LINUX™ JOURNAL

Since 1994: The Original Magazine of the Linux Community

HOW TO HARDEN YOUR SSH CONNECTIONS

JANUARY 2014 | ISSUE 237 | www.linuxjournal.com

SECURITY

ENCRYPTED BACKUP SOLUTIONS
With TrueCrypt and SpiderOak

An Introduction to **QUANTUM CRYPTOGRAPHY**

TOR
Browse the Web Anonymously

TAKING ADVANTAGE OF ENCRYPTION

TIPS FOR USING THE PAX ARCHIVING UTILITY

+

SOLID-STATE DRIVES
Are They Worth It?

usenix
THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION

UPCOMING CONFERENCES

For a complete list of USENIX and USENIX co-sponsored events, see www.usenix.org/conferences

FAST '14: 12th USENIX Conference on File and Storage Technologies

February 17–20, 2014, Santa Clara, CA, USA
www.usenix.org/conference/fast14

2014 USENIX Research in Linux File and Storage Technologies Summit
In conjunction with FAST '14
February 20, 2014, Mountain View, CA, USA
Submissions due: January 17, 2014

NSDI '14: 11th USENIX Symposium on Networked Systems Design and Implementation

April 2–4, 2014, Seattle, WA, USA
www.usenix.org/conference/nsdi14

2014 USENIX Federated Conferences Week

June 17–20, 2014, Philadelphia, PA, USA

USENIX ATC '14: 2014 USENIX Annual Technical Conference
www.usenix.org/conference/atc14
Paper titles and abstracts due January 28, 2014

HotCloud '14: 6th USENIX Workshop on Hot Topics in Cloud Computing

WiAC '14: 2014 USENIX Women in Advanced Computing Summit

HotStorage '14: 6th USENIX Workshop on Hot Topics in Storage and File Systems

UCMS '14: 2014 USENIX Configuration Management Summit

ICAC '14: 11th International Conference on Autonomic Computing

USRE '14: 2014 USENIX Summit on Release Engineering

Do you know about the USENIX Open Access Policy?

USENIX is the first computing association to offer free and open access to all of our conferences proceedings and videos. We stand by our mission to foster excellence and innovation while supporting research with a practical bias. Your membership fees play a major role in making this endeavor successful.

Please help us support open access.
Renew your USENIX membership and ask your colleagues to join or renew today!

www.usenix.org/membership

23rd USENIX Security Symposium

August 20–22, 2014, San Diego, CA, USA
www.usenix.org/conference/usenixsecurity14
Submissions due: Thursday, February 27, 2014

Workshops Co-located with USENIX Security '14

EVT/WOTE '14: 2014 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections

USENIX Journal of Election Technology and Systems (JETS)
Published in conjunction with EVT/WOTE
www.usenix.org/jets
Submissions for Volume 2, Issue 2, due: December 5, 2013
Submissions for Volume 2, Issue 3, due: April 8, 2014

HotSec '14: 2014 USENIX Summit on Hot Topics in Security

FOCI '14: 4th USENIX Workshop on Free and Open Communications on the Internet

HealthTech '14: 2014 USENIX Workshop on Health Information Technologies
Safety, Security, Privacy, and Interoperability of Health Information Technologies

CSET '14: 7th Workshop on Cyber Security Experimentation and Test

WOOT '14: 8th USENIX Workshop on Offensive Technologies

OSDI '14: 11th USENIX Symposium on Operating Systems Design and Implementation

October 6–8, 2014, Broomfield, CO, USA
www.usenix.org/conference/osdi14
Abstract registration due April 24, 2014

Co-located with OSDI '14:

Diversity '14: 2014 Workshop on Diversity in Systems Research

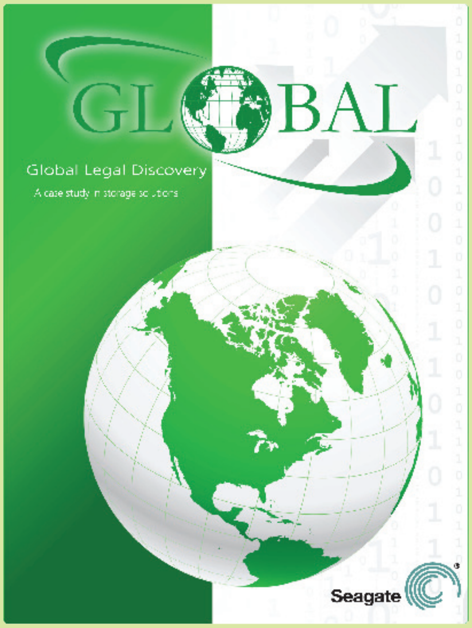
LISA '14: 28th Large Installation System Administration Conference

November 9–14, 2014, Seattle, WA, USA
https://www.usenix.org/conference/lisa14
Submissions due: April 14, 2014

Stay Connected...

twitter.com/usenix
www.usenix.org/youtube
www.usenix.org/gplus
www.usenix.org/facebook
www.usenix.org/linkedin
www.usenix.org/blog

SILICON MECHANICS
zStax
ZFS Unified Storage


GLOBAL
Global Legal Discovery
A case study in storage solutions
Seagate

CONTENTS

JANUARY 2014
ISSUE 237

SECURITY

FEATURES

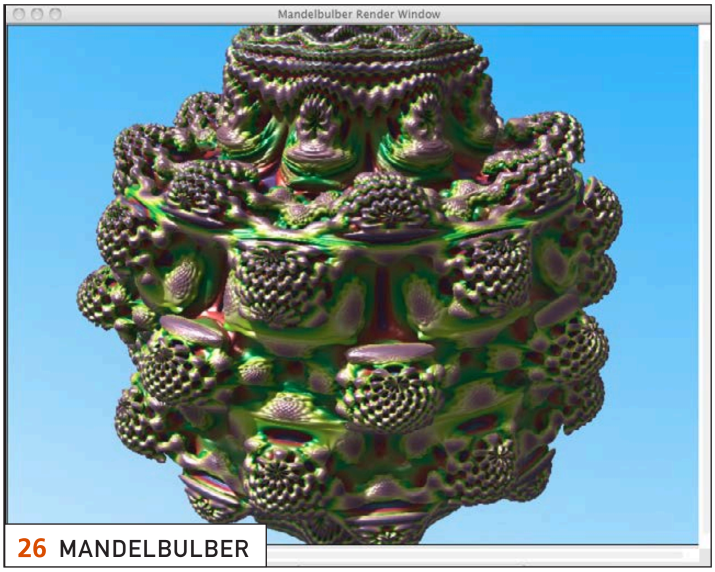
Cover Image © Can Stock Photo Inc. / maxkabakov

INDEPTH

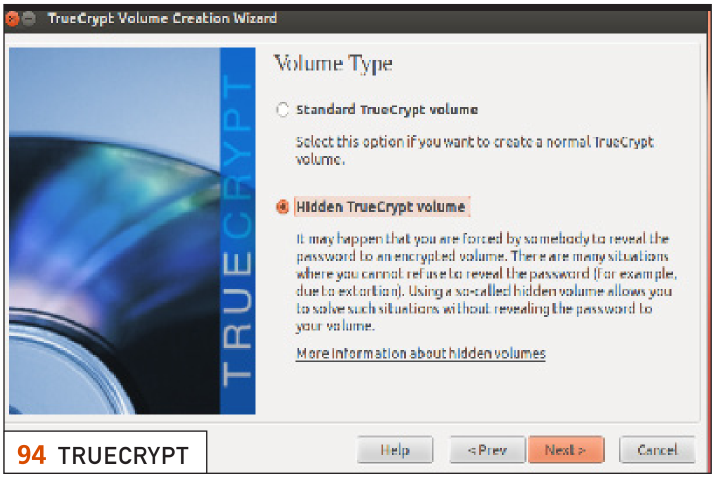
COLUMNS

KNOWLEDGE HUB

IN EVERY ISSUE



Status
Connecting to a relay directory



Volume Type

○ Standard TrueCrypt volume

Select this option if you want to create a normal TrueCrypt volume.

● Hidden TrueCrypt volume

It may happen that you are forced by somebody to reveal the password to an encrypted volume. There are many situations where you cannot refuse to reveal the password (for example, due to extortion). Using a so-called hidden volume allows you to solve such situations without revealing the password to your volume.

More information about hidden volumes

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

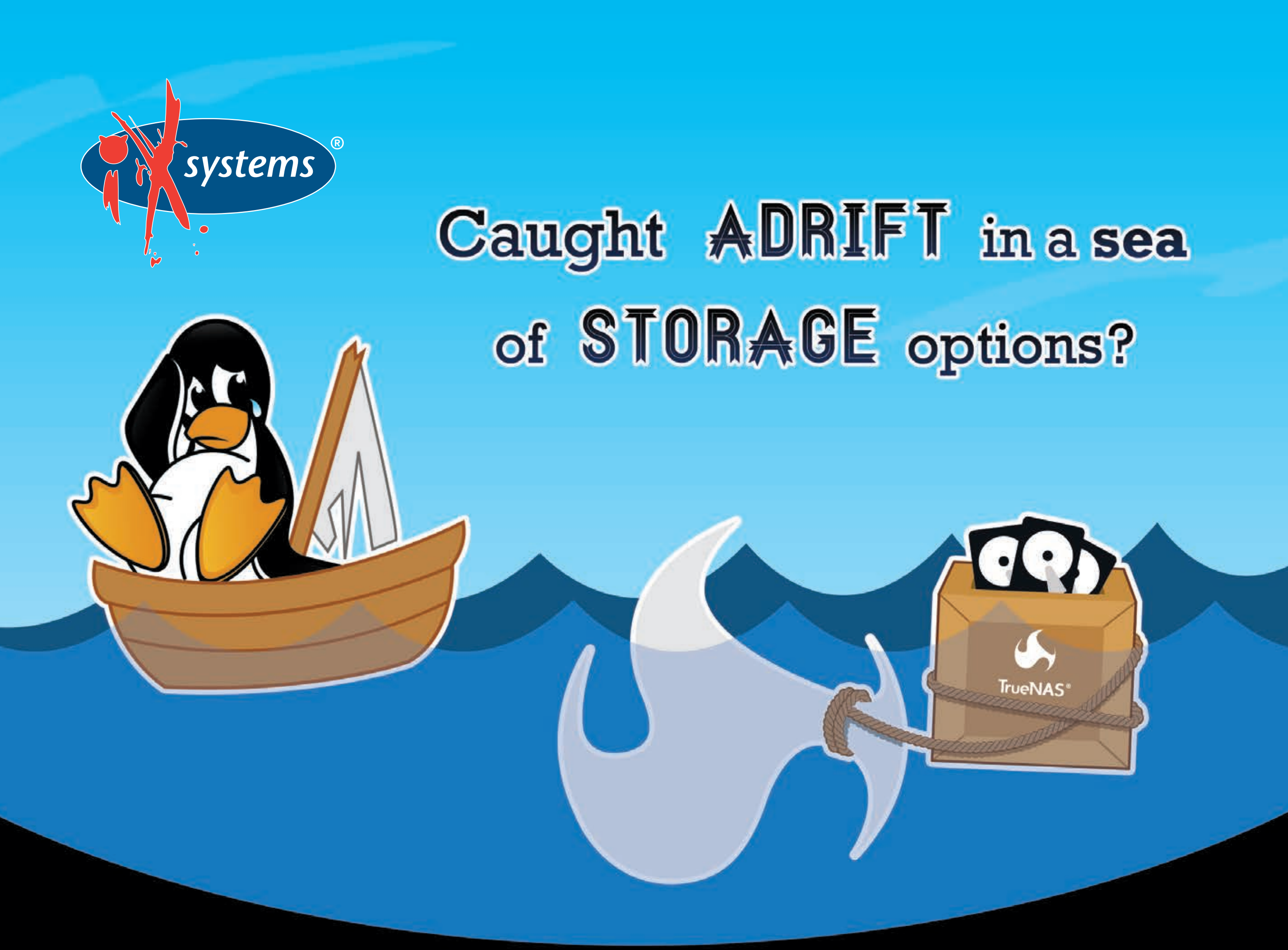
Contributing Editors
Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti • Ludovic Marcotte
Paul Barry • Paul McKenney • Dave Taylor • Dirk Elmendorf • Justin Ryan • Adam Monsen

Publisher	Carlie Fairchild publisher@linuxjournal.com
Director of Sales	John Grogan john@linuxjournal.com
Associate Publisher	Mark Irgang mark@linuxjournal.com
Webmistress	Katherine Druckman webmistress@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

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

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


iX systems®
Caught ADRIFT in a sea of STORAGE options?
TrueNAS®

iX systems
TrueNAS

SHAWN POWERS

Lapsang Souchong!

Back when we were kids, "security" meant little more than having a secret password to keep little siblings out of the treehouse. That's still the case in some situations. Take the title of this column, for instance. If you go to the #linuxjournal IRC channel on FreeNode, saying "Lapsang Souchong" will mark you as part of the inner circle. (Note, this does not make you one of the cool kids...possibly the exact opposite!)

When it comes to computer security, however, things are quite a bit more complex. Whether you want to encrypt your data or lock down network access, Linux provides a wide variety of security tools. This month, we focus on using those tools in our Security issue.

Reuven M. Lerner starts off the issue with instructions on how to integrate Twitter into your applications. Whether you need your app to tweet results, error messages or automatic cat photos, Reuven walks through implementing the API. Dave Taylor follows up with a tutorial on using the ImageMagick suite to watermark and copyright photos. Since I use ImageMagick extensively with my BirdCam project (which you'll hear more about in a month or so), I found his column particularly interesting. If you need to work with photos, especially if direct interaction isn't possible, Dave's column will be interesting for you too.

Kyle Rankin gets into the security mindset this month by approaching privacy. Specifically, he explains how to set up Tor in order to browse the Web in private. Tor is just as useful as it once was, but thankfully, it's gotten easier and easier to implement. I follow Kyle's column with The Open Source Classroom, and this month, I talk about file encryption. Many people are intimidated by the notion of encryption, but it doesn't have to be scary. This month, we'll do just enough encryption to wet your whistle, and hopefully get you interested in learning more.

Although I may have introduced encryption in my column, Subhendu Bera takes things to a whole new level with Quantum Cryptography. Mathematics-based encryption is complex, for sure, but will it be enough as technology advances? Subhendu gives an explanation of

Quantum Cryptography and a quick lesson in Quantum Mechanics as well. If you're interested in the future of cryptography, you'll love his article.

Remember Telnet? Telnet has been replaced in almost every situation by the much more secure SSH protocol. Granted, there still are a few situations that warrant the use of Telnet, but those generally are inside your network and never over the Internet. Just switching to SSH, however, isn't enough to ensure that you're secure. Sure, the connection itself is encrypted, but what if you have a user with a simplistic password? Or a script kiddie scanning for vulnerabilities? Federico Kereki describes how to harden SSH this month, making the wonderful and flexible SSH protocol a little safer to use. Whether you want to limit your allowed users or disable password connections altogether, Federico's article will guide you down the path of better SSH.

I may have started this issue with the basics of file and disk encryption, but if you are looking for more, Tim Cordova is about to be your favorite person. Going far beyond single file or even removable drive encryption, Tim shows how to encrypt your entire hard drive. Then, Tim goes even further and explains how to configure TrueCrypt in conjunction with SpiderOak to make sure your data is not only encrypted, but backed up as well! If you're interested in privacy and encryption, don't miss this article.

We finish off the security issue with Brian Trapp's article on solid-state drives. SSDs have been around for a number of years now, and we're finally to the point that we can provide some longevity statistics and reliability information. Have you been avoiding SSDs because you thought they would wear out? Did you think they had a significantly higher failure rate? Were you worried that you need Windows-specific drivers to make them work? Brian assuages many of those fears and validates those that are valid. SSDs are fast, and they can provide an incredible performance boost in most situations. You owe it to yourself to see if your scenario warrants an SSD. Brian's article will help.

This issue also contains tons of other Linux goodies. We have product announcements, opinion pieces and even fractals. You don't have to be one of the cool kids to enjoy this issue of *Linux Journal*, but it helps to be one of the smart kids. Thankfully, our readers tend to have that attribute in plentiful supply. 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
SPONSORED BY
New Relic.
NOVEMBER 2013 | ISSUE 235 | www.linuxjournal.com
SYSTEM ADMINISTRATION
SIDUS for Extreme Operating System Deduplication
GlusterFS Install, Benchmark and Optimize
Integrate Applications with the Python-Based ZATO PLATFORM
+ OPENSTACK SPECIAL REPORT
A LOOK AT SOLIDRUN'S CuBox Pro
PLAY CLASSIC CONSOLE GAMES ON A Raspberry Pi
Rails 4's NEW POSTGRESQL FEATURES

rss2email—Excellent Article

Thanks to Kyle Rankin for his “Command-Line Cloud rss2email” article in the October 2013 issue. I’ve been lamenting my “loss” of RSS feeds for some time, and this is a perfect solution!
—Steve Hier

I love that Linux affords us multiple solutions to our tech problems. I’ve tried a handful of Google Reader alternatives (settling on commafeed), but I love seeing how other people tackle the problem as well. Kyle’s penchant for simplicity certainly comes through with his preference for rss2email. I’m pretty sure Kyle would be happy with just a constant stream of 1s and 0s, but he’s not quite willing to admit it!—Shawn Powers

LVM, Demystified

Regarding Shawn Powers’ article “LVM, Demystified” in the December 2013 issue: I’ve been a fan of LVM2 from the beginning. (LVM1 really wasn’t ready for Prime Time.)

You said in your article “LVM is an incredibly flexible, ridiculously useful and not terribly complicated to use system.” I agree totally. However, it is not without its idiosyncrasies.

If you do a followup article, you may mention a few things.

1) There was a bug where trying to pvmove an entire volume with multiple LVs on it sometimes hung up LVM (at least the progress of the move), necessitating a reboot. The recommendation if you had a level with this bug was to move each LV individually.

This had the side benefit of allowing you to “defragment” the segments of your LV (by moving the segments in order and filling each PV). This makes no difference to performance, but makes it easier to see "what you have where". Tedious, but it makes the neat freak in me happy.

The Red Hat Advisory was RHBA-2012:0161-1; Bugzilla BZ#706036.

2) The metadata present on each PV now eats up a PE (that is, in your case, "not usable 3.00 MiB", but it's usually 4MB), and it is a good practice to have metadata on every PV! That means that, for example, if you have 5 * 100GB PVs, you don't have 500GB to use, you have 499.9something GB—that is, 500GB minus 20MB (5 PEs, each 4MB in size). This is a problem mainly with SAN LUNs, as they are usually precisely some size.

This means that if you allocated `-L 500G`, it would fail, telling you that you were slightly short of what you needed. A subsequent `-l 15980` would give you almost 500GB and would work. (I think I have my math right here, but you get the picture.)

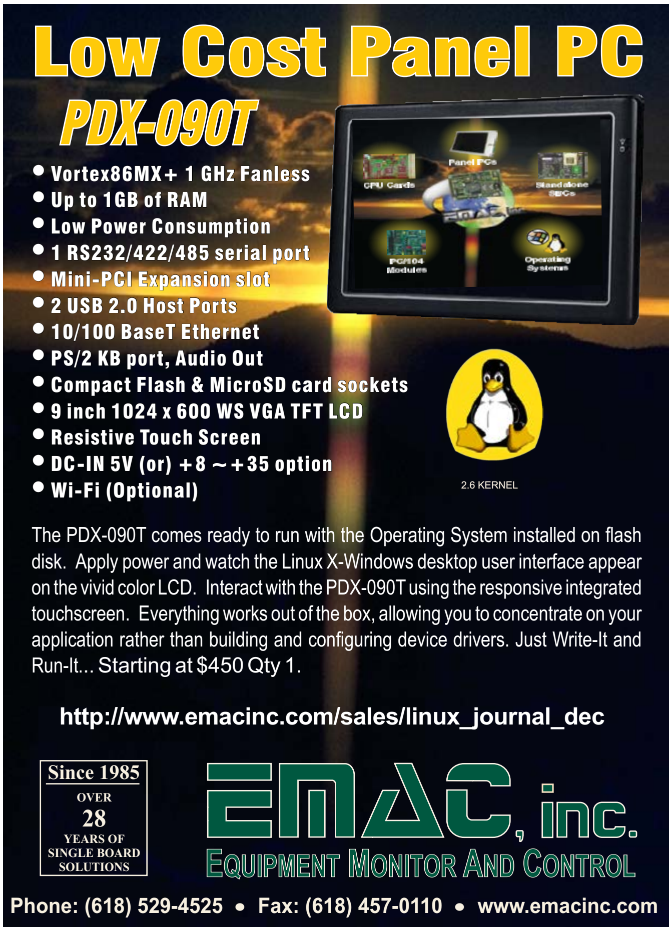
3) `lvdisplay --maps ...` and `pvdisplay --maps` are your best friends if you want to understand basic LVM.

4) Don't try to pvmove a swap volume. Simply allocate a new one and delete the old one.

Excellent article. It's not an easy concept to get across to the novice, but once you understand it, it seems so simple.

—Tom Lovell

It's always tough for me to decide



how far to travel down the rabbit hole when approaching a topic like LVM. By sysadmin standards, I'm a noob myself, since I avoided LVM for so long. I figured it was worthwhile to bring folks up to my comprehension level, even if I wasn't a zen master.

I said all that to say that I really, really appreciate letters like yours. Not only do I get to learn more, but it benefits everyone who reads Linux Journal *as well. And, now I get to go play with more LVM stuff!—Shawn Powers*

Bird Feeder

Shawn Powers' bird-feeder article (see "It's a Bird. It's Another Bird!" in the October 2013 issue) was one of the most appealing I've read in *LJ* since 1994. It's something I often contemplated, but never got beyond that. Many thanks for pointing the way.

An FYI, I alone have turned about six people into active viewers, so I do hope you have plenty of capacity, if only so I don't get locked out now. It's a very pleasant diversion. And you've put out a great bird buffet. Based on my own feeders, you will be kept quite busy keeping them full as word spreads in bird land. And of course, one really has to keep doing it throughout the winter now, as some birds become dependent on them.
—Bob Kline

It was my favorite article to write, up there with the article on the arcade cabinet I built and submitted back when I was a freelancer. I'm starting a followup article now, which will probably be published...hmm...in February? I've been tinkering with BirdCam, adding multiple cameras, motion detection with "motion", archive video creation—all sorts of cool stuff.

Thank you for the e-mail. I'm really glad you enjoyed the article and the camera. I have it scaled out to my Dreamhost account, so it should be able to handle lots of hits. I zoomed in the camera closer to the feeders (you probably noticed), and embedded the window cam and a closeup of the bird bath. It's so funny to see the starlings in the bird bath. I might point a camera there to capture video!—Shawn Powers

Linux Archive DVD

I would be very tempted by the Archive DVD, if there were PDF or Mobi versions of the back issues available on the Archive. I love the idea of using grep to search the HTML versions, but it would be nice to send an issue (once found) to your favorite reading device.

I know matching the original print format with a digital format is a painstaking process. Maybe you could make it clear it is an approximation or use a new "different" automated format for the back issues?

The digital versions of the back issues would be useful for *LJ* readers who have become accustomed to carrying our *LJ* issues on Kindles, tablets or phones.
—Rob

The Archive DVD used to confuse and frustrate me as well. I thought it was a simple collection of past issues that I'd be able to flip through like a pile of magazines. It's grown on me over the years, however, because I see it as more of a collection of articles unbound from the magazine format. Organization is still by issue, yes, but clicking through is a different experience.

Subscribers have access to back issues in whatever digital format is available (all formats for issues going back to September 2011, and PDFs of all formats from April 2005). We don't, unfortunately, have digital versions going all the way back, but those that exist should be accessible on your subscriber page. Hopefully that helps!—Shawn Powers

iPad App Issues

I've been using my iPad for viewing the digital subscription since the printed version ceased to exist. I think there needs to be a major update to your newsstand app. I've downloaded every issue to my iPad, but I cannot view *any* of the downloaded issues without an active Internet connection. For some reason, this evening I'm not able to connect to whatever service controls your downloads. Not only can I not download the latest issue, but I cannot view/read any of my existing already-downloaded issues! Reading my previously downloaded issues should not rely on nor require an active connection to anything. When I'm not having a problem connecting to your servers, all my downloaded issues say "Read" next to them; when I am having an issue, they all switch back to "Download". Please address this issue as soon as possible. Having to give up my print issues was hard enough, but this just compounds the problem.

Thanks for a great magazine!
—Jon Simonds

I don't have an iPad personally, but I've noticed with my wife's that the iOS7 implementation of Newsstand, at least as it pertains to the Linux Journal *app, is frustrating at best. To be honest, I download either the .epub or .pdf directly and peruse the issue from there. We'll work with our vendor to try to get things working right with Newsstand, but I expect the process to be lengthy and frustrating! The downloadable copies you get links for as a subscriber should load right into the iBooks app if you're having issues with the Newsstand app. Hopefully, things will be straightened out soon. I have found in the past that deleting and then re-installing the* Linux Journal *app sometimes helps as well.—Shawn Powers*

WRITE *LJ* A LETTER

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

PHOTO OF THE MONTH

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

LINUX JOURNAL

At Your Service

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

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

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

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

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

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

diff -u
WHAT'S NEW IN KERNEL DEVELOPMENT

A recent bug hunt by kernel developers ended up identifying a long-standing bug in **GCC**. The indications were there from the start, but it took some investigation to nail it down.

Originally, **Fengguang Wu** reported a kernel oops, and used "git bisect" to identify the specific patch that revealed the problem. It was an optimization suggested by **Linus Torvalds** and implemented by **Peter Zijlstra** that aimed at freeing up a hardware register by using the "asm goto" instruction in the kernel's **modify_and_test()** functions.

The first indication that the problem might boil down to a compiler bug was that the patch just seemed correct to folks. Neither Peter nor Linus were able to see anything wrong with it, so they suggested trying to reproduce the oops on kernels compiled with different versions of GCC, and Linus suggested disabling "asm goto" directly to see if that had any effect.

At first, Fengguang found that earlier compilers made no difference. He'd started off using GCC 4.8.1, but 4.6.1 also produced a kernel that would reproduce the oops. But as Linus suspected, disabling "asm goto" in the kernel code did fix the problem. After a while, Fengguang also discovered that the older GCC version 4.4.7 also produced a working kernel, because that compiler had no support for "asm goto".

Gradually, other folks began to be able to reproduce the problem on their own systems. Originally, the issue seemed to affect only 32-bit Linux systems, but ultimately, Linus was able to reproduce the problem on his own 64-bit system. It was harder to trigger on a 64-bit system, but it boiled down to being the same problem. As the scope of the problem began to reveal itself, Linus remarked, "It makes me nervous about all our *traditional* uses of asm goto too, never mind the new ones."

Jakub Jelinek opened a **Bugzilla** ticket against GCC, and folks started thinking about workarounds for the kernel. Even after GCC got a fix for this particular bug, it wouldn't do to allow the kernel to miscompile on any version of GCC, if it possibly could be avoided.

A workaround did end up going into the next Linux kernel release candidate, and a fix went into GCC 2.8.2. Shortly afterward, **Greg Kroah-Hartman** also adopted the kernel workaround in the 3.11.x stable tree.

The reason the kernel needed a workaround in spite of the fact that a real fix went into GCC was because the kernel needs to support the widest possible dispersion of host systems. Anyone, anywhere, with any particular hardware setup, using any particular versions of the various development tools, should be able to build and run the kernel. In some cases that ideal can't be reached, but it remains an ideal nonetheless.

Traditionally, software could mount a filesystem only after registering it with the kernel, so the kernel would know its name and a bit about how to manage it. This has been true even for internal filesystems like **ia64**, **pfmfs**, **anon_inodes**, **bdev**, **pipefs** and **sockfs**. But, **Al Viro** recently said there was no longer any reason to require registration for these filesystems, and he submitted a patch to take out the requirement.

First of all, he and Linus Torvalds agreed that there probably isn't any user code that actually looks up those filesystems in the registry. There's just no reason anyone would want to.

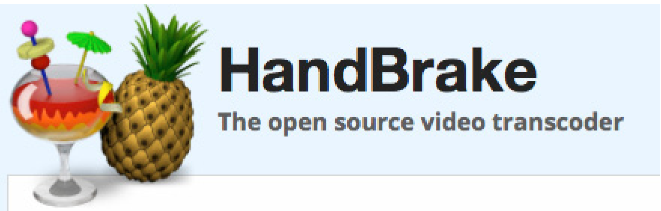
As Al explained on the mailing list, there used to be a need to register all filesystems. But about a decade ago, the **kern_mount()** call changed to take only a pointer to the filesystem, rather than needing to look it up by name.

Ever since then, the need to register these internal filesystems has been minimal. The only remaining dependency was a single data structure initialized by **register_filesystem()** that was needed by all filesystems. But, Al said that even this dependency was eliminated a couple years ago, when the data structure was optimized no longer to need register_filesystem(). By now, Al said, "there's no reason to register the filesystem types that can only be used for internal mounts."

With this change, **/proc/filesystems** would no longer list internal filesystems. And as Linus pointed out, those filesystems wouldn't reliably be listed anywhere on the system. Even **/proc/modules**, Linus said, would list those filesystems only if they'd been compiled as modules.

So, with some mild trepidation, Linus accepted the patch. If no one howls, it'll probably stay.—**ZACK BROWN**

Blu-ray Encryption—Why Most People Pirate Movies



HandBrake

The open source video transcoder

I get a fair amount of e-mail from readers asking how a person could do "questionable" things due to limitations imposed by DRM. Whether it's how to strip DRM from ebooks, how to connect to Usenet or how to decrypt video, I do my best to point folks in the right direction with lots of warnings and disclaimers. The most frustrating DRM by far has been with Blu-ray discs.

Unless I've missed an announcement, there still isn't a "proper" way for Linux users to watch Blu-ray movies on their computers. It's hard enough with Windows or Macintosh, but when it comes to Linux, it seems that turning to the dark side is the only option. In the spirit of freedom, let me point you in the direction of "how", and leave it up to you to decide whether it's a road you want to travel.

When ripping a movie from Blu-ray, I know of only one program that can do the job. MakeMKV is a cross-platform utility that will extract the full, uncompressed movie from most Blu-ray discs. Unfortunately, you have to download the source code and compile it. You need both the binaries and the source download files, and then follow the included directions for compiling the software. Yes, it's a bit complex.

Once you compile MakeMKV, you should be able to use it to extract the Blu-ray disc to your computer. Be warned, the file is *enormous*, and you'll most likely want to compress it a bit. The tool for that thankfully is much easier to install. Handbrake has been the de facto standard video encoding app for a long time, and when paired with MakeMKV, it makes creating playable video files close to painless. I won't go through the step-by-step process, but if the legally questionable act of ripping a Blu-ray disc is something you're comfortable doing, **http://www.makemkv.com** and **http://www.handbrake.fr** are the two software packages you'll want to explore.—**SHAWN POWERS**

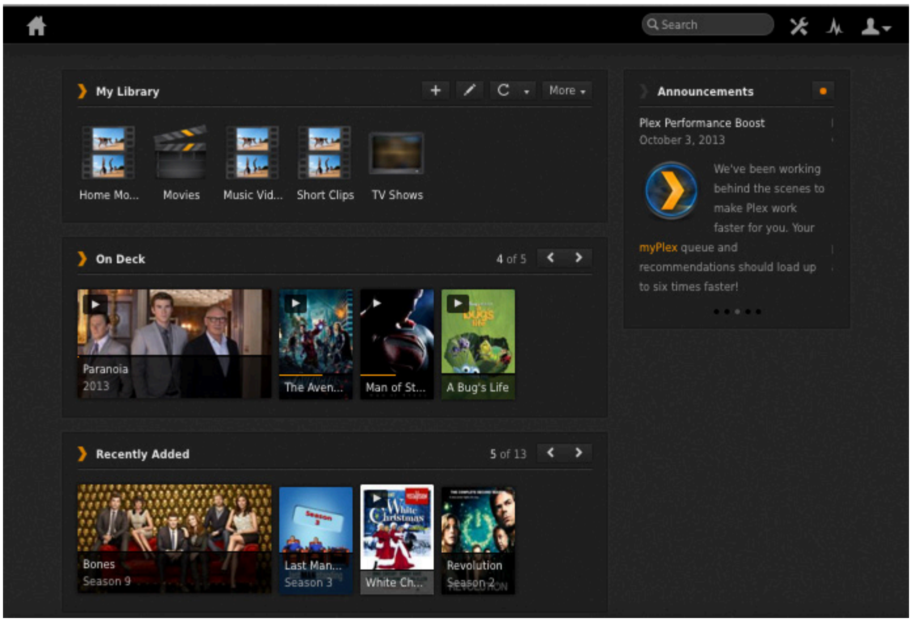
Non-Linux FOSS: Persistence of Vision Raytracer (POV-Ray)

This image is completely computer-generated, created by Gilles Tran, released into public domain.

Back in the mid-1990s, a college friend (hi Russ!) and I would put our old 8088 computers to work rendering ray-traced images for days—literally. The end result would be, by today's standards, incredibly low resolution and not terribly interesting. Still, the thought of a computer system creating realistic photos from nothing more than math equations was fascinating. As you probably already guessed, Russ and I weren't terribly popular.

All these years later, the same ray-tracing software we used back then is now up to version 3.7, and it has been released as free, open-source software. The developers kindly have created a downloadable Windows installer for those folks stuck on a Microsoft operating system. If you think the world is nothing more than math, and you'd like to prove it with ray-traced images, head on over to **http://www.povray.org** and download your copy today. I can't promise it will make you popular, but at least by my standards, it will make you cool!—**SHAWN POWERS**

Stream and Share Your Media with PlexWeb



Plex Performance Boost
October 3, 2013

We've been working behind the scenes to make Plex work faster for you. Your myPlex queue and recommendations should load up to six times faster!

Plex is one of those applications I tend to write about a lot. It's not because I get any sort of kickback or even a discount, but rather it's just an incredible system that keeps getting better. For this piece, I want to talk about PlexWeb, which functions much like the Android app I've mentioned before, but works completely inside a Web browser—almost any Web browser, on any operating system.

You can access PlexWeb by surfing to **http://my.plexapp.com** and logging in with your free account. (If you have a static IP at home, you also can connect directly to your home server by bookmarking the URL generated by plexapp.com.) You will be redirected to your home server, and you'll be able to transcode and stream your movies to any computer, anywhere.

I freely admit that I wish Plex was open source. Thankfully, however, its proprietary code does't mean Linux users are excluded. Whether you're using the Plex app on your Android device, installing Plex Home Theater on your Linux machine or even streaming video to your Aunt Edna's Web browser while visiting over the holidays, Plex is an incredible tool that keeps getting better. PlexWeb is free, but if you're interested in experiencing the latest and greatest Plex has to offer, a PlexPass subscription will get you access to features like Cloud Sync before anyone else gets to see them! To get started with Plex, visit the Web site at **http://www.plexapp.com**.

—SHAWN POWERS

Make Peace with pax

pax is one of the lesser known utilities in a typical Linux installation. That’s too bad, because pax has a very good feature set, and its command-line options are easy to understand and remember. pax is an archiver, like tar(1), but it's also a better version of cp(1) in some ways, not least because you can use pax with SSH to copy sets of files over a network. Once you learn pax, you may wonder how you lived without it all these years.

pax has four modes: list, read, write and copy. Reading and writing are controlled by the `-r` and `-w` options, repectively. In combination, `-rw`, pax acts a little bit like `cp -R`. If neither is used, pax lists the contents of the archive, which may be a file, device or a pipe.

By default, pax operates as a filter: it reads from standard input and writes to standard output, a feature that turns out to be very useful. But usually these days, the target is an archive file, the familiar tarball. Let’s start by creating one:

```
$ cd /tmp
$ mkdir paxample
$ touch paxample/foo
$ pax -wf paxample.tar paxample
```

The `-w` option means “write”—that is, create an archive. The `-f` option provides the name of a file to which to write the archive. If desired, pax can gzip or bzip the file at the same time:

```
$ pax -wzf paxample.tar.gz paxample
```

Like most tar implementations, pax, by default, uses the Posix ustar file format. Because pax was born of a desire to unify archive file formats, many other formats also are supported, but in practice, they’re seldom used. Likely as not, any .tar.gz file you download from the Internet actually will be a ustar archive:

```
$ pax -wzf paxample.tar.gz paxample
$ file paxample.tar*
paxample.tar:    POSIX tar archive
paxample.tar.gz: gzip compressed data
```

The first thing you nearly always want to know about any archive is what’s in it. Listing the contents is the default action in the absence of either a `-r` or `-w` option:

```
$ pax -f paxample.tar
paxample
paxample/foo
```

Note that the archive retains the directory name you specified on the command line. That comes into play later when you read it.

To read an archive, use -r:

```
$ mkdir t
$ cd t
$ pax -rf ../paxample.tar
```

What did that do? Let's look at the source and target directories:

```
$ cd /tmp
$ find paxample t # traverse both trees
paxample
paxample/foo
t
t/paxample
t/paxample/foo
```

When pax read the paxample.tar archive, it created files in the current directory, t. Because the archive included a directory name, paxample, that directory was re-created in the output.

Copying Sets of Files To my mind, pax's -r and -w options make more sense than their -x and -c equivalents in tar—reason enough to switch. But, pax can do more than tar: it can copy files too:

```
$ rm -rf t
$ pax -rw paxample t
$ find t
t
t/paxample
t/paxample/foo
```

Unlike cp(1), pax is an archive utility. Its job isn't to make copies, but to archive files. When pax creates a file, it preserves the file's metadata from its input. The form of the input doesn't matter. In this case, the input isn't from an archive, it's the file itself:

```
$ ls -l paxample/foo t/paxample/foo
-rw-r--r--  1 jklowden  wheel  0 Sep 22 15:45 paxample/foo
-rw-r--r--  1 jklowden  wheel  0 Sep 22 15:45 t/paxample/foo
```

Yes—two identical files with two identical timestamps. The permission bits and ownership can be controlled too, if desired. Take that, cp(1)!

Perhaps you don't want to re-create the directory, or perhaps you want to change it in some way. One option is not to mention the input directory on the command line, but instead provide filenames:

```
$ rm -rf t/paxample/
$ (cd paxample/ && pax -rw * ../t/)
$ find t
t
t/foo
```

That's usually easiest. But if you need something more sophisticated, the -s option rewrites the path—actually, any part of the filename—using a regular expression:

```
$ rm -rf t/*
$ pax -rw -s ':paxample:my/new/path:g' paxample/ t
$ find t
t
t/my
t/my/new
t/my/new/path
t/my/new/path/foo
```

The -s option is handy, for instance, when unpacking a tarball that doesn't have version information in the directory name.

