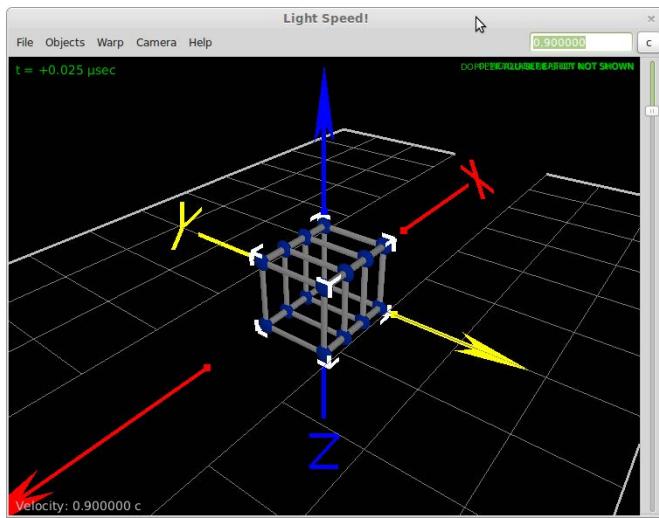


Figure 6. Looking at Just Lorentz Contraction at .9c

effects are selected when you start up. If you want to change which effects are being modeled, you can go to the Warp menu item and select any combination of Lorentz contraction, Doppler red/blue shift, Headlight effect or Optical aberration. So, you can select only the Lorentz contraction and see what it looks like at 90% of the speed of light (Figure 6).

You have quite a bit of control over the camera as well in Lightspeed. You can select the focal length of the camera lens, going from 28mm to 200mm. You even can set a custom lens focal length by selecting Camera→Lens→Custom. You can set the position of the camera precisely by clicking Camera→Position. This displays a pop-up dialog, where you can set the exact x, y and z values for its location (Figure 7).

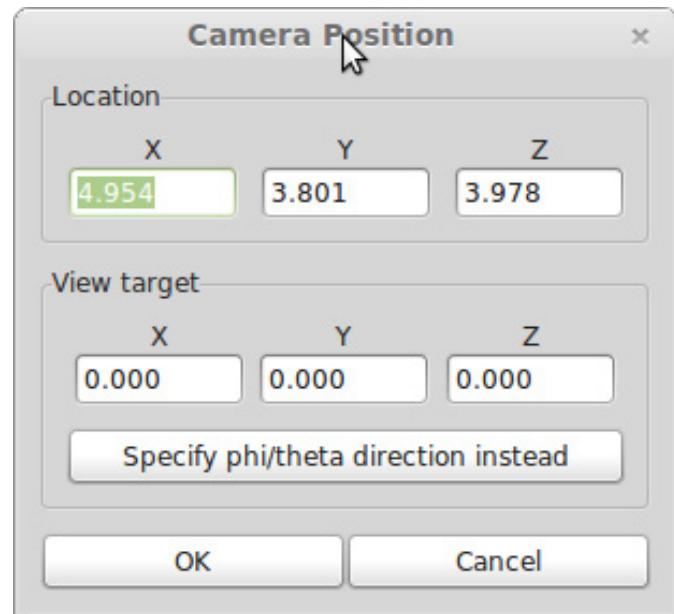


Figure 7. Changing Camera Position for a Better View

You can select what information is displayed on the screen by selecting the menu item Camera→Info display. You can have the velocity, the time, the gamma factor and/or the frame rate displayed on the screen.

The default background color is black, but you can change it to gray, white or very white. This display is an OpenGL display, so you can select the rendering mode. The default is shaded, but you can change it to wireframe rendering.

One of the really cool options is that you can spawn off other cameras by selecting Camera→Spawn camera. This lets you see your object from several different angles at the same time. So, now you can see what it looks like coming and going at the same time (Figure 8).

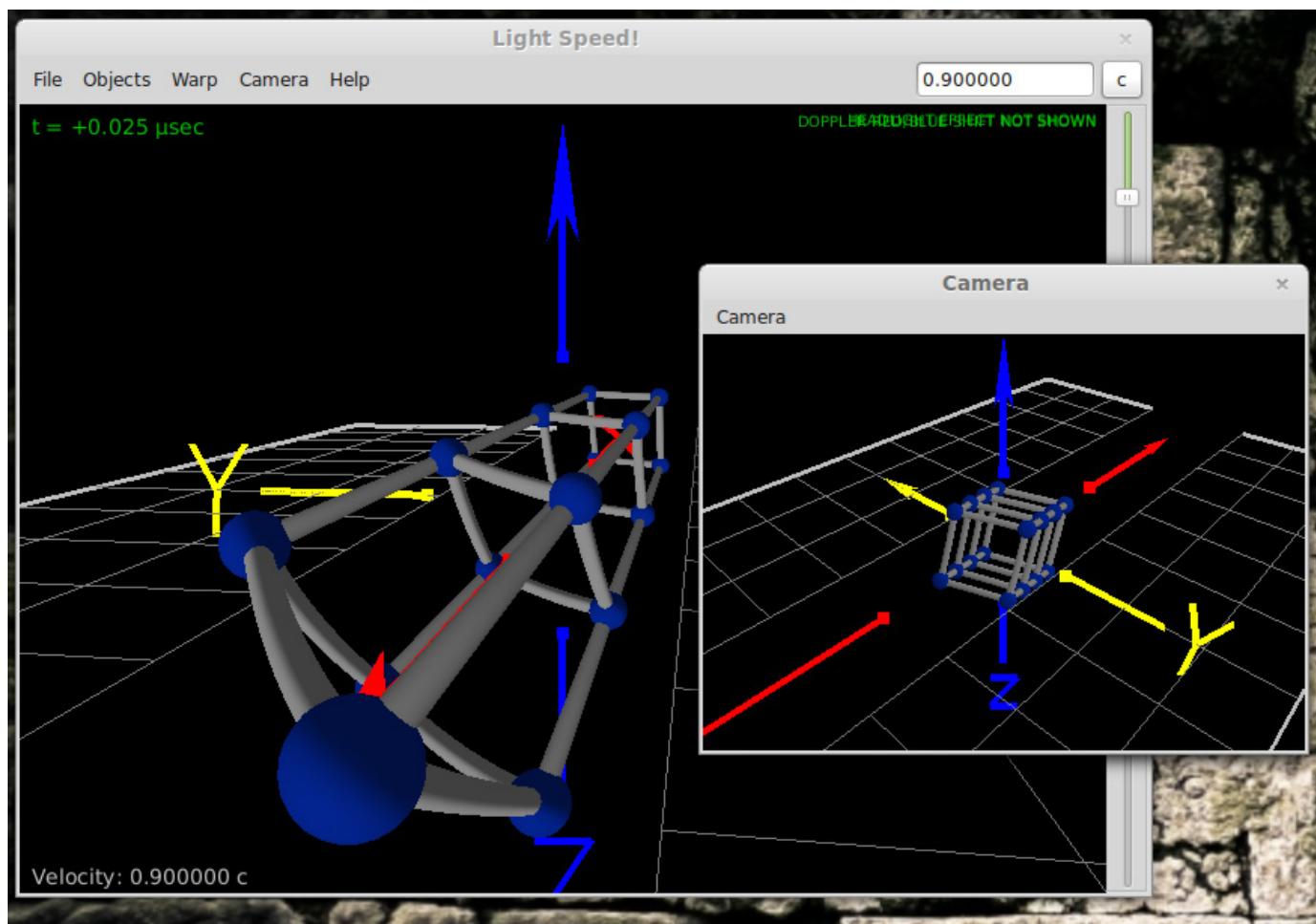


Figure 8. Setting Up Multiple Cameras for Different Perspectives

Once you have the speed and the relativistic effects set, you probably want to interact with the object a bit to see how it looks from different directions and so on. If you click the left mouse button and drag around, your object will be rotated around. If you click the left mouse button and press Shift at the same time, the camera view will be moved around, pointing in different directions. You can move the camera itself by clicking the middle mouse button and moving it up and down or left and right. You can move the camera

in and out from the object by clicking the right mouse button and sliding up and down. If you completely mess up your view, you always can go back to the defaults by clicking the menu item Camera→Reset View.

So, remember this program when you are studying special relativity. Now you finally can see what it would really look like if you were traveling down the tunnel with the protons in the LHC or what the *Starship Enterprise* would look like as it flew by at sublight speed.

—JOEY BERNARD

Linux in the Workplace Poll

We at *Linux Journal* HQ have a lot of Linux in our office. It's on desktops, laptops, servers, phones and tablets. We love Linux so much, we even have a Tux Droid (<http://lnxjr.nl/tux-droid>). We also use a wide variety of non-Linux open-source software for various purposes, and as FOSS has become more and more ubiquitous, we bet you do too. So, we wondered how you, our readers, use Linux and open-source software in your workplaces. Thanks to all who answered our short poll on LinuxJournal.com; we now have a better idea of how much love our favorite operating system gets in your offices.

When asked how much Linux you use in the workplace, we're thrilled to learn that the most-popular response (29%) was "We're allowed to choose the operating system we use, and the cool kids use Linux." Freedom to choose your operating system is something we love to see. And, although we are also excited that a solid 14% of you indicated that everything you touch in your office uses Linux, we appreciate that this is not always possible in every work environment, so we're happy that most of you are in an environment where your preferences are respected, whatever they may be.

We are also glad to report that 66% of you use Linux on your work computers.

The most popular flavor was Ubuntu, with Fedora a pretty distant second. And to the 34% using Windows, we hope it isn't giving you too many headaches!

86% of you use other, non-Linux open-source software, and a full 25% of you use mostly open-source software on Linux. And to the 14% who believe you are able to use open-source software just because no one knows Firefox is open source, let's hope the powers that be remain blissfully ignorant.

The most-popular reasons for using Linux at your companies are that Linux saves money and is more secure, and we couldn't agree more. You most commonly use Linux as a Web server, database server, desktop system, file server, for software development and as a mail server. Not too many of you are using Linux for movies and graphics, but we'd love to hear more from the 8% who are. Please drop us a line if you'd like to show off your work.

We are pleased that 32% of you are in your current position because of your Linux expertise. We also understand that our readers are people of many talents, so 44% of you hold your current position regardless of your Linux knowledge.

Finally, we're relieved that 73% of your bosses do know what Linux is, and to the rest of you, please let me know if you'd like my help in staging an intervention.

[UPFRONT]

Your boss may thank you in the long run, even if he or she thinks it's an air-conditioning company at the moment. Rest assured that I feel your pain. When I first worked for *Linux Journal*, many of my friends were pretty sure I was writing about china and crystal.

Thank you to all who participated in our poll, and as always, please check out LinuxJournal.com for more! Here's the poll and results:

1. HOW MUCH LINUX DO YOU USE IN THE WORKPLACE?

- 14% — Everything I touch uses Linux, including the office coffee pot, the escalator and the florescent lights illuminating the penguin tank.
- 17% — We use mostly Linux at work, but that weird guy in accounting has an iPhone.
- 29% — We're allowed to choose the operating system we use, and the cool kid(s) use Linux.
- 16% — Windows is on all our computers, but I have an Android phone in my pocket.
- 7% — Linux is forbidden in the workplace, but the IT guy secretly uses it in the server room.
- 5% — Penguins are shot on sight, and any mention of Linus Torvalds results in immediate termination.

2. WHAT OPERATING SYSTEM IS ON YOUR WORK COMPUTER?

- 13% — Windows XP
- 21% — Windows 7
- 21% — Ubuntu
- 3% — Kubuntu
- 2% — Xubuntu
- 1% — Other 'buntu
- 6% — Debian
- 6% — Fedora
- 3% — SUSE
- 4% — Red Hat
- 5% — Linux Mint
- 15% — Other distro

3. DOES YOUR COMPANY USE OPEN-SOURCE SOFTWARE (OSS) OTHER THAN LINUX?

- 14% — No.
- 14% — Yes, but only because no one knows Firefox is open source.
- 11% — Yes, but only in the server room.
- 3% — Yes, but only for Web development.

- 14% — Yes, we use some desktop software like LibreOffice.
- 6% — Yes, we use mostly OSS on Windows.
- 1% — Yes, we use mostly OSS on OS X.
- 25% — Yes, we use mostly OSS on Linux.
- 7% — Everything in our office is open source, even the soda we drink.

4. IF YOUR COMPANY USES LINUX, IT IS BECAUSE:

- 56% — Linux saves money.
- 53% — Linux is more secure.
- 28% — Linux runs on older hardware.
- 23% — Linux isn't a Microsoft product.
- 24% — Because I put it there and no one knows.

5. DO YOU HAVE YOUR CURRENT POSITION:

- 32% — Because of Linux expertise.
- 9% — In spite of Linux expertise.
- 44% — It has nothing to do with Linux.
- 6% — I'm not currently employed; thanks for bringing it up.

6. DOES YOUR BOSS KNOW WHAT LINUX IS?

- 73% — Yes.
- 13% — No.
- 7% — Well, my boss thinks Linux is an air-conditioning company.

7. HOW IS LINUX USED AT YOUR COMPANY?

- 60% — Web server
- 55% — Database server
- 53% — Desktop system
- 51% — File server
- 48% — Software development
- 35% — Mail server
- 34% — DNS server
- 34% — Virtual server
- 23% — VPN
- 22% — Spam or virus filtering
- 16% — Directory server
- 13% — Telephony or PBX
- 9% — HPC
- 9% — Movie/graphics development

Note: not all respondents answered each question, and a few questions allowed for multiple answers, so please

don't be alarmed when the answers don't add up to 100%.

—KATHERINE DRUCKMAN

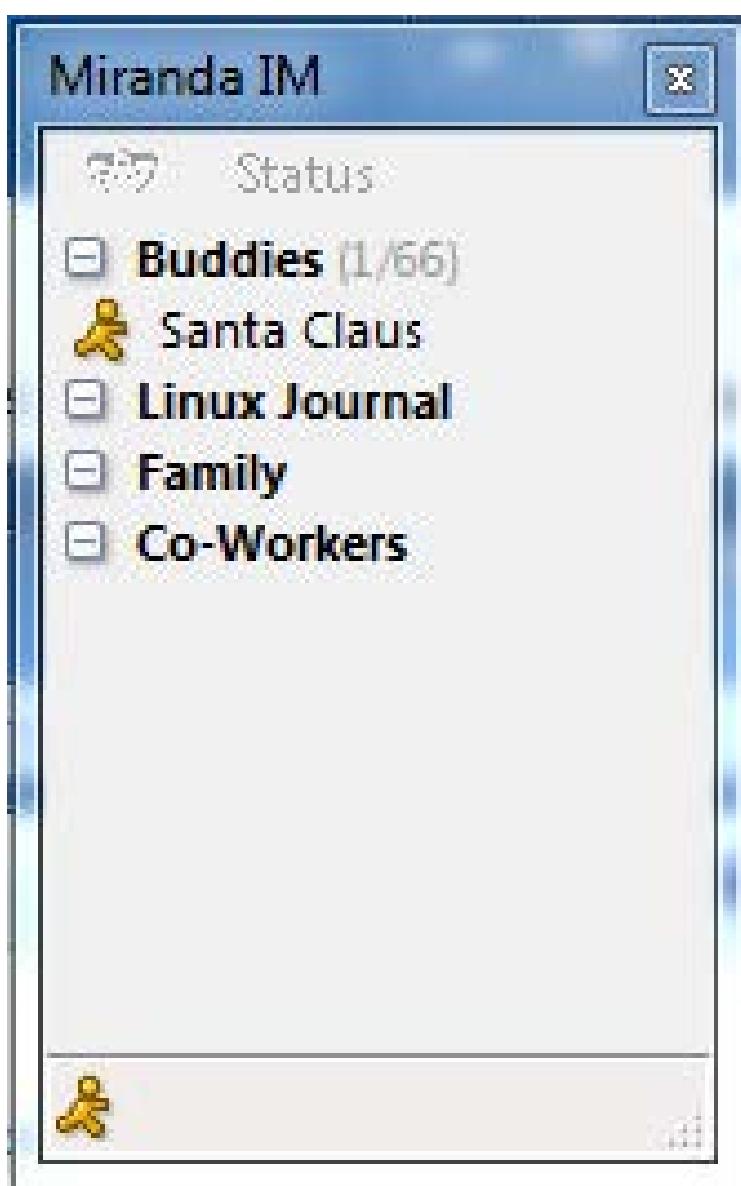
Non-Linux FOSS

If you're a Windows user, choosing an instant-messaging client actually can be a little frustrating. Some of the more-popular clients are loaded with advertising or cost a lot to get full functionality. Normally, I recommend Windows users install the Windows version of Pidgin. Recently, however, I stumbled across a nifty Windows-only IM client named Miranda.

Miranda is open source, ad-free and integrates perfectly into Windows. Granted, Pidgin works brilliantly under Windows, but it always feels like what it is—a port of a program designed for something else. Miranda has only a Windows version, and the one thing it does, it does quite well.

If the standard install of Miranda doesn't quite cover your needs, be sure to check out the add-ons designed for it. Whether you want to add Facebook chatting, integrate with Skype or check the weather, there are tons of add-ons from which to choose. Check it out today at <http://www.miranda-im.org>.

—SHAWN POWERS



ONLY 1&1 OFFERS YOU THE RELIABILITY OF DUAL HOSTING



What is Dual Hosting?

Your website hosted across multiple servers in 2 different data centers, in 2 different geographic locations.

Dual Hosting,
maximum reliability.

1&1 – get more for your website!

- ✓ **More Possibilities:**
65 Click & Build applications.
- ✓ **More Included:**
Free domain*, free e-mail accounts, unlimited traffic, **NEW: Adobe® Dreamweaver® CS5.5** and much more.
- ✓ **More Privacy:**
Free private domain registration.
- ✓ **More Reliability:**
Maximum reliability through hosting simultaneously across two separate data centers.



**SALE ENDS
AUGUST 31, 2012**

ALL 1&1 HOSTING PACKAGES

\$3.99 per month
SAVE UP TO 60%!*

DOMAIN OFFERS: .COM/.ORG JUST \$ 3.99 (first year)*



www.1and1.com



* Offers valid for a limited time only. 12-month minimum contract term and 3-month pre-paid billing cycle apply for web hosting offer. Standard prices apply after the first year for domain and hosting offers. Free domain with Unlimited and Business hosting packages. Visit www.1and1.com for billing information and 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. © 2012 1&1 Internet. All rights reserved.

Resara Me This



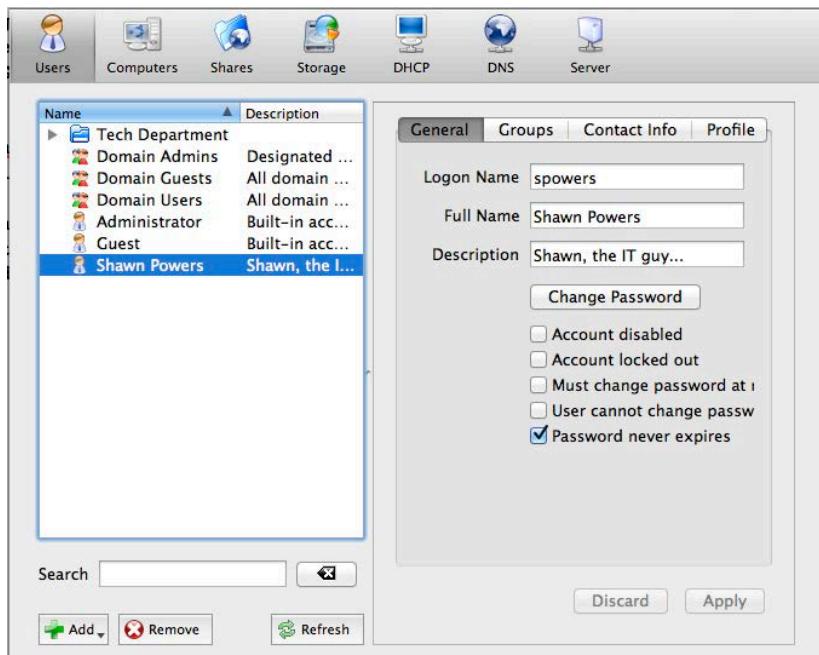
More than any other program, Samba allows Linux desktops to exist in the world of Windows. In fact, Samba historically has allowed Linux to live secretly in the server room as well. It's possible to emulate a Primary Domain Controller from your Linux server, and Windows machines can't tell the difference. The problem is that Microsoft no longer uses PDCs and has turned to Active Directory.

Thankfully, Samba version 4 has just entered beta, and it supports Active Directory emulation! Although it's still in early beta stage, that hasn't stopped the folks at Resara from packaging up the software into a turnkey Active Directory replacement. Offering a fully functional community version along with its only slightly more-functional commercial version, Resara gives you a surprising feature set.

The server itself is built on Ubuntu, and its cross-platform (Linux, Windows, OS X) GUI administration tool runs pretty much anywhere. It offers simple administration of users, groups, computers, DNS and DHCP. If you need further customization of the

Active Directory itself, Resara is compatible with Microsoft's own Group Policy Editor. That, unfortunately, runs only on Windows.

You'll be hearing more about Resara from me in future issues, but for folks looking to implement an Active Directory system without the desire to pay for Microsoft licensing, check out Resara now! The community version is at <http://www.resara.org>, and the commercial version is at <http://www.resara.com>. I'm using the commercial version, both because I want to support the project and because I'm sure I'll take advantage of the technical support!—**SHAWN POWERS**



The administration GUI is simple and effective.







REUVEN M.
LERNER

Database Integrity and Web Applications

Want to improve the integrity of your data? Place constraints in the database, as well as in your application.

NoSQL, the catchall phrase for non-relational databases, is all the rage among Web developers. However, it's somewhat unfair and unhelpful to use the term NoSQL to describe them, given the variety of technologies involved. Even so, there are some fundamental differences between traditional relational databases and their NoSQL counterparts. For one, as the name implies, NoSQL databases don't use the standard SQL query language, and use either their own SQL-like language (for example, MongoDB) or an object-oriented API. Another difference is the lack of two-dimensional tables; whereas SQL databases operate solely with such tables, NoSQL databases eschew them in favor of name-value pairs or hash-like objects. And finally, NoSQL databases typically lack the features that led to the development of relational databases,

namely transactions and data integrity.

There's no doubt that the flexibility NoSQL databases offer is attractive on a number of levels. Just as I enjoy working with dynamic languages in which I don't have to declare my variables (or their types) before I use them, it's nice to be able to store objects in my database without having to define the object structure in advance. If I want to add a new field to my Person object, I just do so, and the database magically catches up.

At the same time, there are many cases when I want the database to be tough with me and enforce the integrity of my data. That is, I want to be sure that even if I have made a mistake in my application, or if a user enters a value that shouldn't be allowed, the database won't allow that bad data to be stored. And yes, I believe that it's good to have

I often have to do what I refer to as “database surgery” on applications that are running for my clients, and it’s always reassuring to know I cannot make a change manually that would corrupt the data.

such checks at the database level, rather than just at the application level—not only because it provides an additional guard against corrupt data, but also because the database often is accessed directly, outside the application itself. I often have to do what I refer to as “database surgery” on applications that are running for my clients, and it’s always reassuring to know I cannot make a change manually that would corrupt the data.

