

CoffeeScript | Git | Facebook | LSB | Beowulf | Unison

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux World

AUGUST 2011 | ISSUE 208 | www.linuxjournal.com

**BUILD A
BEOWULF
HPC SYSTEM**
with the
FedoraLiveCD
Project

**LINUX
STANDARD
BASE**
and Its Role
in the
Community

HOW TO PLAN YOUR OWN LINUX EVENT

WITH GARETH GREENAWAY OF SCALE

Writing Facebook
Applications

Understanding
and Using Git

Use
Podcasts
to Share
Your Linux
Knowledge

Add Nagios
Notifications
to Your Screen
Window

Make
JavaScript
Programming
Easy with
CoffeeScript

HOW-TO:

**UNISON THE FILE
SYNCHRONIZATION PROGRAM**



10 Gig On Board

Blazing Fast, Embedded 10Gb Ethernet

10G Rackmount Servers in the iX-Neutron server line feature the Intel® Xeon® Processor 5600/5500 Series, and come with 10GbE networking integrated onto the motherboard. This eliminates the need to purchase an additional expansion card, and leaves the existing PCI-E slots available for other expansion devices, such as RAID controllers, video cards, and SAS controllers.

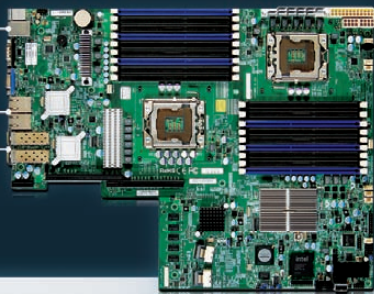
For more information on the iX-1204-10G, or to request a quote, visit: <http://www.iXsystems.com/neutron>

 **30% cost savings/port over equivalent Dual-Port 10 GB PCI Express add-on card solution**

IPMI NIC

GigE NICS

10GbE NICS



10Gb Ethernet Adapters



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

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



KEY FEATURES:

- Supports Dual 64-Bit Six-Core, Quad-Core or Dual-Core, Intel® Xeon® Processor 5600/5500 Series
- 1U Form Factor with 4 Hot-Swap SAS/ SATA 3.5" Drive Bays
- Intel® 5520 chipset with QuickPath Interconnect (QPI)
- Up to 192GB DDR3 1333/1066/800 SDRAM ECC Registered Memory (18 DIMM Slots)
- 2 (x8) PCI-E 2.0 slots + 1 (x4) PCI-E 2.0 (in x8 slot -Low-Profile - 5.5" depth)
- Dual Port Intel® 82599EB 10 Gigabit SFP+ - Dual Port Intel® 82576 Gigabit Ethernet Controller
- Matrox G200eW Graphics
- Remote Management - IPMI 2.0 + IP-KVM with Dedicated LAN
- Slim DVD
- 700W/750W Redundant AC-DC 93%+ High-Efficiency Power Supply



**Powerful.
Intelligent.**

1&1 DUAL HOSTING

THE NEW STANDARD IN WEB HOSTING



No other web host offers more expertise, know-how and quality service than 1&1.

- ✓ **Double Security:**
Your website is simultaneously hosted in 2 locations in our high tech data center!
- ✓ **High-speed Global Network:**
210 GBit/s Connectivity
- ✓ **Environmentally Responsible:**
100% Renewable Energy
- ✓ **Solid Technical Foundation:**
1,000 In-house Developers

SUMMER SPECIAL: 1&1 DUAL ADVANCED PACKAGE

1 YEAR FREE!*

- 2 FREE Domains
- FREE Private Domain Registration
- 500 E-mail Accounts
- 50 FTP Accounts
- UNLIMITED Traffic
- DNS Management
- 1&1 SiteAnalytics
- 1&1 WebsiteBuilder
- ASP, .NET, AJAX, LINQ, PHP5, Perl, SSI
- 5 Microsoft® SQL Databases
- 24/7 Toll-free Customer Support

Need more domains?

.COM with FREE Private Registration just **\$4.99/first year.***



1-877-GO-1AND1

www.1and1.com



1-855-CA-1AND1

www.1and1.ca



*24 month minimum contract term required for Dual Advanced offer. Set-up fee and other terms and conditions may apply. .com offer valid first year only. After first year, standard pricing applies. Visit www.1and1.com for full promotional offer details. Program and pricing specifications and availability subject to change without notice. 1&1 and the 1&1 logo are trademarks of 1&1 Internet AG, all other trademarks are the property of their respective owners. © 2011 1&1 Internet, Inc. All rights reserved.

CONTENTS

AUGUST 2011
Issue 208

FEATURES

34 A CONFERENCE PRIMER WITH GARETH GREENAWAY

Geek Wrangling 101, with an expert.

Shawn Powers

42 FACEBOOK APPLICATION DEVELOPMENT

Port your Web app to Facebook.

Mike Diehl

48 LINUX STANDARD BASE: STATE OF AFFAIRS

LSB and what it means for the Linux community.

Jeff Licquia, Stew Benedict and Vladimir Rubanov

54 GIT

A guide to what many consider the best version control tool ever created.