What Could Go Wrong? If you give the wrong filename to write, you just get an archive by the wrong name—no harm no foul. If you mistype an input archive filename though, you'll find yourself in 1985:

```
$ pax -rf paxample.whoopsie
pax: Failed open to read on paxample.whoopsie (No such file
or directory)

ATTENTION! pax archive volume change required.
Ready for archive volume: 1
Input archive name or "." to quit pax.
Archive name >
```

This is an idea that outlived its usefulness before it was implemented. You could type in the filename here, again, without readline support or tab completion. Well, at least it says what to do:

```
Archive name > .
Quitting pax!
```

How exciting!

As mentioned previously, pax uses standard input and standard output by default. That *is* a feature, but the first time you forget to provide a filename, you may think pax is very, very slow:

```
$ pax  -r paxample.tar
```

Oops! No `-f`. Also no message and no prompt. pax is ignoring the archive filename argument and reading standard input, which in this case, is the keyboard. You could type ^D, for end-of-file, but that forms invalid input to pax. Better to send up a smoke signal:

```
^C
pax: Signal caught, cleaning up.
```

It's even worse the first time you accidentally write to standard output while it's connected to your terminal. You heard it here first: don't do that.

Putting Standard Input to Work Standard input and standard output do have their uses, and here pax really comes into its own. For one thing, you can verify the effect of the `-s` option without creating an archive or the files:

```
$ pax -w -s ':paxample:my/new/path:g' paxample/ | pax
my/new/path
my/new/path/foo
```

Absent the `-f` option, `pax -w` writes to standard output. So rewrite the pathname with `-s`, and pipe the output to pax again, this time using its "list" mode, with neither the `-r` nor `-w` option. By default, pax reads from standard input and, in "list" mode, prints the filenames on the terminal.

That can save a lot of time, not to

mention a mess on the disk, when there are thousands of files.

Suppose you want to copy the paxample directory to another machine. One approach would be to create a tarball, copy to the target, log in to the target and unpack the tarball:

```
$ pax -wf paxample.tar paxample
$ scp paxample.tar oak:/tmp/
paxample.tar             100%   10KB  10.0KB/s   00:00
$ ssh oak
oak[~]$ cd /tmp
oak[tmp]$ pax -rf paxample.tar
oak[tmp]$ ls paxample/
foo
```

But there's a much easier way. Invoke pax on both machines, and connect the output of one to the input of the other:

```
$ pax -w paxample | ssh oak 'cd /tmp/ && pax -r && find paxample'
paxample
paxample/foo
```

`pax -w` writes to standard output. `ssh` reads standard input and attaches it to whatever utility is invoked, which of course in this case is pax again. `pax -r` reads from standard input and creates the files from that "archive".

pax is one of the lesser known utilities in a typical Linux installation. But it's both simple and versatile, well worth the time it takes to learn—recommended.

—JAMES K. LOWDEN

They Said It

Never let the future disturb you. You will meet it, if you have to, with the same weapons of reason which today arm you against the present.
—Marcus Aurelius Antoninus

Temptation rarely comes in working hours. It is in their leisure time that men are made or marred.
—W. N. Taylor

We turn not older with years, but newer every day.
—Emily Dickinson

The human tendency to regard little things as important has produced very many great things.
—Georg Christoph Lichtenberg

Getting fired is nature's way of telling you that you had the wrong job in the first place.
—Hal Lancaster

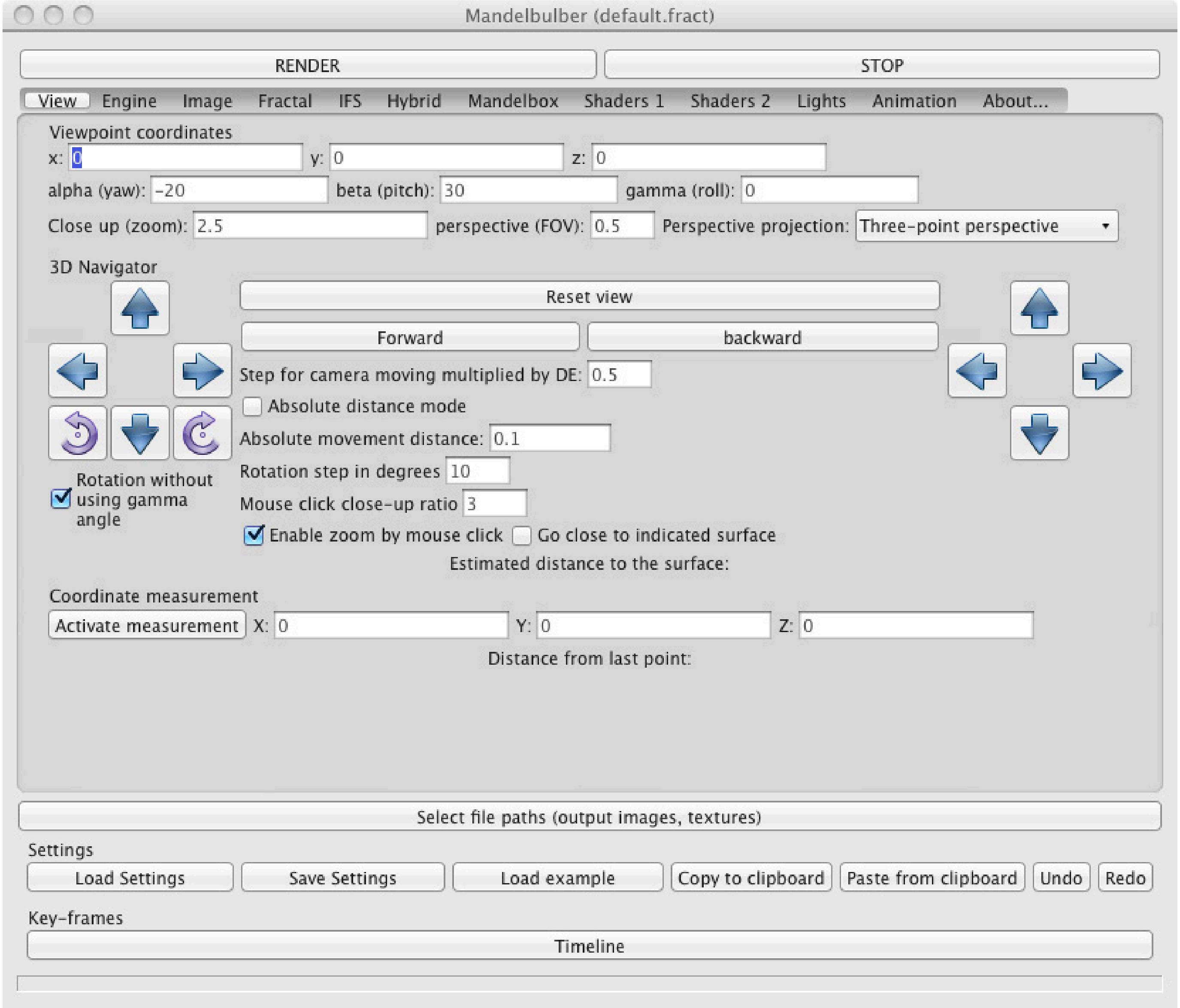
Taking Fractals off the Page

Fractals are one of the weirder things you may come across when studying computer science and programming algorithms. From Wikipedia: "A fractal is a mathematical set that has a fractal dimension that usually exceeds its topological dimension and may fall between integers." This is a really odd concept—that you could have something like an image that isn't made up of lines or of surfaces, but something in between. The term fractal was coined by Benoit Mandelbrot in 1975.

A key property of fractals is that they are self-similar. This means if you zoom in on a fractal, it will look similar to the way the fractal looked originally. The concept of recursion also is very important here. Many types of fractal algorithms use recursion to generate the values in the given set. Almost everyone has seen computer generated images of classic fractals, like the Mandelbrot set or the Cantor set. One thing about all of these classic images is that they are two-dimensional (or actually greater than one and less than two-dimensional, if you want to be pedantic). But there is nothing that forces this to be the case. Fractals can be any dimension, including greater than two. And with modern 3-D graphics cards, there is no reason why you shouldn't be able to examine these and play with them. Now you can, with the software package Mandelbulber (**http://www.mandelbulber.com**).

Mandelbulber is an experimental, open-source package that lets you render three-dimensional fractal images and interact with them. It is written using the GTK toolkit, so there are downloads available for Windows and Mac OS X as well as Linux. Actually, most Linux distributions should include it in their package management systems. If not, you always can download the source code and build it from scratch.

If you want some inspiration on what is possible with Mandelbulber, I strongly suggest you go check



Viewpoint coordinates

x: 0 y: 0 z: 0

alpha (yaw): -20 beta (pitch): 30 gamma (roll): 0

Close up (zoom): 2.5 perspective (FOV): 0.5 Perspective projection: Three-point perspective

3D Navigator

Step for camera moving multiplied by DE: 0.5

☐ Absolute distance mode

Absolute movement distance: 0.1

Rotation step in degrees 10

Mouse click close-up ratio 3

☑ Rotation without using gamma angle

☑ Enable zoom by mouse click ☐ Go close to indicated surface

Estimated distance to the surface:

Coordinate measurement

X: 0 Y: 0 Z: 0

Distance from last point:

Settings

Key-frames

Figure 1. The main window gives you all parameters that control the generation of your fractal.

out the gallery of images that have been generated with this software. There are some truly innovative and amazing images out there, and some of them include the parameters you need in order to regenerate the image on your own. The Mandelbulber Wiki provides a large amount of information (**http://wiki.mandelbulber.com/index.php?title=Main_Page**). When you are done reading this article, check out everything else that you can do with Mandelbulber.

When you first start up Mandelbulber, three windows

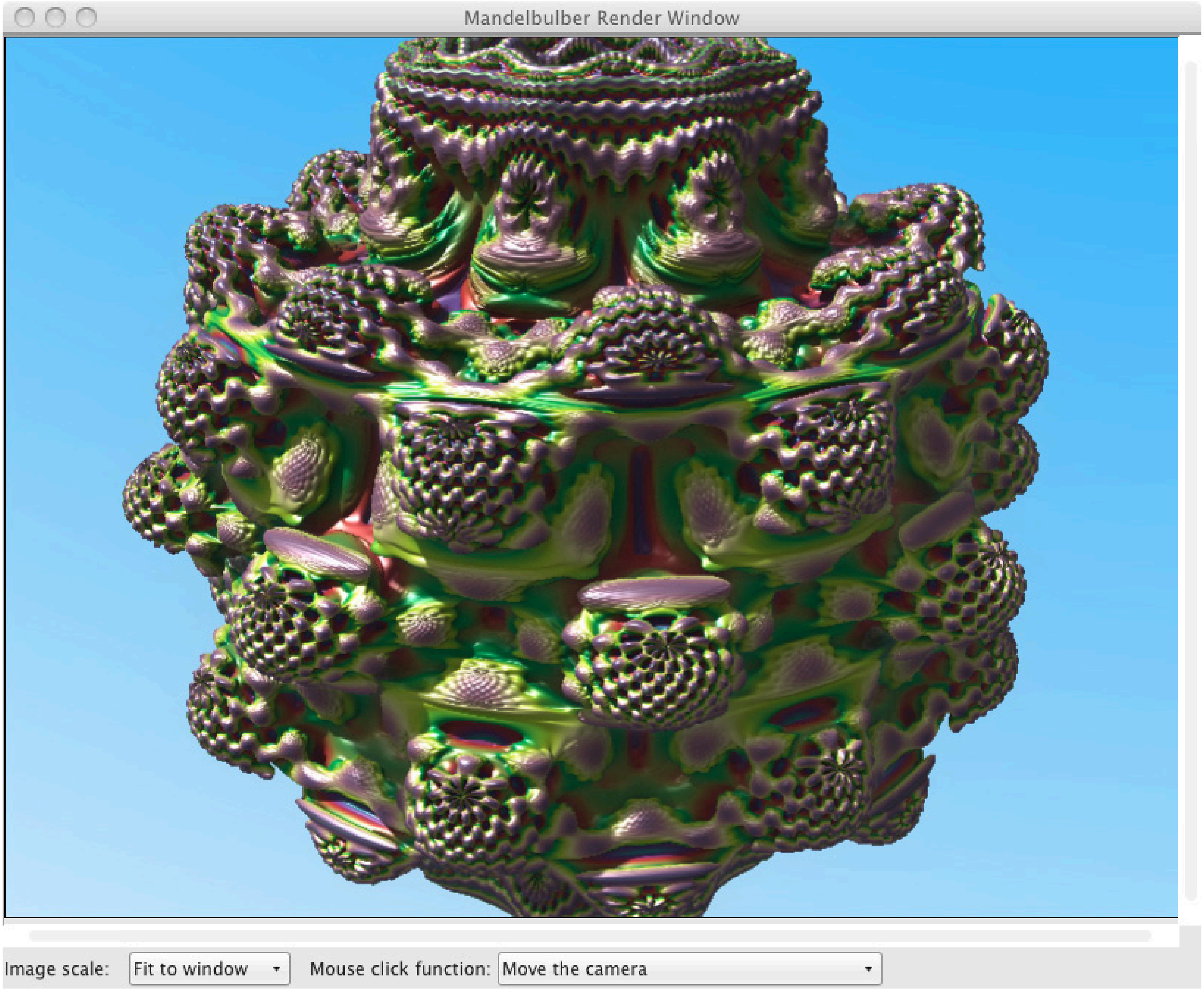


Figure 2. This is what the default 3-D fractal looks like.

open. The first is the parameters window (Figure 1). Along the very top are the two main buttons: render and stop. Below that is a list of 12 buttons that pull up different panes of parameters. You get an initial set of default parameters that will generate a 3-D version of the classic Mandelbrot set. Clicking on the render button will start the rendering process. If you have multiple cores on your machine, Mandelbulber will grab them to help speed up the calculations.

The rendered plot will be drawn in its own window (Figure 2). The third window shows you some measures of how the rendering progressed (Figure 3). You get two histograms

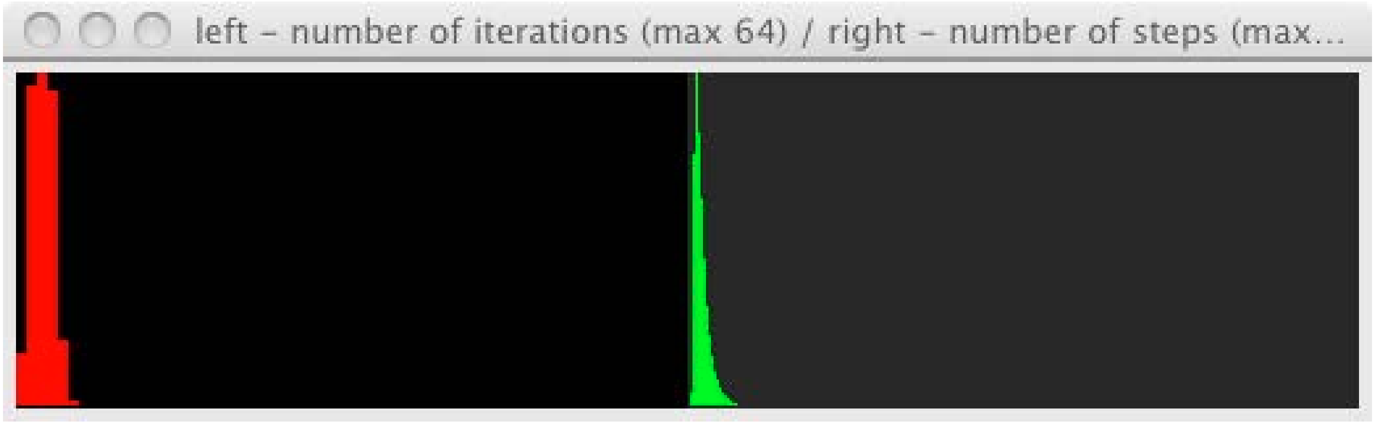

left – number of iterations (max 64) / right – number of steps (max...

Figure 3. Histograms of the Rendering Progression

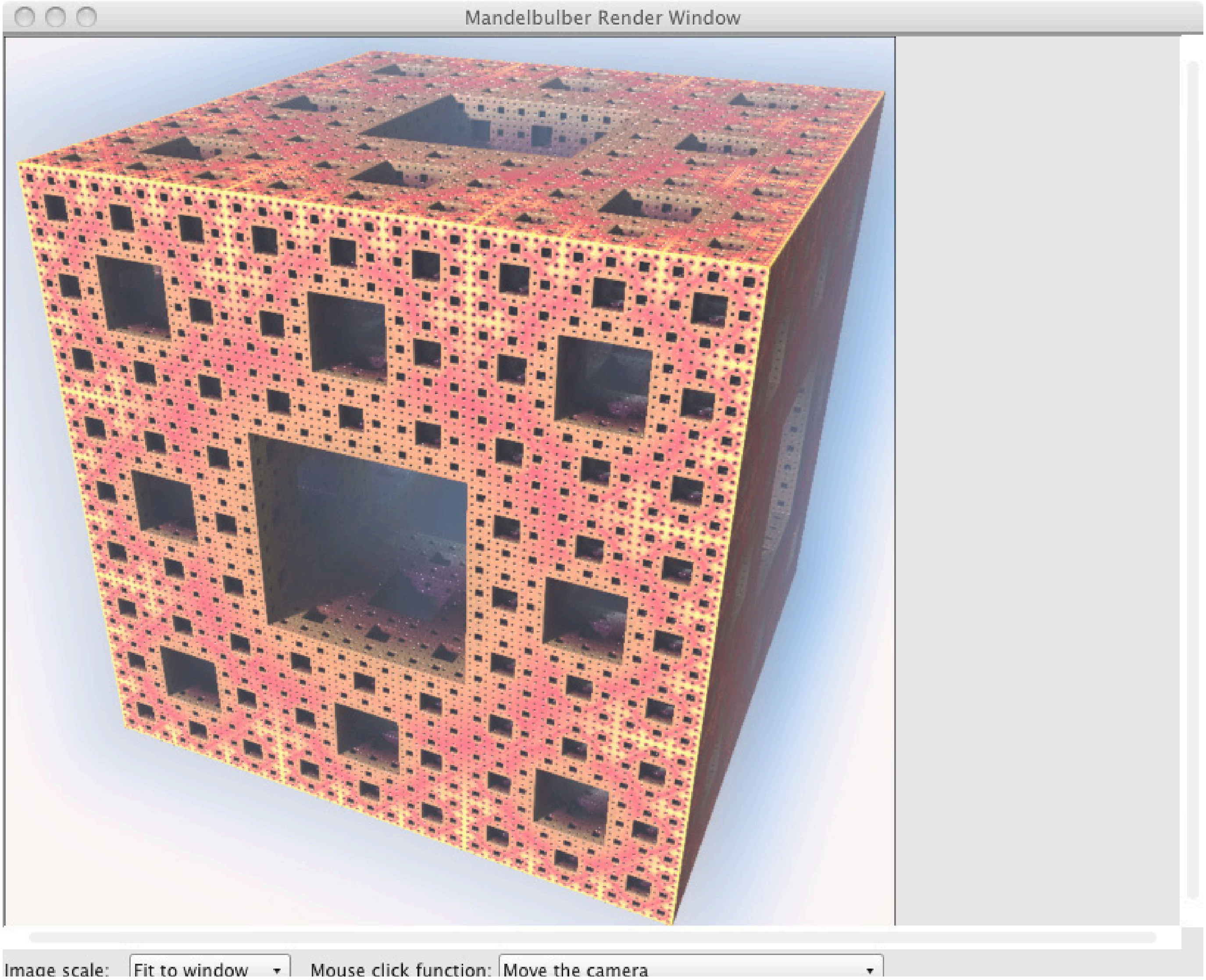
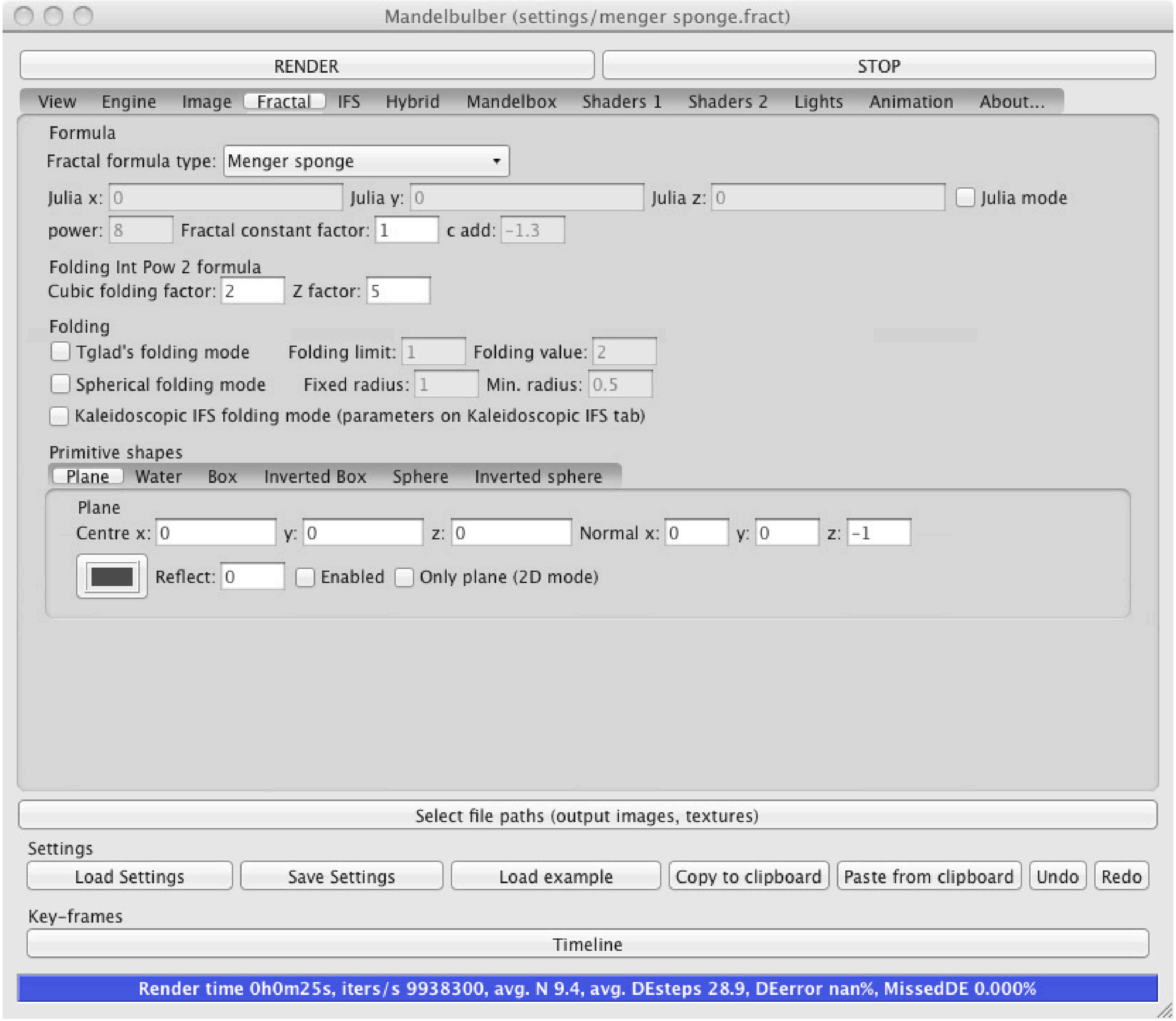

Mandelbulber Render Window
Image scale: Fit to window
Mouse click function: Move the camera

Figure 4. A Sierpinski sponge has infinite surface area and zero volume.

describing the number of iterations and the number of steps.

To generate new images, more than 70 examples are included with the installation of Mandelbulber that you can use as starting points. Clicking on the button Load example pulls up a file dialog where you can load one of them. For example, you could load "menger sponge.fract". Clicking the render button will generate a 3-D Sierpinski sponge (Figure 4). Although technically, the set is only one topological dimension that encloses zero



Formula
Fractal formula type: Menger sponge
Julia x: 0 Julia y: 0 Julia z: 0 □ Julia mode
power: 8 Fractal constant factor: 1 c add: -1.3

Folding Int Pow 2 formula
Cubic folding factor: 2 Z factor: 5

Folding
□ Tglad's folding mode Folding limit: 1 Folding value: 2
□ Spherical folding mode Fixed radius: 1 Min. radius: 0.5
□ Kaleidoscopic IFS folding mode (parameters on Kaleidoscopic IFS tab)

Primitive shapes
Plane
Centre x: 0 y: 0 z: 0 Normal x: 0 y: 0 z: -1
Reflect: 0 □ Enabled □ Only plane (2D mode)

Settings

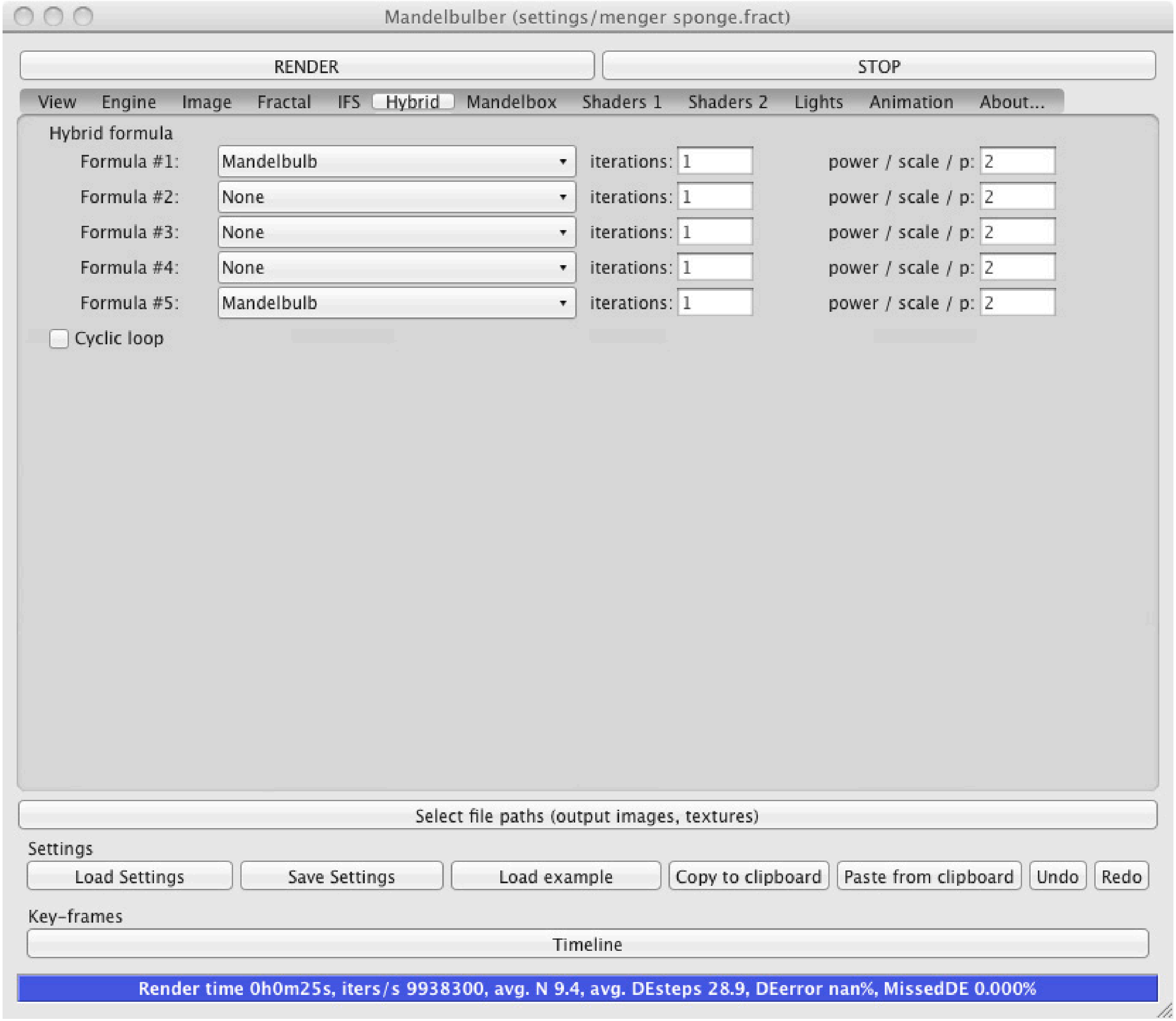
Key-frames

```
Render time 0h0m25s, iters/s 9938300, avg. N 9.4, avg. DEsteps 28.9, DEerror nan%, MissedDE 0.000%
```

Figure 5. There are several different fractal types from which to choose.

volume (aren't fractals weird?).

What can you change in Mandelbulber? Clicking on the fractal button pulls up the pane where you can set the parameters for the fractal itself (Figure 5). You can select from several different types of fractal formula types, such as mandelbulb, quaternion or menger sponge. You can set several options, depending on exactly which fractal type you choose. For example, if you select the iterated function system (IFS),



Hybrid formula

Formula #1: Mandelbulb iterations: 1 power / scale / p: 2

Formula #2: None iterations: 1 power / scale / p: 2

Formula #3: None iterations: 1 power / scale / p: 2

Formula #4: None iterations: 1 power / scale / p: 2

Formula #5: Mandelbulb iterations: 1 power / scale / p: 2

□ Cyclic loop

```
Render time 0h0m25s, iters/s 9938300, avg. N 9.4, avg. DEsteps 28.9, DEerror nan%, MissedDE 0.000%
```

Figure 6. You can create a hybrid system made from a mix of up to five different fractal types.

you then can click on the IFS tab to set several different parameters.

One of the issues is coming up with truly unique, yet aesthetically pleasing, sets of equations with which to experiment. To help in this regard, Mandelbulber has a hybrid option in the list of fractal types. When you select this option, you then can choose the hybrid button and set up to five different fractal equations (Figure 6). With this option, you can create very complex and sophisticated fractals to render.

Mandelbulber doesn't just generate static images of these higher dimensional fractals. There is an option to generate animations of how these images change when some parameter is swept over. To start, you need to click on the Timeline button at the bottom of the view pane. This pulls up a timeline window where you can set the parameters used to generate your animation. The record button puts parameters into the actual keyframe number (Key no. field on the right). It then loads and renders the next keyframe if it is not the last keyframe.

Then, you can add new keyframes with the "insert after" button or delete keyframes with the Delete button. To modify a given keyframe, you can double-click it to set the parameters, and then you can click on record to render the keyframe.

Interpolation between the keyframes is handled by Catmull-Rom splines. Once you have the keyframes handled, you will need to render the full animation. Clicking on the Animation button in the main window brings up the parameters you can set. These include things like the number of frames to render from the keyframes, as well as the start and end frame numbers. You then can click on the Render from key-frames button to generate the animation. On my netbook, this is a pretty long process. For image generation, you also have control over camera position, lighting and shader options. You should be able to generate the exact image or animation that you want.

If you are looking to generate some amazing 3-D landscapes or unique shapes for something science-fictiony, you definitely should check out Mandelbulber—just be prepared to lose several hours as you start playing with all of the parameters available.

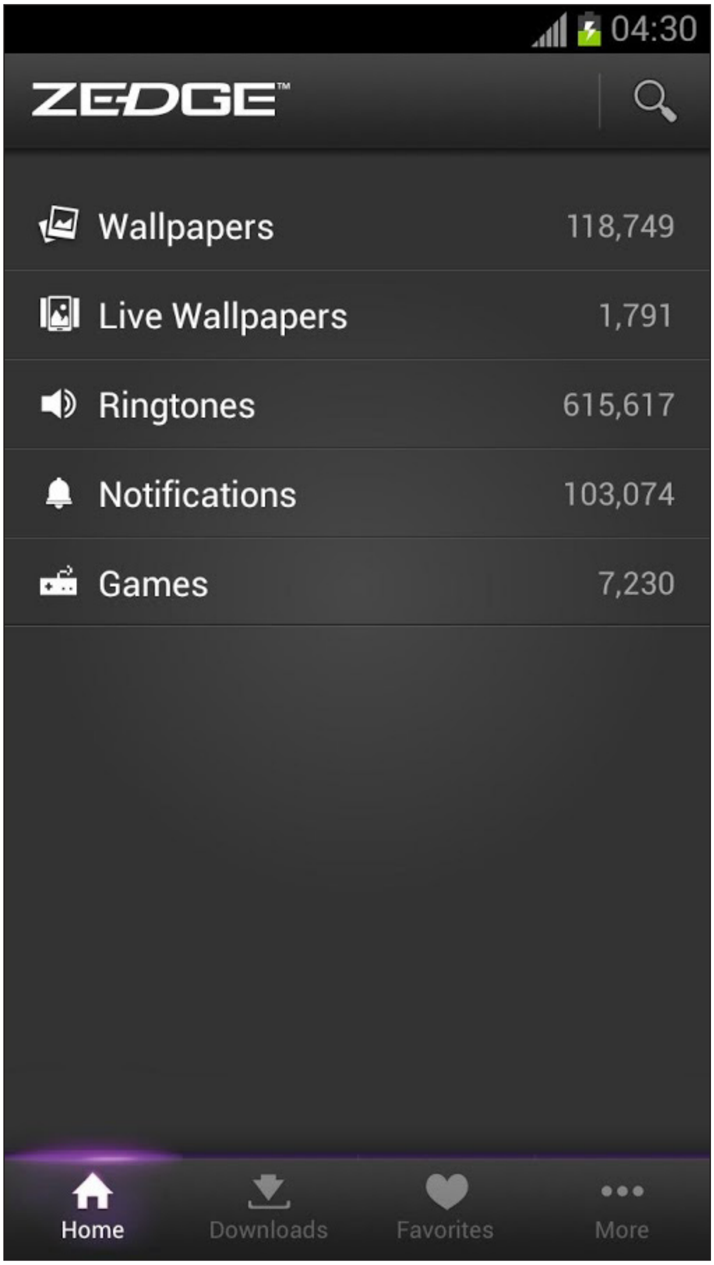
—JOEY BERNARD

12x
11
10
9
8
1

Zedge, for All Your Annoying Ringtones!

I really don't understand folks who use songs as their ringtones. Isn't it annoying or confusing when the song comes on the radio? If it's your favorite song, don't you get desensitized to it when you listen to the CD (or digital equivalent of CD)? Nevertheless, you probably hear dozens of ringtones every day. Those probably vary from "super annoying" to "what a cool ringtone". With Zedge, you can be the person annoying your fellow subway passengers—or making them jealous.

Zedge is a free app in the Google Play store, and the ringtones (and notification sounds and alarm sounds) are completely free as well. I currently use the "WHAAAT?!?!??!" sound from the minions on *Despicable Me* as a notification sound (which is *clearly* super cool and not annoying). My ringtone, which



Wallpapers	118,749
Live Wallpapers	1,791
Ringtones	615,617
Notifications	103,074
Games	7,230

Screenshot from the Google Play store

I hear much less often than in years past, is one I made myself from pasting together sound clips from *Star Trek the Next Generation*. Somehow, my homemade ringtone ended up on Zedge. I know it's mine, because I pasted together sounds that don't actually occur together on the show. I'm terribly proud of my ringtone, and if you'd like to hear it for yourself, search for "Incoming Subspace Signal", it should pop right up. If *Star Trek* isn't up your alley, there are thousands of other options from which to choose. With Zedge, installing them is simple and, of course, free.

Due to its incredible selection, seamless integration and amazing price tag, Zedge is this month's Editors' Choice winner. Check it out today at **https://play.google.com/store/apps/details?id=net.zedge.android**.

—SHAWN POWERS

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

REUVEN M. LERNER

Talking to Twitter

Integrating Twitter into your application is easy, fun and useful.

I'm a very quick adopter of many new software technologies. I try new programming languages, browsers, databases and frameworks without hesitation. But when it comes to social networks, I'm a bit of a Luddite, waiting to see what all the fuss is about before making them a part of my life. Sure, I signed up for Facebook almost as soon as it was available, but I haven't really posted much there. I do use LinkedIn, mostly to collect and find contacts, but I don't post there very often either, unless I'm announcing a presentation that I've added to SlideShare.

Twitter is something of a different story. There are people, it seems, for whom Twitter is the ultimate in communication. I've been on Twitter for some time, but other than an occasional foray into that world, I didn't really pay it much attention. Even now, after having decided several months ago that I should try to get into Twitter more heavily, I find that while I look through my feed several times a day, I tweet only once every few weeks. Call me a dinosaur, but I still prefer to use e-mail to be in touch with friends and family, rather than 140-character messages.

Although I don't see Twitter as a great medium for interpersonal communication, I recently have begun to appreciate it for other reasons. Specifically, I have discovered (perhaps long after the rest of the world has done so) that using Twitter as a sort of public logfile can make a Web application more visible, updating the rest of the world as to the status of your work and your on-line community. Doing so not only lets people hear about what you are doing—and potentially rebroadcast it to the world, by "retweeting" your message to followers—but it also increases your application's SEO, or visibility on various search engines. Finally, you can use Twitter to bring attention to your on-line presence by

The combination of tweeting updates and following other people has had a remarkable and direct effect on the number of visitors who come to my site, the length of time they remain and the number of pages they view.

following other people. (The idea is that when they receive your follow request, they may try to find out more about you, exploring your site or even following you back.)

I might sound like a social-media consultant, but I've seen the difference that Twitter can make in an application. I recently connected my PhD dissertation project (the Modeling Commons, at **http://modelingcommons.org**) to Twitter, such that each public action is sent to the Twitter feed. The combination of tweeting updates and following other people has had a remarkable and direct effect on the number of visitors who come to my site, the length of time they remain and the number of pages they view. Now, I'm not talking about millions of visitors per month. My application is still of interest mainly to a small community of people working with the NetLogo modeling environment. But the change has been obvious, and I grudgingly admit that I owe some of it to Twitter.

In this article, I explore some of the things I did to use Twitter in my application. From a technology perspective, you'll see that the implementation was fairly straightforward. But I think that what I've learned can be of interest to anyone running a Web application, particularly one that is trying to get the word out to the public. In addition, although there are plenty of good reasons to question Twitter's business practices and its relationship with developers, there is no doubt that its attention to detail with its API offers a model for all of us who want to provide APIs to our applications.