Readers of this column know I’m a fan of both Ruby on Rails and of PostgreSQL, and I often use them together on projects. However, because Rails originally was developed under MySQL, which lacks many of the data-integrity aspects of PostgreSQL, the standard Active Record package fails to include such items as foreign keys in its base implementation. This means that although Rails will support PostgreSQL out of the box, it doesn’t provide support for foreign keys, let alone data-integrity checks.

So, in this article, I look at some Ruby gems that make it possible to include foreign-key support in your Active Record models. I also describe ways you can

take advantage of PostgreSQL’s CHECK mechanism to ensure that your data is as safe as possible.

Where Do You Check Your Data?

Before I go into the actual code and implementation, I admit there are several philosophies of where and how to check your data. As I mentioned previously, I prefer to have the data checked at both the database and application levels, although sometimes I’ve been lazy and used the default in Rails that does so only at the application level.

Checking your data only at the application level is attractive in many ways. It makes the implementation easier, and it often will work just fine. It also, in the case of Rails, allows you to work in a single language (that is, Ruby) and put the integrity checks in your model files, where you’re going to be using them.

But as I already said, if I have my checks only in the application, it’s quite possible that when I try to access the database from outside my application, I’ll end up messing things up, if only accidentally. Now, I’ll admit this is the way many Web applications

It would be a bigger mistake to try to do things the other way around—that is, to have the checks only in the database, rather than at the application level.

are designed, and it's not a fatal flaw. But it is something that I'll try to avoid in this article.

It would be a bigger mistake to try to do things the other way around—that is, to have the checks only in the database, rather than at the application level. Imagine, for example, if you were to have a "NOT NULL" constraint in the database, such that a particular column could not contain a NULL value. If you didn't protect against such things at the application layer, you might end up trying to put that data into your database. And although the database itself would not be corrupted, the application would generate an error, confusing and frustrating users.

So, upsetting and annoying as it might be, the best way to do things probably is to duplicate some work—adding the constraints on the database and then having them on the application as well. You could argue that you should have the constraints in place in a third location—in the HTML form the user often will use to submit information to the server. Fortunately, there is at least one technique for handling this sort of thing automatically. The client_side_validations gem for Rails copies the validations

as best as possible, putting them into JavaScript within the user's views.

Foreign Keys

With this background, let's now assume that I want to implement a simple appointment calendar. In order to implement my calendar, I need to keep track of people (each of whom has a first name, a last name, an e-mail address and a telephone number) and appointments (which will indicate the date/time of the appointment, the person with whom I'm meeting and a note about the meeting itself).

At the database level, the tables will be fairly straightforward. This is how I would create the tables if I were doing so manually:

```
CREATE TABLE People (
    id SERIAL,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    email TEXT NOT NULL,
    phone TEXT NOT NULL,
    PRIMARY KEY(id)
);
```

```
CREATE TABLE Appointments (
    id SERIAL,
```

```

person_id INTEGER REFERENCES People NOT NULL,
notes TEXT NOT NULL,
PRIMARY KEY(id)
);

```

The important thing here, for the purposes of this discussion, is the **REFERENCES People** clause in the definition of the Appointments table. With the REFERENCES clause in place, I can do the following:

```

INSERT INTO People (first_name, last_name, email, phone)
VALUES ('Reuven', 'Lerner', 'reuven@lerner.co.il',
➥ '847-230-9795');

INSERT INTO Appointments (person_id, notes) VALUES (1, 'Meet with
➥ myself');

```

Now, here's what happens if I try to delete myself from the People table:

```

atf=# DELETE FROM People WHERE id = 1;
ERROR: update or delete on table "people" violates foreign key
constraint "appointments_person_id_fkey" on table "appointments"
DETAIL: Key (id)=(1) is still referenced from table "appointments".

```

Because Appointments.person_id is a foreign key to People.id, I cannot remove a row from People if Appointments refers to it. This sort of relational integrity is a great thing for my application. No matter how you slice it, it means I cannot remove people from the system if they are scheduled for an appointment.

Rails Integration

If these tables were part of a Web

application that I was developing in Ruby on Rails, I would use database migrations to create them. The migrations would look something like this (if combined into a single file):

```

class CreatePeopleAndAppointments < ActiveRecord::Migration
  def change
    create_table :people do |t|
      t.string :first_name, :null => false
      t.string :last_name, :null => false
      t.string :email, :null => false
      t.string :phone, :null => false
      t.timestamps
    end

    create_table :appointments do |t|
      t.integer :person_id, :null => false
      t.text :notes, :null => false
      t.timestamps
    end
  end
end

```

Now, this will give me the table definitions, but it won't give me the foreign keys, meaning that I could corrupt the database by deleting a single "People" record. In order to do that, at least in the default Rails configuration, I need to use an "execute" command in the migration to send explicit SQL to the database.

Fortunately, there is an easier way.

The Foreigner gem, written by Matthew Higgins, which works with MySQL as well as with PostgreSQL, adds syntax to let you create and remove foreign keys within your migrations.

The Foreigner gem, written by Matthew Higgins, which works with MySQL as well as with PostgreSQL, adds syntax to let you create and remove foreign keys within your migrations. For example, with Foreigner active—putting it in the Gemfile and then running `bundle install`—I can add a new migration that does the following:

```
class AddForeignKey < ActiveRecord::Migration
  def up
    add_foreign_key :appointments, :people
  end

  def down
    remove_foreign_key :appointments, :people
  end
end
```

Sure enough, after running this migration, I have the foreign key that I was hoping for. Again, it's still important for me to have this check in my Rails model; otherwise, I easily could get myself into a situation that the database forbids but the model allows and, thus, generate a bad error for my users.

Note: if you add the foreign key after you already have populated the

database with some data, you might run into trouble. That's because PostgreSQL won't let you add a foreign key if one or more rows fail to abide by the declared constraint. The NOT NULL constraint additionally ensures that you point to *some* person in the system. In other words, every appointment has to be with someone, and it has to be with someone in the People table.

Now, let's say you already have been working on a project, but you have neglected to define foreign keys. You could go through each table, figure out which is pointing to which, and then handle it accordingly, adding the sorts of migrations shown above. But the Immigrant gem (also written and released by Matthew Higgins) looks at Rails models and adds foreign keys for every `has_one` and `has_many` column that points to a `belongs_to` column in another model.

Additional Checks

Foreigner is a great step forward, helping ensure the integrity of your data. But there is another issue, more subtle than that of foreign keys, to which I'm still susceptible. While I have ensured that the

person_id will not be NULL and will point to a record in the People table, I haven't ensured that the People table will contain valid and reasonable values. That is, although the first_name, last_name, e-mail and phone columns will not contain NULLs, they might contain empty strings or values (for example, e-mail addresses) that are invalid.

At the database level, I could handle this sort of problem with a CHECK clause. Such a clause ensures that illegal data—for whatever "illegal" values I want to define—cannot be placed in the database. This could be anything from a certain minimum or maximum length of text string, to a pattern in the text, to minimum or maximum numbers. For example, I often like to indicate that the database may not contain a price lower than 0, and that e-mail addresses need to match a very basic regular expression. (Note that matching e-mail addresses is not for the faint of heart, at least if you want to do it correctly.)

So given my People table, I could define a set of CHECK clauses that would ensure that the first_name field is non-empty. In other words, the first_name cannot be NULL, but it also cannot be the empty string:

```
ALTER TABLE People ADD CONSTRAINT
  people_first_name_non_empty_chks CHECK (first_name <> '');
```

Note that although I could add a number of checks within this single

constraint, I prefer not to do so. That gives me greater flexibility to add and remove constraints, and it also ensures that when a constraint is violated, PostgreSQL will accurately tell me which one it was.

Now, how can I implement this in my Rails migration, and do I want to? My answer, as you can imagine, is that this would indeed be a good thing to include in the database.

Once again, a Ruby gem comes to the rescue. This one, `sexy_pg_constraints`, was written by Maxim Chernyak, but it has since forked and is maintained by several other people.

I can include it in my Rails application by adding the following to my Gemfile:

```
gem 'Empact-sexy_pg_constraints', :require => 'sexy_pg_constraints'
```

and by uncommenting the following line in config/application.rb:

```
config.active_record.schema_format = :sql
```

Simply put, `sexy_pg_constraints` adds a number of additional attributes that you can pass to an `add_column` or `change_column` invocation within your migration. For example, let's say I want to make sure, as before, that the first_name column is never blank—neither NULL nor the empty string. I can do this by saying:

```
class AddConstraints < ActiveRecord::Migration
  def up
```

```
constrain :people, :first_name, :not_blank => true
end

def down
  deconstrain :people, :first_name
end
```

After I apply this migration, I discover the following in my table definition:

```
"people_first_name_not_blank" CHECK
→(length(btrim(first_name::text)) > 0)
```

In other words, I am checking to ensure that after removing all whitespace from either side of the string, the length is greater than 0. Sounds like it worked to me!

sexy_pg_constraints comes with a large number of options, including whitelisting, blacklisting, matching e-mail addresses and checking the format of data. The only thing this gem is missing, by my estimate, is a way to get the model file to communicate automatically with the database and/or

the migrations, so that you don't have to add these in both places manually. Even so, by adding these constraints, you improve the integrity of your data without having to go too far out of the Rails migration framework.

Conclusion

Database constraints are there to save people from themselves, and they are a great feature offered by relational databases. In exchange for a bit of work up front, and for some small performance penalties during runtime, you can ensure that your data remains intact. I would argue that doing this sort of validation is important at both the database and the application levels. A number of Ruby gems, as I explained here, make it possible to do this sort of integration within Ruby on Rails.■

Reuven M. Lerner is a longtime Web developer, consultant and trainer. He is also finishing a PhD in learning sciences at Northwestern University. His latest project, SaveMyWebApp.com, went live this spring. Reuven lives with his wife and children in Modi'in, Israel. You can reach him at reuven@lerner.co.il.

Resources

Information about PostgreSQL and constraints is on the Web at <http://www.postgresql.org/docs/current/static/ddl-constraints.html>, and this should be read by anyone interested in the subject.

The Foreigner and Immigrant gems are on GitHub at <https://github.com/jenseng/foreigner> and <https://github.com/jenseng/immigrant>, respectively.

A recent branch of the sexy_pg_constraints gem is at https://github.com/carbonfive/sexy_pg_constraints.

TXLF

Come be a part of the Texas Linux Fest 2012 - The largest
Linux and Open Source Software conference
in the Lone Star State!

Norris Conference Center in
San Antonio, TX
August 3-4th

register @ <http://texaslinuxfest.org>



texaslinuxfest





DAVE TAYLOR

Scripting Lite: Shell Aliases and Functions

You've probably never paid much attention to the aliases in your .bashrc, but Dave has. In this column, he talks about aliases and command-line functions as an easy way to expand your command-line options.

Let's take a side trip this month, a journey away from the busy thoroughfare of shell script programming and into the relatively peaceful eddy of shell aliases. You probably are thinking "Yeah, that's two paragraphs. Then what?" But stick with me—there's a bit more to aliases than you may have realized.

To start, you can see what system and shell aliases you have simply by typing alias on the command line:

```
$ alias
alias adb='/home/taylor/Documents/Android\ SDK/platform-tools/adb'
alias grep='grep --color=always'
alias ksnap='/home/taylor/Documents/Android\ SDK/tools/ddms'
alias ls='ls -F'
alias scale='/home/taylor/bin/scale.sh'
alias vps='ssh taylor@intuitive.com'
```

For the most part, these are

straightforward substitutions of one word for another. Type in vps, and the shell expands that to be the command ssh taylor@intuitive.com.

Look at the alias for ls though. You can start to see a tiny bit of the sophistication that the shell has by realizing that it's not an alias loop. I mean, ls = ls -F, so shouldn't ls -F expand to ls ls -F -F, and so on? Fortunately, it doesn't, and that's one reason you easily can tweak and expand your command-line capabilities with aliases.

In fact, my sequence is often: type in the command until it's complex enough for an alias, which works until it's sufficiently complex for a shell script.

Let me show you a handy trick by demonstrating how I'd turn a complex

Bash actually has a ton of handy little shortcuts like that, and they're doubly useful if you have complex directory names and filenames!

find command into an alias:

```
$ find . -name "*core*" -print
```

Now let me use a shell shortcut, `!!`, for the most recent command I've typed in within a more-complicated sequence:

```
$ echo "alias findcore=\"!!\" >> ~/.bashrc
```

There's no feedback, so what did it actually do? To find out, examine the last line of the `.bashrc`:

```
$ tail -1 ~/.bashrc
alias findcore="find . -name *core* -print"
```

Nice. Easy. When your alias has become sufficiently complicated that a shell script would be easier, you can do the same basic thing again:

```
$ alias findcore > findcore.sh
```

Or, if you're really organized like I am, perhaps you could put the script into your `~/bin` directory, which changes the last line to:

```
$ alias findcore > ~/bin/findcore.sh
$ vi !$
```

There's another shell shortcut for you: `!$` is the last word of the previous command—in this case, it'll be the filename you just created. Bash actually has a ton of handy little shortcuts like that, and they're doubly useful if you have complex directory names and filenames!

Complicated Aliases

The most common problem with an alias is that you want to be able to have the alias process the arguments you've specified, but not just automatically at the end of the command sequence. For the `ls` alias above, it's no big deal, but what if you wanted to have something like this:

```
alias ls="ls -F | tee -a ~/ls.log"
```

This would let you have a running log of all output from the `ls` command. The alias works fine until you invoke it with something like:

```
$ ls /home
```

in which case it's expanded to:

```
ls -F | tee -a ~/ls.log /home
```

That's going to generate an error.

To get this to work, you need to switch from an alias to a shell function. It's not too common on the command line, but quite reasonable for your .bashrc. Here's how I'd write it:

```
list() {
    ls -F $* | tee -a ~/ls.log
}
```

Experimentation reveals that you can shrink things down to a minimum of two lines, but no shorter. So, do something like this:

```
list() { ls -F $* | tee -a ~/ls.log }
```

and the shell will give you the secondary prompt (usually >) waiting for the closing curly brace. Because you're going to drop this into the .bashrc, it doesn't really matter much.

Let's do some more interesting things with these functions, fancier aliases that demonstrate a bit of the power of these command-line functions.

First off, convert to uppercase:

```
upper() {
    echo $1 | tr '[a-z]' '[A-Z]'
}
```

That looks good, but there's a flaw in this function. Can you see it? The problem is that if I use \$1, I don't get multiword input:

```
$ upper linux journal rocks
LINUX
```

Oops. It's easily fixed by substituting \$* instead. While I'm at it though, let's use defined letter groups and sets rather than explicit ranges:

```
upper() {
    echo $* | tr '[:lower:]' '[:upper:]'
}

$ upper linux journal rocks
LINUX JOURNAL ROCKS
```

That's better.

A very nice feature of most modern Linux systems is a change directory command pair called pushd and popd. Use pushd instead of cd to change your current working directory, and the system will remember where you were. Do a quick popd, and you're back. That's helpful when you're bouncing around the filesystem, but what if you don't have these available?

Implementing a similar directory stack in a couple command functions is actually straightforward. Here's my version:

```
push() {
    current=$(pwd)
    cd $1 ; echo "Moved to $1"; ls -CF
}

pop() {
    echo "Moving back to $current"
    cd $current
}
```

Let's give it a shot:

You can imagine how you really could make some changes to how the Linux command line works for an unsuspecting colleague, but perhaps you should leave that for when a new person joins your team and you feel the need for some hazing.

```
$ push $HOME
Moved to /home/taylor
Desktop/      Dropbox/      Movies/      Presentations/
Documents/    Google Drive/  Music/       Public/
Downloads/   Library/      Pictures/    Sites/
```

What's happened to the "current" shared variable? Let's have a look:

```
$ echo $current
/home/taylor/Desktop
```

To switch back to the previous working directory, therefore, all that's needed is:

```
$ pop
Moving back to /home/taylor/Desktop
```

That's easy!

To make this more sophisticated, you would need to use an array of directory names and keep track of your array depth. It's not too difficult, but I'll leave that one for an enthused reader!

One big difference between aliases and functions, I should note, is that you can use an alias with the same name as a command, but most shells complain about attempts to define a function

where the name collides with an alias:

```
$ ls() {
-bash: syntax error near unexpected token `('
$ unalias ls
$ ls() {
    echo "file listing disabled"
}
$ ls /
file listing disabled
```

Sneaky, isn't it? You can imagine how you really could make some changes to how the Linux command line works for an unsuspecting colleague, but perhaps you should leave that for when a new person joins your team and you feel the need for some hazing.

Anyway, that's it for aliases and command-line functions this time. I encourage you to explore and experiment with what you can do in your .bashrc to make your command-line interaction more pleasant and efficient. ■

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

N900 with a Slice of Raspberry Pi

Now that I've finally got my Raspberry Pi, let the hacks begin. First off: N900 Raspberry Pi hacks.

It may not come as a surprise to anyone who regularly reads my column that I tried to be first in line to order the Raspberry Pi. I mean, what's not to like in a \$35, 700MHz, 256MB of RAM computer with HDMI out that runs Linux? In the end, I didn't make the first batch of 10,000, but I wasn't too far behind either. So, now that I've had a Raspberry Pi for a week, I've already found a number of interesting uses for it. You can expect more Raspberry Pi columns from me in the future (possibly including an update to my beer fridge article), but to start, in this article, I talk about a combination of two of my favorite pocket-size Linux computers: the Raspberry Pi and my Nokia N900.

At first you may wonder why combine the two computers. After all, both are around the same size and have similar

initial hardware specs. Each computer has its own strengths, such as cellular networking and a touchscreen on the N900 and an Ethernet port and HDMI video output on the Raspberry Pi. In this article, I explain how to connect the N900 to the Raspberry Pi in a private USB network, share the N900's cellular connection, and even use the N900 as a pocket-size display. In all of the examples, I use the default Debian Squeeze Raspberry Pi image linked off the main <http://www.raspberrypi.org> page.

Set Up USB Tethering

The first step to using the N900 with the Raspberry Pi is to set up a private USB network between the two devices. There are a number of ways to do this, but I've found that the most effective way is via the Mobile Hotspot application on the

N900. This program started as a way to allow you to tether your computer with your N900 by turning the N900 into a wireless access point; however, because it uses WEP for security, I always favored using Mobile Hotspot's lesser-known USB networking option. That way, I not only get to tether my laptop, but because tethering uses up quite a bit of battery power, by being plugged in over USB, my laptop can keep my N900 charged as well.

By default, the Raspberry Pi is not set up to enable USB networking, but luckily, this is easy to set up. Just log in to your Raspberry Pi and edit the /etc/network/interfaces file as root. Below where it says:

```
iface eth0 inet dhcp
```

add:

```
iface usb0 inet dhcp
```

Now, launch the Mobile Hotspot program on your N900 and make sure it is configured so that the Interface is set to USB, as shown in Figure 1. Then connect the Raspberry Pi to your N900, which should prompt you to select between Mass Storage mode or PC Suite mode. Choose PC Suite mode, and then click the Start button on the Mobile Hotspot GUI. This automatically should set up the USB network for you, and you should see logs like the following in your Raspberry Pi's /var/log/syslog:

```
Jan  1 01:04:44 raspberrypi kernel: usb 1-1.3: new high speed
  ↵USB device number 5 using dwc_otg
Jan  1 01:04:44 raspberrypi kernel: usb 1-1.3: New USB device found,
  ↵idVendor=0421, idProduct=01c8
Jan  1 01:04:44 raspberrypi kernel: usb 1-1.3: New USB device
  ↵strings: Mfr=1, Product=2, SerialNumber=0
Jan  1 01:04:44 raspberrypi kernel: usb 1-1.3:
  ↵Product: N900 (PC-Suite Mode)
Jan  1 01:04:44 raspberrypi kernel: usb 1-1.3:
  ↵Manufacturer: Nokia
Jan  1 01:04:47 raspberrypi kernel: cdc_ether 1-1.3:1.8: usb0:
  ↵register 'cdc_ether' at usb-bcm2708_usb-1.3,
  ↵CDC Ethernet Device, 66:77:ea:fa:12:8c
Jan  1 01:04:47 raspberrypi kernel: usbcore: registered new
  ↵interface driver cdc_ether
Jan  1 01:04:47 raspberrypi kernel: cdc_acm 1-1.3:1.6:
  ↵ttyACM0: USB ACM device
Jan  1 01:04:47 raspberrypi kernel: usbcore: registered
  ↵new interface driver cdc_acm
Jan  1 01:04:47 raspberrypi kernel: cdc_acm: USB Abstract
  ↵Control Model driver for USB modems and ISDN adapters
```

The point-to-point network that is set up turns your N900 into a gateway with the IP address of 10.8.174.1, and your Raspberry Pi is given the IP 10.8.174.10, which you can see from the output of ifconfig on the Raspberry Pi:

```
usb0    Link encap:Ethernet  HWaddr 66:77:ea:fa:12:8c
        inet addr:10.8.174.10  Bcast:10.8.174.255  Mask:255.255.255.0
              UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
              RX packets:309 errors:0 dropped:3 overruns:0 frame:0
              TX packets:204 errors:0 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:1000
              RX bytes:25703 (25.1 KiB)  TX bytes:30676 (29.9 KiB)
```

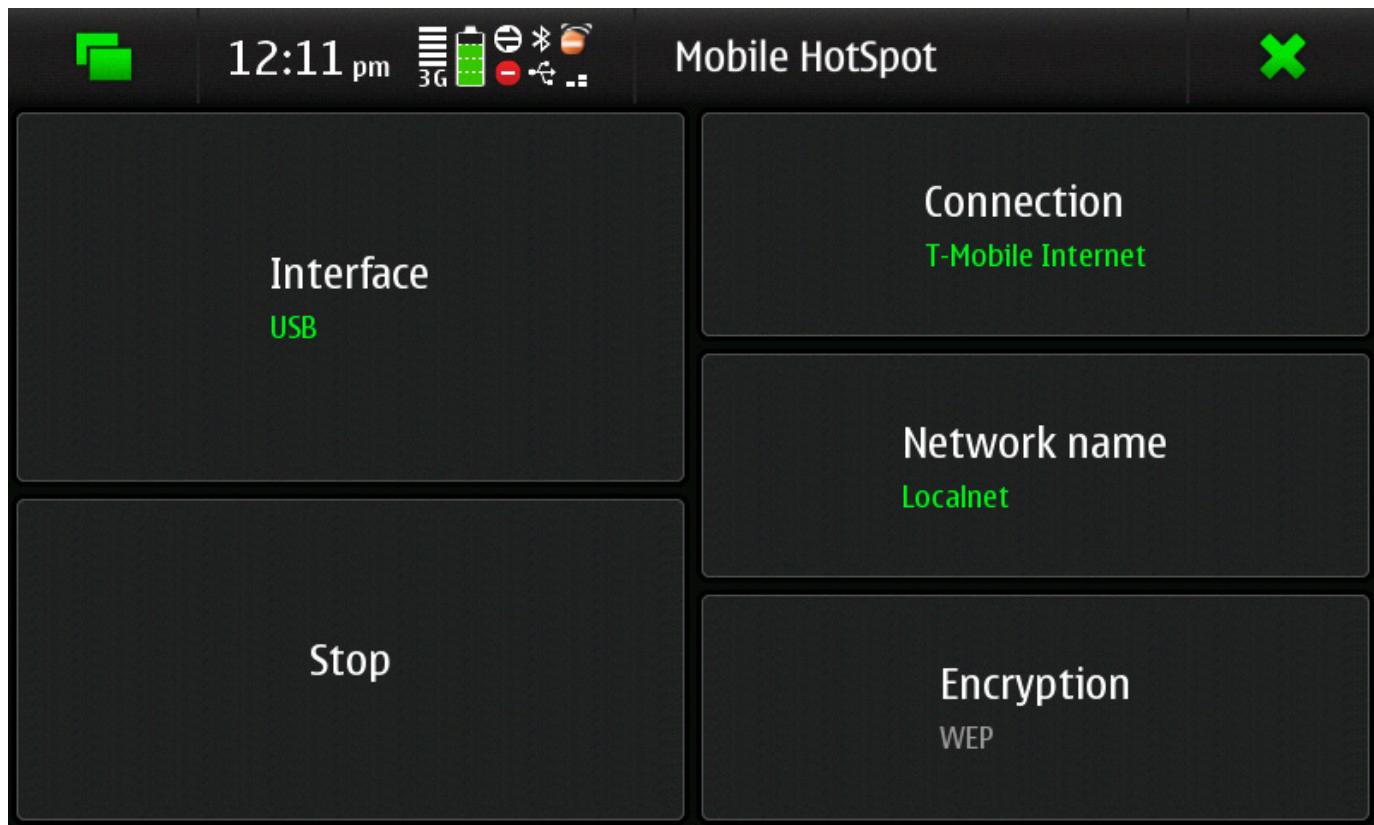


Figure 1. Mobile Hotspot Configured for USB Tethering

Because the N900 is set up as the gateway, your Raspberry Pi can use that cell-phone network for any outbound connections without having to worry about plugging in the Ethernet port. In addition, if you start the SSH service on the Raspberry Pi (`sudo service start ssh`) or better, if you make sure it's enabled at boot, you can `ssh` into your Raspberry Pi from the N900 with `ssh pi@10.8.174.10` from a terminal. If for some reason when you try to `ssh` to this IP, you get a "no route to host" error, investigate your Raspberry Pi logs and confirm that you truly are getting the 10.8.174.10 IP. I found on my system that occasionally I would get a .11 or .12 IP instead.