Henry Van Styn

ON THE COVER

- How to Plan Your Own Linux Event with Gareth Greenaway of SCALE, p. 34
- Writing Facebook Applications, p. 42
- Understanding and Using Git, p. 54
- Build a Beowulf HPC System with the FedoraLiveCD Project, p. 64
- Linux Standard Base and Its Role in the Community, p. 48
- Use Podcasts to Share Your Linux Knowledge, p. 60
- Add Nagios Notifications to Your Screen Window, p. 24
- Make JavaScript Programming Easy with CoffeeScript, p. 16
- How-To: Unison the File Synchronization Program, p. 73

They say work smarter, not harder. They must be using our processor.



Sponsors of Tomorrow.™

The next generation of intelligent server processors

The Intel® Xeon® processor 5600 series automatically regulates power consumption to combine industry-leading energy efficiency with intelligent performance that adapts to your workload. Check out the new intelligent features of the Xeon® 5600 at intel.com/itcenter.



SEE WHAT
INTELLIGENCE
CAN DO



Servers from iXsystems feature the Intel® Xeon® processor 5600 series.



Enterprise Servers
for Open Source
www.iXsystems.com
1-855-GREP-4-IX

Intel is not responsible for and has not verified any statements or computer system product-specific claims contained herein.

iX2216-10G

The **iX2216-10G** features dual on-board Intel® 82599EB 10 Gigabit SFP+ Ports, dual on-board Intel® 82576 Gigabit Ports, and 18 DIMM slots supporting up to 192GB of DDR3 ECC Registered memory. Ideal for HPC, Data Center, Virtualization, Clustering, and Cloud Computing applications.

- Dual Intel® Xeon® 5600 Series Processors
- 2U Form Factor with sixteen 2.5" SAS/SATA Hot-Swap Drive Bays
- On-Board Dual Port Intel® 82599EB 10 Gigabit SFP+



iX1204-10G

- Dual Intel® Xeon® 5600 Series Processors
- 1U Form Factor with 4 Hot-Swap SAS/SATA Drive Bays
- On-Board Dual Port Intel® 82599EB 10 Gigabit SFP+

CONTENTS

AUGUST 2011

Issue 208

COLUMNS

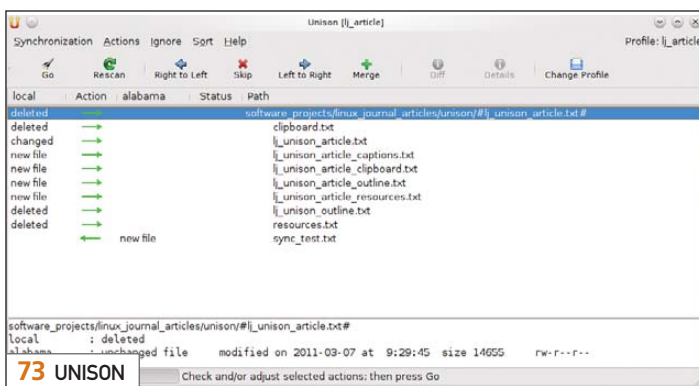
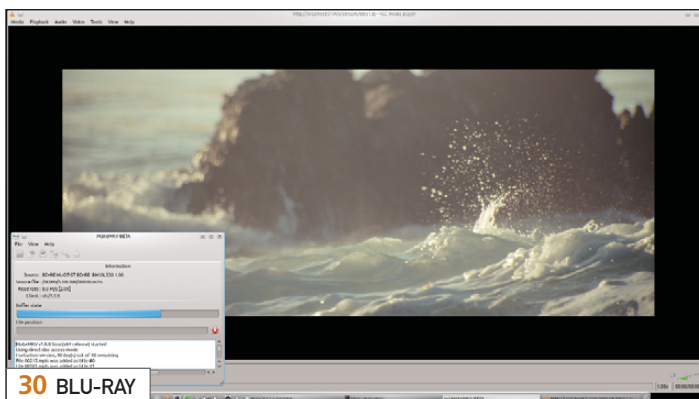
- 16** Reuven M. Lerner's At the Forge
Introducing CoffeeScript
- 22** Dave Taylor's Work the Shell
Calculating Day of the Week
- 24** Kyle Rankin's Hack and /
Nagging Notifications
- 77** Kyle Rankin and Bill Childers'
Tales from the Server Room
It's Always DNS's Fault!
- 80** Doc Searls' EOF
First Brazil, Then the World

INDEPTH

- 60** Podcasting
Do you know something about Linux? A podcast is an excellent way to share your knowledge with others.
Charles Olsen
- 64** How to Build a Beowulf HPC System Using the FedoraLiveCD Project
Build a Red Hat-based Beowulf Cluster using a kickstart file and tools from the FedoraLiveCD project.
Howard Powell
- 69** Radio Dramas in Linux
Tips and tricks for creating radio dramas with open-source software.
Dan Sawyer
- 73** Unison, Having It Both Ways
The file synchronization tool that supports bidirectional updates of files and directories.
Adrian Klaver

IN EVERY ISSUE

- 8** Current_Issue.tar.gz
- 10** Letters
- 12** UPFRONT
- 28** New Products
- 30** New Projects
- 65** Advertisers Index
- 79** Marketplace



More TFLOPS, Fewer WATTS

Microway delivers the fastest and greenest floating point throughput in history

Enhanced GPU Computing with Tesla Fermi

- ▶ 480 Core NVIDIA® Tesla™ Fermi GPUs deliver 1.2 TFLOP single precision & 600 GFLOP double precision performance!
- ▶ New Tesla C2050 adds 3GB ECC protected memory
- ▶ New Tesla C2070 adds 6GB ECC protected memory
- ▶ Tesla Pre-Configured Clusters with S2070 4 GPU servers
- ▶ WhisperStation - PSC with up to 4 Fermi GPUs
- ▶ OctoPuter™ with up to 8 Fermi GPUs and 144GB memory

New Processors

- ▶ 12 Core AMD Opterons with quad channel DDR3 memory
- ▶ 8 Core Intel Xeons with quad channel DDR3 memory
- ▶ Superior bandwidth with faster, wider CPU memory busses
- ▶ Increased efficiency for memory-bound floating point algorithms

Configure your next Cluster today!

