# LINUX JOURNAL™

SPONSORED BY

New Relic®

Since 1994: The Original Magazine of the Linux Community

NOVEMBER 2014 | ISSUE 247 | www.linuxjournal.com

BACK UP LARGE VOLUMES OF DATA WITH ZBACKUP

DEPLOY A STORAGE SOLUTION WITH ZERO DOWNTIME

SHARE ADMIN ACCESS FOR MANY HOSTS SECURELY

+ A Look at the vtop System Monitor

## SYSTEM ADMINISTRATION

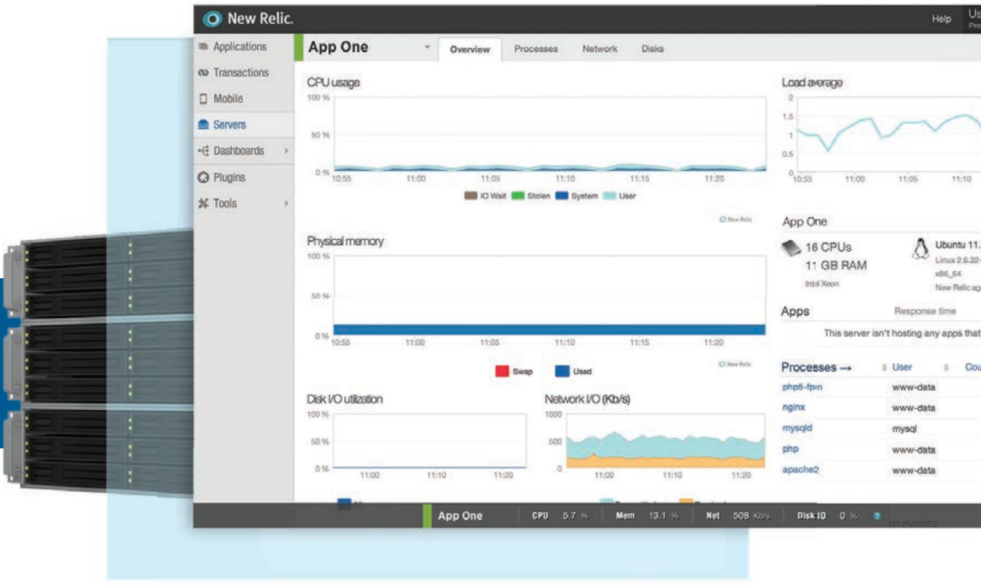
INTRODUCING
The DevOps Mindset

WATCH: ISSUE OVERVIEW

# Healthy servers make for a healthy app.

New Relic Servers™ lets you view and analyze critical system metrics so that you can make sure your application is always in tip-top shape.

**Get server monitoring from the app perspective:**
www.newrelic.com/servers



**Get visibility into:**
- Server and disk capacity
- CPU, memory, and disk I/O utilization
- Processes prioritized by memory or CPU consumption
- Cloud, physical, or hybrid environments

Companies using New Relic

Microsoft  rdio  airbnb  SONY  EA  NBC

New Relic® SERVERS™

# CONTENTS

NOVEMBER 2014
ISSUE 247

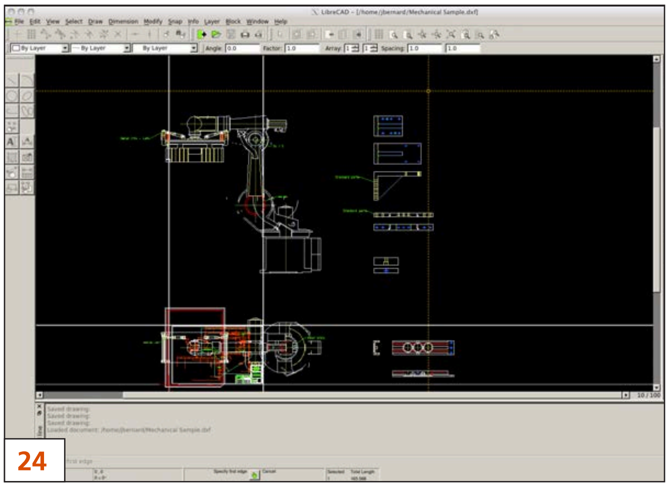
## SYSTEM ADMINISTRATION

### FEATURES

**Cover Image:** © Can Stock Photo Inc. / patrimonio

**ON THE COVER**

- Back Up Large Volumes of Data with zbackup, p. 64
- Deploy a Storage Solution with Zero Downtime, p. 76
- Share Admin Access for Many Hosts Securely, p. 84
- Introducing the DevOps Mindset, p. 52
- A Look at the vtop System Monitor, p. 96

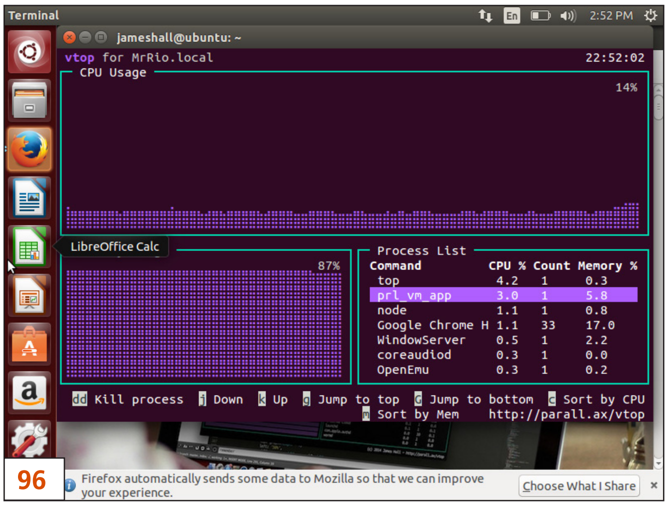

24

I'm Your Moon by Jonathan Coulton - Tomahawk

Jonathan Coulton is a **geek rock** / **folk rock** musician active since 2003 from Brooklyn, known for his songs about geek culture. He is best known for such compositions as "**Still Alive**" and "**Code Monkey**", as well as his cover of Sir Mix-A-Lot's "**Baby Got Back**". He is the Contributing Troubadour for Popular Science magazine and the musical director for John Hodgman's Little Gray Book Lectures. Every week for a year Coulton released a new song for free on his website as part of his Thing a Week podcast...
**Read more about Jonathan Coulton on Last.fm.**

| | | |
|---|---|---|
| 1 | Code Monkey | 06:05 |
| 2 | Re: Your Brains | 04:44 |
| 3 | I Feel Fantastic | 03:09 |
| 4 | Skullcrusher Mountain | 04:17 |

32


```
vtop for MrRio.local                                          22:52:02
 CPU Usage
                                                                   14%

                                     87%    Process List
                                            Command           CPU % Count Memory %
                                             top               4.2   1     0.3
                                             prl_vm_app        3.0   1     5.8
                                             node              1.1   1     0.8
                                             Google Chrome H   1.1   33    17.0
                                             WindowServer      0.5   1     2.2
                                             coreaudiod        0.3   1     0.0
                                             OpenEmu           0.3   1     0.2
 dd Kill process  j Down  k Up  g Jump to top  G Jump to bottom  c Sort by CPU
                                             m Sort by Mem    http://parall.ax/vtop
```

Firefox automatically sends some data to Mozilla so that we can improve your experience.

96

*LINUX JOURNAL* (ISSN 1075-3583) is published monthly by Belltown Media, Inc., 2121 Sage Road, Ste. 395, Houston, TX 77056 USA. Subscription rate is $29.50/year. Subscriptions start with the next issue.

LINUX JOURNAL

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

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

| | |
|---|---|
| **President** | Carlie Fairchild<br>publisher@linuxjournal.com |
| **Publisher** | Mark Irgang<br>mark@linuxjournal.com |
| **Associate Publisher** | John Grogan<br>john@linuxjournal.com |
| **Director of Digital Experience** | Katherine Druckman<br>webmistress@linuxjournal.com |
| **Accountant** | Candy Beauchamp<br>acct@linuxjournal.com |

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

**Editorial Advisory Panel**
Brad Abram Baillio • Nick Baronian • Hari Boukis • Steve Case
Kalyana Krishna Chadalavada • Brian Conner • Caleb S. Cullen
Keir Davis • Michael Eager • Nick Faltys • Dennis Franklin Frey
Victor Gregorio • Philip Jacob • Jay Kruizenga • 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

SILICON MECHANICS
SERVERS • STORAGE • HPC

Are you tired of dealing with proprietary storage?

zStax®
ZFS Unified Storage
Powered by NexentaStor

## **zStax StorCore** ZFS Unified Storage from Silicon Mechanics is truly software-defined storage.

From modest data storage needs to a multi-tiered production storage environment, **zStax StorCore** ZFS unified storage appliances have the right mix of performance, capacity, and reliability to fit your needs.

### zStax StorCore 64

The **zStax StorCore 64** utilizes the latest in dual-processor Intel® Xeon® platforms and fast SAS SSDs for caching. The zStax StorCore 64 platform is perfect for:

- small-medium office file servers
- streaming video hosts
- small data archives

### zStax StorCore 104

The **zStax StorCore 104** is the flagship of the zStax product line. With its highly available configurations and scalable architecture, the zStax StorCore 104 platform is ideal for:

- backend storage for virtualized environments
- mission critical database applications
- always available active archives

Talk with an expert today: 866-352-1173 - http://www.siliconmechanics.com/zstax

SHAWN POWERS

# Folger's Crystals

Every time I write a Bash script or schedule a cron job, I worry about the day I'll star in my very own IT version of a Folger's commercial. Instead of "secretly replacing coffee with Folger's Instant Crystals", however, I worry I'll be replaced by an automation framework and a few crafty FOR loops. If you've ever had nightmares like that, you're in the right place. The truth is, the need for system administrators isn't going down—it's just that our job function is shifting a little. If you stay current, and resolve to be a lifelong learner, system administration is as incredible as it's always been. (And far better than instant coffee! Yuck!) This month, we focus on system administration. It keeps us all relevant, all informed and most important, we should all learn a little something along the way.

Reuven M. Lerner starts off the issue discussing the power of NoSQL databases using PostgreSQL. If that seems like a contradiction in terms, you'll want to read his column for more details. Dave Taylor follows up with part two in his series on a script-based dream interpreter. You also will learn a few handy scripting tips along the way that will be useful regardless of the project you're creating. When the series is done, perhaps the dream interpreter can help me figure out my recurring nightmare of being Winnie the Pooh being hunted by angry bees. Or maybe I should just lay off the honey for my tea.

Kyle Rankin discusses DNS this month, but instead of setting up DNSSEC, he describes how to set up DNS caches to make your networks more efficient. While Kyle doesn't normally care for dnsmasq as a DNS/DHCP dæmon, in this article, he turns to it for its caching abilities. If you see your internal DNS servers getting hammered, a caching situation might be just what you need. I follow Kyle with an article on DevOps. Although it makes sense to start a series on DevOps tools like Chef, Puppet, Ansible and so on, the first thing to do is understand what DevOps is all about. If you ask six people to define DevOps,

**VIDEO:**
Shawn Powers runs through the latest issue.

you'll get a dozen different answers. This month, I try to clear up any confusion or misconceptions, so that you can benefit from the DevOps idea rather than be scared or confused by it.

No system administration issue would be complete without addressing the most important issue facing the sysadmin. No, it's not uptime—it's backups. David Barton walks through using zbackup, which is a deduplicating system for backing up massive amounts of data as efficiently as possible. When a single desktop hard drive can hold 4TB of data or more, the task of backing up becomes monumental. David makes it a little more manageable.

Petros Koutoupis follows David with that other topic sysadmins are concerned with: uptime. Migrating data from one system to another often is expensive and time-consuming, and it usually means proprietary SAN storage. With the advent of High-Availability Logical Volume Management (HA-LVM), that same flexibility comes to folks using commodity hardware and open-source tools. Petros explains the concept and process for creating and maintaining highly available LVM solutions for the data center.

System administrators know that although central authentication is a key to a successful network infrastructure, there also are local accounts on servers and devices that must be kept local, and yet still used. J.D. Baldwin shows an elegant way to manage those local SSH accounts by leveraging ssh-agent to store authentication keys. If you've ever had to change the password on 100 servers because an employee left the company, you understand the problem. J.D. has an awesome solution.

Finally, James Hall describes the process he used when developing the incredibly cool vtop program. A graphical activity monitor for the command line alone is a enough to warrant an article, but James covers his entire process from planning to creating to improving. Even if you have no interest in using a CLI GUI-based program, it's a great way to learn about the open-source journey. If you want to see how an idea becomes a package, James' article is perfect.

We've also got a ton of tips, tricks, hints and pointers throughout the issue. If you want to hear about the latest product announcements, or just discover our favorite app of the month, this issue aims to please. Whether you're a system administrator, a developer or even a new Linux user getting your open-source feet wet, we hope you enjoy this issue as much as we enjoyed putting it together.■

**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

LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
NETWORK SECURITY and SSH What You Need to Know
EMBEDDED
HOW-TO: Sump Pump Monitor with Raspberry Pi and Python
AN INDEPTH LOOK at the U-Boot Environment Anatomy
+ Interview with the Creators of the "Hello World Program"
Advanced Vim Macro Techniques
Use the BeagleBone Black TO COMMAND YOUR ELECTRONIC GEAR

## JuiceSSH Instead of ConnectBot

I have been a Linux user for almost two decades, but it was a good idea to subscribe to *LJ* more than a year ago. I keep finding lots of good articles in every issue.

In your July 2014 issue in the "Remote System Administration with Android" article, Federico Kereki suggests ConnectBot for SSH on Android, although as far as I know, it is no longer maintained. There is a better alternative called JuiceSSH.

Although it's not completely free (but neither do we work for free, right?), it is well worth the money for the features it supports—for example, Mosh (**https://mosh.mit.edu**), which comes in handy over slow mobile connections, dynamic resize of the screen area and so on.

I also have a foldable full-sized QWERTY keyboard and a micro USB to USB female OTG adapter cable so I do not have to use the onscreen keyboard if I know I will have to type a lot, but rather a proper keyboard.
**—Balazs**

***Federico Kereki replies:*** *Mr Kinszler is almost right as to ConnectBot. Although there were no updates for a long time, work on it has seemingly restarted, a pre-release version is available at* **https://github.com/connectbot/connectbot/releases**, *and a new release may be available soon. As to his suggestion regarding JuiceSSH, I haven't tried it out, but I will; it seems to be a valid alternative. Finally, I also agree with him that a external keyboard is best. Personally, I own a rollable Bluetooth keyboard that does the job well.*

## Extremist Interpretations

Regarding Doc Searls' "Stuff That Matters" article in the September 2014 issue: isn't it possible that the word "extremist(s)" in the NSA's surveillance system XKEYSCORE is ambiguous? Isn't it possible, or even likely, that the word "some" is implied in the phrase "advocated by extremists", as in "advocated by *some* extremists?" To me, it seems far more likely that the phrase is an example of the logical argument "All bad-guys are extremists, but not all extremists are bad-guys", than it is leap to the nefarious conclusion that Cory Doctorow describes in his article as "nothing short of bizarre".

Kyle Rankin's own article affirms that being flagged as "extremist" targets you for further surveillance. This implies that the system is in a narrowing down phase and hasn't yet concluded you're a bad guy. For example, the September 11th hijackers were identified from a set of all airline passengers. Indeed, all the September 11th hijackers were airline passengers, but not all airline passengers were hijackers.

I believe this article is premature in its conclusions and is out of character for the intellectually honest work I've come to expect from Doc. So much so, that I believe this conclusion was probably reached to generate letters to the editor. I do freely admit however, that mine is probably an extremist opinion.
**—Jon Redinger**

## Finding Leap Year

For Dave Taylor, here's yet another approach. Just implement the basic rule: a leap year is any year divisible by four, except that divisible by 100 is not, except that divisible by 400 is:

```
$ cat leapyear
yr=$1
if [ "$yr" -le 0 ]
then    echo "usage:  leapyear year"
        exit 1
fi

if [ $yr -eq $(expr $yr / 4 '*' 4) ]
then    if [ $yr -eq $(expr $yr / 100 '*' 100) ]
        then    if [ $yr -eq $(expr $yr / 400 '*' 400) ]
                then    echo $yr is a leap year
                else    echo $yr is not a leap year
                fi
        else    echo $yr is a leap year
        fi
else    echo $yr is not a leap year
fi
```

**—Bill Rausch**

***Dave Taylor replies:*** *Nice, simple straightforward code. Thanks for sending that in, Bill!*

## Calculating Days

Calculating days between dates is more complicated than you would expect at first sight. First, you must ask yourself the question: "Where am I? Europe, Asia, America...."

Second, "which country?" Because all of this has an influence on the date.

Third, "which calendar am I using?"

The most accurate calendar is the Mayan calendar, not the Gregorian or western-known types of time tracking. So I suggest you start from this calendar and first convert the date to the Mayan calendar, and then calculate these days between dates, and then convert to normal dates. This will avoid a lot of mistakes due to "modern calendars".

For example, the Thai calendar is, for us, in the future. In Europe, we have known a lot of calendars in different periods and places.
**—Patrick Op de Beeck**

## Windows XP IE 8 and kernel.org

I am not sure if anyone at *LJ* or any of your readers can answer this. When I try to go to the **http://www.kernel.org** Web site using Win XP and IE 8, it says the page can't be displayed. I tried to find something in XP that would block the site, but had no luck. I can access the site from all other systems on my network, so I know it is not a connection problem. Have any of your readers reported this before? Does anyone know how to fix this? Is Microsoft deliberately blocking the kernel.org site because it is a competing OS?
**—Mike**

*I suspect it's not an intentional blocking of the kernel.org Web site, but probably some outdated implementation of something (DNS? SSL cert?) that makes the sites inaccessible. Since it's XP, and it's Microsoft, I would be shocked if it was something that ever gets fixed. I don't have XP to test it, but it wouldn't surprise me if it failed on every XP install. If you're stuck using that computer, maybe try a different browser? Or a proxy? Otherwise, I do know a pretty awesome operating system you could try.—Shawn Powers*

## NSA's Extremist Forum

Do not believe everything you read in the papers—ref: "you could be fingerprinted". I would almost guarantee there are NSA employees (*nix sysadmins, security managers, etc.) who subscribe to this journal because it is a quality source of information. They are not "extremist", and I can only assume that they are not forbidden from reading this

journal. Perhaps the NSA may contain some people like you and me.
**—Jon**

*Oh, I'm sure there are some fellow Linux nerds that work for the NSA, and most likely, some read* Linux Journal. *Being flagged as possible extremists was likely either automated based on our encryption articles, or manually due to the same sort of thing. Although I find it frustrating to be considered a potential threat, I'm trying to take it as a compliment. Open-source ideals are radical, and we are a passionate bunch. Does that make us suspect? Even if it does, I have no desire to change!—Shawn Powers*

## DNSMasq

Thanks for Shawn Powers' excellent article on DNSMasq in the September 2014 issue. I have a couple small networks where bind has always been an overkill. DNSMasq is great, as is loading data directly from /etc/hosts.
**—Richard Ruth**

*Thanks Richard! I was surprised just how powerful it can be. I don't know why I never considered it outside of an embedded router, but I've been incredibly impressed with it. I'm glad you find it useful as well.—Shawn Powers*

## Linux Digital Microscope

I would like to connect a USB digital microscope to my Linux laptop to show my children all the tiny life that is in pond water.

Do you have any ideas?
**—Pat**

*I think many USB microscopes function like a Webcam, so if you can figure out whether it's UVC-compatible, you should be able to tell if it will work with Linux. I think model numbers and Google will be your best friends in this endeavor.—Shawn Powers*

## Dave Taylor's September 2014 Article

I have two items to mention regarding Dave Taylor's article on calculating days between dates.

1) There is a bug in the code found within the "Days Left in Year" section. The output of the date command pads the results with zeros by default. In bash, a numeric value that begins with a zero is evaluated as an octal (base 8) number. This is demonstrated by the script output:

```
Calculating 366 - 035
There were 337 days left in the starting year
```

366 – 35 = 331—not 337.

If 035 is treated as octal, 035 = 29 (base 10), 366 – 29 = 337 (as reported by the script output quoted above).

The leading zeros can be removed by using a "-" in the output specification for the date command. For instance:

```
$ date -d '01/01/2010' +%-j
1
```

2) I would suggest an alternate approach to the entire script. Please ignore this if you have considered and rejected this approach for some reason. I saw this article alone and no prior articles in the series.

Use the seconds-since-epoch output format. For instance:

```
#!/bin/bash
# daysbetween.bash

dateEnd=${1}
dateStart=${2}
echo ${dateEnd} through ${dateStart}

epochSecondsEnd=$( date -d "${dateEnd}" +%s )
epochSecondsStart=$( date -d "${dateStart}" +%s )
let elapsedSeconds=${epochSecondsEnd}-${epochSecondsStart}
echo elapsed seconds: ${elapsedSeconds}
let elapsedDays=${elapsedSeconds}/60/60/24
echo elapsed days: ${elapsedDays}
let doubleCheck=${elapsedSeconds}-${elapsedDays}*60*60*24
if [ "${doubleCheck}x" != "0x" ] ; then
  echo "double check failed: ${doubleCheck}"
fi
```

The +%s format gives a positive value for dates after Jan 1, 1970, and a negative value for dates prior to Jan 1, 1970.

Using this approach, there is no need to calculate any "from beginning" or "to end" of year blocks nor worry about leap years.

However, there is an issue. In a normal day, there are 24*60*60 seconds. The date command reports that March 9, 2014, has only 23 hours. From my shell:

```
$ ./daysbetween.bash 2014-03-10 2014-03-09
2014-03-10 through 2014-03-09
elapsed seconds: 82800
elapsed days: 0
double check failed: 82800
$ date --version
date (GNU coreutils) 8.23
...
```

A 24-hour day contains: 24*60*60 = 86400 seconds. A 23-hour day contains: 23*60*60 = 82800 seconds.

As you can see from the output, the

date command reports that March 9 had 82800 seconds, or 23 hours.

I do not know if this is a bug in the date command, or if it's intentional because of some date-adjusting convention (such as an hour equivalent of a leap year), or something else.

All other tests I performed raised no errors.

*Addendum to the seconds-since-epoch approach:*

The 23-hour day corresponds with daylight-savings time. "Spring forward" gives a 23-hour day, and "fall back" gives a 25-hour day.

This problem can be avoided by switching to GMT. A modified script is presented below:

```
#!/bin/bash
# daysbetween.bash

dateEnd=${1}
dateStart=${2}
echo ${dateStart} through ${dateEnd}

epochSecondsEnd=$( date -d "TZ=\"GMT\" ${dateEnd}" +%s )
echo "${epochSecondsEnd}"
read epochSecondsStart startTZ <
 ➥<(date -d "TZ=\"GMT\" ${dateStart}" +%s)
echo "${epochSecondsStart}"

let elapsedSeconds=${epochSecondsEnd}-${epochSecondsStart}
echo elapsed seconds: ${elapsedSeconds}

let elapsedDays=${elapsedSeconds}/60/60/24
echo elapsed days: ${elapsedDays}

let doubleCheck=${elapsedSeconds}-${elapsedDays}*24*60*60
if [ "${doubleCheck}x" != "0x" ] ; then
  echo "double check failed: ${doubleCheck}" >&2
fi
```

**—Chris**

## Doc Searls' "Stuff That Matters"

Doc Searls' article in the September 2014 issue about privacy on the Net and targeting by government's security agencies is interesting, but it seems to be a bit naïve, especially compared to what the author actually writes about, giving the examples of Israel and London's effective security systems.

The author doesn't seem to understand that such privacy protector systems like TOR can be used not only to protect the privacy of normal and perfectly honest citizens but also by terrorist and criminal organizations around the world, so I guess that's

perfectly understandable that a security agency would be interested in monitoring and tracking them and their users. At least, if someone has nothing to hide, he's not in danger; security operators are experienced enough to understand whether they are tracking an employee who protects his privacy or a terrorist who wants to blow up a building.
**—Walter**

***Doc Searls replies:*** *I think you missed my point, which is that we are in the earliest days of personal privacy technology development in the on-line world. To get to that point, I borrowed interest in actual attacks going on in the real world at the time, including rockets pointed at my head in Israel and the NSA flagging* Linux Journal *readers as terrorism suspects.*

*If we all had "nothing to hide", we wouldn't wear clothing. And really, how many of us trust the world's "security operators" to protect our privacy? The ones at the NSA sure failed in our own case.*

## WRITE *LJ* A LETTER

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

**PHOTO OF THE MONTH**
Remember, send your Linux-related photos to ljeditor@linuxjournal.com!

LINUX JOURNAL

## At Your Service

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

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

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

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

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

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

# Bye-bye bottlenecks. Hello happy users.

New Relic APM™ provides deep visibility into your app's performance, from the end-user experience down to the line of code.

**Make your app faster today**
www.newrelic.com/apm

Companies using New Relic

Microsoft airbnb SONY EA NBC Tango GROUPON rdio zendesk intuit

New Relic® APM™

## diff -u
## WHAT'S NEW IN KERNEL DEVELOPMENT

Hardware errors are tough to code for. In some cases, they're impossible to code for. A particular brand of hardware error is the **Machine-Check Exception** (MCE), which means a CPU has a problem. On **Windows** systems, it's one of the causes of the Blue Screen of Death.

Everyone wants to handle hardware errors well, because it can mean the difference between getting a little indication of what actually went wrong and getting no information at all.

**Andy Lutomirski** recently suggested some code to clean up **non-maskable interrupts** (NMIs), which also typically indicate some sort of hardware failure. But over the course of discussion, folks raised questions about how to handle various cases—for example, when an MCE came immediately after an NMI. Typically NMIs are not interruptable by any other code, but should an exception be made for MCEs? If the OS detects a CPU error while already processing another hardware error, should it defer to the more pressing CPU issue or not?

There was a bit of debate, but ultimately **Linus Torvalds** said that an MCE meant that the system was dead. Any attempt to handle that in software, he said, was just in order to crash as gracefully as possible. But he felt that the kernel should not make any complicated effort in that case, since the end result would just be the same crash. Deadlocks, race conditions and other issues that normally would be important, simply weren't in this case. Make a best effort to log the event, he said, and forget the rest.

Elsewhere, he elaborated more vociferously, saying, "MCE is frankly misdesigned. It's a piece of shit, and any of the hardware designers that claim that what they do is for system stability are out to lunch. This is a prime example of what *not* to do, and how you can actually spread what was potentially a localized and recoverable error, and make it global and unrecoverable." And he added:

> Synchronous MCEs are fine for synchronous errors, but then trying to turn them "synchronous" for other CPUs (where they *weren't* synchronous errors) is a major mistake. External errors punching through irq context is wrong, punching through NMI is just inexcusable.
>
> If the OS then decides to take down the whole machine, the OS—not the hardware—can choose to do something that will punch through other CPUs' NMI blocking (notably, init/reset), but the hardware doing this on its own is just broken if true.

**Tony Luck** pointed out that **Intel** actually was planning to fix this in future machines, although he acknowledged that turn-around time for chips was likely to be very long. However, as **Borislav Petkov** pointed out, even after the fix went in, Linux still would need to support the bad hardware.

The tightrope-walk of **container security** has some controversy. One group believes that containers should be able to do whatever an independent system could do. Another group believes that certain abilities render the container inherently insecure. The first group says that without these features, the container isn't truly offering a complete environment. The second group says that's how the cookie crumbles.

**Seth Forshee** recently posted some patches to allow containerized systems to see **hot-plugged devices**, just the way a non-containerized system could. But this, apparently, was a bridge too far. **Greg Kroah-Hartman** said he had long since expressed a clear policy against adding namespaces to devices. And, that was exactly how Seth's code made the hot-plugged devices visible to the containerized system.

It turns out that there are valid use-cases for wanting a containerized system to be able to see hot-plugged devices. **Michael H. Warfield** described one such. And, Seth described his own—he needed hot-plug support in order to implement **loopback devices** within the container.

Greg said loopback support in a container was a very bad idea, since it provided all sorts of opportunities to leak data out of the container and into the host system—a security violation.

He said this was not a "normal" use-case for containers. To which **Serge Hallyn** replied that any feature used by a non-containerized system was a "normal" use case for containerized systems.

Serge argued that these features inevitably would go into containers. There was no way to keep them out. As long as containers excluded features that were included in non-containerized systems, there would be people with an incentive to bridge the gap. Why not bridge it now and fix the bugs as they showed up?

But Richard said, "There are so many things that can hurt you badly. With user namespaces, we expose a really big attack surface to regular users. [...] I agree that user namespaces are the way to go, all the papering with LSM over security issues is much worse. But we have to make sure that we don't add too many features too fast."

And, Greg added that Seth's code was too hacky, implementing just what Seth needed, rather than addressing the overarching issue of how to handle namespaces properly within a container.