N900 as a Remote Display

Now that I have a point-to-point local network between my N900 and Raspberry Pi, I've removed the need to connect a network cable, but I still have that pesky HDMI cable to get rid of. After all, you may want to hack on your Raspberry Pi in places where you don't have an HDMI-enabled display handy. Luckily, with a few tweaks you can use your N900 touchscreen as a display for the Raspberry Pi and still be able to use a keyboard or mouse you have connected to the Raspberry Pi for input.

Unfortunately, I can't just connect the composite out or HDMI out of the Raspberry Pi into the N900, but

what I can do is take advantage of the relatively low-latency local USB network and share the N900 X display over VNC. The first step is to install the `x11vnc` package on the Raspberry Pi with `sudo apt-get install x11vnc`.

Once `x11vnc` is installed, I need to set it up so that it automatically launches when X launches. I suppose this isn't absolutely necessary. After all, you could `ssh` in every time and start it yourself, but I think having it automatically launch is much more convenient. To do this, create a file called `/home/pi/.config/autostart/x11vnc.desktop` with the following contents:

[Desktop Entry]

```
Name=X11VNC Server
Comment=Share this desktop by VNC
Exec=x11vnc -forever
Icon=computer
Terminal=false
Type=Application
StartupNotify=false
#StartupWMClass=x11vnc_port_prompt
Categories=Network;RemoteAccess;
```

Next, I need to change the settings for the `x11-common` package so that it allows X sessions to be launched by any user. This is necessary so that I can run `startx` at boot time automatically. Without this change, X will detect it's not being run from a console session, and it will error out. To do this, run `sudo dpkg-reconfigure x11-common`, and when prompted to select "Users allowed

to start the X server", select Anybody.

The final step is to start X at boot time. There are a number of ways to do this, but one of the easiest ways on the Raspberry Pi is via the `/boot/boot.rc` file. By default, the file is not there, but if present, it allows you to specify commands to run during the boot process so you can do things like enable SSH or start X. The following `/boot/boot.rc` file does both:

```
# sourced from rc.local on Raspberry Pi
#
# Name this file as "boot.rc" and put it on the boot
# partition if you want to run it.

# echo "Checking ssh and enabling if absent"
if ! ls /etc/rc5.d | grep "^S..ssh\$" >/dev/null; then
    insserv ssh
    service ssh start
fi
su pi -c startx
```

Once you boot the Raspberry Pi and set up the local USB network, if you `ssh` in and run `ps -ef`, you should see that `x11vnc` is running. Now you can launch a VNC client from the N900 (I prefer Presence VNC) and connect to 10.8.174.10, and you should see a copy of your Raspberry Pi X session (Figure 2). If you had the HDMI cable connected when the Raspberry Pi booted, the X session should be at full 1080p resolution, which might show up a bit small on the N900 screen. However, if you boot without HDMI connected

Sure, it's not as nice as a giant 1080p display, but then it's hard to fit one of those in your pocket.

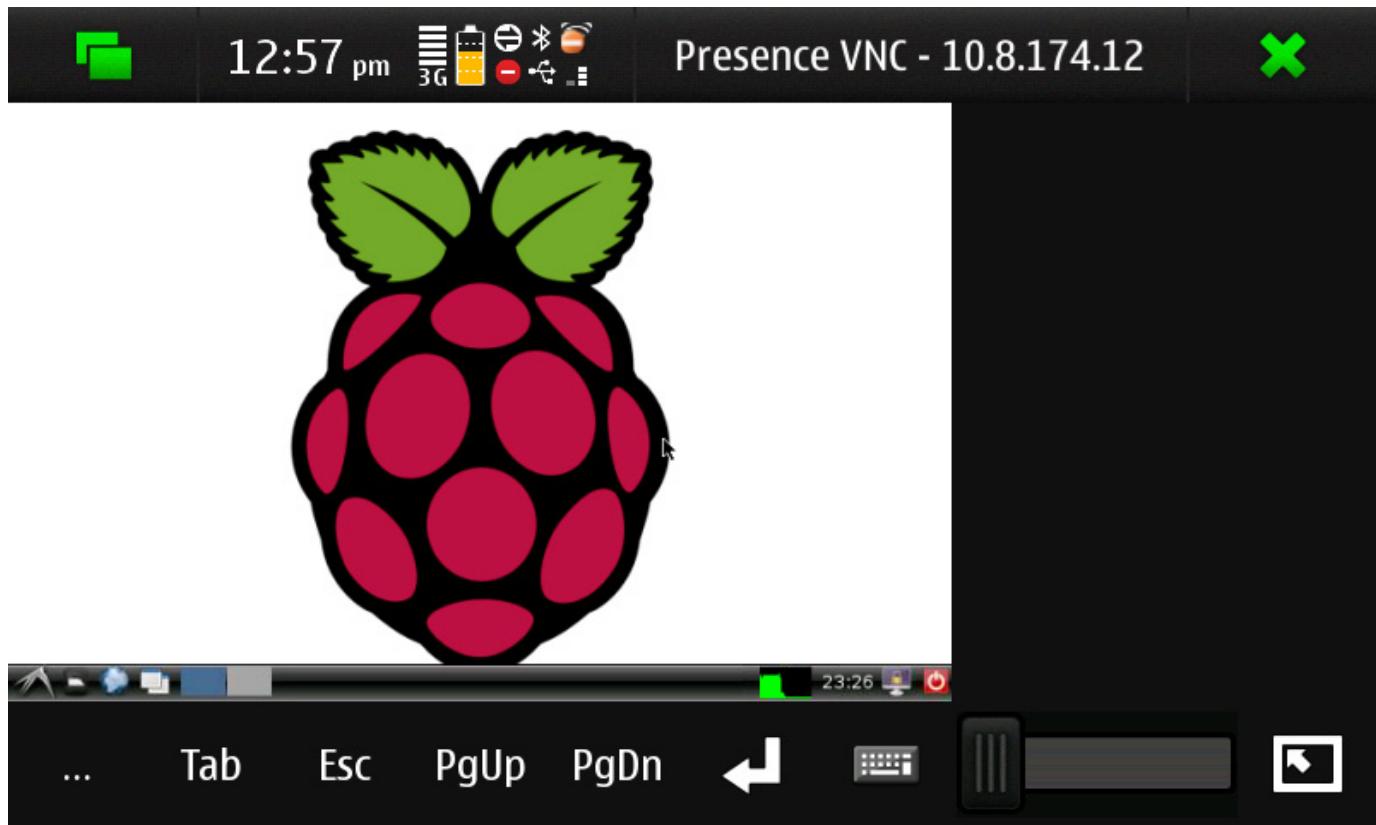


Figure 2. Presence VNC on N900 Connected to the Raspberry Pi

(which is the general use case for this hack), the X session will be configured for composite output and be at a more manageable 640x480. At this resolution, once you tell Presence VNC to go full screen, it uses up the full N900 display, and because you are sharing a real X session on the Raspberry Pi, you can use any keyboard or mouse you have plugged in to it. Sure, it's not as nice as a giant 1080p display, but then it's hard to fit one of those in your pocket.

Although I've talked about the N900 a

lot for this hack, the same principles should work to turn just about any device that can run a VNC client into a display for the Raspberry Pi, provided the two devices can connect over the network. In fact, if you are one of the many people who carry a color tablet around anyway, that would be a quite ideal display for the Raspberry Pi. ■

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.



CloudOps 2012SM

Cloud Computing for Asset Managers

Optimizing Cost Management, Operational Efficiency, Security, & Compliance

September 27 – 28, 2012

The Princeton Club

New York, New York



**The most comprehensive and pointed conference detailing the intersection of
cloud computing and asset management:**

- Physical and logical **security concerns** pertinent to the financial sector
- **Bottom dollar logistics:** Impacting cost management and measuring your ROI
- **Operational efficiency for asset managers:** Disaster recovery, mobility, compliance, multi-tenancy, scalability, flexibility, low-latency and more!
- Best practices for **vendor selection** and due diligence to ensure optimal leverage, security, and cost management
- **Cloud governance:** New paradigms for IT management, financial control of the cloud, delegated access, streamlining and maintaining data, access and ownership concerns
- **Cloud compliance:** Ensuring SEC/FINRA readiness, as well as ensuring that you know what your investors expect from you
- Defining the cloud and identifying appropriate platforms for your firm: From back-office and administrative systems to mission-critical applications and systems
- Cloud preparation and readiness: Cost profiling, initial forays, strategic planning, and governance

**To Register: Call 800-280-8440 or visit us at www.frallc.com
Mention FMP171 and save 15% off the standard rate**



SHAWN POWERS

Mirror, Mirror, Down the Hall

You can't host the whole Internet yourself, but you can host entire distributions!

If you're the sort of person who installs Linux a lot, or if you have a large number of servers/desktops to keep updated, having a local mirror can be a real time and bandwidth saver. You may have a huge Internet connection and want to

set up a public mirror, but for the sake of this article, let's assume you just want a personal mirror. I have a local mirror both at home and at work, and I've never regretted setting it up.

Before setting up your local mirror,

| Index of / | | | | |
|------------|-----------|-------------------|------|-------------|
| [ICO] | Name | Last modified | Size | Description |
| [DIR] | centos/ | 27-Jun-2012 11:28 | - | |
| [DIR] | partner/ | 27-Jun-2012 01:04 | - | |
| [DIR] | releases/ | 26-Jun-2012 21:40 | - | |
| [DIR] | ubuntu/ | 27-Jun-2012 02:10 | - | |

Apache/2.2.22 (Ubuntu) Server at [REDACTED] Port 80

Commands: Use arrow keys to move, '?' for help, 'q' to quit, '<->' to go back.
 Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list

Figure 1. My local mirror is simple, but very useful.

there are a few considerations you should keep in mind. First, mirrors tend to take a lot of space. My current Ubuntu mirror comes in at 550GB, which doesn't include the ISO images or the Partner repository. Granted, that has multiple architectures, but still, the hard drive space is significant.

Another issue with maintaining a local mirror is that it takes a long time to download initially. Depending on your Internet connection and the speed of the mirror you choose to replicate, it can take days or even weeks! With

my 16Mbit connection at home, the daily updates take about 30 minutes on average, but that initial download took a couple days.

Before delving into the process of setting up and maintaining your mirror, if you're an Ubuntu user, you might be just as happy installing squid-deb-proxy. Using squid-deb-proxy effectively will share any previously downloaded updates any computer on your network makes. With the help of avahi, it can be done automagically! See the squid-deb-proxy sidebar for more details.

squid-deb-proxy

If you're using only Ubuntu and don't want to set up an entire mirror of files you'll rarely (if ever) use, squid-deb-proxy might be right up your alley. On your server, you need to install two packages: squid-deb-proxy and avahi-utils. When they're installed, start them up:

```
sudo start squid-deb-proxy  
sudo start squid-deb-proxy-avahi
```

And you're done! The only other step is to install the squid-deb-proxy-client package on your workstations, and any deb files downloaded first will look at your local network cache before downloading from the Internet. Because it uses avahi (Zeroconf), the process is automatic, so if you take your laptop out of your network, it updates like normal from the Internet!

The only downside is that the installation process isn't squid-deb-proxy-aware yet, so the network installs still will come from the Internet. If you're just looking to speed up updates, however, squid-deb-proxy is simple to set up!

In order to be a proper netizen, you should locate a mirror that is close to you.

Hosting Options

For the sake of simplicity, I recommend using HTTP to serve your mirrored files. You can add other protocols if you like (rsync, FTP or NFS), but at the very least, I recommend HTTP. Because most mirrors use a subdirectory for file storage, you generally can just add a folder or symlink to your current Web server and never touch the Web server's config files. For this article, I assume you're adding the mirror to your /var/www/default folder. If you're using a distro that is not Debian-based, adjust your folder structure as necessary.

The Initial Sync

Eventually, I'll explain how to automate the update process, but for the initial sync, I recommend watching to make sure it works right. The method for mirroring Ubuntu is similar to mirroring CentOS; it's basically a single rsync command.

In order to be a proper netizen, you should locate a mirror that is close to you. A list of Ubuntu mirrors is located at <https://launchpad.net/ubuntu/+archivemirrors>, and a list of CentOS mirrors is at <http://www.centos.org/modules/tinycontent/index.php?id=13>.

Once you've determined the mirror you

want to use (make sure to choose one that supports rsync), create your local mirror directories:

```
sudo mkdir /var/www/default/ubuntu  
sudo mkdir /var/www/default/centos
```

Again, you can adjust the location of your mirrors however you like, the above just creates folders in the default Apache server location in Ubuntu. Using your local mirror address, type the following to mirror the Ubuntu repository:

```
sudo rsync -a --progress \  
rsync://your.ubuntu.mirror.com/ubuntu \  
/var/www/default/ubuntu
```

Then make a pot of coffee—or 20. You should get screens full of text as your local mirror is created. Once that is finished, a similar one-liner will create your CentOS mirror:

```
sudo rsync -a --progress \  
rsync://your.centos.mirror.org/CentOS/* \  
/var/www/default/centos
```

This will take a long time too. You may want to switch to decaf for this second batch of coffee.

Look at Yourself in the Mirror

Assuming everything was done correctly, you should have a working mirror! Test it out by going to `http://your.server.ip/ubuntu` and `http://your.server.ip/centos`, and see if you get a listing that looks like a mirror. If you get a message about “forbidden” from your Web server, you’ll need to add indexing options in order to see directory listings. Even without the indexing options added, however, the mirror itself should work.

Now What?

Depending on the mirror you chose for CentOS, you may or may not have a mirror that includes ISO files. If you don’t want the ISO files, or a particular release version, you can, of course, modify the `rsync` command appropriately. For example, adding `--excludeisos` will completely ignore the entire `isos` folder, saving you lots of room and lots of bandwidth. I encourage you to tweak the `rsync` commands to create a mirror that serves your needs.

Once you have your mirrors tweaked how you like them, I recommend giving them a good test. Install a system from your mirror. With Ubuntu, the best way to do that is by getting the minimal install CD from <https://help.ubuntu.com/community/Installation/MinimalCD> and booting it via CD or USB. When the installer asks you to choose a mirror, scroll all the way to the top of the list and enter

your server address manually. If all goes well, the system should find your server and install Ubuntu completely over the network. It’s really pretty cool to see.

For CentOS, you need to find a mirror containing ISO files and download the netinstall ISO. Look in the `iso` folder inside the version number folder for the `iso` file ending with `netinstall.iso`.

Booting from the netinstall ISO is simple, but knowing the mirror URL is a little tricky. You’ll want to use your local URL for something like this: `http://server.ip/centos/6.2/os/i386`. Obviously, you’ll need to replace the “6.2” with whatever version of the netinstall ISO file you downloaded. The installer should download `install.img`, and from there the installer should look just like an install using CDs or DVD.

Keeping It All Up to Date

Hopefully, you now have a fully working mirror on your local server. Of course, in the time it took you to read this far into the article, it’s probably already outdated. That means you need some way to keep things updated, preferably without any work on your part. Thank you cron.

At the very least, the commands shown above to do the initial `rsync` will do the trick if you put them into cron. I recommend removing the `--progress` flag and adding `--delete-after` so that you don’t get so much feedback and so obsolete files are deleted from your mirror. If you get tired of the nightly

Listing 1. Mirror Script

```
#!/bin/bash
## Mirror Script /usr/local/bin/mirror-sync.sh
## Originally created by Peter Noble, modified by many nerds

## Point our log file to somewhere and set up our admin email
log=/var/log/mirrorsync.log
adminmail=shawn@brainofshawn.com

# Set to 0 if you do not want to receive email
sendemail=1

# Subject is the subject of our email
subject="Your Mirror Did Its Thing"

# Set up the server to mirror
remote=rsync://archive.ubuntu.com/ubuntu

# Set up the local directory for your mirror
local=/var/www/default/ubuntu

## Initialize some other variables
complete="false"
failures=0
status=1
pid=$$

echo "`date +%x-%R` - $pid - Started Mirror Sync" >> $log
while [[ "$complete" != "true" ]]; do

    if [[ $failures -gt 0 ]]; then
        ## Sleep for 5 minutes for sanity's sake
        ## The most common reason for a failure at this
        ## point is that the rsync server is handling
        ## too many concurrent connections.
        sleep 5m
    fi

    if [[ $1 == "debug" ]]; then
        echo "Working on attempt number $failures"
        rsync -a --delete-after --progress $remote $local
        status=$?
    else
        rsync -a --delete-after $remote $local >> $log
        status=$?
    fi

    if [[ $status -ne "0" ]]; then
        complete="false"
        (( failures += 1 ))
    else
        echo "`date +%x-%R` - $pid - Finished" >> $log
        # Send the email
        if [[ -x /usr/bin/mail && "$sendemail" -eq "1" ]]; then
            mail -s "$subject" "$adminmail" <<OUTMAIL
Summary of Mirror Synchronization
PID: $pid
Failures: $failures
Finish Time: `date`

Sincerely,
$HOSTNAME

OUTMAIL
        fi
        complete="true"
    fi
done

exit 0
```

Rather than putting the rsync commands directly into your crontab, I recommend creating a couple scripts with some error management.

e-mail messages from the output, you might want to add > /dev/null to the end of the cron line, so you're notified only if there are errors.

Unfortunately, things often go wrong. Rather than putting the rsync commands directly into your crontab, I recommend creating a couple scripts with some error management. The script I use (Listing 1) is pieced together from several people, cobbled a bit on my own, and has been shared all over. Feel free to take from it and modify it as you like. After reading this article, it should be self-explanatory, and it can be adapted for Ubuntu or CentOS.

But I Want More!

Once you have the basic mirrors on your server, you'll quickly notice a few things are missing. With Ubuntu, you won't have any ISO images. If you want to have a local copy of ISOs, you can get that with another simple rsync command:

```
rsync -a --progress \
rsync://rsync.releases.ubuntu.com/releases \
/var/www/default/releases/
```

Just like with the main archive, on subsequent rsyncs, I recommend

removing the --progress flag and adding --delete-after. That will keep your mirror clean. It's possible to create a script for this process as well, but honestly, I just put the rsync one-liner in my crontab. There aren't too many changes to the ISO repository; it really changes only when there is a release.

Ultra Small Panel PC

- ARM9 400Mhz Fanless Processor
- Up to 1 GB Flash & 256 MB RAM
- 4.3" WVGA 480 x 272 TFT LCD
- Analog Resistive Touchscreen
- 10/100 Base-T Ethernet
- 3 RS232 & 1 RS232/422/485 Port
- 1 USB 2.0 (High Speed) Host port
- 1 USB 2.0 (High Speed) OTG port
- 2 Micro SD Flash Card Sockets
- SPI & I2C, 4 ADC, Audio Beeper
- Battery Backed Real Time Clock
- Operating Voltage: 5V DC or 8 to 35V DC
- Optional Power Over Ethernet (POE)
- Optional Audio with Line-in/out
- Pricing starts at \$375 for Qty 1

New - PPC-E4+



2.6 KERNEL

The PPC-E4+ is an ultra compact Panel PC that comes ready to run with the Operating System fully configured on the onboard flash. The dimensions of the PPC-E4+ are 4.8" by 3.0", about the same as that of popular touch cell phones. The PPC-E4+ is small enough to fit in a 2U rack enclosure. Apply power and watch either the Linux X Windows or the Windows CE User Interface appear on a vivid 4.3" color LCD. Interact with the PPC-E4+ using the responsive integrated touch-screen. Everything works out of the box, allowing you to concentrate on your application rather than building and configuring device drivers. Just Write-It and Run-It.

www.emacinc.com/panel_pc/ppc_e4+.htm



EMAC, inc.
EQUIPMENT MONITOR AND CONTROL

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

Because there is no rsync server for the Partner repo, the best way to create and maintain a mirror is by using debmirror.

The Partner repository is a little trickier to mirror. There is no rsync server set up for mirroring the Partner repository, and I suspect it is because the Partner repo contains commercial software and mirroring it violates distribution rights or some such thing. I'm not a lawyer, but as long as you're mirroring on your local network for your own personal use, I don't think you run any risk of breaking the law.

Because there is no rsync server for the Partner repo, the best way to create and maintain a mirror is by using debmirror. debmirror is a neat tool that uses the HTTP protocol to download packages and organizes them to match the remote repository. It can be used for mirroring the entire Ubuntu (or Debian) repository, but I usually use rsync for the main archive, because it's so simple.

Setting up debmirror requires some work with GPG keys and modifying a script to fit your needs. The folks at Ubuntu have outlined the process (although they don't mention it can be used to mirror the Partner repo!) here: <http://help.ubuntu.com/community/Debmirror>.