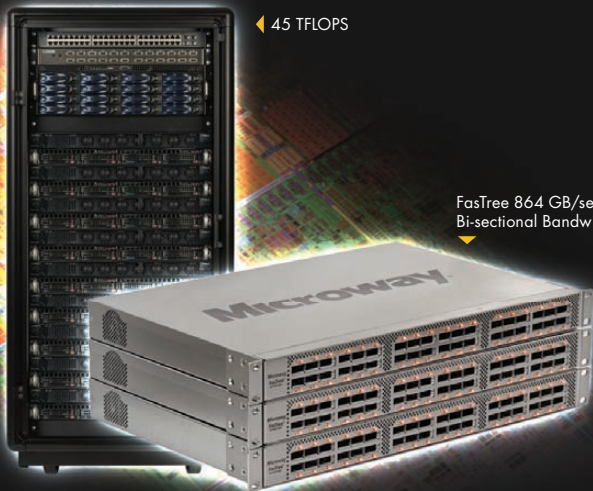
www.microway.com/quickquote
508-746-7341



2.5 TFLOPS

10 TFLOPS

5 TFLOPS



4.5 TFLOPS

FasTree 864 GB/sec
Bi-sectional Bandwidth

FasTree™ QDR InfiniBand Switches and HCAs

- ▶ 36 Port, 40 Gb/s, Low Cost Fabrics
- ▶ Compact, Scalable, Modular Architecture
- ▶ Ideal for Building Expandable Clusters and Fabrics
- ▶ MPI Link-Checker™ and InfiniScope™ Network Diagnostics

Achieve the Optimal Fabric Design for your Specific MPI Application with ProSim™ Fabric Simulator

Now you can observe the real time communication coherency of your algorithms. Use this information to evaluate whether your codes have the potential to suffer from congestion. Feeding observed data into our IB fabric queuing-theory simulator lets you examine latency and bi-sectional bandwidth tradeoffs in fabric topologies.



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

Microway
Technology you can count on™

LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

**DIGITAL EDITION
NOW AVAILABLE!**

Read it first

Get the latest issue before it
hits the newsstand

Keyword searchable

Find a topic or name
in seconds

Paperless archives

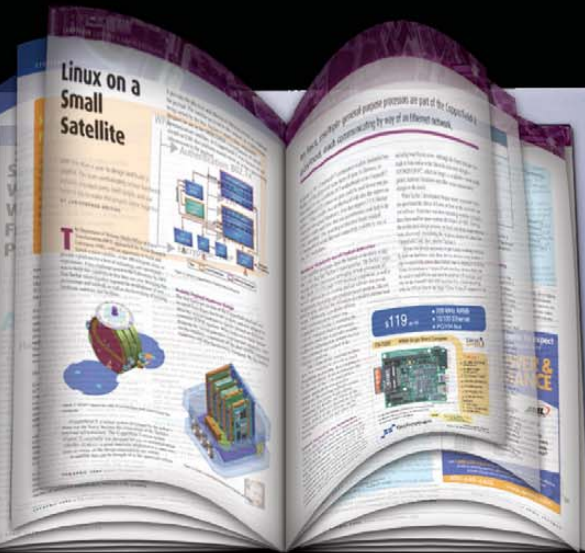
Download to your computer for
convenient offline reading

Same great magazine

Read each issue in
high-quality PDF

Try a Sample Issue!

www.linuxjournal.com/DLISSUE



LINUX JOURNAL

Executive Editor Jill Franklin
jill@linuxjournal.com

Senior Editor Doc Searls
doc@linuxjournal.com

Associate Editor Shawn Powers
shawn@linuxjournal.com

Art Director Garrick Antikajian
garrick@linuxjournal.com

Products Editor James Gray
newproducts@linuxjournal.com

Editor Emeritus Don Marti
dmarti@linuxjournal.com

Technical Editor Michael Baxter
mab@cruzio.com

Senior Columnist Reuven Lerner
reuven@lerner.co.il

Security Editor Mick Bauer
mick@visi.com

Hack Editor Kyle Rankin
lj@greenfly.net

Virtual Editor Bill Childers
bill.childers@linuxjournal.com

Contributing Editors

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

Proofreader Geri Gale

Publisher Carlie Fairchild
publisher@linuxjournal.com

General Manager Rebecca Cassity
rebecca@linuxjournal.com

Senior Sales Manager Joseph Krack
joseph@linuxjournal.com

Associate Publisher Mark Irgang
mark@linuxjournal.com

Webmistress Katherine Druckman
webmistress@linuxjournal.com

Accountant Candy Beauchamp
acct@linuxjournal.com

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

Editorial Advisory Panel

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

Advertising

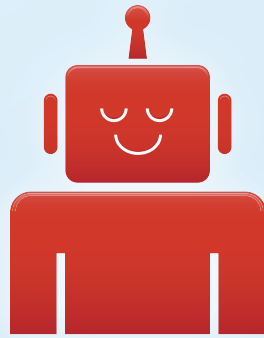
E-MAIL: ads@linuxjournal.com
URL: www.linuxjournal.com/advertising
PHONE: +1 713-344-1956 ext. 2