Greg also said he supported loopback devices within containers, but he and **James Bottomley** said that the security issues were real, and the implementation had to take account of them. It wasn't enough simply to implement the feature and then fix bugs. The feature needed a proper design that addressed these concerns.—**ZACK BROWN**

## They Said It

**Everything that is really great and inspiring is created by the individual who can labor in freedom.**
***—Albert Einstein***

**Any government is potentially the worst client in the world you could ever possibly want to have.**
***—Thomas Heatherwick***

**When you give each other everything, it becomes an even trade. Each wins all.**
***—Lois McMaster Bujold***

**Enjoyment is not a goal, it is a feeling that accompanies important ongoing activity.**
***—Paul Goodman***

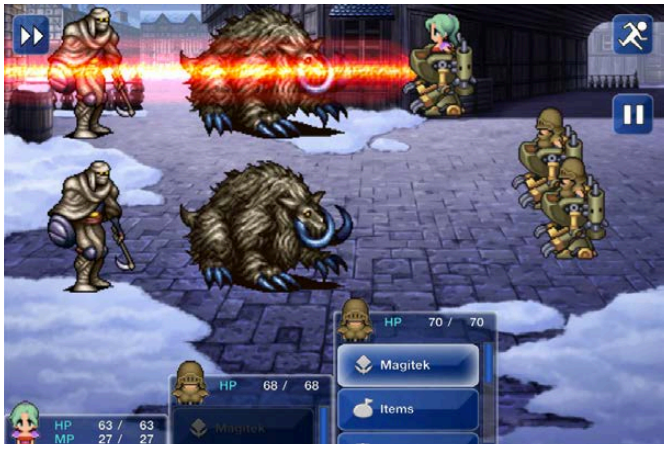
**I must create a system, or be enslaved by another man's.**
***—William Blake***

# Android Candy: Party Like It's 1994!

I really stink at video games. I write about gaming occasionally, but the truth of the matter is, I'm just not very good. If we play *Quake*, you'll frag me just about as often as I respawn. I don't have great reflexes, and my coordination is horrible. But if you give me an RPG, a 12-pack of Coke, and a three-day weekend, I'll be a level 96 blood elf by dawn of the second day.

Yes, in my youth I was a bit of a nerd. I stayed home weekends playing *Chrono-trigger*, *The Secret of Mana*, *Zelda*, *Dragon Warrior* and, of course, *Final Fantasy*. I was happy to discover the other day that those same *Final Fantasy* games I loved as a youngster are available in all their remade glory on the Android platform! They are unfortunately a little pricey, with each installment weighing in at $15.99, but they've been re-created specifically for the touch screen, and they are really fun!

If you wonder which game to buy (and you don't plan to buy them all, like some of us did), I highly recommend *Final Fantasy VI*.



It was the best game on the Super Nintendo, and I think it's the best game on Android as well. Of course, if you're okay with slightly more awkward gameplay, the old titles are easy to find in ROM format in the questionable corners of the Internet. There are several really good SNES emulators for Android that will allow you to play those original ROM files completely free. Honestly, however, if you can afford the $15.99, the remakes are far more enjoyable to play.

Check them out on the Google Play Store: **https://play.google.com/store/apps/developer?id=SQUARE%20ENIX%20Co.%2CLtd.&hl=en**.

**—SHAWN POWERS**

XAMPP Apache + MySQL + PHP + Perl

## Non-Linux FOSS

One of my career iterations put me in charge of a Windows server that had Apache and PHP installed on it to serve as a Web server for the corporate intranet. Although I was happy to see Apache used as the Web server dæmon, the installation on the Windows server was the most confusing and horrifying mess I've ever seen. To this day, I'm not sure which of the three Apache instances was actually serving files, and there were at least six PHP folders in various places on the hard drive, each with a different version number.

If you're in a situation where you're required to use Windows, but don't want to worry about the nightmare of installing Apache and PHP (much less MySQL) on your machine, I urge you to check out XAMMP. It's not a new program, but that's one of its greatest features. It's basically just a single installer for Windows, OS X or Linux that installs Apache with PHP and MySQL. Its maturity means that even on a Windows system, it should install and work like you'd expect open-source software to work.

Although XAMMP can be used to serve files to the actual Internet, it was designed for individuals to install on their own workstations to test their code. And in that situation, it works really well. If you have a server connected to the Internet, I still recommend using a Linux server with a proper Apache/PHP installation, but if you're stuck using a Windows workstation, XAMMP can give you a stable, open-source Web server platform that you can rely on. Grab a copy at **http://www.apachefriends.org**.

**—SHAWN POWERS**

drupalize.me

## Instant Access to Premium Online Drupal Training

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

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

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

```
 *
 * Usage: $(obj).insertAtCursor(text);
 *   obj = a textarea or textfield
 *   text = a string to insert
 */
$.fn.extend({
  insertAtCaret: function(myValue){
    return $(this).each(function(){
      // If target element is hidden, don't do it.
      if ($(this).is(':hidden') || $(this).parents(':hidde
        return;
      }
      if (document.selection) {
        this.fo
        sel = c
        sel.tex
        this.fo
      }
      else if (
        var sta
        var end
        var scr
        this.va
        this.fo
        // High
```

# Drafting on Linux

One common scientific task is designing new hardware to help make measurements. A powerful tool to help with this design work is a Computer Aided Design system, or CAD software. Several tools are available on Linux for doing CAD work. In this article, I take a look at LibreCAD (**http://www.librecad.org**).

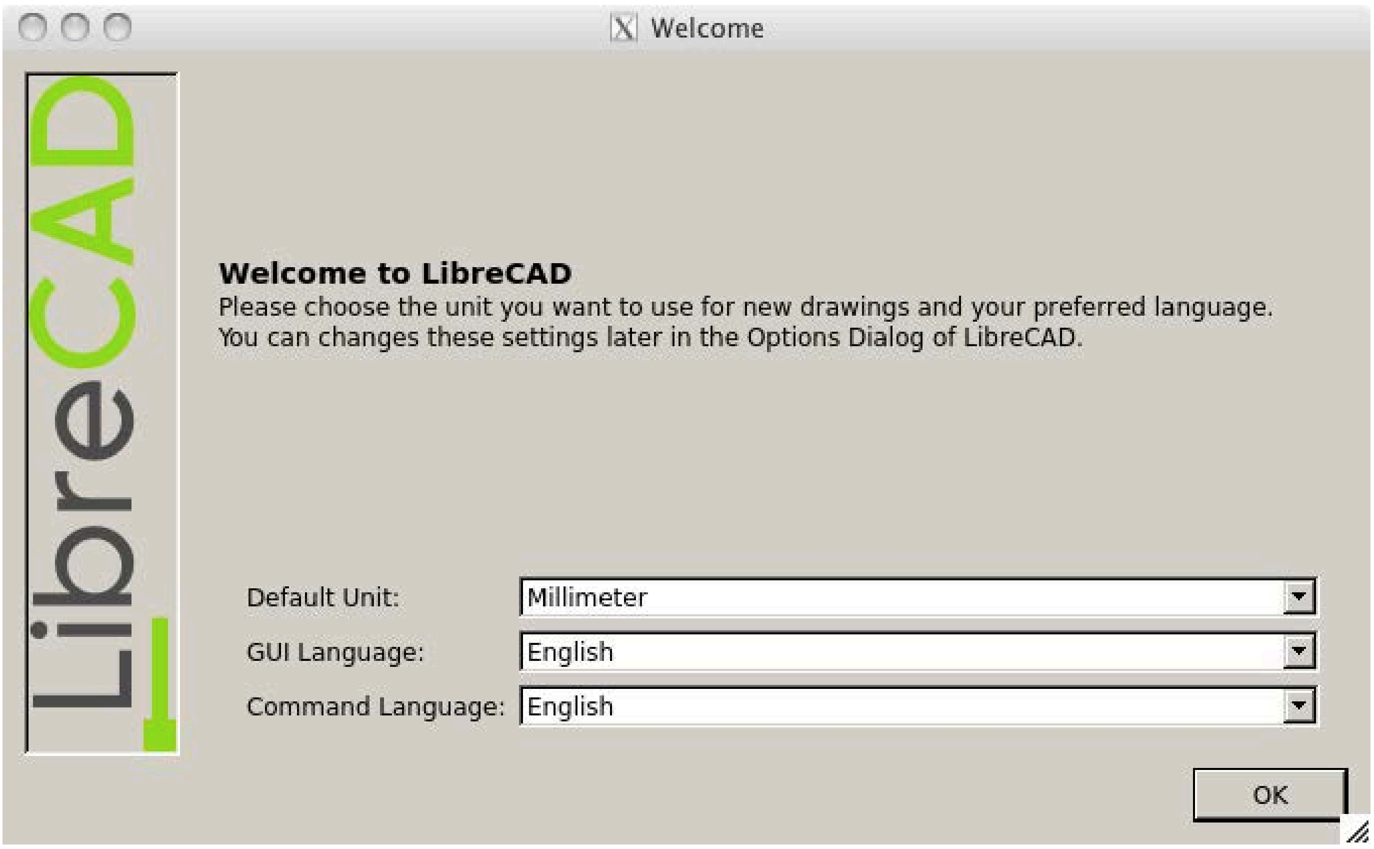
LibreCAD started as an extension of QCad. For a short while, it was called CADuntu, before finally being named LibreCAD. It should be available as a package in most distributions.

In Debian-based distributions, you can install it with the command:

```
sudo apt-get install librecad
```

And, you always can install it from source if you want the latest and greatest features.

Once LibreCAD is installed, you can launch it from the application launcher for your desktop, or you can run the `librecad` command



**Welcome to LibreCAD**
Please choose the unit you want to use for new drawings and your preferred language.
You can changes these settings later in the Options Dialog of LibreCAD.

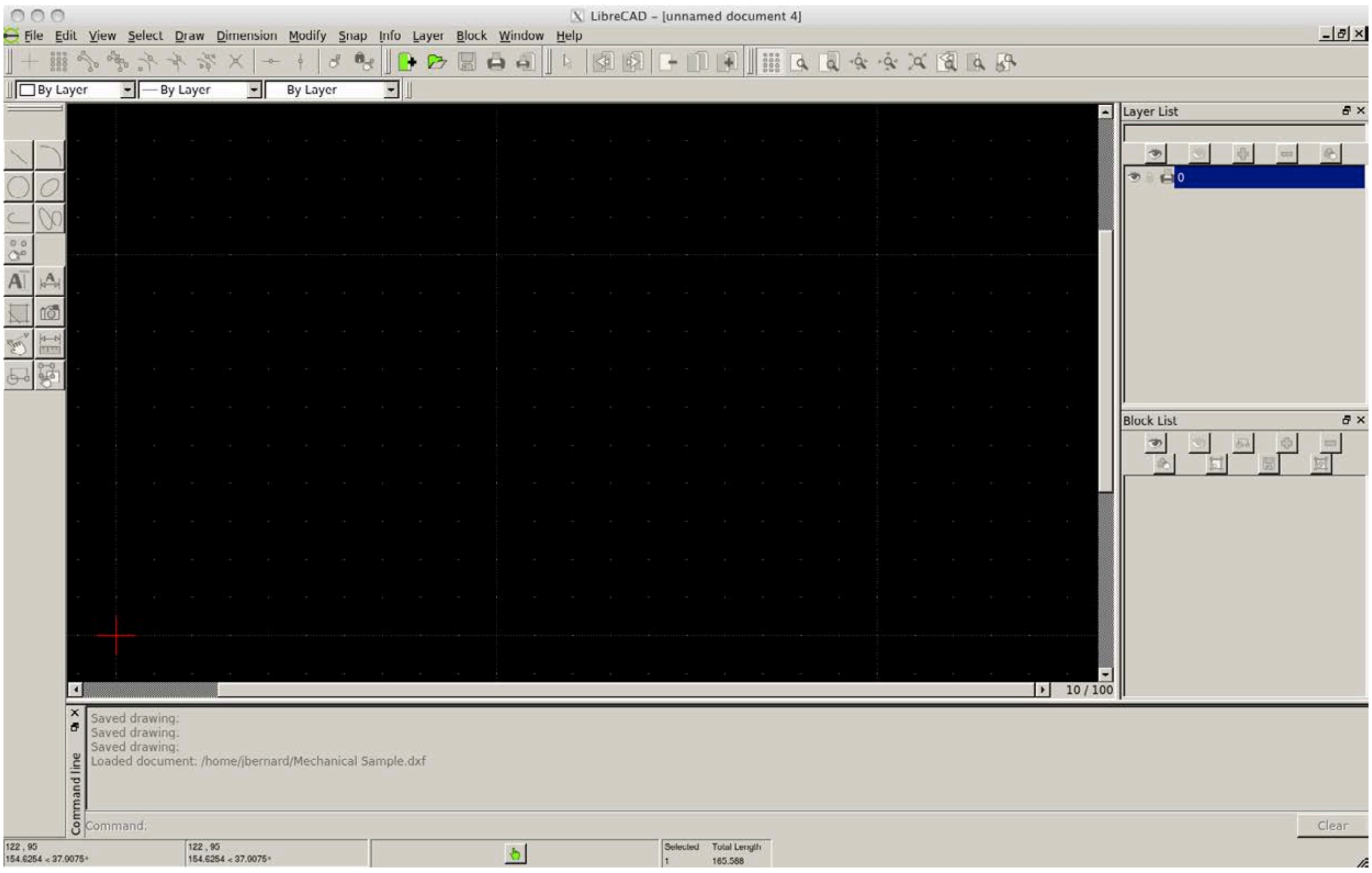
Default Unit: Millimeter
GUI Language: English
Command Language: English

**Figure 1. When you start LibreCAD the first time, you need to set some initial options.**

from a terminal. The first time you start LibreCAD, you will be greeted with a welcome window (Figure 1).

Here, you will be presented with the ability to set options for the default unit, the GUI language and the command language. Once you set those options, you will see a blank canvas where you can start your new project (Figure 2). The main window is the actual drawing canvas where you can set up your design. On the left-hand side, you should see a palette of drawing tools. On the right-hand side, you will see two smaller windows containing the layer list and the block list for your design.

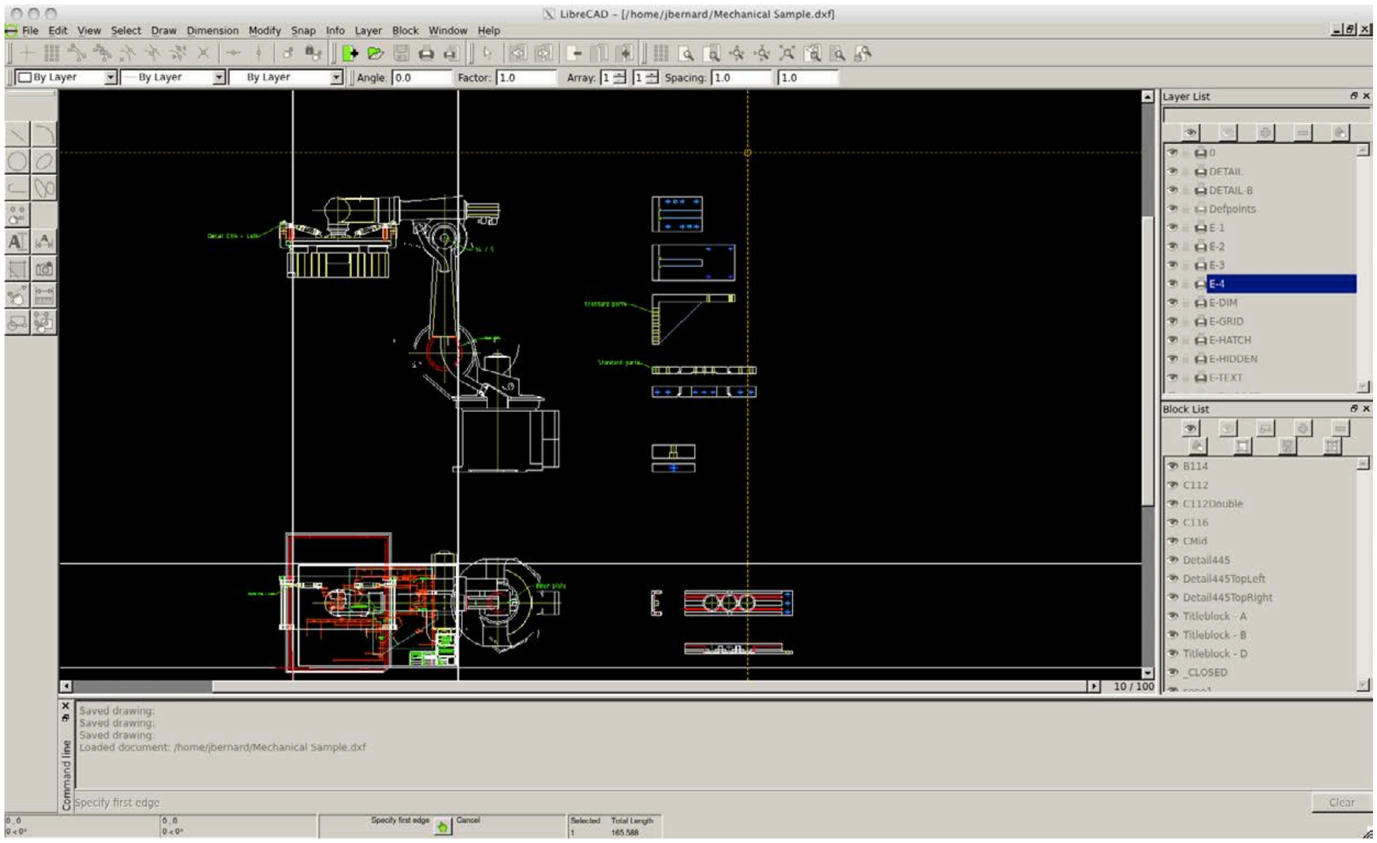
If you already have done some design work, you can import that work into LibreCAD. You can insert an image to your design by clicking the menu item File→Import→Insert Image. LibreCAD can handle most common file formats. If you had been working with another CAD program and have a DXF file of that work, you can import it by clicking on the menu item File→Import→Block (Figure 3). This option also handles CXF files, in case


LibreCAD - [unnamed document 4]
File Edit View Select Draw Dimension Modify Snap Info Layer Block Window Help
By Layer
By Layer
By Layer
Layer List
Block List
10 / 100
Command line
Clear

```
Saved drawing:
Saved drawing:
Saved drawing:
Loaded document: /home/jbernard/Mechanical Sample.dxf
Command:
```

```
122 , 95                   122 , 95                   Selected   Total Length
154.6254 < 37.0075°        154.6254 < 37.0075°        1          165.588
```

Figure 2. LibreCAD starts up with a blank canvas, ready for your new project.


LibreCAD - [/home/jbernard/Mechanical Sample.dxf]
File Edit View Select Draw Dimension Modify Snap Info Layer Block Window Help
By Layer
By Layer
By Layer
Angle: 0.0
Factor: 1.0
Array: 1 1
Spacing: 1.0 1.0
Layer List
Block List
Command line
Clear

Layer List: 0, DETAIL, DETAIL-B, Defpoints, E-1, E-2, E-3, E-4, E-DIM, E-GRID, E-HATCH, E-HIDDEN, E-TEXT

Block List: B114, C112, C112Double, C116, CMid, Detail445, Detail445TopLeft, Detail445TopRight, Titleblock - A, Titleblock - B, Titleblock - D, _CLOSED

```
Saved drawing:
Saved drawing:
Saved drawing:
Loaded document: /home/jbernard/Mechanical Sample.dxf
Specify first edge
```

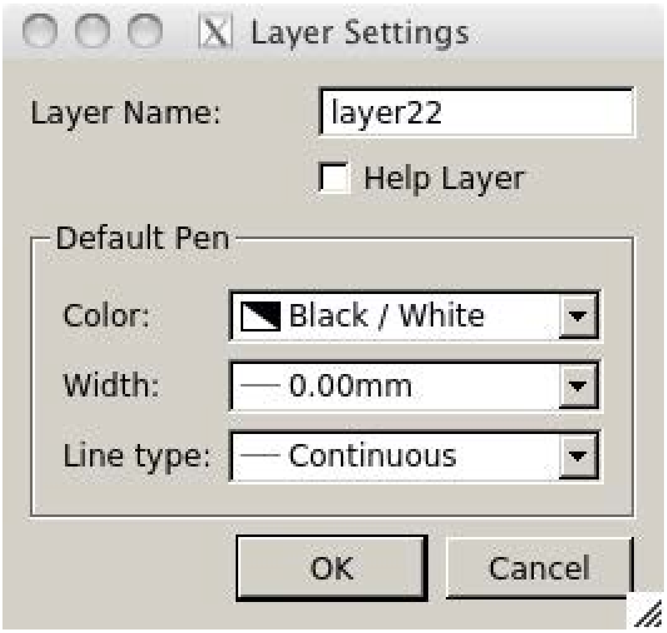
```
0 , 0        0 , 0        Specify first edge   Cancel   Selected   Total Length
0 < 0°       0 < 0°                                     1          165.588
```

Figure 3. You can import DXF files from lots of places.

you were using those.

You may have a text file with raw point data for the object you are trying to draw. If so, you can click on the menu item File→Import→Read ascii points. This will pop up an option window where you can define what the points represent and how to treat them. You even can import GIS data from a shape file with the menu item File→Import→shape file.

Now you should be ready to start designing your project. Clicking the icons in the palette on the left-hand side opens a new palette with



Layer Settings

Layer Name: layer22

☐ Help Layer

Default Pen

Color: Black / White

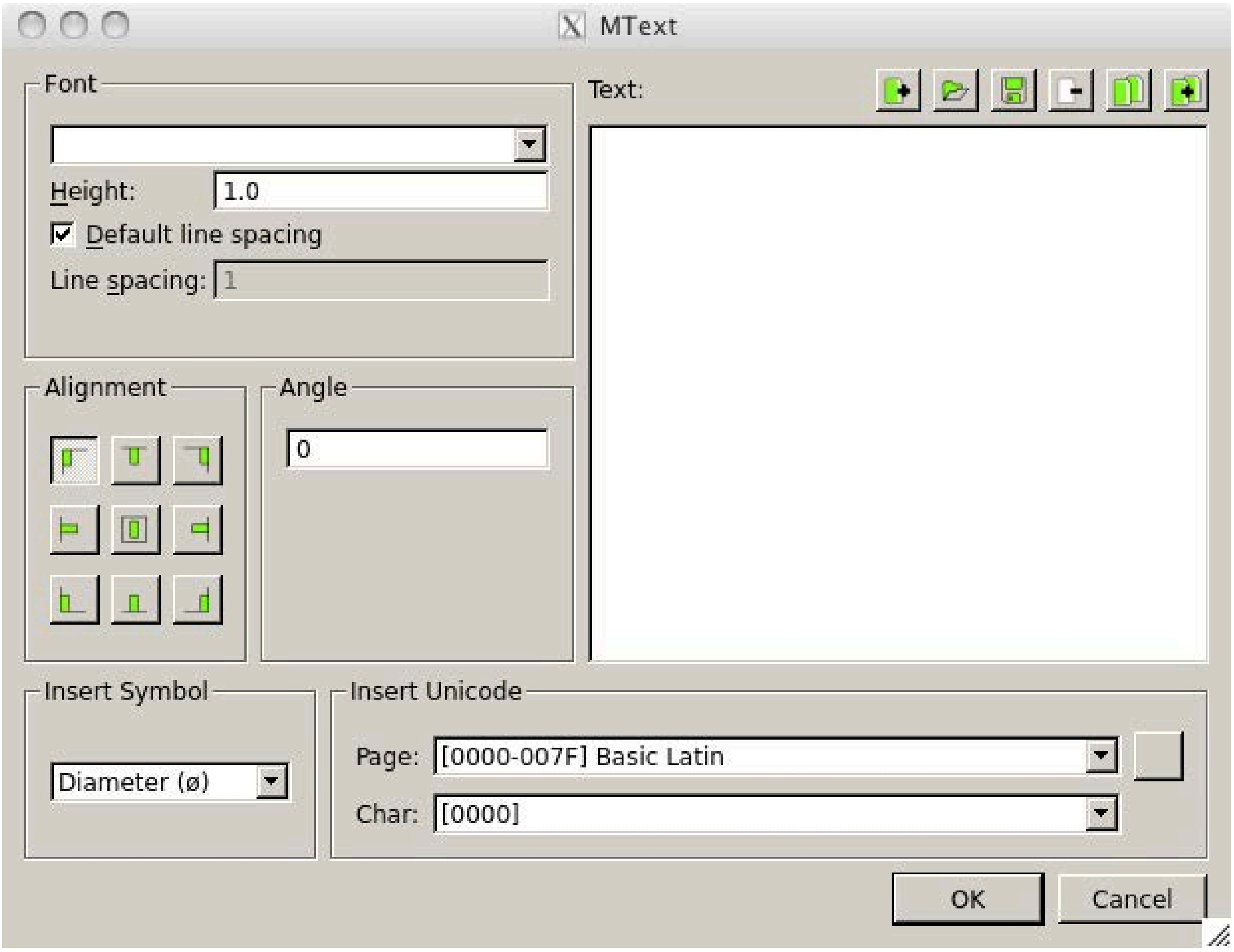
Width: 0.00mm

Line type: Continuous

Figure 4. You can set options for new layers added to your project.

multiple options for each of the categories. For example, if you click on the circle icon, you will see a new palette giving you the option to draw circles with either two points on the circumference, a point at the center and one at the circumference or a circle that fits within a triangle, among many other options.

The other icons in the drawing palette also provide tools for drawing many other components, such as lines, arcs and splines. All of these items are drawn on the default layer that you get with a new project. You can add a new layer by clicking the plus icon in the top pane on the right-hand side. This will pop up a new option window where you can set things like the layer name and the drawing color for the new layer (Figure 4).



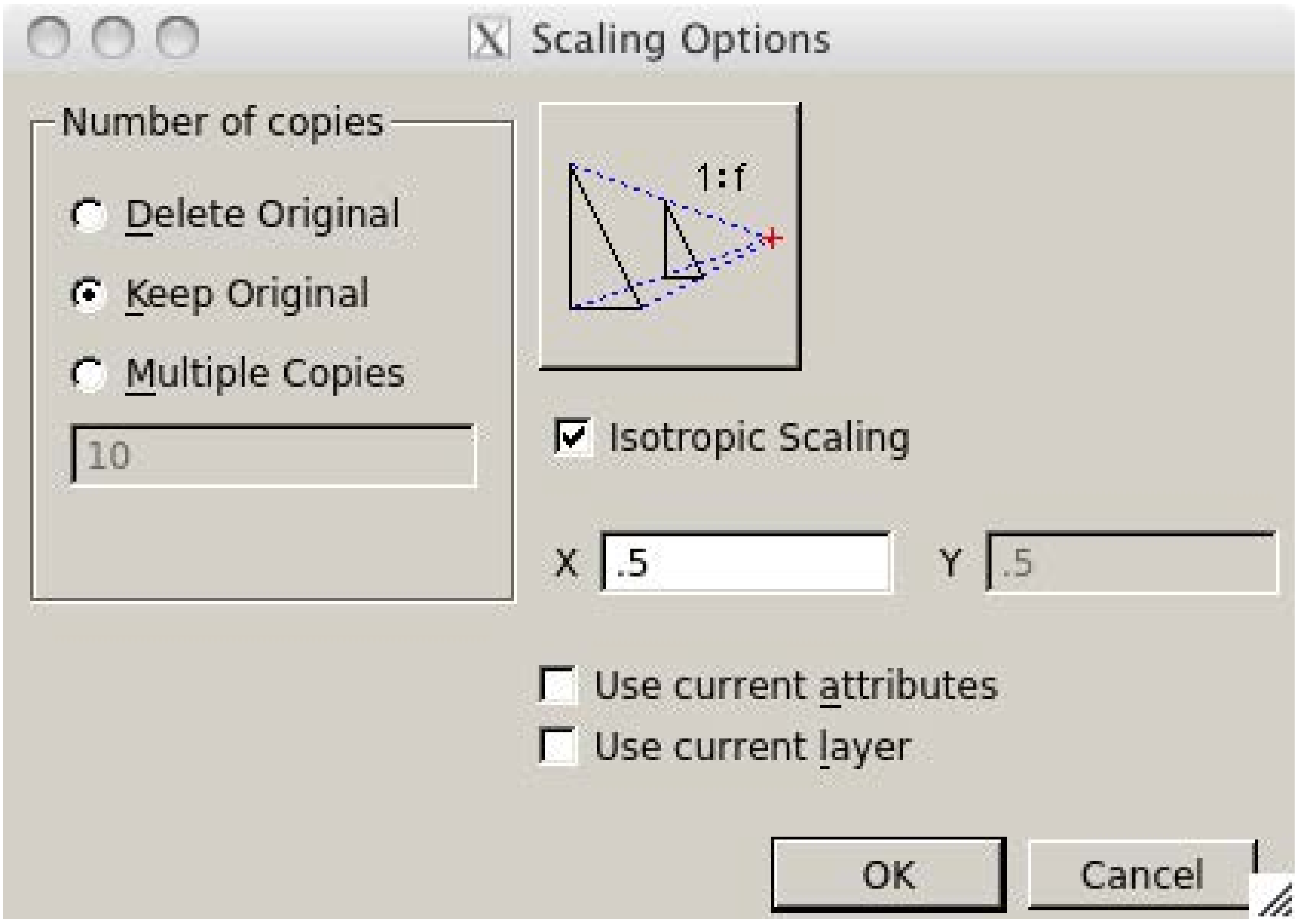
Height: 1.0
☑ Default line spacing
Line spacing: 1
Angle: 0
Insert Symbol: Diameter (ø)
Page: [0000-007F] Basic Latin
Char: [0000]

Figure 5. You can set several options when you add a multi-line text object.