I've also created a quick little video outlining how I keep my mirrors in sync on the *LJ* Web site. Here is a

video showing my setup:

<http://www.linuxjournal.com/video/mirror-partner-repo-canonical>.

Waste Not, Want Not?

Hosting your own mirrors takes a lot of disk space. The initial syncs take a ton of bandwidth. If you run a network that would benefit from a local mirror, however, it really can be worth the effort.

Once you have a local mirror set up, if you can afford the bandwidth, I encourage you to become a public mirror. Canonical and CentOS have methods and requirements for becoming a public and official mirror. Although not everyone can do it, it's folks like you and me who make the Open Source community so strong. Even if you can't contribute code, perhaps you're in a situation to contribute bandwidth. I'm not, as neither of my locations has enough bandwidth to support a public mirror, but perhaps someone reading this article will! ■

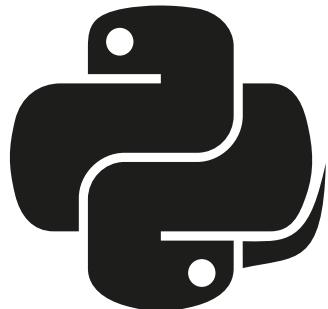
Shawn Powers is the Associate Editor for *Linux Journal*. He's also the Gadget Guy for LinuxJournal.com, and he has an interesting collection of vintage Garfield coffee mugs. Don't let his silly hairdo fool you, he's a pretty ordinary guy and can be reached via e-mail at shawn@linuxjournal.com. Or, swing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

nz.pycon.org

Registrations
and
Talk Submissions
now open!



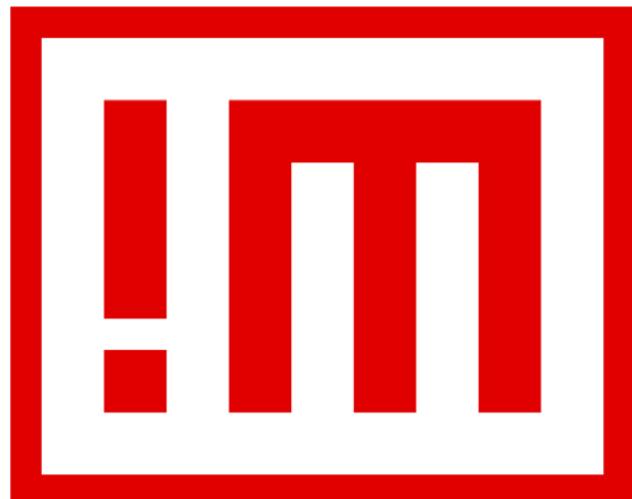
KIWI PYCON
2012
DUNEDIN
1st-2nd Sept.
New Zealand



NoMachine's NoMachine

Even if your friends and colleagues aren't enlightened enough to run Linux, you still can rule their PCs remotely. NoMachine's new NoMachine v.4 is a free, remote desktop control application that now runs on Mac OS and Windows in addition to the previously supported Linux. This multiplatform support allows any PC running Linux, Mac OS or Windows to control any remote computer running these same OSes, including full access to files and programs, real-time audio and video streaming, secure file sharing even through firewalls, screen capture and printing to any printer, all at a high level of performance.

<http://www.nomachine.com>



Sonatype Insight for Continuous Integration



about—the discovery of security and licensing issues with third-party components. Insight for CI, Sonatype's latest intelligent tool for component-based software development, enables developers to find quality, security and licensing problems and enforce open-source policy at build time, before fixes become costly and time consuming. Insight for CI supports agile development processes with analysis of every component in every build, alerting developers immediately of any changes or policy violations that put their project at risk. Support for Hudson and Jenkins also is included. Sonatype hosts the Central Repository, the software industry's primary source for open-source components, housing more than 300,000 components and serving more than five billion requests per year.

<http://www.sonatype.com>

Sonatype Insight for Continuous Integration (CI) solves a major problem that anyone developing software today should care



Userful MultiSeat 5

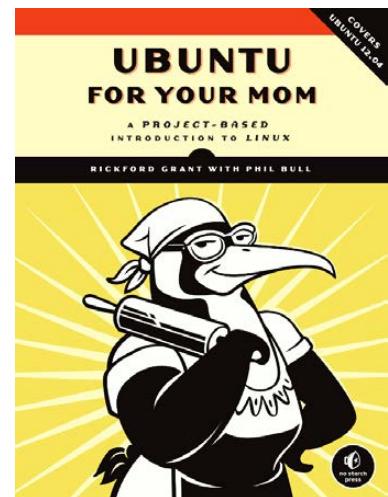
Are you a solution provider or OEM looking for new revenue opportunities? If so, Userful Corporation wants you to take a peek at the new Userful MultiSeat 5, a product it calls “the first Linux-based software to deliver virtual desktops on Ethernet true zero client hardware”. Userful MultiSeat 5 is reported to deliver PC-like performance at half the cost of thin clients, as well as support flawless playback of 20+ simultaneous HD video streams from a single, standard mid-range PC. The solution is targeted at applications like digital signage, kiosks and interactive touch displays, VDI, cloud computing, in-flight and in-room entertainment systems, casino gaming systems, government connectivity projects, schools, universities and others. Support is included for zero clients from OEM device-makers, such as HP, Viewsonic, Acer, Atrust, ThinGlobal and LG. Userful says that users worldwide now can afford blazing-fast computing performance with dozens of Ethernet true zero clients, powered only by a single PC or server located anywhere within the LAN.

<http://www.userful.com>

Rickford Grant and Phil Bull's *Ubuntu for Your Mom* (No Starch)

The Ubuntu Linux distribution makes Linux easy, and *Ubuntu for Your Mom* makes it even easier. Full of tips, tricks and helpful pointers, this pain-free guide is perfect for those interested in, but nervous about, switching to the Linux operating system. In the book, authors Rickford Grant and Phil Bull walk readers through common tasks like installing and playing games, accessing social networks, troubleshooting common hardware and software problems, connecting with the Ubuntu community, interacting with Windows and more. Other topics covered in a step-by-step manner include system installation and administration, enjoying various media, syncing mobile devices, editing and sharing digital photos and videos, creating documents, customizing the system's look and feel and working with the command line (or avoiding it altogether).

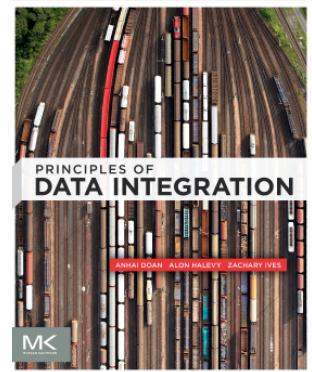
<http://www.nostarch.com>



AnHai Doan, Alon Halevy and Zachary Ives' *Principles of Data Integration* (Morgan Kaufmann)

How do you approach answering queries when your data is stored in multiple databases that were designed independently by different people? This is the challenge that the authorial trio of AnHai Doan, Alon Halevy and Zachary Ives solve with their new Morgan Kaufmann book *Principles of Data Integration*. The title is billed as the first comprehensive textbook of data integration, covering everything from theoretical principles to implementation issues and current challenges raised by the semantic Web and cloud computing. It provides an extensive introduction to the theory and concepts underlying today's data integration techniques, with detailed instruction for their application using concrete examples throughout to explain the concepts. It further provides the tools for developing a complete and concise package of algorithms and applications. A companion Web site and Facebook page add a dynamic range of interactive content and resources.

<http://store.elsevier.com>



Juniper Systems' and SDG Systems' RAMPAGE 6



The new RAMPAGE 6 rugged notepad with Android is the result of a collaboration between Juniper Systems and SDG Systems. The RAMPAGE 6 is a unique iteration of Juniper Systems' Mesa Rugged Notepad, which features a 5.7-inch viewing display, IP67 ingress protection rating for water and dust, integrated 2–5 meter GPS receiver, optional integration of a 1D/2D barcode scanner, and optional Class I, Division 2 certification for use in hazardous

locations. The Android 2.3 operating system on the RAMPAGE 6 offers many advantages for data collection, including easy multitasking, a modern user interface, rich programming environment, multiplatform development, abundant application data storage, open-source flexibility, and the opportunity for a custom Android interface developed by SDG Systems. Additionally, its optional kiosk mode allows only certain applications to be accessible by the user, successfully creating a single-purpose device without distractions.

<http://www.junipersys.com>, <http://www.sdgsystems.com>



Corsair Force Series 3 SSD Notebook Upgrade Kits

Give your beloved laptop a new lease on life with Corsair's Force Series 3 SSD Notebook Upgrade Kits, available in 120GB and 240GB sizes. Corsair says that these solid-state drives enable users to increase their laptop's performance, battery life and reliability in an affordable manner. SSDs provide significant advantages over standard hard drives, such as faster data read and write speeds, which can reduce software load and boot times. And because SSDs have no moving mechanical parts, they help laptops run cooler, reduce battery drain and resist data loss from bumps. The upgrade kit, which includes a USB-to-SATA cable and migration software, makes switching simple. The drives come in a slim 7mm-high case designed to fit in most space-constrained laptops.

<http://www.corsair.com>

Altova's MetaTeam

The beauty of the MetaTeam team management and project collaboration cloud service, says developer Altova, lies in "breaking down the mysteries of team organization into simple steps that anyone—not just project managers—can use effectively". Specifically designed to foster agile collaboration, MetaTeam makes teams more productive by creating an environment of transparency and information accessibility. At the same time, for larger teams and more elaborate projects, it raises accountability and commitment by spelling out individual roles and responsibilities. Yet MetaTeam is not just for developers. Business teams in any capacity can access its powerful tools to enhance collaboration and efficiency in managing projects of any kind. This cloud service includes integrated apps for managing to-dos, team member roles and responsibilities, project-specific terms and documentation, structured decision making, team member information and more. Threaded discussions and customizable e-mail alerts promote communication and clarity.

<http://www.altova.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.

A GUIDE TO USING LZO Compression IN Hadoop

**Deploy and implement MapReduce programs
that take advantage of the LZO compression
techniques supported by Hadoop.**

ARUN VISWANATHAN

Compression is the process of reducing the size of actual data by using an algorithm to encode the information. Compression provides the following benefits:

- Reduces the hard disk space occupied by the data.
- Uses lower transmission bandwidth.
- Reduces the time taken to copy or transfer the data from one location to another.

However, it also comes with a

kind of data, and it would be beneficial for enterprises to reduce the space used to store the data. The Hadoop framework supports a number of mechanisms, such as gzip, bzip and lzo to compress the data that is stored in HDFS.

LZO Compression

Lempel-Ziv-Oberhumer (or LZO) is a lossless algorithm that compresses data to ensure high decompression speed. It has the following characteristics:

- Data compression is similar to other popular compression techniques, such as gzip and bzip.

In Hadoop, using LZO helps reduce data size and causes shorter disk read times.

problem. Before putting the compressed data to some use, it first must be decompressed. After processing, the data must be compressed again. This increases the time it takes an application to process it before it can use this data.

As Hadoop adoption grows in corporate communities, you see data in terms of TeraBytes and PetaBytes that is stored in HDFS by large enterprises. Because Hadoop works well on commodity hardware, high-end servers are not required for storing and processing this

- It enables very fast decompression.
- It supports overlapping compression and in-place decompression.
- Compression and decompression happen on a block of data.
- It requires no additional memory for decompression except for source buffers and destination buffers.

In Hadoop, using LZO helps reduce

data size and causes shorter disk read times. Furthermore, the block structure of LZO allows it to be split for parallel processing in MapReduce programs. These characteristics make LZO suitable for use in Hadoop.

In this article, I look at the procedure for enabling LZO in Hadoop-based frameworks and look at a few examples of LZO's usage.

Prerequisites

The following describes the software that was set up in CentOS

5.5-based machines.

Set up and configure the Cloudera Distribution of Hadoop (CDH3) or Apache Hadoop 0.20.x in a cluster of two or more machines. Refer to the Cloudera or Apache Hadoop Web sites for more information on setting up Hadoop. Alternatively, you also could use the Cloudera demo VM as a single-node cluster for testing.

Next, install the LZO package in the system. Download and install the package from its Linux distribution repository. For this article, I installed

New: Intel Xeon E5 Based Clusters

Benchmark Your Code on Our Xeon E5 Based Tesla Cluster with:
AMBER, NAMD, GROMACS, LAMMPS, or Your Custom CUDA Codes



**Microway MD SimCluster with
8 Tesla M2090 GPUs
8 Intel Xeon E5 CPUs and InfiniBand
2X Improvement over Xeon 5600 Series**



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

Microway

this RPM: lzo-2.04-1.el5.rf.i386.rpm.

There are two ways to install the LZO-specific jars that can be used by Hadoop:

- Download and build the hadoop-lzo project from Twitter that will provide the necessary jars (see Resources).
- Download the prebuilt jars in RPM or Debian packages from the hadoop-gpl-packing project. For this article, I used this RPM: hadoop-gpl-packaging-0.2.0-1.i386.rpm.

The following binaries will be installed on the machine:

```
$HADOOP_GPL_HOME/lib/*.jar  
$HADOOP_GPL_HOME/native
```

HADOOP_GPL_HOME is the directory where the hadoop-lzo project will store the built binaries.

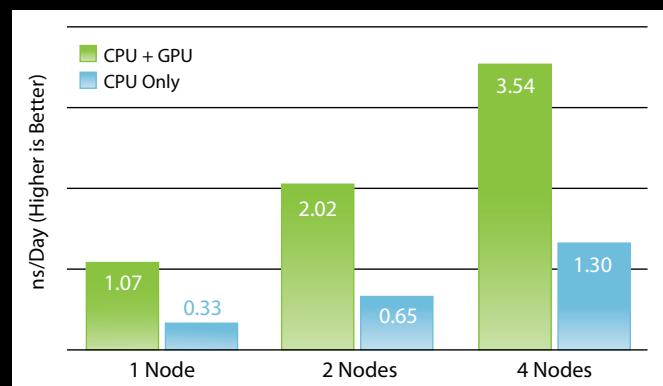
Using the prebuilt RPMs, the binaries will be installed in the /opt/hadoopgpl folder.

Note: if you are using a cluster of more than one machine, the above

Harness Microway's Proven GPU Expertise

**Thousands of GPU cluster nodes installed.
Thousands of WhisperStations delivered.**

- Award Winning BioStack - LS
- Award Winning WhisperStation Tesla - PSC with 3D



Configure Your WhisperStation or Cluster Today!
www.microway.com/tesla or 508-746-7341



three steps need to be done for all the machines in the cluster.

Deploy LZO in the Hadoop Ecosystem

First, install LZO for Hadoop. Then, add the Hadoop GPL-related jars to the Hadoop path:

```
$ cp $HADOOP_GPL_HOME/lib/*.jar  
$HADOOP_HOME/lib/
```

Next, run the following command, depending on the platform you're using:

```
$ tar -cBf - -C $HADOOP_GPL_HOME/native/ * |  
➥tar -xBvf - -C $HADOOP_HOME/lib/native/
```

Then, update the Hadoop configuration

files to register external codecs in the codec factory. Refer to Listing 1 to add the lines to the \$HADOOP_HOME/conf/core-site.xml file.

Refer to Listing 2 to add the lines to the \$HADOOP_HOME/conf/mapred-site.xml file.

The LZO files also need to be added in the Hadoop classpath. In the beginning of the \$HADOOP_HOME/conf/hadoop-env.sh file, add the entries as shown in Listing 3.

Install LZO for Pig

Add the Hadoop GPL-related jars to the Pig path:

```
$ cp $HADOOP_GPL_HOME/lib/*.jar $PIG_HOME/lib/
```

Next, run the following command,

Listing 1. Adding LZO Codecs to Hadoop core-site.xml

```
<!-- Add LZO Compression Codecs -->  
<property>  
    <name>io.compression.codecs</name>  
    <value>org.apache.hadoop.io.compress.GzipCodec,org.apache.hadoop  
➥.io.compress.DefaultCodec,com.hadoop.compression.lzo.  
➥LzoCodec,com.hadoop.compression.lzo.LzopCodec,org.apache.  
➥hadoop.io.compress.BZip2Codec</value>  
</property>  
<property>  
    <name>io.compression.codec.lzo.class</name>  
    <value>com.hadoop.compression.lzo.LzoCodec</value>  
</property>
```

depending on the platform you're using:

```
$ tar -cBf - -C $HADOOP_GPL_HOME/native/* |  
➥tar -xBvf - -C $PIG_HOME/lib/native/
```

Additionally, you'll need to make changes to the Pig Script and configuration files to register the external codecs in the codec factory.

Listing 2. Adding LZO Codecs to Hadoop mapred-site.xml

```
<!-- Add LZO Codecs details -->  
<property>  
  <name>mapreduce.map.output.compress</name>  
  <value>true</value>  
</property>  
<property>  
  <name>mapreduce.map.output.compress.codec</name>  
  <value>com.hadoop.compression.lzo.LzoCodec</value>  
</property>
```

Listing 3. Adding LZO-Related Libraries to hadoop-env.sh

```
export  
HADOOP_CLASSPATH="$HADOOP_HOME/lib/hadoop-lzo.jar:  
➥$HADOOP_CLASSPATH:$CLASS_FILES"  
  
# For 32-bit machines  
export  
JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native/Linux-i386-32:  
➥$HADOOP_HOME/lib/native  
# For 64-bit machines  
export  
JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native/Linux-amd64-64:  
➥$HADOOP_HOME/lib/native
```

Refer to Listing 4 to add the lines to the \$PIG_HOME/conf/pig.properties file,

and refer to Listing 5 to add the lines to the \$PIG_HOME/bin/pig script file.

Listing 4. Adding LZO Codecs to the Pig Properties File

```
mapred.output.compression.codec=
➥com.hadoop.compression.lzo.LzoCodec
mapred.map.output.compression.codec=
➥com.hadoop.compression.lzo.LzoCodec

# For 32-bit machines
mapred.child.java.opts=-Djava.library.path=
➥/opt/hadoopgpl/native/Linux-i386-32
# For 64-bit machines
#mapred.child.java.opts=-Djava.library.path=
➥/opt/hadoopgpl/native/Linux-amd64-64
```

Listing 5. Adding LZO-Related Libraries to the Pig Script

```
# For 32-bit machines
PIG_OPTS="$PIG_OPTS -Djava.library.path=
➥/opt/hadoopgpl/native/Linux-i386-32"
# For 64-bit machines
#PIG_OPTS="$PIG_OPTS -Djava.library.path=
➥/opt/hadoopgpl/native/Linux-amd64-64"

# Add hadoop lzo to CLASSPATH
for f in $PIG_HOME/lib/hadoop*lzo*.jar; do
  CLASSPATH=${CLASSPATH}:$f;
Done
```

Install LZO for HBase

Copy the Hadoop GPL jars to the HBase lib directory:

```
$ cp $HADOOP_GPL_HOME/lib/*.jar $HBASE_HOME/lib/
```

Run either of the following commands, depending on the platform you're using:

```
$ cp $HADOOP_GPL_HOME/native/Linux-i386-32/lib/*  
→$HBASE_HOME/lib/native/Linux-i386-32/
```

or:

```
$ cp $HADOOP_GPL_HOME/native/Linux-amd64-64/lib/*  
→$HBASE_HOME/lib/native/Linux- amd64-64/
```

Using LZO Compression

Let's look at a sample program for testing LZO in Hadoop. The code in Listing 6 shows a sample MapReduce program that reads an input file in LZO-compressed format. To generate compressed data for use with this word counter, run the lzop program on a regular data file. Similar sample code is provided with the Elephant-Bird Project.

Sample Program for Testing LZO in Pig

The PigLzoTest program shown in Listing 7 achieves the same result as the MapReduce program described previously with the only difference being it is written using Pig.

The last line in Listing 7 calls a

user-defined function (UDF) to write the output in LZO format. The code snippet in Listing 8 shows the contents of this class. The LZOTextStorer class shown in Listing 8 extends the com.twitter.elephantbird.pig.store.LzoBaseStoreFunc class provided by the Elephant-Bird Project for writing the output in the LZO format.

Sample Program for Testing LZO in HBase

To use LZO in HBase, specify a per-column family compression flag while creating the table:

```
create 'test', {NAME=>'colfam:', COMPRESSION=>'lzo'}
```

Any data that is inserted into this table now will be stored in LZO format.