Subscriptions

E-MAIL: subs@linuxjournal.com
URL: www.linuxjournal.com/subscribe
PHONE: +1 818-487-2089
FAX: +1 818-487-4550
TOLL-FREE: 1-888-66-LINUX
MAIL: PO Box 16476, North Hollywood, CA 91615-9911 USA
Please allow 4-6 weeks for processing address changes and orders
PRINTED IN USA

LINUX is a registered trademark of Linus Torvalds.





Lullabot™

Learn Drupal & jQuery

FROM THE COMFORT OF
YOUR LIVING ROOM



The Lullabot Learning Series includes everything you need to become a Drupal & jQuery expert from the comfort of your living room! The videos are available in both DVD format and high-definition video download.

Purchase the videos at <http://store.lullabot.com>



SHAWN POWERS

I'll Be in My Hole

Tech folks tend to get a bad rap as basement-dwelling creatures unable to function in polite society. And really, that's true for only 80% of us. Of course, I'm teasing, but it's hard to deny Linux users and open-source enthusiasts tend to belong to unique communities. We thought it might be nice to focus on our Linux community this month, and so the articles you're about to read should feel as cozy as a Hobbit hole.

Reuven M. Lerner knows his audience and introduces a new programming language called CoffeeScript. The coffee-/java-/bean-naming convention is nothing new, but CoffeeScript makes JavaScripting fun and easy. If you've ever been frustrated with JavaScript, you'll want to read Reuven's article. Following that, Dave Taylor continues his series on determining the day of the week. If you truly do live in a basement or Hobbit hole, you might need your computer to determine what day it is, as sunlight might be a rare sight.

Kyle Rankin doesn't help the geek stereotype when everything he does is within a single, dark console window. Kyle uses a screen session to connect to his favorite programs (all command-line) from anywhere. This month, he shows how to get Nagios to send notifications to a screen session instead of e-mail or SMS messages. Bill Childers is back with us this month as well, and he and Kyle continue their new column, Tales from the Server Room. If you're a sysadmin, you'll probably relate to their misery and maybe learn something along the way.

No community is a community without gatherings. Granted, in our world, those gatherings often are in IRC rooms or in on-line forums. We do have our meatspace meetups, however, and if you've never been to a LinuxFest, I highly recommend attending one. This month, I interview Gareth Greenaway from the Southern California Linux Expo (SCALE) and ask him what it takes to host a Linux conference. Gareth is easy to talk to, and he has some practical advice for anyone interested in attending a LinuxFest or even hosting their own. To contrast the idea of getting together in person, Mike Diehl talks about Facebook application development. Whether you're interested in integrating services or creating a new zombie-pony-themed *Farmville*, you'll want to read Mike's article.

Some of the core values of our Linux community

are standards and collaboration. The Linux Standards Base attempts to keep the various flavors of Linux compatible with each other. Sometimes that works better than others, but we have the current "State of Affairs" regarding the LSB for you this month. Jeff Licquia, Stew Benedict and Vladimir Rubanov provide an overview of the LSB and talk about how to be compliant. Henry Van Styn follows them up with an article on Git. Once a group of people start to collaborate on software, revision control is vital. Henry discusses Git and how its distributed model makes it so powerful.

As magazine publishers, we surely understand the importance of communication when it comes to community. With the Linux world, however, we communicate in multiple ways. One of those methods, which is near and dear to my heart, is podcasting. Kyle Rankin and I do the *Linux Journal* Insider podcast every month, and if we can do it, you can do it. Charles Olsen walks through the process of creating a podcast from the planning stages all the way to the publishing stages. Content, of course, is up to you. Dan Sawyer takes a slightly different angle, and instead of podcasting, he shows how to edit radio dramas. If you tend to make stuff up in your podcast, you already may be walking the line between fact and fiction, so we wanted to make sure you were prepared either way.

What group of geeks would be worth their salt without allowing their computers to form a community as well? Whether you want multiple computers to share a filesystem with Unison (Adrian Klaver shows how), or if you want your systems to form a cohesive community of their own in a Beowulf cluster (Howard Powell describes how to do it with Fedora Live CDs), this issue has you covered. We've also got the full lineup you expect every month. UpFront articles, new product announcements—lots of goodies to soothe your geek itch. And, of course, if the thought of being a part of the Linux community offends you, well, feel free to read this magazine by yourself. We won't tell. ■

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](https://freenode.net) IRC channel on Freenode.net.

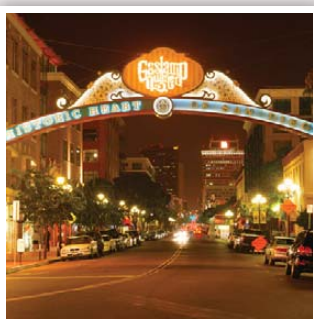
hostingcon 2011

AUGUST 8-10 | SAN DIEGO, CALIFORNIA

Enter discount code
LINUXJOURNAL2011
to save up to \$60!