You can toggle visibility of the various layers by clicking the eye icon to the right in the layer list. When you have a layer set the way you want it, you can make it uneditable by clicking on the lock icon for that layer. That way, you won't accidentally change it while you work on other layers.

If you need to add labels explaining parts of your design, you can click on the multi-line text option in the tool palette. This will pop up a window where you can enter the text and set options like font and color (Figure 5).

Once you have the basic objects drawn on your project, you can use the many modification tools available to fine-tune your drawing or to generate more complex objects based on some modification of one of the basic types. These modifications are available under the Modify menu item. You can do things like scaling, mirroring,


Scaling Options

Number of copies
- Delete Original
- [✓ Keep Original]
- Multiple Copies
- 10

[✓ Isotropic Scaling]

X .5  Y .5

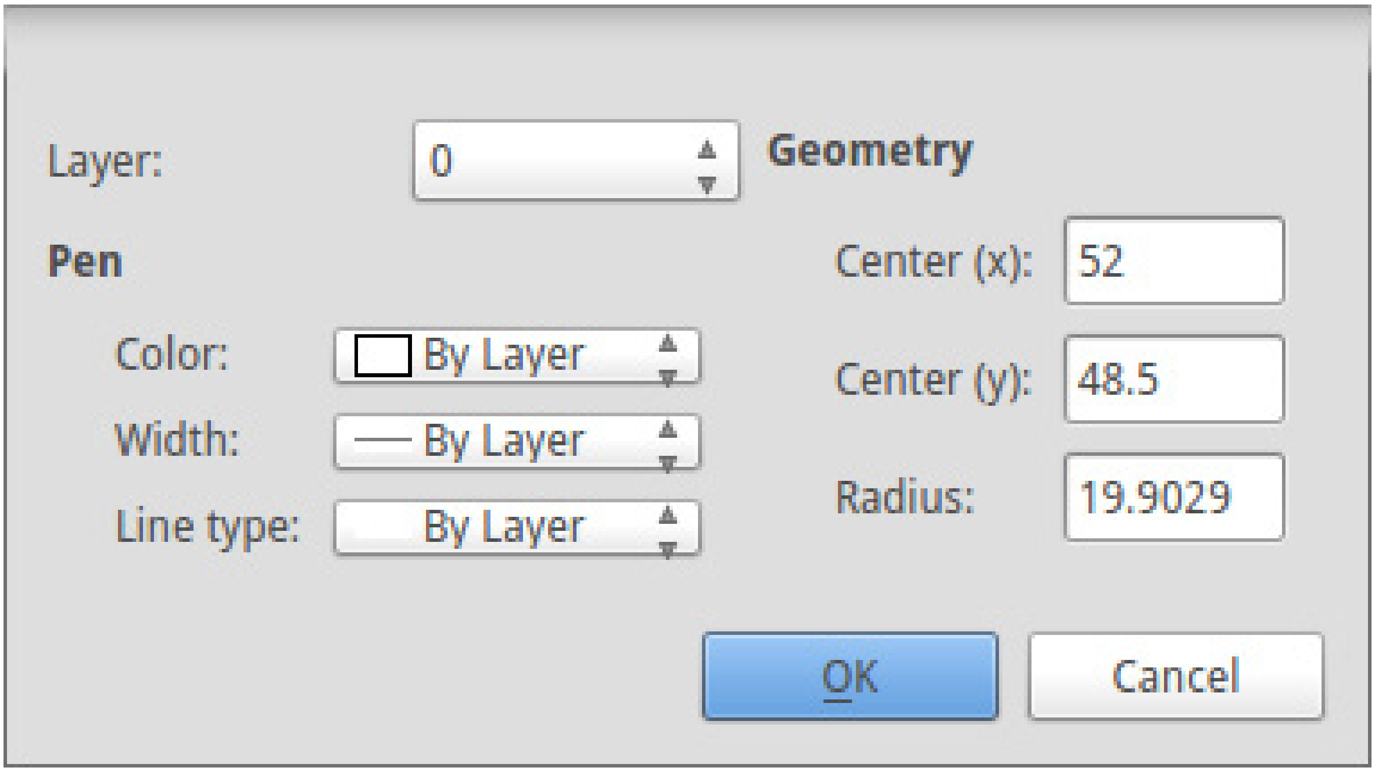
☐ Use current attributes
☐ Use current layer

Figure 6. You can set several options for scaling an element of your drawing.

rotating and moving.

Using these tools, however, isn't very intuitive. Say you want to scale an element of your drawing. The first thing you will want to do is to click on the Modify→Scale menu item. You next will notice that the command-line box at the bottom of the window has changed, asking you to "Select to scale". You then need to click on the element you want to scale, say a line element, and press the Enter key. The command-line window then will change to saying "Specify reference point". LibreCAD scales based on a reference point to act as a point of perspective, so go ahead and click on a point. This will pop up an option window where you can set things like the scaling factor and whether it is isotropic (Figure 6). When you click OK, LibreCAD will apply the modification and refresh the drawing.

You also can change the properties of the different elements, which may be needed to clarify parts of special interest. To do this, you need to click on the Modify→Properties menu item, and then click on the element in question. This will pop up a dialog box where you can edit properties that



Layer: 0

Color: By Layer

Width: By Layer

Line type: By Layer

Center (x): 52

Center (y): 48.5

Radius: 19.9029

Figure 7. You can change both the display properties of your circle as well as the physical properties.

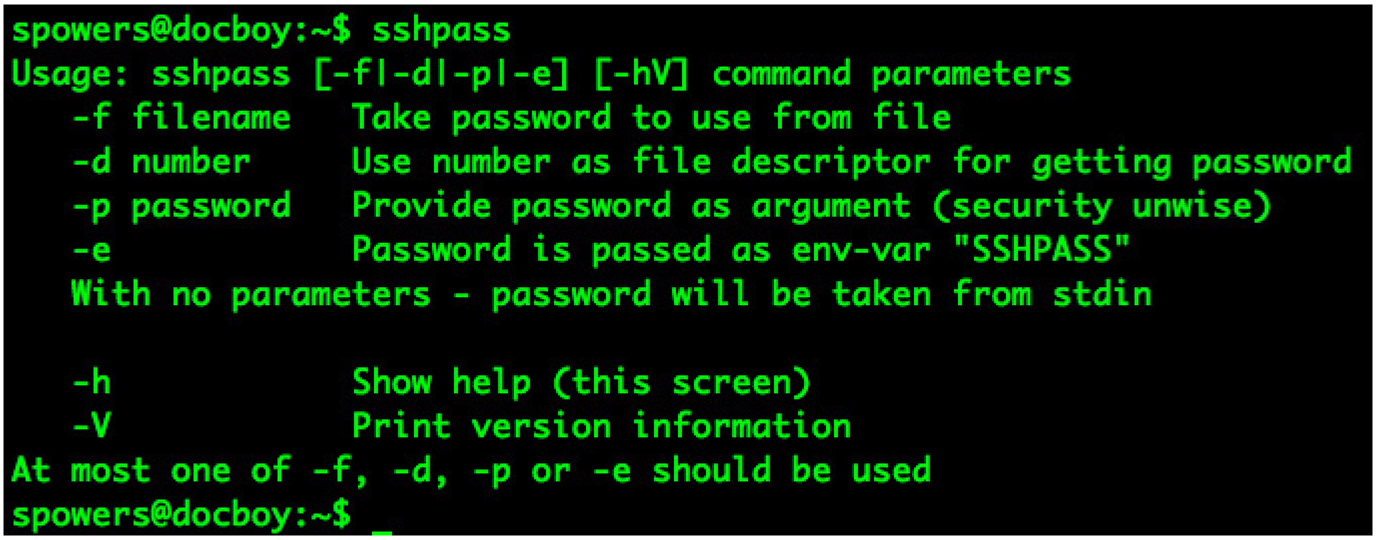
apply for that element type (Figure 7).

When you have finished your design, you will want to share it with others. The default file format is the Drawing Exchange Format (.dxf). LibreCAD supports versions 2007, 2004, 2000, R14 and R12. If you need to, you also can save it as an LFF Font file (.lff), a QCad file (.cxf) or a Jww Drawing file (.jww). If you just want a simplified output, you can click on the File→Export menu item and save it in one of a large number of image file formats. With these options, you should be able to share your design with most people.

Hopefully, this article has shown you enough to help you decide whether LibreCAD might be a good fit for your next design project. If so, you can find even more information on the LibreCAD Wiki and forum. A great deal of examples are available on the Internet that will show you just what is possible with a good CAD system. And, because these examples are available in DXF files, you can load them in LibreCAD and play with the possibilities.

**—JOEY BERNARD**

# The Awesome Program You Never Should Use



```
spowers@docboy:~$ sshpass
Usage: sshpass [-f|-d|-p|-e] [-hV] command parameters
   -f filename   Take password to use from file
   -d number     Use number as file descriptor for getting password
   -p password   Provide password as argument (security unwise)
   -e            Password is passed as env-var "SSHPASS"
   With no parameters - password will be taken from stdin

   -h            Show help (this screen)
   -V            Print version information
At most one of -f, -d, -p or -e should be used
spowers@docboy:~$ _
```

I've been hesitating for a couple months about whether to mention sshpass. Conceptually, it's a horrible, horrible program. It basically allows you to enter an SSH user name and password on the command line, so you can create a connection without any interaction. A far better way to accomplish that is with public/private keypairs. But it's still something I find useful from time to time, and I'd rather mention it with all the warnings in the world than to pretend it doesn't exist.

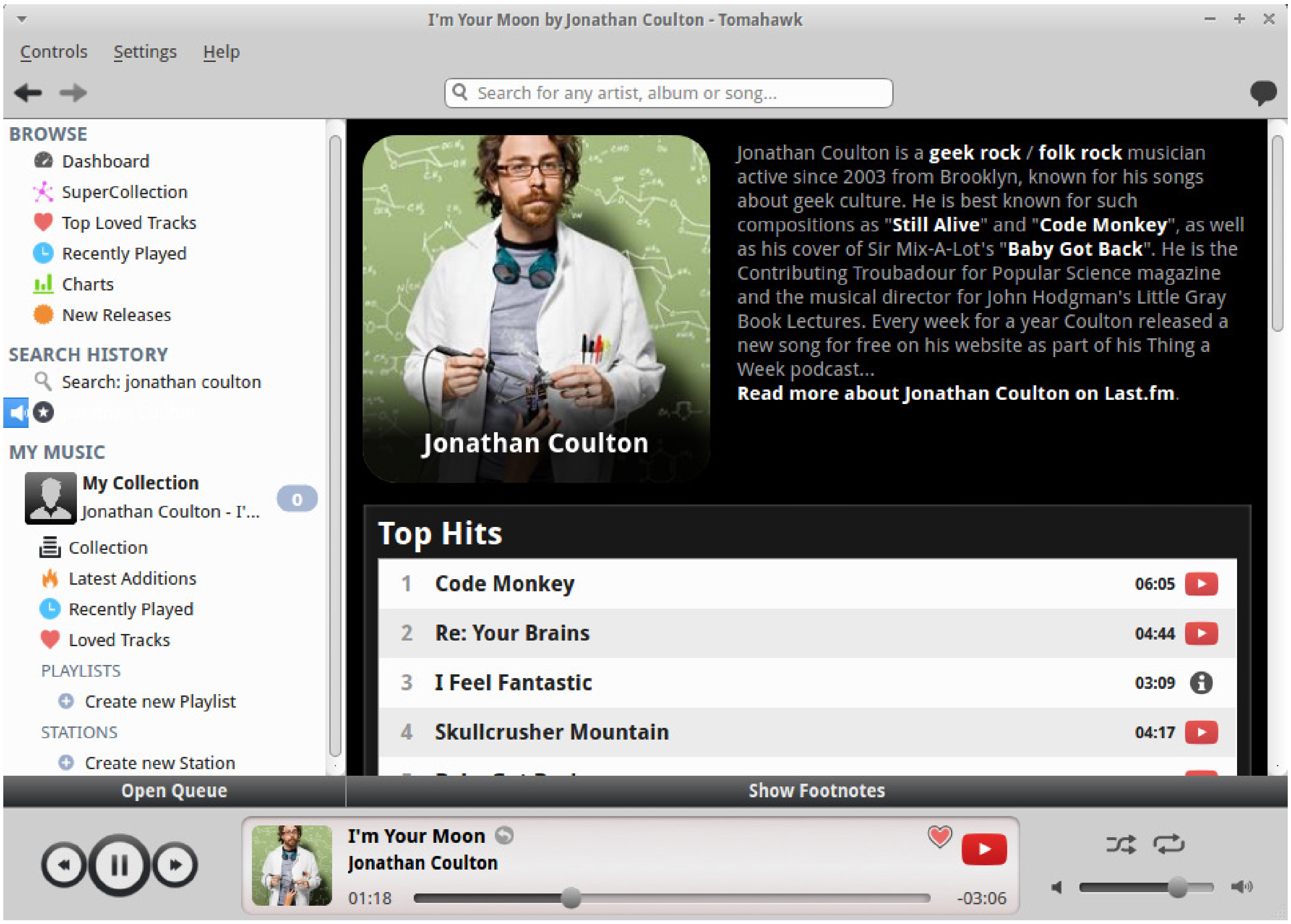
So, sshpass—it's a simple tool, but in a pinch, it can be incredibly helpful. You use it with the user name and password as command-line arguments (with some variations, see the help screen in the screenshot), and it injects them into your `ssh` (or `scp`) command.

Again, this is a horribly insecure method for entering passwords. However, I find it particularly useful for setting up new machines, especially computers or devices in a closed environment. I've also used it to send files via `scp` to hundreds of machines in my local network that I'll never need to connect to again. It's a dangerous tool, but can be a lifesaver if you need it. Search your distribution's repositories, as it's available for most systems. And remember, don't ever use it!

**—SHAWN POWERS**

# Tomahawk, the World Is Your Music Collection

I don't listen to music very often, but when I do, my tastes tend to be across the board. That's one of the reasons I really like Pandora, because the music selection is incredible (in fact, I can't recommend the Pithos client for Pandora enough—I've written about it in past issues). Unfortunately, with Pandora, you don't get to pick specific songs. That's usually okay for me, but sometimes I want to hear a particular song by a particular artist. Even worse, sometimes



Jonathan Coulton is a **geek rock** / **folk rock** musician active since 2003 from Brooklyn, known for his songs about geek culture. He is best known for such compositions as "**Still Alive**" and "**Code Monkey**", as well as his cover of Sir Mix-A-Lot's "**Baby Got Back**". He is the Contributing Troubadour for Popular Science magazine and the musical director for John Hodgman's Little Gray Book Lectures. Every week for a year Coulton released a new song for free on his website as part of his Thing a Week podcast...
**Read more about Jonathan Coulton on Last.fm**.

Top Hits

| | | |
|---|---|---|
| 1 | Code Monkey | 06:05 |
| 2 | Re: Your Brains | 04:44 |
| 3 | I Feel Fantastic | 03:09 |
| 4 | Skullcrusher Mountain | 04:17 |

I want to hear a particular version of a song. I've purchased 3–4 different versions of a song, only to discover none of them were what I wanted.

Enter Tomahawk. It behaves much like a traditional music application, and it will play music from your hard drive or network shares. Its real strength, however, is its ability to connect to on-line resources to find songs. When it finds those songs, it treats them just like a local file. You can create playlists with a mix of local and remote media, and search across an entire array of on-line services. Tomahawk will connect to Spotify, last.fm, Jamendo, Beets, Subsonic and tons of other sources. Of particular note, I love that there is a YouTube plugin that will search YouTube for songs! (The YouTube plugin isn't included by default, but it's free to install.)

Due to its ability to blur the lines between local and streaming media, while functioning as a traditional desktop music app, Tomahawk earns this month's Editors' Choice award. If you have fickle music tastes, or just want to listen to your various music collections in a central place, I urge you to give Tomahawk a try: http://www.tomahawk-player.org.

**—SHAWN POWERS**

REUVEN M. LERNER

# PostgreSQL, the NoSQL Database

**Thinking NoSQL? Believe it or not, PostgreSQL might be a great choice.**

**One of the most** interesting trends in the computer world during the past few years has been the rapid growth of NoSQL databases. The term may be accurate, in that NoSQL databases don't use SQL in order to store and retrieve data, but that's about where the commonalities end. NoSQL databases range from key-value stores to columnar databases to document databases to graph databases.

On the face of it, nothing sounds more natural or reasonable than a NoSQL database. The "impedance mismatch" between programming languages and databases, as it often is described, means that we generally must work in two different languages, and in two different paradigms. In our programs, we think and work with objects, which we carefully construct. And then we deconstruct those objects, turning them into two-dimensional tables in our database. The idea that I can manipulate objects in my database in the same way as I can in my program is attractive at many levels.

In some ways, this is the holy grail of databases: we want something that is rock-solid reliable, scalable to the large proportions that modern Web applications require and also convenient to us as programmers. One popular solution is an ORM (object-relational mapper), which allows us to write our programs using objects. The ORM then translates those objects and method calls into the appropriate SQL, which it passes along to the database. ORMs certainly make it more convenient to work with a relational database, at least when it comes to simple queries. And to no

**But ORMs have their problems as well, in no small part because they can shield us from the inner workings of our database.**

small degree, they also improve the readability of our code, in that we can stick with our objects, without having to use a combination of languages and paradigms.

But ORMs have their problems as well, in no small part because they can shield us from the inner workings of our database. NoSQL advocates say that their databases have solved these problems, allowing them to stay within a single language. Actually, this isn't entirely true. MongoDB has its own SQL-like query language, and CouchDB uses JavaScript. But there are adapters that do similar ORM-like translations for many NoSQL databases, allowing developers to stay within a single language and paradigm when developing.

The ultimate question, however, is whether the benefits of NoSQL databases outweigh their issues. I have largely come to the conclusion that, with the exception of key-value stores, the answer is "no"—that a relational database often is going to be a better solution. And by "better", I mean that relational databases are more reliable, and even more scalable, than many of their NoSQL cousins. Sure, you might need to work hard in order to get the scaling to work correctly, but there is no magic solution. In the past few months alone, I've gained several new clients who decided to move from NoSQL solutions to relational databases, and needed help with the architecture, development or optimization.

The thing is, even the most die-hard relational database fan will admit there are times when NoSQL data stores are convenient. With the growth of JSON in Web APIs, it would be nice to be able to store the result sets in a storage type that understands that format and allows me to search and retrieve from it. And even though key-value stores, such as Redis, are powerful and fast, there are sometimes cases when I'd like to have the key-value pairs connected to data in other relations (tables) in my database.

If this describes your dilemma, I have good news for you. As I write this, PostgreSQL, an amazing database and open-source project, is set to release version 9.4. This new version, like all other PostgreSQL versions, contains a number of optimizations, improvements and usability features. But two of the most intriguing features to me are HStore and JSONB, features that actually turn PostgreSQL into a NoSQL database.

Fine, perhaps I'm exaggerating a bit here. PostgreSQL was and always will be relational and transactional, and adding these new data types hasn't changed that. But having a key-value store within PostgreSQL opens many new possibilities for developers. JSONB, a binary version of JSON storage that supports indexing and a large number of operators, turns PostgreSQL into a document database, albeit one with a few other features in it besides.

In this article, I introduce these NoSQL features that are included in PostgreSQL 9.4, which likely will be released before this issue of *Linux Journal* gets to you. Although not every application needs these features, they can be useful—and with this latest release of PostgreSQL, the performance also is significantly improved.

## HStore

One of the most interesting new developments in PostgreSQL is that of HStore, which provides a key-value store within the PostgreSQL environment. Contrary to what I originally thought, this doesn't mean that PostgreSQL treats a particular table as a key-value store. Rather, HStore is a data type, akin to `INTEGER`, `TEXT` and `XML`. Thus, any column—or set of columns—within a table may be defined to be of type `HSTORE`. For example:

```
CREATE TABLE People (
    id   SERIAL,
    info HSTORE,
    PRIMARY KEY(id)
);
```

Once I have done that, I can ask PostgreSQL to show me the definition of the table:

```
\d people
              Table "public.people"
-----------------------------------------------------------------
| Column | Type    | Modifiers                                  |
-----------------------------------------------------------------
| id     | integer | not null default                           |
|        |         | ➥nextval('people_id_seq'::regclass)        |
-----------------------------------------------------------------
| info   | hstore  |                                            |
-----------------------------------------------------------------
 Indexes:
        "people_pkey" PRIMARY KEY, btree (id)
```

As you can see, the type of my "info" column is hstore. What I have effectively created is a (database) table of hash tables. Each row in the "people" table will have its own hash table, with any keys and values. It's typical in such a situation for every row to have the same key names, or at least some minimum number of overlapping key names, but you can, of course, use any keys and values you like.

Both the keys and the values in an HStore column are text strings. You can assign a hash table to an HStore column with the following syntax:

```
INSERT INTO people(info) VALUES ('foo=>1, bar=>abc, baz=>stuff');
```

Notice that although this example inserts three key-value pairs into the HStore column, they are stored together, converted automatically into an HStore, splitting the pairs where there is a comma, and each pair where there is a => sign.

So far, you won't see any difference between an HStore and a TEXT column, other than (perhaps) the fact that you cannot use text functions and operators on that column. For example, you cannot use the || operator, which normally concatenates text strings, on the HStore:

```
UPDATE People SET info = info || 'abc';
ERROR:  XX000: Unexpected end of string
LINE 1: UPDATE People SET info = info || 'abc';
                                              ^
```

PostgreSQL tries to apply the || operator to the HStore on the left, but cannot find a key-value pair in the string on the right, producing an error message. However, you can add a pair, which will work:

```
UPDATE People SET info = info || 'abc=>def';
```

As with all hash tables, HStore is designed for you to use the keys to retrieve the values. That is, each key exists only once in each HStore value, although values may be repeated. The only way to retrieve a value is via the key. You do this with the following syntax:

```
SELECT info->'bar' FROM People;
----------------
| ?column? |   |
----------------
| abc      |   |
----------------
(1 row)
```

Notice several things here. First, the name of the column remains without any quotes, just as you do when you're retrieving the full

**The big news in version 9.4 is that GiN and GIST indexes now support HStore columns, and that they do so with great efficiency and speed.**

contents of the column. Second, you put the name of the key after the -> arrow, which is different from the => ("hashrocket") arrow used to delineate key-value pairs within the HStore. Finally, the returned value always will be of type TEXT. This means if you say:

```
SELECT info->'foo' || 'a' FROM People;
----------------
| ?column? |   |
----------------
| 1a       |   |
----------------
(1 row)
```

Notice that ||, which works on text values, has done its job here. However, this also means that if you try to multiply your value, you will get an error message:

```
SELECT info->'foo' * 5 FROM People;
info->'foo' * 5 from people;
                    ^
Time: 5.041 ms
```

If you want to retrieve `info->'foo'` as an integer, you must cast that value:

```
SELECT (info->'foo')::integer * 5 from people;
----------------
| ?column? |   |
----------------
|   5      |   |
----------------
(1 row)
```

Now, why is HStore so exciting? In particular, if you're a database person who values normalization, you might be wondering why someone even would want this sort of data store, rather than a nicely normalized table or set of tables.

The answer, of course, is that there are many different uses for a database, and some of them can be more appropriate for an HStore. I never would suggest storing serious data in such a thing, but perhaps you want to keep track of user session information, without keeping it inside of a binary object.

Now, HStore is not new to PostgreSQL. The big news in version 9.4 is that GiN and GIST indexes now support HStore columns, and that they do so with great efficiency and speed.

Where do I plan to use HStore? To be honest, I'm not sure yet. I feel like this is a data type that I likely will want to use at some point, but for now, it's simply an extra useful, efficient tool that I can put in my programming toolbox. The fact that it is now extremely efficient, and its operators can take advantage of improved indexes, means that HStore is not only convenient, but speedy, as well.

## JSON and JSONB

It has long been possible to store JSON inside PostgreSQL. After all, JSON is just a textual representation of JavaScript objects ("JavaScript Object Notation"), which means that they are effectively strings. But of course, when you store data in PostgreSQL, you would like a bit more than that. You want to ensure that stored data is valid, as well as use PostgreSQL's operators to retrieve and work on that data.

PostgreSQL has had a JSON data type for several years. The data type started as a simple textual representation of JSON, which would check for valid contents, but not much more than that. The 9.3 release of PostgreSQL allowed you to use a larger number of operators on your JSON columns, making it possible to retrieve particular parts of the data with relative ease.

However, the storage and retrieval of JSON data was never that efficient, and the JSON-related operators were particularly bad on this front. So yes, you could look for a particular name or value within a JSON column, but it might take a while.

That has changed with 9.4, with the introduction of the JSONB data type, which stores JSON data in binary form, such that it is both more compact and more efficient than the textual form. Moreover, the same GIN and GIST indexes that now are able to work so well with HStore data also are able to work well, and quickly, with JSONB data. So you can search for and retrieve text from JSONB documents as easily (or more) as would have been the case with a document database, such as MongoDB.

I already have started to use JSONB in some of my work. For example, one of the projects I'm working on contacts a remote server via an API. The server returns its response in JSON, containing a large number of name-value pairs, some of them nested. (I should note that using a beta version of PostgreSQL, or any other infrastructural technology, is only a good idea if you first get the client's approval, and explain the risks and benefits.)

**Now, I'm a big fan of normalized data. And I'm not a huge fan of storing JSON in the database.**

Now, I'm a big fan of normalized data. And I'm not a huge fan of storing JSON in the database. But rather than start to guess what data I will and won't need in the future, I decided to store everything in a JSONB column for now. If and when I know precisely what I'll need, I will normalize the data to a greater degree.

Actually, that's not entirely true. I knew from the start that I would need two different values from the response I was receiving. But because I was storing the data in JSONB, I figured it would make sense for me simply to retrieve the data from the JSONB column.

Having stored the data there, I then could retrieve data from the JSON column:

```
SELECT id, email,
       personal_data->>'surname' AS surname
       personal_data->>'forename' as given_name
  FROM ID_Checks
 WHERE personal_data->>'surname' ilike '%lerner%';
```

Using the double-arrow operator (->>), I was able to retrieve the value of a JSON object by using its key. Note that if you use a single arrow (->), you'll get an object back, which is quite possibly not what you want. I've found that the text portion is really what interests me most of the time.

## Resources

Blog postings about improvements to PostgreSQL's GiN and GIST indexes, which affect the JSON and HStore types:

- **http://obartunov.livejournal.com/172503.html**
- **http://obartunov.livejournal.com/174887.html**
- **http://obartunov.livejournal.com/175235.html**

PostgreSQL documentation is at **http://postgresql.org/docs**, and it includes several sections for each of HStore and JSONB.

## Conclusion

People use NoSQL databases for several reasons. One is the impedance mismatch between objects and tables. But two other common reasons are performance and convenience. It turns out that modern versions of PostgreSQL offer excellent performance, thanks to improved data types and indexes. But they also offer a great deal of convenience, letting you set, retrieve and delete JSON and key-value data easily, efficiently and naturally.

I'm not going to dismiss the entire NoSQL movement out of hand. But I will say that the next time you're thinking of using a NoSQL database, consider using one that can already fulfill all of your needs, and which you might well be using already—PostgreSQL.■

**Reuven M. Lerner is a Web developer, consultant and trainer. He recently completed his PhD in Learning Sciences from Northwestern University. You can read his blog, Twitter feed and newsletter at http://lerner.co.il. Reuven lives with his wife and three children in Modi'in, Israel.**

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

Powerful: Rhino

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

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

Tablet: Raven

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

Rugged: Tarantula

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

EmperorLinux
...where Linux & laptops converge

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

Model specifications and availability may vary.

DAVE TAYLOR

# Mad Libs for Dreams, Part II

**Dream Interpreter—Dave mucks about with some free association and word substitution to create a dream interpretation script as suggested by a reader. Along the way, he also re-examines the problem of calculating leap years and shows off a handy text formatting trick too.**

**I'm in the middle** of writing what I'll call a *Mad Libs* for dream interpretation script, as detailed in my article in the October 2014 issue, but before I get back to it, I have to say that more people have written to me about the leap year function presented many months ago than any other topic in the history of this column.

I never realized people were so passionate about their leap years—and to consider that it's to compensate for the fact that our 365-day calendar is shorter than a solar year by almost six hours per year, starting way back in 1592, an extra day was added every four years (approximately).

The variety of solutions sent in were quite impressive, including some that were presented in FORTRAN and other classic scientific programming languages. Yes, FORTRAN.

The simplest solution proved to be letting Linux itself do the heavy lifting and just check to see how many days were in a given calendar year by using GNU `date` for a given year:

```
date -d 12/31/YEAR +%j
```

If it's 366, it's a leap year. If it's 365, it isn't—easy.

But the winner is reader Norbert Zacharias who sent in this link: **http://aa.usno.navy.mil/faq/docs/JD_Formula.php**. You can go there and enjoy the delightful complexity of this US Navy solution!