Conclusion

In this article, I looked at the process for building and setting up LZO in Hadoop. I also looked at the sample implementation processes across MapReduce, Pig and HBase frameworks. LZO compression helps in reducing the space used by data that is stored in the HDFS. It also provides an added performance benefit due to the splittable block architecture that it follows. Faster read times of LZO compressed data with reduced decompression time makes it ideal as

Listing 6. Sample MapReduce Program to Test LZO in a Hadoop Cluster

```
/**  
 * MapReduce Word count Sample  
 * Input File ~V LZO compressed file  
 * Run com.hadoop.compression.lzo.LZOIndexer /  
 * com.hadoop.compression.lzo.  
 * DistributedLZOIndexer to create .lzo.index file to further  
 * improve the read speed of LZO compressed files.  
 * If the input lzo files are indexed, the input format will take  
 * advantage of it. The input file/directory is taken as the first  
 * argument. The output directory is taken as the second argument.  
 * Uses NullWritable for efficiency.  
 */  
public class SimpleLZOWC extends Configured implements Tool {  
  
    private SimpleLZOWC () {}  
  
    public static class LzoWordCountMapper extends Mapper<LongWritable,  
    Text, Text, LongWritable> {  
        private final LongWritable one = new LongWritable(1L);  
        private final Text word = new Text();  
  
        @Override  
        protected void map(LongWritable key, Text value, Context context)  
        throws IOException, InterruptedException {  
            String line = value.toString();  
            StringTokenizer tokenizer = new StringTokenizer(line);  
            while (tokenizer.hasMoreTokens()) {  
                word.set(tokenizer.nextToken());  
                context.write(word, one);  
            }  
        }  
    }  
  
    public int run(String[] args) throws Exception {  
        Job job = new Job(getConf());  
        job.setJobName("Simple LZO Word Count");  
  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(LongWritable.class);  
  
        job.setJarByClass(getClass());  
        job.setMapperClass(LzoWordCountMapper.class);  
        job.setCombinerClass(LongSumReducer.class);  
        job.setReducerClass(LongSumReducer.class);  
  
        // Use the custom LzoTextInputFormat class.  
        job.setInputFormatClass(LzoTextInputFormat.class);  
        job.setOutputFormatClass(TextOutputFormat.class);  
  
        FileInputFormat.setInputPaths(job, new Path(args[0]));  
        FileOutputFormat.setOutputPath(job, new Path(args[1]));  
  
        return job.waitForCompletion(true) ? 0 : 1;  
    }  
  
    public static void main(String[] args) throws Exception {  
        int exitCode = ToolRunner.run(new LzoWordCount(), args);  
        System.exit(exitCode);  
    }  
}
```

Listing 7. Sample Program to Test LZO in Pig

```
/**  
 * Pig Word count Sample  
 * Input File ~V LZO compressed file  
 * Run com.hadoop.compression.lzo.LZOIndexer /  
 * com.hadoop.compression.lzo.  
 * DistributedLZOIndexer to create .lzo.index file to further  
 * improve the read speed of LZO compressed files.  
 * Output - output directory is taken as the second argument.  
 *  
 * To generate data for use with this word counter, run lzop  
 * on a data file  
 * Usage: PigLzoTest <input-file> <output-folder>  
 */  
public class PigLzoTest {  
  
    /**  
     * @param args  
     */  
    public static void main(String[] args) {  
        try {  
            PigServer pigServer = new PigServer("mapreduce");  
            pigServer.registerJar("lib/elephant-bird-2.0.jar");  
            pigServer.registerJar("lzotest.jar");  
  
            runWordCountQuery(pigServer, args[0], args[1]);  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }  
  
    /**  
     * Pig Script for Word Count  
     * @param pigServer  
     * @throws IOException  
     */  
    public static void runWordCountQuery(PigServer pigServer,  
    String inputFile, String outputFile) throws IOException {  
        pigServer.registerQuery("A = load '" + inputFile + "';");  
        pigServer.registerQuery("B = foreach A generate  
    flatten(TOKENIZE((chararray)$0)) as word;");  
        pigServer.registerQuery("C = filter B by word matches '\\\\w+';");  
        pigServer.registerQuery("D = group C by word;");  
        pigServer.registerQuery("E = foreach D generate group as word,  
    COUNT(C) as count;");  
        pigServer.registerQuery("F = order E by count desc;");  
  
        pigServer.registerQuery("store F into '" + outputFile + "'  
    using com.hadoop.compression.lzo.LzoTextStorer();");  
    }  
}
```

Listing 8. Sample Pig UDF to Write the Output in LZO Format

```
/**  
 * Write the LZO file line by line, passing each  
 * line as a single-field Tuple to Pig.  
 */  
public class LzoTextStorer extends LzoBaseStoreFunc {  
    private static final TupleFactory tupleFactory_ =  
        TupleFactory.getInstance();  
  
    protected enum LzoTextLoaderCounters { LinesRead }  
  
    public LzoTextStorer() {}  
  
    @Override  
    public OutputFormat getOutputFormat() throws IOException {  
        return new TextOutputFormat<WritableComparable, Text>();  
    }  
  
    @Override  
    public void putNext(Tuple tuple) throws IOException {  
        if (writer == null)  
            System.out.println("Writer is null");  
  
        int numElts = tuple.size();  
        StringBuilder sb = new StringBuilder();  
        for (int i = 0; i < numElts; i++) {  
            String field;  
            try {  
                field = String.valueOf(tuple.get(i));  
            } catch (ExecException ee) {  
                throw ee;  
            }  
            sb.append(field);  
  
            if (i == numElts - 1) {  
                // Last field in tuple.  
                sb.append('\n');  
            } else {  
                sb.append('\t');  
            }  
        }  
  
        Text text = new Text(sb.toString());  
        try {  
            writer.write(NullWritable.get(), text);  
        } catch (InterruptedException e) {  
            throw new IOException(e);  
        }  
    }  
}
```

a compression algorithm for storing data in the HDFS. It is already a popular technique that is used by a number of social Web companies, such as Twitter, Facebook and so on, internally to store data. Twitter also has provided the open-source Elephant-Bird Project that provides the basic classes for using LZO. ■

Arun Viswanathan is a Technology Architect with the Cloud Labs at Infosys Ltd, a global leader in IT and business consulting services. Arun has about nine years of experience with expertise in Java; Java EE application development; and defining, architecting and implementing large-scale, mission-critical, IT solutions across a range of industries. He is currently involved in design, development and consulting for big-data solutions using Apache Hadoop.

Resources

Lempel-Ziv-Oberhumer Algorithm:

<http://en.wikipedia.org/wiki/Lempel%E2%80%93Ziv%E2%80%93Oberhumer>

Using LZO Compression in Hadoop: <http://wiki.apache.org/hadoop/UsingLzoCompression>

Cloudera Demo VM: <https://ccp.cloudera.com/display/SUPPORT/Cloudera's+Hadoop+Demo+VM>

LZO Package for Red Hat-Based Distributions: <http://rpmforge.sw.be/redhat/el5/en/i386/rpmforge/RPMS/lzo-2.04-1.el5.rf.i386.rpm>

Twitter Hadoop-lzo Project: <http://github.com/kevinweil/hadoop-lzo>

RPM for LZO for Linux: <http://code.google.com/p/hadoop-gpl-packing>

Hadoop GPL Compression FAQ: <http://code.google.com/a/apache-extras.org/p/hadoop-gpl-compression/wiki/FAQ>

LZOP binaries for Windows and Linux: <http://www.lzop.org>

Hadoop at Twitter: <http://www.cloudera.com/blog/2009/11/hadoop-at-twitter-part-1-splittable-lzo-compression>

Twitter Elephant-Bird Project: <https://github.com/kevinweil/elephant-bird>

Introducing **Vagrant**

**Have you ever heard the following?
“Welcome to the team! Here’s a list of 15
applications to install, the instructions are in
the team room, somewhere. See you in a week!”
Or: “What do you mean it broke production,
it runs fine on my machine?” Or:
“Why is this working on her machine
and his machine, but not my machine?”**

JAY PALAT

D

development environments are becoming more complex, with more moving parts and tricky dependencies.

Virtualization has been a huge boon for the IT industry in saving costs, increasing flexibility and maintaining control over complex environments. Rather than focusing on virtualization on the delivery side, let's look at how you can provide that flexibility and control to developers to manage multiple development environments easily using Vagrant.

What Is Vagrant?

Vagrant is an open-source (MIT) tool for building and managing virtualized development environments developed by Mitchell Hashimoto and John Bender. Vagrant manages virtual machines hosted in Oracle VirtualBox, a full x86 virtualizer that is also open source (GPLv2).

A virtual machine is a software implementation of a computer, running a complete operating system stack on a virtualizer. It is a full implementation of a computer with a virtual disk, memory and CPU. The machine running the virtualizer is the Host system. The virtual machine running on the virtualizer is the Guest system. As far as the Guest operating system is concerned, it is running on real hardware. From the perspective of the Host, all of the Guest's resources are

used by the virtualizer program. A Box, or base image, is the prepackaged virtual machine that Vagrant will manage.

Installing Vagrant

Starting in version 1.0, Vagrant provides two installation methods: packaged installers for supported platforms or a universal install with Ruby Gems. This article covers installation using Gems. This method has three parts: 1) install VirtualBox, 2) install Ruby and 3) install Vagrant itself.

VirtualBox is available from the VirtualBox home page with builds for Windows, OS X, Linux and Solaris. Note that Oracle provides the Oracle VM VirtualBox Extension Pack on the Download site that provides additional features to the virtualizer. The Extension Pack has a separate license (Personal Use and Evaluation License) and is not needed to use Vagrant, but if the Box you are using was created using the Extension Pack, you will need to install the Extension Pack as well.

Ruby is a popular dynamically typed object-oriented scripting language. Ruby is available out of the box in OS X, and most Linux distributions also have a Ruby package available. For Windows users, the RubyInstaller Project provides an easy way to install the Ruby runtime.

Ruby libraries and applications are

available in packages called RubyGems or Gems. Ruby comes with a package management tool called gem. To install Vagrant, run the gem command:

```
> gem install vagrant
```

Vagrant is a command-line tool. Calling `vagrant` without additional arguments will provide the list of available arguments. I'll visit most of these commands within this article, but here's a quick overview:

- `init` — create the base configuration file.
- `up` — start a new instance of the virtual machine.
- `suspend` — suspend the running guest.
- `halt` — stop the running guest, similar to hitting the power button on a real machine.
- `resume` — restart the suspended guest.
- `reload` — reboot the guest.
- `status` — determine the status of vagrant for the current Vagrantfile.
- `provision` — run the

provisioning commands.

- `destroy` — remove the current instance of the guest, delete the virtual disk and associated files.
- `box` — the set of commands used to add, list, remove or repackag box files.
- `package` — used for the creation of new box files.
- `ssh` — ssh to a running guest.

The last thing you need to do in your installation is set up a base image. A Box, or base image, is the prepackaged virtual machine that Vagrant will manage. Use the `box` command to add the Box to your environment. The `vagrant box add` command takes two arguments, the name you use to refer to the Box and the location of the Box:

```
> vagrant box add lucid32 http://files.vagrantup.com/lucid32.box
```

This command adds a new Box to the system called "lucid32" from a remotely hosted site over HTTP. Vagrant also will allow you to install a Box from the local filesystem:

```
> vagrant box add rhel5.7 rhel5.7-20120120-1223.box
```

```
[vagrant] Downloading with Vagrant::Downloaders::File...
[vagrant] Copying box to temporary location...
[vagrant] Extracting box...
[vagrant] Verifying box...
[vagrant] Cleaning up downloaded box...
>
```

Now there are two Boxes installed:

```
>vagrant box list
lucid32
rhel5.7
```

It is important to note that you can reuse these base images. A Box can be the base for multiple projects without contamination. Changes in any one project will not change the other projects that share a Box. As you'll see, changes in the base can be shared to projects easily. One of the powerful concepts about using Vagrant is that the development environment is now totally disposable. You persist your critical work on the Host, while the Guest can be reloaded quickly and provisioned from scratch.

Starting Vagrant

Create a directory on the Host as your starting point. This directory is your working directory. Vagrant will share this directory between the Guest and the Host automatically. Developers can

edit files from their preferred Editors or IDEs without updating the Guest. Changes made in the Host or Guest are immediately visible to the user from either perspective:

```
> mkdir ProjectX
> cd ProjectX
```

To get started, you need a Vagrantfile. The Vagrantfile is similar to a Makefile, a set of instructions that tells Vagrant how to build the Guest. Vagrant uses Ruby syntax for configuration. The simplest possible Vagrantfile would be something like this:

```
Vagrant::Config.run do |config|
  config.vm.box = "lucid32"
end
```

This configuration tells Vagrant to use all the defaults and the Box called lucid32. There actually are four Vagrantfiles that are read: the local version in the current directory, a user version in `~/.vagrant.d/`, a Box version in the Box file and an initial config installed with the Gem. Vagrant reads these files starting with the Gem version and finishing with the current directory. In the case of conflicts, the most recent version wins, so the current directory overrides the `~/.vagrant.d`, which overrides the

Box version and so on. Users can create a new Vagrantfile simply by running:

```
> vagrant init
```

This creates a Vagrantfile in the current directory. The generated Vagrantfile has many of the common configuration parameters with comments on their use. The file tries to be self-documenting, but additional information is available from the Vagrant Web site. To run most Vagrant commands, you need to be in the same directory as the Vagrantfile.

Let's give it a try:

```
$ vagrant up  
[default] Importing base box 'lucid32'...  
[default] The guest additions on this VM do not match the install  
version of VirtualBox! This may cause things such as forwarded  
ports, shared folders, and more to not work properly. If any of  
those things fail on this machine, please update the guest  
additions and repackage the box.
```

```
Guest Additions Version: 4.1.0  
VirtualBox Version: 4.1.8  
[default] Matching MAC address for NAT networking...  
[default] Clearing any previously set forwarded ports...  
[default] Forwarding ports...  
[default] -- 22 => 2222 (adapter 1)  
[default] Creating shared folders metadata...  
[default] Clearing any previously set network interfaces...  
[default] Booting VM...  
[default] Waiting for VM to boot. This can take a few minutes.
```

```
[default] VM booted and ready for use!  
[default] Mounting shared folders...  
[default] -- v-root: /vagrant
```

Let's break this down. I'm running with VirtualBox 4.1.8 and a Guest at 4.1.0. In this case it works smoothly, but the warning is there to help troubleshoot issues should they appear. Next, it sets up the networking for the Guest:

```
[default] Matching MAC address for NAT networking...  
[default] Clearing any previously set forwarded ports...  
[default] Forwarding ports...  
[default] -- 22 => 2222 (adapter 1)
```

I used the default network setting of Network Address Translation with Port Forwarding. I am forwarding port 2222 on the Host to port 22 on the Guest. Vagrant starts the Guest in headless mode, meaning there is no GUI interface that pops up. For users who want to use the GUI version of the Guest, the option is available in the Vagrantfile to run within a window. Once the network is set up, Vagrant boots the virtual machine:

```
[default] VM booted and ready for use!  
[default] Mounting shared folders...  
[default] -- v-root: /vagrant
```

After the Guest has started, Vagrant will add the shared folder. Vagrant uses

the VirtualBox extensions to mount the current folder (ProjectX in this case) as /vagrant. Users can copy and manipulate files from the Host OS in the ProjectX directory, and all the files and changes will be visible in the Guest. If the shared folder isn't performing well because you have a large number of files, Vagrant does support using NFS. However, it does require that NFS is supported by both the Guest *and* Host systems. At this time, NFS is not supported on Windows Hosts.

To access the Guest, Vagrant Boxes have a default user with username:vagrant, password:vagrant and a default shared insecure ssh-key. Vagrant users usually have sudo rights if they need to make changes to the Guest OS. You can ssh to the Guest by using `ssh vagrant@localhost 2222` on the Host or with the vagrant shortcut:

```
>vagrant ssh
```

Although you can automatically reach your Guest using `vagrant ssh`, sometimes it is useful to use `scp` or `git`, which refers to your SSH setup. To make life easier, you can update the entry in your `~/.ssh/config` with information specific to the Guest (Windows users can achieve similar results by making similar updates to their PuTTY configuration):

```
>vagrant ssh-config
Host default
  HostName 127.0.0.1
  User vagrant
  Port 2222
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  PasswordAuthentication no
  IdentityFile /Users/jay/.vagrant.d/insecure_private_key
  IdentitiesOnly yes
```

By plugging this information in to `~/.ssh/config`, you can access your Guest with `scp` to move a file `my_file` from /vagrant to the Host desktop:

```
> scp default:/vagrant/my_file.txt ~/Desktop/my_file.txt
```

What if you want to run a Web server on your Guest that's accessible by the Host? Let's add one. Edit your Vagrant file, adding the following line:

```
config.vm.forward_port 80, 8080
```

The `forward_port` parameter allows the Host to forward ports from the Host to the Guest. In this case, you're forwarding port 8080 on the Host to 80 on the Guest. By default, Vagrant uses Network Address Translation networking, meaning the Guest can access the network through the Host. Machines on the Host network cannot reach the Guest directly unless the

ports are forwarded. Vagrant also provides Host-only networking, where a Guest can access only the host or other Guests on the Host, and bridge networking where the Guest will be on the same network as the Host. Once you update the Vagrantfile, restart the system:

```
>vagrant reload
```

Make sure that when you open port forwarding in Vagrant you also are opening the appropriate firewall permissions on the Guest! If you try to make a connection to a Guest via port forwarding with a firewall turned on, the initial handshake will work with the host port (8080 in the example here) but fail if port 80 is blocked by the Guest firewall.

Finishing Vagrant

Once you're done with your work, you have a few choices. If you're just done for the day, or trying to get a few resources back for the Host, you can simply suspend Vagrant:

```
>vagrant suspend
```

and resume:

```
>vagrant resume
```

when you want to restart. All state is

maintained, and nothing is lost. Once an environment is completely finished, or if you want to start over from a clean slate, you can run `vagrant destroy`. This will remove the Guest and any changes to content outside of the shared folders (`/vagrant`). Shared folders reside on the Host and will survive the destruction of the Guest.

Provisioning a Guest

Creating and maintaining virtual images can be a delicate balance in terms of configuration. Over-configure the image, and it gets stale faster as applications are replaced with patches. Under-configure the image with a simple base, and each time a new instance is created, the user must spend the time configuring the Box to the exacting specifications of the project. With Vagrant, there is a good middle ground. A base image can be created and minimally updated to manage OS-level patches and libraries, while applications can be provisioned using tools like Chef, Puppet or shell scripts.

Many systems administrators are already familiar with Chef and Puppet, two configuration management tools that allow for consistent creation and maintenance of a system. By plugging Chef and Puppet into Vagrant, users can take immediate advantage of the scripts that already have been created by system administrators to create up-to-date

images for development.

Chef and Puppet are both supported to work in solo/standalone mode and server mode. In Chef Solo or Puppet standalone, the configuration for the images are all self-contained. In server mode, Chef or Puppet Server users can leverage the existing Chef or Puppet infrastructure already established in their companies.

This article doesn't go into great detail about Chef, but let's look at a simple example of how using Chef can be helpful. Vagrant can run without additional servers in Chef-solo mode. To set up Chef, you first need a Cookbook. Cookbooks are a collection of Recipes. A Recipe is a description of how you would like the system configured. Recipes also are written in Ruby. For this example, I've downloaded Cookbooks available from the Opscode community site.

In this example, let's install Apache2 and MySQL on our Guest. Here's the setup. Create a directory called "cookbooks" under the ProjectX directory. Download the Cookbooks for Apache2 and MySQL from <http://community.opscode.com> (see Resources). In addition, you'll need two dependencies: the apt recipe to get the latest updates and openssl, which is required for the MySQL recipe.

Extract these files and save them to

the cookbooks directory:

```
> ls -l cookbooks/
total 0
drwxr-xr-x@ 10 jay  staff  340 Feb 16 18:49 apache2
drwxr-xr-x@  9 jay  staff  306 Feb 14 12:00 apt
drwxr-xr-x@  9 jay  staff  306 Feb 16 18:23 mysql
drwxr-xr-x@  7 jay  staff  238 Jun  3  2011 openssl
```

In my Vagrantfile, I have added the following lines:

```
config.vm.provision :chef_solo do |chef|
  chef.cookbooks_path = "cookbooks"
  chef.add_recipe("apt")
  chef.add_recipe("openssl")
  chef.add_recipe("mysql::server")
  chef.add_recipe("apache2")
  # You may also specify custom JSON attributes:
  chef.json = {
    :mysql => {
      :server_root_password => "MYSQL PASSWORD"
    }
  }
end
```

When Vagrant runs the provisioning step (included in `vagrant up` but also available on a running Guest by using `vagrant provision`), it will install apt, openssl, mysql and apache2. Recipes can take arguments as well. In this example, I've included a parameter

for the mysql recipe that sets the server root password. The parameters available for a given recipe are included in the documentation. The parameters are passed as JSON attributes to Vagrant, which will pass them along to Chef.

Managing this with Vagrant allows you to enforce change controls around your environments. Vagrantfiles and Cookbooks can be checked into source control and managed just like code resources. Branching and tagging environment configurations can be used to track changes as projects develop or requirements and packages are updated.

Creating a New Image

The traditional method is to use VirtualBox to create a new virtual machine image using the standard operating system installer. Because operating system steps vary, I'm not going to go into the details here, but here are some key steps:

- When choosing the Virtual Disk storage details, choose "Dynamically Allocated"—you want your disk to start small but be able to increase as needed.
- If you want to use Chef or Puppet with your application, you'll need to install the Chef or Puppet applications

when creating the base image.

- Install the VirtualBox Guest Additions. This is necessary to support shared folders!

Let's set up the vagrant user. Create a user with the user name vagrant. Add this user to an admin group. Update the sudoers file to make sure that users in the admin group do not need a password to access sudo. For a group called "admin", it should look like this:

```
%admin ALL=NOPASSWD:ALL
```

Also make sure that the `Defaults requiretty` is commented out. You also want to make sure you can ssh into the Vagrant account. By default, Vagrant will try to use the insecure SSH key with `vagrant ssh`. To add the key to your Vagrant account, do the following as the vagrant user:

```
> mkdir .ssh  
> chmod 755 .ssh  
> curl -L http://github.com/mitchellh/vagrant/raw/master/  
→keys/vagrant.pub  
> > .ssh/authorized_keys  
> chmod 644 .ssh/authorized_keys
```

Once the image is complete, you'll need to package the image into a Box.

To create a set of default configurations for the Box, you can create a new Vagrantfile. Use `vagrant package`, which takes two arguments `--base` (the name of the image in VirtualBox that you created) and `--include` (which takes the files you wish to include in your Box):

```
> vagrant package --base RHEL-5.7-64 --include Vagrantfile
```

In this example, you created a package with the base image called RHEL-5.7-64 and included your custom Vagrantfile.

Summary

Vagrant is a powerful tool for creating and managing flexible development environments. In this article, I covered the basic use and creation of Vagrant images.■

Jay Palat is the Lead for Mobility Applications at the University of Pittsburgh Technology Development Center (TDC). His experience ranges from fast startups like ModCloth to developing the IBM Support Site. He has an active interest in mobile, cloud and devops technology and techniques. He has a Masters of Information Technology from Carnegie Mellon University and BS in Computer Science from University of Pittsburgh.

Resources

Vagrant's Home on the Web: <http://vagrantup.com>

VirtualBox: <https://www.virtualbox.org/wiki/Downloads>

Ruby: <http://www.ruby-lang.org/en/downloads>

RubyInstaller (for Windows): <http://rubyinstaller.org>

Opscode Community Site: <http://community.opscode.com>

Chef Cookbooks:

- mysql: <http://community.opscode.com/cookbooks/mysql>
- apache2: <http://community.opscode.com/cookbooks/apache2>
- apt: <http://community.opscode.com/cookbooks/apt>
- openssl: <http://community.opscode.com/cookbooks/openssl>

Vagrantbox.es (a Web Site That Lists Boxes): <http://vagrantbox.es>

Writing Samba VFS Modules

Although Samba does a very good job of serving files for Windows (and Linux) clients, sometimes you want it to do more. This article introduces the Samba VFS Layer and shows how to create a simple VFS module to extend Samba's functionality.

RICHARD SHARPE

I am sure that many of you are aware of the Linux Virtual File System (VFS) layer, which allows Linux to support many different filesystems, such as EXT3, EXT4, XFS, btrfs and so on. However, I suspect you are less aware that Samba, the Windows file and print server for UNIX, also has a VFS. Just like the Linux VFS, the Samba VFS allows you to extend Samba's functionality to encompass different filesystems and perform some neat tricks. On the other hand, the Samba VFS lives in userspace, which makes it easier to develop and debug Samba VFS modules.

In this article, I introduce you to writing a Samba VFS module through the use of a simple example. The example is that of auditing access to a specific file. Although this example is a little simplistic for the sake of this article, it could be extended in many ways to be more powerful.