Learn. Network. Grow.

Register Now and Save with Early Bird Rates: www.hostingcon.com



Join us at HostingCon 2011, the premier conference and trade show for the hosted services industry. The best and brightest from the industry will be in attendance to learn about the latest ideas and technology affecting their businesses. Whether you are showcasing your new services, or you want to network with your business partners, you'll find a fun and educational atmosphere in sunny San Diego.

Platinum Sponsors:

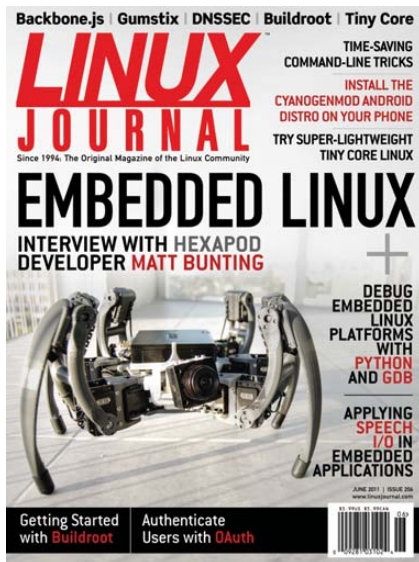
An iNET Interactive Event

Microsoft



Parallels
Optimized Computing





May 2011 Cover

My manga reading/drawing daughter saw the cover of the May 2011 issue of *LJ* and asked me why there was manga on it. I told her, “Linux is just cool like that!” Then, she asked if there was an article about manga in the magazine. I told her about past issues covering Blender, GIMP and so on, which prompted a discussion about the great open-source software she uses every day.

Thanks for the great cover!

--
Eric Miller

That’s awesome! It’s great when things like cover art can make Linux relevant for people who otherwise might not care.—Ed.

Utility I Wrote (Open Source, of Course)

I’m a longtime reader and fan of *LJ*, and I thought you might be interested in a tool I wrote for system admins and developers named chip. It’s basically a logfile multiplexer and monitor: <https://github.com/katzgrau/chip>.

I remember you wrote about swatch (www.linuxjournal.com/article/4776), and chip basically does what swatch does, but additionally, it can do it on multiple remote files. That becomes super handy when you want to monitor multiple

production logs on hosts behind a load balancer. You don’t have to set up handlers like swatch, and you also can just use chip to see of all of your logs like a pimped version of tail.

I’ve used it here at Yahoo pretty frequently during the past month for deployments and bug investigations.

Here’s a blog post, if you’re interested: codefury.net/2011/04/video-chip-a-log-file-monitor-multiplexer.

--
Kenny Katzgrau

Thanks Kenny. Sharing tools is one of the things that makes the Open Source community so great. I appreciate you sharing with us!—Ed.

Receiving Digital TV Signals, No Longer Analog Signals

Regarding the new digital TV world: I am receiving three stations via digital signal on my digital-to-analog-converter TV, with my outdoor antenna. Two other stations, in the same area, hardly come in at all. The first group transmits at 1MW power; the other group transmits at only 0.25MW power. My signal strength meter seems consistent with these numbers (above). My question is, why is hardly anything written on the transmit power of the sending stations? Why do some “stations” get to transmit at 1MW, while others can transmit only at one-fourth of this? All the complaints about digital TV signals abound, but nobody ever seems to mention the power used by the sending stations to send their signals. I believe this has to be the main reason for fair-to-bad TV receptions of digital signals, not all the malarkey you read—for example, airplanes, leaves, slight temperature changes and so on. I get better reception in cold, cloudy, rainy weather than on “nice days”! And, I live more than 60 miles from the transmitters of these five stations. Who decides who gets what MWs?

--
R. E. Mackey

I didn’t know the answer to this, so I called a friend of mine who manages local

low-power television stations. His understanding of the situation is that the FCC dictates to broadcasters the power at which they can transmit based on several technical factors. Those factors include things like terrain, desired broadcast coverage, distance from other broadcasts running on close frequencies and so on. In fact, the FCC sometimes will dictate that the signal can’t be broadcasted omnidirectional. I’m close to the Canadian border, and some stations are not allowed to broadcast toward Canada.

Before researching, my guess was that broadcast power was based on fees paid to the FCC. As it turns out, that’s not the case. The fees do vary widely based on location, however, and usually frequencies are auctioned off to the highest bidder.

Finally, you might want to check out www.antennaweb.org. It offers some great tools to help you get the best possible reception.—Ed.

777 Permissions

While reading the article titled “Numeric Relativity” by Joey Bernard on page 14 of the May 2011 issue, I was a little disappointed to notice that the instructions for installing the Einstein Toolkit contain a `chmod 777`. Being a systems administrator, this made my eye twitch a bit. The world write bit should be used only under rare circumstances. In fact, I would argue that it should be used only for common storage locations, such as `/tmp`, and even then, it should be accompanied by the sticky bit (that is, `1777`). In most cases, `700` or `755` permissions are most appropriate.

--
Elliott Forney

Sky Survey Using Linux

There’s an incredible survey of the entire night sky, using a laptop powered by Fedora Linux, and I thought *Linux Journal* readers may be interested: skysurvey.org.

--
Jim Gagnon

How cool! Thanks for the link.—Shawn, the space nut.