Registering with Twitter

I'm going to assume that anyone reading this article already has created a Twitter account or is able to figure out how to do so at Twitter.com. And of course, via the Twitter.com Web site, you can do all the things that you might expect, such as tweeting, retweeting, following and searching.

Twitter's API allows you to do all of these things via code. That is, you don't need to go and compose tweets personally. You can write a program that will do so for you. In order for this to happen, you need to do two things: register with Twitter's API service and install a library that knows how to communicate with the Twitter API.

In order to register with the Twitter API, you need to go to the "developer" site at **http://dev.twitter.com**. Note that you need to sign in with your Twitter user name and password, even if you already are signed in to the main Twitter site. The two sites do not seem to share login sessions.

Once you're on the developer site, you need to create a new application. The application name needs to be unique, but don't worry about it too much. You need to provide not only a name, but also a description and a URL that is associated with the application. Agree to the terms, fill in the Captcha, and you'll be on your way. Note that many types of Twitter applications exist, with many applications (including mobile) that post to Twitter on behalf of a user. The model I demonstrate in this article is of an application sending updates to Twitter, which means you won't have such issues—you don't need a callback URL or any special login configuration.

Perhaps the most confusing thing (to me, at least) about setting things up with Twitter was that the default permissions for an application allows you to retrieve tweets, but not post to them. To allow your application read-write access, go to the settings tab and indicate that you want the read-write access, or even read, write and direct message. You won't be using all of these capabilities for this example, but without write permission, your application will not be able to post to Twitter.

And now for the most important part, the keys: Twitter's authentication model requires two tokens. First, there is your access token, which allows you to access Twitter via the API. The second is the "consumer key", which describes your particular application and usage. Each of these keys has an accompanying secret, which you should treat as a password. As such, putting these secrets directly in your application probably is a bad idea. You would be better off putting them in environment variables, thus avoiding having the secrets in version control.

“Twitter” Gem for Ruby

Readers of this column know that I love the Ruby language, so it won’t come as a surprise to hear that I intend to use Ruby for my examples. However, there are Twitter API clients in virtually every modern language, making it easy to access from whatever you prefer to use in your programming.

The twitter Ruby gem, as is the case for all Ruby gems (libraries), is available for installation via the gem program, which comes with modern versions of Ruby. The gem currently is maintained by Erik Michaels-Ober, also known as “sferik” on GitHub. You can type:

```
gem install twitter -V
```

and the gem should be installed. On many systems, including those not running a Ruby version manager like rvm, you need to execute the above line while logged in as root.

Once you have installed the gem, you can use it. There are three parts to this process: bringing the gem into the program, configuring it to use your keys and secrets, and then executing a Twitter command. The first is handled with the Ruby `require` command, which looks at installed gems, as well as the Ruby core and standard libraries.

Configuration of the client is handled fairly straightforwardly from within a block that looks like this (filling in the values you got from Twitter’s API documentation):

```
twitter_client = Twitter::REST::Client.new do |config|
  config.consumer_key = CONSUMER_KEY
  config.consumer_secret = CONSUMER_SECRET
  config.oauth_token = OAUTH_TOKEN
  config.oauth_token_secret = OAUTH_SECRET
end
```

Notice that you are not merely executing the “new” method on `Twitter::REST::Client`, but that you also are returning a value. Thus, in contrast to previous versions of Ruby’s Twitter gem, you should accept the returned object, which is then the basis for all of the additional actions you wish to take.

Finally, you send the tweet with the “update” method:

```
tweet = twitter_client.update("Hello, world. Tweet tweet.")
```

Invoking the #update method has the effect of sending the message to Twitter. If you go to the Web page for your Twitter user, you’ll find that a new message has been sent, as if you had typed it.

If you capture the return value from the invocation of `twitter_client.update`, you’ll see that it is an instance of `Twitter::Tweet`, a Ruby object that represents a tweet. This object provides the functionality that you would want and expect from something associated from Twitter. For example:

```
tweet.user          # tells us who wrote the tweet
tweet.retweeted?    # indicates whether it was retweeted
tweet.favorited?    # indicates whether it was marked as a favorite
```

Now, it's also possible that you will not get a tweet object back at all, but rather that the "update" method will raise an exception. For example, Twitter forbids users from sending an identical tweet, at least within a short period of time. Thus, if you send the above "Hello, world" tweet (from the example above) a second time, you'll get an exception:

```
Twitter::Error::Forbidden: Status is a duplicate.
```

Of course, you can catch such errors with:

```
begin
  tweet = twitter_client.update("Hello again,
  ➥@reuvenmlerner  Tweet tweet.")
rescue Twitter::Error::Forbidden => e
  puts "You already tweeted that."
rescue => e
  puts e.class    # Twitter::Error::Forbidden
  puts e.message  # 'Status is a duplicate.'
end
```

If you include a Twitter @username, hashtag or URL in your tweet, the appropriate magic will happen automatically. Thus:

```
tweet = twitter_client.update("Go to @reuvenmlerner's
 ➥site at http://lerner.co.il/")
```

In the above tweet, the URL automatically will be shortened, using Twitter's standard t.co domain. Similarly, the @reuvenmlerner (my Twitter handle) will turn into a link. You can access both of these using methods on your tweet:

```
tweet.urls           # returns an array of Twitter::Entity::URI
tweet.user_mentions  # returns an array of
                     # Twitter::Entity::UserMention
```

You can more generally ask Twitter for information about tweets. For example, you can get the most recent tweets a user has sent with:

```
twitter_client.user_timeline("reuvenmlerner")
```

which returns an array of tweet objects. You can apply the "text" method to the first element, thus getting the text back from the user's most recent tweet:

```
twitter_client.user_timeline("reuvenmlerner")[0].text
```

But where would you use such API calls? Why would you want to use Twitter on your site?

If there are URLs embedded in the tweet, you can get those back:

```
twitter_client.user_timeline("reuvenmlerner")[1].urls
```

This method returns an array of `Twitter::Entity::URI` objects, each of which has attributes, such as "url" and "expanded URL".

Integrating into Your Application

As you can see, working with Twitter is surprisingly easy. The startup time for connecting to Twitter can take a little bit of time—up to two seconds, in my experience—but tweeting and querying Twitter take very little time. It's obvious, as a consumer of the API, that they have worked hard to make it execute as quickly as possible. This is a lesson to all of us who create APIs. We all know that Web pages should load quickly, and that slow load times can discourage people from staying on a site.

API calls typically are embedded within another application, meaning that if the API call takes time, the application itself will feel sluggish. As a result, a slow API call will lead to slow responses from the API clients—and may discourage people from using your API.

But where would you use such API calls? Why would you want to use Twitter on your site?

One simple use of the Twitter API would be to display a user's most recent tweets. For example, if your company (or you personally) use Twitter to send messages about what you are doing, you can see that it would be fairly easy to include those tweets in a Web page. Using an MVC system, such as Rails, you simply would grab the tweets (with the "user_timeline" method, as shown above), and stick the results on your home page. Now your home page provides another view to your Twitter feed, re-enforcing its importance and usage to your company.

I have been doing something slightly different. As I mentioned previously, I have begun to use Twitter to log public activity in the application I've developed for my dissertation. Every time a new

The biggest technical challenge I have faced so far in all of this is the issue of duplicate tweets.

user joins, new content is posted or someone adds a posting to a discussion forum, I send a new tweet on the subject. In and of itself, this doesn't do very much; Twitter is full of text and URLs. But I have certainly found by ensuring that my tweets are followed and seen by a large number of people, I have increased the number of users coming to my site.

In other words, by tweeting about activity on my site, I have given my site additional exposure to the world. Moreover, people who really want to see what my application is doing can follow the link in their Twitter feed and follow along.

By adding a #NetLogo hashtag to my tweets, I also have made it possible, and even easy, for my tweets (and thus my site) to be found and identified by people searching Twitter for mentions of our modeling environment. The fact that Google indexes tweets increases my site's visibility on-line among people who are searching for modeling-related sites.

The net effect has been rather huge. Within two weeks of starting to use Twitter to announce updates on my site, the number of people coming to visit has increased dramatically. Not coincidentally, my site's ranking in Google has improved noticeably.

Now, if this were a commercial site, rather than a free infrastructure for collaborative modeling, I would want to check a second thing, namely the "conversion rate"—that is, how many people who came to my site also became paying customers. But for my small, educational site, it has been fascinating to see what a difference tweeting made.

And what did I do? Truth be told, not much. I set up things such that a new tweet would be sent, using the "update" method demonstrated above, every time a new model version, forum posting or person was added to the system. Because of the relatively low latency on the "update" method, I even do this inline on an `after_create` callback within Rails, rather than queueing it in a background job.

The biggest technical challenge I have faced so far in all of this is the issue of duplicate tweets. When I first set up the Twitter feed, I defined the tweet for an additional discussion forum post to be:

```
Reuven Lerner has added a comment about the Foobar model!
```

The problem with this style of tweet is that it quickly can lead to duplicates—and thus errors from within the application. As a result, I have made sure that every tweet has a unique number in it somewhere, typically counting how many similar objects already have been created. For example:

```
Reuven Lerner wrote the 5th comment about the Foobar model!
```

The above ensures—assuming that user and model names are unique—that there cannot be duplicates, thus avoiding the problem.

Beyond the advantages for users, SEO and people interested in following my work, I also have found it to be enormously satisfying to see tweets come out even when I'm not aware of it. It's similar in some ways to seeing my children's creative output, but (obviously) less emotionally charged.

Conclusion

Adding automatic tweets to a Web application is easy to do and can have significant benefits. For your users, it gives them a way to follow what is happening in your application without needing to visit the site or use an RSS reader. For your site, automatic tweets will help bring in new visitors, improve SEO and generally improve your project's visibility.■

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

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

Resources

Twitter, of course, is at **http://twitter.com**. The developer and API documentation is at **http://dev.twitter.com**. The Ruby gem for Twitter, which apparently has been downloaded more than one million times (!), is at **http://sferik.github.io/twitter**.

DAVE TAYLOR

Easy Watermarking with ImageMagick

Script auteur Dave Taylor explores smart ways to use ImageMagick and Bash to copyright and watermark images in bulk.

Let's start with some homework. Go to Google (or Bing) and search for "privacy is dead, get over it". I first heard this from Bill Joy, cofounder of Sun Microsystems, but it's attributed to a number of tech folk, and there's an element of truth to it. Put something on-line and it's in the wild, however much you'd prefer to keep it under control.

Don't believe it? Ask musicians or book authors or film-makers. Now, whether the people who would download a 350-page PDF instead of paying $14 for a print book are hurting sales, that's another question entirely, but the Internet is public and open, even the parts that we wish were not.

This means if you're a photographer or upload images you'd like to protect or control, you have a difficult task ahead of you. Yes, you can add some code to your Web pages that makes it impossible to right-click to save the image, but it's impossible to shut down theft of intellectual property completely in the on-line world.

This is why a lot of professional photographers don't post images on-line that are bigger than low-resolution thumbnails. You can imagine that wedding photographers who make their money from selling prints (not shooting the wedding) pay very close attention to this sort of thing!

Just as people have learned to accept poor video in the interest of candor and funny content thanks to YouTube, so have people also learned to accept low-res images for free rather than paying even a nominal fee for license rights and a high-res version of the photograph or other artwork.

There is another way, however, that's demonstrated by the stock photography companies on-line: watermarking.

You've no doubt seen photos with

embedded copyright notices, Web site addresses or other content that mars the image but makes it considerably harder to separate it from its original source.

It turns out that our friend ImageMagick is terrific at creating these watermarks in a variety of different ways, and that's what I explore in this column. It's an issue for a lot of content producers, and I know the photos I upload constantly are being ripped off and reused on other sites without permission and without acknowledgement.

To do this, the basic idea is to create a watermark-only file and then blend that with the original image to create a new one. Fortunately, creating the new image can be done programmatically with the `convert` program included as part of ImageMagick.

Having said that, it's really mind-numbingly complex, so I'm going to start with a fairly uninspired but quick way to add a watermark using the `label:` feature. In a nutshell, you specify what

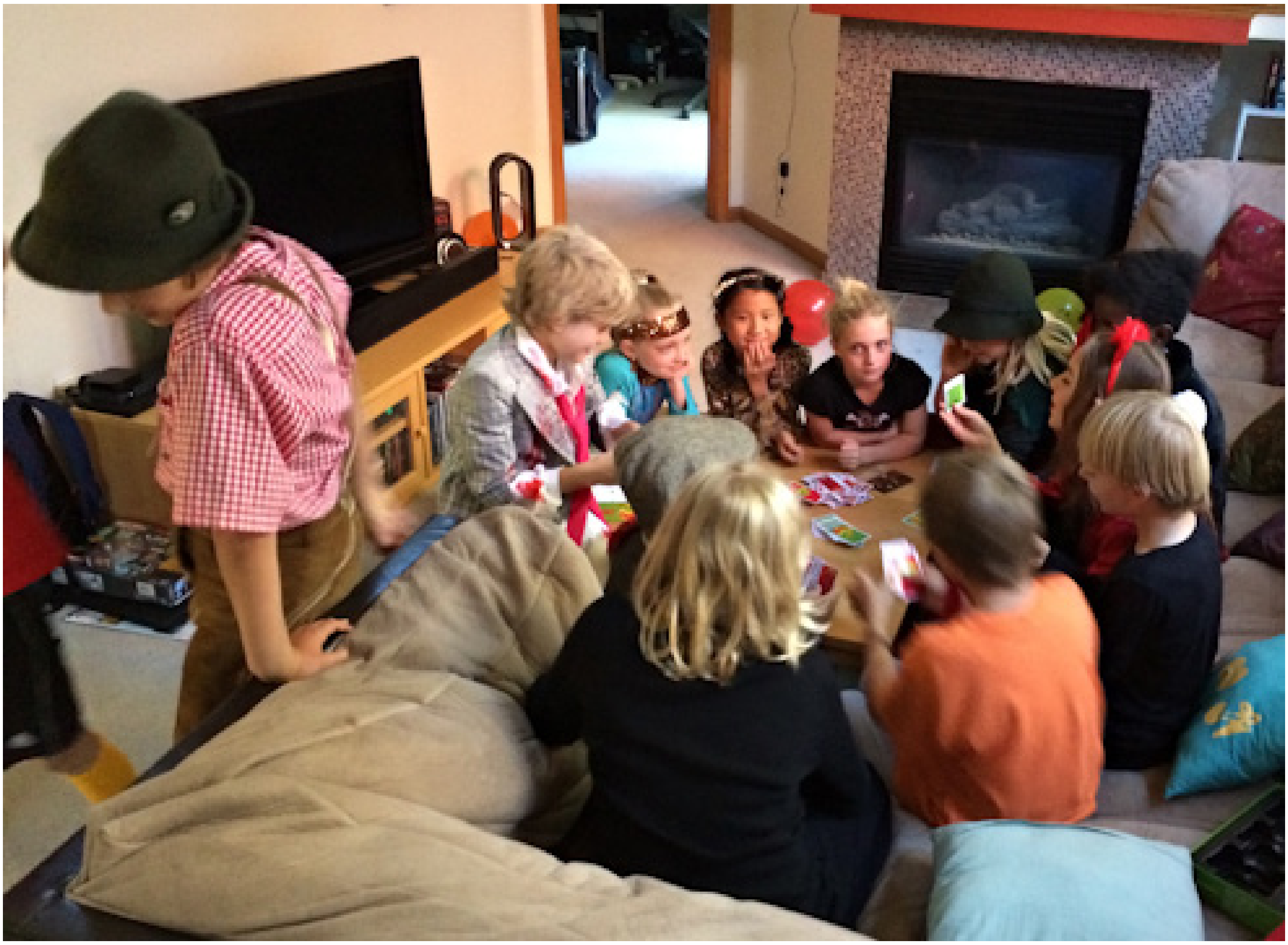


Figure 1. Original Image, Kids at a Party

text you want, where you want it on the image, the input image filename and the output image filename. Let's start with an image (Figure 1).

You can get the dimensions and so forth of the image with `identify`, of course:

```
$ identify kids-party.png
kids-party.png PNG 493x360 493x360+0+0 8-bit
➥DirectClass 467KB 0.000u 0:00.000
```

You can ignore almost all of this; it's just the size that you care about, and that's shown as 493x360.

Now, let's use `composite` to add a simple label:

```
$ composite label:'AskDaveTaylor.com' kids-party.png \
  kids-party-labelled.png
```

Figure 2 shows the image with the label applied.

That's rather boring, although it's effective in a rudimentary sort of way. Let's do something more interesting now, starting by positioning the text

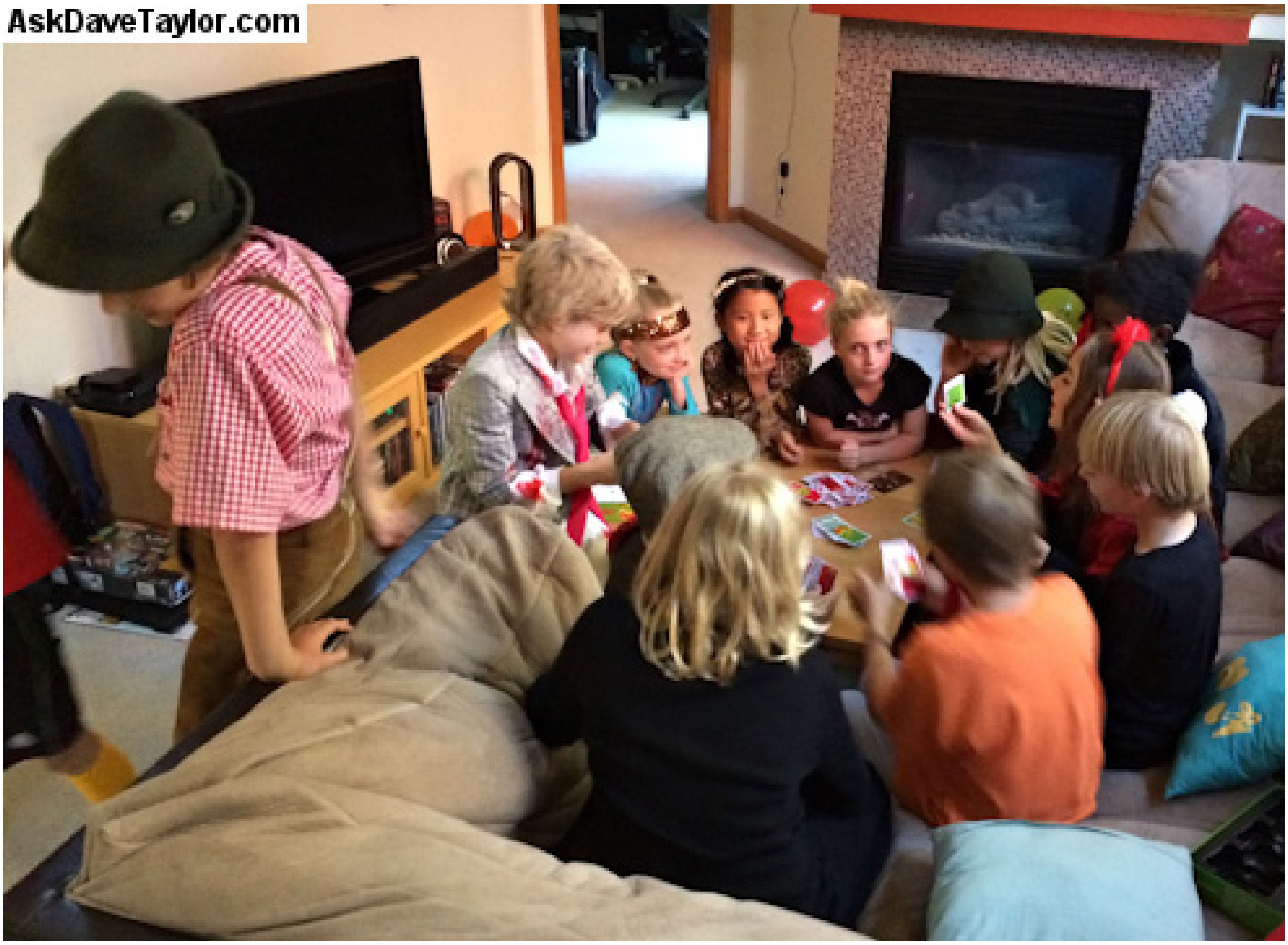


Figure 2. Label Added, No Styling

centered on the bottom but also adding space below the image for the caption:

```
$ convert kids-party.png -background Khaki \
  label:'AskDaveTaylor.com' \
  -gravity center -append party-khaki.png
```

Here I've added a background color for the new text (khaki) and tapped the complicated but darn useful `gravity` capability to center the text within the new `append` (appended) image space. Figure 3 shows the result.

I'm not done yet though. For the next example, let's actually have the text superimpose over the image, but with a semi-transparent background.

This is more ninja ImageMagick, so it involves a couple steps, the first of which is to identify the width of the original source image. That's easily done:

```
width=$(identify -format %w kids-party.png)
```

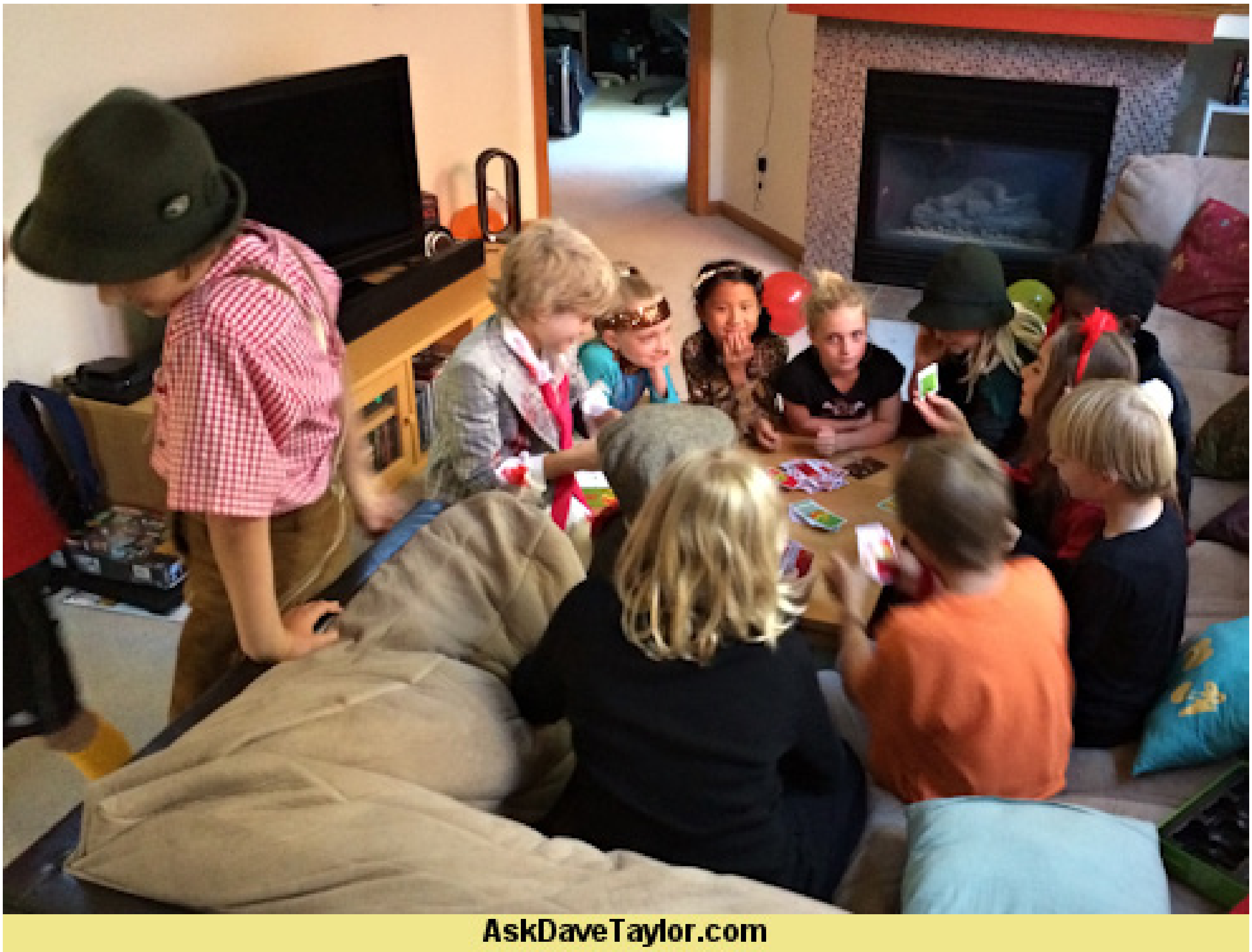


Figure 3. Caption against a Khaki Background

Run it, and you'll find out:

```
$ echo $width
493
```

Now, let's jump into the `convert` command again, but this time, let's specify a background color, a fill and a few other things to get the transparency to work properly:

```
$ convert -background '#0008' -fill white -gravity center \
  -size ${width}x30 caption:AskDaveTaylor.com \
  kids-party.png +swap -gravity south -composite \
  party-watermark.png
```

I did warn you that it'd be complex, right? Let's just jump to the results so you can see what happened (Figure 4).

You can experiment with different backgrounds and colors, but for now, let's work with this and jump to the second part of the task, turning this into a script that can fix a set of images in a folder. The basic structure

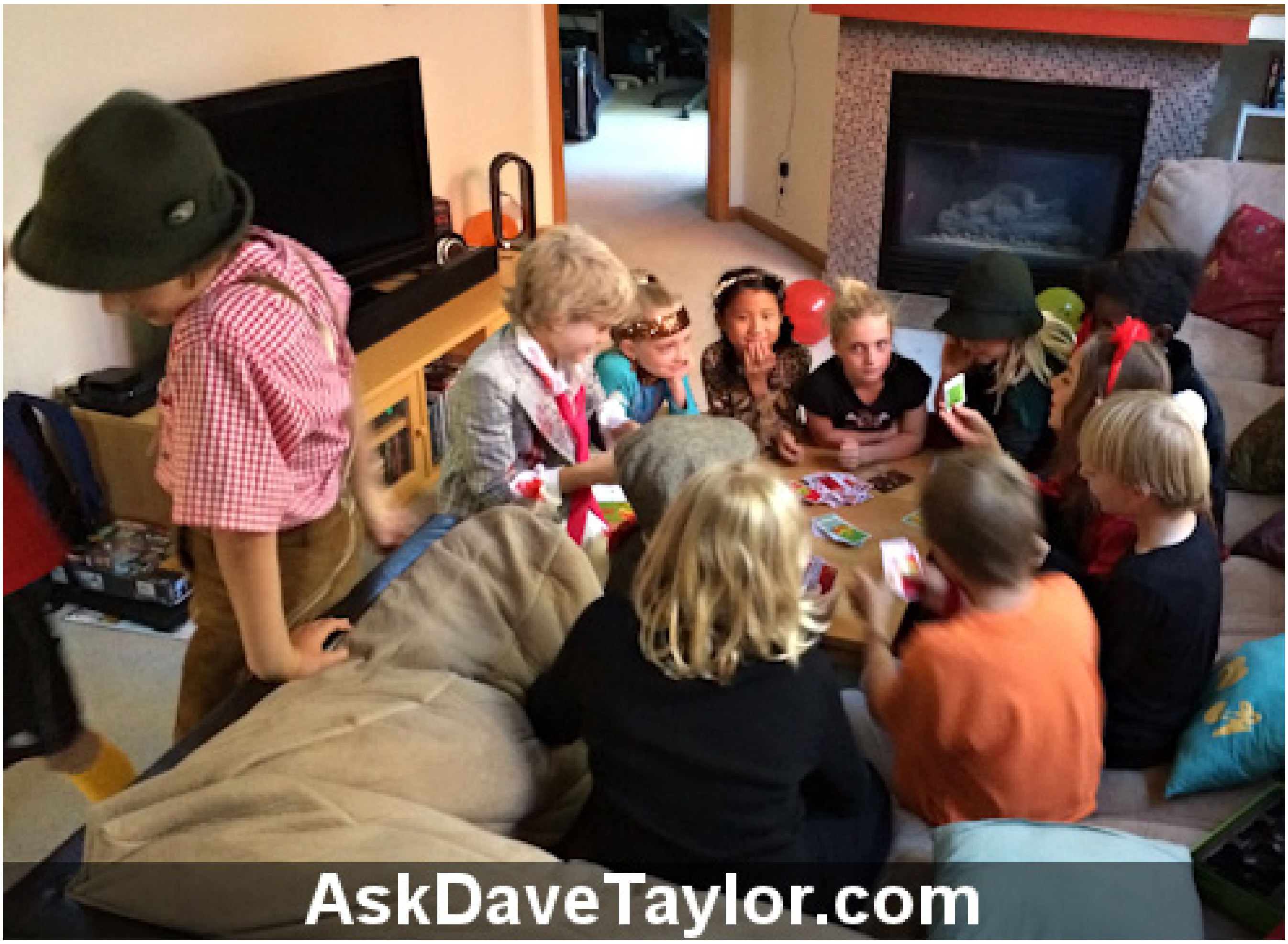


Figure 4. Improved Semi-Transparent Label

for this script will be easy actually:

```
for every image file
   calculate width
   create new watermarked version
   mv original to a hidden directory
   rename watermarked version to original image name
done
```

Because Linux is so "dot file"-friendly, let's have the script create a ".originals" folder in the current folder so that it's a nondestructive watermark process. Here's the script:

```
savedir=".originals"
mkdir $savedir

if [ $? -ne 0 ] ; then
  echo "Error: failed making $savedir."
  exit 1
fi

for image in *png *jpg *gif
do
 if [ -s $image ] ; then   # non-zero file size
    width=$(identify -format %w $image)
    convert -background '#0008' -fill white -gravity center \
       -size ${width}x30 caption:AskDaveTaylor.com \
       $image +swap -gravity south -composite new-$image
     mv $image $savedir
     mv new-$image $image
    echo "watermarked $image successfully"
  fi
done
```

You can see that it translates pretty easily into a script, with the shuffle of taking the original images and saving them in .originals.

The output is succinct when I run it in a specific directory:

```
watermarked figure-01.png successfully
watermarked figure-02.png successfully
watermarked figure-03.png successfully
watermarked figure-04.png successfully
```

Easily done.

You definitely can go further with all the watermarking in ImageMagick, but my personal preference is to tap into the reference works that already are on-line, including this useful, albeit somewhat confusing, tutorial: **http://www.imagemagick.org/Usage/annotating**.

However you slice it, if you're going to make your images available on-line in high resolution, or if they're unique and copyrighted intellectual property, knowing how to watermark them from the command line is a darn helpful skill.■

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

KYLE RANKIN

A Bundle of Tor

For privacy, windows have blinds, and Internet users have the Tor browser bundle.

I don't know how many readers know this, but my very first *Linux Journal* column ("Browse the Web without a Trace", January 2008) was about how to set up and use Tor. Anonymity and privacy on the Internet certainly take on a different meaning in the modern era of privacy-invading software and general Internet surveillance. I recently went back and read my original column, and although the first few paragraphs were written six years ago, they seem just as relevant today:

> Is privacy dead? When I think about how much information my computer and my gadgets output about me on a daily basis, it might as well be. My cell phone broadcasts my general whereabouts, and my Web browser is worse—every site I visit knows I was there, what I looked at, what browser and OS I use, and if I have an account on the site, it could know much more.

Even if you aren't paranoid (yet), you might want to browse the Web anonymously for many reasons. For one, your information, almost all of it, has value, and you might like to have some control over who has that information and who doesn't. Maybe you just want to post a comment to a blog without the owner knowing who you are. You even could have more serious reasons, such as whistle-blowing, political speech or research about sensitive issues such as rape, abuse or personal illness.

Whatever reason you have for anonymity, a piece of software called Tor provides a secure, easy-to-setup and easy-to-use Web anonymizer. If you are curious about how exactly Tor works,

> you can visit the official site at **http://tor.eff.org**), but in a nutshell, Tor installs and runs on your local machine. Once combined with a Web proxy, all of your traffic passes through an encrypted tunnel between three different Tor servers before it reaches the remote server. All that the remote site will know about you is that you came from a Tor node.

The rest of the article went into detail on how to use the Knoppix live disk to download and install Tor completely into ramdisk. Tor has come a long way since those days though, so I decided it was high time to revisit this topic and explain the best way to set up Tor on your personal machine today.

Get the Tor Browser Bundle

In the past, Tor installation meant installing the Tor software itself, configuring a proxy and pulling down a few browser plugins. Although you still can set it up that way if you want, these days, everything is wrapped up in a tidy little package called the Tor browser bundle. This single package contains Tor, its own custom Web browser already configured with privacy-enhancing settings and a user interface that makes it easy to start and stop Tor on demand.

The first step is to visit **https://www.torproject.org** and check the lock icon in your navigation bar to make sure the SSL certificate checks out. If your browser gives you some sort of certificate warning, it's possible you aren't visiting the official Tor site, and you should stop right there and attempt to get Tor from a different computer. On the main page is a large Download Tor button for you to click. If you are browsing the site from a Linux system (which of course you are), you will be presented with links to a 32-bit and 64-bit browser bundle package, so click the one that corresponds with the appropriate architecture for your system.

While the software downloads, I highly recommend you do two things. First, next to the button you clicked to download Tor, there should be a hyperlink labeled "sig". Click this link to download the signature you will use to verify that the Tor package you downloaded was legitimate (I'll talk about how to do that in a minute). The second thing you should do is scroll down the page and start reading the section titled "Want Tor to really work?" to familiarize yourself with some of the extra habits you should take on if you really do want to browse the Web anonymously.

Verify the Software

After you download the Tor browser bundle and the signature file, you should have two files in your directory:

- tor-browser-gnu-linux-x86_64-2.3.25-14-dev-en-US.tar.gz
- tor-browser-gnu-linux-x86_64-2.3.25-14-dev-en-US.tar.gz.asc

The first of these files is the software itself, and the second file is the GPG signature. Although a lot of software uses MD5 or SHA1 checksums so you can validate the software you downloaded was complete, this checksum is different. The .asc file is a cryptographic signature you can use to prove that the software you just download actually was provided to you by the Tor project and not by some malicious third party. The site provides documentation on how to verify this signature for different operating systems at **https://www.torproject.org/docs/verifying-signatures.html.en**, but since you use Linux, here you will run the following commands. First, pull down the key that was used to sign this package. Currently, this would be Erinn Clark's key (0x416F061063FEE659), which you can import with the following command:

```
$ gpg --keyserver x-hkp://pool.sks-keyservers.net
➥--recv-keys 0x416F061063FEE659
```

Once the key has been imported, you should check its fingerprint:

```
$ gpg --fingerprint 0x416F061063FEE659
pub   2048R/63FEE659 2003-10-16
      Key fingerprint = 8738 A680 B84B 3031 A630 F2DB 416F 0610 63FE E659
uid                  Erinn Clark <erinn@torproject.org>
uid                  Erinn Clark <erinn@debian.org>
uid                  Erinn Clark <erinn@double-helix.org>
sub   2048R/EB399FD7 2003-10-16
```

If the fingerprint doesn't match what you see above, something fishy is going on and you shouldn't trust this package. Of course, if you are frequent GPG users, you may want even better assurances. Hopefully, you have someone you already trust within your GPG keyring who has been to a key-signing party with Erinn Clark. If so, it would help validate that the key is legitimate.

Once you have validated the fingerprint, `cd` to the directory that has the browser bundle and .asc file, and run the following command:

```
$ gpg --verify
➥tor-browser-gnu-linux-x86_64-2.3.25-14-dev-en-US.tar.gz{.asc,}
```

```
gpg: Signature made Fri 01 Nov 2013 01:25:10 PM PDT
 ➥using RSA key ID 63FEE659
gpg: Good signature from "Erinn Clark <erinn@torproject.org>"
gpg:                 aka "Erinn Clark <erinn@debian.org>"
gpg:                 aka "Erinn Clark <erinn@double-helix.org>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature
              ➥belongs to the owner.
Primary key fingerprint: 8738 A680 B84B 3031 A630
 ➥F2DB 416F 0610 63FE E659
```

If the output says "Good signature", everything checked out. Again, you will see a warning if you don't have someone in your chain of trust that already trusts this key.

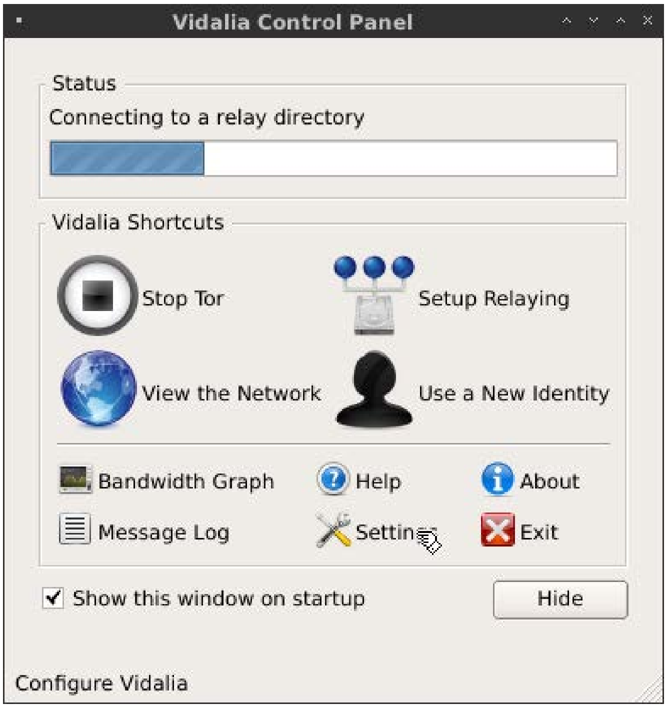
Install and Use Tor

At this point, it's relatively trivial to install and use Tor. Just use tar to extract the .tar.gz file into your home directory or wherever else you'd like it to be, and then run the start-tor-browser script inside:

```
$ tar zxvf tor-browser-gnu-linux-x86_64-2.3.25-14-dev-en-US.tar.gz
$ ./tor-browser_en-US/start-tor-browser
```

You should see a GUI window pop up that looks like Figure 1.

It may take a little time for your Tor network to finish configuring, but once it does, you will know, because a browser that looks like Figure 2 will appear.