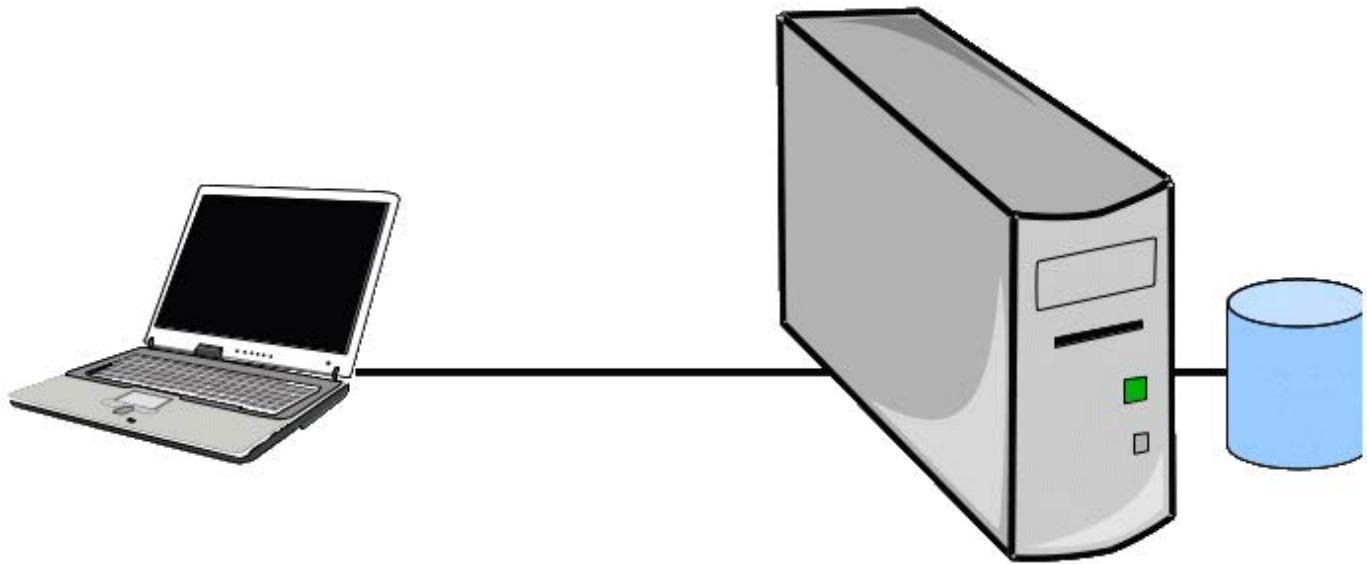


Figure 1. A Windows Client Accessing Storage on a Linux Server

A Quick Introduction to the Samba VFS Layer

Figure 1 shows how Samba normally is used. Users on Windows clients access files on a Samba server via the network.

Figure 1 depicts a Windows client (although it could be running anything) connected to a server via the network and accessing files on the storage connected to the server. You would configure Samba to give the client access to parts of the filesystem of the Linux system on which it is running. You do this with shares.

Samba is told how to provide access to files by setting configuration information in the `smb.conf` file, which might be `/etc/samba/smb.conf` (if you have installed Samba from RPMs) or which might be `/usr/local/samba/etc/smb.conf` (if you have built Samba from source and

installed it on your system).

The following shows a very simple `smb.conf` file. You should consult some of the documents listed in the Resources section for more information on each of the parameters shown in this example:

[global]

```
workgroup = workgroup
server string = %h server
security = user
```

[data]

```
path = /path/to/data
read only = no
vfs objects = vfs_demo
vfs_demo:audit path = aaa
```

Samba allows you to implement VFS modules in two different ways:

1. As built-in and statically linked modules.
2. As shared libraries.

You tell Samba about VFS modules for each share, and you can pass module-specific parameters to a module. An example of specifying a VFS module is shown in the smb.conf fragment above. The module for the [data] share is called vfs_demo, and it has one parameter called audit path.

If you do not specify any modules, Samba does not load any (except for the default, discussed below).

The `vfs_demo` Module

In this article, I explain how to develop a simple Samba VFS module that allows you to specify auditing of accesses to files. The auditing that is performed is very simple and is not the same as security auditing that Windows performs using Security ACLs (SACLs), but it serves to illustrate some of the things you need to do in developing a Samba VFS. In particular, this module: 1) will allow you to specify the path prefix for files to be audited, and 2) will log all accesses of such files to /var/log/messages using syslog.

It has some deficiencies as well, which you might consider dealing with if you want to improve the demo module.

Various aspects of the code for this module, as well as how to build it and how to debug it, are discussed below after a more thorough introduction to the Samba VFS Layer.

More Detail on the Samba VFS Layer

Samba has a very flexible VFS system that allows you to combine functionality from several modules to customize the behavior of any share. Figure 2 shows how VFS modules relate to the main Samba code.

As Figure 2 suggests, a number of components are involved in the processing of requests from Windows clients:

1. The main Samba code, which receives all incoming requests and sends all outgoing responses, interprets the smb.conf file and so forth.
2. The VFS Layer, which provides infrastructure for the VFS modules themselves.
3. A default VFS module, `vfs_default.c`, which provides default actions in case no modules have been specified for a share.
4. The system layer, which the default VFS module calls to provide standard Samba actions for all of the VFS

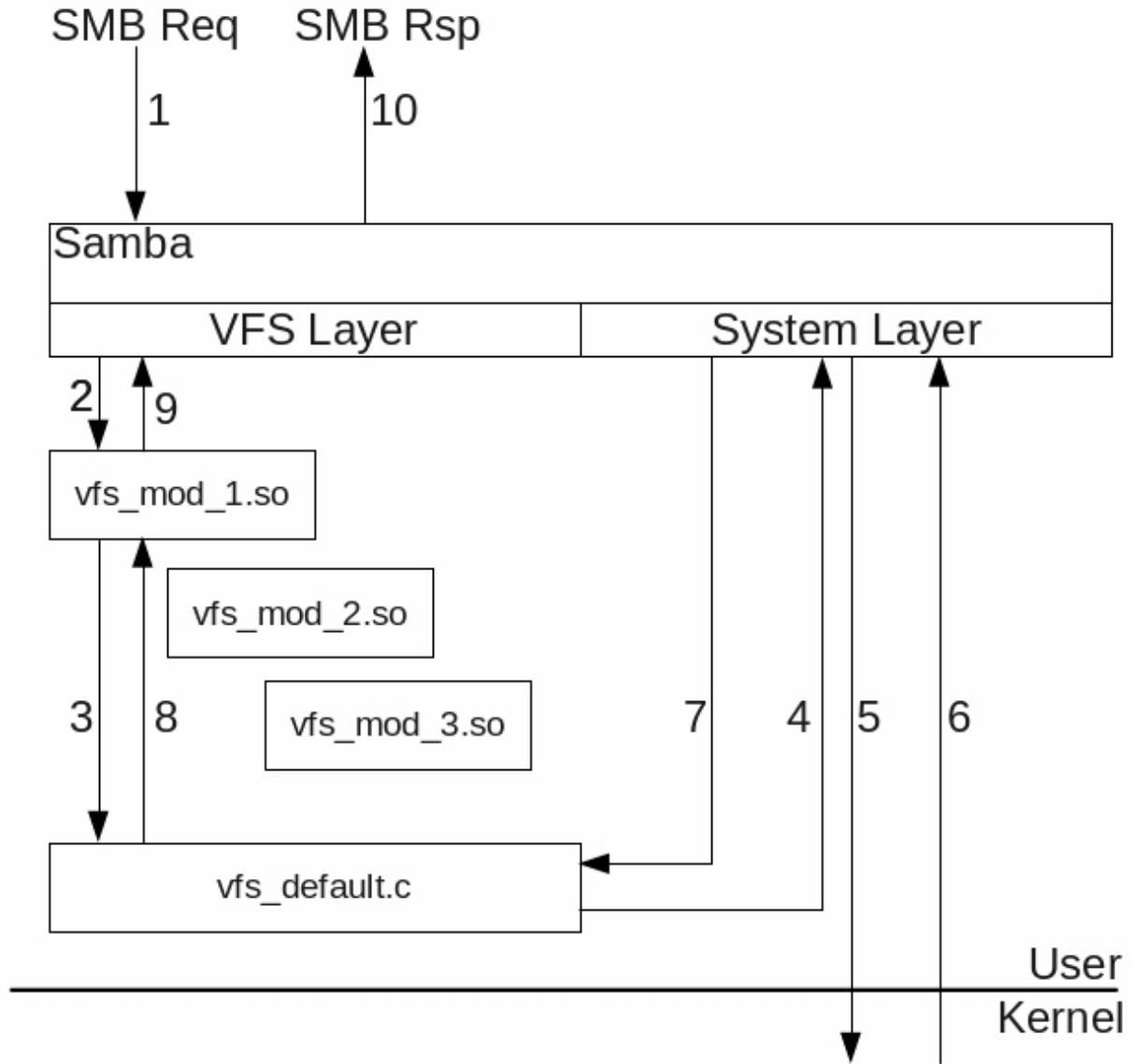


Figure 2. The Relationship between VFS Modules and the Samba Code

routines. They mostly translate into UNIX/Linux system calls.

One of the most important aspects of the Samba VFS Layer is that VFS modules are stackable and, depending on the way they are coded, can:

1. Completely replace existing Samba functionality.
2. Augment existing Samba functionality.
3. Do both, perhaps based on filename or VFS function.

Because they are stackable, you can write a VFS module to augment or replace a small subset of the 100 or so VFS functions that Samba uses, which makes your life as a VFS module writer much easier.

Because they are stackable, you can write a VFS module to augment or replace a small subset of the 100 or so VFS functions that Samba uses, which makes your life as a VFS module writer much easier.

First, however, when a Windows client connects to a share, Samba finds the required VFS module, and if it is a shared library, loads that shared library. It then calls the specified module's connect routine to allow the module to perform whatever initialization it has to.

Then, during normal operation, the following sequence of steps will be performed for requests from the Windows client:

1. An SMB request is received from the client by Samba and undergoes initial processing within Samba itself. At some point, Samba then calls a VFS function to complete the processing of this request.
2. The VFS Layer then calls the requested VFS function in the

first VFS module, `vfs_mod_1`, in the stack that implements that VFS function. The VFS function in this module could perform some initialization at that point.

3. That VFS function then calls the next module in the stack that implements the requested VFS function, which in this case is `vfs_default.c`.
4. After performing some processing, if needed, `vfs_default.c` calls the relevant System Layer function in Samba.
5. Again, after performing relevant setup or other processing, the called System Layer function issues a system call.
6. After performing the system call, the kernel returns to the System Layer function that called it. Here, further processing can be performed.
7. The System Layer function first called

then returns to `vfs_default.c` where further processing can be performed.

8. `vfs_default.c` returns to the first VFS module in the stack, which again can perform additional processing.
9. The first VFS module then returns to Samba.
10. Samba prepares a response with the data returned and sends it to the Windows client.

Finally, when a Windows client disconnects from a share, Samba calls the disconnect routine associated with the first VFS module in the stack that has defined one.

There are some 100 or more VFS functions in the Samba VFS layer divided into the following classes:

1. Disk/share functions, which relate to connecting and disconnecting, quota handling and so forth.
2. Directory functions for performing operations on directories, like opening them, listing them and so on.
3. File functions, which constitute the largest group of operations. These include things like reading and writing files, creating them, opening

them and so on.

4. NT ACL functions for getting and setting NT Security Descriptors on files and directories.
5. POSIX ACL functions for getting and setting POSIX ACLs on files and directories.
6. Extended Attribute functions for getting and setting XATTRs on files and directories.
7. AIO functions, for performing Async IO operations on files.
8. Off-line functions for handling off-line operations on files.

However, in general, you need to worry only about the specific set of functions you need to implement, and in the case of this `vfs_demo` module, you need to be concerned only with a small set of routines.

Registering the `vfs_demo` Module with Samba

As you might have guessed, each VFS module registers itself with Samba and specifies the VFS functions that it implements. To do this, you need to create a C function in the format shown in Listing 1.

Here you have told Samba that you are registering a module called “vfs_demo”, the version of the VFS interface you

conform to and that you implement the eight functions specified. The first two are a connect routine and a

Listing 1. Registering Your Module with Samba

```
#include "includes.h"
#include "smbd/smbd.h"

#undef DBGC_CLASS
#define DBGC_CLASS DBGC_VFS

#define DEMO_MODULE_NAME "vfs_demo"

static struct vfs_fn_pointers vfs_demo_fns = {
    .connect_fn = demo_connect,
    .disconnect_fn = demo_disconnect,

    .opendir_fn = demo_opendir,
    .fdopendir_fn = demo_fdopendir,
    .mkdir_fn = demo_mkdir,
    .rmdir_fn = demo_rmdir,

    .open_fn = demo_open,
    .create_file_fn = demo_create_file,
};

NTSTATUS vfs_demo_init(void);
NTSTATUS vfs_demo_init(void)
{
    return smb_register_vfs(SMB_VFS_INTERFACE_VERSION,
                           DEMO_MODULE_NAME,
                           &vfs_demo_fns);
}
```

disconnect routine. The remaining ones relate to opening files and directories, creating files and directories, and removing directories.

Note: for reasons pointed out in the Samba VFS documentation (see Resources), you should call your registration routine `<module_name>_init`. This allows your module to be built in the Samba source tree or outside it and as a shared module or a static module. In this case, I called it `vfs_demo_init` and the Makefile that is generated below will take care of ensuring that the correct name is used in a shared module (as long as the patch mentioned below is applied).

Building Your VFS Module

Now that you have some code, it is time to think about building your module. Although you can add your module to the Samba source code, doing that is more complicated than the approach suggested here, which builds your VFS module outside the Samba source tree. However, you will need to have the Samba source code handy.

First, download the latest Samba source code (Samba 3.6.4 at the time of this writing) using:

```
wget http://ftp.samba.org/pub/samba/old-versions/samba-3.6.4.tar.gz
```

Next, unpack the source and build it

(for those who have not done this before, here are the steps):

```
tar xvf samba-3.6.4.tar.gz  
cd samba-3.6.4/source3  
.configure.developer  
make
```

Then, as root:

```
make install
```

Note: if you do not want to replace the Samba version on your development system, you will have to download the source packages for your Linux distro (source RPMs for those distros based on RPMs) and build that RPM. Here I assume you are happy to build Samba from source and install it on your system. This might mean that you have to erase the distro-supplied Samba packages to avoid confusion with them.

In addition, you need to tell the system how to find the Samba shared libraries that are needed. You can do this as root with:

```
echo /usr/local/samba/lib > /etc/ld.so.conf.d/samba-local-build.conf  
ldconfig
```

You can use `ldconfig -v` to verify that it found your shared libraries.

Now that you have the Samba source and have built Samba, you can create a

directory and add the code for the demo module to it:

```
mkdir demo_vfs  
cd demo_vfs  
# Use vim to add the code
```

Next, you need to copy several files from the examples/VFS directory of the Samba source:

```
cp ./samba-3.6.4/examples/VFS/config* .  
cp ./samba-3.6.4/examples/VFS/autogen.sh .  
cp ./samba-3.6.4/examples/VFS/install-sh .  
cp ./samba-3.6.4/examples/VFS/configure.in .  
cp ./samba-3.6.4/examples/VFS/Makefile.in .
```

After that, you should apply the following patch to Makefile.in to allow your code to be built as a shared module:

```
--- Makefile.in      2012-04-22 17:30:45.631698392 -0700  
+++ Makefile.in.new 2012-04-22 17:30:20.619140189 -0700  
@@ -36,7 +36,7 @@  
  
%.$(OBJEXT): %.c  
    @echo "Compiling $"  
-   @$(CC) $(FLAGS) -c $<  
+   @$(CC) $(FLAGS) -c $< -D$*_init=init_samba_module  
  
install: default
```

(At the time of this writing, this patch has been applied to the Samba master

branch, but it has not yet made it into the 3.6.x branch. The patch makes it easier for your module to be moved into the Samba source tree if it proves useful enough and makes documenting modules easier. You simply could edit Makefile.in and append `-D$*_init=init_samba_module` to the line shown rather than creating a patch file.)

These files allow you to build your module; however, you must perform a few actions similar to those performed above when building the Samba source:

```
./autogen.sh  
.configure --with-samba-source=../samba-3.6.4/source3
```

After that, you should simply be able to run `make` to build your source. However, `make` will fail because there are no definitions for the eight functions that you are registering.

Note: if your OS does not use GNUmake, you will have problems with the above instructions because the Makefile created in the above steps uses GNUmake's syntax, and it will not work with other versions of make. The simplest way to fix this is to install GNUmake.

Creating the Needed VFS Functions

The connect and disconnect functions must perform some unique actions in this example, because the remaining six

Listing 2. demo_connect Function

```
static int demo_connect(vfs_handle_struct *handle,
                       const char *service,
                       const char *user)
{
    int res = 0;
    struct demo_struct *ctx = NULL;

    /*
     * Allow the next module to handle connections first
     * If we get an error, don't do any of our initialization.
     */
    res = SMB_VFS_NEXT_CONNECT(handle, service, user);
    if (res) {
        return res;
    }

    /*
     * Get some memory for the dir we are interested in
     * watching and our other context info.
     */
    ctx = talloc_zero(handle, struct demo_struct);
    if (!ctx) {
        DEBUG(0, ("Unable to allocate memory for our context,
                  can't proceed!\n"));
        errno = ENOMEM;
        return -1;
    }

    ctx->audit_path = lp_parm_const_string(SNUM(handle->conn),
                                            DEMO_MODULE_NAME,
                                            "audit path",
                                            NULL);

    DEBUG(10, ("audit path is \"%s\"", ctx->audit_path));

    res = sem_init(&ctx->send_sem, 0, LOG_QUEUE_SIZE);
    if (res) {
        DEBUG(1, ("Unable to initialize send sem: %s\n",
                  strerror(errno)));
        goto error_no_thread;
    }

    res = sem_init(&ctx->recv_sem, 0, 0);
    if (res) {
        DEBUG(1, ("Unable to initialize recv sem: %s\n",
                  strerror(errno)));
        goto error_no_thread;
    }

    res = pthread_create(&ctx->log_thread,
                         NULL,
                         logging_thread,
                         ctx);
    if (res) {
        DEBUG(1, ("Unable to create our background thread: %s\n",
                  strerror(errno)));
        goto error_no_thread;
    }

    SMB_VFS_HANDLE_SET_DATA(handle, ctx, NULL,
                           struct demo_struct, goto error);

    return res;
error:

error_no_thread:
    talloc_free(ctx);
    return res;
}
```

functions all rely on a background thread to send auditing messages to syslog so they can be added to /var/log/messages. Listing 2 is the demo_connect function.

The key points to note here are:

1. You should use talloc for allocating memory for your module, and you should choose an appropriate context. When allocating memory that should last for the duration of the connection to the share, use the VFS handle for a context.
2. You should use `lp_parm_const_string` to parse module parameters.
3. You should use the `SMB_VFS_HANDLE_SET_DATA` macro to set up the context that all your other routines need when they are called.
4. You should call the next VFS routine in the stack, in this case `SMB_VFS_NEXT_CONNECT`, to give it a chance to do its module initialization work as well. (However, see the Samba VFS document mentioned below for cases where you do not want to call the Next module.)

While in general you can call the next VFS routine before, after or during your processing, in this case, it is called first to make cleanup easier in the connect

routine, because you are creating a thread to perform deferred work.

The next function to look at is the `demo_disconnect` function (Listing 3).

In this case, you:

1. Call the next VFS function, `SMB_VFS_NEXT_DISCONNECT`.
2. Use a function called `create_cmd` to tell your background thread.
3. Use `pthread_join` to wait for the background thread to exit.
4. Free up the context you created.

At this point, it is worth mentioning that all of the VFS functions have names like those seen above (such as `SMB_VFS_CONNECT` and `SMB_VFS_NEXT_CONNECT`), and their meanings are:

1. Functions like `SMB_VFS_CONNECT`, `SMB_VFS_GET_NT_ACL` and so on call that VFS function from the top of the stack of VFS modules. Within a specific VFS function, you would not call that function recursively; however, you might call other VFS functions to perform useful actions. For example, the `vfs_acl_xattr.c` module (in `samba-3.6.4/source3/modules`) calls `SMB_VFS_GETXATTR` to get extended attributes on files.

Listing 3. demo_disconnect Function

```
static void demo_disconnect(vfs_handle_struct *handle)
{
    int res = 0, *thread_res = NULL;
    struct demo_struct *ctx;
    struct cmd_struct *cmd;

    /* Let the next module do any cleanup it needs to */
    SMB_VFS_NEXT_DISCONNECT(handle);

    SMB_VFS_HANDLE_GET_DATA(handle,
                           ctx,
                           struct demo_struct,
                           goto ctx_error);

    /*
     * Tell the background thread to exit
     */
    cmd = create_cmd(ctx, LOG_EXIT, NULL);
    if (!cmd || !send_cmd(cmd)) {
        return; /* Not much more to do here ... kill the thread? */
    }

    res = pthread_join(ctx->log_thread, (void **)&thread_res);
    if (res || *thread_res) {
        DEBUG(10, ("Error cleaning up thread: res: %s, "
                  "thread_res: %s\n",
                  strerror(errno), strerror(*thread_res)));
        return;
    }

    /*
     * This is not absolutely needed since that structure used
     * the handle as a talloc context ...
     */
    talloc_free(ctx);

    return;
ctx_error:
    DEBUG(10, ("Error getting context for connection!\n"));
    return;
}
```

2. Functions like `SMB_VFS_NEXT_CONNECT`, `SMB_VFS_GET_NT_ACL` and so on call the next function of the same type in the stack of modules. You usually would call the next function for your VFS function

somewhere in your VFS function unless you have completely handled everything or an error has occurred and there is no longer any need to call the next function.

Listing 4. demo_opendir Function

```
static DIR *demo_opendir(vfs_handle_struct *handle,
                         const char *fname,
                         const char *mask,
                         uint32 attr)
{
    DIR *res;
    struct demo_struct *ctx;

    SMB_VFS_HANDLE_GET_DATA(handle,
                            ctx,
                            struct demo_struct,
                            return NULL);

    if (ctx->audit_path && strstr(fname, ctx->audit_path)) {
        struct cmd_struct *cmd;
        DEBUG(10, ("Found %s in the path %s\n",
                   ctx->audit_path, fname));

        cmd = create_cmd(ctx, LOG_LOG, fname);
        if (!cmd || !send_cmd(cmd)) {
            DEBUG(1, ("Error logging. Continuing!\n"));
        }
    }

    /* Allow the next module to handle the OPENDIR as we are done */
    res = SMB_VFS_NEXT_OPENDIR(handle, fname, mask, attr);
    return res;
}
```

Finally, Listing 4 shows one of the actual functions for performing auditing when a file has been accessed.

Testing and Debugging

Once you have all the functions implemented (either by entering them from scratch or downloading them from GitHub or by taking an existing module and modifying it), you can build the VFS module and install it.

Building the module is usually as simple as running `make`. If any errors are reported, fix the errors and re-run `make` until the build completes successfully.

The output from the build process you are using here is a shared module called `vfs_demo.so`. This file needs to be moved to the directory where Samba looks for shared modules. This is usually `/usr/local/samba/lib/vfs`. You will need to copy `vfs_demo.so` to that location as root:

```
sudo cp vfs_demo.so /usr/local/samba/lib/vfs
```

Once you have done that, you can test whether it works. First, you need to ensure that you have a share defined that uses the newly created VFS module. The following `smb.conf` file defines such a share and includes some additional parameters to help with debugging:

[global]

```
workgroup = workgroup
```

```
server string = %h server
security = user
log level = 10
log file = /var/log/samba/%m.log
panic action = sleep 99999
[data]
path = /path/to/data
read only = no
vfs objects = vfs_demo
vfs_demo:audit path = aaa
```

The interesting things to note here are:

1. The log level has been set to 10, giving you lots of information about what is going on and going wrong with your new VFS module.
2. The log file has been set to `/var/log/samba/%m.log`, which will create a separate log file for each client that connects. This will make it easier for you to see errors during testing.
3. Finally, you have specified a panic action in case your module causes Samba to crash (for example, by seg faulting). The panic action causes the `smbd` that crashed to sleep for a long time, giving you time to attach with `gdb` to see why you crashed (although oftentimes you

can tell why you crashed by looking at the traceback in the log file).