MAGAZINE

PRINT SUBSCRIPTIONS: Renewing your subscription, changing your address, paying your invoice, viewing your account details or other subscription inquiries can instantly be done on-line, www.linuxjournal.com/subs. Alternatively, within the U.S. and Canada, you may call us toll-free 1-888-66-LINUX (54689), or internationally +1-818-487-2089. E-mail us at subs@linuxjournal.com or reach us via postal mail, Linux Journal, PO Box 16476, North Hollywood, CA 91615-9911 USA. Please remember to include your complete name and address when contacting us.

DIGITAL SUBSCRIPTIONS: Digital subscriptions of *Linux Journal* are now available and delivered as PDFs anywhere in the world for one low cost. Visit www.linuxjournal.com/digital for more information or use the contact information above for any digital magazine customer service inquiries.

LETTERS TO THE EDITOR: We welcome your letters and encourage you to submit them at 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, www.linuxjournal.com/author.

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, www.linuxjournal.com/advertising. Contact us directly for further information, ads@linuxjournal.com or +1 713-344-1956 ext. 2.

ON-LINE

WEB SITE: Read exclusive on-line-only content on *Linux Journal's* Web site, www.linuxjournal.com. Also, select articles from the print magazine are available on-line. Magazine subscribers, digital or print, receive full access to issue archives; please contact Customer Service for further information, subs@linuxjournal.com.

FREE e-NEWSLETTERS: Each week, *Linux Journal* editors will tell you what's hot in the world of Linux. Receive late-breaking news, technical tips and tricks, and links to in-depth stories featured on www.linuxjournal.com. Subscribe for free today, www.linuxjournal.com/enewsletters.

Linux to the Rescue!

Situation: 1998-vintage Compaq small-form-factor desktop (Pentium-II 400MHz) running Fedora Core 5 as a home firewall/DHCP/DNS server. After a power failure, it refuses even to POST. The lights come on, but nobody's home. I'd buy a used Dell SFF system to replace it, but I'm looking at several hours of work to install and configure with Fedora 14. There goes my evening plans, sigh.

Brilliant idea: move the old hard disk to the new system and see what happens, but I'm quite skeptical based on my experiences with Windows and changing out hardware.

Result: it boots! It works! Nary a hiccup. I'm back in business in 15 minutes. I got my evening back, thanks to Linux.

--
Jim Garrison

I feel obliged to recommend a system upgrade, as FC5 reached end of life back in 2007. I also feel obliged to note how awesome it is that a system running FC5 still is running well! And, you gotta love Linux. It makes most hardware, "just work".—Ed.

It Starts with the Kids

A few years ago, I read a book called *The Culture Code* by Clotire Rapaille. One concept that he explained intrigued me. He described how in the 1970s, Japan's consumption of coffee was 0%. If you were to go to a restaurant and ask for coffee, it was not on the menu. How do you impose a new beverage into a nation that does not have coffee in its vocabulary? You start by making candies infused with coffee flavor. Who eats these candies? Children. This is exactly what was done. They whet a generation's appetite with caffeine, and in around the year 2000, coffee accounted for billions and billions of Japan's GDP. What is my point? Next paragraph.

Last month, I was honored with the opportunity to expose home-schooling parents to a presentation on "Free Software to Augment your Child's Learning". In this presentation, I attempted to cover various open-source projects as well as operating systems that are freely available to everyone. If we start to infuse the idea of Linux and open source in the conscience of the generation coming after us, maybe, just maybe, Linux would be synonymously in the minds of people when they say "PC", instead of the other OSes available. I've

started with the home-schooling community in Ontario, Canada. What "Baby Penguins" are you going to target?

--
Dean Anderson

You're totally preaching to the choir here, as I've been pushing Linux in education for more than a decade. (I am the Technology Director for a school district in Michigan.) I didn't know about Japan and coffee, but it's a great story and encouraging for those of us in the trenches. I do hope the next generation "gets it" when it comes to the advantages Linux and open source can provide. Great job!—Ed.

Dropbox

I received the June 2011 issue of *LJ* today, and as usual, I proceeded to read it cover to cover on the day it arrived. I immediately noticed your article on Dropbox, as it is an integral part of my daily toolkit.

For my part, I find one of the key uses of Dropbox is to turn regular desktop applications into Web-synchronized applications. As many *LJ* readers do, I work across different laptops and desktops, but I would like to keep some key apps immediately available, with the same data, no matter where I am working, such as time-tracking and my to-do list. There are many Web-based solutions for this, but I found a couple client applications I prefer to use for their feature sets.

For time-tracking, I use Klok 2 (www.getklok.com), and for task lists, I use Task Coach (www.taskcoach.org). By moving the data files for these applications into Dropbox, I immediately have my time-tracking and to-do lists synchronized across all my working environments. Thanks Dropbox for the synchronizing feature, and thanks to the *Linux Journal* team for a great publication.

--
Randy Earl

I use Dropbox in a similar way as well! I find it works great to sync Pidgin preferences, so I don't need to enter all my account information on each computer I use. I also keep my desktop background photos (all space photos from NASA) in sync, so they are available on every computer. Syncing is awesome technology, and I really look forward to open-source tools maturing to the point that they rival Dropbox's abilities.—Ed.

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

This edition of diff -u is dedicated to **David Brownell**, a kernel hacker who gave a lot his time and creativity to Linux, and a lot of help and encouragement to many open-source developers. Rest in peace, David. May your name linger long in the source tree.