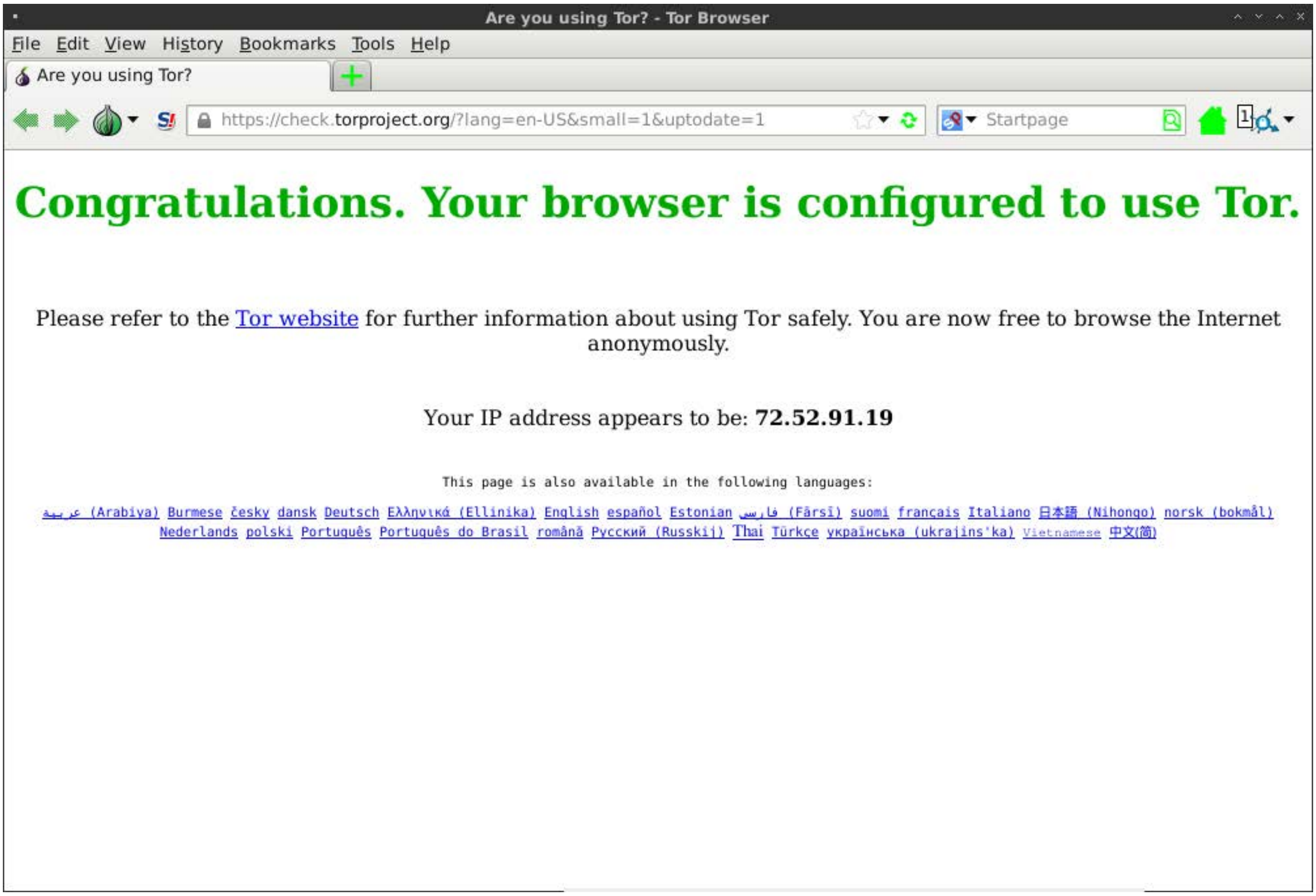

Vidalia Control Panel

Status
Connecting to a relay directory

☑ Show this window on startup

Figure 1. The Vidalia Control Panel Window

The initial Tor check page not only validates that you are using the Tor network, it also displays your current IP address. If you ever notice that IP address matches your home IP address, or if you don't see this congratulations window at all, for some reason your Tor instance isn't working properly, so you shouldn't do anything within the browser that is privacy-sensitive. Note that because you may be exiting the Tor network from an exit node in a different country, certain sites like Google, for instance, that try to be helpful and display the site in a country's native language may present you


Are you using Tor? - Tor Browser
File Edit View History Bookmarks Tools Help
Are you using Tor?
https://check.torproject.org/?lang=en-US&small=1&uptodate=1
Startpage

Congratulations. Your browser is configured to use Tor.

Please refer to the Tor website for further information about using Tor safely. You are now free to browse the Internet anonymously.

Your IP address appears to be: **72.52.91.19**

This page is also available in the following languages:

عربية (Arabiya) Burmese česky dansk Deutsch Ελληνικά (Ellinika) English español Estonian فارسی (Fārsī) suomi français Italiano 日本語 (Nihongo) norsk (bokmål)
Nederlands polski Português Português do Brasil română Русский (Russkij) Thai Türkçe українська (ukrajins'ka) Vietnamese 中文(简)

Figure 2. Congratulations, Tor works.

with Japanese, German or some other language as you visit.

If you go back to the Vidalia Control Panel, you'll notice a number of different options. You can view a map of the current global Tor network; you can click the Setup Relaying button to add your machine to the network of Tor nodes, and if you click Use a New Identity, you will stop using the three Tor nodes you currently are using and will set up a new connection with different Tor nodes. Although Tor itself does this routinely as you use it, sometimes you may want to get a different endpoint so a Web site stops displaying output in a language you don't understand.

Special Tor Browser Plugins

It's important to note that this special Tor browser has been configured with extra plugins and settings to enhance your privacy. For instance, by default, the

Noscript plugin is installed and enabled, which blocks JavaScript, Java and other plugins and allows them only for sites that you trust. The browser also includes the HTTPS Everywhere plugin that defaults to using HTTPS for any site you try to visit. You also will see a small onion icon in the navigation bar that you can use to tweak your Tor preferences inside the browser.

Once you are done browsing anonymously, close your browser and go back to the Vidalia Control Panel. If you are done using Tor completely, click the Stop Tor button, and then click exit to close the application. Browsing the Web anonymously and privately has never been this easy.■

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.

Sinatra | HTML5 | Catalyst | Drupal 7 | Moodle
LINUX JOURNAL
INTRO TO THE SINATRA MICRO-FRAMEWORK
THE BEST WAY TO CLIENT CONCURRENCY
WEB DEVELOPMENT
CREATE Web Apps with Catalyst and Perl
NEW COLUMN: THE OPEN-SOURCE CLASSROOM

SHAWN POWERS

Encrypting Your Cat Photos

Encryption is powerful and scary. Let's remove the scary.

The truth is, I really don't have anything on my hard drive that I would be upset over someone seeing. I have some cat photos. I have a few text files with ideas for future books and/or short stories, and a couple half-written starts to NaNoWriMo novels. It would be easy to say that there's no point encrypting my hard drive, because I have nothing to hide. The problem is, we wrongly correlate a "desire for privacy" with "having something to hide". I think where I live, in America, we've taken our rights to privacy for granted. Rather than the traditional "he must be hiding porn or bombs", think about something a little more mundane.

I live in Michigan. It's cold here in the winter, and I tend to keep my thermostat set around 75 degrees. That might seem high to you, but for my family, it's just right. Thanks to the privacy of my own home, my neighbors don't know how toasty warm we keep it. Some of those neighbors would be very upset to see how "wasteful" the Powers family is in the winter. In fact, there's one local man who makes it a point to let everyone know that anything over 60 degrees is ecologically wasteful. I don't want to get into a fight with Old Man Icebritches, so we just keep our comfortable house a secret. We don't have anything to hide, but it's not something everyone needs to know about.

Obviously my example is silly, but hopefully it makes you think. Modern Linux allows us to encrypt our data easily and reliably, so why not take advantage of it?

How Does It Work?

I won't go into too much detail about how encryption works, but a basic understanding is necessary for even the simplest implementation. To encrypt and decrypt a file, two

Modern Linux allows us to encrypt our data easily and reliably, so why not take advantage of it?

"keys" are required. One is the private key, which is just that, private. I like to think of the private key as an actual key—you can make copies if you want, but it's not wise to do so. The more copies of your private keys you make, the more likely someone nefarious will get one and break into your apartment—er, I mean files.

The public key is more like a schematic for a lock that only you can open (with your private key). You make this key available for anyone. You can post it on a Web site, put it in your e-mail, tattoo it on your back, whatever. When others want to create a file that only you can see, they encrypt it using your public key.

This one-to-many scenario also has a cool side effect. If you encrypt something using your private key, anyone can decrypt it using your public key. This may sound silly, but what makes such a scenario useful is that although the encrypted file isn't protected from prying eyes, it is guaranteed to be from you. Only a file encrypted with your private key can be decrypted with your public key. In this way, encrypting something with your private key digitally "signs" the file.

Usually it works like this:

1. You have a file you want to send to Suzy, so you encrypt it with Suzy's public key. Only Suzy can open it, but there's no way for Suzy to know that you are the one who sent it, since anyone could encrypt a file with her public key.

2. Therefore, you take the file you encrypted with Suzy's public key and encrypt *that* file with *your* private key. Suzy will have to decrypt the file twice, but she'll know it came from you.

3. Suzy receives the file and decrypts the first layer with your public key, proving it came from you.

4. Suzy then decrypts the second layer of encryption with her private key, as that's the only key able to decrypt the original file. (Because you originally encrypted

it with her public key.)

That scenario is when encryption is used for safely transferring files, of course. It's also quite common simply to encrypt your files (or partitions) so that no one can see them unless you decrypt them first. Let's start with file encryption, because that's what most people will want to do on their systems.

Starting Simple

Before I go into more complex type setting, let's discuss simply encrypting a file. There are various programs to handle encryption. In fact, it's easy to get overwhelmed with the available options for file and system encryption. Today, let's use a basic (but very powerful) command-line tool for encrypting a file. GPG (Gnu Privacy Guard) is an open-source implementation of PGP (Pretty Good Protection). It allows encryption and signing, and manages multiple keys and so on. For this example, let's simply encrypt a file.

Let's say you have a file called secret_manifesto.txt, which contains the secrets to life, the universe and everything. Using GPG, you can encrypt the file with a passphrase. Using a passphrase is far simpler than using a public and private key pair, because it's simply encrypted using your passphrase. This does make your file more susceptible to cracking (using rainbow tables or other hacking tools), but like the label on the tin says, it's Pretty Good Protection. To encrypt your file, you can do this:

```
# gpg -c secret_manifesto.txt
# Enter passphrase:
# Repeat passphrase:
```

Once complete, you'll have a new file in the same directory. It will be named secret_manifesto.txt.gpg by default. This is a binary file, which means it's fairly small, but it can't be copy/pasted into an e-mail or IM. For portability, you can add the -a flag, which will create an encrypted file that contains only ASCII text:

```
# gpg -a -c secret_manifesto.txt
# Enter passphrase:
# Repeat passphrase:
# ls -l
-rw-rw-r--  1 spowers spowers    6 Nov 23 1:26 secret_manifesto.txt
-rw-rw-r--  1 spowers spowers  174 Nov 23 1:27 secret_manifesto.txt.asc
-rw-rw-r--  1 spowers spowers   55 Nov 23 1:26 secret_manifesto.txt.gpg
```

Notice there is now a file with .asc as the extension. This is text-only, but you can see in the code snippet that it's also much larger than the binary encrypted file, and much much larger than the original text file. Once you've encrypted your file, if you truly want to keep your information secret, it would be wise to delete the original text file.

To decrypt the file, you'll again use the gpg program. The same command will decrypt either file, whether it's binary or ASCII:

```
# gpg secret_manifesto.txt.asc
# gpg: CAST5 encrypted data
# Enter passphrase:
# gpg: encrypted with 1 passphrase
# File `secret_manifesto.txt' exists. Overwrite? (y/N)
```

Notice in the example above, I hadn't deleted the original text file, so gpg gave me the option of overwriting. Once complete, I have my original file back, unencrypted. If you just have a file or two you want to protect, the command-line gpg program might be all you need. If you'd rather have an area on your system that automatically encrypts everything you save, it's a little more complicated. It's still not terribly difficult, but let's start with a fairly simplistic model.

Encrypting a USB Drive

Like I mentioned earlier, there are many options when it comes to encryption. One of the more popular methods of encrypting partitions is the LUKS (Linux Unified Key Setup) system. A USB drive with a LUKS-formatted partition should be detected automatically by most systems. In fact, if you're using a desktop environment like Ubuntu Desktop, encrypting a USB drive is a simple check box during the formatting process. Although that's a perfectly acceptable way to encrypt your USB drive, I'm going to demonstrate how to do it on the command line, so you understand what's actually happening behind the scenes.

Step 1: identify your USB drive. If you type `dmesg` after plugging in your USB drive, you should get all sorts of system information, including the device name of your freshly plugged-in USB device. Make sure you have the correct device identified, because what you're doing will destroy any data on the drive. You wouldn't want to format the wrong disk accidentally. (It should go without saying, but I'll say it anyway, make sure there's nothing on your USB drive that you want to save—this is a destructive process.)

Step 2: partition the USB drive. Assuming that your USB drive is the

/dev/sdb device on your system, you need to create a single partition on the drive. Let's use fdisk. Below is the interaction with fdisk required. Basically, you create a new empty partition with the o command, then write changes with w. Then, you'll restart fdisk and use the n command to create a new primary partition, using the defaults so that the entire drive is used:

```
# sudo fdisk /dev/sdb

Command (m for help): o
Building a new DOS disklabel with disk identifier 0x1234567.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Command (m for help): w
The partition table has been altered!

# sudo fdisk /dev/sdb
Command (m for help): n
Command action
e   extended
p   primary partition (1-4)
p
Partition number (1-4, default 1): 1
Using default value 1
First sector (2048-1016522, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-1016522, default 1016522):
Using default value 1016522

Command (m for help): w
The partition table has been altered!
```

Now you have a USB drive with a single partition (/dev/sdb1), but there is no filesystem on it. That's exactly what you want, because the LUKS system creates an encryption layer on the partition *before* you put a filesystem on it. So before creating a filesystem, let's create the LUKS layer on the partition, using the cryptsetup program. If you don't have cryptsetup, search for it in your distribution's repository; it should be there. To create the LUKS encrypted partition layer:

```
# cryptsetup luksFormat /dev/sdb1

WARNING!
========
This will overwrite data on /dev/sdb1 irrevocably.

Are you sure? (Type uppercase yes): YES
Enter LUKS passphrase:
Verify passphrase:
```

Follow the directions, and be sure to remember your passphrase! Note, that a "passphrase" is usually more than just a word. It's most often a phrase, thus the name. The longer the phrase, the tougher to crack.

In fact, when you put the USB drive into your computer, if you have a modern GUI desktop, it should prompt you for a password and mount it automatically.

Once the process completes, you have an encrypted partition, but it's not mounted or formatted yet. The first step is to mount the partition, which again uses the cryptsetup utility:

```
# cryptsetup luksOpen /dev/sdb1 my_crypto_disk
Enter passphrase for /dev/sdb1:
```

When you type in your passphrase, the device name you entered will be mounted like a virtual hard drive. Usually, it's mounted under /dev/mapper/devicename, so this example mounts a partition at /dev/mapper/my_crypto_disk.

This device is now being accessed as an unencrypted volume. As long as it stays mounted, it will act like any other unencrypted volume. That means you need to write a filesystem to it if you want to use it:

```
# mkfs.vfat /dev/mapper/my_crypto_disk -n my_crypto_disk
mkfs.vfat 3.0.9 (31 Jan 2010)
```

Now the drive is fully functional and can be mounted like any other disk. In fact, when you put the USB drive into your computer, if you have a modern GUI desktop, it should prompt you for a password and mount it automatically. Then you can eject it like a normal disk, and it will be encrypted until you next enter your passphrase. It's simple to unmount and, therefore, re-encrypt the drive on the command line too, using cryptsetup:

```
# cryptsetup luksClose my_crypto_disk
```

That's Only the Tip of the Iceberg

In this article, my hope is to peel back some of the mystery behind encryption. It's simple to encrypt and decrypt a file. It's not too much more difficult (especially if you use the GUI desktop tools) to encrypt an entire USB drive. With most distributions, it's possible to encrypt the entire home directory during the installation process!

Once you get the encryption bug, I must warn you, you'll want to start encrypting everything.

When encryption is set up on your entire home directory, however, there are some issues you need to address. For example, jobs that run while you're not logged in most likely will not have access to your home directory. If you have cron jobs that need access to your home directory, you should rewrite them to access data elsewhere on the system. I find a happy medium between security and convenience is to encrypt a USB drive and store my personal data on it.

Once you get the encryption bug, I must warn you, you'll want to start encrypting everything. That's not a bad thing, but like the home directory scenario, you'll run into a few snags. Cross-platform accessibility is a big one if you go between systems. For situations like that, I highly recommend TrueCrypt (**http://www.truecrypt.org**). I've mentioned TrueCrypt in UpFront pieces before, but it's basically an open-source, cross-platform encryption system that allows you to encrypt files, folders, partitions and more while being able to access that data on any system. Windows, Mac and Linux clients are all available, and the community has great support.

You don't have to have something to hide in order to desire encryption for your files. Just like it's wise to lock your house at night, even if you live in a good neighborhood, it's a smart move to encrypt your personal data. If you want to share your photos of Mr Whiskerton in his cute little beanie hat with everyone on the Internet, that's your right. But others don't need to see those things if they're being nosey and poking around your hard drive!■

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.

archive: 1994-2013
LINUX JOURNAL
TWENTY YEARS of LINUX JOURNAL on One DVD

Innodisk's FlexiArray SE108 and HD224 Storage Appliances

The secret to the performance advances in Innodisk's FlexiArray line of storage appliances is the company's novel FlexiRemap Technology, which deals with the challenges of I/O performance, data endurance and affordability. FlexiRemap, notes Innodisk, innovates in software and firmware, creating a new category of Flash-collaborating storage appliances (in contrast to Flash-aware or Flash-optimized) that deliver sustained high IOPS, even for random write operations. Innodisk's first storage appliances to leverage this technology, the new FlexiArray SE108 and HD224, are designed to provide cost-effective performance for high-performance computing, cloud computing and I/O bound server applications. Typical application areas include cloud computing, virtualization and HPC. The slim SE108 offers up to 2TB of storage in a 1U-rackmount package; the HD224 provides up to 8TB in a 2U-rackmount unit, with 8x 10GbE SFP+ interfaces. Both units offer redundant hot-swappable SSDs and power modules.

http://flexiarray.innodisk.com

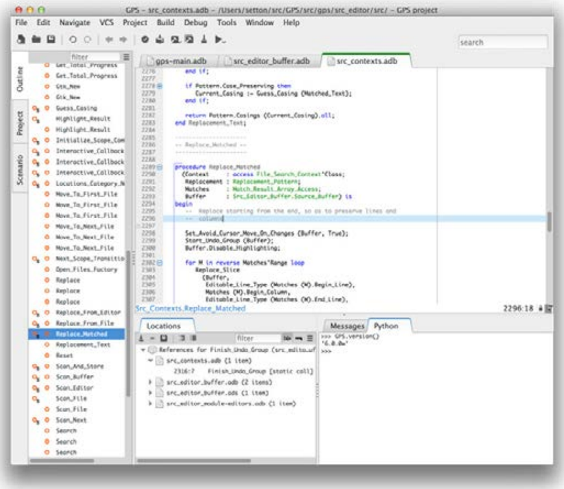
salesforce.com
ORACLE
SAP
SUGARCRM
SharePoint
magic
sage
Microsoft Dynamics
facebook
Google

Magic Software Enterprises' Magic xpi Integration Platform

With most core enterprise systems in place, organizations of all sizes are looking to business process integration and automation to increase operational efficiency and competitiveness. The updated Magic xpi Integration Platform from Magic Software Enterprises is a cloud-ready integration platform that enables users to unlock data from enterprise systems like SugarCRM, Sage and SYSPRO. In the new release, the aforementioned three platforms now enjoy certified, prebuilt adapters for optimized integration, which complement existing adapters for Oracle JD Edwards EnterpriseOne, JD Edwards World, SAP, IBM Lotus Notes, Microsoft Dynamics, Microsoft SharePoint and Salesforce, and more. In addition, an In-Memory Data Grid (IMDG) architecture is the new standard. IMDG offers cost-effective elastic scalability, built-in clustering and failover capabilities, which support enterprise needs for business continuity, faster processing and increasing transaction loads spurred by new mobile, cloud and big-data use cases.

http://www.magicsoftware.com

AdaCore's GNAT Programming Studio


GPS – src_contexts.adb – /Users/setton/src/GPS/src/gps/src_editor/src/ – GPS project
File Edit Navigate VCS Project Build Debug Tools Window Help
search
gps-main.adb
src_editor_buffer.adb
src_contexts.adb
Src_Contexts.Replace_Matched
2296:18
Locations
Messages
Python

```
      end if;

      if Pattern.Case_Preserving then
         Current_Casing := Guess_Casing (Matched_Text);
      end if;

      return Pattern.Casings (Current_Casing).all;
   end Replacement_Text;

   ---------------------
   -- Replace_Matched --
   ---------------------

   procedure Replace_Matched
     (Context     : access File_Search_Context'Class;
      Replacement : Replacement_Pattern;
      Matches     : Match_Result_Array_Access;
      Buffer      : Src_Editor_Buffer.Source_Buffer) is
   begin
      --  Replace starting from the end, so as to preserve lines and
      --  columns

      Set_Avoid_Cursor_Move_On_Changes (Buffer, True);
      Start_Undo_Group (Buffer);
      Buffer.Disable_Highlighting;

      for M in reverse Matches'Range loop
         Replace_Slice
           (Buffer,
            Editable_Line_Type (Matches (M).Begin_Line),
            Matches (M).Begin_Column,
            Editable_Line_Type (Matches (M).End_Line),
```

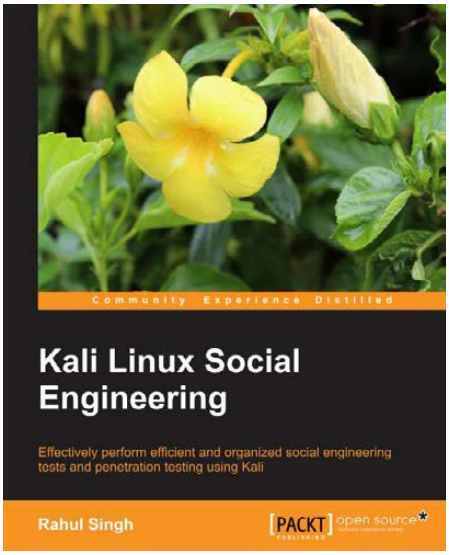
```
References for Finish_Undo_Group (src_edito.uf
  src_contexts.adb (1 item)
    2316:7   Finish_Undo_Group [static call]
  src_editor_buffer.adb (2 items)
  src_editor_buffer.ads (1 item)
  src_editor_module-editors.adb (1 item)
```

```
>>> GPS.version()
'6.0.0w'
>>>
```

"Usability" is the word that best captures the essence of the new version 6.0 release of AdaCore's GNAT Programming Studio (GPS) graphical IDE. This "major engineering effort" features a significantly revised and cleaner user interface that eases program navigation and editing. The revised look and feel, which exploits the latest Gtk+/GtkAda graphical toolkit, is supported by a new relational database at the heart of the GPS engine, making code navigation much more efficient. GPS 6.0 also brings improved performance and new functionality, including language support for SPARK 2014, syntax highlighting and tool tips for Ada 2012 and SPARK 2014 aspects, editor enhancements and a number of additions to the scripting API.

http://www.adacore.com

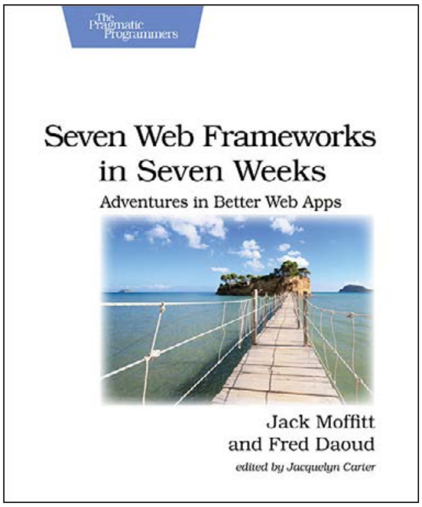
Rahul Singh's *Kali Linux Social Engineering* (Packt Publishing)


Community Experience Distilled
Kali Linux Social Engineering
Effectively perform efficient and organized social engineering tests and penetration testing using Kali
Rahul Singh
[PACKT] open source
PUBLISHING

The new book *Kali Linux Social Engineering* by Rahul Singh exists to help you master the social engineering toolkit, or SET, found in the security-focused Kali Linux distribution. With Singh's book in hand, readers can learn how security can be breached using social-engineering attacks, as well as attain a very unique ability to perform a security audit based on social engineering attacks. Starting with attacks using Kali, this book describes in detail various Web site attack vectors and client side attacks that can be performed through SET. This book covers some of the most advanced techniques that currently are being utilized by attackers to get inside secured networks, covering phishing (credential harvester attack), Web jacking attack method, spear phishing attack vector, Metasploit browser exploit method, Mass mailer attack and more.

http://www.packtpub.com

Jack Moffitt and Fred Daoud's *Seven Web Frameworks in Seven Weeks* (Pragmatic Bookshelf)


The Pragmatic Programmers
Seven Web Frameworks in Seven Weeks
Adventures in Better Web Apps
Jack Moffitt and Fred Daoud
edited by Jacquelyn Carter

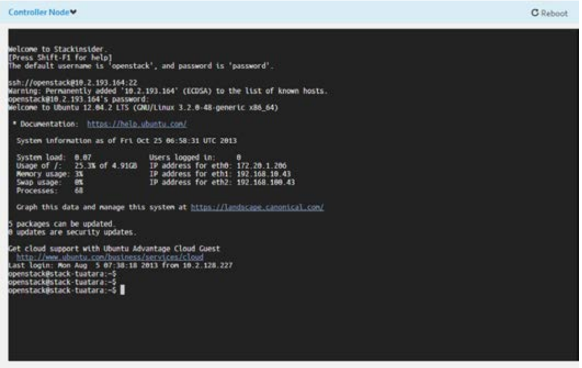
There's something to the "Seven in Seven Weeks" concept in the tech books from Pragmatic Bookshelf. The latest addition in this practical series is Jack Moffitt and Fred Daoud's *Seven Web Frameworks in Seven Weeks: Adventures in Better Web Apps*. Whether you need a new tool or merely a dose of inspiration, this work explores your options and gives you sufficient exposure to each one, along with tips for creating better apps. The authors cover frameworks that leverage modern programming languages, employ unique architectures, live client-side instead of server-side or embrace type systems. Covered frameworks include Sinatra, CanJS, AngularJS, Ring, Webmachine, Yesod and Immutant. The breakneck evolution of Web apps demands innovative solutions, and this survey of frameworks and their unique perspectives is designed to inspire and promote new thinking for dealing with daily programming challenges.

http://www.pragprog.com

OpenLogic's AWS Marketplace Offerings

OpenLogic's vision is to keep enterprise customers running on some of the world's best open-source packages. To convert this vision into reality, the firm intends to make available more than 50 new preconfigured stacks through the Amazon Web Services (AWS) Marketplace, including production-level support for JBoss, Apache HTTP, Tomcat, MySQL, PostgreSQL, ActiveMQ and the CentOS operating system. These are in addition to OpenLogic's existing offerings on AWS. Enterprise support will include both 12x5 business-hour support and 24x7 production-level support. Products will be offered for use at an hourly rate. OpenLogic adds that OLEX, its open-source scanning, governance and provisioning portal, allows organizations to embrace open source with confidence.

http://www.openlogic.com

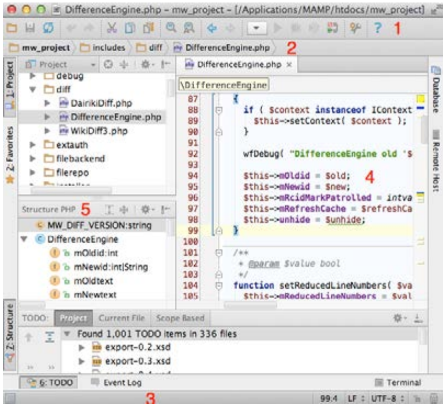


```
Welcome to Stackinsider.
[Press Shift-F1 for help]
The default username is 'openstack', and password is 'password'.

ssh://openstack@10.2.193.164:22
Warning: Permanently added '10.2.193.164' (ECDSA) to the list of known hosts.
openstack@10.2.193.164's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Fri Oct 25 06:58:31 UTC 2013

  System load:  0.07              Users logged in:     0
  Usage of /:   25.3% of 4.91GB   IP address for eth0: 172.20.1.206
  Memory usage: 3%                IP address for eth1: 192.168.10.43
  Swap usage:   0%                IP address for eth2: 192.168.100.43
  Processes:    68

  Graph this data and manage this system at https://landscape.canonical.com/

5 packages can be updated.
0 updates are security updates.

Get cloud support with Ubuntu Advantage Cloud Guest
  http://www.ubuntu.com/business/services/cloud
Last login: Mon Aug  5 07:38:18 2013 from 10.2.128.227
openstack@stack-tuatara:~$
openstack@stack-tuatara:~$
openstack@stack-tuatara:~$
```

Stackinsider Deployment-as-a-Service Cloud Platform

Stackinsider's approach to OpenStack is packaging it as a Deployment-as-a-Service (DaaS) cloud platform, which the company says is the first of its kind to be public and free. Designed to make OpenStack technology adoption significantly easier and faster than conventional approaches, the Stackinsider DaaS approach consolidates and streamlines key OpenStack distributions and real-world applications for a wide range of uses. DaaS has integrated all popular IaaS deployment toolchains including RDO, FUEL, Puppet, DevStack and Chef. Some popular applications like Moodle and SugarCRM also are provided for PaaS prototyping. This public DaaS cloud is available for download at Stackinsider's Web site. http://www.stackinsider.com

JetBrains' PhpStorm

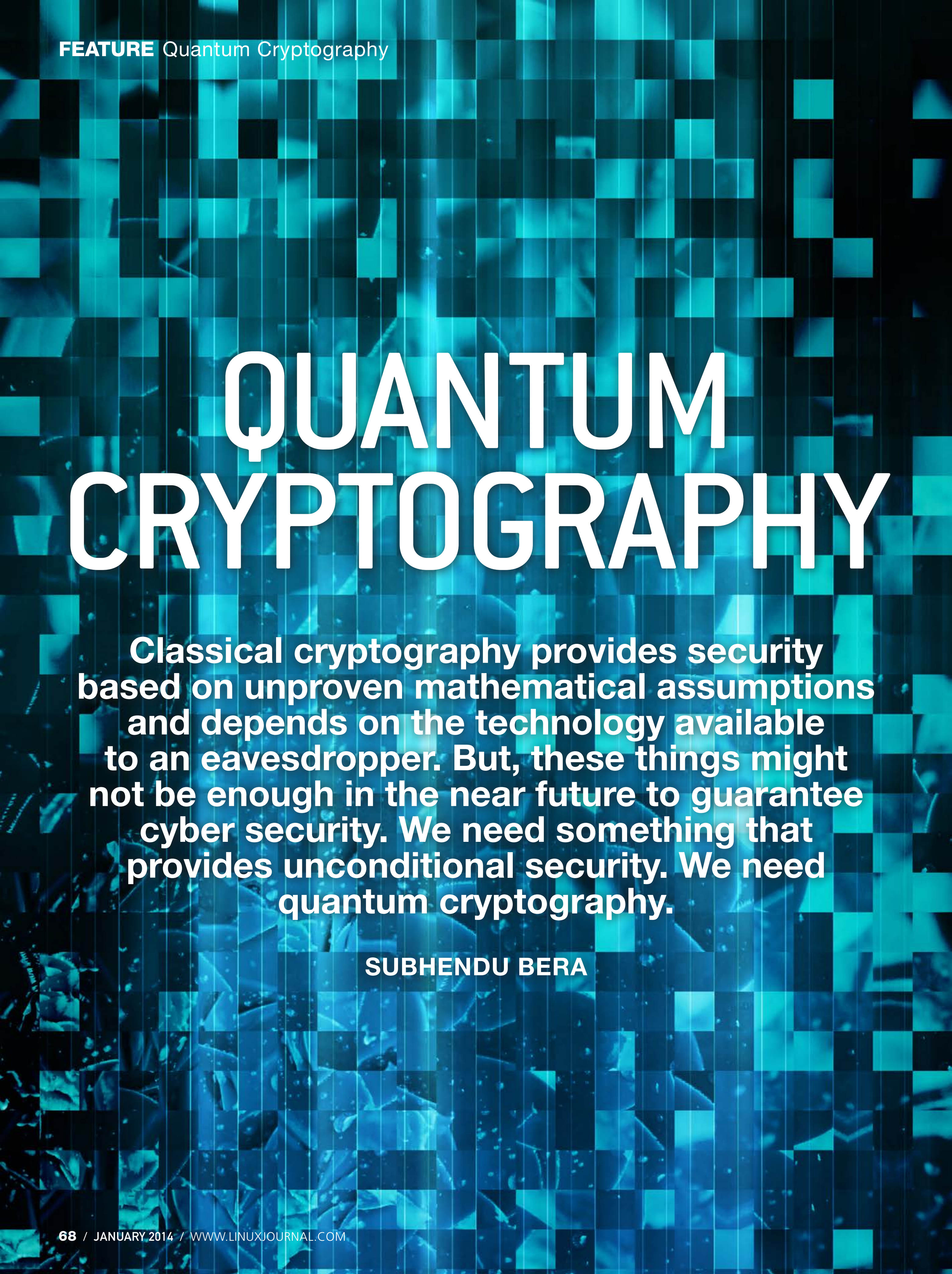


```
{
  if ( $context instanceof IContext
    $this->setContext( $context );
  }

  wfDebug( "DifferenceEngine old '$

  $this->mOldid = $old;
  $this->mNewid = $new;
  $this->mRcidMarkPatrolled = intva
  $this->mRefreshCache = $refreshCa
  $this->unhide = $unhide;
}

/**
 * @param $value bool
 */
function setReducedLineNumbers( $va
  $this->mReducedLineNumbers = $val
```

```
Found 1,001 TODO items in 336 files
  export-0.2.xsd
  export-0.3.xsd
```

For JetBrains, developing a new version of the PhpStorm IDE for PHP means more than keeping on top of the latest changes in Web languages. It is also about supporting and integrating modern tools and popular frameworks, not to mention removing obstacles on the road to productive Web development. Of course, the new PhpStorm 7 supports the latest PHP 5.5 with improved PHP syntax coloring, new refactorings, code inspections and quick-fixes. Support also has been added for various front-end Web technologies, such as different JavaScript templates, Web Components and modern stylesheets. Built-in tools for Vagrant, SSH console and local terminal and Google App Engine for PHP have been added too. Finally, support has been enhanced for various frameworks, including Drupal, Symfony2 and others. http://www.jetbrains.com/phpstorm

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.



QUANTUM CRYPTOGRAPHY

Classical cryptography provides security based on unproven mathematical assumptions and depends on the technology available to an eavesdropper. But, these things might not be enough in the near future to guarantee cyber security. We need something that provides unconditional security. We need quantum cryptography.

SUBHENDU BERA

Imagine you want to send a message to your friend, and you don't want others to be able to read the message. You lock your message in a box using a key and send the box to your friend. Your friend also has a key to unlock that box, so he easily can open the box and read the message. In general, this is the technique used by cryptographic algorithms. Locking the message in the box is like encryption, and unlocking the box is like decryption. Before sending the message to the receiver, the data is encrypted using an encryption algorithm and a secret key. On the receiver side, the encrypted data is decrypted using the reverse encryption algorithm.

Classical cryptographic algorithms mostly rely on mathematical approaches to secure key transmission. The security they offer is based on unproven assumptions and depends on the technology available to an eavesdropper. But, rapidly growing parallel and quantum technologies may be a threat to these classical cryptography techniques in the near future. One of the solutions to these threats is quantum cryptography.

What is quantum cryptography? Quantum cryptography is a complex topic, because it brings into play something most people find hard to understand—quantum mechanics. So first, let's focus on some basic quantum physics that you'll need to know to understand this article.

QUANTUM CRYPTOGRAPHY IS A COMPLEX TOPIC, BECAUSE IT BRINGS INTO PLAY SOMETHING MOST PEOPLE FIND HARD TO UNDERSTAND—QUANTUM MECHANICS.

Simple Quantum Physics

Quantum, in physics, is a discrete natural unit, or packet of energy, charge, angular momentum or other physical property. Light, for example, appears in some respects as a continuous electromagnetic wave, but on the submicroscopic level, it is emitted and absorbed in discrete amounts or quanta. These particle-like packets (quanta) of light are called photons, a term also applicable to quanta of other forms of electromagnetic energy, such as

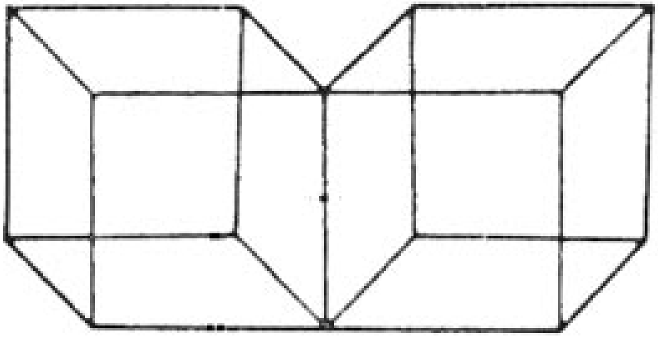


Figure 1. Necker Cubes

X rays and gamma rays.

One unique thing about quanta is that they can exist in all of their possible states at once. This also applies to photons. This means that in whatever direction a photon can spin—say, diagonally, vertically and horizontally—it does so all at once. Quantum of light in this state is called unpolarized photons. This is like someone moving north, south, east, west, up and down all at the same time. This property is called superposition. One thing you should keep in mind is that measuring something that is in its superposition causes it to collapse into a definite state (one of all the possible states). Figure 1 should help describe superposition.

Looking at Figure 1, you can identify one of four possibilities: either both squares are protruding forward or both are backward, or one is forward and the other is backward. Each time you look at the diagram, only one possibility is true. In a sense, all four options exist together, but when you look at the diagram, it collapses into just one. This is the essence of quantum superposition.