Once you have created that smb.conf in /usr/local/samba/etc/smb.conf and started Samba as root with:

```
/usr/local/samba/sbin/smbd
```

you can check to see if it is running with:

```
ps -ax | grep smbd
```

There should be two instances of smbd running. If they are not running, check for problems in /var/log/samba/smbd.log (although if the problems occur early enough, you will have to start Samba with /usr/local/samba/sbin/smbd -i -F to determine why it failed before it could start logging information).

Once Samba is running happily, you can test your VFS module with smbclient:

```
smbclient //localhost/data -Usome-user%some-password
```

Of course, you will have to ensure that the user you are testing with exists and that the user's password is set. You can do this with:

```
smbpasswd -a some-user
```

Then you can set the user's password with:

```
smbpasswd some-user
```

where you will be prompted to enter the password. Once you have set the password for your user, you can try smbclient as shown above.

If all is working correctly, you should expect to see something like the following:

```
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.6.4...]
smb: \>
```

Then you can issue commands to test your VFS module's functions. The sort of smbclient commands to try include things like put /path/to/some-file aaa (where the target file matches the audit path specified above). If your module is working correctly, you would see the following in /var/log/messages:

```
Apr 29 13:02:27 localhost Samba Audit: File aaa accessed
Apr 29 13:02:27 localhost Samba Audit: File aaa accessed
```

As you can see, one deficiency of the module currently is that there are multiple messages logged for some accesses.

Obtaining the Full Source Code

You can obtain the full source code for this demo module from GitHub via:

```
git clone git@github.com:RichardSharpe/demo-samba-vfs.git
```



9th Annual

HIGH PERFORMANCE COMPUTING FOR WALL STREET Show and Conference

SEPTEMBER 19, 2012 (WEDS)

ROOSEVELT HOTEL, NYC

Madison Ave and 45th St, next to Grand Central Station

Plan now to attend, to exhibit and to sponsor.

Big Data, Cloud, Linux, Low Latency, Networks, Data Centers, Cost Savings. Wall Street markets will assemble at the 2012 HPC Sept. 19 show to see these new systems live on the show floor.

High Performance Computing, Big Data, Cloud, Linux, Low Latency, Data Centers, Networking Systems, Virtualization, Optimization, Linux, Grid, Blade, Cluster – the largest meeting of High Performance Computing in New York in 2012.

This **HPC networking opportunity** will assemble 800 Wall Street IT professionals at one time and one place in New York in September 2012.

This **HPC conference is focused on Speed, Low Latency, Networks, Data Centers, lower computer costs.**

This show will cover High Performance Computing, High Frequency Trading, Low Latency, Networks and Switch Solutions, Data Centers, Virtualization, Grid, Blade, Cluster, overcoming Legacy systems.

Our Show is an efficient one-day showcase and networking opportunity.

Leading companies will be showing their newest products on-the-show floor live.

Register in advance for the full conference program which includes general sessions, drill down sessions, an industry luncheon, coffee breaks, exclusive viewing times in the exhibits, and more. Save \$100. \$295 in advance. \$395 on site.

Don't have time for the full Conference? Attend the free Show.
Register in advance at: www.flaggmgmt.com/hpc

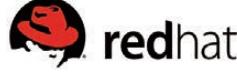


Wall Street IT speakers and Gold Sponsors will lead drill-down sessions in the Grand Ballroom program.



| |
|---------------------------------------|
| Show Hours: Weds, Sept 19 8:00 - 4:00 |
| Conference Hours: 8:30 - 4:50 |

Sponsors



ARISTA



SYBASE
An SAP Company



DataDirect
NETWORKS

NIMBUS DATA

FORSYTHE
Realize the business value of IT.™

SUPERMICRO



AMD

C4I
COALITION 4 INNOVATION

Myricom

Syncfusion

LINUX
MAGAZINE

LINUX
JOURNAL

TRADERS
ONLINE

Integration
developer news

Information
management
How Your Business Works

MONEY
management executive

FINANCE
ON WINDOWS

HPC
WIRE

HPC
In the Cloud

SECURITIES
TECHNOLOGY
MONITOR

WINDOWS
IN FINANCIAL SERVICES
The Resource for Microsoft in the Financial Enterprise

WSTA
Wall Street Technology Association

Show & Conference: Flagg Management Inc
353 Lexington Avenue, New York 10016
(212) 286 0333 fax: (212) 286 0086
flaggmgmt@msn.com

Visit: www.flaggmgmt.com/hpc

In the directory retrieved, you will find source code for the Samba master branch as well as the Samba 3.6 branch. Most of what has been discussed here relates to the Samba 3.6 branch. You also will find some README files.

The code in that repository builds against the Samba master branch and against Samba 3.6.x.

Final Thoughts

One of the issues you should be aware of is that the Samba VFS has changed over time; however, it is allowed to change only when a new major version is released. For example, there are differences between the Samba 3.5.x VFS, the Samba 3.6.x VFS and the VFS in the Samba master branch.

Indeed, the Samba master branch's VFS interface has been changed to make the VFS routine names more consistent as well as introducing at

least two new VFS functions:

1. An FSCTL function so that VFS writers now can write FSCTL handling functions.
2. A FILE AUDIT function so that VFS writers can handle file access auditing using the Windows approach.

However, you now should be able to understand the Samba VFS Layer well enough to understand what existing modules are doing and even write your own VFS modules when the need arises. ■

Richard Sharpe is a software engineer in Silicon Valley where he works on storage systems. He has been an open-source contributor for a long time, and he has contributed code to Ethereal/Wireshark, Samba and SCST. Richard recently has gotten back into contributing to Samba and has made a number of improvements to the Samba VFS Layer, some to ease building VFS modules out of the Samba source tree and some to improve functionality.

Resources

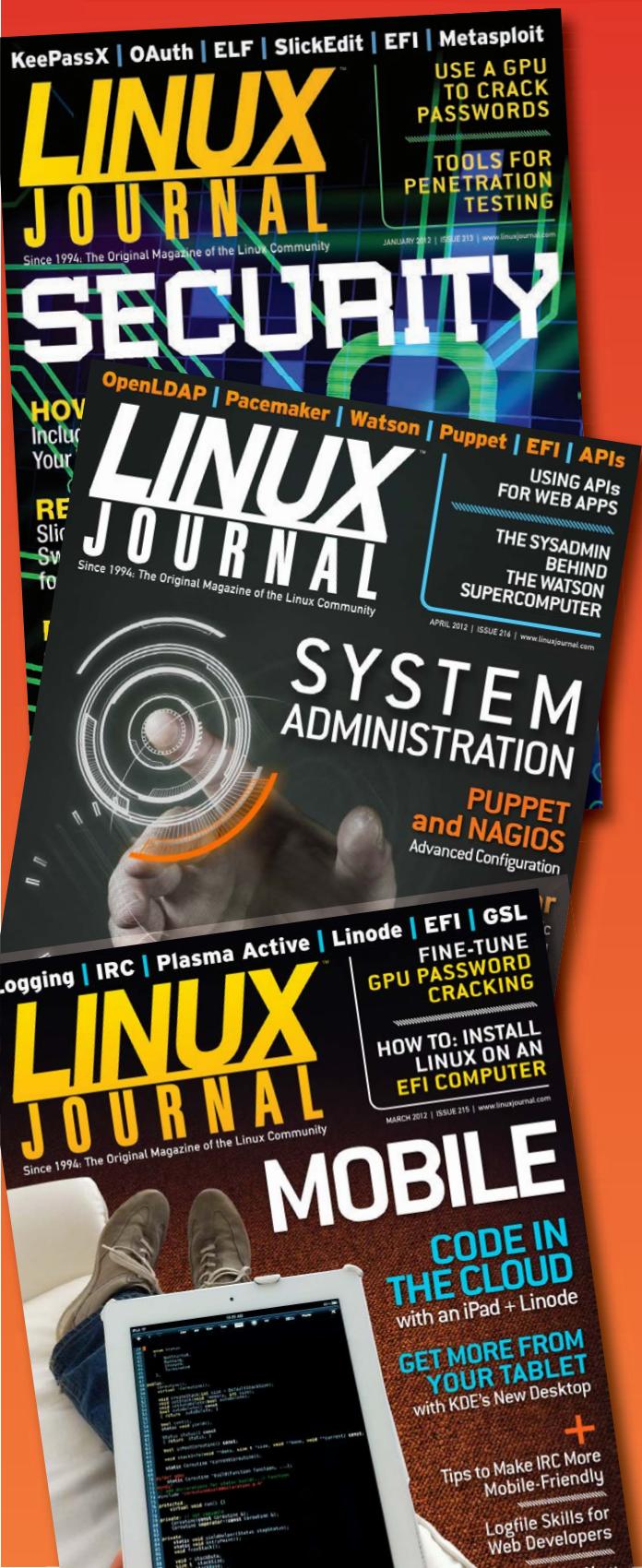
A Detailed Introduction to the Samba VFS: <http://samba.org/~sharpe/The-Samba-VFS.pdf>

“Introducing Samba” by John Blair: <http://www.linuxjournal.com/article/2716>

Samba-3 by Example: Practical Exercises to Deployment by John H. Terpstra: <http://www.informit.com/title/0131472216>

The Official Samba-3 HOWTO and Reference Guide, edited by John H. Terpstra and Jelmer R. Vernooij: <http://www.informit.com/title/0131882228>

If You Use Linux, You Should Be Reading **LINUX JOURNAL**TM



- » In-depth information providing a full 360-degree look at featured topics relating to Linux
- » Tools, tips and tricks you will use today as well as relevant information for the future
- » Advice and inspiration for getting the most out of your Linux system
- » Instructional how-tos will save you time and money

Subscribe now for instant access!
For only \$29.50 per year—less than \$2.50 per issue—you'll have access to *Linux Journal* each month as a PDF, in ePUB & Kindle formats, on-line and through our Android & iOS apps. Wherever you go, *Linux Journal* goes with you.

SUBSCRIBE NOW AT:
WWW.LINUXJOURNAL.COM/SUBSCRIBE

Tarsnap: On-line Backups for the Truly Paranoid

If you've ever worried about your data in the cloud or wished backups were easier to automate, the high-security, command-line-friendly approach of Tarsnap backups might make you rest a lot easier at night. **ANDREW FABBRO**

Storing backups in the cloud requires a level of trust that not everyone is willing to give. While the convenience and low cost of automated, off-site backups is very compelling, the reality of putting personal data in the hands of complete strangers will never sit quite right with some people.

Enter Tarsnap—"on-line backups for the truly paranoid". Tarsnap is the brainchild of Dr Colin Percival, a former FreeBSD Security Officer. In 2006, he began research and development on a new solution for "encrypted, snapshotted remote backups", culminating in the release of Tarsnap in 2008.

Unlike other on-line backup solutions, Tarsnap uses an open, documented

cryptographic design that securely encrypts your files. Rather than trusting a vendor's cryptographic claims, you have full access to the source code, which uses open-source libraries and industry-vetted protocols, such as RSA, AES and SHA.

Tarsnap provides a command-line client that operates very much like the traditional UNIX tar command. Familiar syntax, such as `tarsnap cf` and `tarsnap xv f` works as users would expect, except that instead of manipulating local tarballs, the client is working with cloud-based archives. These archives are stored on Amazon S3 with EC2 servers to handle client connections. To minimize bandwidth costs, Tarsnap uses smart rsync-like block-oriented

Unlike other on-line backup solutions, Tarsnap uses an open, documented cryptographic design that securely encrypts your files.

snapshot operations that upload only data set changes and minimize transmission costs.

Although you could roll your own backup system by encrypting your data, running an rsync server off-site and so on, Tarsnap's scriptability, professional cryptographic design and utility pricing make it an attractive service for clueful users.

Open-Source Strong Crypto

Tarsnap compresses, encrypts and cryptographically signs every byte you send to it. No knowledge of cryptographic protocols is required, but if you are an interested enthusiast or feel more comfortable if you can look under the hood, you'll find the source code and full design thoroughly documented on <http://tarsnap.com>.

During installation, the user creates a keyfile, which optionally can be passphrase-protected. The file contains two 2048-bit RSA keys for signing and encryption. AES-256 session keys are created at runtime and encrypted with the RSA keys. The HMAC-SHA-256 hash function is used to verify data blocks and prevent tampering,

and the same hash function is used elsewhere to verify communication between the client and server.

If those cryptographic terms make your eyes glaze over, the simple description is that Tarsnap encrypts and verifies your data in motion and at rest using industry-standard protocols. From a user perspective, the only thing that needs to be safeguarded is the keyfile. As you might expect, if it's lost, the keys to decrypt data stored with Tarsnap are unavailable and your data is no longer usable. After generation, this keyfile ideally should be stored securely off-site.

Efficient Design

Tarsnap backs up variable-length, deduplicated blocks of data. This means that if you change one file in a directory, only that file is backed up. Indeed, only the changes to that file are backed up if the file previously existed. If you move files around, Tarsnap is smart enough to recognize and skip previously backed-up blocks. Anytime you wish, you can take a backup of a directory, which creates a new archive. Behind the scenes, Tarsnap

This is not just “pay by the drink” cloud pricing—it’s practically “pay by the atom”.

will retain whatever files are necessary to make that archive whole.

Suppose you have a directory with 100MB of files and a 10% daily change rate. If you create a Tarsnap archive every day of that directory, you will see the following consumption pattern:

- Monday (initial): 100MB archive created (100MB uploaded, 100MB stored).
- Tuesday: 10MB archive created (10MB new data uploaded, 110MB total data stored).
- Wednesday: 10MB archive created (10MB new data uploaded, 120MB total data stored).

If on Thursday you delete the Monday archive, the Tuesday archive will be “made whole” by retaining whatever files from Monday are needed. This gives you tremendous flexibility to pick restore points. Of course, you can keep multiple archives with however many restore points you wish.

Your Current Account Balance Is **\$4.992238237884881224**

Tarsnap works on a prepaid utility-metered model. Subscribers deposit

a minimum of \$5.00 and are charged only for the storage and bandwidth they consume. Although the cost is higher than plain Amazon S3 service, it reflects both the cryptographic, compression and deduplication value-add of Tarsnap. At the time of this writing, Tarsnap costs 30 cents per gigabyte-month for storage and 30 cents per gigabyte transmitted.

This cost may make Tarsnap infeasible for large, whole-server terabyte-size backups. However, it is ideal for critical, sensitive files that must be durable, available and safe in the event an attacker succeeded in compromising them. With no minimum charge or monthly fees, Tarsnap is very economical for small data sets or for data that compresses well. Some examples:

- Backing up 100MB of files with 10% daily change rate for a month would cost only 30 cents.
- A gigabyte that is backed up weekly with a 20% change rate would cost \$1.40 a month.

Tarsnap bills based on attodollars (quintillionths of a dollar) to avoid

profiting through rounding. This means your account balance is tracked to 18 decimal places. This is not just “pay by the drink” cloud pricing—it’s practically “pay by the atom”. Some users find that a small deposit lasts them months or years.

Important Flexibility

One of Tarsnap’s best features is how easy it is to script. The ability to put a `tarsnap cf` command into a shell script makes use in cron jobs very straightforward, which encourages unattended, automated backups—the best kind.

Crucially, Tarsnap also supports a division of responsibilities. You can use the `tarsnap-keymgmt` tool to create keyfiles with limited authority. You may have one keyfile that lives on your server with permission to create archives, but not the authority to delete them. A master key with full privileges could be kept off-site, so that if attackers were to compromise your server, they would be unable to destroy your backups.

Using Tarsnap

To get started with Tarsnap, register at tarsnap.com, deposit some funds into your account, and download the client.

The client is available only as source, but the straightforward `./configure ; make install` process is very easy. The client is supported on all major

Linux distributions (as well as BSD-based systems). Take a quick peek at the download page to make sure you have the required operating system packages, as some of the development packages are not installed in typical Linux configurations.

If you are using a firewall, be aware that Tarsnap communicates via TCP on port 9279.

There are only two critical configuration items: the location of your keyfile and the location of your Tarsnap cache. Both are set in `/usr/local/etc/tarsnap.conf`. A `tarsnap.conf.example` is provided, and you probably can just copy the example as is. It defines your Tarsnap key as `/root/tarsnap.key` and your cache directory as `/usr/local/tarsnap-cache`, which will be created if it doesn’t exist. The `cachedir` is a small state-tracking directory that lets Tarsnap keep track of backups.

Next, register your machine as follows. In this case, I’m setting up Tarsnap service for a machine called `helicarrier`. The e-mail address and password are the ones I used when I signed up for service with Tarsnap:

```
# tarsnap-keygen --keyfile /root/tarsnap.key
→--user andrew@fabbro.org --machine helicarrier
Enter tarsnap account password:
#
```

I have a directory I’d like to back up

with Tarsnap:

```
# ls -l /docs
total 2092
-rw-rw---- 1 andrew 1833222 Jun 14 16:38 2011 Tax Return.pdf
-rw----- 1 andrew 48568 Jun 14 16:41 andrew_passwords.psafe3
-rw----- 1 tina 14271 Jun 14 16:42 tina_passwords.psafe3
-rw-rw-r-- 1 andrew 48128 Jun 14 16:41 vacation_hotels.doc
-rw-rw-r-- 1 andrew 46014 Jun 14 16:35 vacation_notes.doc
-rw-rw-r-- 1 andrew 134959 Jun 14 16:44 vacation_reservation.pdf
```

To back up, I just tell Tarsnap what name I want to call my archive (“docs.20120701” in this case) and which directory to back up. There’s no requirement to use a date string in the archive name, but it makes versioning straightforward, as you’ll see:

```
# tarsnap cf docs.20120701 /docs
tarsnap: Removing leading '/' from member names
Total size Compressed size
All archives 2132325 1815898
(unique data) 2132325 1815898
This archive 2132325 1815898
New data 2132325 1815898
```

In my `tarsnap.conf`, I enabled the `print-stats` directive, which gives the account report shown. Note the compression, which reduces storage costs and improves cryptographic security. The “compressed size” of the “unique data” shows how much data is actually stored at Tarsnap, and you pay only for the compressed size.

The next day, I back up `docs` again to “`docs.20120702`”. If I haven’t made many

changes, the backup will proceed very quickly and use little additional space:

```
# tarsnap cf docs.20120702 /docs
tarsnap: Removing leading '/' from member names
Total size Compressed size
All archives 4264650 3631796
(unique data) 2132770 1816935
This archive 2132325 1815898
New data 445 1037
```

As you can see, although the amount of data for “all archives” has grown, the actual amount of “unique data” has barely increased. Tarsnap is smart enough to avoid backing up data that has not changed.

Now let’s list the archives Tarsnap has stored:

```
# tarsnap --list-archives
docs.20120701
docs.20120702
```

To demonstrate Tarsnap’s smart approach to storage further, I will delete the oldest backup:

```
# tarsnap df docs.20120701
Total size Compressed size
All archives 2132325 1815898
(unique data) 2132325 1815898
This archive 2132325 1815898
Deleted data 445 1037
```

The “all archives” number has dropped because now I have only one

If I have a local calamity and want to restore that data, it is just another simple Tarsnap command to get my files back.

archive, but the “unique data” has not changed much because it is still retaining all files necessary to satisfy my “docs.20120702” archive. If I list it, I can see my data is still there:

```
# tarsnap tvf docs.20120702
drwxrwxr-x 0 andrew 0 Jun 14 20:52 docs/
-rw----- 0 andrew 48568 Jun 14 16:41 docs/andrew_passwords.psafe3
-rw-rw-r-- 0 andrew 46014 Jun 14 16:35 docs/vacation_notes.doc
-rw-rw-r-- 0 andrew 134959 Jun 14 16:44 docs/vacation_reservation.pdf
-rw-rw-r-- 0 andrew 48128 Jun 14 16:41 docs/vacation_hotels.doc
-rw----- 0 tina 14271 Jun 14 16:42 docs/tina_passwords.psafe3
-rw-rw--- 0 andrew 1833222 Jun 14 16:38 docs/2011 Tax Return.pdf
```

I use a date string for convenient versioning, but I could just as easily use any naming convention for the archive, such as “docs.1”, “docs.2” and so on. For my personal backups, I have a cron job that invokes Tarsnap nightly with a date-string-named archive:

```
tarsnap cf docs.`date '%+Y%m%d'` /docs
```

If I have a local calamity and want to restore that data, it is just another simple Tarsnap command to get my files back. Note that like traditional tar, Tarsnap removes the leading slash so all files are restored relative to the

current working directory:

```
# cd /
# rm -rf docs
# tarsnap xvf docs.20120702
x docs/
x docs/andrew_passwords.psafe3
x docs/vacation_notes.doc
x docs/vacation_reservation.pdf
x docs/vacation_hotels.doc
x docs/tina_passwords.psafe3
x docs/2011 Tax Return.pdf
```

Tips

If you want to run Tarsnap as a nonroot user, create a `.tarsnaprc` file in your home directory. The syntax is identical to the `tarsnap.conf` discussed above. For example:

```
$ cat ~/tarsnap.conf
cachedir /home/andrew/tarsnap-cache
keyfile /home/andrew/tarsnap.key
print-stats
```

If you have other services or users contending for your Internet connection, use `--maxbw-rate` to specify a maximum bytes per second that Tarsnap will be allowed to use.

The `print-stats` command gives

your account status information when used interactively, but for batch operations (such as running Tarsnap out of cron), you can suppress the output by removing that directive from your `tarsnap.conf` or by invoking Tarsnap with `--no-print-stats`.

Finally, you can play with the `--dry-run` and `-v` flags to simulate Tarsnap backup operations without actually burning network and disk. Once you've got your command constructed exactly as you want it, remove `--dry-run`.

License

Tarsnap is not distributed under an open-source license, although all client source is provided (and compiled by the user during install). However, the company regularly contributes back to projects whose code it utilizes, such as libarchive. Tarsnap also has open-sourced some of its own projects, including the `scrypt` package, the `spiped` secure pipe daemon and the `kivaloo` NoSQL data store.

Further Information

The Tarsnap home page (<http://tarsnap.com>) has a wealth of documentation and information, as well as links to the Tarsnap IRC channel, mailing list and FAQ. The “Technical Details” section is absorbing reading for those interested in the deep details of Tarsnap’s cryptographic approach and history.

Tarsnap also pays significant cash bounties for bugs found in the product, ranging from a few dollars for small cosmetic bugs up to a couple thousand dollars if someone finds a serious security flaw. This transparent approach is further comfort for the truly paranoid.