**Even better, the noun → free association phrase mapping is a one-way translation, so we don't even really need to save it.**

Now, back to dreams—a perfect segue!

In my last article, I started working on a reader-suggested script that would let people type in a few sentences describing a dream, then extract all the nouns and prompt the user for a free association synonym (or, I suppose, antonym), then echo back the original description with all the substitutions.

With the addition of a noun list and a simple technique for deconstructing what has been given to identify the nouns, most of the code actually is written. Even better, the noun → free association phrase mapping is a one-way translation, so we don't even really need to save it. This means that a sed sequence like:

```
s/old/new/g
```

will work just fine, and because that can be appended to multiple substitutions, it just might prove super easy.

Here's the code stub that prompts users for a new word for each existing noun they've entered:

```
for word in $nouns
do
  echo "What comes to mind when I say $word?"
done
```

To expand it as needed is easy:

```
echo "What comes to mind when I say $word?"
read newword
sedstring="$sedstring;s/$word/$newword/g"
```

That's it. Let's put that in place and see what happens when we create a half-dozen noun substitutions. I'll skip some of the I/O and just tell you that the phrase I entered was "The rain in spain falls mainly on the plain" and that the script then dutifully identified "rain", "spain" and "plain" as nouns.

The result:

```
What comes to mind when I say rain?
storm
What comes to mind when I say spain?
soccer
What comes to mind when I say plain?
jane
build sed string
➥;s/rain/storm/g;s/spain/soccer/g;s/plain/jane/g
```

Great. We're close to being done with the script—really close. In fact, all that's left is:

```
cat $dream | sed $sedstring
```

Let's try it:

```
$ dreamer.sh
Welcome to Dreamer. To start, please describe in a few sentences
the dream you'd like to explore. End with DONE in all caps on its
own line. The rain in Spain falls mainly on the plain.
DONE
Hmm.... okay. I have identified the following words as nouns:
 rain spain plain
Are you ready to do some free association? Let's begin...
What comes to mind when I say rain?
storm
What comes to mind when I say spain?
soccer
What comes to mind when I say plain?
jane
The result:
  The storm in Spain falls mainly on the jane.
```

By George, I think we have it! Here's the final code:

```
#!/bin/sh

# dreamer - script to help interpret dreams. does this by
#    asking users to describe their most recent dream,
#    then prompts them to free associate words
#    for each of the nouns in their original description.

nounlist="nounlist.txt"
dream="/tmp/dreamer.$$"

input=""; nouns=""

trap "/bin/rm -f $dream" 0      # no tempfile left behind

echo "Welcome to Dreamer. To start, please describe in a
➥few sentences"
echo "the dream you'd like to explore. End with "DONE"
➥in all caps on "
echo "its own line."

until [ "$input" = "DONE" -o "$input" = "done" ]
do
  echo "$input" >> $dream
  read input    # let's read another line from the user...
done

for word in $( sed 's/[[:punct:]]//g' $dream | tr '[A-Z]'
➥'[a-z]' | tr ' ' '\n')
do
  # is the word a noun? Let's look!
  if [ ! -z "$(grep -E "^${word}$" $nounlist)" ] ; then
    nouns="$nouns $word"
  fi
done

echo "Hmm.... okay. I have identified the following words as nouns:"
echo "$nouns"

echo "Are you ready to do some free association? Let's begin..."

for word in $nouns
do
```

**To be fair, this is a bit of an odd script to write, but the basic concept of breaking input down into individual words, processing those words and reassembling the output is something that does have wider applicability.**

```
  echo "What comes to mind when I say $word?"
  read newword
  sedstring="$sedstring;s/$word/$newword/g"
done

echo "The result:"
cat $dream | sed "$sedstring" | fmt | sed 's/^/  /'
echo ""
exit 0
```

To be fair, this is a bit of an odd script to write, but the basic concept of breaking input down into individual words, processing those words and reassembling the output is something that does have wider applicability. For example, you might use common acronyms but need to have them spelled out for a final report, or language substitution or redacting specific names.

There's also another trick worth noting on the last output line. Let's look at the statement again:

```
cat $dream | sed "$sedstring" | fmt | sed 's/^/  /'
```

The first two sections of this pipe do the word substitution. No rocket science there (well, unless your rocket happens to run Bourne Shell, but that's a somewhat anxiety-provoking concept). What's interesting are the last two elements.

The `fmt` command wraps overly long or short lines to make them all fill in to be around 80 characters long, and then the final `sed` statement prefaces every line with a double space. I actually use this frequently because I like my scripts to be able to output arbitrary length text that's nice and neat.

Let's grab that great journal from Ishmael and use it as an example:

```
$ cat moby.txt
Call me Ishmael.
Some years ago - never mind how long precisely - having little or no
money in my purse, and nothing particular to interest me on shore, I
thought I would sail about a little and see the watery part
of the world.
It is a way I have of driving off the spleen and regulating the
circulation. Whenever I find myself growing grim about the mouth;
```

```
whenever it is a damp, drizzly November in my soul; whenever I find
myself involuntarily pausing
before coffin
warehouses, and bringing up the rear
of every funeral I meet; and especially whenever my hypos get such an
upper hand of me, that it requires a strong moral principle to prevent
me from deliberately stepping into the street, and methodically
knocking people's hats off - then, I account it high time to get to
sea as soon as I can.
```

Run that output through the `fmt` command, however, and it all cleans up perfectly:

```
$ cat moby.txt | fmt
Call me Ishmael. Some years ago - never mind how long
precisely - having little or no money in my purse, and nothing
particular to interest me on shore, I thought I would sail
about a little and see the watery part of the world. It is
a way I have of driving off the spleen and regulating the
circulation. Whenever I find myself growing grim about the
mouth; whenever it is a damp, drizzly November in my soul;
whenever I find myself involuntarily pausing before coffin
warehouses, and bringing up the rear of every funeral I meet;
and especially whenever my hypos get such an upper hand of me,
that it requires a strong moral principle to prevent me from
deliberately stepping into the street, and methodically knocking
people's hats off - then, I account it high time to get to sea
as soon as I can.
```

Now let's indent each line by those two spaces:

```
$ cat moby.txt | fmt | sed 's/^/  /'
  Call me Ishmael.  Some years ago - never mind how long
  precisely - having little or no money in my purse, and
  nothing particular to interest me on shore, I thought I
  would sail about a little and see the watery part of the
  world.  It is a way I have of driving off the spleen and
  regulating the circulation.  Whenever I find myself growing
  grim about the mouth; whenever it is a damp, drizzly November
  in my soul; whenever I find myself involuntarily pausing
  before coffin warehouses, and bringing up the rear of every
  funeral I meet; and especially whenever my hypos get such an
  upper hand of me, that it requires a strong moral principle to
  prevent me from deliberately stepping into the street, and
  methodically knocking people's hats off - then, I account it
  high time to get to sea as soon as I can.
$
```

See how that works? You also can preface each line with ">" or any other sequence you'd like. Easy enough!

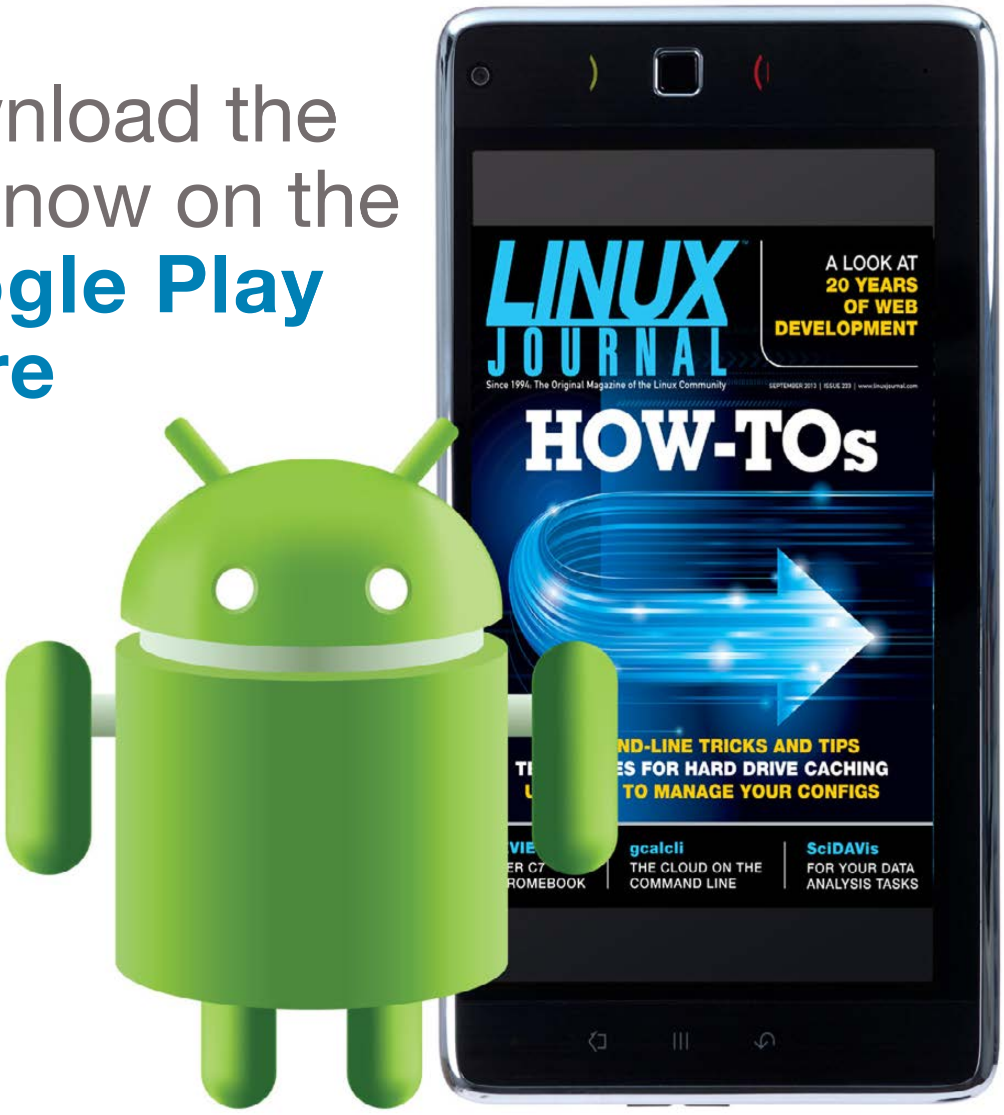
Well, that's it for this month. Next month, we'll dig into, um, I don't know. What should we explore next month, dear reader?■

**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 just completed a 10th anniversary revision to the book, coming very soon from O'Reilly and NoStarch Press). You can find him on Twitter as @DaveTaylor and more generally at his tech site http://www.AskDaveTaylor.com.**

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

LINUX JOURNAL on your Android device

Download the app now on the Google Play Store


LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
A LOOK AT 20 YEARS OF WEB DEVELOPMENT
HOW-TOs
ND-LINE TRICKS AND TIPS
ES FOR HARD DRIVE CACHING
TO MANAGE YOUR CONFIGS
gcalcli
THE CLOUD ON THE COMMAND LINE
SciDAVis
FOR YOUR DATA ANALYSIS TASKS

www.linuxjournal.com/android

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

# Localhost DNS Cache

KYLE RANKIN

**This month, Kyle covers one of his favorite topics—no, it's not mutt—it's DNS.**

**Is it weird** to say that DNS is my favorite protocol? Because DNS *is* my favorite protocol. There's something about the simplicity of UDP packets combined with the power of a service that the entire Internet relies on that grabs my interest. Through the years, I've been impressed with just how few resources you need to run a modest DNS infrastructure for an internal network.

Recently, as one of my environments started to grow, I noticed that even though the DNS servers were keeping up with the load, the query logs were full of queries for the same hosts over and over within seconds of each other. You see, often a default Linux installation does not come with any sort of local DNS caching. That means that every time a hostname needs to be resolved to an IP, the external DNS server is hit no matter what TTL you set for that record.

This article explains how simple it is to set up a lightweight local DNS cache that does nothing more than forward DNS requests to your normal resolvers and honor the TTL of the records it gets back.

There are a number of different ways to implement DNS caching. In the past, I've used systems like nscd that intercept DNS queries before they would go to name servers in /etc/resolv.conf and see if they already are present in the cache. Although it works, I always found nscd more difficult to troubleshoot than DNS when something went wrong. What I really wanted was just a local DNS server that honored TTL but would forward all requests to my real name servers. That way, I would get the speed and load benefits of a local cache, while also being able to troubleshoot any errors with standard DNS tools.

The solution I found was dnsmasq. Normally I am not a big advocate for dnsmasq, because it's often touted

**As a heavy user of configuration management systems, I prefer the servicename.d configuration model, as it makes it easy to push different configurations for different uses.**

as an easy-to-configure full DNS and DHCP server solution, and I prefer going with standalone services for that. Dnsmasq often will be configured to read /etc/resolv.conf for a list of upstream name servers to forward to and use /etc/hosts for zone configuration. I wanted something completely different. I had full-featured DNS servers already in place, and if I liked relying on /etc/hosts instead of DNS for hostname resolution, I'd hop in my DeLorean and go back to the early 1980s. Instead, the bulk of my dnsmasq configuration will be focused on disabling a lot of the default features.

The first step is to install dnsmasq. This software is widely available for most distributions, so just use your standard package manager to install the dnsmasq package. In my case, I'm installing this on Debian, so there are a few Debianisms to deal with that you might not have to consider if you use a different distribution. First is the fact that there are some rather important settings placed in /etc/default/dnsmasq. The file is fully commented, so I won't paste it here. Instead, I list two variables I made sure to set:

```
ENABLED=1
IGNORE_RESOLVCONF=yes
```

The first variable makes sure the service starts, and the second will tell dnsmasq to ignore any input from the resolvconf service (if it's installed) when determining what name servers to use. I will be specifying those manually anyway.

The next step is to configure dnsmasq itself. The default configuration file can be found at /etc/dnsmasq.conf, and you can edit it directly if you want, but in my case, Debian automatically sets up an /etc/dnsmasq.d directory and will load the configuration from any file you find in there. As a heavy user of configuration management systems, I prefer the servicename.d configuration model, as it makes it easy to push different configurations for different uses. If your distribution doesn't set up this directory for you, you can just edit /etc/dnsmasq.conf directly or look into adding an option like this to dnsmasq.conf:

```
conf-dir=/etc/dnsmasq.d
```

In my case, I created a new file called /etc/dnsmasq.d/dnscache.conf with the following settings:

```
no-hosts
no-resolv
listen-address=127.0.0.1
bind-interfaces
server=/dev.example.com/10.0.0.5
server=/10.in-addr.arpa/10.0.0.5
server=/dev.example.com/10.0.0.6
server=/10.in-addr.arpa/10.0.0.6
server=/dev.example.com/10.0.0.7
server=/10.in-addr.arpa/10.0.0.7
```

Let's go over each setting. The first, no-hosts, tells dnsmasq to ignore /etc/hosts and not use it as a source of DNS records. You want dnsmasq to use your upstream name servers only. The no-resolv setting tells dnsmasq not to use /etc/resolv.conf for the list of name servers to use. This is important, as later on, you will add dnsmasq's own IP to the top of /etc/resolv.conf, and you don't want it to end up in some loop. The next two settings, listen-address and bind-interfaces ensure that dnsmasq binds to and listens on only the localhost interface (127.0.0.1). You don't want to risk outsiders using your service as an open DNS relay.

The server configuration lines are where you add the upstream name servers you want dnsmasq to use. In my case, I added three different upstream name servers in my preferred order. The syntax for this line is `server=/domain_to_use/nameserver_ip`. So in the above example, it would use those name servers for dev.example.com resolution. In my case, I also wanted dnsmasq to use those name servers for IP-to-name resolution (PTR records), so since all the internal IPs are in the 10.x.x.x network, I added 10.in-addr.arpa as the domain.

Once this configuration file is in place, restart dnsmasq so the settings take effect. Then you can use `dig` pointed to localhost to test whether dnsmasq works:

```
$ dig ns1.dev.example.com @localhost

; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> ns1.dev.example.com @localhost
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4208
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;ns1.dev.example.com.                    IN      A

;; ANSWER SECTION:
ns1.dev.example.com.    265     IN      A       10.0.0.5

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Sep 18 00:59:18 2014
;; MSG SIZE  rcvd: 56
```

Here, I tested ns1.dev.example.com and saw that it correctly resolved to 10.0.0.5. If you inspect the `dig` output, you can see near the bottom of the output that `SERVER: 127.0.0.1#53(127.0.0.1)` confirms that I was indeed talking to 127.0.0.1 to get my answer. If you run this command again shortly afterward, you should notice that the TTL setting in the output (in the above example it was set to 265) will decrement. Dnsmasq is caching the response, and once the TTL gets to 0, dnsmasq will query a remote name server again.

After you have validated that dnsmasq functions, the final step is to edit /etc/resolv.conf and make sure that you have nameserver 127.0.0.1 listed above all other nameserver lines. Note that you can leave all of the existing name servers in place. In fact, that provides a means of safety in case dnsmasq ever were to crash. If you use DHCP to get an IP or otherwise have these values set from a different file (such as is the case when resolvconf is installed), you'll need to track down what files to modify instead; otherwise, the next time you get a DHCP lease, it will overwrite this with your new settings.

I deployed this simple change to around 100 servers in a particular environment, and it was amazing to see the dramatic drop in DNS traffic, load and log entries on my internal name servers. What's more, with this in place, the environment is even more tolerant in the case there ever were a real problem with downstream DNS servers—existing cached entries still would resolve for the host until TTL expired. So if you find your internal name servers are getting hammered with traffic, an internal DNS cache is something you definitely should consider.■

---

**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.**

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

SHAWN POWERS

# DevOps: Better Than the Sum of Its Parts

**Chef, a garden rake for the DevOps farm.**

**Most of us** longtime system administrators get a little nervous when people start talking about DevOps. It's an IT topic surrounded by a lot of mystery and confusion, much like the term "Cloud Computing" was a few years back. Thankfully, DevOps isn't something sysadmins need to fear. It's not software that allows developers to do the job of the traditional system administrator, but rather it's just a concept making both development and system administration better. Tools like Chef and Puppet (and Salt Stack, Ansible, New Relic and so on) aren't "DevOps", they're just tools that allow IT professionals to adopt a DevOps mindset. Let's start there.

## What Is DevOps?

Ask ten people to define DevOps, and you'll likely get 11 different answers. (Those numbers work in binary too, although I suggest a larger sample size.) The problem is that many folks confuse DevOps with DevOps tools. These days, when people ask me, "What is DevOps?", I generally respond: "DevOps isn't a thing, it's a way of doing a thing."

The worlds of system administration and development historically have been very separate. As a sysadmin, I tend to think very differently about computing from how a developer does. For me, things like scalability and redundancy are critical, and my success often is gauged by uptime. If things are running, I'm successful. Developers have a different way of approaching their jobs, and need to consider things like efficiency, stability, security and features. Their success often is measured by usability.

Hopefully, you're thinking the

traits I listed are important for both development *and* system administration. In fact, it's that mindset from which DevOps was born. If we took the best practices from the world of development, and infused them into the processes of operations, it would make system administration more efficient, more reliable and ultimately better. The same is true for developers. If they can begin to "code" their own hardware as part of the development process, they can produce and deploy code more quickly and more efficiently. It's basically the Reese's Peanut Butter Cup of IT. Combining the strengths of both departments creates a result that is better than the sum of its parts.

Once you understand what DevOps really is, it's easy to see how people confuse the tools (Chef, Puppet, New Relic and so on) for DevOps itself. Those tools make it so easy for people to adopt the DevOps mindset, that they become almost synonymous with the concept itself. But don't be seduced by the toys—an organization can shift to a very successful DevOps way of doing things simply by focusing on communication and cross-discipline learning. The tools make it easier, but just like owning a rake doesn't make someone a farmer, wedging DevOps tools into your organization doesn't create a DevOps team for you. That said, just like any farmer appreciates a good rake, any DevOps team will benefit from using the plethora of tools in the DevOps world.

## The System Administrator's New Rake

In this article, I want to talk about using DevOps tools as a system administrator. If you're a sysadmin who isn't using a configuration management tool to keep track of your servers, I urge you to check one out. I'm going to talk about Chef, because for my day job, I recently taught a course on how to use it. Since you're basically learning the concepts behind DevOps tools, it doesn't matter that you're focusing on Chef. Kyle Rankin is a big fan of Puppet, and conceptually, it's just another type of rake. If you have a favorite application that isn't Chef, awesome.

If I'm completely honest, I have to admit I was hesitant to learn Chef, because it sounded scary and didn't seem to do anything I wasn't already doing with Bash scripts and cron jobs. Plus, Chef uses the Ruby programming language for its configuration files, and my programming skills peaked with:

```
10 PRINT "Hello!"
20 GOTO 10
```

Nevertheless, I had to learn about it so I could teach the class. I can tell you with confidence, it was worth it. Chef requires basically zero programming knowledge. In fact, if no one mentioned that its configuration files were Ruby, I'd just have assumed the syntax for the conf files was specific and unique. Weird config files are nothing new, and honestly, Chef's config files are easy to figure out.

## Chef: Its Endless Potential

DevOps is a powerful concept, and as such, Chef can do amazing things. Truly. Using creative "recipes", it's possible to spin up hundreds of servers in the cloud, deploy apps, automatically scale based on need and treat every aspect of computing as if it were just a function to call from simple code. You can run Chef on a local server. You can use the cloud-based service from the Chef company instead of hosting a server. You even can use Chef completely server-less, deploying the code on a single computer in solo mode.

Once it's set up, Chef supports multiple environments of similar infrastructures. You can have a development environment that is completely separate from production, and have the distinction made completely by the version numbers of your configuration files. You can have your configurations function completely platform agnostically, so a recipe to spin up an Apache server will work whether you're using CentOS, Ubuntu, Windows or OS X. Basically, Chef can be the central resource for organizing your entire infrastructure, including hardware, software, networking and even user management.

Thankfully, it doesn't have to do all that. If using Chef meant turning your entire organization on its head, no one would ever adopt it. Chef can be installed small, and if you desire, it can grow to handle more and more in your company. To continue with my farmer analogy, Chef can be a simple garden rake, or it can be a giant diesel combine tractor. And sometimes, you just need a garden rake. That's what you're going to learn today. A simple introduction to the Chef way of doing things, allowing you to build or not build onto it later.

## The Bits and Pieces

Initially, this was going to be a multipart article on the specifics

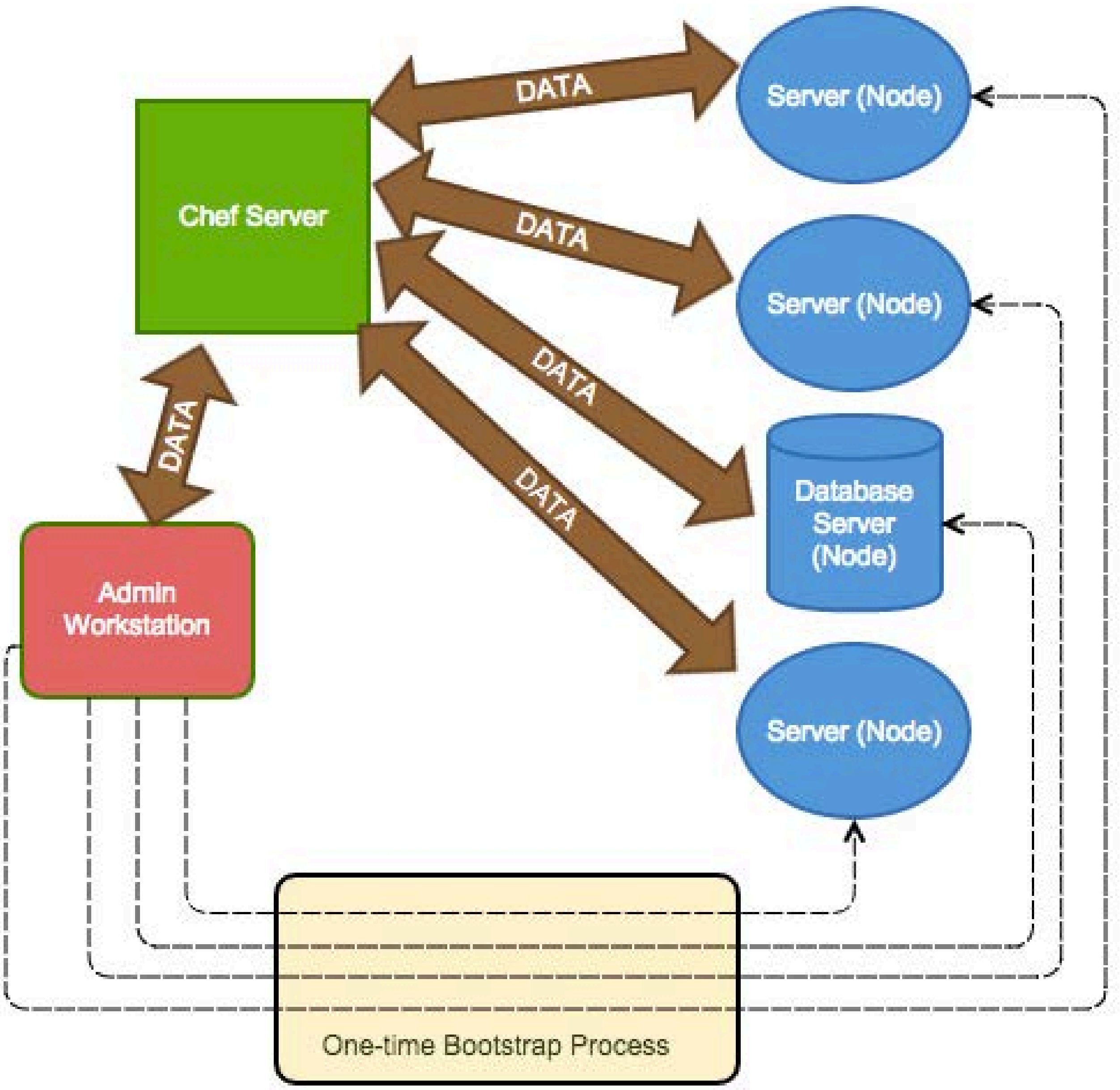


**Figure 1. This is the basic Chef setup, showing how data flows.**

of setting up Chef for your environment. I still might do a series like that for Chef or another DevOps configuration automation package, but here I want everyone to understand not only DevOps itself, but what the DevOps tools do. And again, my example will be Chef.

At its heart, Chef functions as a central repository for all your configuration files. Those configuration files also include the ability to carry out functions on servers. If you're a sysadmin, think of it as a central, dynamic /etc directory along with a place all your

Bash and Perl scripts are held. See Figure 1 for a visual on how Chef's information flows.

The Admin Workstation is the computer at which configuration files and scripts are created. In the world of Chef, those are called cookbooks and recipes, but basically, it's the place all the human-work is done. Generally, the local Chef files are kept in a revision control system like Git, so that configurations can be rolled back in the case of a failure. This was my first clue that DevOps might make things better for system administrators, because in the past all my configuration revision control was done by making a copy of a configuration file before editing it, and tacking a .date at the end of the filename. Compared to the code revision tools in the developer's world, that method (or at least my method) is crude at best.

The cookbooks and recipes created on the administrator workstation describe things like what files should be installed on the server nodes, what configurations should look like, what applications should be installed and stuff like that. Chef does an amazing job of being platform-neutral, so if your cookbook installs Apache, it generally can install Apache without you needing to specify what type of system it's installing on. If you've ever been frustrated by Red Hat variants calling Apache "httpd", and Debian variants calling it "apache2", you'll love Chef.

Once you have created the cookbooks and recipes you need to configure your servers, you upload them to the Chef server. You can connect to the Chef server via its Web interface, but very little actual work is done via the Web interface. Most of the configuration is done on the command line of the Admin Workstation. Honestly, that is something a little confusing about Chef that gets a little better with every update. Some things can be modified via the Web page interface, but many things can't. A few things can *only* be modified on the Web page, but it's not always clear which or why.

With the code, configs and files uploaded to the Chef Server, the attention is turned to the nodes. Before a node is part of the Chef environment, it must be "bootstrapped". The process isn't difficult, but it is required in order to use Chef. The client software is installed on each new node, and then configuration files and commands are pulled from the Chef server. In fact, in order for Chef to function, the nodes must be configured to poll the server periodically for any changes. There is no "push" methodology to send changes or updates to the node, so regular client updates are important. (These are generally performed via cron.)

At this point, it might seem a little silly to have all those extra steps when a simple FOR loop with some SSH commands could accomplish the same tasks from the workstation, and have the advantage of no Chef client installation or periodic polling. And I confess, that was my thought at first too. When programs like Chef really prove their worth, however, is when the number of nodes begins to scale up. Once the admittedly complex setup is created, spinning up a new server is literally a single one-liner to bootstrap a node. Using something like Amazon Web Services, or Vagrant, even the creation of the computers themselves can be part of the Chef process.