Through the use of polarization filters, you can force the photon to

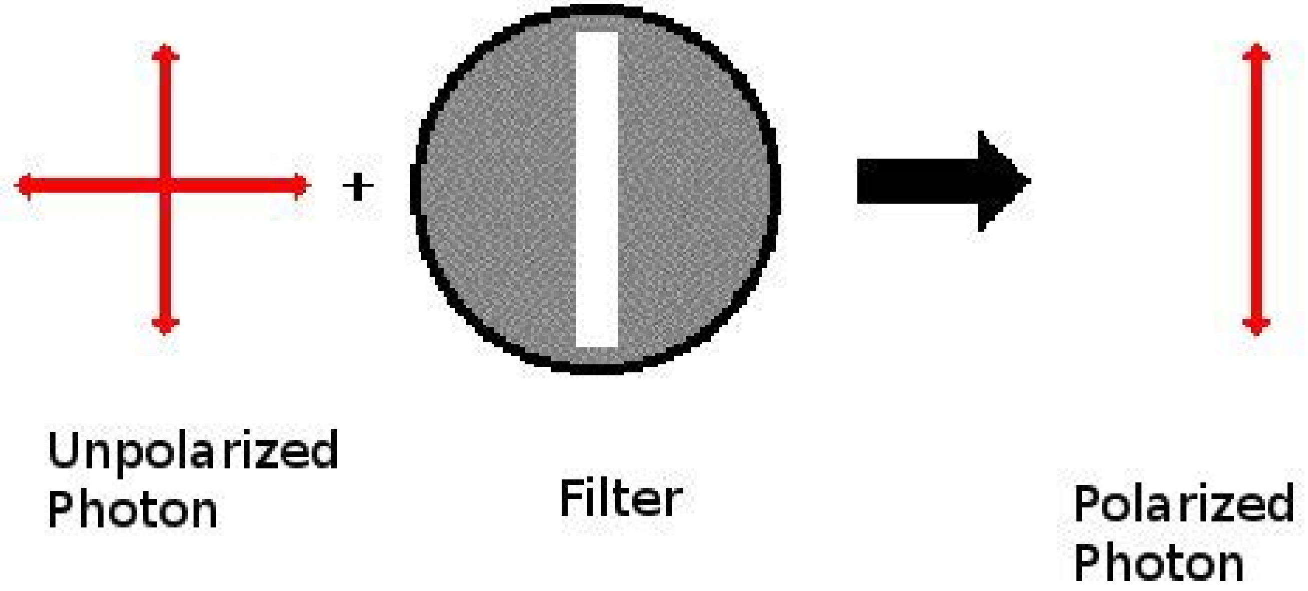


Figure 2. Polarizing Photons

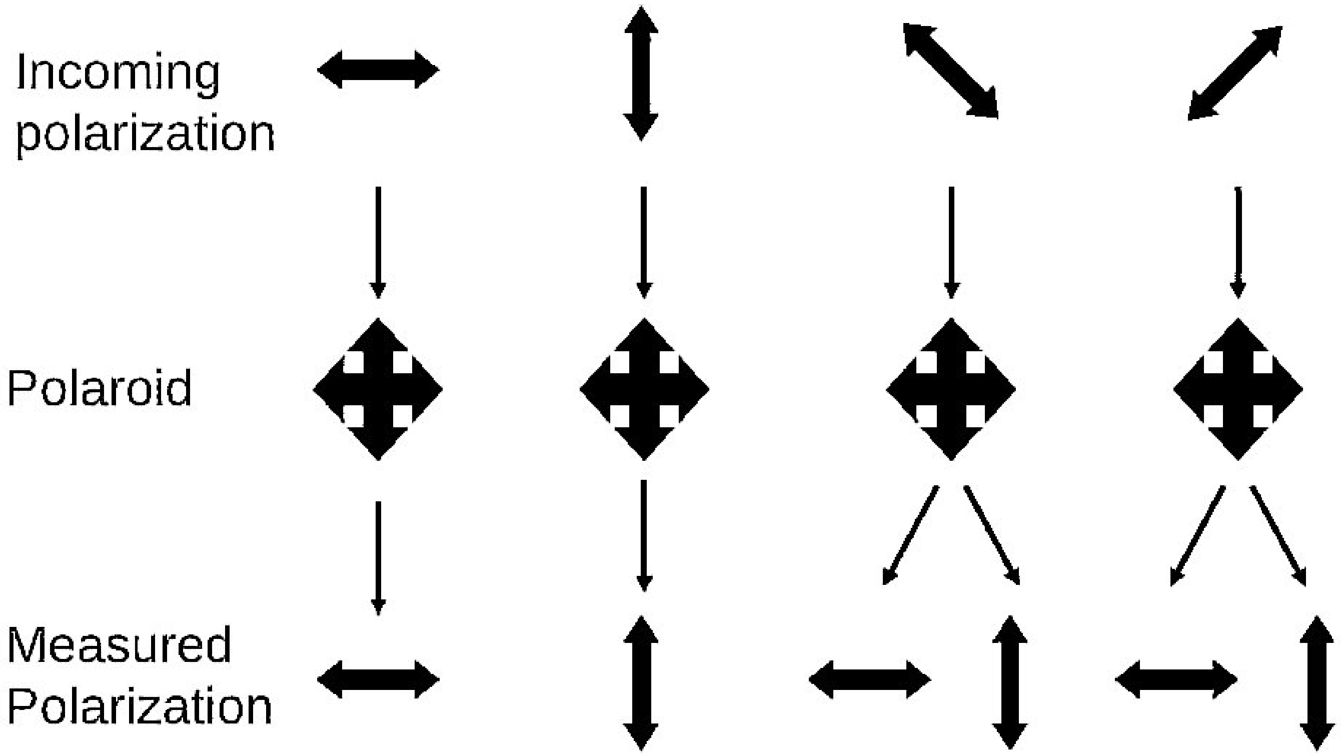


Figure 3. Effect of Various Basis on Polarized Photons

take one of its states, or technically, polarize it. If you use a vertical polarizing filter, some photons will be absorbed, and some will emerge on the other side of the filter. Those photons that aren't absorbed will emerge on the other side with a vertical spin. Thus, you can polarize the photons to your required orientation using suitable filters.

The foundation of quantum physics is the unpredictability factor. This unpredictability is pretty much defined by Heisenberg's Uncertainty Principle. This principle says that certain pairs of physical properties are related in such a way that measuring one property prevents the observer from knowing the value of the other. But, when dealing with photons for encryption, Heisenberg's Principle can be used to your advantage. When measuring the polarization of a photon, the choice of what direction to measure affects all subsequent measurements. The thing about photons is that once they are polarized, they can't be measured accurately again, except by a filter like the one that initially produced their current spin. So if a photon with a vertical spin is measured through a diagonal filter, either the photon won't pass through the filter or the filter will affect the photon's behavior, causing it to take a diagonal spin. In this sense, the information on the photon's original polarization is lost.

In the diagram in Figure 3, I have used the wrong basis for the last two cases, and you can see that I have changed the polarization of two photons.

Quantum Information

The bit is the fundamental concept of classical computation and classical information. Quantum computation and quantum information are built upon an analogous concept: the quantum bit, or qbit for short. Just as a classical bit has a state of either 0 or 1, a qbit is like a bit, but it is in superposition between 0 and 1. Two possible states for a qbit are the states “$|0 >$” and “$|1 >$” . This notation is called Dirac notation. A qbit can be fully expressed as: $a|0 > +b|1 >$ with $a^2 + b^2 = 1$. When we measure a qbit, we get a 0 with probability a^2 and 1 with b^2.

Now consider a quantum computer with two qbits. There are four possible states: $|00 >$, $|01 >$, $|10 >$ and $|11 >$, and its superposition is $a|00>+b|01>+c|10>+d|11>$, where a^2, b^2, c^2 and d^2 are the probabilities of finding two qbits in any of the four states. In a quantum computer, the two bits are in all possible states at one time. So it is possible to add a number to the two bits, which means we can add the number to 00,01,10,11 and compute the result at the same time. This ability to operate on all states at one time makes it so powerful.

Here the number of parallel operations depends on the number of qbits used. If N number of qbits are used, then 2^N operations can be done in parallel, and this inherent parallelism makes quantum computers so fast. But the question is, how do you encode a photon as a qbit? We know a photon has its own spin in all possible directions. As in certain

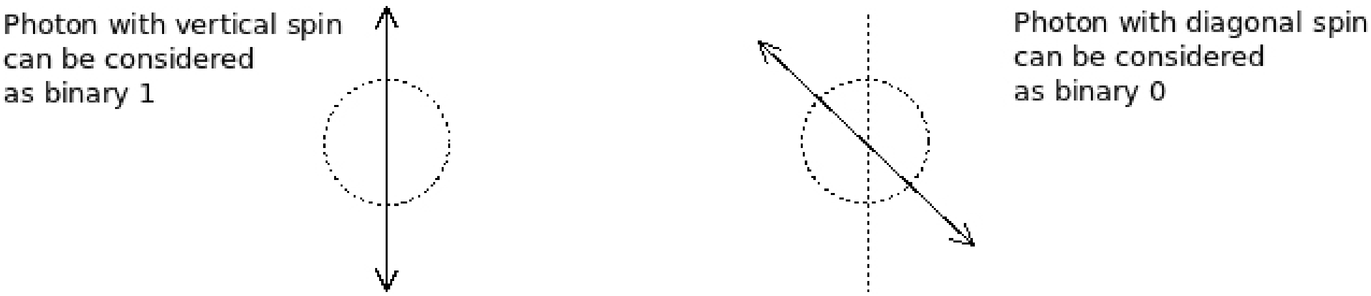


Figure 4. Encoding Polarized Photons as Binary Values

digital systems, we consider +5 volts as 1 and 0 volts as 0, and we can use the spin property of a photon to encode a photon as a qbit. We can use the photon's spin in a particular direction as 1 and the spin in the other direction as 0—say, a photon with vertical spin will be considered as 1 and a photon with an angular spin as 0.

Quantum Cryptography

Before starting to describe what quantum cryptography is, let me introduce three names I use throughout this article: Alice, Bob and Eve. Alice is sending the message, and Bob is receiving the message. Eve is in between them, trying to intercept the message. What Eve does is somehow collect the secret key to the message and decrypts it. Now, if Alice somehow can send the key of the message to Bob without any interception, she can send the message without problems.

Now, let me discuss the BB84 protocol. It is based on the name of the inventors Charles Bennet and Gilles Brassard, and it was invented in 1984. Quantum cryptography follows two steps. The first one is sending the secret key, and the second step is sending the message. Here, Alice and Bob make use of two fundamentally different communication channels:

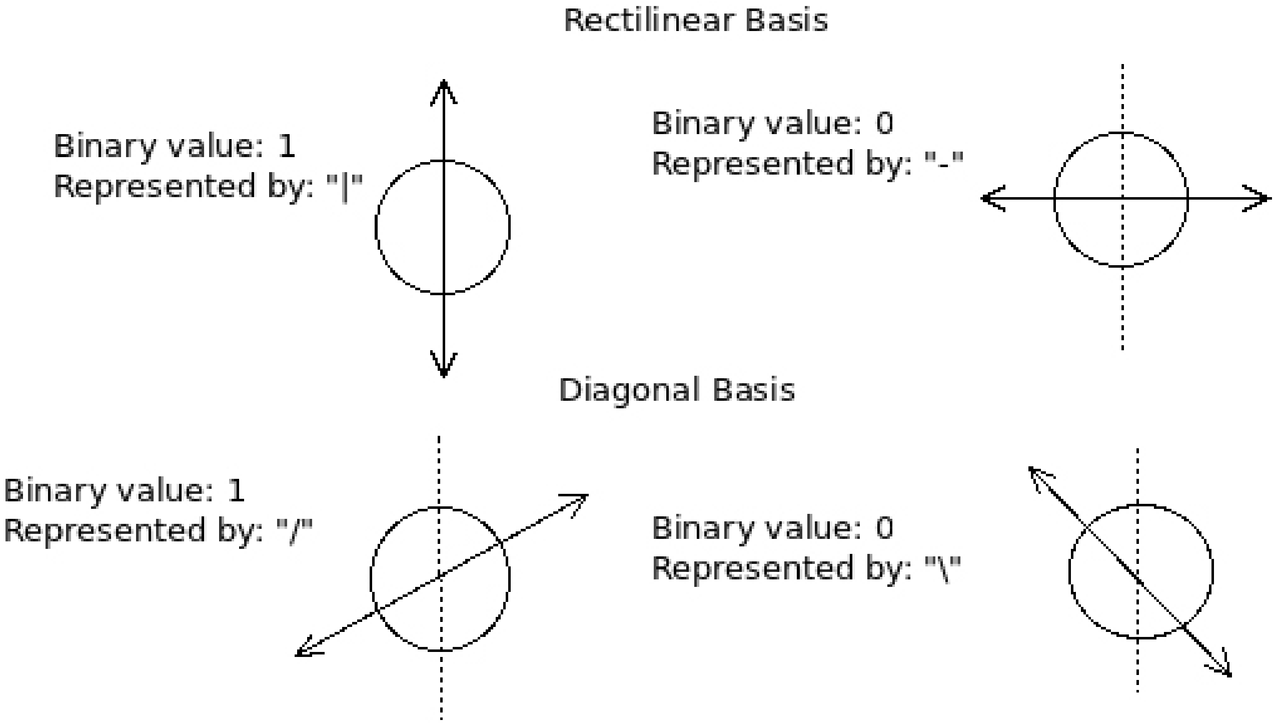


Figure 5. Binary Encoding of Photons in My Examples

a classical channel and a quantum channel. A classical channel is something that you use on the Internet to transfer data. In a classical channel, Eve can observe the bit-stream without affecting the data. But, a quantum channel is something different. It is capable of sending information in terms of quantum, and Eve can't observe the data without affecting the data. In the BB84 protocol, the secret key is sent through the quantum channel, but the message is sent through the ordinary channel but encrypted by the secret key. The first step is called Quantum Key Distribution (QKD). In this step, Alice and Bob use the quantum channel for communication.

IF HE USES THE SAME BASIS, HE WILL GET THE EXACT BIT THAT ALICE SENT; OTHERWISE, THERE IS A 50% CHANCE THAT HE WILL GET A WRONG BIT.

First, let's imagine there is no Eve between Alice and Bob. Let's assume that Alice is using two types of polarizer: one is a diagonal polarizer (X) and one a rectilinear polarizer (+). In a rectilinear basis, a photon with a spin "|" (that is, up to down) is considered as 1, and a "-" (that is, left to right) is 0. In a diagonal basis, a photon with a spin "/" is considered as 1, and "\" is 0. The diagram shown in Figure 5 should help you understand how I'm representing photons as binary values.

Now Alice has a key, and for each bit, she will select a random basis (either diagonal or rectilinear) to encode the bit to send. Nobody, not even Bob, knows what basis Alice is using. Bob will receive the encoded qbits, and Bob will use random basis to decode the qbits. If he uses the same basis, he will get the exact bit that Alice sent; otherwise, there is a 50% chance that he will get a wrong bit. For example, if Alice uses a diagonal basis to encode 1, and Bob also uses diagonal basis to decode that, then he will get a 1. If he uses a rectilinear basis, then there is a 50%

Table 1. Alice Sending the Secret Key 100101

	ALICE	BOB
Basis used	+,X,+,+,X,X	+,+,+,X,+,X

chance that he will get a 1 and a 50% chance of getting 0. As Bob is also using random basis, there's a 50% chance that he will use the right basis (that is, he will use the basis that Alice used) and will decode 50% of qbits exactly, and for the 50% wrong basis, he will decode 25% of qbits exactly, and that means Bob will decode 75% of qbits exactly.

Alice and Bob will exchange the basis they used for each bit using the normal channel without revealing their bits. They can check for which bits they both used the same basis, and those bits will be used as the secret key. Consider the example shown in Table 1 where Alice is sending the secret key 100101.

In this case, Bob will decode the key as 1,0/1,0,0/1,0/1,1. Because Bob has used some wrong basis to measure the qbits, he may get a 0 or 1 randomly on those cases. Then, they will exchange their basis with others, and they will find that in positions 2, 4 and 5, Bob used the wrong basis. So they will use the rest of the bit (1st, 3rd and 6th bit) string as the secret key—that is, 101. The rest is simple, just encrypt the message using that key and send it.

The situation becomes critical when Eve comes into action. As they are connecting using the public channel, it is quite possible that Eve will intercept the communication. In this case, as with the previous case, Alice encodes the bit information using any basis and sends it to Bob, but now Eve intercepts the qbits. Like Bob, Eve also has a decoder of the qbit. But Eve also doesn't know the basis Alice is using, so like Bob, she also randomly uses basis to decode the qbits. There is a 50% chance that Eve will use the right basis, and a 50% chance she will use the wrong basis. For the correct 50%, the photon's spin direction will not be affected, but for the wrong 50%, the photon's spin direction will be changed. For the 50% of qbits for which Eve used the right basis, Bob will use a 25% right basis and 25% wrong basis, and for the right 25% of qbits, he will get a 25% right qbit, and for the wrong 25% basis Bob used, he will get 12.5% of qbits correct just due to probability. That means from the first 50% for which Eve used the right basis, Bob will get 37.5% correct qbits. For the rest of the 50%, again Bob will use 25% right and 25% wrong basis. From this, Bob will get 12.5% and 12.5% due to probability, which means he will get 25% right qbits. So when Eve is between them, Bob will have 37.5 + 25 = 62.5% accuracy. Figure 6 demonstrates this calculation.

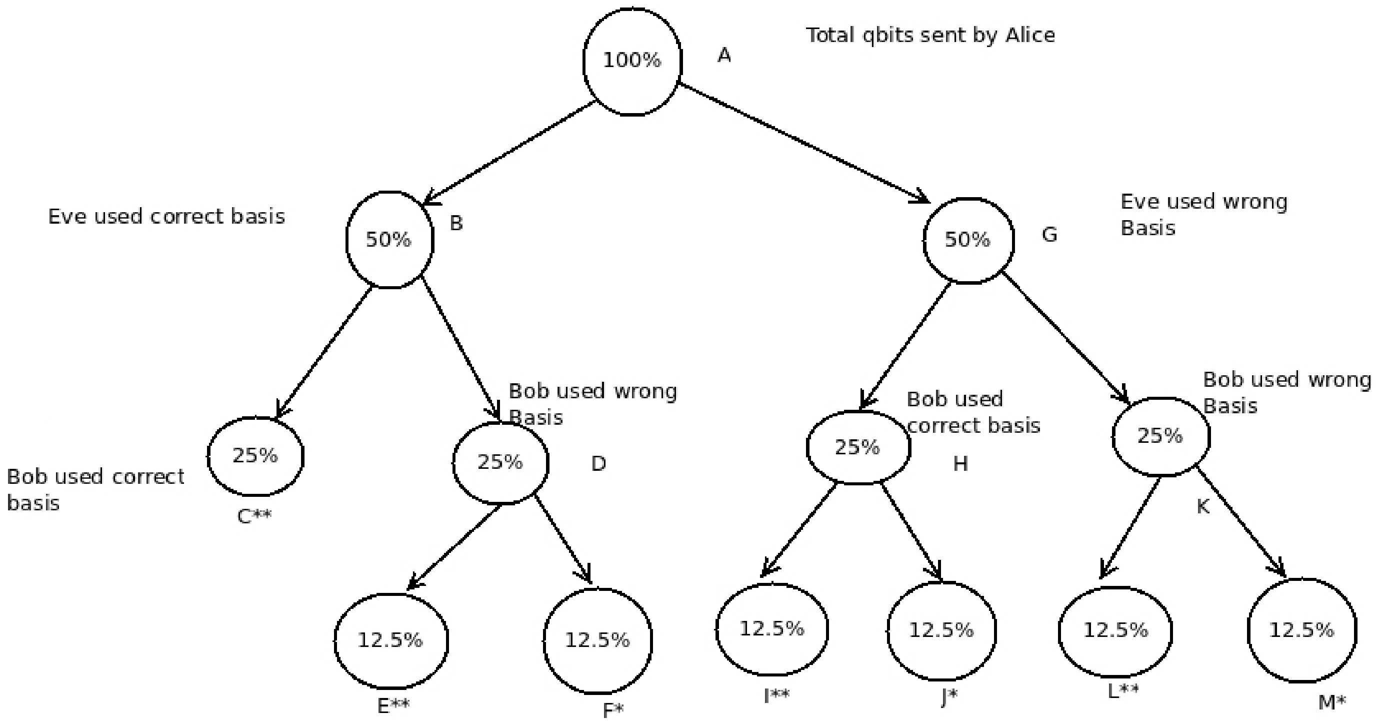


Figure 6. Accuracy Calculation for Bob When Eve Is Intercepting

In Figure 6, the node with "**", like C**, represents the nodes where Bob decoded the qbits correctly, and the node with "*", like F*, represents the nodes where Bob decoded the qbits incorrectly. One question that may arise is why does Bob get 12.5% accuracy (in E,L) when he used the wrong basis? Remember that when you use a wrong basis to decode a qbit, there is a 50% chance that you will get a 0, and a 50% chance that you will get a 1. By this logic, Bob will have 12.5% accuracy from D. Similarly, in the case of I, when Bob has used the correct basis (with respect to Alice's basis) but Eve already has changed the polarization of the qbits using the wrong basis, Bob has a 50% chance of being right and a 50% chance of being wrong.

So overall, Bob gets 12.5% right qbits in I and 12.5% wrong qbits in J. Now they will match the basis they used for each qbit, and they will use the bits where Bob used the correct basis, and they will throw out the bits for which Bob used the wrong basis. Now they need to check whether Eve is listening. For that purpose, they will use a subset of the matched key (after throwing out the bits for which Bob used wrong basis) and compare with others using the normal channel. Bob will have 100% accuracy if Eve

Table 2. Alice Sending a Key of 01101011 to Bob Using Two Types of Polarization

Alice's basis	+	X	+	+	X	X	X	X
Alice's data	0	1	1	0	1	0	1	1
Eve's basis	+	+	X	+	X	X	X	+
Eve's data	0	1	0	0	1	1	1	0
Bob's basis	+	+	+	X	+	X	X	X
Bob's data	0	0	0	0	0	1	1	1

is not there; otherwise, Bob will have 75% accuracy in the basis comparison. If the accuracy is 100%, they will discard the set of bits they used for matching, and the rest of the bit string will be used as the key to encrypt the message. If 100% accuracy is not observed, they will try again to get a key using QKD.

In Table 2, Alice is sending a key of "01101011" to Bob using two types of polarization as stated above.

Now Alice and Bob will compare their basis, and they will find that Bob has guessed the 1st, 3rd, 7th and 8th basis correctly. So they will throw out the bits for the remaining positions—that is, the 2nd, 4th, 5th and 6th. Now the key is "0011". They will choose the first two bits for matching, and then they will find that their second bit in the key is different, which means Eve is between them. Then they will repeat the same procedure again until they get a 100% key match. When they get a key, they easily can encrypt the message using the key and send it via the public network.

Limitations

In practice, the quantum channel also will be affected by noise, and it will be hard to distinguish between noise and eavesdropping.

If Eve wants, she can intercept the quantum channel just to not allow Alice and Bob to communicate.

No amplifiers are used on the optical fiber carrying the quantum signal. Such devices would disrupt the communication in the same way an eavesdropper does. This implies, in turn, that QKD's range is limited.

Following the no-cloning theorem, QKD can provide only a 1:1 connection. So the number of links will increase N(N – 1)/2, as N represents the number of nodes.

Research

Researchers have been developing such systems for more than a decade. The DARPA Quantum Network, which became fully operational in BBN's laboratory in October 2003, has been continuously running in six nodes, operating through the telecommunications fiber between Harvard University, Boston University and BBN since June 2004. The DARPA Quantum Network is the world's first quantum cryptography network, and perhaps also the first QKD system providing continuous operation across a metropolitan area (**http://arxiv.org/abs/quant-ph/0503058**).

NIST performs core research on the creation, transmission, processing and measurement of optical qbits. It demonstrated high-speed QKD systems that generate secure keys for encryption and decryption of information using a one-time pad cipher, and extended them into a three-node quantum communications network (**http://w3.antd.nist.gov/qin/index.shtml**).

Toshiba's Quantum Key Distribution System delivers digital keys for cryptographic applications on fiber-optic-based computer networks based on quantum cryptography. In particular, it allows key distribution over standard telecom fiber links exceeding 100km in length and bit rates sufficient to generate 1 megabit per second of key material over a distance of 50km—sufficiently long for metropolitan coverage (**https://www.toshiba-europe.com/research/crl/qig/quantumkeyserver.html**).

The current status of quantum cryptography in Japan includes an inter-city QKD testbed based on DPS-QKD, a field test of a one-way BB84 system over 97km with noise-free WDM clock synchronization, and so on ("Toward New Generation Quantum Cryptography—Japanese Strategy" by Nukuikita, Koganei).

The 973 Program and 863 program of China have funded support to the QKD research (Post-Quantum Cryptography: Third International Workshop, Pqcrypto 2010, Darmstadt, Germany, May 25–28, 2010, Proceedings, 1st ed.).

In Europe, the SEcure COmmunication based on Quantum Cryptography (SECOQC, 2004–2008) project was funded for the same reason (http://vcq.quantum.at/publications/all-publications/details/643.html).

In 2004, ID Quantique was the first in the world to bring a quantum key distribution system to a commercial market. ID Quantique's QKD product was used in conjunction with layer 2 Ethernet encryption to secure elections in Geneva. Other companies, like MagicQ, QinetiQ and NEC, also are working in this field. Companies claim to offer or to be developing QKD products, but limited information is publicly available. However, it's likely that the situation will evolve in the near future (http://swissquantum.idquantique.com/?-Quantum-Cryptography-#).■

Subhendu Bera is from West Bengal (India). He completed his Master of Science degree in Computer Science from Banaras Hindu University and his Bachelor of Science degree in Computer Science from University of Calcutta. Currently, he is preparing for entrance for a PhD. He likes to play with machine learning tools, and in his spare time, he reads, blogs and plays cricket and chess.

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

Resources

W. Chen, H.-W. Li, S. Wang, Z.-Q. Yin, Z. Zhou, Y.-H. Li, Z.-F. Han and G.C. Guo (2012). "Quantum Cryptography", *Applied Cryptography and Network Security*, Dr. Jaydip Sen (Ed.), ISBN: 978-953-51-0218-2, InTech, available from **http://www.intechopen.com/books/applied-cryptography-and-network-security/quantum-cryptography**

"Quantum Cryptography Hits the Fast Lane" by Adrian Cho: **http://news.sciencemag.org/sciencenow/2010/04/quantum-cryptography-hits-the-fa.html**

"Do we need quantum cryptography?" by Peter Rohde: **http://www.peterrohde.org/2012/06/29/do-we-need-quantum-cryptography**

"A Little (q)bit of Quantum Computing" by Douglas Eadline: **http://www.linux-mag.com/id/8753**

"What is a quantum computer?" by Dr Boaz Tamir: **http://thefutureofthings.com/column/5/what-is-a-quantum-computer.html**

Quantum Computation and Quantum Information by Michael A. Nielsen and Isaac L. Chuang, Cambridge University Press, 2011.

"Quantum Communication": **http://w3.antd.nist.gov/qin/index.shtml**

More Secure SSH Connections

Thwart would-be attackers by hardening your SSH connections.

FEDERICO KEREKI

If you need remote access to a machine, you'll probably use SSH, and for a good reason. The secure shell protocol uses modern cryptography methods to provide privacy and confidentiality, even over an unsecured, unsafe network, such as the Internet. However, its very availability also makes it an appealing target for attackers, so you should consider hardening its standard setup to provide more resilient, difficult-to-break-into connections. In this article, I cover several methods to provide such extra protections, starting with simple configuration changes, then limiting access with PAM and finishing with restricted, public key certificates for passwordless restricted logins.

Where Is SSH?

As defined in the standard, SSH uses port 22 by default. This implies that with the standard SSH configuration, your machine already has a nice target to attack. The first method to consider is quite simple—just change the port to an unused, nonstandard port, such as 22022. (Numbers above 1024 are usually free and safe, but check the Resources at the end of this article just to avoid possible clashes.) This change won't affect your remote users much. They will just need to add an extra parameter to their connection, as in `ssh -p 22022 the.url.for.your.server`. And yes, this kind of change lies fully in what's called "security through obscurity"—doing things obscurely, hoping that no one will get wise to your methods—which usually is just asking for problems. However, it will help at least against script kiddies, whose scripts just try to get in via port 22 instead of being thorough enough to try to scan your machine

Knock for SSH

Trying to attack your machine will be harder if the would-be invader cannot even find a possible SSH door. The methods shown in this article are compatible with the port-knocking technique I wrote about in a previous article ("Implement Port-Knocking Security with knockd", January 2010), so I won't go into `knockd` configuration here. By using all techniques together, attackers will have an even harder time getting to your machine (where all the other measures shown in this article will be waiting), because they won't even be able to start trying to attack your box.

for all open ports.

In order to implement this change, you need to change the /etc/ssh/sshd_config file. Working as root, open it with an editor, look for a line that reads "Port 22", and change the 22 to whatever number you chose. If the line starts with a hash sign (#), then remove it, because otherwise the line will be considered a comment. Save the file, and then restart SSH with `/etc/init.d/sshd restart`. With some distributions, that could be `/etc/rc.d/init.d/sshd restart` instead. Finally, also remember to close port 22 in your firewall and to open the chosen port so remote users will be able to access your server.

While you are at this, for an extra bit of security, you also could add or edit some other lines in the SSH configuration file (Listing 1). The `Protocol` line avoids a weaker, older version of the SSH protocol. The `LoginGraceTime` gives the user 30 seconds to accomplish a login. The `MaxAuthTries` limits users to three wrong attempts at entering the password before they are rejected. And finally, `PermitRootLogin` forbids a user from logging in remotely as root (any attacker who managed to get into your machine still would have to be able to break into the root account; an extra hurdle), so would-be attackers will have a harder time at getting privileges on your machine.

Listing 1. These little SSH configuration changes can add a bit of security

```
Port             22022
Protocol             2
LoginGraceTime      30
MaxAuthTries         3
PermitRootLogin     no
```

Be sure to restart the SSH service dæmon after these changes (`sudo /etc/init.d/sshd restart` does it), and for now, you already have managed to add a bit of extra safety (but not much really), so let's get down to adding more restrictions.

Who Can Use SSH?

Your machine may have several servers, but you might want to limit remote access to only a few. You can tweak the sshd_config file a bit more, and use the `AllowUsers`, `DenyUsers`, `AllowGroups` and `DenyGroups` parameters. The first one, `AllowUsers`, can be followed by a list of user names (or even patterns, using the common * and ? wild cards) or user@host pairs, further restricting access to the user only from the given host. Similarly, `AllowGroups` provides a list of group name patterns, and login is allowed only for members

> **From a software engineering viewpoint, it would just be awful if each and every program had to invent and define and implement its own authentication logic.**

of those groups. Finally, `DenyUsers` and `DenyGroups` work likewise, but prohibit access to specific users and groups. Note: the priority order for rules is `DenyUsers` first, then `AllowUsers`, `DenyGroups` and finally `AllowGroups`, so if you explicitly disallow users from connecting with `DenyUsers`, no other rules will allow them to connect.

For example, a common rule is that from the internal network, everybody should be able to access the machine. (This sounds reasonable; attacks usually come from outside the network.) Then, you could say that only two users, fkereki and eguerrero, should be able to connect from the outside, and nobody else should be able to connect. You can enable these restrictions by adding a single line `AllowUsers *:192.168.1.*,fkereki,eguerrero` to the SSH configuration file and restarting the service. If you wanted to forbid jandrews from remote connections, an extra `DenyUsers jandrews` would be needed. More specific rules could be added (say, maybe eguerrero should be able to log in only from home), but if things start getting out of hand with too many rules, the idea of editing the ssh configuration files and restarting the server begins to look less attractive, and there's a better solution through PAM, which uses separate files for security rules.

The PAM Way

If you google for meanings of PAM, you can find several definitions, ranging from a cooking oil spray to several acronyms (such as Power Amplitude Modulation or Positive Active Mass), but in this case, you are interested in Pluggable Authentication Modules, a way to provide extra authentication rules and harden access to your server. Let's use PAM as an alternative solution to specify which users can access your server.

From a software engineering viewpoint, it would just be awful if each and every program had to invent and define and implement its

PAM, PAM Everywhere

Although there is no “official” list of PAMs, most distributions are likely to include the following:

- pam_access: allows or denies access according to the file /etc/security/access.conf.
- pam_cracklib: checks passwords against dictionaries.
- pam_debug: used for testing only.
- pam_deny: always denies access.
- pam_echo: displays the contents of a file.
- pam_env: sets or unsets environment variables.
- pam_exec: lets you run an external command.
- pam_group: grants group memberships to the user.
- pam_lastlog: shows the date and time of the user’s last log in.
- pam_ldap: allows authentication against an LDAP server.
- pam_limits: lets you set system resource limits, through the file /etc/security/limits.conf.
- pam_listfile: an alternative to pam_access, with some extra options.
- pam_mail: checks if the user has pending mail.
- pam_make: runs `make` in a given directory.
- pam_motd: displays the “message of the day” file, usually /etc/motd.
- pam_nologin: blocks all logins should file /etc/nologin exist.
- pam_permit: always allows access.
- pam_pwcheck: checks passwords for strength.
- pam_pwhistory: checks new passwords against recently used ones to avoid repetition.
- pam_rootok: usually is included in /etc/pam.d/su as a “sufficient” test so root can act as any other user without providing a password.
- pam_selinux: sets the default security context for SELinux.
- pam_sepermit: allows or denies login depending on SELinux state.
- pam_shells: allows access only if the user’s shell is listed in the file /etc/shells.
- pam_succeed_if: checks for account characteristics, such as belonging to a given group.
- pam_tally: just keeps count of attempted accesses and can deny access if too many attempts fail.
- pam_time: restricts access based on rules in the file /etc/security/time.conf.
- pam_umask: lets you set the file mode creation mask (think `umask`) for newly created files.
- pam_unix (or pam_unix2): provides classical UNIX-style authentication per the /etc/passwd and /etc/shadow files.
- pam_userdb: authenticates the user against a Berkeley database.
- pam_warn: records logs in the system logs.
- pam_wheel: provides root access only to members of group wheel.

File locations vary, but you can check /usr/lib/security or /lib/security (or read lib64 for lib, for 64-bit Linux) to see what modules you actually have. For more information on each module, try `man name.of.the.module`, but don’t try to execute them from the command line, for they can’t be run that way.

own authentication logic. How could you be certain that all applications did implement the very same checks, in the same way, without any differences? PAM provides a way out; if a program needs to, say, authenticate a user, it can call the PAM routines, which will run all the checks you might have specified in its configuration files. With PAM, you even can change authentication rules on the fly by merely updating its configuration. And, even if that's not your main interest here, if you were to include new biometrics security hardware (such as fingerprint readers, iris scanners or face recognition) with an appropriate PAM, your device instantly would be available to all applications.

PAMs can be used for four security concerns: account limitations (what the users are allowed to do), authorization (how the users identify themselves), passwords and sessions. PAM checks can be marked optional (may succeed or fail), required (must succeed), requisite (must succeed, and if it doesn't, stop immediately without trying any more checks) and sufficient (if it succeeds, don't run any more checks), so you can vary your policies. I don't cover all these details here, but rather move on to the specific need of specifying who can (or cannot) log in to your server. See the PAM, PAM Everywhere sidebar for a list of some available modules.

PAM configurations are stored in /etc/pam.d, with a file for each command to which they apply. As root, edit /etc/pam.d/sshd, and add an `account required pam_access.so` line after all the `account` lines, so it ends up looking like Listing 2. (Your specific version of the file may have some different options; just add the single line to it, and that's it.) You'll also have to modify the sshd configuration file (the same one that you modified earlier) so it uses PAM; add a `UsePAM yes` line to it, and restart the sshd dæmon.

The `account` part is what is important here. After using the standard UNIX methods for checking your password (usually against the files /etc/passwd and /etc/shadow), it uses the module `pam_access.so` to check if the user is in a list, such as shown in Listing 3. Both `account` modules are `required`, meaning that the user must pass both checks in order to proceed. For extra restrictions, you might want to look at `pam_listfile`, which is similar to `pam_access` but provides even more options, and `pam_time`, which lets you fix time restrictions. You also would need to add extra `account`

Listing 2. Adding `pam_access.so` to the account PAM checks lets you specify which users have SSH access to your machine.

```
account  required    pam_unix2.so
account  required    pam_access.so

auth     required    pam_env.so
auth     required    pam_unix2.so
auth     required    pam_nologin.so

password requisite   pam_pwcheck.so nullok cracklib
password required    pam_unix2.so use_authtok nullok

session  required    pam_limits.so
session  required    pam_unix2.so
session  optional    pam_umask.so
```

lines to the /etc/pam.d/sshd file.

You need to edit /etc/security/access.conf to specify which users can access the machine (Listing 3). Each line in the list starts with either a plus sign (login allowed) or a minus sign (login disabled), followed by a colon, a user name (or ALL), another colon and a host (or ALL). The `pam_access.so` module goes down the list in order, and depending on the first match for the user, it either allows or forbids the connection. The order of the rules is important. First, jandrews is forbidden access, then everybody in the internal network is allowed to log in to the server. Then, users fkereki and eguerrero are allowed access from any machine. The final `-:ALL:ALL` line is a catchall that denies access to anybody not specifically allowed to log in in the previous lines, and it always should be present.

Note that you could use this configuration for other programs

Listing 3. The file /etc/security/access.conf specifies which users have access and from which hosts.

```
-:jandrews:ALL
+:ALL:192.168.1.
+:fkereki:ALL
+:eguerrero:ALL
-:ALL:ALL
```

and services (FTP, maybe?), and the same rules could be applied. That's an advantage of PAM. A second advantage is that you can change rules on the fly, without having to restart the SSH service. Not messing with running services is always a good idea! Using PAM adds a bit of hardening to SSH to restrict who can log in. Now, let's look at an even safer way of saying who can access your machine by using certificates.

Passwordless Connections

Passwords can be reasonably secure, but you don't have them written down on a Post-It by your computer, do you? However, if you use a not-too-complex

Listing 4. Generating a public/private key pair with `ssh-keygen` is simple. Opt for using a passphrase for extra security.