Tarsnap’s current version is 1.0.32, released on February 22, 2012, for Linux, BSD, OS X, Solaris, Minix and Cygwin. ■

LINUX JOURNAL on your Android device

Download
app now in
the **Android
Marketplace**



www.linuxjournal.com/android

Andrew Fabbro is a senior technologist living in the Portland, Oregon, area. He's used Linux since Slackware came on floppies and presently works for Con-way, a Fortune 500 transportation company.

Q&A with Dr Colin Percival

Q. In May 2012, you retired as FreeBSD Security Officer to focus more on Tarsnap. Sounds like Tarsnap is doing very well. Can you share any stats on your growth to date?

A: I've heard "startup company" defined as "time to double revenue or the number of users is measured in months", and I've heard "highly successful startup company" defined as "time to double revenue or the number of users is measured in weeks". By those definitions, Tarsnap is a startup company, but not a highly successful one.

And yes, that is dodging the question, but Tarsnap is my primary source of income, and I come from a culture that considers someone's income to be a private matter, so I don't want to publish precise numbers.

Q. Looking at your Bug Bounties page, you've paid

Dr Colin Percival



**Keep out of
Direct Sunlight**

out more than \$2,000 to users who've submitted bug reports. Why a "bug bounty" system as opposed to the traditional bug reporting in open-source projects?

A: I gave a talk about this at BSDCan'12 and AusCERT'12 (in consecutive weeks, no less—a word

of advice, don't ever try to attend back-to-back conferences 10,000 miles apart). In short, bug bounties help get more people looking at code, and help encourage people to report anything they see that seems wrong.

Probably a better question is why I offered bounties for all bugs rather than just for security bugs—aside from Knuth's famous prizes, I don't know of any other case where bounties extend beyond security vulnerabilities. The answer here is roughly the same though—there's a lot of people who won't look for security bugs because they don't have a security background, but offering cash for *all* bugs gets them interested...and my experience with FreeBSD is that many security vulnerabilities aren't found by security people, but rather by other developers just saying "something here looks weird".

Q. What do you think are the weakest points of the Tarsnap design, from a security perspective? Is there anything that should keep the truly paranoid up at night?

The weakest link in Tarsnap's security, without question, is me. I wrote the Tarsnap client code to

encrypt and sign everything on the client side, but how do you know that it does what I claim?

If you're paranoid, you should look at the code yourself and make sure it does what I claim it does. And when I release a new version, you should look at that too—compare against the previous version and make sure that the changes are sensible.

If the CIA kidnaps me and threatens to torture me until I decrypt someone's data, I won't be able to do anything to help them. But if the CIA kidnaps me and threatens to torture me unless I insert a Trojan horse into the next version of Tarsnap...well, I'm not optimistic about my ability to withstand torture.

Q. Tarsnap's client is very UNIX-nerd-friendly, with its familiar tar syntax. Do you have any interest or plans for a graphical interface for less-sophisticated users?

A: This is absolutely something I plan on doing...in the future. I have very little experience with anything GUI, so I'll probably end up paying someone to produce a GUI—ideally an open-source GUI for tar, since Tarsnap uses almost exactly the same command-

LINUX JOURNAL

on your e-Reader

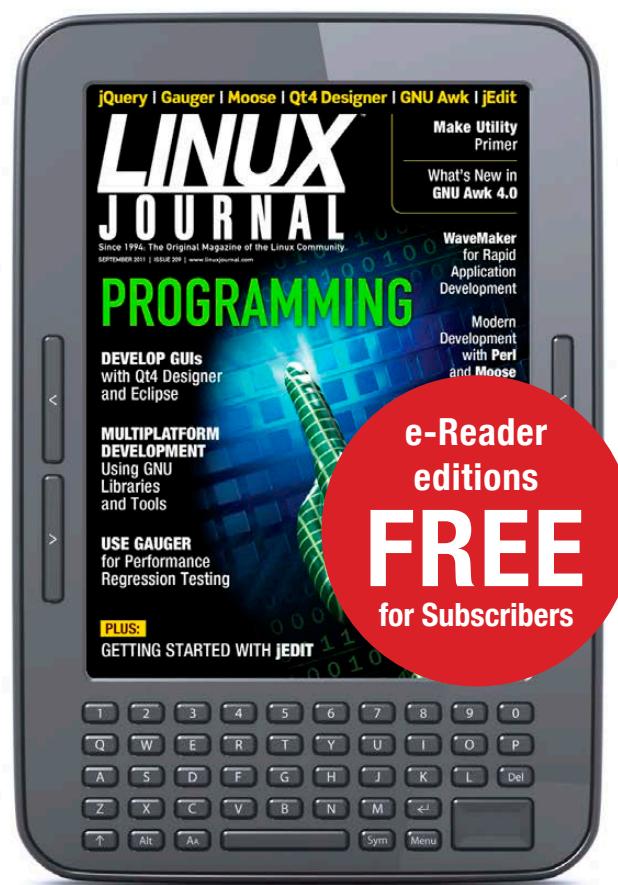
line interface, and I think a good GUI front end to tar would be useful for its own sake too.

Q. Why attodollars and not femtodollars or zeptodollars?

A: With attodollars, I can express everything as a pair of 64-bit integers.

Q. Do you have any new features in the works for upcoming Tarsnap releases?

A: I'm mostly working on performance improvements these days. There's a few frequently requested features that I might add, but in general, there are good reasons when features don't exist—either they're impossible (for example, server-side expiration of old archives—the server has no way to know which blocks should be freed when an old archive is deleted) or would require a substantial redesign (for example, renaming archives—the client-server protocol has write transactions and delete transactions, but renaming would need to atomically write some files and delete others).



Customized
Kindle and Nook
editions
now available

LEARN MORE



DOC SEARLS

Debugging Free Markets That Still Aren't

Free markets should value free customers. Today's mass markets still value captive ones. Some of us are starting to fix that.

I've been writing for *Linux Journal* since 1996, and putting my learnings to work along the way.

For example, in 1998, when a bunch of geeks got together and decided to expand the free software conversation to include a broader category they called open source (<http://www.catb.org/~esr/open-source.html>), I watched—and learned—in amazement as their strategy succeeded, in spades. Before February 1998, “open source” was hardly uttered outside a few specialized circles, such as the military. Thanks to Eric S. Raymond and other hacker evangelists, the term became common parlance throughout civilization.

So, a year later, when I got together with three other guys and put *The Cluetrain Manifesto* up on the Web

(<http://cluetrain.com>), our first thesis was “markets are conversations”, in part because we saw it proven already. That summer, when we expanded the Web site into a whole book, we said this in Chapter Four:

What's more, these new conversations needn't just happen at random. They can be created on purpose. “We hackers were actively aiming to create new kinds of conversations outside of traditional institutions”, Raymond says. “This wasn't an accidental byproduct of doing neat techie stuff; it was an explicit goal for many of us as far back as the 1970s. We intended this revolution.”

Nice job.

Above *Cluetrain*'s 95 theses, was an alpha clue, written by Chris Locke and addressed to marketers who were busy failing to understand how profoundly this new thing, the Internet, changed markets by making its human occupants—and not just marketers—more powerful. It said this:

we are not seats or eyeballs or end users or consumers. we are human beings and our reach exceeds your grasp. deal with it.

For years the number increased by an average of one per day, until it reached five thousand, where it stands. Google's meter is pegged there.

Yet for all the successes of free software, Linux, open source and *Cluetrain* (in that order), a problem persists. As human beings, our reach does not yet exceed the grasp of marketers. And, in many ways, the problem we saw at the height of the dot-com bubble is worse today, as we are lifted by marketing gas to the

Before February 1998, “open source” was hardly uttered outside a few specialized circles, such as the military. Thanks to Eric S. Raymond and other hacker evangelists, the term became common parlance throughout civilization.

More than any other single line, that adrenalized us. It also placed *Cluetrain* clearly on the side of people. As Jakob Nielsen put it to me at the time (in words close to exactly these), “You guys defected from marketing, and sided instead with people in their fight against marketing.” (We were all techies, but three of us were also marketing wretches, fed up with the whole thing.)

When Google Books made possible search across countless books, I was able to observe with it the number of books in which the word “cluetrain” appeared.

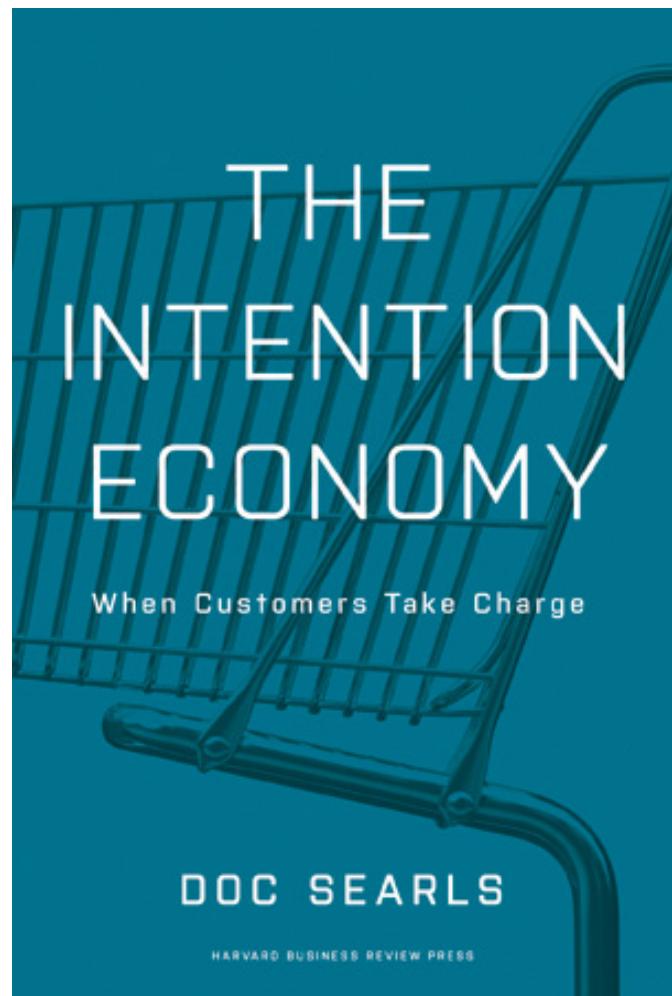
height of a new bubble variously called “Facebook”, “social media”, “big data” and “personalized advertising”.

So, in 2006, when I was given a fellowship with the Berkman Center for Internet and Society at Harvard University, I decided to launch a development project there. The purpose of the project was to give human beings a reach that exceeded marketers’ grasp. We called it ProjectVRM (<http://blogs.law.harvard.edu/vrm>), in which the last three letters stood for Vendor Relationship Management. The term was meant as the customer-side

counterpart of Customer Relationship Management, the \$18 billion enterprise software and services business that gives the world call centers, junk mail and other ways of creating distance between customers and the companies that purport to serve them. The idea wasn't to fight CRM, but to engage it—and organizations of all kinds—from the customer side, by providing individuals with means for both independence and engagement.

As of today, there are dozens of VRM development efforts, operating in dozens of countries. All are still at the narrow end of the adoption curve, but it's clear where they're all headed. Once VRM succeeds, I believe, we will enjoy an Intention Economy, based on what each of us intends as sovereign human beings participating in a free and open marketplace. In it I claim that free customers will prove more valuable than captive ones, because our reach will outperform vendors' grasp—and render that grasping obsolete.

So, starting a couple years ago, I began work on a book titled *The Intention Economy: When Customers Take Charge*, for Harvard Business Review Press, based on progress in VRM development. It came out in May of this year, and it's available in all the usual places and forms, including digital and audio ones. With permission of the publisher, I would like to share with you one chapter from the book. There are other chapters that



actually go into free software, open source, Linux and other topics more typical of *Linux Journal*, but mostly what I do in those chapters is leverage the stuff I learned here, along with the rest of you. This chapter is one that visits a larger problem rooted in the ethos of Business as Usual, especially on the Web. ■

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.

YOUR CHOICE OF CAPTOR

Excerpt from *The Intention Economy*

"Find out just what any people will quietly submit to and you have the exact measure of the injustice and wrong which will be imposed on them."—Frederick Douglass

"The term 'client-server' was invented because we didn't want to call it 'master-slave'."—Craig Burton

DOC SEARLS

The Argument

The World Wide Web has become a World Wide Ranch, where we serve as calves to Web sites' cows, which feed us milk and cookies.

Wholly cow

The Internet and the Web are not the same things. The Internet is a collection of disparate networks whose differences are transcended by protocols that put every end at zero functional distance from every other end. The World Wide Web is one application that runs on the Internet. Others include e-mail, messaging, file transfer, chat and newsgroups, to name just a few. But the World Wide Web is the big one—so big that we tend to assume it's the whole thing, especially since the Web is where we spend most of our time online and when it has absorbed so many other

formerly separate activities. E-mailing, for example.

Sir Tim Berners-Lee, who invented the Web in 1989, later said it aspired to be "a universal linked information system, in which generality and portability are more important than fancy graphics techniques and complex extra facilities. The aim would be to allow a place to be found for any information or reference which one felt was important, and a way of finding it afterwards." He did not intend to create a vast online shopping mall, an industrial park of advertising mills, or home to a billion-member "social network" owned by one company. But that's what the Web is today. True, somewhere in the midst of all that other stuff, you can still find the simple, linked collection of documents that Sir Tim meant for the Web to be in the first place. But that collection is

getting harder and harder to find, mostly because there's no money in it.

Thus, the Web we know today is largely the commercial one that appeared in 1995, when Netscape and Microsoft created the first retailing Web servers along with the first widely used Web browsers. The commercial Web's early retailing successes—notably Amazon and eBay—remain sturdy exemplars of selling online done well. A key to their continued success is *personalization*, made possible by something called the *cookie*: a small text file, placed in your browser by the Web site, containing information to help you and the site remember where you were the last time you visited.

Those ur-cookies have since evolved (among other things) into breeds of Trojan marketing files. Variously called "Flash cookies" (based on Adobe's proprietary Flash technology), "tracking bugs", "beacons", and other names, they track your activities as you go about your business on the Web, reporting back what they've found to one or more among thousands of advertising companies, most of which you've never heard of (but some of which we visited in the last chapter).

The online advertising business says the purpose of these tracking methods is to raise the quality of the advertising

you get. But most of us care less about that than we do about being followed by companies and processes we don't know and wouldn't like if we did.

While much complaining is correctly addressed to the creepy excesses of the online advertising business, little attention has been paid to the underlying problem, which is the design of the commercial Web itself. That design is *client-server*, which might also be called calf-cow. Clients are the calves. Servers are the cows. Today, there are billions of commercial cows, each mixing invisible cookies into the milk they feed to visiting calves.

Client-server by design is submissive-dominant, meaning it puts servers in the position of full responsibility for defining relationships and maintaining details about those relationships. And, since this is all we've known since 1995, most of the "solutions" we've come up with are more sites and site-based services. In other words, better cows.

What are your names?

Back before computing got personal, and all the computing that mattered was done in enterprises, one of the biggest problems was a proliferation of *namespaces*. (In technical terms, a namespace is an identifier or a directory for them.) Different software systems

each had their own namespaces. For the enterprise, this meant an employee, or information about an employee, might be known by different names and attributes, within each of the separate software systems used in human resources, marketing, sales, accounting, and so on. For an employee, this meant maintaining up to dozens of different logins and passwords. For the HR and IT departments, it meant integration hell, or the impossibility of any integration at all. To this day, the namespace problem is one of the most vexing in all of enterprise computing.

Out on the Web, that's solved for sellers, because each tends to have one customer-facing system. But for us customers, it's still a mess, because the old corporate namespace problem is now ours, multiplied by the number of relationships we have on the Web. Thus, we are required to keep track of as many different logins and passwords as there are Web sites. This makes the problem as big as the sum of all Web sites. At last count (as I write this), there are over 200 million registered domains, about half of which have the commercial .com suffix. Since some domains have many retail sites (eBay and Etsy, for example), the total number of commercial sites is much higher. Search for "privacy policy" and Google will tell you it appears on

more than a billion Web pages. Even if one login and password might get you to all of eBay and Etsy, we are still talking about hundreds of login-password combinations. Enterprises don't care, because it's not their problem. They care only about their relationship with you. Not about your relationships with every other company. That leaves you and me with an umpty-million (or -billion) namespace problem.

Login rolling

There are partial workarounds. You can set up browsers to automatically fill out forms and auto-complete commonly typed strings of words and numbers. There are programs that remember passwords for you. But the most popular "single sign-on" (SSO) methods are provided by grace of your relationship with Facebook, Google, Twitter, or Yahoo. Those ubiquitous "Connect with Facebook" buttons, for example, are the user-facing side of a program called Facebook Connect.

When Facebook announced Facebook Connect in December 2008, Mark Zuckerberg wrote this in the company blog [<http://blog.facebook.com/blog.php?post=41735647130>]:

Over the summer we announced [<http://blog.facebook.com/blog.php?post=24577977130>]

an extension of the Facebook Platform called Facebook Connect [<http://www.facebook.com/help/?page=229348490415842>]. Facebook Connect makes it easier for you to take your online identity with you all over the Web, share what you do online with your friends and stay updated on what they're doing. You won't have to create separate accounts for every website, just use your Facebook login wherever Connect is available.

To call your Facebook identity your "online identity" is delusional as well as presumptuous, because none of us interact only with Facebook on the Web. But that by itself isn't the only problem. The larger problem with Facebook Connect is unintended data spillage. For example, what if you don't want to share "what you do online with friends", or if you think you're just taking Facebook's shortcut when you log in for the first time at some other site? In other words, what if you only want Facebook Connect to be what it presents itself as: a simple login option for you at another site—with no "social". Or, if you do want to be "social", how about being selective about what Facebook data you're willing to share—exposing some

data to some sites and different data (or no data at all) to other sites? In other words, what if your "privacy preferences" aren't just the one-set-fits-all selections you've made at Facebook?

The site I Shared What?!? [<http://ISharedWhat.com>], created by VRM developer Joe Andrieu, does an excellent job of simulating what you reveal about yourself and others when you use Facebook Connect. Here's what it just told me:

You've just shared: your basic details, your news feed, your friends on Facebook, the activities listed on your profile, the interests listed on your profile, music listed on your profile, books listed on your profile, movies listed on your profile, television shows listed on your profile, all the pages you have "liked", your shared links.

That's after I jiggered my privacy settings in Facebook to reveal as little as possible. The results were the same both times, before and after. I'm no dummy, but—as of this writing—I have no idea how to keep from spilling rivers of personal data when using Facebook Connect. So I just avoid it.

But I'm the exception. As of December 2010, a quarter-billion people were

logging into 2 million Web sites using Facebook Connect, with 10,000 more Web sites being added every day.

Privacy is the marquee issue here, but the deeper problem is control. Right now you don't have much control over your identity on the Web. Kim Cameron, father of the cross-platform Identity Metasystem when he was an Architect at Microsoft, says [<http://www.identityblog.com/?p=838>] your "natural identity"—who you are and how you wish to be known to others—gets

no respect on the commercial Web. Every cow thinks you are its calf, "branded" almost literally. In the World Wide Dairy, all of us walk around with a brand for every Web site that gives us milk and cookies. Our virtual hides are crowded with more logos than a NASCAR racer.

Who you are to the other cows is not of much interest to any one cow, unless they "federate" your identity with other cows. Think of federation as "large companies having safe sex with customer data". That's what I called it in a keynote

LINUX JOURNAL

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



Available on the
App Store

linuxjournal.com/ios

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



I gave at Digital ID World in the early 2000s. It still applies.

Tough teats

A similar problem comes up when you have multiple accounts with one site or service, and therefore multiple namespaces, each with its own login and password. For example, I use four different Flickr accounts, each with its own photo directory:

- Doc Searls—<http://www.flickr.com/photos/docsearls>
- Linux Journal—<http://www.flickr.com/photos/linuxjournal>
- Berkman Center—<http://www.flickr.com/photos/berkmancenter>
- Infrastructure—<http://www.flickr.com/photos/infrastructure>

The first is mine alone. The second I share with other people at *Linux Journal*. The third I share with other people at the Berkman Center. The fourth I share with other people who also write for the same blog.

Flickr in each case calls me by the second person singular “you” and does not federate the four. To them, I am

four different individuals: one cow, four calves. (Never mind that three of those sites have many people uploading pictures, each pretending to be the same calf.) My only choice for dealing with this absurdity is deciding which kind of four-headed calf I wish to be. Either I use one browser with four different logins and passwords, or I use four different browsers, each with its own jar of cookies. Both choices are awful, but I have to choose one. So I take the second option and use one browser per account—on just one laptop. When I use other laptops, or my iPhone, my Android, or the family Nokia N900, an iPod Touch, or an iPad, I’m usually the first kind of calf, using one browser to log in and log out every time I post pictures to a different account. Which I mostly don’t do at all, because it’s one big pain in my many asses.

Crazy as all this is, it hardly matters from the cow side of the cow-calf relationship. In fact, the condition is generally considered desirable. After all, the World Wide Ranch is a real marketplace: one where cows compete for calves. Hey, what could be more natural?

The media, even online, are no help either. Business publications, both online and off, love to cover what I call “vendor sports”, in which customers are prizes for corporate trophy cases.

Advertiser Index

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

That's why business writers often regard "owning" customers as a desirable thing. For example, in "Rim, Carriers Fight Over Digital Wallet", from the March 18, 2011, *Wall Street Journal*, Phred Dvorak and Stuart Weinberg write, "RIM and carriers like Rogers Communications Inc. in Canada, and AT&T Inc. and T-Mobile in the U.S., disagree over exactly where on the phone the credentials should reside—and thus *who will control the customers, revenue and applications that grow out of mobile payments.*" The italics are mine.

In the dairy farm model of the marketplace, captive customers are more valuable than free ones. Since this has been an ethos of mass marketing for a century and a half, the milk-and-cookies system of customer control has proven appealing to businesses outside the online world as well as within it. As of today, both worlds are worse off for it as well.

So, then

For free markets to mean more than "your choice of captor", we need new systems that operate on the principle that free customers are more valuable—to both sellers and themselves—than captive ones. Improving slavery does not make people free. We need full emancipation. That's the only way we'll get free markets worthy of the name.■

| ADVERTISER | URL | PAGE # |
|---|---|--------|
| 1&1 | http://www.1and1.com | 31 |
| CLOUDOPS 2012: CLOUD COMPUTING FOR ASSET MANAGERS | http://www.frallc.com | 51 |
| DRUPALCON MUNICH | http://munich2012.drupal.org | 33 |
| EMAC, INC. | http://www.emacinc.com | 57 |
| HPC ON WALL STREET | http://www.flaggmgrt.com/hpc | 101 |
| iXSYSTEMS, INC. | http://www.ixsystems.com | 7 |
| KIWI PYCON | http://nz.pycon.org | 59 |
| LINUXCON NORTH AMERICA | https://events.linuxfoundation.org/events/linuxcon | 2 |
| MICROWAY, INC. | http://www.microway.com | 66, 67 |
| SILICON MECHANICS | http://www.siliconmechanics.com | 3 |
| TEXAS LINUX FEST | http://texaslinuxfest.org | 41 |
| USENIX OSDI CONFERENCE | https://www.usenix.org/conference/osdi12 | 17 |

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>.