## To Host or Not to Host

The folks at Chef have made the process of getting a Chef Server instance as simple as signing up for a free account on their cloud infrastructure. They maintain a "Chef Server" that allows you to upload all your code and configs to their server, so you need to worry only about your nodes. They even allow you to connect five of your server nodes for free. If you have a small environment, or if you don't have the resources to host your own Chef Server, it's tempting just to use their pre-configured cloud service. Be warned, however, that it's free only because they hope you'll start to depend on the service and eventually pay for connecting more than those initial five free nodes.

They have an enterprise-based self-hosted solution that moves the Chef Server into your environment like Figure 1 shows. But it's important to realize that Chef is open source, so there is a completely free, and fully functional open-source version of the server you can download and install into your environment as well. You do lose their support, but if you're just starting out with Chef or just playing with it, having the open-source version is a smart way to go.

## How to Begin?

The best news about Chef is that incredible resources exist for learning how to use it. On the **http://getchef.com** Web site, there is a video series outlining a basic setup for installing Apache on your server nodes as an example of the process. Plus, there's great documentation that describes the installation process of the open-source Chef Server, if that's the path you want to try.

Once you're familiar with how Chef works (really, go through the training videos, or find other Chef fundamentals training somewhere), the next step is to check out the vibrant Chef community. There are cookbooks and recipes for just about any situation you can imagine. The cookbooks are just open-source code and configuration files, so you can tweak them to fit your particular needs, but like any downloaded code, it's nice to start with something and tweak it instead of starting from scratch.

DevOps is not a scary new trend invented by developers in order to get rid of pesky system administrators. We're not being replaced by code, and our skills aren't becoming useless. What a DevOps mindset means is that we get to steal the awesome tools developers use to keep their code organized and efficient, while at the same time we can hand off some of the tasks we hate (spinning up test servers for example) to the developers, so they can do their jobs better, and we can focus on more important sysadmin things. Tearing down that wall between development and operations truly makes everyone's job easier, but it requires communication, trust and a few good rakes in order to be successful. Check out a tool like Chef, and see if DevOps can make your job easier and more awesome.■

---

**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.**

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

## Resources

Chef Fundamentals Video Series: **https://learn.getchef.com/fundamentals-series**

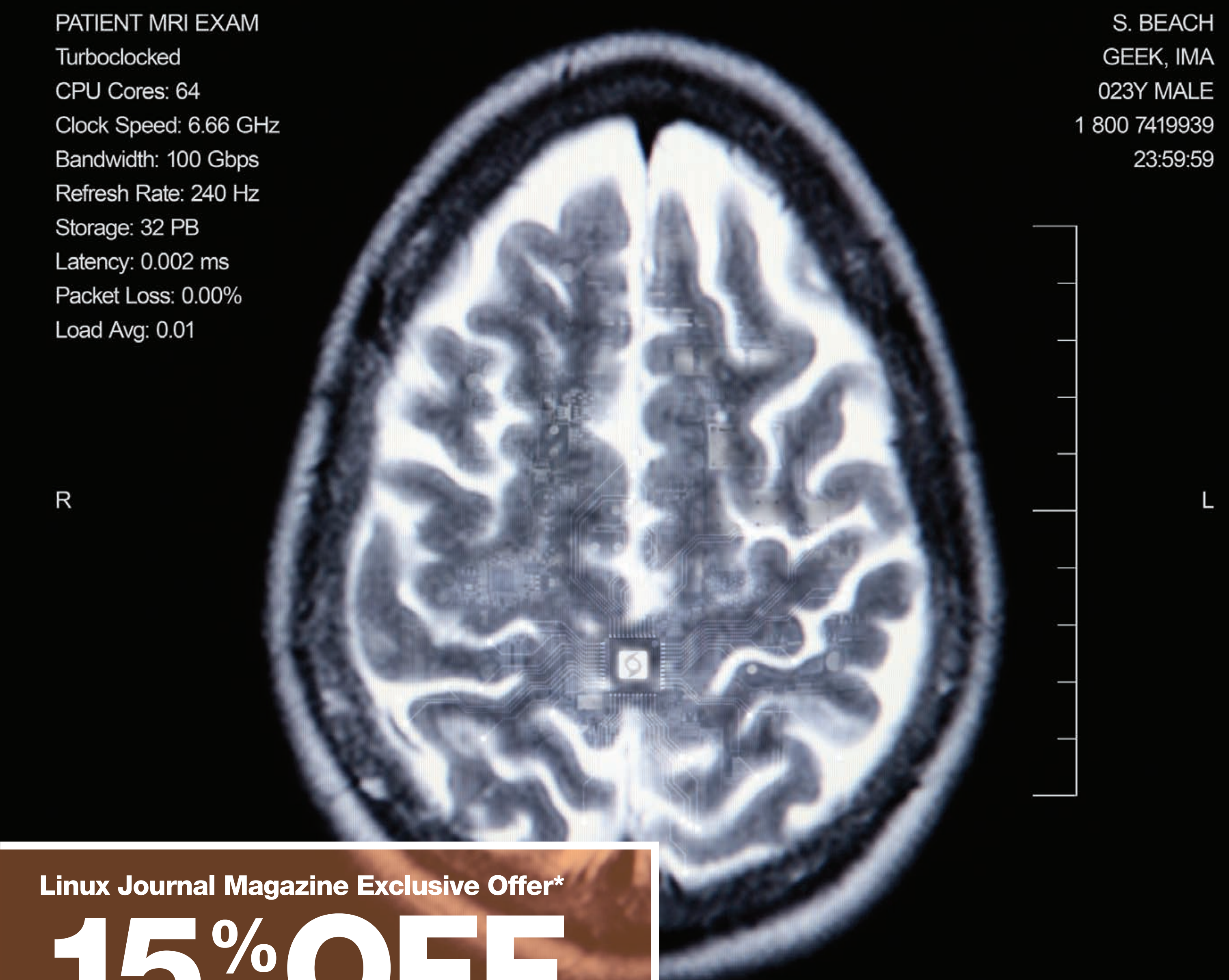
Chef Documentation: **https://docs.getchef.com**

Community Cookbooks/Tools: **https://supermarket.getchef.com**

ServerBeach

DEDICATED SERVERS. BY GEEKS FOR GEEKS.

**we get how geeks think.**



**Linux Journal Magazine Exclusive Offer***

**15% OFF**

Call 1.888.840.9091 | serverbeach.com

Sign up for any dedicated server at ServerBeach and get 15% off*. Use the promo code: ***LJ15OFF*** when ordering.

* Offer expires December 31st, 2010.

## Wibu-Systems' CodeMeter Embedded Driver

Embedded systems developers seeking to protect their IPs are the target customers for Wibu-Systems' CodeMeter Embedded Driver, a comprehensive security solution that secures embedded software against reverse-engineering by encrypting and signing the binary code. CodeMeter protects embedded systems, programmable logic controllers and industrial PCs. The new CodeMeter Embedded Driver 1.7—a rebranded version of a product called CodeMeter Compact Driver 1.6—offers new features and functionality that are applicable specifically to embedded systems. New features include an option to use the HID mode on dongles for communication with the device without displaying drive status, protection of the secure boot process, support for the file I/O interface for Linux and Android, and support for the Secure Disk standard for reading and writing API-based data without enumeration by the operating system. The driver is available for VxWorks 7.0, Linux Embedded, Windows Embedded, Android and QNX, as well as for different ARM, x86 and PowerPC platforms.
http://www.wibu.com/us

## Linutop OS

Intruders beware, because the new Linutop OS 14.04 is here—the easiest way to set up an ultra-secure PC, says its maker Linutop. Linutop OS 14.04 is a customized version of Ubuntu 14.04 LTS that comes loaded with the light XFCE classic graphic environment, as well as an array of ready-to-use Linux applications, such as Firefox 28, LibreOffice 4, VLC 2 and Linutop Kiosk. Version 14.04 offers three core enhancements, namely a Linutop Kiosk for a secured Internet access point, Digital Signage functionality for display of any media type and enhanced security and privacy. Linutop's system can be locked in read-only mode, preventing alterations by viruses or other mishaps. Linutop requires only a minimal HD space (850MB) and requires minimal processing power: PIII 800MHz and 512MB of RAM. Linutop OS can be installed quickly on a hard drive, USB key or Flash memory.
http://www.linutop.com

## Logic Supply's ML400 Series Industrial PCs

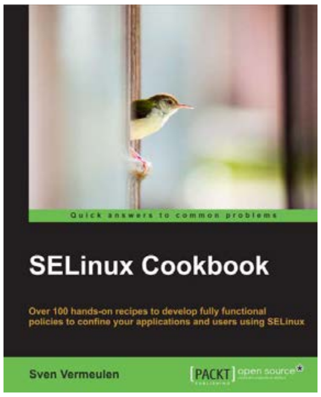
For Logic Supply, the new ML400 Series of industrial PCs is more than just the next step in the evolution of its product line. Rather, says Logic, it's a distinct break from the "black box" paradigm that has ruled the industrial hardware market. Logic Supply's new ML400 Series is a line of high-performance, boldly styled, rugged Mini-ITX systems for commercial applications where reliability is paramount. These fanless, ventless PCs are the company's smallest to date and are engineered for use in harsh environments. The models available at launch for the ML400 series offer a versatile range of I/O and Intel processing capabilities, advanced EMI protection and next-generation storage in order to maintain an ultra-compact footprint.
http://www.logicsupply.com

## Silicon Mechanics, Inc.'s Rack-Mount Servers with Intel Xeon E5-2600 v3

Hardware-maker Silicon Mechanics, Inc., is leveraging the latest Intel Xeon processor E5-2600 v3 product family to create a line of new servers that "will thrill customers looking to save on operating expenses". Thanks in large part to the new processor features—more cores, more cache, faster memory and an updated chipset—the Silicon Mechanics rack-mount servers feature a well rounded balance of cost, performance and energy use. These five of the company's most popular models sport efficient DDR4 memory, processors with new power-management features and extensive performance improvements. Finally, the new servers offer customers a great deal of flexibility regarding memory, storage and power management, making it easy to find a configuration with the ideal features for nearly any application and budget, says the company.
http://www.siliconmechanics.com

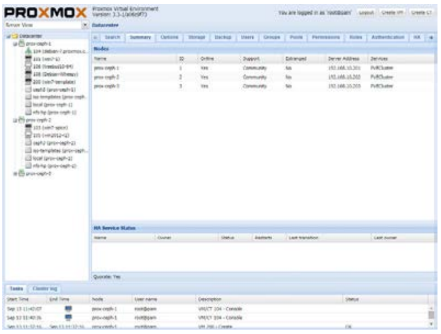
## Red Hat Software Collections

In order to keep up with developers' needs while maintaining production stability, Red Hat keeps the Red Hat Software Collections' release schedule at a more frequent release schedule than RHEL. The Collections, recently upgraded to v1.2, is a package of essential Web development tools, dynamic languages, open-source databases, C and C++ compilers, the Eclipse IDE, and a variety of development and performance management tools. These updated components can be installed alongside versions included in base Red Hat Enterprise Linux. Highlights of the upgrade are the Red Hat Developer Toolset 3.0, included in the Collections for the first time and also bringing the Eclipse IDE to RHEL 7 for the first time; DevAssistant 0.9.1, a tool for setting up development environments and publishing code; Maven 3.0, a build automation tool for Java projects; Git 1.9.4, which previously was only part of the Red Hat Developer Toolset; Nginx 1.6 Web server and Web proxy; and the latest stable versions of popular dynamic languages and open-source databases. Red Hat Software Collections 1.2 is available to eligible users of Red Hat Enterprise Linux 6 and 7.
http://www.redhat.com


Quick answers to common problems
SELinux Cookbook
Over 100 hands-on recipes to develop fully functional policies to confine your applications and users using SELinux
Sven Vermeulen
PACKT open source

## Sven Vermeulen's *SELinux Cookbook* (Packt Publishing)

If you are a Linux system or service administrator and want to (wisely) burnish your SELinux skills, then Packt Publishing and tech author Sven Vermeulen have a book for you. It's called *SELinux Cookbook*, and it carries a breathless subtitle that sums it up better than any bumbling Linux journalist could: "Over 100 hands-on recipes to develop fully functional policies to confine your applications and users using SELinux". These policies can be custom to users' own needs, and users can build readable policy rules from them. Readers can learn further about the wide range of security controls that SELinux offers by customizing Web application confinement. Finally, readers will understand how some applications interact with the SELinux subsystem internally, ensuring that they can confront any challenge they face. Author Sven Vermeulen is the project lead of Gentoo Hardened's SELinux integration project and wrote Packt's *SELinux System Administration* book as well.
http://www.packtpub.com



## Proxmox Server Solutions GmbH's Proxmox Virtual Environment (VE)

Proxmox Virtual Environment (VE) is a Debian GNU/Linux-based open-source virtualization management solution for servers. Proxmox VE supports KVM-based guests, container-virtualization with OpenVZ and includes strong high-availability support based on Red Hat Cluster and Corosync. Maker Proxmox Server Solutions recently announced a security-focused version 3.3, whose key additions include an HTML5 console, Proxmox VE Firewall, two-factor authentication, a ZFS storage plugin and Proxmox VE Mobile. Proxmox is proudest of the distributed Proxmox VE Firewall, which is designed to protect the whole IT infrastructure. It allows users to set up firewall rules for all hosts, the cluster, virtual machines and containers. The company notes that Proxmox VE is used by 62,000 hosts in 140 countries, its GUI is available in 17 languages, and the active community counts more than 24,000 forum members.
http://www.proxmox.com

## Opera Software ASA's Opera TV Ads SDK

Over time, Opera has become much more than a browser maker. Opera's latest development is part of the company's somewhat new niche in the media convergence space: Opera TV Ads SDK. The new solution is targeted at app publishers, Smart TV device manufacturers and pay-TV operators seeking to better monetize their content by serving video advertising on any platform. Opera TV Ads SDK previously was available exclusively to apps distributed via the Opera TV Store application platform and developed through the Opera TV Snap technology. With this new release, the solution is available as a standalone feature for any HTML5 app or Smart TV device, whether on the Opera TV Store or other application platforms. Opera says that Opera TV Ads SDK offers a one-stop solution for placement of video advertising anywhere inside the device user interface, including targeting users across apps and interactive advertising via linear broadcast.
http://www.opera.com/tv

**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.**

# Ideal Backups with zbackup

Do you need to back up large volumes of data spread over many machines with "Time Machine"-like snapshots? Read on!

DAVID BARTON

Data is growing both in volume and importance. As time goes on, the amount of data that we need to store is growing, and the data itself is becoming more and more critical for organizations. It is becoming increasingly important to be able to back up and restore this information quickly and reliably. Using cloud-based systems spreads out the data over many servers and locations.

Where I work, data has grown from less than 1GB on a single server to more than 500GB spread out on more than 30 servers in multiple data centers. Catastrophes like the events at Distribute IT and Code Spaces demonstrate that ineffective backup practices can destroy a thriving business. Enterprise-level backup solutions typically cost a prohibitive amount, but the tools we need to create a backup solution exist within the Open Source community.

## zbackup to the Rescue

After switching between many different backup strategies, I have found what is close to an ideal backup solution for our particular use case. That involves regularly backing up many machines with huge numbers of files as well as very large files and being able to restore any backup previously made.

The solution combines zbackup, rsync and LVM snapshots. zbackup works by deduplicating a stream—for example, a tar or database backup—and storing the blocks into a storage pool. If the same block ever is encountered again, the previous one is reused.

Combining these three elements gives us a solution that provides:

- Multiple versions: we can store complete snapshots of our system every hour, and deduplication means the incremental storage cost for each new backup is negligible.

- Storing very large files: database backups can be very large but differ in small ways that are not block-aligned (imagine inserting one byte at the beginning of a file). Byte-level deduplication means we store only the changes between the versions, similar to doing a diff.

- Storing many small files: backing up millions of files gives a much smaller number of deduplicated blocks that can be managed more easily.

- Easily replicating between disks and over a WAN: the files in the storage pool are immutable; new blocks are stored as new files. This makes rsyncing them to other drives or machines very fast and efficient. It also means we can synchronize them to virtually any kind of machine or file storage.

- Compression: compressing files gives significant size reductions, but using it often stops rsync or deduplication from working. zbackup compresses the blocks after deduplication, so rsyncing is still efficient. As mentioned previously, only new blocks need to be rsynced.

- Fast backups: backups after the first one are done at close to the disk-read speed. More important, by running zbackup on each server, the majority of the CPU and I/O load is decentralized. This means there is minimal CPU or I/O required on the central server and only deduplicated blocks are transferred, providing scalability.

- Highly redundant: by synchronizing to external drives and other servers, even corruption or destruction of the backups means we can recover our information.

## Comparing Alternatives

There are many alternatives to using zbackup. I compare some of the options below:

- tape: has a relatively high cost, and takes a long time to read and write as the entire backup is written. This is a good option for archival storage, but it is unsuitable for frequent snapshots because you can't write a 500GB tape every hour.

- rsnapshot: does not handle small changes in large files in any reasonable way, as a new copy is kept for each new version. Taking snapshots of large numbers of files causes a huge I/O load on the central backup server when they are copied and when they are deleted. It is also very slow to synchronize the hard links to another device or machine.

- Tarsnap: this is an excellent product and very reasonably priced. Slow restores and being dependent on a third party for storage make this a good fallback option but possibly unsuitable as your only method of backup.

- Git: doesn't handle large files

efficiently (or in some cases fails completely). It also doesn't easily handle anything with Git control files in it, so it makes backing up your Git repositories a real challenge. As Git is so poor at large files, tarring directories and using the tar file is not feasible.

- ZFS/BTRFS: filesystem snapshots are very fast and work well for small files. Even the smallest change in a file requires the file to be re-copied (this is not strictly true for ZFS if deduplicating is enabled; however, this has a significant memory load and it works only if the file is unchanged for most of its blocks, like an Mbox file or database backing store).

- Duplicity: this seems similar to zbackup and has many of the same benefits, except deduplicating between files with different names. Although it has been in beta for a long time, it seems to have many features for supporting remote back ends, whereas zbackup is simply a deduplicating block store.

## Summary of Approach

The key part of this approach is using zbackup in step 1. The backups produced by zbackup have remarkable properties compared to the other backup formats, as discussed previously, so that the remaining steps can be tailored depending on the level of availability and durability you need.

1. Each virtual server uses zbackup to back up to a local deduplicated block store. This means every snapshot is available locally if needed.

2. The zbackup store then is replicated to a central backup server where it can be recovered if needed.

3. The zbackup stores on the central server are replicated out to other servers.

4. The backups also are synchronized to external storage—for example, a USB drive. We rotate between drives so that drives are kept off-site and in case of disaster or backup corruption.

5. Finally, we snapshot the filesystem where the zbackup stores are located.

## Using zbackup

zbackup fits right into the UNIX philosophy. It does two seemingly simple things that make it behave almost like a file. The first is taking a stream of data passed to stdin and writing it to a block store. A handle to the data is stored in a small backup file, stored next to the block store. The second is taking that backup file and writing the original data to stdout.

During the process, zbackup will identify blocks of data that it has seen before and deduplicate it and then compress any new data before writing it out to disk. When deduplicating data, zbackup uses a sliding window that moves a byte at a time, so that if you insert a single byte into a file, it still can identify the repeated blocks. This is in contrast to block-level deduplication like that found in ZFS.

To start using zbackup, you must install it from source. This is very easy to do; just follow the instructions on the **http://zbackup.org** Web site.

Assuming you have installed zbackup, and that /usr/local/bin is in your path, start by initializing a block store (in these examples, I am running as root, but that is not a requirement):

```
# zbackup init --non-encrypted  /tmp/zbackup/
```

Hopefully you don't use /tmp for your real backups! You can list out the block store as below—the Web site has great information on what goes where. The main one to keep in mind is backups; this is where your backup files go:

```
# ls /tmp/zbackup
backups  bundles  index  info
```

Let's back up a database backup file—this takes a while the first time (Listing 1).

To check where that went, look at Listing 2. As you can see, the backup file is only 135 bytes. Most of the data is stored in /bundles, and it is less than one tenth the size of the

**Listing 1. Backing Up One File**

```
# ls -l /tmp/database.sql
-rw-r--r-- 1 root root 406623470 Sep 14 17:41 /tmp/database.sql
# cat /tmp/database.sql | zbackup backup
 ➥/tmp/zbackup/backups/database.sql
Loading index...
Index loaded.
Using up to 8 thread(s) for compression
```

**Listing 2. Check the Backup**

```
# ls -l /tmp/zbackup/backups/database.sql
-rw------- 1 root root 135 Sep 14 17:43
 ➥/tmp/zbackup/backups/database.sql
# du --max-depth=1 /tmp/zbackup/
8       /tmp/zbackup/backups
208     /tmp/zbackup/index
29440   /tmp/zbackup/bundles
```

original database.

Now, make a small change to the backup file to simulate some use and then back it up again (see Listing 3). This example illustrates an important point, that zbackup will not change any file in the data store. You can rename the files in the /backup directory if you choose. You also can have subdirectories under /backups, as shown in Listing 4, where the backup finally works.

This should complete much more quickly, both because the file is cached and because most of the blocks already have been deduplicated:

```
# du --max-depth=0 /tmp/zbackup/
29768 /tmp/zbackup/
```

In this example, the changes I made to the file have only slightly increased the size of the backup.

Let's now restore the second backup. Simply pass the backup handle to zbackup restore, and the file is written to stdout:

```
# zbackup restore /tmp/zbackup/backups/1/2/3/database.sql >
 ➥/tmp/database.sql.restored
```

Now you can check the file you restored to prove it is the same as

**Listing 3. Backing Up a File Again**

```
# cat /tmp/database.sql | zbackup --silent backup
 ➥/tmp/zbackup/backups/database.sql
Won't overwrite existing file /tmp/zbackup/backups/database.sql
```

**Listing 4. Backing Up a File, Part 2**

```
# mkdir -p /tmp/zbackup/backups/1/2/3/
# cat /tmp/database.sql | zbackup --silent backup
 ➥/tmp/zbackup/backups/1/2/3/database.sql
```

**Listing 5. Checking the Restored File**

```
# ls -l /tmp/database.sql*
-rw-r--r-- 1 root root 406622180 Sep 14 17:47 /tmp/database.sql
-rw-r--r-- 1 root root 406622180 Sep 14 17:53
 ➥/tmp/database.sql.restored
# md5sum /tmp/database.sql*
179a33abbc3e8cd2058703b96dff8eb4  /tmp/database.sql
179a33abbc3e8cd2058703b96dff8eb4  /tmp/database.sql.restored
```

**Listing 6. tar and Back Up a Directory**

```
# tar -c /tmp/files | zbackup
 ➥--silent backup /tmp/zbackup/backups/files.tar
# du --max-depth=0 /tmp/zbackup
97128   /tmp/zbackup
```

the file you originally backed up (Listing 5).

Of course, in most cases, you aren't backing up a single file. This is where the UNIX philosophy works well—because tar can read from stdin and write to stdout, you simply can chain zbackup to tar. Listing 6 shows an example of backing up a large directory structure in /tmp/files/ using tar piped to zbackup.

Now there are two backups of the database file and a tarred backup of /tmp/files in the one zbackup store. There is nothing stopping you from calling your backup file files.tar.gz or anything else; however, this is going to be very confusing later on. If you name your backup file based on the name of the file to which it restores, it makes it much easier to work out what each backup is.

Now you can restore this backup using the example in Listing 7. Most of the example is creating the directory to restore to and comparing the restored backup to the original.

If you are backing up frequently, it makes sense to organize your backups in directories by date. The example in Listing 8 has a directory for each month, then a subdirectory for each day and, finally, a subdirectory for each time of

**Listing 7. Restoring from zbackup**

```
# mkdir /tmp/files.restore
# cd /tmp/files.restore/
# zbackup --silent restore /tmp/zbackup/backups/files.tar | tar -x
# diff -rq /tmpfiles.restore/tmp/files/ /tmp/files/
```

**Listing 8. Organize Your Backups**

```
# export DATEDIR=`date "+%Y-%m/%d/%H:%M"`
# mkdir -p /tmp/zbackup/backups/$DATEDIR
# tar -c /tmp/files | zbackup --silent backup
 ➥/tmp/zbackup/backups/$DATEDIR/files.tar
# cat /tmp/database.sql | zbackup backup
 ➥/tmp/zbackup/backups/$DATEDIR/database.sql
```

day—for example, 2014-09/12/08:30/—and all the backups for that time go in this directory.

Run this on a daily or hourly basis, and you can restore any backup you have made, going back to the beginning of time. For the files I am backing up, the zbackup data for an entire year is less than storing a single uncompressed backup.

The zbackup directory has the extremely nice property that the files in it never change once they have been written. This makes it very fast to rsync (since only new files in the backup need to be read) and very fast to copy to other media like USB disks. It also makes it an ideal candidate for things like filesystem snapshots using LVM or ZFS.

Once you have your backups in zbackup, you can ship it to a central server and drop it to USB or tape, or upload it to Amazon S3 or even Dropbox.

## Benchmarks/Results

All this is good in theory, but the critical question is "How does it perform?" To give you an idea, I have run some benchmarks on a server that has multiple similar versions of the same application—for example, training, development, UAT. There are roughly 5GB of databases and 800MB of Web site files. The

Table 1. Multiple Web Sites

| | SPACE | TIME | FILES |
|---|---|---|---|
| **tar** | 743M | 25s | 1 |
| **tar & gzip** | 382M | 44s | 1 |
| **zbackup** | 105M | 38s | 203 |
| **zbackup 2** | 4K | 30s | 206 |
| **zbackup 3** | 632k | 30s | 209 |

Table 2. Single Web Site

| | SPACE | TIME | FILES |
|---|---|---|---|
| **tar** | 280M | 8s | 1 |
| **tar & gzip** | 74M | 9s | 1 |
| **zbackup** | 66M | 17s | 131 |

server has eight cores and plenty of memory, although all buffers were flushed prior to each benchmark.

*All Web Sites:* this is a collection of 30,000 files taking roughly 800MB of space. Table 1 illustrates the results. zbackup delivers a backup that is roughly a quarter of the size of the gzipped tar file. Each new backup adds three files—by design, zbackup never modifies files but only adds them.

The first time zbackup runs and backs up the entire directory, it takes longer, as there is no deduplicated data in the pool. On the first run, all eight cores were fully used. On slower machines, throughput is less due to the high CPU usage.

The second time, zbackup was run over an identical file structure, only 4k of additional storage was used. The backup also runs faster because most of the data already is present.

The third time, four files of exactly 100,000 random bytes were placed in the filesystem.

*Single Web Site:* the compression performance of zbackup in the first test is in large part because there are multiple similar copies of the same Web site. This test backs up only one of the Web sites to provide another type of comparison.

The results are shown in Table 2. The compression results are not much better than gzip, which demonstrates how effective the deduplication is when doing multiple Web sites.

*Database Files:* this is a backup of a database dump file, text format uncompressed. The results are

Table 3. Database File

| | SPACE | TIME | FILES |
|---|---|---|---|
| **tar** | 377M | 2s | 1 |
| **tar & gzip** | 43M | 10s | 1 |
| **zbackup** | 29M | 32s | 192 |
| **zbackup 2** | 4M | 3s | 200 |
| **zbackup 3** | 164K | 3s | 210 |

shown in Table 3.

The first run is zbackup backing up a testing database of 377M. The deduplication and compression give significant gains over tar and gzip, although it runs much slower.

The second zbackup was a training database that is similar to the testing database, but it has an additional 10MB of data, and some of the other data also is different. In this case, zbackup very effectively removes the duplicates, with very little extra storage cost.

The final zbackup was randomly removing clusters of rows from the backup file to simulate the changes that come from updates and deletes. This is the typical case of backing up a database over short periods of time, and it matches very closely with my observation of real-word performance.

*Network Performance:* by design, zbackup does not modify or delete files. This means the number of added files and the additional disk space is all you need to synchronize over the network. Existing files never need to be updated.

Rather than benchmarking this, I have reviewed the real logs for our server. Synchronizing 6GB of data with more than 30,000 files typically takes less than ten seconds. Compared with the previous method of rsyncing the directory tree and large files that used to take between one to three minutes, this is an enormous improvement.

The central server has a slow disk and network; however, it is easily able to cope with the load from synchronizing the zbackup. I suspect even a Raspberry Pi would have enough performance to act as a synchronization target.

As they say, your mileage may vary. There are many factors that can alter the performance you get, such as:

- Disk speed.

- CPU performance (which is particularly important for the first backup).

zbackup makes it relatively simple to encrypt the data stored in the backup.

- Nature of the files—for example, binary database backups will compress less than text backups.

- Existence of multiple copies of the same data.

## Data Integrity and Security

Deduplicating the data, zbackup is particularly vulnerable to file corruption. A change to a single file could make the entire data store useless. It is worthwhile to check your media to ensure they are in good condition. On the plus side, you probably can copy an entire year's worth of backups of 200GB of data to another disk in less than an hour.

Having multiple versions of backups available in the same zbackup store is not the same as having multiple copies. Replicating your zbackup store to other disks or servers does not solve the problem. As an example, if someone were to modify some files in the backup store, and then that was blindly replicated to every machine or disk, you would have many exact copies of a worthless backup.