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/fkereki/.ssh/id_rsa):
Created directory '/home/fkereki/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/fkereki/.ssh/id_rsa.
Your public key has been saved in /home/fkereki/.ssh/id_rsa.pub.
The key fingerprint is:
84:13:e6:07:a3:b1:b4:c6:9f:29:b8:40:58:f5:23:26 fkereki@fedoraxfce
The key's randomart image is:
+--[ RSA 2048]----+
|  ..+ =          |
|.. o O =         |
|..E O * o        |
|.  = o B         |
|. . . + S        |
| . . .           |
|  .              |
|                 |
|                 |
+-----------------+
```

password (so it can be determined by brute force or a dictionary attack), then your site will be compromised for so long as the attacker wishes. There's a safer way, by using public/private key logins, that has the extra advantage of requiring no passwords on the remote site. Rather, you'll have a part of the key (the "private" part) on your remote machine and the other part (the "public" part) on the remote server. Others won't be able to impersonate you unless they have your private key, and it's computationally unfeasible to calculate. Without going into how the key pair is created, let's move on to using it.

First, make sure your sshd configuration file allows for private key logins. You should have `RSAAuthentication yes` and `PubkeyAuthentication yes` lines in it. (If not, add them, and restart the service as described above.) Without those lines, nothing I explain below will work. Then, use `ssh-keygen` to create a public/private key pair. By directly using it without any more parameters (Listing 4), you'll be asked in which file to save the key (accept the standard), whether to use a passphrase for extra security (more on this below, but you'd better do so), and the key pair will be generated. Pay attention to the name of the file in which the key was saved. You'll need it in a moment.

Now, in order to be able to connect to the remote server, you need to copy it over. If you search the Internet, many sites recommend directly editing certain files in order to accomplish this, but using `ssh-copy-id` is far easier. You just have to type

Listing 5. After generating your public/private pair, you need to use `ssh-copy-id` to copy the public part to the remote server.

```
$ ssh-copy-id -i /home/fkereki/.ssh/id_rsa.pub fkereki@192.168.1.107
The authenticity of host '192.168.1.107 (192.168.1.107)'
 ➥can't be established.
RSA key fingerprint is 16:a4:d8:6a:ee:e0:8d:f4:72:a8:af:42:75:1d:28:3b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.107' (RSA) to the list
 ➥of known hosts.
fkereki@192.168.1.107's password:
```

ssh-copy-id -i the.file.where.the.key.was.saved remote.user@remote.host specifying the name of the file in which the public key was saved (as you saw above) and the remote user and host to which you will be connecting (Listing 5). And you're done.

In order to test your new passwordless connection, just do ssh remote.user@remote.host. If you used a passphrase, you'll be asked for it now. In either case, the connection will be established, and you won't need to enter your password for the remote site (Listing 6).

Now, what about the passphrase? If you create a public/private key pair without using a passphrase, anybody who gets access to your machine and the private key immediately will have access to all the remote servers to which you have access. Using the passphrase adds another level of security to your log in process. However, having to enter it over and over again is a bother. So, you would do better by using ssh-agent, which can "remember" your passphrase and enter it automatically whenever you try to log in to a remote server. After running ssh-agent, run ssh-add

Listing 6. After you've copied the public key over, you can log in to the remote server without a password. You will have to enter your passphrase though, if you used one when generating the public/private pair.

```
$ ssh fkereki@192.168.1.107
Enter passphrase for key '/home/fkereki/.ssh/id_rsa':
Last login: Mon Jan 10 18:40:11 2011

6.0 Light Final built on March 31, 2009 on Linux 2.6.27.12
You are working as fkereki
Frequently used programs:
Configuration    : vasm
File manager     : mc (press F2 for useful menu)
Editor           : mcedit, nano, vi
Multimedia       : alsamixer, play
vector:/~
$ logout
Connection to 192.168.1.107 closed.
```

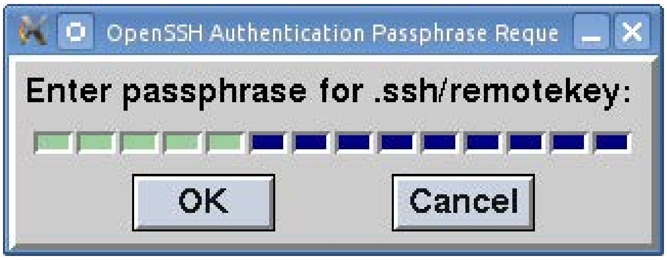
to add your passphrase. (You could run it several times if you have many passphrases.) After that, a remote connection won't need a passphrase any more (Listing 7). If you want to end a session, use `ssh-agent -k`, and you'll have to re-enter the passphrase if you want to do a remote login.

You also may want to look at `keychain`, which allows you to reuse `ssh-agent` between logins. (Not all distributions include this command; you may have to use your package manager to install it.) Just do `keychain the.path.to.your.private.key`, enter your passphrase (Figure 1), and until you reboot the server or specifically run `keychain -k all` to stop `keychain`, your passphrase will be stored, and you won't have to re-enter it. Note: you even could log out and log in again, and your key still would be available. If you just want to clear all cached keys, use `keychain --clear`.

If you use a passphrase, you could take your private keys with you on a USB stick or the like and use it from

Listing 7. Using `ssh-agent` frees you from having to re-enter your passphrase.

```
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-Rvhhx30943/agent.30943; export SSH_AUTH_SOCK;
SSH_AGENT_PID=30944; export SSH_AGENT_PID;
echo Agent pid 30944;

$ ssh-add
Enter passphrase for /home/fkereki/.ssh/id_rsa:
Identity added: /home/fkereki/.ssh/id_rsa (/home/fkereki/.ssh/id_rsa)

$ ssh fkereki@192.168.1.107
Last login: Mon Jun 10 18:44:15 2013 from 192.168.1.108
6.0 Light Final built on March 31, 2009 on Linux 2.6.27.12
You are working as fkereki
Frequently used programs:
Configuration    : vasm
File manager     : mc (press F2 for useful menu)
Editor           : mcedit, nano, vi
Multimedia       : alsamixer, play
```



Enter passphrase for .ssh/remotekey:

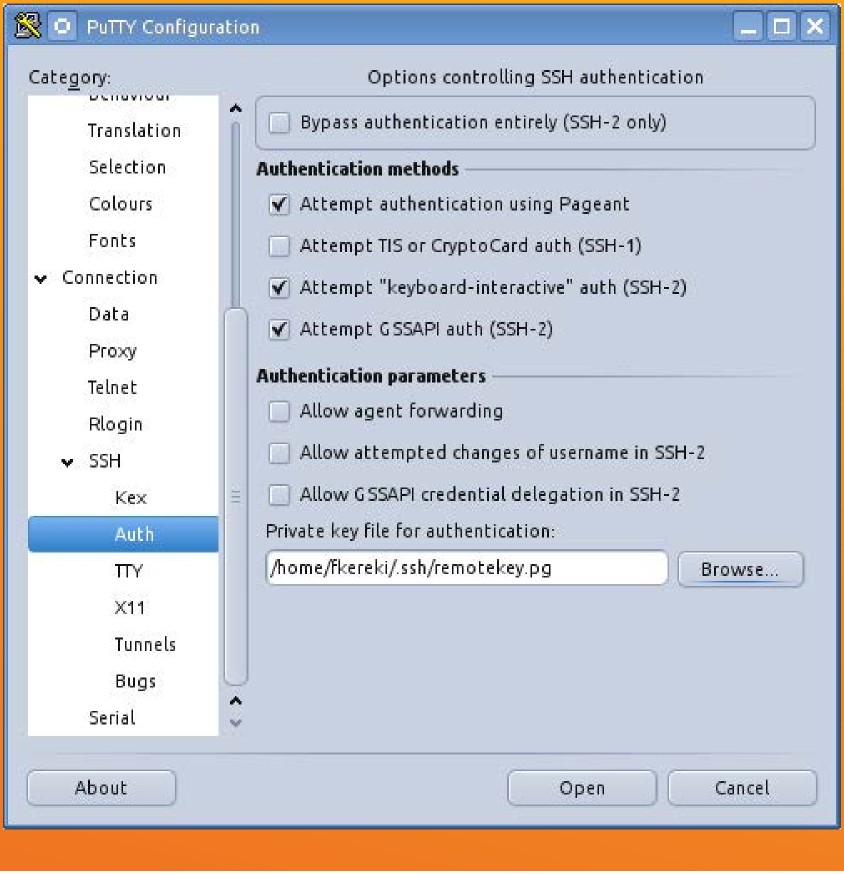
Figure 1. By entering your passphrase once with `keychain`, it will be remembered even if you log out.

any other machine in order to log in to your remote servers. Doing this without using passphrases would just be too dangerous. Losing your USB stick would mean automatically compromising all the remote servers you could log in to. Also, using a passphrase is an extra safety measure. If others got hold of your private key, they wouldn't be able to use it without first determining your passphrase.

Finally, if you are feeling quite confident that all needed users have their passwordless logins set up, you could go the whole mile and disable common passwords by editing the

Using SSH and PuTTY

You can use SSH public/private pairs with the common PuTTY program, but not directly, because it requires a specific, different key file. In order to convert your SSH key, you need to do `puttygen $HOME/.ssh/your.private.key -o your.private.key.file.for.putty`. Afterward, you simply can open PuTTY, go to Connection, SSH, Auth and browse for your newly generated "Private key file for authentication".



Options controlling SSH authentication

☐ Bypass authentication entirely (SSH-2 only)

Authentication methods

☑ Attempt authentication using Pageant

☐ Attempt TIS or CryptoCard auth (SSH-1)

☑ Attempt "keyboard-interactive" auth (SSH-2)

☑ Attempt GSSAPI auth (SSH-2)

Authentication parameters

☐ Allow agent forwarding

☐ Allow attempted changes of username in SSH-2

☐ Allow GSSAPI credential delegation in SSH-2

Private key file for authentication: /home/fkereki/.ssh/remotekey.pg sshd configuration file and setting `PasswordAuthentication no` and `UsePAM no`, but you'd better be quite sure everything's working, because otherwise you'll have problems.

Conclusion

There's no definitive set of security measures that can 100% guarantee that no attacker ever will be able to get access to your server, but adding extra layers can harden your setup and make the attacks less likely to succeed. In this article, I described several methods, involving modifying SSH configuration, using PAM for access control and public/private key cryptography for passwordless logins, all of which will enhance your security. However, even if these methods do make your server harder to attack, remember you always need to be on the lookout and set up as many obstacles for attackers as you can manage.■

Federico Kereki is a Uruguayan systems engineer with more than 20 years of experience developing systems, doing consulting work and teaching at universities. He currently is working with a good jumble of acronyms: SOA, GWT, Ajax, PHP and, of course, FLOSS! Recently, he wrote the *Essential GWT* book, in which you also can find some security concerns for Web applications. You can reach Federico at fkereki@gmail.com.

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

Resources

The SSH protocol is defined over a host of RFC (Request for Comments) documents; check **http://en.wikipedia.org/wiki/Secure_Shell#Internet_standard_documentation** for a list.

Port numbers are assigned by IANA (Internet Assigned Numbers Authority), and you can go to **http://www.iana.org/assignments/port-numbers** for a list.

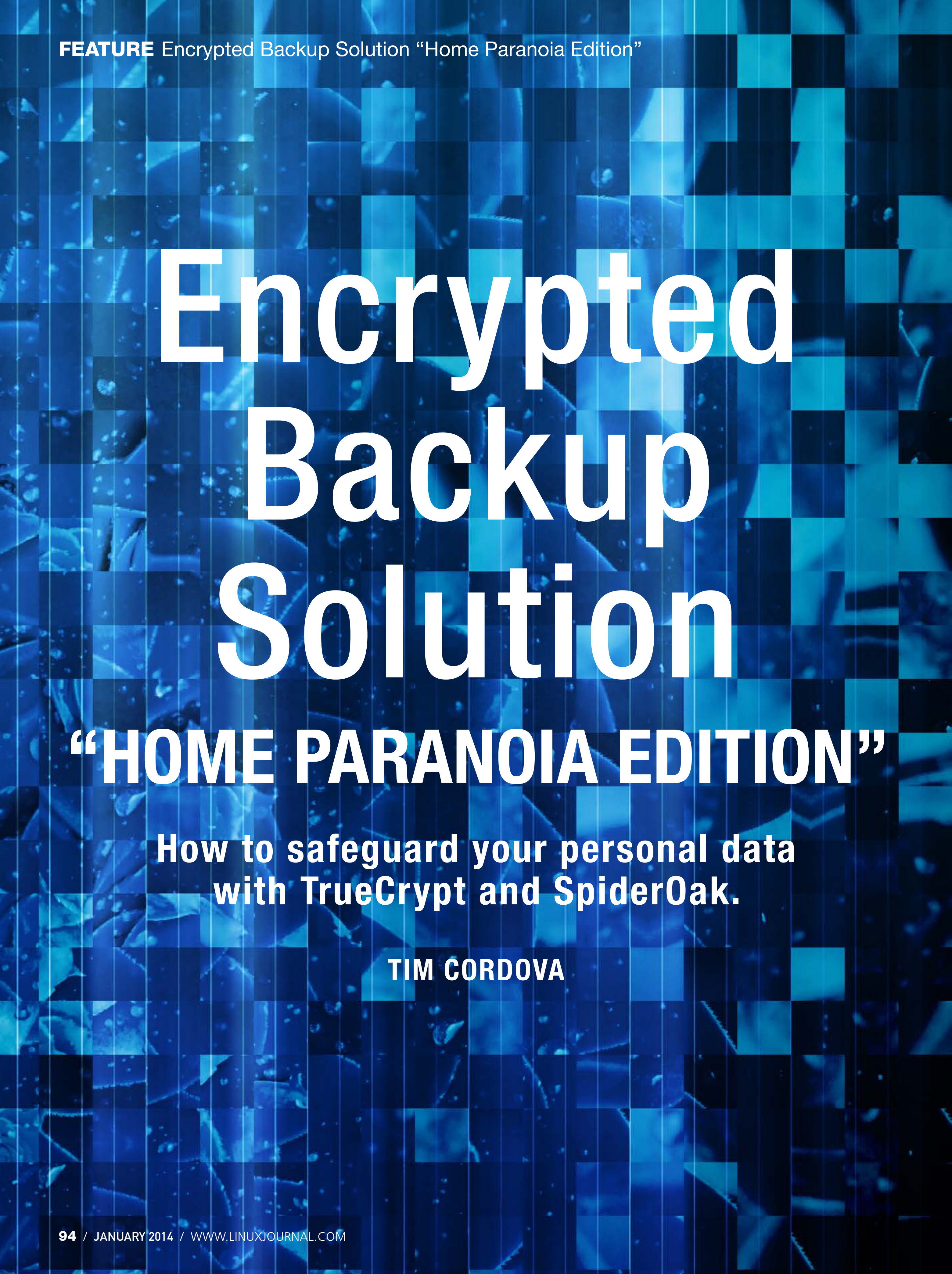
The primary distribution site for PAM is at **http://www.linux-pam.org**, and the developers' site is at **https://fedorahosted.org/linux-pam**.

Read **http://www.funtoo.org/wiki/Keychain** for more on `keychain` by its author, Daniel Robbins.

You can see the RSA original patent at **http://www.google.com/patents?vid=4405829** and the RSA Cryptography Standard at **http://www.emc.com/emc-plus/rsa-labs/pkcs/files/h11300-wp-pkcs-1v2-2-rsa-cryptography-standard.pdf**.

For extra security measures, read "Implement Port-Knocking Security with knockd", in the January 2010 issue of *Linux Journal*, or check it out on-line at **http://www.linuxjournal.com/article/10600**.

```
 *
 * 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
```



Encrypted Backup Solution

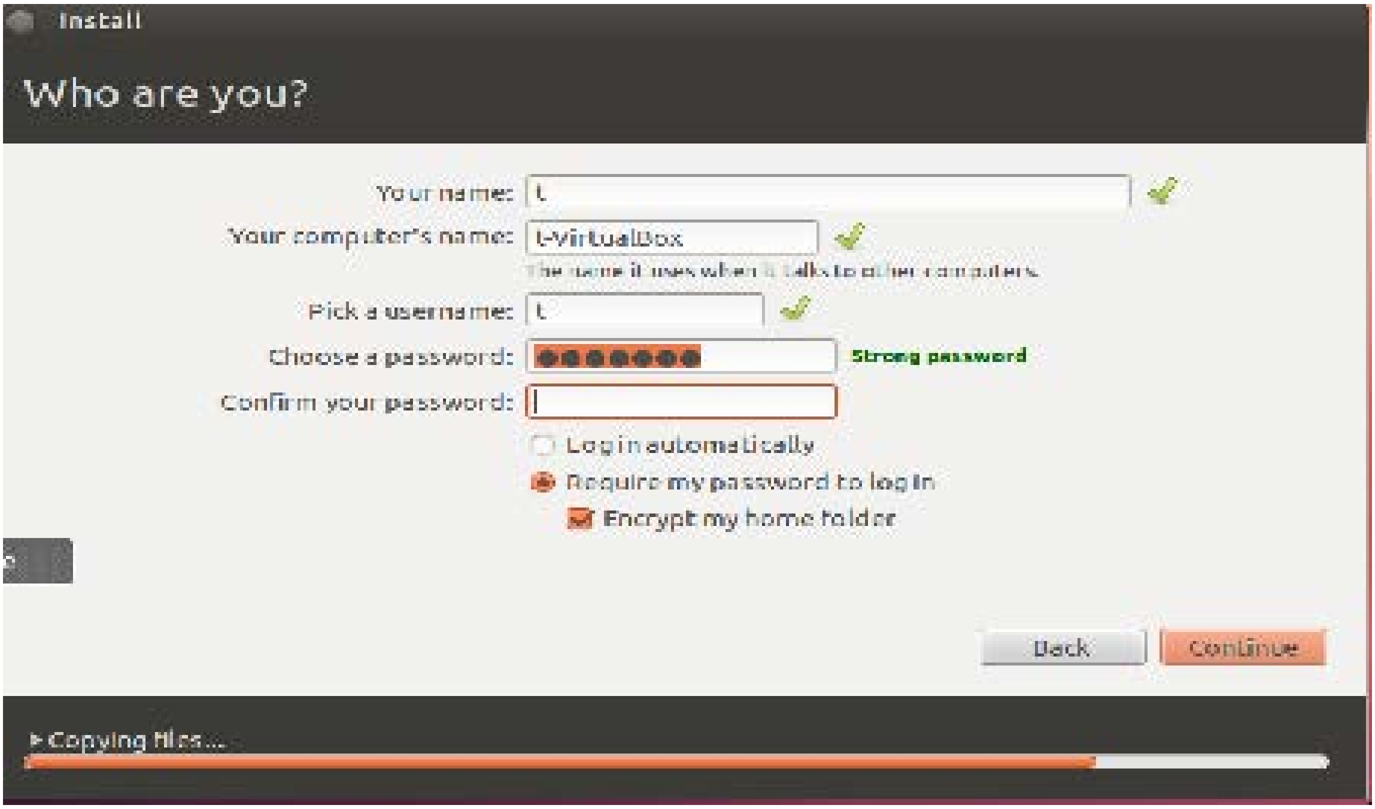
“HOME PARANOIA EDITION”

How to safeguard your personal data with TrueCrypt and SpiderOak.

TIM CORDOVA

There are so many cases of personal identifiable information (PII) or any type of data exposed on the Internet today. The details provided in this article may assist in safeguarding your tax information, social security number or password file. The setup this article describes will help keep your personal data at home safe and secure in this "cyber-security"-connected world. This includes virtual/physical security compromises—the only truly secure system is one that is unplugged and locked in a vault. This solution is not all-encompassing and does have limitations, but it is sound enough for safeguarding personal data.

The first step is addressing the physical aspect of security. This is a critical step, because some notable compromises are a direct result of someone having physical access to a system. You always should prepare yourself for the possibility that your beloved electronic devices could be in hands of someone other than you at any given moment. This situation could occur on a train, or in a coffee shop, automobile or home, and you must assume your data is lost when it is outside your control.



Install

Who are you?

Your name: t

Your computer's name: t-VirtualBox

The name it uses when it talks to other computers.

Pick a username: t

Choose a password: ●●●●●●● Strong password

Confirm your password:

○ Log in automatically

● Require my password to log in

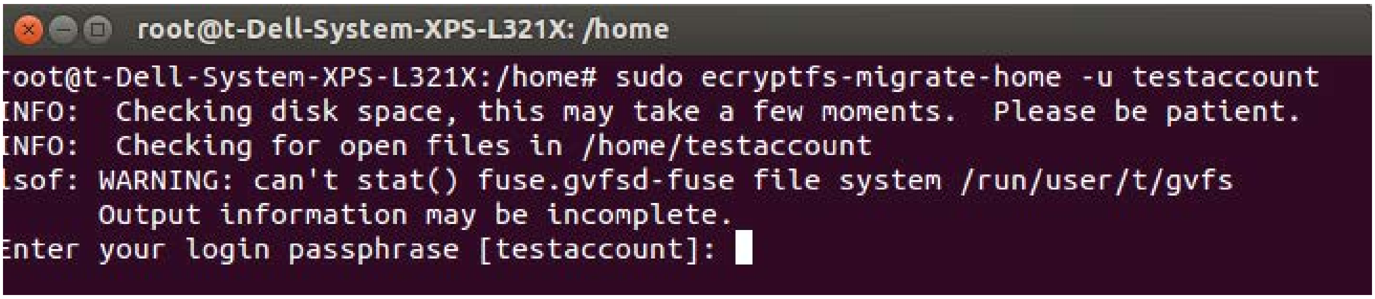
☑ Encrypt my home folder

▸ Copying files...

Figure 1. Setup screen for encrypting your home directory in Ubuntu during initial operating system installation.

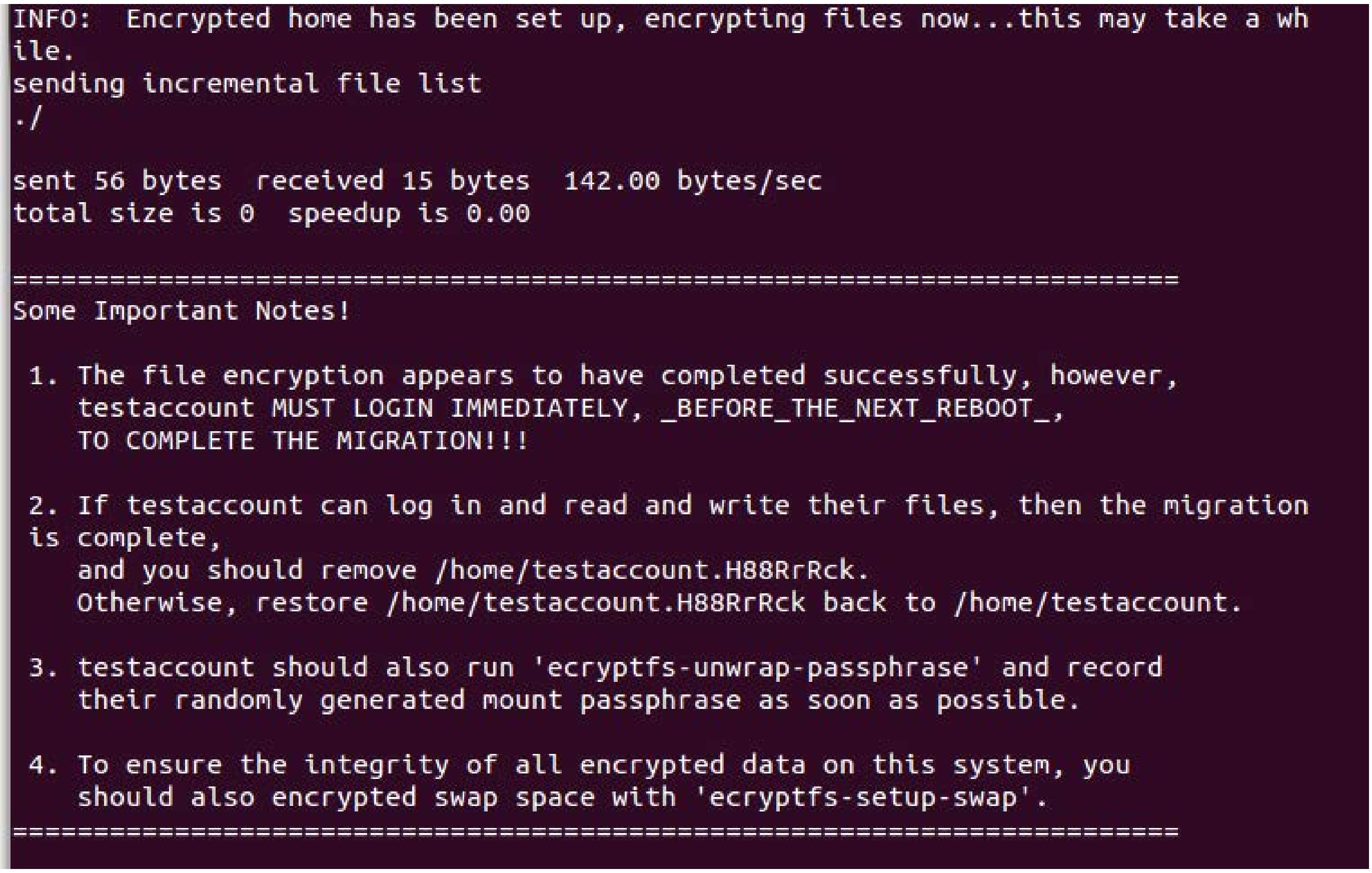
This article describes utilizing whole disk encryption to reduce some of the risks provided by a great open-source Linux operation system (Ubuntu 12.10). Whole disk encryption is a key factor, especially when considering all of the recent events concerning stolen government laptops that contained millions of social security numbers.

The next key step in safeguarding



```
root@t-Dell-System-XPS-L321X: /home
oot@t-Dell-System-XPS-L321X:/home# sudo ecryptfs-migrate-home -u testaccount
INFO:  Checking disk space, this may take a few moments.  Please be patient.
INFO:  Checking for open files in /home/testaccount
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/t/gvfs
     Output information may be incomplete.
Enter your login passphrase [testaccount]: 
```

Figure 2. If encrypting your home folder was missed during initial installation, use `encryptft-utils` to encrypt your home directory.



```
INFO:  Encrypted home has been set up, encrypting files now...this may take a wh
ile.
sending incremental file list
./

sent 56 bytes  received 15 bytes  142.00 bytes/sec
total size is 0  speedup is 0.00

========================================================================
Some Important Notes!

 1. The file encryption appears to have completed successfully, however,
    testaccount MUST LOGIN IMMEDIATELY, _BEFORE_THE_NEXT_REBOOT_,
    TO COMPLETE THE MIGRATION!!!

 2. If testaccount can log in and read and write their files, then the migration
 is complete,
    and you should remove /home/testaccount.H88RrRck.
    Otherwise, restore /home/testaccount.H88RrRck back to /home/testaccount.

 3. testaccount should also run 'ecryptfs-unwrap-passphrase' and record
    their randomly generated mount passphrase as soon as possible.

 4. To ensure the integrity of all encrypted data on this system, you
    should also encrypted swap space with 'ecryptfs-setup-swap'.
========================================================================
```

Figure 3. This is important feedback information “record passphrase as soon as possible” that will be generated from the `encryptfs-migrate-home` command.

your personal information is by adding another security layer by encrypting home directories during the initial installation (Figure 1). You may be the only one using this system; however, if others are able to access your system while it's running, this may slow them down from trying to access information contained in a home directory.

You will need to run the command:

```
sudo apt-get install ecryptfs-utils cryptsetup
```

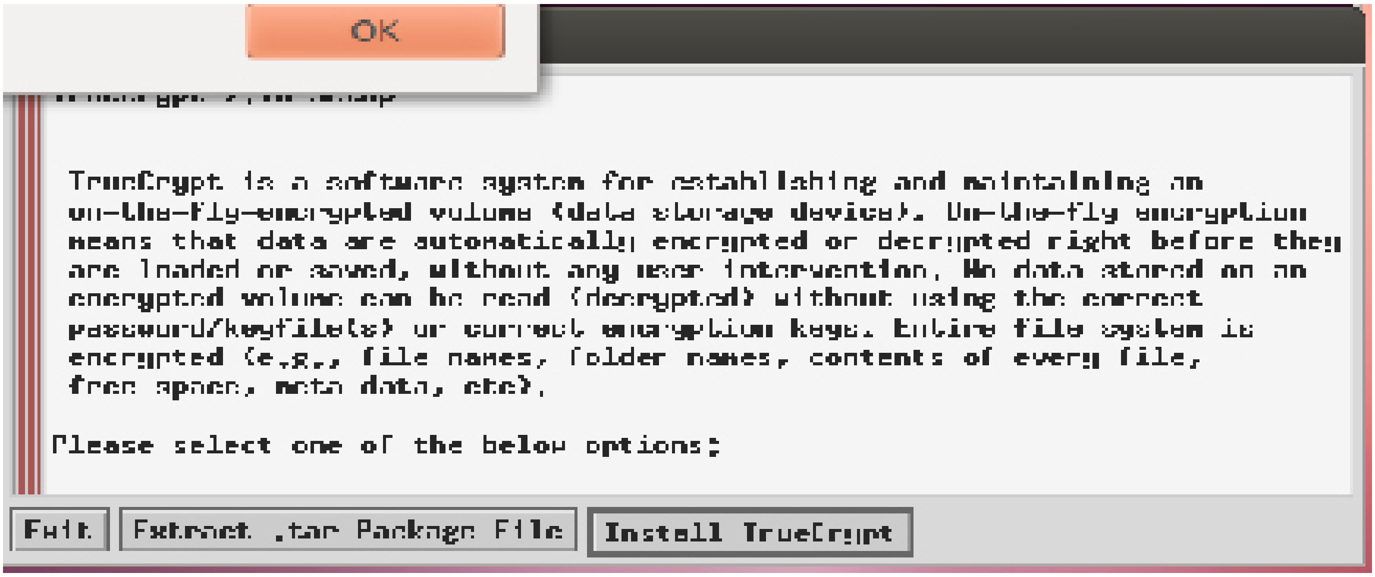
using an advanced packaging tool-capable distribution. This will install the encrypting utilities needed to encrypt your home directory.

The next step is to log in or create another user account with root privileges to run the following command on the user's home directory (Figure 2):

```
sudo ecryptfs-migrate-home -u your-user-name
```

Then, you need to log in to the encrypted home directory account before rebooting the machine (as stated in the important note screen), providing a roll-back opportunity in the event of any unexpected complications during the encryption process.

Use `encryptfs-unwrap-passphrase` to record your randomly generated mount passphrase. Keep this passphrase safe, because you may need it to recover your encrypted files. Also, ensure that you reboot your system and remove the un-encrypted backup folder (Figure 3).



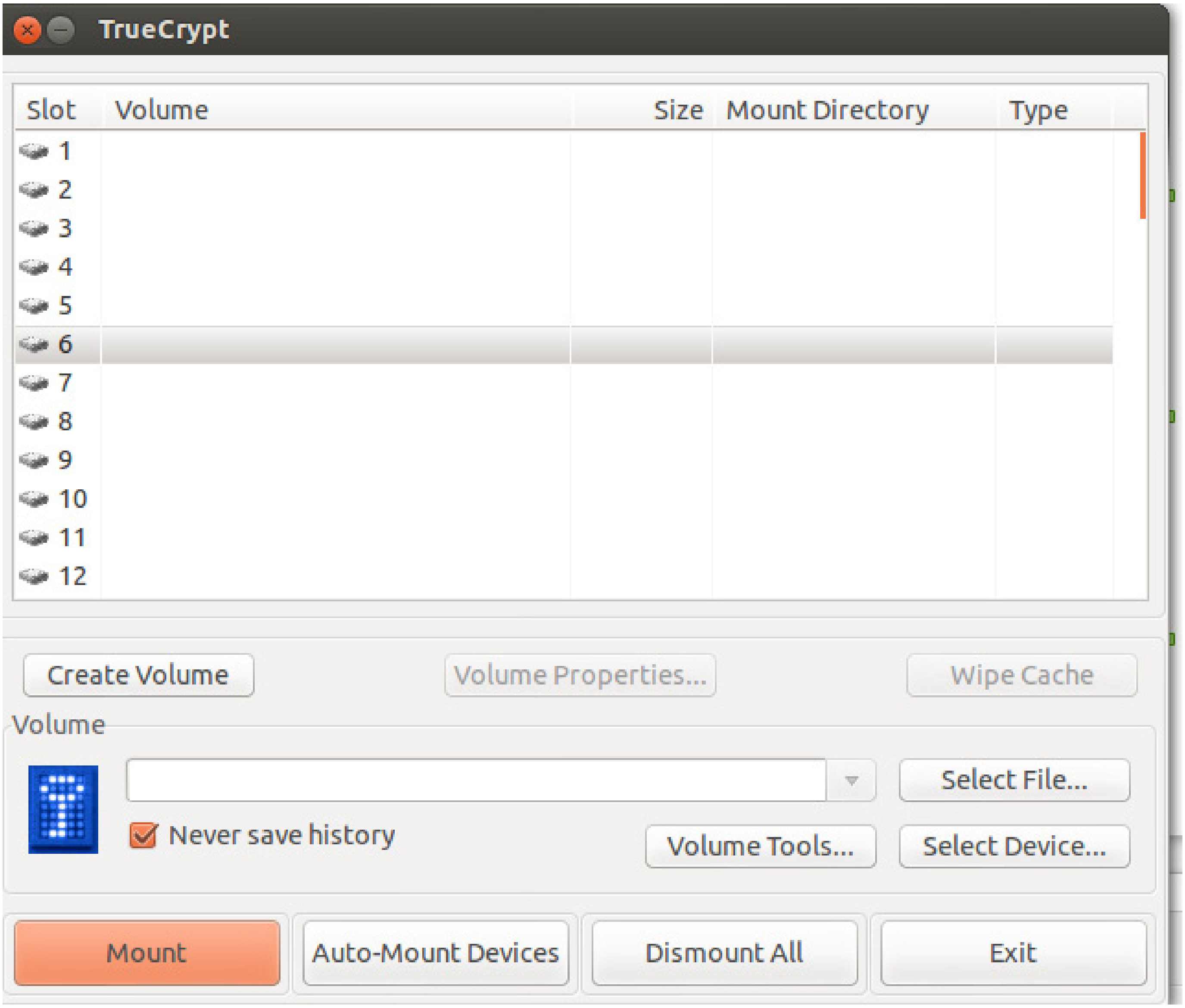
TrueCrypt is a software system for establishing and maintaining an on-the-fly-encrypted volume (data storage device). On-the-fly encryption means that data are automatically encrypted or decrypted right before they are loaded or saved, without any user intervention. No data stored on an encrypted volume can be read (decrypted) without using the correct password/keyfile(s) or correct encryption keys. Entire file system is encrypted (e.g., file names, folder names, contents of every file, free space, meta data, etc).

Please select one of the below options:

Figure 4. TrueCrypt Installation Button

A third step in the process is to utilize a great open-source application called TrueCrypt to provide encrypted containers to store personal information. This easy process includes visiting the TrueCrypt Web site at **http://www.truecrypt.org/downloads** to download the latest package (truecrypt-7.1a-linux-x86.tar.gz, at the time of this writing), and run the following commands and script:

```
tar -xvf truecrypt-7.1a-linux-x86.tar.gz
sudo ./truecrypt-7.1a-linux-x86
select ? Install TrueCrypt at the gui menu.
```



Slot	Volume	Size	Mount Directory	Type
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				

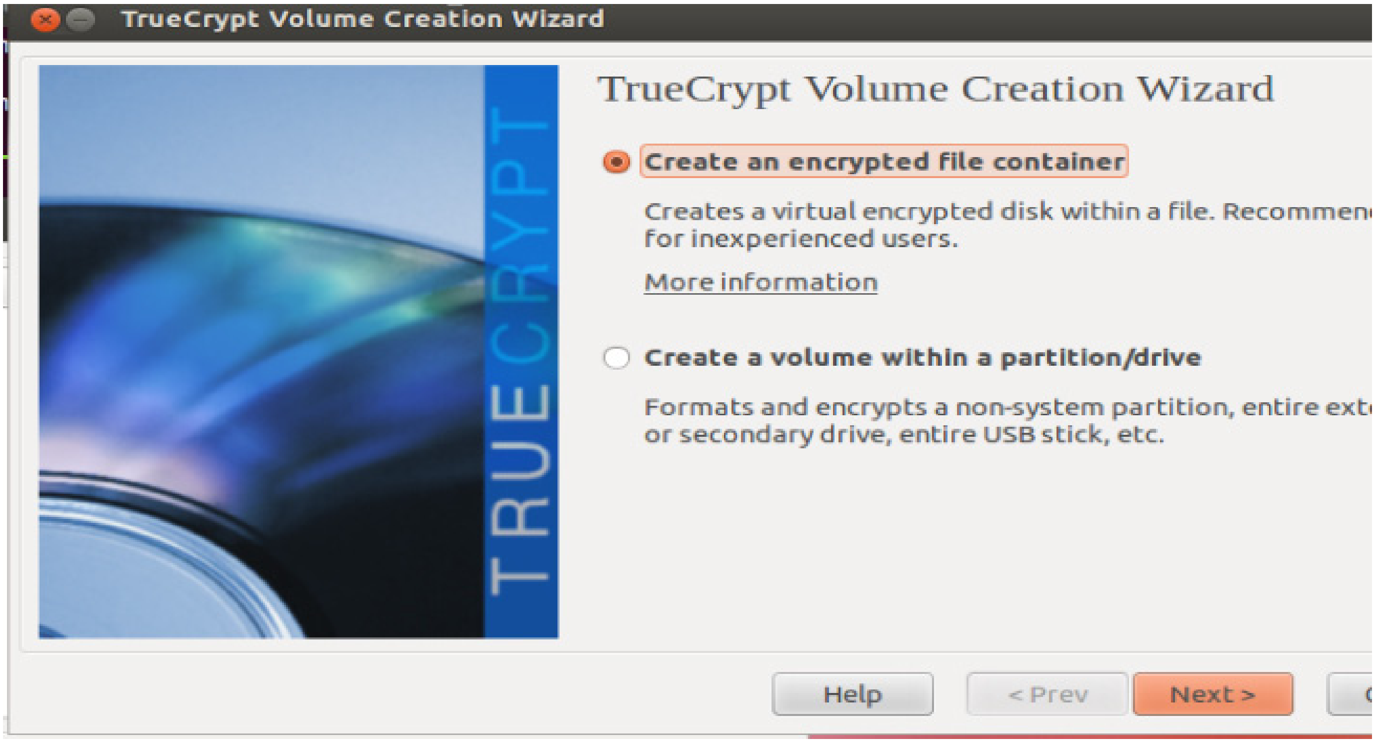
☑ Never save history

Figure 5. TrueCrypt Create Volume Button Screen

The next step is to create an encrypted container. This container will store personal identifiable information (PII) or any file that you want to keep safe on your local computer, and it will create another layer of security. The process for creating a basic container is by selecting the default options during initial installation (Figure 4). Once the software is installed, starting the application is a breeze using the command `truecrypt &` or via the GUI menu system by selecting the create volume button.

There are two options when creating a volume: choosing an encrypted file container or a volume within a partition/drive (Figures 5 and 6). You also will have a choice of using a standard TrueCrypt volume or a hidden TrueCrypt volume (Figure 7). The idea behind a hidden container is to reveal an outside container password, and your hidden container encrypted within the outside container (**http://www.truecrypt.org/docs/hidden-volume**).

On the next menu, simply select an encryption algorithm, hash algorithm and size of container. Multiple books and papers provide



TrueCrypt Volume Creation Wizard

● Create an encrypted file container

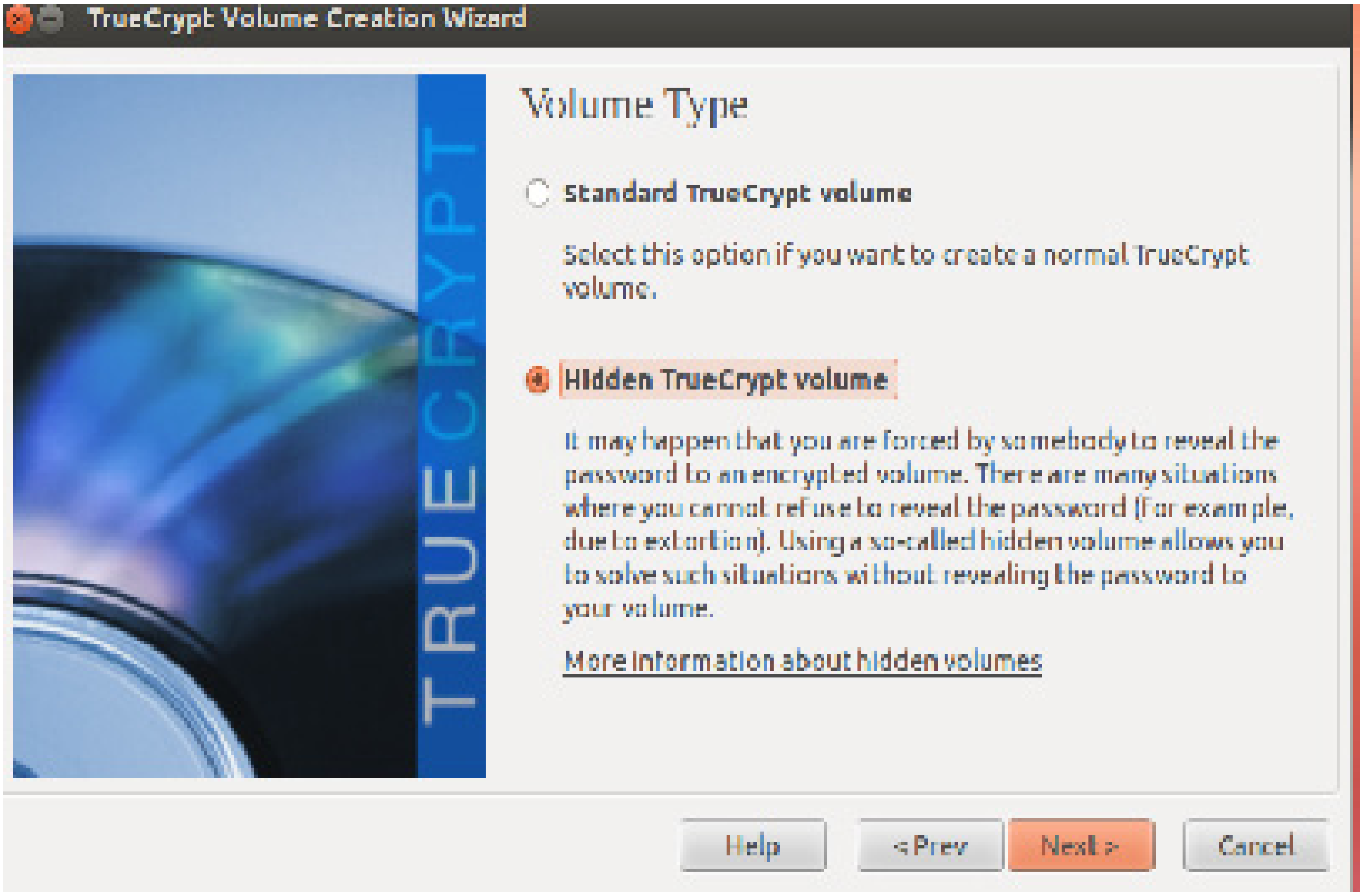
Creates a virtual encrypted disk within a file. Recommen
for inexperienced users.

More information

○ Create a volume within a partition/drive

Formats and encrypts a non-system partition, entire ext
or secondary drive, entire USB stick, etc.

Figure 6. After the create volume button is selected, you will be presented with two options for creating an encrypted file container or creating a volume within a partition/drive.



Volume Type

○ Standard TrueCrypt volume

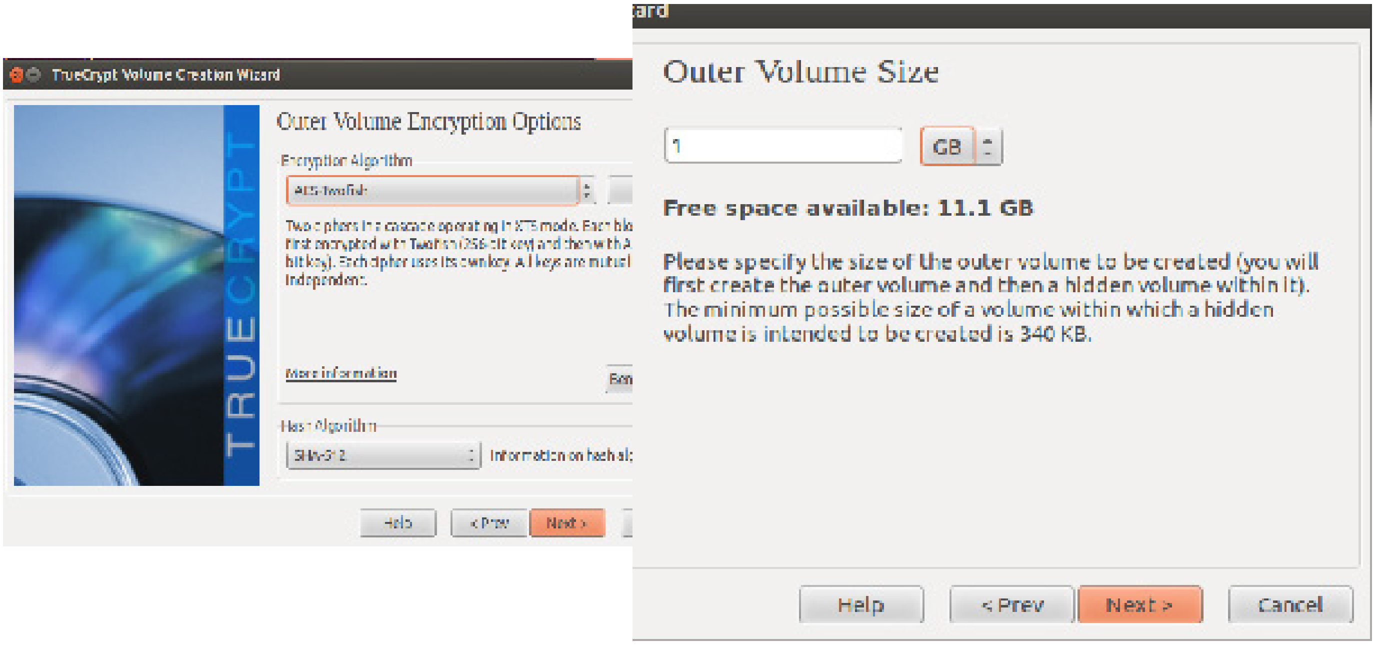
Select this option if you want to create a normal TrueCrypt volume.

● Hidden TrueCrypt volume

It may happen that you are forced by somebody to reveal the password to an encrypted volume. There are many situations where you cannot refuse to reveal the password (for example, due to extortion). Using a so-called hidden volume allows you to solve such situations without revealing the password to your volume.

More information about hidden volumes

Figure 7. The next menu item gives you the option of creating a standard or hidden volume.



Outer Volume Encryption Options

Encryption Algorithm: AES-Twofish

Two ciphers in a cascade operating in XTS mode. Each blo first encrypted with Twofish (256-bit key) and then with A bit key). Each cipher uses its own key. All keys are mutual independent.

More information

Hash Algorithm: SHA-512

Information on hash al

Outer Volume Size

1 GB

Free space available: 11.1 GB

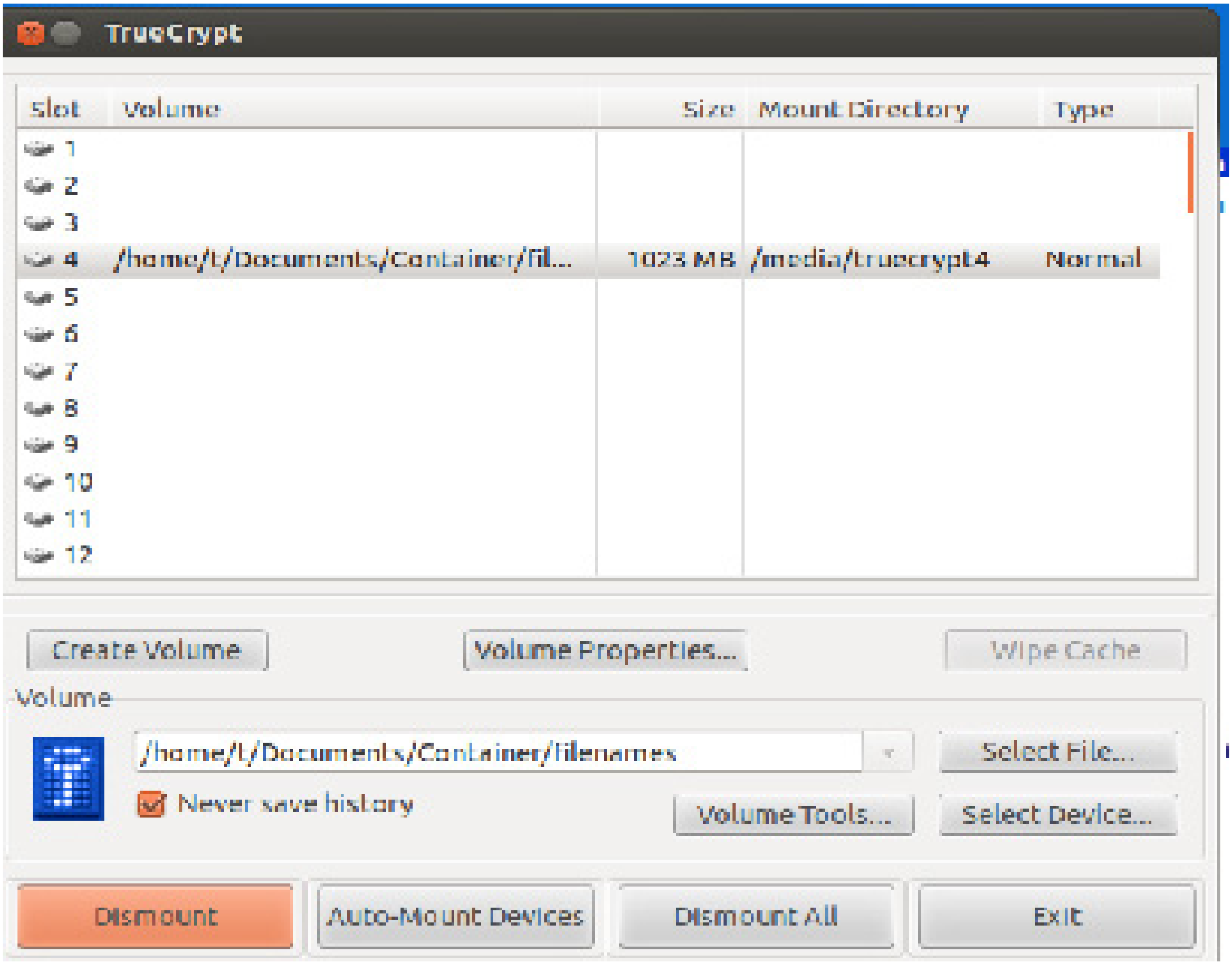
Please specify the size of the outer volume to be created (you will first create the outer volume and then a hidden volume within it). The minimum possible size of a volume within which a hidden volume is intended to be created is 340 KB.

Figure 8. After the standard volume is selected, the next options are to select the encryption and hash algorithms, and size of the volume.

specific information on the differences between these algorithms and hashes (AES with a 256/14 rounds and Sha-512 default hashing function). The size of your container depends on the amount of information you want to protect (Figure 8).

The next step is to select your preferred filesystem type (ext3, ext4 and so on). Once the volume-creating process is completed, mount your volume using the TrueCrypt application and start saving your private files to this encrypted container.

A safe and secure on-line storage location for your newly created encrypted container is essential for backing up data in the cloud. A couple options are available for



Slot	Volume	Size	Mount Directory	Type
1				
2				
3				
4	/home/t/Documents/Container/fil...	1023 MB	/media/truecrypt4	Normal
5				
6				
7				
8				
9				
10				
11				
12				

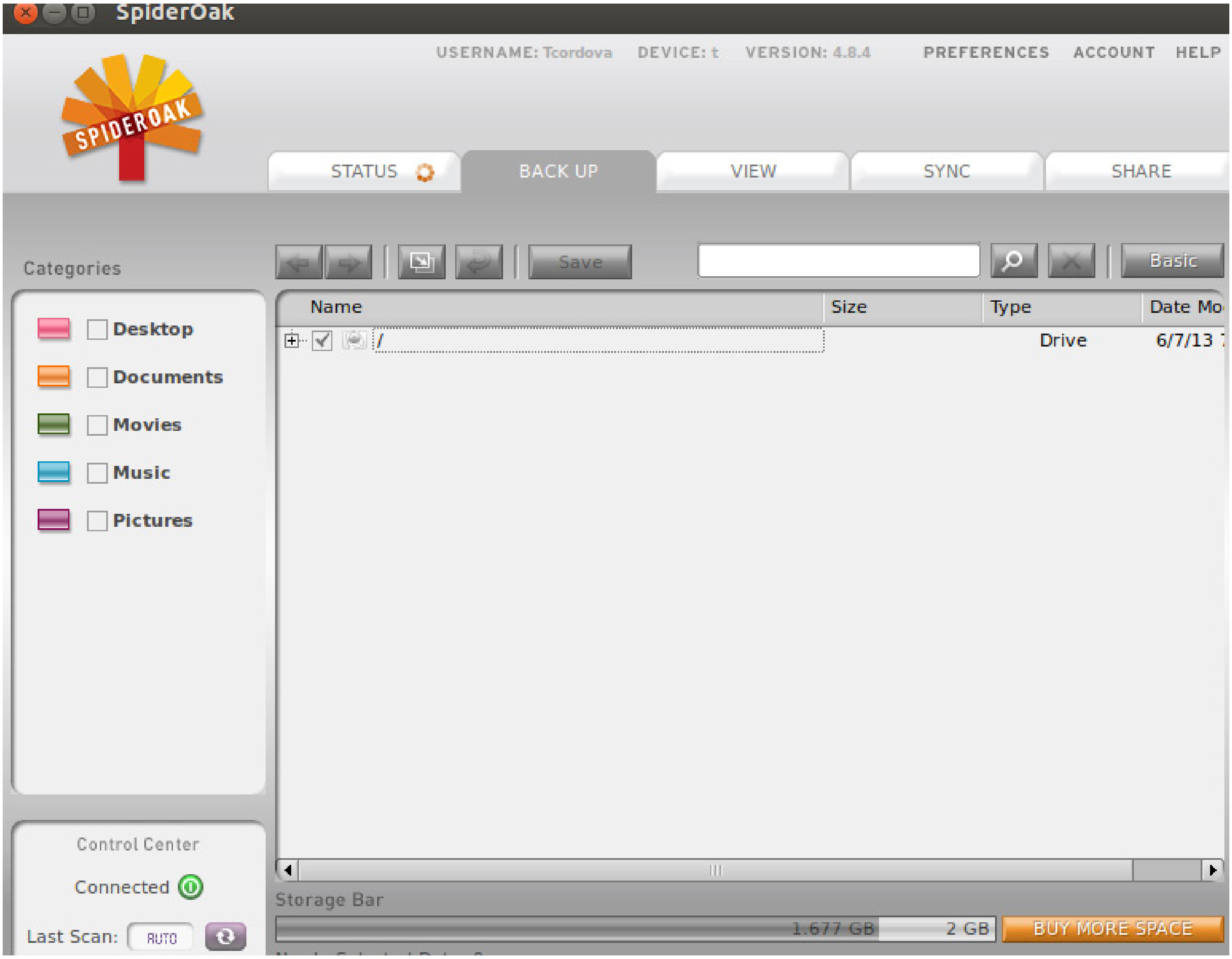
/home/t/Documents/Container/filenames

☑ Never save history

Figure 9. Select the newly created standard volume to mount an accessible unencrypted share.

an on-line storage location, such as Dropbox, Evernote, AWS and SpiderOak. The final choice for secure cloud storage is with the company called SpiderOak, and this is based on the company’s “Zero-Knowledge” privacy policy that states: “we never have any knowledge of your password and no way to retrieve or reset it, even in emergencies. It’s our way of ensuring that our customer’s data is always completely secure—even from us!” (**https://spideroak.com/faq/category/privacy_passwords**).

The company also provides two-factor authentication for extra protection of requiring a user name, password and a token. The token will be sent to your mobile phone whenever you need



USERNAME: Tcordova DEVICE: t VERSION: 4.8.4

Name	Size	Type	Date Mo
☑ /		Drive	6/7/13

Connected

Last Scan: AUTO

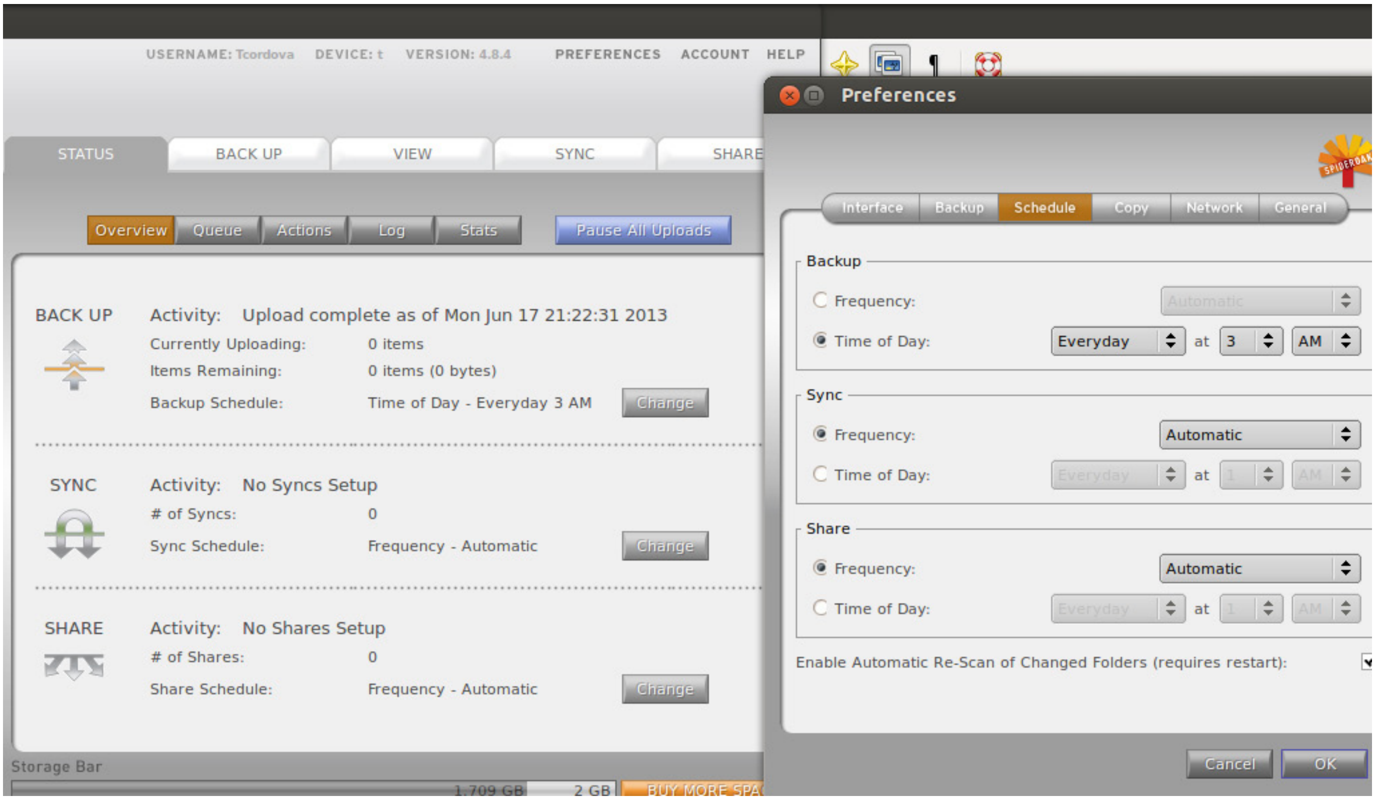
1.677 GB 2 GB

Figure 10. The backup tab in the SpiderOak application allows you to select your encrypted volume.

to log in to a Web site or mobile device. The majority of big-name providers are offering two-factor authentication since the traditional password/passphrase does not offer enough protection. Seeing how this solution is deployed on a dedicated desktop and requires the token to authenticate, it provides a true two-channel authentication solution. Of course, using two-factor authentication does not guarantee safety, but it does require the attacker to use sophisticated methods, and attackers generally are lazy and look for easy targets.

Installing SpiderOak is straightforward for all the Debian users out there. It includes downloading and installing the spideroak_4.8.4_i386.deb package from **https://spideroak.com/opendownload** and using `sudo dpkg -i spideroak_4.8.4_i386.deb` to install this package on your favorite Ubuntu platform.

Identify a local upload folder as the staging point for your TrueCrypt container. Once you have a shared location that will host your TrueCrypt container, simply open your SpiderOak application



USERNAME: Tcordova DEVICE: t VERSION: 4.8.4

BACK UP
Activity: Upload complete as of Mon Jun 17 21:22:31 2013
Currently Uploading: 0 items
Items Remaining: 0 items (0 bytes)
Backup Schedule: Time of Day - Everyday 3 AM

SYNC
Activity: No Syncs Setup
# of Syncs: 0
Sync Schedule: Frequency - Automatic

SHARE
Activity: No Shares Setup
# of Shares: 0
Share Schedule: Frequency - Automatic

1.709 GB 2 GB

Backup
○ Frequency: Automatic
● Time of Day: Everyday at 3 AM

Sync
● Frequency: Automatic
○ Time of Day: Everyday at 1 AM

Share
● Frequency: Automatic
○ Time of Day: Everyday at 1 AM

Enable Automatic Re-Scan of Changed Folders (requires restart): ☑

Figure 11. A SpiderOak application status and backup menu provides a means to back up your encrypted volume automatically in specified intervals.

Listing 1. SpiderOak/TrueCrypt Backup Script

```
#!/usr/bin/python
'''
SpiderOak, TrueCrypt, dis-mount, Backup Script
@author: Tim
'''
import os
import string
import datetime
import hashlib
FolderandFileLoc = "FolderandFileLoc"
SpiderOakPath = " "
TrueCryptPath = " "
LogFilepath = " "
safefile = " "

def readconfigfile(SpiderOakPath,TrueCryptPath,LogFilepath,safefile,
 ➥Setupfileopen):
    # This will read the configuration and assign path location
    now = datetime.datetime.now()
    holdstr = ""
    for line in Setupfileopen:
        holdstr = str.split(line)
        if string.find(line,"SpiderOakPath") > -1:
            SpiderOakPath = holdstr[1]
        elif string.find(line,"TrueCryptPath") > - 1:
            TrueCryptPath = holdstr[1]
        elif string.find(line, "LogFilepath") > -1:
            LogFilepath = holdstr[1]
        elif string.find(line,"safefile") > -1:
            safefile = holdstr[1]

    fo = open(LogFilepath,"a")
    try:
        fo = open(LogFilepath,"a")
        fo.write (str(now) + "- Path Variable SpiderOakPath
         ➥used -> " + SpiderOakPath + "\n")
        fo.write (str(now) + "- Path Variable TrueCryptPath
         ➥used -> " + TrueCryptPath + "\n")
        fo.write (str(now) + "- Path Variable LogFilepath
         ➥used -> " + LogFilepath + "\n")
        fo.write (str(now) + "- Path Variable hold
         ➥used -> " + safefile + "\n")
    except: fo.error
    shutdowntruecrypt(fo,now)
    copycontainer(fo,SpiderOakPath,TrueCryptPath,
    ➥LogFilepath,safefile,now)
    fo.close


def shutdowntruecrypt(fo,now):
    # Test to see if the truecypt is running
    # If not then Shut it down
    foundstring = 0
    try:
        f = os.popen( "ps ax" )
    except: os.error

    for line in f:
        if string.find(line, 'truecrypt') > -1:
            foundstring = 1
                break

    if foundstring == 1:
        try:
            dismount = os.system("truecrypt -d")
            if dismount == 0:
                fo.write (str(now) + "- True Crypt0service found
                 ➥and the volume is dis-mounted \n");
            else:
                fo.write (str(now) + "- Failed to
                 ➥dismount service \n ");
        except: os.error
    else:
        fo.write (str(now) + "- mount was not open \n ");

def copycontainer(fo,SpiderOakPath,TrueCryptPath,
➥LogFilepath,safefile,now):
    #Set Destination and Copy to new location

    Holddestfilesum = TrueCryptPath + safefile
    Holdorigfilesum = SpiderOakPath + "/" + safefile
    checksumdest = md5filecheck(Holddestfilesum)
    checksumorig = md5filecheck(Holdorigfilesum)


    runstring = "cp "  # This will only copy over updates
                       # to this file
    runstring += TrueCryptPath
    runstring += safefile
    runstring += "  "
    runstring += SpiderOakPath  # This will only send over any
                                # updates to this file
    testdiff = os.system("diff " + Holddestfilesum + "
     ➥" + Holdorigfilesum)


    if testdiff !=0:
        try:
            os.system(runstring)
            testdiff = os.system("diff " + Holddestfilesum + "
             ➥" + Holdorigfilesum)
            if testdiff != 0 :
                fo.write (str(now) + TrueCryptPath + safefile +
                 ➥" File Copied to " + SpiderOakPath + "\n")
                fo.write(str(now) +  " ---- Processing Complete ----")
            else:
                fo.write(str(now) + TrueCryptPath + safefile +
                 ➥"File failed to copy " + SpiderOakPath + "\n")
        except: os.error

    else:
        fo.write (str(now) + " File has not been changed
         ➥no copy was performed\n")



Setupfileopen = open(FolderandFileLoc,"r")
readconfigfile(SpiderOakPath,TrueCryptPath,LogFilepath,safefile,
 ➥Setupfileopen)
Setupfileopen.close()
```

and select the backup tab. Then, drill down until you find your TrueCrypt container location, such as home/username/SpiderO/Upload.

The next step is to configure your backup frequency using the overview tab and selecting the change button (Figures 10 and 11).

Many other configuration options are available using this interface. For this example, use only these two options for a secure cloud backup.

The last couple steps in this encrypted backup solution are to move the TrueCrypt container from the working location to the designated SpiderOak export folder and create a cron job to run the script.

I created a Python script to accomplish the copy function, but I could have created any type of script. This script is used to ensure that the TrueCrypt application is not running, verify whether there were changes to the container and then copy over the container if there were changes. This script requires a configuration file called FolderandFileLoc to function and the Python script BackupScript.py. The configuration file parameters are SpiderOakPath, TrueCryptPath and LogFilepath, a running log to verify whether a copy was successful and the Safefile filename.

The final step is to create a cron job to call the Python script:

```
0 5 * * * cd /home/t/workspace/BackupScript/src; /usr/bin/python
/home/t/workspace/BackupScript/src/BackupScript.py
```

This personal encrypted solution is something that works great at home when utilized on a daily basis. Many apps are available on the Internet for managing passwords and data, but this one is easy to implement and provides layers of encryption. I am confident that using the described encrypted containers and storage location provides enough security for private personal data, but it may not be an ideal solution for an enterprise with various regulatory agencies. Use the described methods at your own risk, and ensure that your passwords or passphrases are safeguarded, because your data will be lost with a forgotten password.■

Tim Cordova is a computer geek who had a Commodore 64 at age 9, and has a love for Linux, family, information security and longboard surfing. He currently works as an information security professional at a large contracting company and has more than 15 years of experience.

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

WEBCASTS

ActiveState
Code to Cloud: Smarter, Safer, Faster™

A Call to Arms for Private Cloud Builders

Sponsor: ActiveState | **Topic:** Cloud Computing **ON DEMAND**

The era of elastic IT is here. Businesses are realizing that the cloud not only allows cost reduction, but provides opportunities for innovation and growth. Elastic clouds enable next-generation applications that drive revenue opportunities, increase agility, and make IT teams competitive with public cloud systems.

In this presentation, Randy and John talk about the forces driving this change, and outline an action plan for building an elastic cloud infrastructure and dynamic applications using DevOps and Platform-as-a-Service.

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

ActiveState
Code to Cloud: Smarter, Safer, Faster™

Private PaaS for the Agile Enterprise

Sponsor: ActiveState | **Topic:** Virtualization

If you already use virtualized infrastructure, you are well on your way to leveraging the power of the cloud. Virtualization offers the promise of limitless resources, but how do you manage that scalability when your DevOps team doesn't scale? In today's hypercompetitive markets, fast results can make a difference between leading the pack vs. obsolescence. Organizations need more benefits from cloud computing than just raw resources. They need agility, flexibility, convenience, ROI, and control.

Stackato private Platform-as-a-Service technology from ActiveState extends your private cloud infrastructure by creating a private PaaS to provide on-demand availability, flexibility, control, and ultimately, faster time-to-market for your enterprise.

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

IBM®

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**

Linux Backup and Recovery Webinar

Sponsor: Storix | **Topic:** Backup and Recovery

Most companies incorporate backup procedures for critical data, which can be restored quickly if a loss occurs. However, fewer companies are prepared for catastrophic system failures, in which they lose all data, the entire operating system, applications, settings, patches and more, reducing their system(s) to "bare metal." After all, before data can be restored to a system, there must be a system to restore it to.

In this one hour webinar, learn how to enhance your existing backup strategies for better disaster recovery preparedness using Storix System Backup Administrator (SBAdmin), a highly flexible bare-metal recovery solution for UNIX and Linux systems.

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

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**

Solid-State Drives: Get One Already!

Brian describes how SSDs compare to HDDs with regard to longevity and reliability and provides the results from some real-world performance benchmarking.

BRIAN TRAPP

I've been building computers since the 1990s, so I've seen a lot of new technologies work their way into the mainstream. Most were the steady, incremental improvements predicted by Moore's law, but others were game-changers, innovations that really rocketed performance forward in a surprising way. I remember booting up *Quake* after installing my first 3-D card—what a difference! My first boot off a solid-state drive (SSD) brought back that same feeling—wow, what a difference!

However, at a recent gathering of like-minded Linux users, I learned that many of my peers hadn't actually made the move to SSDs yet. Within that group, the primary reluctance to try a SSD boiled down to three main concerns:

- I'm worried about their reliability; I hear they wear out.
- I'm not sure if they work well with Linux.
- I'm not sure an SSD really would make much of a difference on my system.

Luckily, these three concerns are based either on misunderstandings, outdated data, exaggeration or are just not correct.

SSD Reliability Overview

How SSDs Differ from Hard Drives: Traditional hard disk drives (HDDs) have two mechanical delays that can come into play when reading or writing files: pivoting the read/write head to be at the right radius and waiting until the platter rotates until the start of the file reaches the head (Figure 1). The time it takes for the drive to get in place to read a new file is called seek time. When you hear that unique hard drive chatter, that's the actuator arm moving around to access lots of different file locations. For example, my hard drive (a pretty typical 7,200 RPM consumer drive from 2011) has an average seek time of around 9ms.

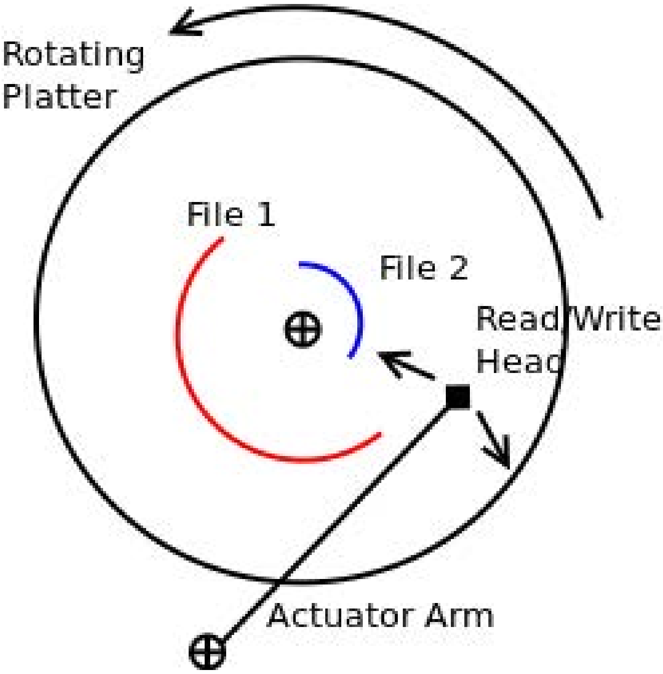


Figure 1. Hard Drive

Instead of rotating platters and read/write heads, solid-state drives store data to an array of Flash memory chips. As a result, when a new file is requested, the SSD's internal memory can find and start accessing the correct storage memory locations in sub-milliseconds. Although reading from Flash isn't terribly fast by itself, SSDs can read from several different chips in parallel to boost performance. This parallelism and the near-instantaneous seek times make solid-state drives significantly faster than hard drives in most benchmarks. My SSD (a pretty typical unit from 2012) has a seek time of 0.1ms—quite an improvement!

Reliability and Longevity: Reliability numbers comparing HDDs and SSDs are surprisingly hard to find. Fail rate comparisons either didn't have enough years of data, or were based on old first-generation SSDs that don't represent drives currently on the market. Though SSDs reap the benefits of not having any moving parts (especially beneficial for mobile devices like laptops), the conventional wisdom is that current SSD fail rates are close to HDDs. Even if they're a few percentage points higher or lower, considering that *both* drive types have a nonzero failure rate, you're going to need to have a backup solution in *either* case.

Apart from reliability, SSDs do have a unique longevity issue, as the NAND Flash cells in storage have a unique life expectancy limitation. The longevity of each cell depends on what type of cell it is. Currently, there are three types of NAND Flash cells:

- SLC (Single Later Cell) NAND: one bit per cell, ~100k writes.
- MLC (Multi-Layer Cell) NAND: two bits per cell, ~10k to 3k writes, slower than SLC. The range in writes depends on the physical size of the cell—smaller cells are cheaper to manufacture, but can handle fewer writes.
- TLC (Three-Layer Cell) NAND: ~1k writes, slower than MLC.

Interestingly, all three types of cells are using the same transistor structure behind the scenes. Clever engineers have found a way to make that single Flash cell hold more information in MLC or TLC mode, however. At programming time, they can use a low, medium-low, medium-high or high voltage to represent four unique states (two bits) in one single cell. The downside is that as the cell is written several thousand times, the oxide insulator at the bottom of the floating gate starts to degrade, and the amount of voltage required for each state increases (Figure 2). For SLC it's not a huge deal because the gap between states is so big, but for MLC, there are four states instead of two, so the amount of room between each state's voltage is shortened. For TLC's three bits of

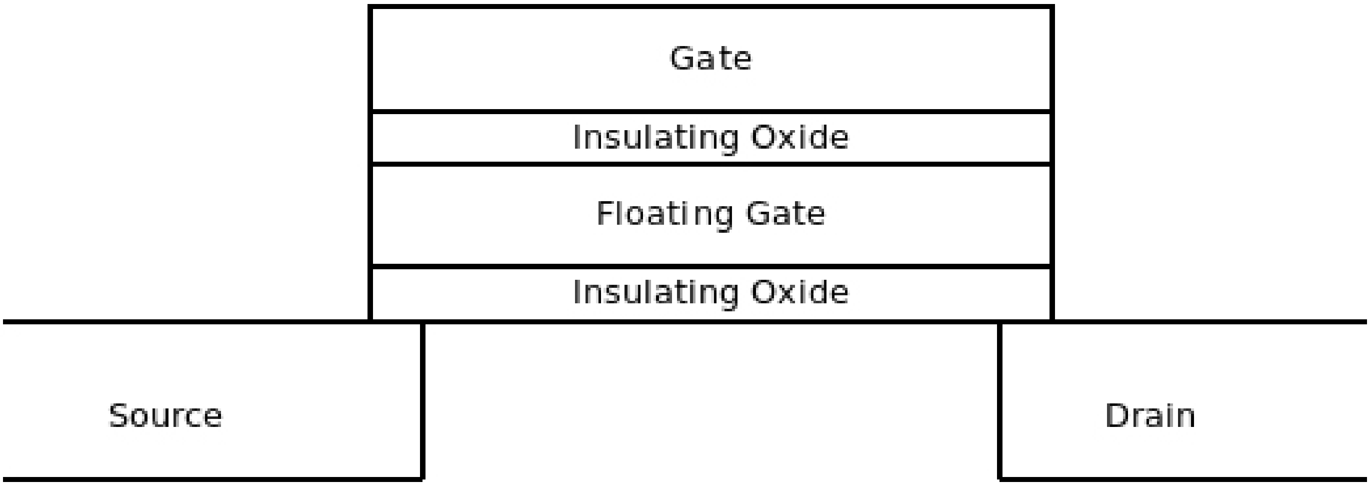


Figure 2. A NAND Flash Cell

information there are six states, so the distances between each voltage range is even shorter.

The final twist is write amplification. Even though the OS is sending 1MB of data, the SSD actually may be doing more writes behind the scenes for things like wear leveling and inefficient garbage collection if TRIM support isn't enabled (see the TRIM section later in this article). Most real-world write amplification values I've seen are in the 1.1 to 3.0 range, depending on how compressible the data is and how clever the SSD is at garbage collection and wear leveling.

So, how long can you expect an SSD to last for you? Longevity depends on how much data you write, and the tune2fs utility makes it really easy to estimate that from your existing filesystems. Run `tune2fs -l /dev/<device>`. (Tip: if you're using LVM, the stats will be under the dm-X device instead of the sdaX device.) The key fields of interest are "Filesystem created" and "Lifetime writes". Use those to figure out the average GB/day since the filesystem was created. For my laptop, it was 2.7GB/day, and for my workstation it was 6.3GB/day. With those rates, plus a rough guess for write amplification, you can estimate how much life you'd get out of any SSD.

$$\text{Est. Lifespan (y)} = \frac{\text{SSDCapacity(GB)} * \text{(WriteLimit based on cell type)}}{\text{DailyWriteRate (GB/day)} * \text{WriteAmplification} * \text{365 (days/yr)}}$$

So if I was sizing a 256GB Samsung 840 Evo (which uses TLC cells), with a 6.3GB/day write rate and a write amplification of 3, it should give me around 37 years of service before losing the ability to write new data.

SSD Considerations for Linux

TRIM: Undelete utilities work because when you delete a file, you're really only removing the filesystem's pointer to that file, leaving the file contents behind on the disk. The filesystem knows about the newly freed space and eventually will reuse it, but the drive doesn't. HDDs can overwrite data just as efficiently as writing to a new sector, so it doesn't really hurt them, but this can slow down SSDs' write operations, because they can't overwrite data efficiently.

An SSD organizes data internally into 4k pages and groups 128 pages into a 512k block. SSDs can write only into empty 4k pages and erase in big 512k block increments. This means that although SSDs can write very quickly, *overwriting* is a much slower process. The TRIM command keeps your SSD running at top speed by giving the filesystem a way to tell the SSD about deleted pages. This gives the drive a chance to do the slow overwriting procedures in the backgroupd, ensuring that you always have a large pool of empty 4k pages at your disposal.

Linux TRIM support is not enabled by default, but it's easy to add. One catch is that if you have additional software layers between your filesystem and SSD, those layers need to be TRIM-enabled too. For example, most of my systems have an SSD, with LUKS/dm-crypt for whole disk encryption, LVM for simple volume management and then, finally, an ext4 formatted filesystem. Here's how to turn on TRIM support, starting at the layer closest to the drive.

dm-crypt and LUKS: If you're not using an encrypted filesystem, you can skip ahead to the LVM instructions. TRIM has been supported in dm-crypt since kernel 3.1. Modify /etc/crypttab, adding the discard keyword for the devices on SSDs:

```
#TargetName Device                             KeyFile  Options
sda5_crypt  UUID=9ebb4c49-37c3...d514ae18be09  none     luks,discard
```

Note: enabling TRIM on an encrypted partition does make it easier for attackers to brute-force attack the device, since they would now know which blocks are not in use.

LVM: If you're not using LVM, you can skip ahead to the filesystem section. TRIM has been supported in LVM since kernel 2.6.36.

In the "devices" section of /etc/lvm/lvm.conf, add a line `issue_discards = 1`:

```
devices {
        ...
        issue_discards = 1
        ..
}
...
```

Filesystem: Once you've done any required dm-crypt and LVM edits, update initramfs, then reboot:

```
sudo update-initramfs -u -k all
```

Although Btrfs, XFS, JFS and ext4 all support TRIM, I cover only ext4 here, as that seems to be the most widely used. To test ext4 TRIM support, try the manual TRIM command: `fstrim <mountpoint>`. If all goes well, the command will work for a while and exit. If it exits with any error, you know there's something wrong in the setup between the filesystem and the device. Recheck your LVM and dm-crypt setup.

Here's an example of the output for

/ (which is set up for TRIM) and /boot (which is not):

```
~$ sudo fstrim /
~$ sudo fstrim /boot
fstrim: /boot: FITRIM ioctl failed: Inappropriate ioctl for device
```

If the manual command works, you can decide between between using the automatic TRIM built in to the ext4 filesystem or running the `fstrim` command. The primary benefits of using automatic TRIM is that you don't have to think about it, and it nearly instantly will reclaim free space. One down side of automatic TRIM is that if your drive doesn't have good garbage-collection logic, file deletion can be slow. Another negative is that if the drive runs TRIM quickly, you have no chance of getting your data back via an undelete utility. On drives where I have plenty of free space, I use the fstrim command via cron. On drives where space is tight, I use the automatic ext4 method.

If you want to go the automatic route, enabling automatic TRIM is easy—just add the `discard` option to the options section of the relevant /etc/fstab entries. For manual TRIM, just put the `fstrim <mountpoint>` in a cron job or run it by hand at your leisure.

Regardless of whether you use the `discard` option, you probably want to add the `noatime` option to /etc/fstab. With atime on (the default), each time a file is accessed, the access time is updated, consuming some of your precious write cycles. (Some tutorials ask you to include nodiratime too, but noatime is sufficient.) Because most applications don't use the atime timestamp, turning it off should improve the drive's longevity:

```
/dev/mapper/baldyl-root  /  ext4  noatime,discard,errors=remount-ro 0 1
```

Partition alignment: When SSDs first were released, many of the disk partitioning systems still were based on old sector-based logic for placing partitions. This could cause a problem if the partition boundary didn't line up nicely with the SSD's internal 512k block erase size. Luckily, the major partitioning tools now default to 512k-compatible ranges:

- fdisk uses a one megabyte boundary since util-linux version 2.17.1 (January 2010).
- LVM uses a one megabyte boundary as the default since version 2.02.73 (August 2010).

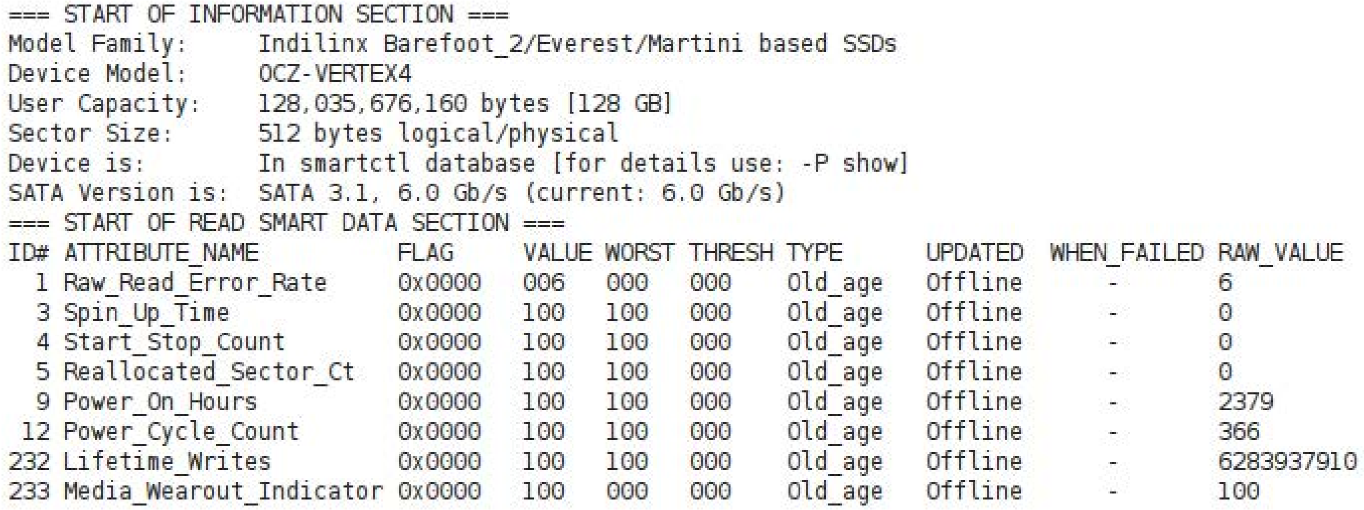
If you're curious whether your partitions are aligned to the right boundaries, here's example output from an Intel X25-M SSD with an erase block size of 512k:

```
~$ sudo sfdisk -d /dev/sda
Warning: extended partition does not start at a cylinder boundary.
DOS and Linux will interpret the contents differently.
# partition table of /dev/sda
unit: sectors

/dev/sda1 : start=     2048, size=   497664, Id=83, bootable
/dev/sda2 : start=   501758, size=155799554, Id= 5
/dev/sda3 : start=        0, size=        0, Id= 0
/dev/sda4 : start=        0, size=        0, Id= 0
/dev/sda5 : start=   501760, size=155799552, Id=83
```

Since the primary partition (sda5) starts and ends at a number evenly divisible by 512, things look good.

Monitoring SSDs in Linux:

I already covered running `tune2fs -l <device>` as a good place to get statistics on a filesystem device, but those are reset each time you reformat the filesystem. What if you want to get a longer range of statistics, at the drive level? smartctl is the tool for that. SMART (Self-Monitoring, Analysis and Report Technology) is part of the ATA standard that provides a way for drives to track and report key statistics, originally for the purposes of predicting drive failures. Because drive write volume is so important to SSDs, most manufacturers are including this in the SMART output. Run `sudo smartctl -a /dev/<device>` on an SSD device, and you'll get a whole host of interesting statistics. If you see the



```
=== START OF INFORMATION SECTION ===
Model Family:     Indilinx Barefoot_2/Everest/Martini based SSDs
Device Model:     OCZ-VERTEX4
User Capacity:    128,035,676,160 bytes [128 GB]
Sector Size:      512 bytes logical/physical
Device is:        In smartctl database [for details use: -P show]
SATA Version is:  SATA 3.1, 6.0 Gb/s (current: 6.0 Gb/s)
=== START OF READ SMART DATA SECTION ===
ID# ATTRIBUTE_NAME          FLAG     VALUE WORST THRESH TYPE      UPDATED  WHEN_FAILED RAW_VALUE
  1 Raw_Read_Error_Rate     0x0000   006   000   000    Old_age   Offline      -       6
  3 Spin_Up_Time            0x0000   100   100   000    Old_age   Offline      -       0
  4 Start_Stop_Count        0x0000   100   100   000    Old_age   Offline      -       0
  5 Reallocated_Sector_Ct   0x0000   100   100   000    Old_age   Offline      -       0
  9 Power_On_Hours          0x0000   100   100   000    Old_age   Offline      -       2379
 12 Power_Cycle_Count       0x0000   100   100   000    Old_age   Offline      -       366
232 Lifetime_Writes         0x0000   100   100   000    Old_age   Offline      -       6283937910
233 Media_Wearout_Indicator 0x0000   100   000   000    Old_age   Offline      -       100
```

Figure 3. smartctl Output (Trimmed)

message "Not in smartctl database" in the smartctl output, try building the latest version of smartmontools.

Each vendor's label for the statistic may be different, but you should be able to find fields like "Media_Wearout_Indicator" that will count down from 100 as the drive approaches the Flash wear limit and fields like "Lifetime_Writes" or "Host_Writes_32MiB" that indicate how much data has been written to the drive (Figure 3).

Other Generic Tips

Swap: if your computer is actively using swap space, additional RAM probably is a better upgrade than an SSD. Given the fact that longevity is so tightly coupled with writes, the last thing you want is to be pumping multiple gigabytes of swap on and off the drive.

HDDs still have a role: if you have the space, you can get the best of both worlds by keeping your hard drive around. It's a great place for storing music, movies and other media that doesn't require fast I/O. Depending on how militant you want to be about SSD writes, you can mount folders like /tmp, /var or even just /var/log on the HDD to keep SSD writes down. Linux's flexible mounting and partitioning tools make this a breeze.

SSD free space: SSDs run best when there's plenty of free space for them to use for wear leveling and garbage collection. Size up and manage your SSD to keep it less than 80% full.

Things that break TRIM: RAID setups can't pass TRIM through to the underlying drives, so use this mode with caution. In the BIOS, make sure your controller is set to AHCI mode and not IDE emulation, as IDE mode doesn't support TRIM and is slower in general.

SSD Performance

Now let's get to the heart of the matter—practical, real-world examples of how an SSD will make common tasks faster.

Test Setup Prior to benchmarking, I had one SSD for my Linux OS, another SSD for when I needed to boot in to Windows 7 and an HDD for storing media files and for doing low-throughput, high-volume work (like debugging JVM dumps or encoding video). I used `partimage` to back up the HDD, and then I used a Clonezilla bootable CD to clone my Linux SSD onto the HDD. Although most sources say you don't have to worry about fragmentation on ext4, I used the ext4 defrag utility e4defrag on the HDD just to give it the best shot at keeping up with the SSD.

Here's the hardware on the

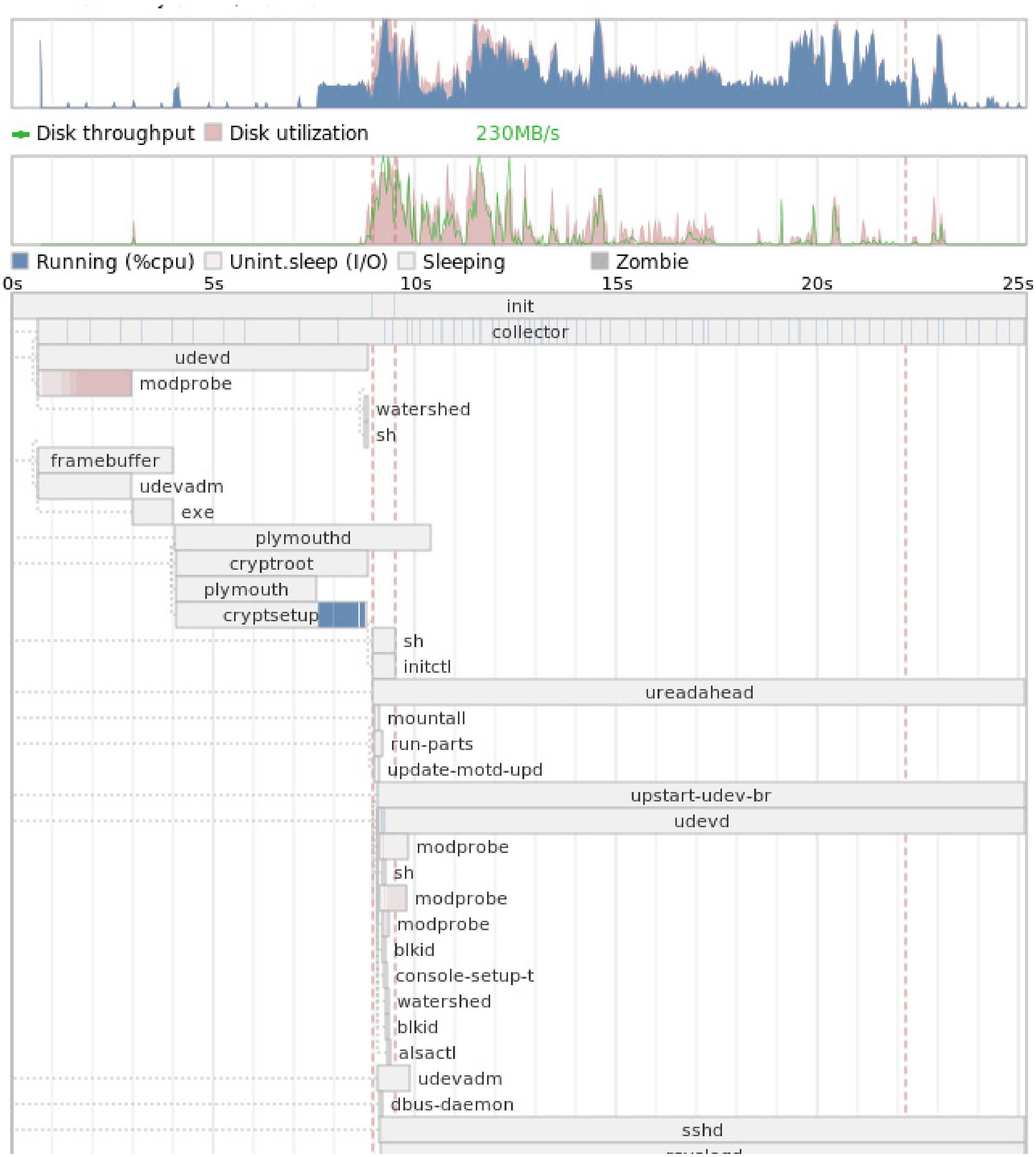


Figure 4. bootchart Output

development workstation I used for benchmarking—pretty standard stuff:

- CPU: 3.3GHz Intel Core i5-2500k CPU.
- Motherboard: Gigabyte Z68A-D3H-B3 (Z68 chipset).
- RAM: 8GB (2x4GB) of 1333 DDR3.
- OS: Ubuntu 12.04 LTS (64-bit, kernel 3.5.0-39).
- SSD: 128GB OCZ Vertex4.
- HDD: 1TB Samsung Spinpoint F3, 7200 RPM, 32MB cache.

I picked a set of ten tests to try to showcase some typical Linux operations. I cleared the disk cache after each test with `echo 3 | sudo tee /proc/sys/vm/drop_caches` and rebooted after completing a set. I ran the set five times for each drive, and plotted the mean plus a 95% confidence interval on the bar charts shown below.

Boot Times: Because I'm the only user on the test workstation and use whole-disk encryption, X is set up with automatic login. Once cryptsetup prompts me for my disk password, the system will go right past the typical GDM user login to my desktop. This complicates how to measure boot times, so to get the most accurate measurements, I used the bootchart package that provides a really cool Gantt chart showing the boot time of each component (partial output shown in Figure 4). I used the Xorg process start to indicate when X starts up, the start of the Dropbox panel applet to indicate when X is usable and subtracted the time spent in cryptsetup (its duration depends more on how many tries it takes me to type in my disk password than how fast any of the disks are). The SSD crushes the competition here.

Table 1. Boot Times

Test	HDD (s)	SSD (s)	% Faster
Xorg Start	19.4	4.9	75%
Desktop Ready	33.4	6.6	80%

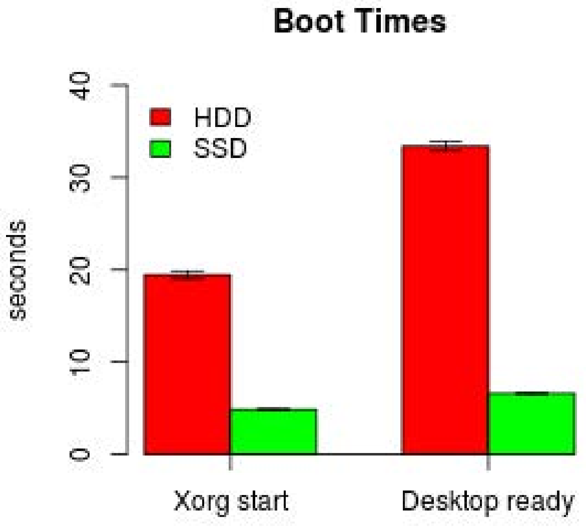


Figure 5. Boot Times

Table 2. Application Launch Times

Test	HDD (s)	SSD (s)	% Faster
Eclipse	26.8	11.0	59%
Tomcat	19.6	17.7	10%
TF2	72.2	67.1	7%

Table 3. File I/O

Test	HDD (s)	SSD (s)	% Faster
create	1.5	0.5	67%
copy	3.3	1.1	69%
read	2.2	0.2	63%

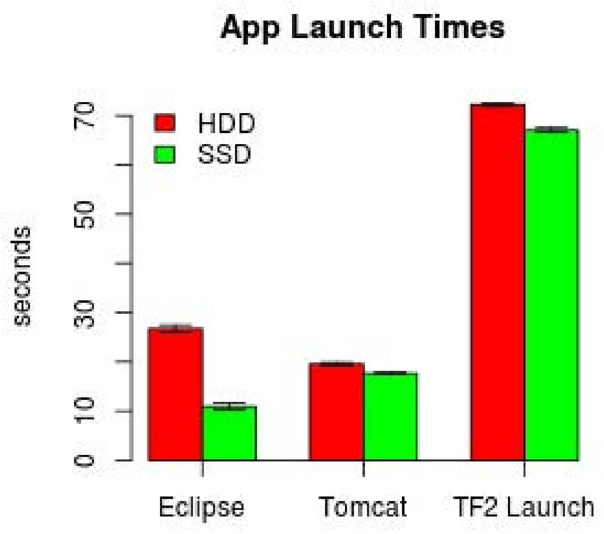


Figure 6. Application Launch Times

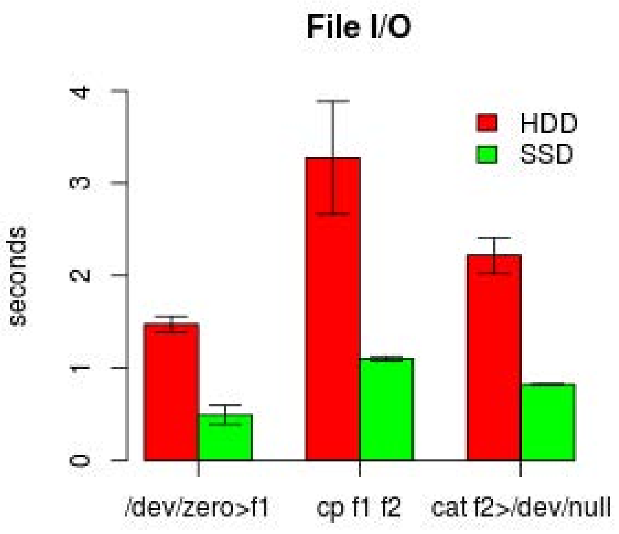


Figure 7. File I/O

Application Start Times: To test application start times, I measured the start times for Eclipse 4.3 (J2EE version), *Team Fortress 2* (*TF2*) and Tomcat 7.0.42. Tomcat had four WAR files at about 50MB each to unpackage at start. Tomcat provides the server startup time in the logs, but I had to measure Eclipse and *Team Fortress* manually. I stopped timing Eclipse once the workspace was visible. For *TF2*, I used the time between pressing "Play" in the Steam client and when the *TF2* "Play" menu appears.

There was quite a bit of variation between the three applications, where Eclipse benefited from an SSD the most, and the gains in Tomcat and *TF2* were present but less noticeable.

Single-File Operations: To test single-file I/O speed, I created a ~256MB file via `time dd if=/dev/zero of=f1 bs=1048576 count=256`, copied it to a new file and then read it via `cat`, redirecting to /dev/null. I used the time utility to capture the real elapsed time for each test.

Multiple File Operations: First, I archived the 200k files in my 1.1GB Eclipse workspace via `tar -c ~/workspace > w.tar` to test

Table 4. Multi-File I/O

Test	HDD (s)	SSD (s)	% Faster
tar	123.2	17.5	86%
find & fgrep	34.3	12.3	64%

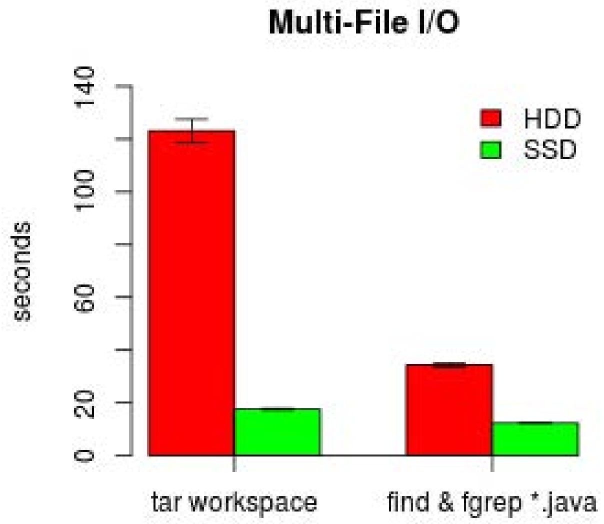


Figure 8. Multi-File I/O

archiving speed. Second, I used `find -name "*.java" -exec fgrep "Foo" {} > /dev/null` to simulate looking for a keyword in the 7k java files. I used the time utility to capture the real elapsed time for each test. Both tests made the HDD quite noisy, so I wasn't surprised to see a significant delta.

Summary

If you haven't considered an SSD, or were holding back for any of the reasons mentioned here, I hope this article prompts you to take the plunge and try one out.

For reliability, modern SSDs are performing on par with HDDs. (You need a good backup, either way.) If you were concerned about longevity, you can use data from your existing system to approximate how long a current generation MLC or TLC drive would last.

SSD support has been in place in Linux for a while, and it works well even if you just do a default installation of a major Linux distribution. TRIM support, some ext4 tweaks and monitoring via tune2fs and smartctl are there to help you maintain and monitor overall SSD health.

Finally, some real-world performance benchmarks illustrate how an SSD will boost performance for *any* operation that uses disk storage, but especially ones that involve many different files.

Because even OS-only budget-sized SSDs can provide significant performance gains, I hope if you've been on the fence, you'll now give one a try.■

Brian Trapp serves up a spicy gumbo of Web-based yield reporting and analysis tools for hungry semiconductor engineers at one of the leading semiconductor research and development consortiums. His signature dish has a Java base with a dash of JavaScript, Perl, Bash and R, and his kitchen has been powered by Linux ever since 1998. He works from home in Buffalo, New York, which is a shame only because that doesn't really fit the whole chef metaphor.

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

DOC SEARLS

Returning to Ground from the Web's Clouds

Fixing problems of centralization with more centralized systems only makes the problem worse.

The Net as we know it today first became visible to me in March 1994, when I was among several hundred other tech types gathered at Esther Dyson's PC Forum conference in Arizona. On stage was John Gage (**http://en.wikipedia.org/wiki/John_Gage**) of Sun Microsystems, projecting a Mosaic Web browser (**http://en.wikipedia.org/wiki/Mosaic_(web_browser)**) from a flaky Macintosh Duo (**http://en.wikipedia.org/wiki/PowerBook_Duo**), identical to the one on my lap. His access was to Sun over dial-up.

Everybody in the audience knew about the Net, and some of us had been on it one way or another, but few of us had seen it in the fullness John demonstrated there. (At that date, there were a sum total of just three Internet Service Providers.) James Fallows (**http://www.theatlantic.com/james-fallows**) was in the crowd, and he described it this way (**http://listserv.aera.net/scripts/wa.exe?A2=ind9406&L=aera-f&D=0&P=351**) for *The Atlantic*:

> In the past year millions of people have heard about the Internet, but few people outside academia or the computer industry have had a clear idea of what it is or how it works. The Internet is, in effect, a way of combining computers all over the world into one big computer, which you seemingly control from your desk. When connected to the Internet, you can boldly prowl through computers in Singapore, Buenos Aires, and

While relying on the Web and its clouds has increased the range of things we can do on the Net, our freedom to act independently has declined.

Seattle as if their contents resided on your own machine.

In the most riveting presentation of the conference, John Gage, of Sun Microsystems, demonstrated the World Wide Web, the gee-whizziest portion of the Internet, in which electronic files contain not only text but also graphics and sound and video clips. Using Mosaic, a free piece of "navigator" software that made moving around the Web possible, Gage clicked on icons on his screen exactly as if he were choosing programs or directories on his own hard disk. He quickly connected to a Norwegian computer center that had been collecting results during the Winter Olympics in Lillehammer and checked out a score, duplicating what Internet users had done by the millions every day during the games, when CBS-TV was notoriously late and America-centric in reporting results.

Note the terms here. John used Mosaic to "control", "boldly prowl" and "navigate" his way around the Web, which was the "gee-whizziest portion" of the Net.

That portion has since become conflated with the whole thing. Today we use browsers to do far more than navigate the Web. Protocols that once required separate apps—file transfer, e-mail, instant messaging—are now handled by browsers as well. We now also can use browsers to watch television, listen to radio and read publications. It's hard to name anything a computer can do that isn't also doable (and done) in a browser. Serving up most of those capabilities are utility Web services, provided by Amazon, Apple, Dropbox, Evernote, Google, Yahoo and many more, each with their own clouds. The growth of the Web, atop the Net, also has provided a conceptual bridge from computers to smartphones and tablets. Today nearly every mobile app would be useless without a back-end cloud.

While relying on the Web and its clouds has increased the range of things we can do on the Net, our freedom to act independently has declined. The browser that started out as a car on the "information superhighway" has become a shopping cart that gets re-skinned with every commercial site it visits, carrying away tracking beacons that report our activities back to centralized servers over which we have little if any control. The wizards among us might be adept at maintaining some degree of liberty from surveillance, but most muggles are either clueless about the risks or make do with advertising and tracking blockers. This is less easy in the mobile world, where apps are more rented than owned, and most are maintained by vendor-side services.

Thus, we've traded our freedom for the conveniences of centralization. The cure for that is decentralization: making the Net personal, like it promised to be in the first place—and still is, deep down.

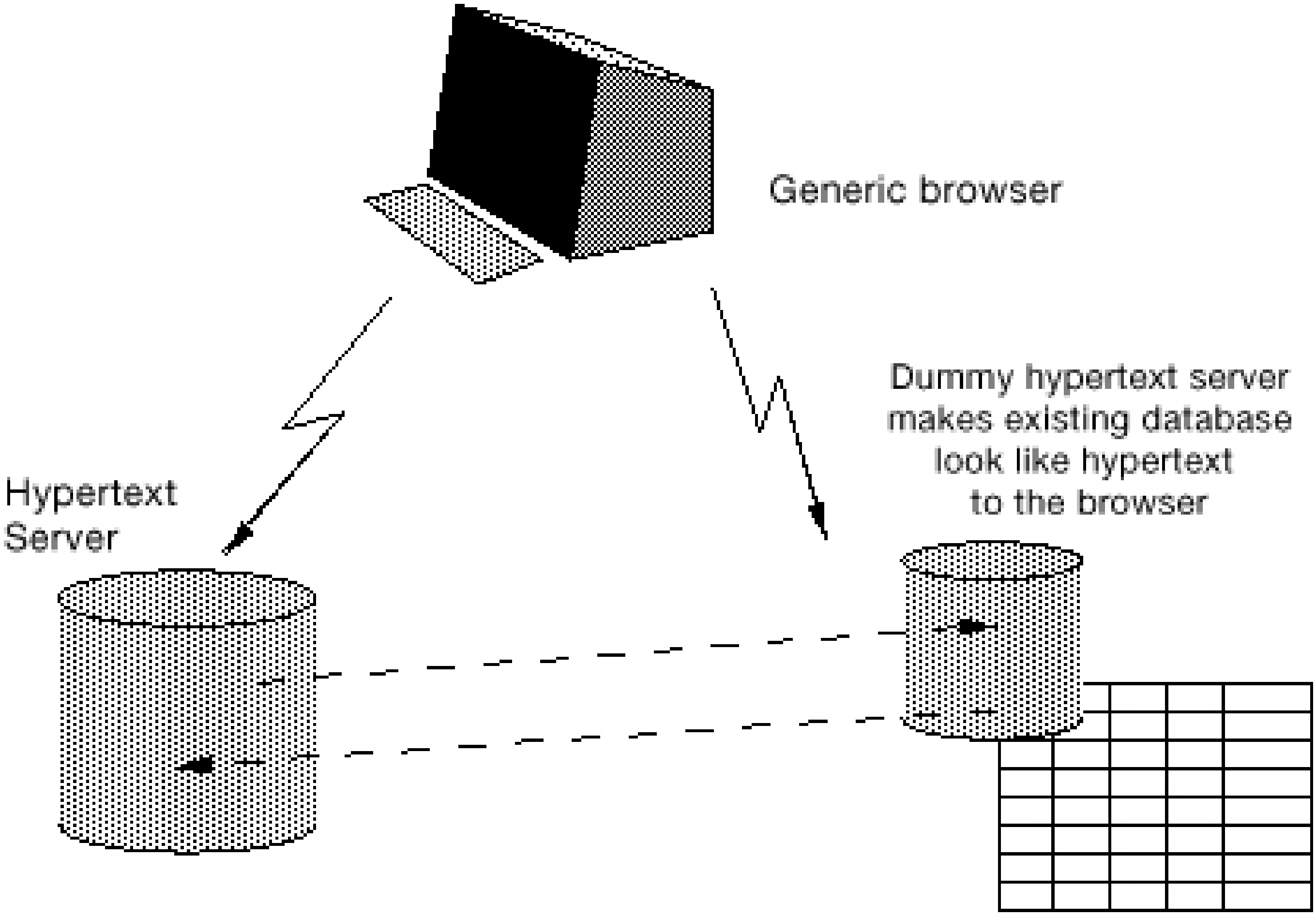

Generic browser
Dummy hypertext server makes existing database look like hypertext to the browser
Hypertext Server

Figure 1. Servers Generating a Hypertext Representation

It should help to remember that the Web is *polycentric* while the Net is *decentralized*. By polycentric, I mean server-based: every server is a center. So, even though Tim Berners-Lee wanted the Web to be what he called "a distributed hypertext system" for "universal linked information" (**http://www.w3.org/History/1989/proposal.html**), what he designed was servers "generating a hypertext representation", as shown in Figure 1.

Today this looks like your e-mail on a Google server—or your photos on Instagram or your tweets on Twitter. There's nothing wrong with any of those, just something missing: your independence and autonomy.

Meanwhile, the Net beneath the Web remains decentralized: a World of Ends (**http://worldofends.com**) in which every end is a functional distance of zero from every other end. "The end-to-end principle is the core architectural guideline of the Internet" says RFC 3724. Thus, even though the Internet is a "collection of networks", what collects them are the transcendent purposes of the Net's ends, which

Figure 2. It helps to think of the Net as the ground we walk and drive on, and the Web as clouds in the sky.

What Eben calls for is not merely to suffer the problems of centralization, but to solve them.

consist of you, me, Google and every other node.

If you want to grok the problems of centralization fully, and their threat to personal freedom, to innovation and to much else, watch, listen to or read Eben Moglen's lectures titled "Snowden and the Future" (**http://snowdenandthefuture.info**), given in November and December 2013 at Columbia University, where Eben has been teaching law for 26 years. The lectures are biblical in tone and carry great moral weight. For us in the Linux community, they are now in the canon.

What Eben calls for is not merely to suffer the problems of centralization, but to solve them. This requires separating the Net and the Web. For me, it helps to think of the Net as the ground we walk and drive on, and the Web as clouds in the sky, as I've illustrated with the photo in Figure 2.

There are many possibilities for decentralized solutions on the Net's ground, and I hope readers will remind us of some. Meanwhile, I'll volunteer a pair I've been watching lately. One is TeleHash, and the other is XDI.

TeleHash (**http://telehash.org**) is the brainchild of Jeremie Miller, father of Jabber and the XMPP protocol for instant messaging. Its slogan is "JSON + UDP + DHT = Freedom", and it is described as "a new wire protocol enabling applications to connect privately in a real-time and fully distributed manner, freeing them from relying on centralized data centers". The rest of the index page says:

> **What**
>
> It works by sending and receiving small encrypted bits of JSON (with optional binary payloads) via UDP using an efficient routing system based on Kademlia (**http://en.wikipedia.org/wiki/Kademlia**), a proven and popular Distributed Hash Table.
>
> **Demo**
>
> It's very much in the R&D stages yet, but check out hash-im (**https://github.com/quartzjer/hash-im**) for a simple demo.

> **Status**
> The current spec (**https://github.com/telehash/telehash.org/blob/master/protocol.md**) is implemented in a few languages (any help here would be great!), and prototype apps are being created to test it. Questions can be directed at Twitter (**https://twitter.com/jeremie**), or to Jeremie Miller directly.

XDI (**http://xdi.org**) is a mostly-baked standard. Its purpose is "to define a generalized, extensible service for sharing, linking, and synchronizing data over digital networks using structured data formats (such as JSON and XML) and XRIs (Extensible Resource Identifiers), a URI-compatible abstract identifier scheme defined by the OASIS XRI Technical Committee" (**https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xdi**). Wikipedia (at the moment) says (**http://en.wikipedia.org/wiki/XDI**):

> The main features of XDI are: the ability to link and nest RDF graphs to provide context; full addressability of all nodes in the graph at any level of context; representation of XDI operations as graph statements

Advertiser Index

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

ADVERTISER	URL	PAGE #
Drupalize.me	http://www.drupalize.me	93
Emac, Inc.	http://www.emacinc.com	11
EmperorLinux	http://www.emperorlinux.com	23
iXsystems	http://www.ixsystems.com	7
SCALE	https://www.socallinuxexpo.org/scale11x/	33
Silicon Mechanics	http://www.siliconmechanics.com	3
USENIX Conferences	https://www.usenix.org/conferences	2
WearablesDevCon	http://www.wearablesdevcon.com	15

so authorization can be built into the graph (a feature called XDI link contracts); standard serialization formats including JSON and XML; and a simple ontology language for defining shared semantics using XDI dictionary services.

XDI graphs can be serialized in a number of formats, including XML and JSON. Since XDI documents are already fully structured, XML adds very little value, so JSON is the preferred serialization format. The XDI protocol can be bound to multiple transport protocols. The XDI TC is defining bindings to HTTP and HTTPS, however it is also exploring bindings to XMPP and potentially directly to TCP/IP.

XDI provides a standardized portable authorization format called XDI link contracts (**http://en.wikipedia.org/wiki/Link_contract**). Link contracts are themselves XDI documents (which may be contained in other XDI documents) that enable control over the authority, security, privacy, and rights of shared data to be expressed in a standard machine-readable format and understood by any XDI endpoint.

This approach to a globally distributed data sharing network models the real-world mechanism of social contracts (**http://en.wikipedia.org/wiki/Social_contract**), and legal contracts that bind civilized people and organizations in the real world today. Thus, XDI can be a key enabler of the Social Web (**http://en.wikipedia.org/wiki/Social_Web**). It has also been cited as a mechanism to support a new legal concept, Virtual Rights (**http://www.virtualrights.org**), which are based on a new legal entity, the "virtual identity", and a new fundamental right: "to have or not to have a virtual identity".

It's early for both of these. But I know in both cases the mentality of the developers is on the ground of the Net and not lost in the clouds of the Web. We'll need a lot more of that before we all get our freedom back.■

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.