Pekka Enberg has been working on a helpful **virtualization tool** to make it relatively straightforward to boot a virtual system on a running Linux box. There are various virtualization tools available, but Pekka says this one is intended to be a clean, lightweight alternative that anyone could use.

Tony Ibbs has released **KBUS**, available at kbus-messaging.org. It's a message-passing framework, intended to support software running on embedded systems. The main idea is that it will be very reliable: a process that dies before it can send a needed message still will have a message sent by KBUS on its behalf; messages flying through the system will all arrive in a predictable order, making their significance easier to interpret; and messages that are sent by one process are as nearly as possible guaranteed to arrive at their destination.

Dan Rosenberg kicked off a new drive to increase Linux security, when he posted some patches to hide the `/proc/slabinfo` file from regular users. It turned out that his approach was too much of a blunt instrument, with too little gain, for certain folks' taste. But in the process of debating its various merits, a bunch of folks, including **Linus Torvalds**, dove into the memory allocation code, trying to figure out ways of preventing various security exploits that had been seen in the recent past.

David Johnston alerted the kernel folks to the idea that some of his code had been included in the kernel, without attributing him as the author and without having been licensed by him to be included in the source tree. It appeared to be a clear **copyright violation**. David actually said he'd be pleased and honored to have his code in the kernel; he just wanted to make sure the attribution was correct, and the license issues sorted out. So a few folks worked on finding out how the code had gotten into the kernel in the first place and how to fix the attribution. These kinds of debates always are fascinating, because they put everyone in the awkward position of already having done something wrong and trying to figure out the best legal and ethical way to back out of it again.

A bit of political wrangling: **David Brown** tried to remove **Daniel Walker** and **Bryan Huntsman** from the **MAINTAINERS file**, as the official maintainers of **ARM/Qualcomm MSM** machine support. Bryan was fine with this, but Daniel had not given his approval. It turned out that Bryan and David were both industry folks, and Daniel wanted to remain a maintainer, partly in order to make sure they "did right by the community". As Linus Torvalds pointed out during the discussion, there was no real reason for any good maintainers to be supplanted if they didn't want to step down. Their contribution could be only positive, and so the argument by the industry folks amounted to an exclusivity that Linus said he found distasteful. And as **Pavel Machek** pointed out, the fact that David had thought it would be okay to do this in the first place was a good enough argument that he wasn't yet ready to be a sole maintainer.

—ZACK BROWN



XBMC, Now with Less XB!

Xbox Media Center (XBMC) is one of those projects whose name makes less and less sense as time goes on. Sure, people still are using XBMC on an actual Microsoft Xbox, but for the most part, XBMC now is run on computers. In fact, recent versions of XBMC installed on an ION-based nettop makes just about the perfect media center. Version 10 (Dharma) introduced a fancy plugin system that allows XBMC to be extended beyond its built-in media-playing abilities. The next version, currently in development, will focus partially on recording as well as playback.

When it comes to performance, it's hard to beat XBMC. It's faster and more responsive than a Boxee Box, has local media playback unlike the Roku, and is open source, unlike Microsoft's media center options. It does currently lack in premium on-line streaming, but that's largely because the live version is based on Linux. It's a trade-off I'm willing to make. (I actually keep a Roku for that purpose and use XBMC for everything else.)

Check out the latest features and download a copy for your operating system at xbmc.org.

—SHAWN POWERS

NON-LINUX FOSS



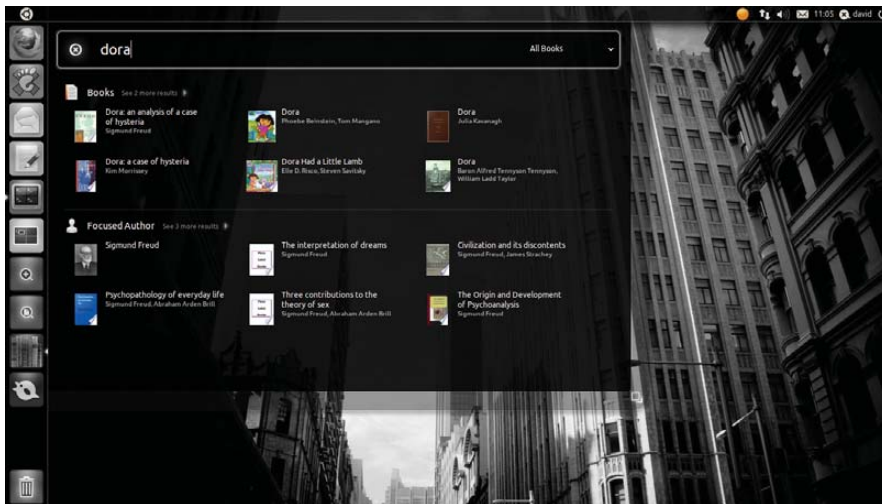
ports Gmail-style archiving, tabs and better searching, and it's completely extensible with add-ons.

Oh, and if you're stuck with Windows or OS X, no problem! Thunderbird always has been a cross-platform e-mail solution, and it still is. It makes the transition from one platform to another easier for users, and if you don't like your e-mail to be kept on some Web company's servers, it might be the ideal e-mail solution for you. Check it out at getthunderbird.com.