For that reason, I include snapshots of the filesystem to guard against this and also rotate our media and regularly check the backups. As an alternative, you could rsync just new files from the server being backed up and ignore deletions or file updates.

The design of zbackup means that retrieving a backup also checks it for consistency, so it is worthwhile to try restoring your backups on a regular basis.

Another point to consider is whether there is a single company, credential or key that, if compromised, could cause the destruction of all your backups. Although it is useful to have multiple media and servers, if a single hacker can destroy everything, you are vulnerable in the same way the two companies mentioned in the introduction were. Physical media that is rotated off-site is a good way to achieve this, or else a separate server with a completely different set of credentials.

zbackup makes it relatively simple

to encrypt the data stored in the backup. If you are storing your backups on insecure or third-party machines, you may want to use this facility. When managing backups for multiple servers, I prefer to encrypt the media where the backups are stored using LUKS. This includes the drives within the servers and the removable USB drives.

## Other Considerations

It is particularly important that you don't compress or encrypt your files as part of a process before you pass them to zbackup. Otherwise, you will find the deduplication will be completely ineffective. For example, Postgres allows you to compress your backups when writing the file. If this option were used, you would get no benefit from using zbackup.

In the architecture here, I have suggested doing the zbackup on each server rather than centralizing it. This means that although duplicates within a server are merged, duplicates between servers are not. For some applications, that may not be good enough. In this case, you might consider running zbackup on the virtualization host to deduplicate the disk files.

zbackup and tar are both stream-oriented protocols. This means that restoring a single file requires the computer to restore the entire backup and untar only the file you require. For small backups, this may be fine, but if your directory structures are very large, it may be worthwhile to back up directories individually rather than in one go. For example, you might choose to back up Web sites individually.

zbackup currently is limited by the speed at which the data can be read in and streamed to the deduplication process. A file must be read in full and then deduplicated even if it hasn't changed. This is roughly equivalent to `rsync -c` (that is, checksum the file content rather than just comparing the file metadata). To scale to really large data sizes, zbackup may need to incorporate some of the tar facilities within itself, so that if it can determine a file hasn't changed (by inode and metadata), it deduplicates the file without reading it.■

---

**David Barton is the Managing Director of OneIT, a company specializing in custom business software development. David has been using Linux since 1998 and managing the company's Linux servers for more than ten years.**

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

# High-Availability Storage with HA-LVM

Deploy a storage solution with zero downtime.

PETROS KOUTOUPIS

In recent years, there has been a trend in which data centers have been opting for commodity hardware and software over proprietary solutions. Why shouldn't they? It offers extremely low costs and the flexibility to build an ecosystem the way it is preferred. The only limitation is the extent of the administrator's imagination. However, a question needs to be asked: "How would such a customized solution compare to its proprietary and more costly counterpart?"

Open-source projects have evolved and matured enough to stay competitive and provide the same feature-rich solutions that include volume management, data snapshots, data deduplication and so on. Although an often overlooked and longtime-supported concept is high availability.

The idea behind high availability is simple: eliminate any single point of failure. This ensures that if a server node or a path to the underlying storage goes down (planned or unplanned), data requests still can be served. Now there are multiple layers to a storage-deployed solution that can be configured for high availability and that is why this article focuses strictly on HA-LVM.

## HA-LVM

High Availability Logical Volume Manager (HA-LVM) is an add-on to the already integrated LVM suite. It enables a failover configuration for shared volumes—that is, if one server in a cluster fails or is taken down for maintenance, the shared storage configuration will fail over to the secondary server where all I/O requests will resume, uninterrupted. An HA-LVM configuration is an active/passive configuration. This means that a single server accesses the shared storage at any one time. In many cases, this is an ideal approach, as some of advanced LVM features, such as snapshot and data deduplication, are not supported in an active/active environment (when more than one server accesses the shared storage).

A very important component to HA-LVM is the CLVM or Clustered LVM dæmon. When enabled, the CLVM dæmon prevents corruption of LVM metadata and its logical volumes, which occurs if multiple machines make overlapping changes. Although in an active/passive configuration, this becomes less of a concern. To accomplish this, the dæmon relies on a Distributed Lock Manager or DLM. The purpose

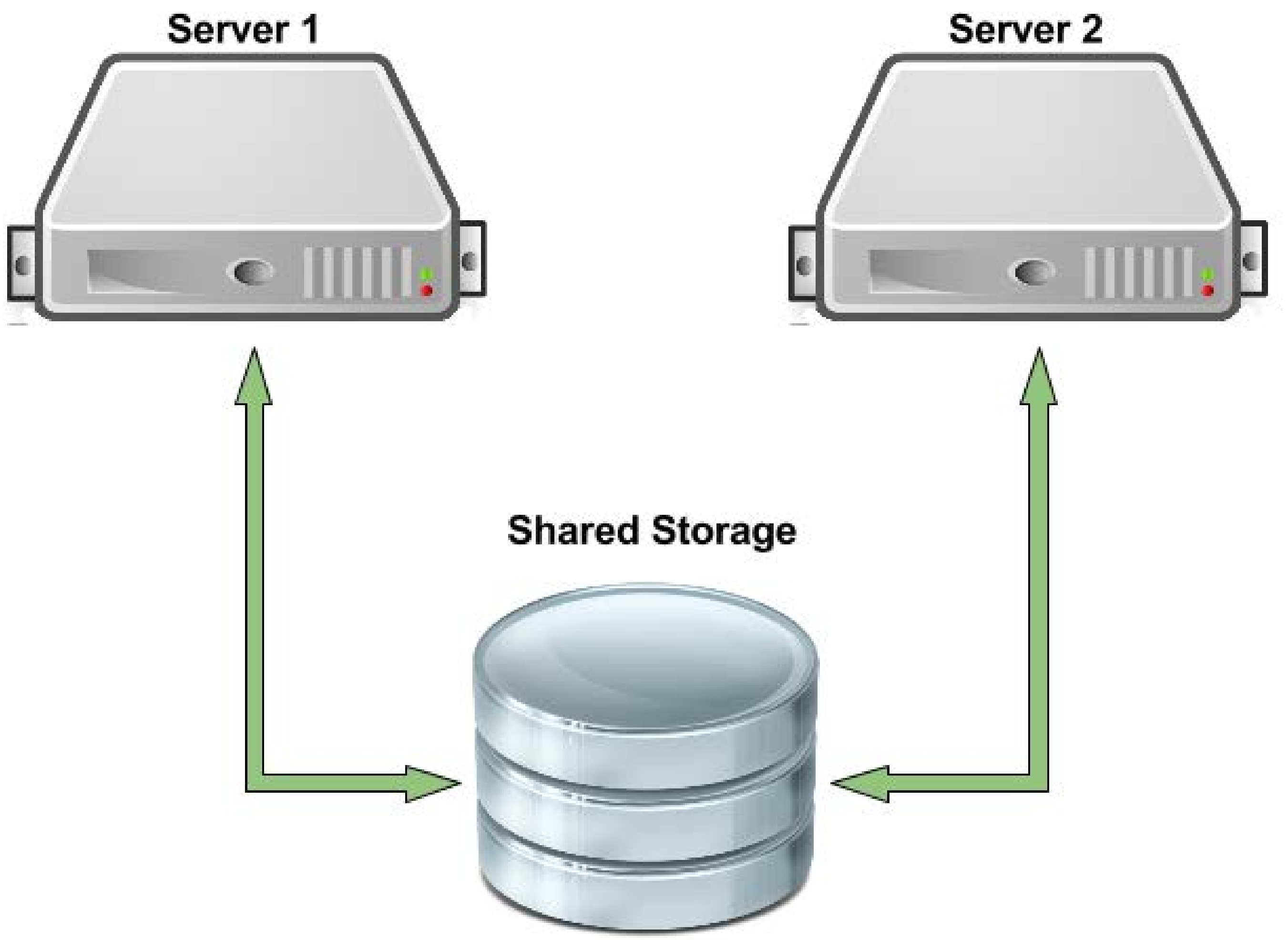


Figure 1. A Sample Configuration of Two Servers Accessing the Same Shared Storage

of the DLM is to coordinate disk access for CLVM.

The following example will cluster two servers that have access to the same external storage (Figure 1). This external storage could be a RAID-enabled or JBOD enclosure of disk drives, connected to the servers via a Fibre Channel, Serial Attached SCSI (SAS), iSCSI or other Storage

## CLVM

CLVM is not compatible with MD RAID, as it does not support clusters yet.

## CLVM Dæmon

The CLVM dæmon distributes LVM metadata updates across the cluster, and it must be running on all nodes in that cluster.

## JBOD

**A JBOD (or Just a Bunch Of Disks) is an architecture using multiple hard drives, but not in a redundant configuration.**

Area Network (SAN) mapping. The configuration is storage protocol-agnostic and requires only that the clustered servers see the same shared block devices.

### Configuring the Cluster

Almost all Linux distributions offer the required packages. However, the names may differ in each. You need to install lvm2-cluster (in some distributions, the package may be named clvm), the Corosync cluster engine, the Red Hat cluster manager (or cman), the Resource Group manager dæmon (or rgmanager) and all their dependencies on all participating servers. Even though the Red Hat cluster manager contains the Linux distribution of the same name in its package description, most modern distributions unrelated to Red Hat will list it in their repositories.

Once the appropriate clustering packages have been installed on all participating servers, the cluster configuration file must be configured to enable the cluster. To accomplish this, create and modify /etc/cluster/cluster.conf with the following:

```
<cluster name="lvm-cluster" config_version="1">
    <cman two_node="1" expected_votes="1" />
    <clusternodes>
        <clusternode name="serv-0001" nodeid="1">
            <fence>
            </fence>
        </clusternode>
        <clusternode name="serv-0002" nodeid="2">
            <fence>
            </fence>
        </clusternode>
    </clusternodes>
    <logging debug="on">
    </logging>
    <dlm protocol="tcp" timewarn="500">
    </dlm>
    <fencedevices>
    </fencedevices>
    <rm>
    </rm>
</cluster>
```

Note that the clusternode name is the server's hostname (change where necessary). Also, make sure the cluster.conf file is identical on all servers in the cluster.

The Red Hat cluster manager needs

**You now have a working cluster. The next step is to enable the Clustered LVM in High Availability mode.**

to be started:

```
$ sudo /etc/rc.d/init.d/cman start
Starting cluster:
   Checking if cluster has been disabled at boot...    [  OK  ]
   Checking Network Manager...                         [  OK  ]
   Global setup...                                     [  OK  ]
   Loading kernel modules...                           [  OK  ]
   Mounting configfs...                                [  OK  ]
   Starting cman...                                    [  OK  ]
   Waiting for quorum...                               [  OK  ]
   Starting fenced...                                  [  OK  ]
   Starting dlm_controld...                            [  OK  ]
   Tuning DLM kernel config...                         [  OK  ]
   Starting gfs_controld...                            [  OK  ]
   Unfencing self...                                   [  OK  ]
   Joining fence domain...                             [  OK  ]
```

If a single node in the cluster is not active, it will appear as off-line:

```
$ sudo clustat
Cluster Status for lvm-cluster @ Sun Aug  3 11:31:51 2014
Member Status: Quorate

 Member Name                    ID   Status
 ------ ----                    ---- ------
 serv-0001                           1 Online, Local
 serv-0002                           2 Offline
```

Otherwise, when all servers are configured appropriately and the cman service is enabled, all nodes will appear with an `Online` status:

```
$ sudo clustat
Cluster Status for lvm-cluster @ Sun Aug  3 11:36:43 2014
Member Status: Quorate

 Member Name                    ID   Status
 ------ ----                    ---- ------
 serv-0001                           1 Online
 serv-0002                           2 Online, Local
```

You now have a working cluster. The next step is to enable the Clustered LVM in High Availability mode. In this scenario, you have a single volume from the shared storage enclosure mapped to both servers. Both servers are able to observe and access this volume as /dev/sdb.

The /etc/lvm/lvm.conf file needs to be modified for this. The locking_type parameter in the global section has to be set to the value 3. It is set to

1 by default:

```
# Type of locking to use. Defaults to local file-based
# locking (1).
# Turn locking off by setting to 0 (dangerous: risks metadata
# corruption if LVM2 commands get run concurrently).
# Type 2 uses the external shared library locking_library.
# Type 3 uses built-in clustered locking.
# Type 4 uses read-only locking which forbids any operations
# that might change metadata.
locking_type = 3
```

On one of the servers, create a volume group, logical volume and filesystem from the designated shared volume:

```
$ sudo pvcreate /dev/sdb
$ sudo vgcreate -cy shared_vg /dev/sdb
$ sudo lvcreate -L 50G -n ha_lv shared_vg
$ sudo mkfs.ext4 /dev/shared_vg/ha_lv
$ sudo lvchange -an shared_vg/ha_lv
```

The example above carves out a 50GB logical volume from the volume group and then formats it with an Extended 4 filesystem. The `cy` option used with the `vgcreate` (volume group create) command enables the volume group for clustered locking. The an option with the `lvchange` (logical volume change) command deactivates the logical volume. You will be relying on the CLVM and resource manager (read below) dæmons to handle activations based on the failover feature additions made in the same /etc/cluster/cluster.conf file created earlier. When active, the the shared volume will be accessible from /dev/shared_vg/ha_lv.

Add the necessary failover details to the cluster.conf file:

```
<rm>
   <failoverdomains>
       <failoverdomain name="FD" ordered="1" restricted="0">
          <failoverdomainnode name="serv-0001" priority="1"/>
          <failoverdomainnode name="serv-0002" priority="2"/>
       </failoverdomain>
   </failoverdomains>
   <resources>
       <lvm name="lvm" vg_name="shared_vg" lv_name="ha-lv"/>
       <fs name="FS" device="/dev/shared_vg/ha-lv"
         ➥force_fsck="0" force_unmount="1" fsid="64050"
         ➥fstype="ext4" mountpoint="/mnt" options=""
         ➥self_fence="0"/>
   </resources>
   <service autostart="1" domain="FD" name="serv"
    ➥recovery="relocate">
       <lvm ref="lvm"/>
       <fs ref="FS"/>
   </service>
</rm>
```

The “rm” portion of the cluster.conf

**The purpose of the fencing agent is to handle a problematic node before it causes noticeable issues to the cluster.**

file utilizes the resource manager (or rgmanager). In this addition to the configuration file, you inform the cluster manager that serv-0001 should have ownership and sole access to the shared volume first. It will be mounted locally at the /mnt absolute path. If and when serv-0001 goes down for any reason, the resource manager then will perform a failover that will enable sole access to the shared volume, mounted at /mnt on serv-0002. All pending I/O requests sent to serv-0001 will resume on serv-0002.

On all servers, restart the cman service to enable the new configuration:

```
$ sudo /etc/rc.d/init.d/cman restart
```

Also, on all servers, start the rgmanager and clvmd services:

```
$ sudo /etc/rc.d/init.d/rgmanager start
Starting Cluster Service Manager:                    [  OK  ]
$ sudo /etc/rc.d/init.d/clvmd start
Starting clvmd:                                      [  OK  ]
```

Assuming that no errors were observed, you now should have a running cluster configured in an active/passive configuration. You can validate this by checking the accessibility of the shared volume on all servers. It should be seen, enabled and mounted on serv-0001 and not on serv-0002. Now comes the moment of truth—that is, testing the failover. Manually power down serv-0001. You will notice the rgmanager kicking in and enabling/mounting the volume on serv-0002.

**NOTE:** To enable these services automatically on reboot, use chkconfig to start the services on all appropriate runlevels.

## Summary

In an ideal configuration, fencing agents will need to be configured in the /etc/cluster/cluster.conf file. The purpose of the fencing agent is to handle a problematic node before it causes noticeable issues to the cluster. For example, if a server suffers from a kernel panic, is not communicating with the other servers in the cluster, or something else just as devastating, the IPMI utilities can be configured to reboot the server in question:

```
<clusternode name="serv-0001" nodeid="1">
     <fence>
         <method name="1">
             <device name="ipmirecover1"/>
         </method>
     </fence>
 </clusternode>
 <clusternode name="serv-0002" nodeid="2">
     <fence>
         <method name="1">
             <device name="ipmirecover2"/>
         </method>
     </fence>
 </clusternode>

[ ... ]

   <fencedevices>
       <fencedevice agent="fence_ipmilan" ipaddr="192.168.1.50"
        ➥login="ADMIN" passwd="ADMIN" name="ipmirecover1"/>
       <fencedevice agent="fence_ipmilan" ipaddr="192.168.10"
        ➥login="ADMIN" passwd="ADMIN" name="ipmirecover2"/>
   </fencedevices>
```

The primary objective of HA-LVM is to provide the data center with enterprise-class fault tolerance at a fraction of the price. No one ever wants to experience server downtimes, and with an appropriate configuration, no one has to. From the data center to your home office, this solution can be deployed almost anywhere.■

---

**Petros Koutoupis is a full-time Linux kernel, device driver and application developer for embedded and server platforms. He has been working in the data storage industry for more than eight years and enjoys discussing the same technologies.**

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

## Resources

clvmd(8): Linux man page

Appendix F. High Availability LVM (HA-LVM): **https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Cluster_Administration/ap-ha-halvm-CA.html**

# Sharing Admin Privileges for Many Hosts Securely

**The ssh-agent program can hold your decrypted authentication keys in memory. This makes a lot of things possible—one of them is controlling shared accounts on large numbers of hosts.**

**J. D. BALDWIN**

The problem: you have a large team of admins, with a substantial turnover rate. Maybe contractors come and go. Maybe you have tiers of access, due to restrictions based on geography, admin level or even citizenship (as with some US government contracts). You need to give these people administrative access to dozens (perhaps hundreds) of hosts, and you can't manage all their accounts on all the hosts.

This problem arose in the large-scale enterprise in which I work, and our team worked out a solution that:

- Does *not* require updating accounts on more than one host whenever a team member arrives or leaves.

- Does *not* require deletion or replacement of Secure Shell (SSH) keys.

- Does *not* require management of individual SSH keys.

- Does *not* require distributed sudoers or other privileged-access management tools (which may not be supported by some Linux-based appliances anyway).

- And most important, does *not* require sharing of passwords or key passphrases.

It works between any UNIX or Linux platforms that understand SSH key trust relationships. I personally have made use of it on a half-dozen different Linux distros, as well as Solaris, HP-UX, Mac OS X and some BSD variants.

In our case, the hosts to be managed were several dozen Linux-based special-purpose appliances that did not support central account management tools or sudo. They are intended to be used (when using the shell at all) as the root account.

Our environment also (due to a government contract) requires a two-tier access scheme. US citizens on the team may access any host as root. Non-US citizens may access only a subset of the hosts. The techniques described in this article may be extended for *N* tiers without any real trouble, but I describe the case N == 2 in this article.

## The Scenario

I am going to assume you, the reader, know how to set up an SSH trust relationship so that an account on one host can log in directly, with no password prompting, to an account on another. (Basically, you simply create a key pair and copy the public half to the remote host's ~/.ssh/authorized_keys file.) If you don't know how to do this, stop reading now and go learn. A Web search for "ssh trust setup" will yield thousands of links—or, if you're old-school, the AUTHENTICATION section of the ssh(1) man page will do. Also see ssh-copy-id(1), which can greatly simplify the distribution of key files.

Steve Friedl's Web site has an excellent Tech Tip on these basics, plus some material on SSH agent-forwarding, which is a neat trick to centralize SSH authentication for an individual user. The Tech Tip is available at **http://www.unixwiz.net/techtips/ssh-agent-forwarding.html**.

I describe key-caching below, as it is not very commonly used and is the heart of the technique described herein.

For illustration, I'm assigning names to players (individuals assigned to roles), the tiers of access and "dummy" accounts.

Hosts:

- darter — the hostname of the central management host on which all the end-user and utility accounts are active, all keys are stored and caching takes place; also, the sudoers file controlling access to utility accounts is here.
- n1, n2, ... — hostnames of target hosts for which access is to be granted for *all* team members ("n" for "non-special").
- s1, s2, ... — hostnames of target hosts for which access is to be granted only to *some* team members ("s" for "special").

Accounts (on darter only):

- univ — the name of the utility account holding the SSH keys that all target hosts (u1, u2, ...) will trust.
- spec — the name of the utility account holding the SSH keys that only special, restricted-access, hosts (s1, s2, ...) will trust.
- joe — let's say the name of the guy administering the whole scheme is "Joe" and his account is "joe". Joe is a trusted admin with "the keys to the kingdom"—he cannot be a restricted user.
- andy, amy — these are users who are allowed to log in to all hosts.
- alice

- ned, nora — these are users who are allowed to log in *only* to "n" (non-special) hosts; they never should be allowed to log in to special hosts s1, s2, ...

- nancy

You will want to create shared, unprivileged utility accounts on darter for use by unrestricted and restricted admins. These (per our convention) will be called "univ" and "rstr", respectively. No one should actually directly log in to univ and rstr, and in fact, these accounts should not have passwords or trusted keys of their own. All logins to the shared utility accounts should be performed with su(1) from an existing individual account on darter.

## The Setup

Joe's first act is to log in to darter and "become" the univ account:

```
$ sudo su - univ
```

Then, under that shared utility account, Joe creates a .ssh directory and an SSH keypair. This key will be trusted by the root account on every target host (because it's the "univ"-ersal key):

```
$ mkdir .ssh    # if not already present
$ ssh-keygen -t rsa -b 2048 -C "universal access
 ➥key gen YYYYMMDD" -f
.ssh/univ_key
   Enter passphrase (empty for no passphrase):
```

*Very important:* Joe assigns a strong passphrase to this key. The passphrase to this key will not be generally shared.

(The field after `-C` is merely a comment; this format reflects my personal preference, but you are of course free to develop your own.)

This will generate two files in .ssh: univ_key (the private key file) and univ_key.pub (the public key file). The private key file is encrypted, protected by the very strong passphrase Joe assigned to it, above.

Joe logs out of the univ account and into rstr. He executes the same steps, but creates a keypair named rstr_key instead of univ_key. He assigns a strong passphrase to the private key file—it can be the same passphrase as assigned to univ, and in fact, that is probably preferable from the standpoint of simplicity.

Joe copies univ_key.pub and rstr_key.pub to a common location for convenience.

For every host to which access is granted for everyone (n1, n2, ...), Joe uses the target hosts' root credentials to copy *both* univ_key.pub and

**Any user who uses SSH keys whose key files are protected by a passphrase may cache those keys using a program called ssh-agent.**

rstr_key.pub (on separate lines) to the file .ssh/authorized_keys under the root account directory.

For every host to which access is granted for only a few (s1, s2, ...), Joe uses the target hosts' root credentials to copy *only* rstr_key.pub (on a single line) to the file .ssh/authorized_keys under the root account directory.

So to review, now, when a user uses `su` to "become" the univ account, he or she can log in to *any* host, because univ_key.pub exists in the authorized_keys file of n1, n2, ... and s1, s2, ....

However, when a user uses `su` to "become" the rstr account, he or she can log in *only* to n1, n2, ..., because those hosts' authorized_keys files contain rstr_key.pub, but *not* univ_key.pub.

Of course, in order to unlock the access in both cases, the user will need the strong passphrase with which Joe created the keys. That seems to defeat the whole purpose of the scheme, but there's a trick to get around it.

### The Trick

First, let's talk about key-caching. Any user who uses SSH keys whose key files are protected by a passphrase may cache those keys using a program called ssh-agent. ssh-agent does not take a key directly upon invocation. It is invoked as a standalone program without any parameters (at least, none useful to us here).

The output of ssh-agent is a couple environment variable/value pairs, plus an echo command, suitable for input to the shell. If you invoke it "straight", these variables will not become part of the environment. For this reason, ssh-agent always is invoked as a parameter of the shell built-in `eval`:

```
$ eval $(ssh-agent)
Agent pid 29013
```

(The output of `eval` also includes an echo statement to show you the PID of the agent instance you just created.)

Once you have an agent running, and your shell knows how to communicate with it (thanks to the environment variables), you may cache keys with it using the command `ssh-add`. If you give `ssh-add` a key file, it will prompt you for the passphrase. Once you provide the correct passphrase, ssh-agent will hold the unencrypted key in memory. Any invocation of SSH will check with ssh-agent before attempting authentication. If the key in memory matches the public key on the remote host, trust is established, and the login simply happens with no entry of passwords or passphrases.

(As an aside: for those of you who use the Windows terminal program PuTTY, that tool provides a key-caching program called Pageant, which performs much the same function. PuTTY's equivalent to ssh-keygen is a utility called PuTTYgen.)

All you need to do now is set it up so the univ and rstr accounts set themselves up on every login to make use of persistent instances of ssh-agent. Normally, a user manually invokes ssh-agent upon login, makes use of it during that session, then kills it, with `eval $(ssh-agent -k)`, before exiting. Instead of manually managing it, let's write into each utility account's .bash_profile some code that does the following:

1. First, check whether there is a current instance of ssh-agent for the current account.

2. If not, invoke ssh-agent and capture the environment variables in a special file in /tmp. (It should be in /tmp because the contents of /tmp are cleared between system reboots, which is important for managing cached keys.)

3. If so, find the file in /tmp that holds the environment variables and source it into the shell's environment. (Also, handle the error case where the agent is running and the /tmp file is not found by killing ssh-agent and starting from scratch.)

All of the above assumes the key already has been unlocked and cached. (I will come back to that.)

Here is what the code in .bash_profile looks like for the univ account:

```
/usr/bin/pgrep -u univ 'ssh-agent' >/dev/null

RESULT=$?

if [[ $RESULT -eq 0 ]]  # ssh-agent is running
then
    if [[ -f /tmp/.env_ssh.univ ]]   # bring env in to session
    then
```

```
        source /tmp/.env_ssh.univ
    else    # error condition
        echo 'WARNING:  univ ssh agent running, no environment
         ➥file found'
        echo '          ssh-agent being killed and restarted ... '
        /usr/bin/pkill -u univ 'ssh-agent' >/dev/null
        RESULT=1     # due to kill, execute startup code below
    fi

if [[ $RESULT -ne 0 ]] # ssh-agent not running, start
 ➥it from scratch
then
    echo "WARNING:  ssh-agent being started now;
     ➥ask Joe to cache key"
    /usr/bin/ssh-agent > /tmp/.env_ssh.univ
    /bin/chmod 600 /tmp/.env_ssh.univ
    source /tmp/.env_ssh.univ
fi
```

And of course, the code is identical for the rstr account, except s/univ/rstr/ everywhere.

Joe will have to intervene *once* whenever darter (the central management host on which all the user accounts and the keys reside) is restarted. Joe will have to log on and become univ and execute the command:

```
$ ssh-add ~/.ssh/univ_key
```

and then enter the passphrase. Joe then logs in to the rstr account and executes the same command against ~/.ssh/rstr_key. The command `ssh-add -l` lists cached keys by their fingerprints and filenames, so if there is doubt about whether a key is cached, that's how to find out. A single agent can cache multiple keys, if you have a use for that, but it doesn't come up much in my environment.

Once the keys are cached, they will stay cached. (`ssh-add -t <N>` may be used to specify a timeout of *N* seconds, but you won't want to use that option for this shared-access scheme.) The cache must be rebuilt for each account whenever darter is rebooted, but since darter is a Linux host, that will be a rare event. Between reboots, the single instance (one per utility account) of ssh-agent simply runs and holds the key in memory. The last time I entered the passphrases of our utility account keys was more than 500 days ago—and I may go several hundred more before having to do so again.

The last step is setting up sudoers to manage access to the utility accounts. You don't *really* have to do this. If you like, you can set (different) passwords for univ and rstr and simply let the users hold them. Of course, shared passwords aren't a great idea to begin with. (That's one of the major points of this whole scheme!) Every time one of the users of the univ account leaves the team, you'll have to change that password and distribute the new one (hopefully securely and out-of-band) to all the remaining users.

No, managing access with sudoers is a better idea. This article isn't here to teach you all of—or any of—the ins and outs of sudoers' Extremely Bizarre Nonsensical Frustration (EBNF) syntax. I'll just give you the cheat code.

Recall that Andy, Amy, Alice and so on were all allowed to access all hosts. These users are permitted to use sudo to execute the `su - univ` command. Ned, Nora, Nancy and so on are permitted to access only the restricted list of hosts. They may log in only to the rstr account using the `su - rstr` command. The sudoers entries for these might look like:

```
User_Alias  UNIV_USERS=andy,amy,alice,arthur        # trusted

User_Alias  RSTR_USERS=ned,nora,nancy,nyarlathotep  # not so much


# Note that there is no harm in putting andy, amy, etc. into

# RSTR_USERS as well. But it also accomplishes nothing.


Cmnd_Alias  BECOME_UNIV = /bin/su - univ

Cmnd_Alias  BECOME_RSTR = /bin/su - rstr


UNIV_USERS   ALL= BECOME_UNIV

RSTR_USERS   ALL= BECOME_RSTR
```

Let's recap. Every host n1, n2, n3 and so on has *both* univ and rstr key files in authorized_keys.

Every host s1, s2, s3 and so on has *only* the univ key file in authorized_keys.

When darter is rebooted, Joe logs in to both the univ and rstr accounts and executes the ssh-add command with the private key file as a parameter. He enters the passphrase for these keys when prompted.

Now Andy (for example) can log in to darter, execute:

```
$ sudo su - univ
```

and authenticate with his password. He now can log in as root to *any* of n1, n2, ..., s1, s2, ... without further authentication. If Andy needs to check the functioning of ntp (for example) on each of 20 hosts, he can execute a loop:

```
$ for H in n1 n2 n3 [...] n10 s1 s2 s3 [...] s10

> do

>    ssh -q root@$H 'ntpdate -q timeserver.domain.tld'

> done
```

and it will run without further intervention.

Similarly, nancy can log in to darter, execute:

```
$ sudo su - rstr
```

and log in to any of n1, n2 and so on, execute similar loops, and so forth.

## Benefits and Risks

Suppose Nora leaves the team. You simply would edit sudoers to delete her from RSTR_USERS, then lock or delete her system account.

"But Nora was fired for misconduct! What if she kept a copy of the keypair?"

The beauty of this scheme is that access to the two key files *does not matter*. Having the public key file isn't important—put the public key file on the Internet if you want. It's public!

Having the encrypted copy of the private key file doesn't matter. Without the passphrase (which only Joe knows), that file may as well be the output of /dev/urandom. Nora never had access to the raw key file—only the caching agent did.

Even if Nora kept a copy of the key files, she cannot use them for access. Removing her access to darter removes her access to every target host.

And the same goes, of course, for the users in UNIV_USERS as well.

There are two caveats to this, and make sure you understand them well.

Caveat the first: it (almost) goes without saying that anyone with root access to darter obviously can just become root, then `su - univ` at any time. If you give someone root access to darter, you are giving that person full access to all the target hosts as well. That, after all, is the meaning of saying the target hosts "trust" darter. Furthermore, a user with root access who does not know the passphrase to the keys still can recover the raw keys from memory with a little moderately sophisticated black magic. (Linux memory architecture and clever design of the agent prevent non-privileged users from recovering their own agents' memory contents in order to extract keys.)

Caveat the second: obviously, anyone holding the passphrase can make (and keep) an unencrypted copy of the private keys. In our example, only Joe had that passphrase, but in practice, you will want two or three trusted admins to know the passphrase so they can intervene to re-cache the keys after a reboot of darter.

If anyone with root access to your central management host (darter, in this example) *or* anyone holding private key passphrases should leave the team, you will have to generate new keypairs and replace the contents of authorized_keys on every target host in your enterprise. (Fortunately, if you are careful, you can use the old trust relationship to create the new one.)

For that reason, you will want to entrust the passphrase only to individuals whose positions on your team are at least reasonably stable. The techniques described in this article are probably not suitable for a high-turnover environment with no stable "core" admins.

One more thing about this: you don't need to be managing tiered or any kind of shared access for this basic trick to be useful. As I noted above, the usual way of using an SSH key-caching agent is by invoking it at session start, caching your key, then killing it before ending your session. However, by including the code above in your own .bash_profile, you can create your own file in /tmp, check for it, load it if present and so on. That way, the host always has just one instance of ssh-agent running, and your key is cached in it permanently (or until the next reboot, anyway).

Even if you don't want to cache your key *that* persistently, you still can make use of a single ssh-agent and cache your key with the timeout (-t) option mentioned earlier; you still will be saving yourself a step.

Note that if you do this, however, anyone with root on that host will have access to any account of yours that trusts your account on that machine—so *caveat actor*. (I use this trick only on personal boxes that only I administer.)

The trick for personal use is becoming obsolete, as Mac OS X (via SSHKeyChain) and newer versions of GNOME (via Keyring) automatically *know* the first time you SSH to a host with which you have a key-based authentication set up, then ask you your passphrase and cache the key for the rest of your GUI login session. Given the lack of default timeouts and warnings about root users' access to unlocked keys, I am not sure this is an unmixed technological advance. (It is possible to configure timeouts in both utilities, but it requires that users find out about the option, and take the effort to configure it.)

## Acknowledgements

I gratefully acknowledge the technical review and helpful suggestions of David Scheidt and James Richmond in the preparation of this article.■

**J.D. Baldwin has been a UNIX, Linux and Web user and administrator going back to SunOS 1.1 (1984), Slackware 3.0 (1995) and Apache 1.2 (1997). He currently works in network security for a large multinational company. J.D. is a graduate and former faculty member of the US Naval Academy and has an MS in Computer Science from the University of Maryland. He lives with his wife in their empty nest in southwest Michigan. You can reach him at baldwin@panix.com.**

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

## WEBCASTS

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

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

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

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

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

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

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

## WHITE PAPERS

### White Paper: JBoss Enterprise Application Platform for OpenShift Enterprise

**Sponsor:** DLT Solutions

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

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

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

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

## WHITE PAPERS

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

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

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

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

### Standardized Operating Environments for IT Efficiency

**Sponsor:** Red Hat

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

**Benefits of an SOE:**

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

**SOE leads to:**

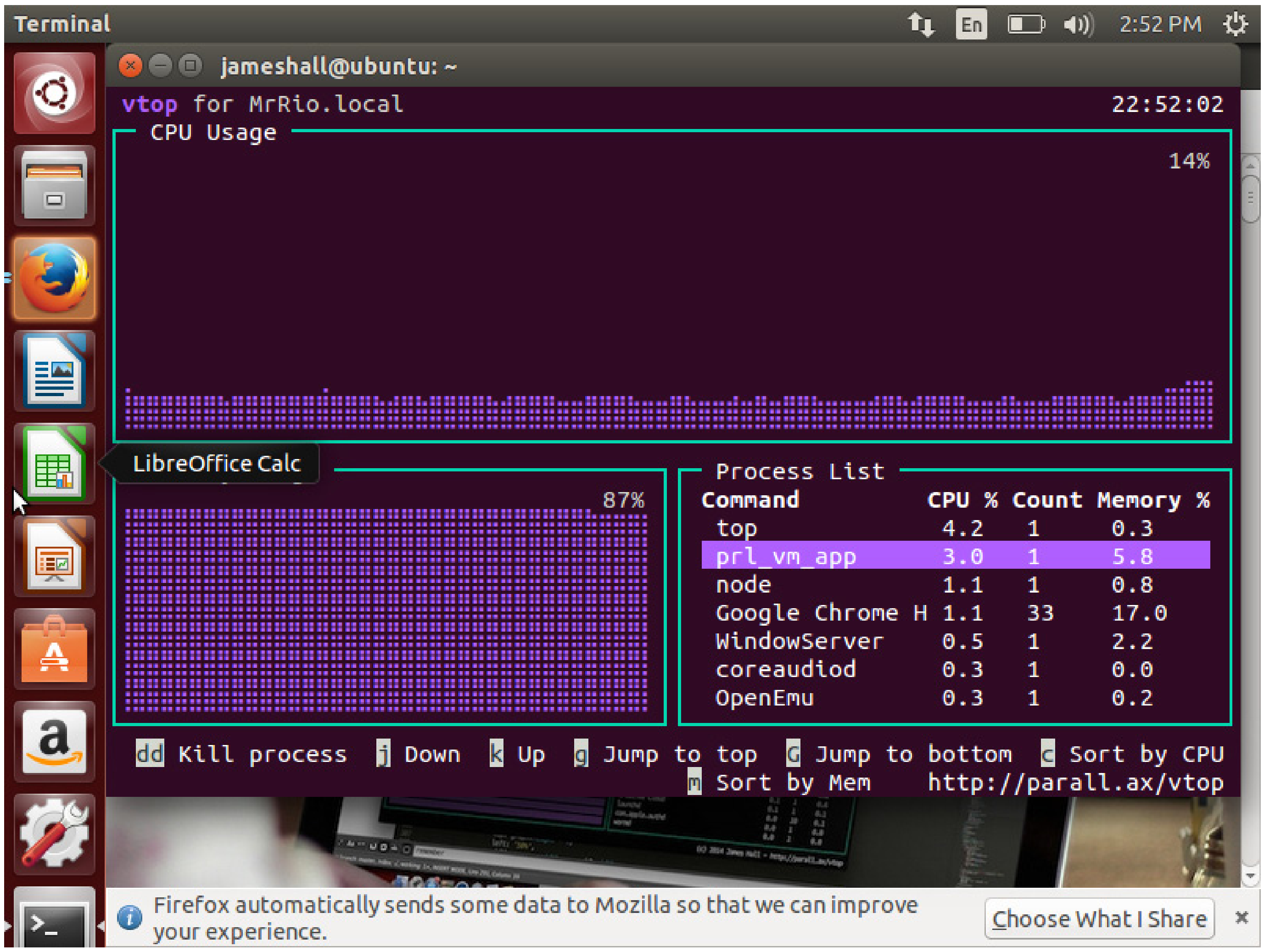
- Dramatically reduced deployment time.
- Software deployed and configured in a standardized manner.
- Simplified maintenance due to standardization.
- Increased stability and reduced support and management costs.
- There are many benefits to having an SOE within larger environments, such as:
  - Less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

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

# Rethinking the System Monitor

**vtop is a graphical activity monitor for the command line. In this article, I take you through how I wrote the app, how it works underneath and invite you to help extend it.**

JAMES HALL



```
vtop for MrRio.local                                          22:52:02
 CPU Usage
                                                                  14%

                                     87%    Process List
                                            Command         CPU % Count Memory %
                                             top             4.2   1       0.3
                                             prl_vm_app      3.0   1       5.8
                                             node            1.1   1       0.8
                                             Google Chrome H 1.1   33      17.0
                                             WindowServer    0.5   1       2.2
                                             coreaudiod      0.3   1       0.0
                                             OpenEmu         0.3   1       0.2

 dd Kill process  j Down  k Up  g Jump to top  G Jump to bottom  c Sort by CPU
                                          m Sort by Mem    http://parall.ax/vtop
```

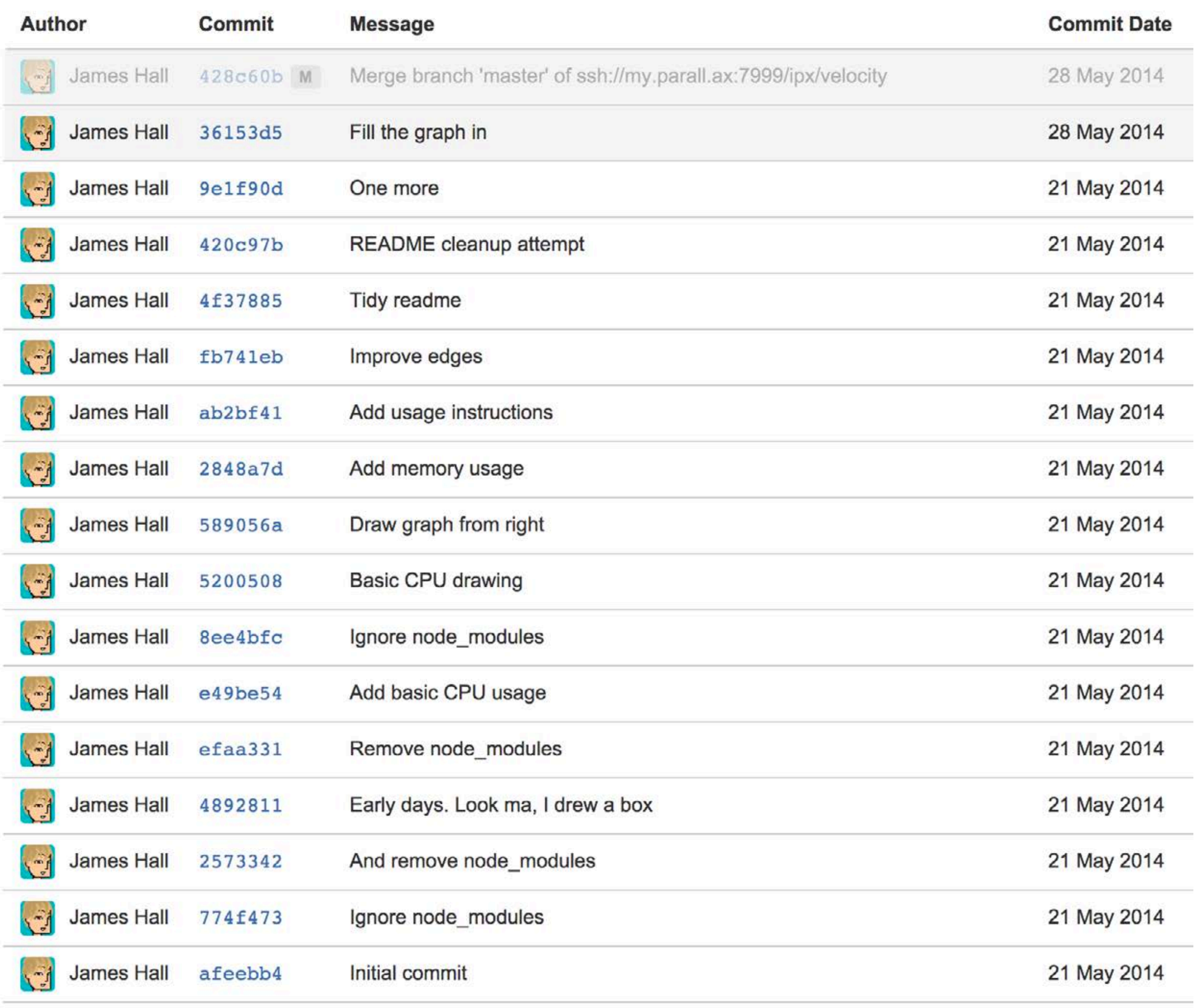
Firefox automatically sends some data to Mozilla so that we can improve your experience.

Figure 1. vtop Running on Ubuntu

**System monitoring tools** have been with us since the early days of computing, but on the terminal, many people still use the top command. Now, let me introduce you to my open-source activity monitor called vtop. It uses Unicode Braille characters for richer visualization on the command line.

## Background

For many, the top command has been a key way to monitor rogue processes on *nix systems. William LeFebvre wrote the original top command more than 30 years ago on a Vax running BSD UNIX. He was inspired by the Vax VMS operating system that listed the most CPU-hungry processes along



| Author | Commit | Message | Commit Date |
|---|---|---|---|
| James Hall | 428c60b M | Merge branch 'master' of ssh://my.parall.ax:7999/ipx/velocity | 28 May 2014 |
| James Hall | 36153d5 | Fill the graph in | 28 May 2014 |
| James Hall | 9e1f90d | One more | 21 May 2014 |
| James Hall | 420c97b | README cleanup attempt | 21 May 2014 |
| James Hall | 4f37885 | Tidy readme | 21 May 2014 |
| James Hall | fb741eb | Improve edges | 21 May 2014 |
| James Hall | ab2bf41 | Add usage instructions | 21 May 2014 |
| James Hall | 2848a7d | Add memory usage | 21 May 2014 |
| James Hall | 589056a | Draw graph from right | 21 May 2014 |
| James Hall | 5200508 | Basic CPU drawing | 21 May 2014 |
| James Hall | 8ee4bfc | Ignore node_modules | 21 May 2014 |
| James Hall | e49be54 | Add basic CPU usage | 21 May 2014 |
| James Hall | efaa331 | Remove node_modules | 21 May 2014 |
| James Hall | 4892811 | Early days. Look ma, I drew a box | 21 May 2014 |
| James Hall | 2573342 | And remove node_modules | 21 May 2014 |
| James Hall | 774f473 | Ignore node_modules | 21 May 2014 |
| James Hall | afeebb4 | Initial commit | 21 May 2014 |

Figure 2. A Flurry of Early Commits

**The original vtop was a quick hack, mostly written in a day, and like all the best open-source software, it scratched an itch.**

with an ASCII bar chart. The bar chart didn't make it across into his version; the author went instead for a text-based approach to displaying data that has stuck with us.

While the GUI-world enjoys increasingly feature-rich tools, terminal applications sadly have lagged behind. Graphical representations in system monitoring tools are nothing new. KSysguard and GNOME's System Monitor sport fancy graphs and charts, but this isn't much use to us on the command line.

Although there's absolutely nothing wrong with top's text-based approach, it's not what I needed when I set out to write vtop. The original vtop was a quick hack, mostly written in a day, and like all the best open-source software, it scratched an itch. I needed to see CPU spikes to debug some strange behaviour, and I couldn't use the graphical tools for Linux, because I didn't want to install all that bloat on my servers. Just looking at the numbers in top doesn't give you much of an idea of how it's fluctuating over time.

I started hashing out the initial version, not worrying too much about the tidiness of the code (I was trying to debug a problem quickly after all). I ended up getting carried away with it, and I almost forgot to go back and debug my original issue.

I ran the code on the remote server and was delighted at how immediately useful it was, even in its crude and ugly form. I committed the code and showed it to my colleagues at work. The reaction was a mixture of delight ("How do you even do that?") and horror (at my sloppy programming <blush>), but I knew this idea had legs.

## Write One to Throw Away

Worrying too much about the architecture early can be a waste of time. It's usually best to write one to throw away, and this code base certainly needed binning. The best structure for the application was far more obvious once I had a working prototype.

I sketched out what I thought it should look like: a large area at the top for CPU usage, then two smaller boxes for memory and a process list. I started a new project and got to work.

I decided to write vtop using Node.js. It's built on Chrome's V8 JavaScript engine and allows you to write fast and scalable applications. This choice could pave the way for a Web-based front end to be added in the future. JavaScript is coming into its own—it's no longer the scrappy, badly implemented language that everyone used to make sparkles follow their cursors on their Geocities pages. Node.js has evolved the language—it's now a fully formed toolchain with a thriving community. There's a Node package for just about anything you can think of; you really can hit the ground running by picking up other people's modules instead of writing from scratch.

At the beginning of the rewrite, I made an outline using simple box drawing characters that I used to love playing with in my early DOS programming days. Although this worked okay, I felt there might be an easier way. I'd seen ncurses and wondered if there was anything more modern kicking about. I eventually came across Blessed.

Blessed abstracts away the complexities of drawing a GUI in the terminal. You tell it where to draw boxes, and they are resized automatically based on the terminal width and height. You also can listen to scroll wheel and click events to enable even easier interaction. I highly recommend checking it out.

I created a couple boxes in Blessed and populated the text content of the first one with the Braille characters. Then I easily was able to add different colors to the app.

## Design Goals

The rewrite forced me to think about my design goals for the project. I was keen to have other developers get involved, and hopefully, it can be used for purposes I never imagined. The design goals can be distilled to these three:

1. Extendible: plugins should be easy and quick to write, with clear separation of UI code and data collection code. (There's still a little work to do in this area.)

2. Accessible: when it comes to servers, the terminal rules the roost, and nothing beats the convenience of being able to dive straight in over SSH and fire up a command. That's not to say

that a Web-based GUI would be unwelcome, but each feature should work from the command line too.

3. Visual: it should take advantage of the latest and greatest techniques—a visually appealing interface using color and Unicode characters to great effect.

## Braille Display

Terminals have come a long way since the early days. xterm added 256-color support (which is just a sequence of escape codes printed out as text) and mouse support (which is your terminal sending text escape codes). Pretty much all terminal emulators support Unicode now, and in vtop, we use this to our advantage.

Unicode Braille characters give you a convenient 8x2 grid of dots in every possible combination, starting at Unicode point 0x2800. We can use these as faux-pixels. You take a grid of coordinates, and break it up into chunks for each character, and then just output them to the screen like you would any other text. There are 256 combinations (two states—on and off for each of the eight dots, which is $2^8$), and you can calculate which character you need by combining the hexadecimal numbers for each Braille dot and adding that to the starting point. Below are Braille Characters Representing a Slope on a Graph:

```
.
. .
. .   .
. .   . .
```

See **http://jsfiddle.net/MrRio/90vdrs01/3/**.

For example, the first character above would be 0x1 + 0x2 + 0x4 +

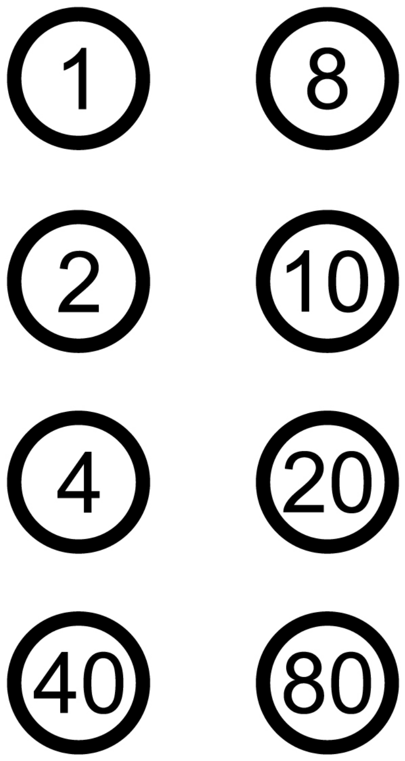


**Figure 3. Hexadecimal Values for Each Braille Dot (Public Domain)**

0x40 + 0x10 + 0x20 + 0x80 = 0xF7, then add this to the base of 0x2800 to get 0x28F7. You can try this in your browser's JavaScript panel:

```
String.fromCharCode(0x1 + 0x2 + 0x4 + 0x40 + 0x10
➥+ 0x20 + 0x80 + 0x2800);
```

There's a brilliant Node.js library that abstracts away this detail for you called node-drawille. It allows you to plot onto a canvas and returns Braille characters using the same method described here.

## Other Features

The main feature is the graphical interface, but it has a few other tricks up its sleeve:

- Vim-like keybindings: if you use vim, your muscle memory is tied to its keyboard shortcuts. Type j/k to move up and down the list and h/l to change the scale on the graphs. The arrow keys work fine too!

- Grouped processes: vtop will group together processes with the same name. Many applications are multiprocess—for example, Google Chrome spawns a new process for each tab to increase stability and security. You can get a true overall value of the CPU percentage it's taking up. It's also great for monitoring Web servers like Apache and nginx.

- Killing processes: simply type `dd` to make a process die. This is also the vim shortcut for deleting a line.

- Sorting by CPU or memory: typing `c` will sort the list by CPU; no prizes for guessing which key you press to sort by memory.

## Installation

Simply install npm with your favourite package manager. Then to install the command globally, just type:

```
npm -g install vtop
```

Upgrade notifications appear within the application, and it can be upgraded with a single key stroke.

## Contributing

**Getting Started with the Codebase:** First off, start by forking the project on GitHub: **https://github.com/MrRio/vtop**.

One you've got your own fork, you can clone the source from GitHub (make sure to replace "MrRio" with your own GitHub user name):

```
git clone git@github.com:MrRio/vtop.git
cd vtop
make
./bin/vtop.js
```

The last command runs your development version of vtop rather than the globally installed one.

Now you can begin hacking with the code.

To give you an idea of where to start, let me guide you through the main areas of the application. The entry point for the application is bin/vtop.js. This is a hybrid JS file and shell executable. It first runs as a shell script, detects the name of the node executable (which differs depending on the platform), enables xterm-256color and then runs itself as JavaScript. It then includes the main app.js file in the root.

Then the app.js file loads in the required libraries, the most important of which are Drawille for the Braille output, Blessed for the GUI and commander, which is used to parse command-line options. It then globs the themes/ directory for a list of themes and loads itself up via the init() function.

- drawHeader is responsible for drawing the title bar, with the time and any update notifications.

- drawFooter prints all the available commands across the footer and a link to the Web site.

- drawChart is responsible for drawing Braille charts, and drawTable for the process list, although this could do with refactoring into new files to allow for more display options to be contributed.

Sensors are loaded in from the sensors/ folder and polled at the desired frequency. Then the draw methods take this data and push it on to the screen.

**Themes:** A theme is a simple JSON file containing foreground and background colors for each element. Simply bob your theme into the themes/ directory, and then run `vtop -theme yourtheme`. Send a Pull Request, and as long as it isn't too similar to another theme, we'll include it.

The themes files are broken up per component and handed straight over to Blessed's style parameter for each component. It's possible to change the characters used for the box lines, or even add

**Sensors may need extending with more properties and methods depending on the kinds of things people want to build with them.**

bold and underline (check out the Blessed documentation at **https://github.com/chjj/blessed** for more information):

```
{
 "name": "Brew",
 "author": "James Hall",
 "title": {
  "fg": "#187dc1"
 },
 "chart": {
  "fg": "#187dc1",
  "border": {
   "type": "line",
   "fg": "#56a0d1"
  }
 },
 "table": {
  "fg": "fg",
  "items": {
   "selected": {
    "bg": "#56a0d1",
    "fg": "bg"
   },
   "item": {
    "fg": "fg",
    "bg": "bg"
   }
  },
  "border": {
   "type": "line",
   "fg": "#56a0d1"
  }
 },
 "footer": {
  "fg": "fg"
 }
}
```

**Sensors:** vtop currently has three sensors, CPU, Memory and Process List. A sensor has a title, a type (which decides the kind of renderer to use), a polling frequency with a function and a currentValue. The sensors know nothing about the UI, and their sole job is to output a single number or a list for the table type. vtop then takes this information and plots it out.

Sensors may need extending with more properties and methods depending on the kinds of things people want to build with them. For example, an Apache req/s sensor may need to be able to report its largest value, so vtop can adjust the scale, or the memory sensor could be extended to report multiple values for used, buffered, cached and free memory.

The following is an example sensor file—as you can see, they're pretty straightforward to write. Why not try modifying the file to have it report something else:

```
/**
 * CPU Usage sensor
 *
 * (c) 2014 James Hall
 */
var os = require('os-utils');
var plugin = {
 /**
 * This appears in the title of the graph
 */
 title: 'CPU Usage',
 /**
 * The type of sensor
 * @type {String}
 */
 type: 'chart',
 /**
 * The default interval time in ms that this plugin
 * should be polled. More costly benchmarks should
 * be polled less frequently.
 */
 interval: 200,
 initialized: false,
 currentValue: 0,
 /**
 * Grab the current value, from 0-100
 */
 poll: function() {
   os.cpuUsage(function(v){
     plugin.currentValue = (Math.floor(v * 100));
     plugin.initialized = true;
   });
  }
};
module.exports = exports = plugin;
```

If you have a basic understanding of JS, you can see how simple building a sensor really is. If you can give vtop a number, it can plot it. You could get these from existing npm modules or by parsing output of other Linux command-line utilities.

## Submitting a Pull Request

There are many tutorials on the Internet for getting started with Git (the **http://git-scm.com** Web site is good). It's much less scary than you think.

For features, simply make a branch called "feature/name-of-feature" and for bugfixes, "bugfix/name-of-fix". Don't worry about getting it perfect first time. Send your code in early for feedback, and people will help you refine it and get the code into the master branch.

I look forward to seeing what you come up with!

## Other Monitoring Software

There's more than one way to skin a cat, and this is especially true on Linux.

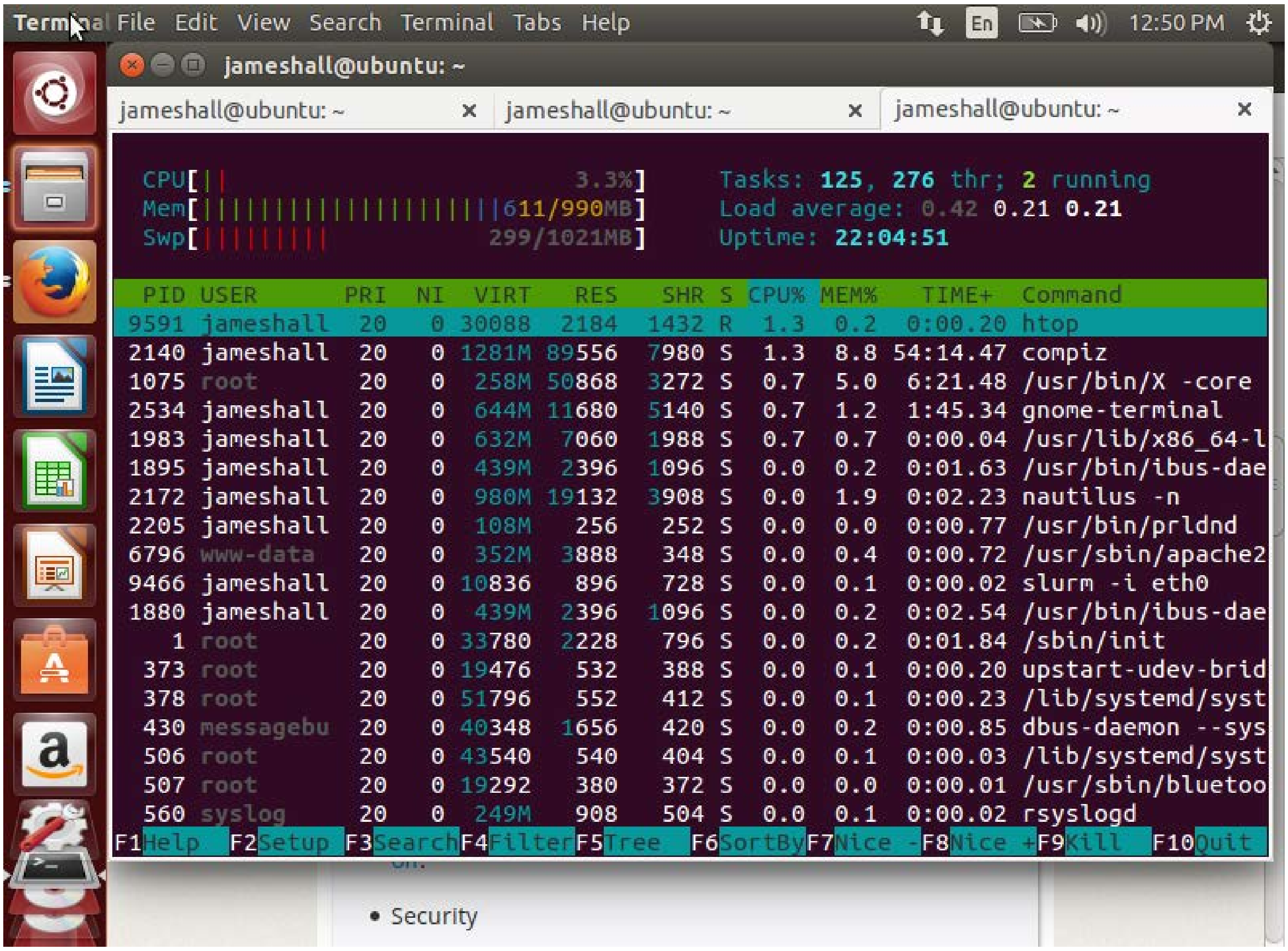
I've rounded up a few of my favorite monitoring tools outside the usual top command. Some of these tools even may be easily integrated into vtop as sensors.

**htop:** This is a feature-rich interactive process viewer and has been around for years. The author tweeted me to ask if he could use the Braille graphing idea. I'm very excited to see how this develops (**https://twitter.com/hisham_hm/status/477618055452037120**).

**iotop:** This is a great tool for measuring applications that are hammering your Input/Output. It calculates the number of bytes used. It's written in Python and parses information out of /proc/vmstat.

**netstat:** This ships as part of Linux and Windows, and it allows you to see all open connections. It's often useful to pipe this command into `more`:

```
netstat | more
```



```
  CPU[||                            3.3%]     Tasks: 125, 276 thr; 2 running
  Mem[||||||||||||||||||||||||611/990MB]     Load average: 0.42 0.21 0.21
  Swp[||||||||||              299/1021MB]     Uptime: 22:04:51

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 9591 jameshall  20   0 30088  2184  1432 R  1.3  0.2  0:00.20 htop
 2140 jameshall  20   0 1281M 89556  7980 S  1.3  8.8 54:14.47 compiz
 1075 root       20   0  258M 50868  3272 S  0.7  5.0  6:21.48 /usr/bin/X -core
 2534 jameshall  20   0  644M 11680  5140 S  0.7  1.2  1:45.34 gnome-terminal
 1983 jameshall  20   0  632M  7060  1988 S  0.7  0.7  0:00.04 /usr/lib/x86_64-l
 1895 jameshall  20   0  439M  2396  1096 S  0.0  0.2  0:01.63 /usr/bin/ibus-dae
 2172 jameshall  20   0  980M 19132  3908 S  0.0  1.9  0:02.23 nautilus -n
 2205 jameshall  20   0  108M   256   252 S  0.0  0.0  0:00.77 /usr/bin/prldnd
 6796 www-data   20   0  352M  3888   348 S  0.0  0.4  0:00.72 /usr/sbin/apache2
 9466 jameshall  20   0 10836   896   728 S  0.0  0.1  0:00.02 slurm -i eth0
 1880 jameshall  20   0  439M  2396  1096 S  0.0  0.2  0:02.54 /usr/bin/ibus-dae
    1 root       20   0 33780  2228   796 S  0.0  0.2  0:01.84 /sbin/init
  373 root       20   0 19476   532   388 S  0.0  0.1  0:00.20 upstart-udev-brid
  378 root       20   0 51796   552   412 S  0.0  0.1  0:00.23 /lib/systemd/syst
  430 messagebu  20   0 40348  1656   420 S  0.0  0.2  0:00.85 dbus-daemon --sys
  506 root       20   0 43540   540   404 S  0.0  0.1  0:00.03 /lib/systemd/syst
  507 root       20   0 19292   380   372 S  0.0  0.0  0:00.01 /usr/sbin/bluetoo
  560 syslog     20   0  249M   908   504 S  0.0  0.1  0:00.02 rsyslogd
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Figure 4. The htop Interactive Process Viewer

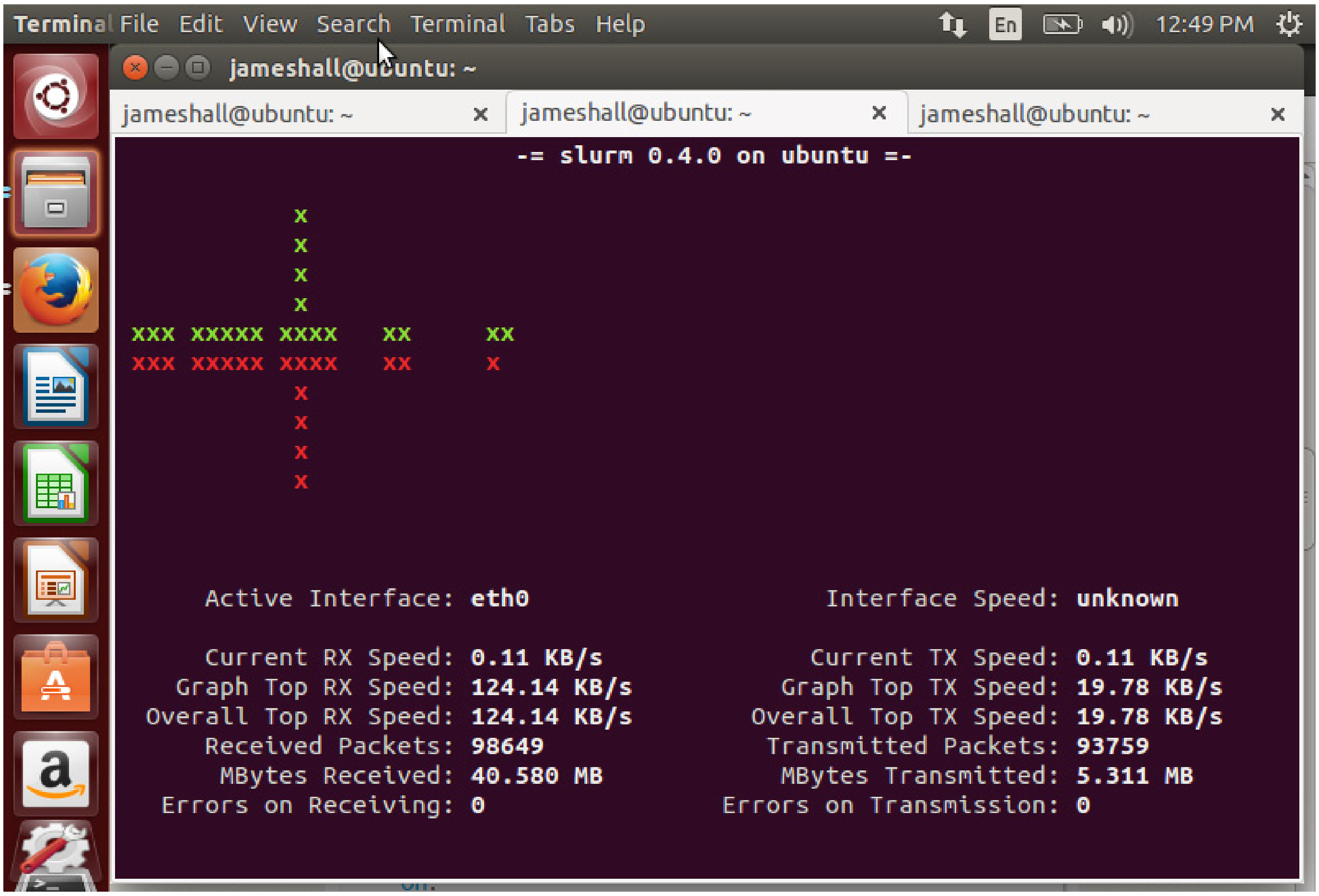
**apachetop:** This parses Apache (and Apache-compatible) log files on the fly to give you real-time requests, per-second stats, most popular pages and more. It's very handy for monitoring AJAX and other Web requests that aren't tracked in your favourite Web-based analytics.

**NetHogs:** This a great tool to see where all your Internet bandwidth is going. It lists each hog individually by KB/sec. It doesn't require you to load any special kernel modules—just fire it up and find the offending process straightaway.

**Slurm:** This tool helps you visualize network activity on your system by plotting red and green "x" characters.

## The Future

It's time to think more about how our computers can represent data over time, and how we can use tools that are more visual than top. What do you want from a system monitor? Do you need to see what's going on inside an app? Do you need to see



```
                          -= slurm 0.4.0 on ubuntu =-


           x
           x
           x
           x
xxx xxxxx xxxx   xx     xx
xxx xxxxx xxxx   xx     x
           x
           x
           x
           x




     Active Interface: eth0                    Interface Speed: unknown

     Current RX Speed: 0.11 KB/s              Current TX Speed: 0.11 KB/s
   Graph Top RX Speed: 124.14 KB/s          Graph Top TX Speed: 19.78 KB/s
 Overall Top RX Speed: 124.14 KB/s        Overall Top TX Speed: 19.78 KB/s
     Received Packets: 98649               Transmitted Packets: 93759
      MBytes Received: 40.580 MB            MBytes Transmitted: 5.311 MB
  Errors on Receiving: 0                Errors on Transmission: 0
```

Figure 5. Slurm

Figure 6. How can you help build vtop?

the number of Web server requests, the temperature of sensors or the throughput of a database server? What other visualizations could be done with Braille or other characters?

Roll up your sleeves, and let's make something cool!■

James Hall is the author of the popular jsPDF library and also founder of a digital agency in UK called Parallax (http://parall.ax).

## Resources

vtop: **http://parall.ax/vtop**

vtop GitHub Repository: **https://github.com/MrRio/vtop**

Blessed: **https://github.com/chjj/blessed**

Node-drawille: **https://github.com/madbence/node-drawille**

DOC SEARLS

# Big Bad Data

**Obsession with Big Data has gotten out of hand. Here's how.**

I'm writing this on September 11, 2014, 13 years after the famous day when terrorist hijackers flew planes into buildings, killing thousands and changing the world for the worse. I also spent the last three days getting hang time with Bill Binney (**http://en.wikipedia.org/wiki/William_Binney_%28U.S._intelligence_official%29**), who says the 9/11 attacks could have been prevented. Bill makes this claim because he led an NSA project designed to find clues and put them together. It was called ThinThread (**http://en.wikipedia.org/wiki/ThinThread**). The NSA discontinued ThinThread three weeks before the attacks, opting eventually to go with another project called Trailblazer (**http://en.wikipedia.org/wiki/Trailblazer_Project#Background**). Bill says ThinThread would have cost $9 million to deploy. Trailblazer ended up costing hundreds of millions of dollars and sucked (**https://en.wikipedia.org/wiki/Trailblazer_Project#Whistleblowing**).

Like its successors, such as PRISM (**http://en.wikipedia.org/wiki/PRISM_%28surveillance_program%29**), Trailblazer was all about collecting everything it could from everywhere it could. "At least 80% of all audio calls, not just metadata", Bill tells us (**http://www.theguardian.com/commentisfree/2014/jul/11/the-ultimate-goal-of-the-nsa-is-total-population-control**), "are recorded and stored in the US. The NSA lies about what it stores." At the very least, revelations by Bill and other sources (such as Edward Snowden and Chelsea Manning) make it clear that the Fourth Amendment (**https://en.wikipedia.org/wiki/Probable_cause**) no longer protects American citizens from unreasonable searches and seizures. In the era of Big Data everywhere, it's reasonable to grab all of it.

Surveillance also has a chilling effect on what we say. Talk about ________ and the Feds might flag you as a ________. Among other

**As a reader, you're probably already on some NSA list. I'd say "be careful", but it's too late.**

things, Edward Snowden and Glenn Greenwald (**https://en.wikipedia.org/wiki/Glenn_Greenwald**) revealed that *Linux Journal* has been placed (**http://www.linuxjournal.com/content/nsa-linux-journal-extremist-forum-and-its-readers-get-flagged-extra-surveillance**) under suspicion (**http://www.linuxjournal.com/content/stuff-matters**) by an NSA program called XKeyscore (**https://en.wikipedia.org/wiki/XKeyscore**). As a reader, you're probably already on some NSA list. I'd say "be careful", but it's too late.

The differences between ThinThread and what the NSA now does are ones of method and discretion. ThinThread's method was to watch for suspect communications in real time on international data pipes, and to augment or automate the work of human analysts whose job was finding bad actors doing bad things while also protecting people's rights to privacy. The scope of data collected by the NSA since then has veered toward the absolute. In sworn testimony (**https://publicintelligence.net/binney-nsa-declaration**), in support of the Electronic Frontier Foundation's suit (**https://www.eff.org**) against the NSA (Jewel v. NSA, **https://www.eff.org/cases/jewel**), Bill said this about the size of the agency's data processing and storage plans:

> The sheer size of that capacity indicates that the NSA is not filtering personal electronic communications such as email before storage but is, in fact, storing all that they are collecting. The capacity of NSA's planned infrastructure far exceeds the capacity necessary for the storage of discreet, targeted communications or even for the storage of the routing information from all electronic communications. The capacity of NSA's planned infrastructure is consistent, as a mathematical matter, with seizing both the routing information and the contents of all electronic communications.

So the NSA has been into Big Data since at least a decade before the term came into common use (Figure 1).

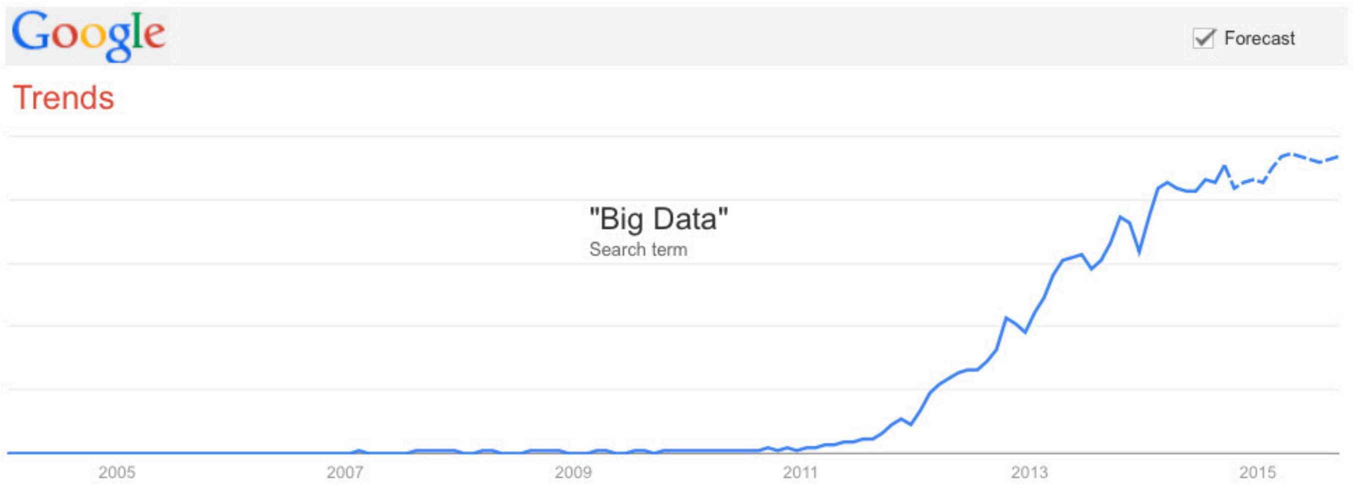


**Figure 1. Big Data Trends (Source: Google Trends, September 11, 2014)**

The year 2011 was, not coincidentally, when McKinsey (**http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation**) and Big Tech Vendors began driving the demand for Big Data solutions with aggressive marketing of the meme.

The pitch went like this: the world is turning into data, in quantities exploding at an exponential rate. It is essential to get in front of that wave and take advantage of it, or to risk drowning in it. With Big Data, you can "unlock value", "gain insights", "improve performance", "improve research", "segment marketing and services", "improve decision-making". And, of course, "save lives".

Lots of the pitching talked about science and health, where the advantages of more data always have been obvious. On the science side, that imperative surely helped sway the NSA toward Trailblazer and PRISM and away from ThinThread, which was about doing more with less. But now the Big Data meme is hitting a plateau, as you can see in the graph in Figure 1. There is also a backlash against it (**http://www.economist.com/blogs/economist-explains/2014/04/economist-explains-10**), given the degree to which we also are surveilled by marketers. In "How Big Data is Like Big Tobacco—Part 1" (**http://www.forbes.com/sites/sap/2014/08/26/how-big-data-is-like-big-tobacco-part-1**), Tim Walsh, SAP's Global Vice President, Customer Engagement and Commerce, writes this for *Forbes*:

Big Data is running down a similar

path. Deception? Check. Users are only now realizing on a broad basis that many companies are watching, recording and manipulating them constantly. It's not just what you buy. That's primitive stuff. Every site you visit, everything you "like", every person you interact with online, every word you type in "free" email or chat service, every picture you take (yes, including those you thought were instantly deleted), every physical place you go with that mobile device, the middle of the night drunken surfing—yes, yes and yes.

And it's not just online activity. Remember, companies have been at this for decades. All the publicly available information is now being tied together with your digital life to deliver an incredibly intimate picture of who you are and what you are likely to want, spend, do. Just leave it to Big Data to make the predictions. (What's the best way to make an accurate prediction? Manipulate the outcome!)

Anyone not living in a gun shack has a profile that runs to literally thousands of data elements. You don't need to be a Facebook addict to have a file 6 inches thick that carries your purchase history, voter registration, residence, major credit events, network of friends, etc. That list is growing exponentially because now the cottage data industry has become Big Data, with limitless resources. Increasingly, Big Data isn't even bothering to ask user consent for any of this. As they say: "Not paying for the product? You are the product." The government (US and EU) is taking notice and taking action. Users feel deceived and governments have picked up the scent.

In "Eight (No, Nine!) Problems With Big Data" in *The New York Times* (**http://www.nytimes.com/2014/04/07/opinion/eight-no-nine-problems-with-big-data.html?_r=1**), Gary Marcus and Ernest Davis lay out more issues:

1. "...although big data is very good at detecting correlations, especially subtle correlations that an analysis of smaller data sets might miss, it never tells us which correlations are meaningful."

2. "...big data can work well as an adjunct to scientific inquiry but rarely succeeds as a wholesale replacement."

**We are also barely revealed by the junk that marketing surveillance systems pick up when they follow us around with cookies, tracking beacons and other intrusive and unwelcome things.**

3. "...many tools that are based on big data can be easily gamed."

4. "...even when the results of a big data analysis aren't intentionally gamed, they often turn out to be less robust than they initially seem."

5. "...the echo-chamber effect, which also stems from the fact that much of big data comes from the web."

6. "...the risk of too many correlations."

7. "...big data is prone to giving scientific-sounding solutions to hopelessly imprecise questions."

8. "...big data is at its best when analyzing things that are extremely common, but often falls short when analyzing things that are less common."

9. "...the hype."

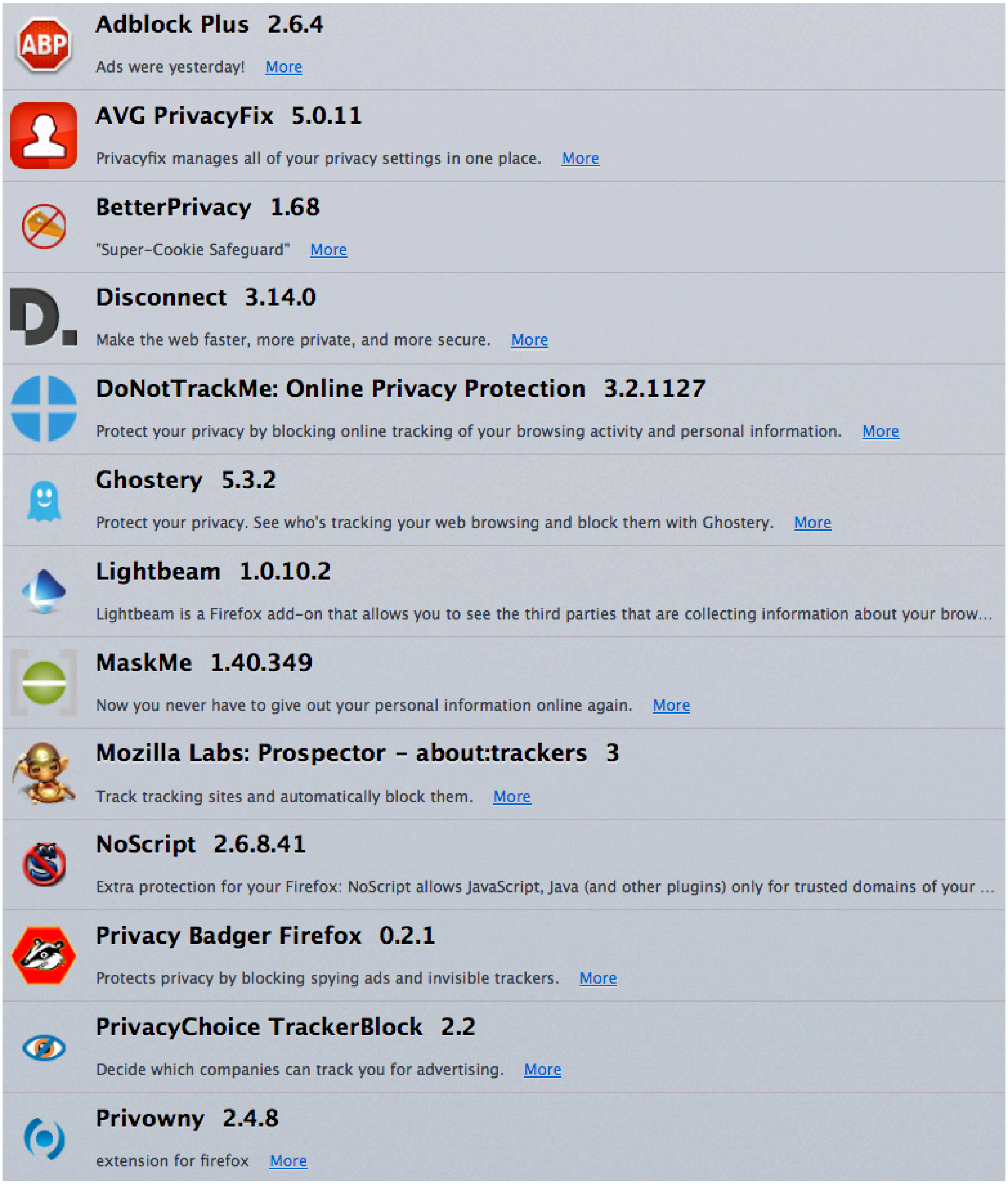
Another problem: it tends not to work. In "Where Big Data Fails... and Why" (**http://blog.primal.com/where-big-data-failsand-why**), Peter Sweeney explains how increasing the size of the data and complexity of the schema (expressiveness and diversity of knowledge) results in poor price/performance toward achieving marketing's holy grail of "personalized media". His bottom line: "These analytical approaches inevitably break down when confronted with the small data problems of our increasingly complex and fragmented domains of knowledge."

There is nothing more complex and fragmented than a human being—especially if you're a robot who wants to get personal with one. Each of us not only differs from everybody else, but from ourselves, from one moment to the next. So, while big data works well for making generalizations about populations of people, at the individual level it tends to fail. We are also barely revealed by the junk that marketing surveillance systems pick up when they follow us around with cookies, tracking beacons and other intrusive and unwelcome things. Here's how Peter Sweeney lays it out, verbatim:

- "The individual interests and preferences of end-users are only partially represented in the media."
- "Individual user profiles and activity do not provide sufficient data for modeling specific interests."
- "Market participants do not produce sufficient data about individual products and services."
- "Media and messaging are only a shadow of the interests of end-users; direct evidence of end-user interests is relatively sparse."

This is why the popularity of ad blockers (most of which also block tracking) are high, and growing rapidly. This is the clear message of "Adblocking Goes Mainstream" (**http://downloads.pagefair.com/reports/adblocking_goes_mainstream_2014_report.pdf**), published on September 9, 2014, by PageFair (**https://pagefair.com**) and Adobe. Here are some results, verbatim:

- "In Q2 2014 there were approximately 144 million monthly active adblock users globally (4.9% of all internet users); a number which has increased 69% over the previous 12 months."
- "Google Chrome is bringing ad blocking to the masses and seeing the largest increase of adblockers, up by 96% to approximately 86 million monthly active users between Q2 2013 and Q2 2014."
- "Share of ads blocked by 'end-user installed' browsers is 4.7x higher than by 'pre-installed' browsers."
- "Adblock adoption is happening all over the world—Poland, Sweden, Denmark, and Greece are leading the way with an average of 24% of their online populations using adblocking software in Q2 2014."
- "Countries like Japan, Spain, China and Italy are catching up; with their percentage of online populations that use adblock plug-ins growing as much as 134% over the last 12 months."



Adblock Plus 2.6.4
Ads were yesterday! More

AVG PrivacyFix 5.0.11
Privacyfix manages all of your privacy settings in one place. More

BetterPrivacy 1.68
"Super-Cookie Safeguard" More

Disconnect 3.14.0
Make the web faster, more private, and more secure. More

DoNotTrackMe: Online Privacy Protection 3.2.1127
Protect your privacy by blocking online tracking of your browsing activity and personal information. More

Ghostery 5.3.2
Protect your privacy. See who's tracking your web browsing and block them with Ghostery. More

Lightbeam 1.0.10.2
Lightbeam is a Firefox add-on that allows you to see the third parties that are collecting information about your brow...

MaskMe 1.40.349
Now you never have to give out your personal information online again. More

Mozilla Labs: Prospector - about:trackers 3
Track tracking sites and automatically block them. More

NoScript 2.6.8.41
Extra protection for your Firefox: NoScript allows JavaScript, Java (and other plugins) only for trusted domains of your ...

Privacy Badger Firefox 0.2.1
Protects privacy by blocking spying ads and invisible trackers. More

PrivacyChoice TrackerBlock 2.2
Decide which companies can track you for advertising. More

Privowny 2.4.8
extension for firefox More

Figure 2. Privacy Extensions

This is the market talking. So is what's shown in Figure 2.

Figure 2 shows all the extensions for ad and tracking blocking I've added in Firefox.

I may be an extreme case (my interest in this stuff is professional, so I check everything out), but few of us like being spied on, or what being spied on does to us—whether it's biting our tongues or leading us to reject the very thing that pays for the free goods we enjoy on the Web.

There are legal and policy solutions to the problem of government surveillance. On the legal front we have the EFF and others, filing suits against the government (**https://www.eff.org/nsa-spying**) and making clear arguments on the open Web. On the policy front we have our votes, plus the combined efforts of the EFF, StandAgasinstSpying (**https://standagainstspying.org**), DemandProgress (**http://demandprogress.org/campaigns**), Sunlight Foundation (**http://sunlightfoundation.com**) and others.

On the business side, we have the clear message that ad and tracking blocking sends, plus the high cost of Big Data-based surveillance—which at some point will start making an

## Advertiser Index

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

| ADVERTISER | URL | PAGE # |
|---|---|---|
| **Drupalize.me** | http://www.drupalize.me | 23 |
| **EmperorLinux** | http://www.emperorlinux.com | 41 |
| **New Relic** | http://www.newrelic.com | 3, 17 |
| **ServerBeach** | http://serverbeach.com | 59 |
| **Silicon Mechanics** | http://www.siliconmechanics.com | 7 |

**ATTENTION ADVERTISERS**

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

ROI argument against itself. My own favorite argument against surveillance-based advertising is the one for old-fashioned brand advertising. This is what Don Marti (our former Editor-in-Chief, **http://zgp.org/%7Edmarti**) has been doing lately. For example (**http://zgp.org/%7Edmarti/business/monkey-badger/#.VBoCi0u0bwI**):

> Your choice to protect your privacy by blocking those creepy targeted ads that everyone hates is not a selfish one. You're helping to re-shape the economy. You're helping to move ad spending away from ads that target you, and have more negative externalities, and towards ads that are tied to content, and have more positive externalities.

The most positive externality, for us here at *Linux Journal*—and for journalism in general—is journalism itself. Brand advertising isn't personal. It's data-driven only so far as it needs to refine its aim toward populations. For example, people who dig Linux. Brand advertising supports editorial content in a nice clean way: by endorsing it and associating with it.

By endorsing journalism for exactly what it does, brand advertising is a great supporter. (It supports a lot of crap too, but that's beside the point here.) On the other hand, surveillance-driven personalized advertising supports replacing journalism with click-bait.

Don has a simple solution:

> So let's re-introduce the Web to advertising, only this time, let's try it without the creepy stuff (**http://zgp.org/targeted-advertising-considered-harmful/#what-next-solutions**). Brand advertisers and web content people have a lot more in common than either one has with database marketing. There are a lot of great opportunities on the post-creepy web, but the first step is to get the right people talking.

So, if you advertise something Linux-y, call our sales department.■

**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.**

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