—SHAWN POWERS

BOOKS LENS



This screenshot, from the project page, shows the search results for "Dora".

If you are an Ubuntu user and a fan of the new Unity interface, you might be interested in a new lens in development by David Callé. The Books Lens provides a real-time search interface for e-books. It currently interfaces with Google Books, Project Gutenberg and Forgotten Books. By the time you read this, that list probably will have grown.

The Books Lens instantly finds metadata on your search, including book covers, and gives you direct links to the books themselves. For us book nerds, it's quite a handy little tool. Check it out yourself at <https://launchpad.net/unity-books-lens>, or install the PPA with:

```
sudo apt-add-repository ppa:davidc3/books-lens
```

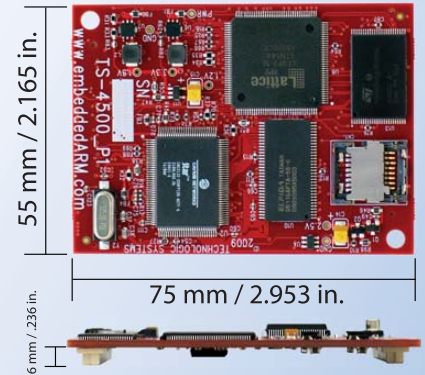
Then, install the package unity-books-lens.

—SHAWN POWERS

TS-SOCKET Macrocontrollers


Jump Start Your Embedded Design

Series starts at **\$92** qty100



TS-SOCKET Macrocontrollers are CPU core modules that securely connect to a baseboard using the TS-SOCKET dual connector standard with common pin-out interface. COTS baseboards are available or design your own baseboard for a custom solution with drastically reduced design time and complexity. Start your embedded system around a TS-SOCKET Macrocontroller to reduce your overall project risk and accelerate time to market.

- TS-4200: Atmel ARM9 with super low power
- TS-4300: Cavium ARM11 with dual 600 MHz and FPU
- TS-4500: Cavium ARM9 at very low cost
- TS-4700: Marvell PXA168 with video and 1.2 GHz CPU
- TS-4800: Freescale iMX515 with video and 800 MHz CPU
- Several COTS baseboards for evaluation & development

 Design your solution with one of our engineers

- Over 25 years in business
- Never discontinued a product
- Engineers on Tech Support
- Open Source Vision
- Custom baseboards w/ excellent pricing and turn-around time
- Most products ship next day



We use our stuff.

visit our TS-7800 powered website at
www.embeddedARM.com
(480) 837-5200

To CFD, or Not to CFD?

One area that chews up a lot of cycles on machines around the world is CFD. What is CFD? CFD is short for Computational Fluid Dynamics. The general idea is to model the flow of gases and liquids (or fluids) as they interact with solid surfaces. This type of modeling is used in designing aircraft, automobiles, submarines and fan blades—basically, anything that travels through water or air. As you increase the complexity of the surfaces, or increase the complexity of the flow (such as going from subsonic to supersonic), the amount of computer time needed to model it goes up. One of the big packages available to do CFD is the suite of programs made by Ansys. Several groups here at my university use it. But, there is an open-source option available, OpenFOAM (www.openfoam.com). This month, I describe what you can accomplish with OpenFOAM. The OpenFOAM Project includes binary packages as deb files or RPM files. You also can download a source code package of files or even download directly from the Git repository.

OpenFOAM (Open Source Field Operation and Manipulation) basically is a set of C++ libraries that are used in the various processing steps. OpenFOAM, just like most other CFD packages, breaks down the work to be done into three separate steps. The first step is called pre-processing. In pre-processing, you define the problem you are trying to model. This involves defining the boundary conditions given by the solid objects in your model. You also describe the characteristics of the fluid you are trying to model, including viscosity, density and any other properties that are important for your model. The next step is called the solver step. This is where you actually solve the equations that describe your model. The third step is called post-processing. This is where you take a look at the results and visualize them so that you can see what happens in your model. An obvious consequence of this breakdown is that most of the computational work takes place during the solver phase. The pre- and post-processing steps usually can be done on a desktop, while the solver step easily can use up 50 or 100 processors. OpenFOAM includes several pre- and post-processing utilities, along with several solvers. But the real power comes from the fact that, because

OpenFOAM is library-based, you can build your own utilities or solvers by using OpenFOAM as a base. This is very useful in a research environment where you may be trying something no one else ever has.

A model in OpenFOAM is called a case. Cases are stored as a set of files within a case directory. Many other CFD packages use a single file instead. A directory allows you to separate the data from the control parameters from the properties. Case files can be edited using any text editor, such as emacs or vi. Pre-processing involves creating all of the files required for the case you are investigating.

The first step is mesh generation. Your fluid (be it a liquid or a gas) is broken down into a collection of discrete cells, called a mesh. OpenFOAM includes a number of utilities that will generate a mesh based on a description of the geometry of your fluid. For example, the `blockMesh` utility generates simple meshes of blocks, and the `snappyHexMesh` utility generates complex meshes of hexahedral or split-hexahedral cells. If you want to generate a basic block mesh, you would lay out the details in a dictionary file, called `blockMeshDict`, in the subdirectory `constant/polyMesh` within your case subdirectory. The file starts with:

```
FoamFile
{
    version      2.0;
    format       ascii;
    class        dictionary;
    object       blockMeshDict;
}
```

Case files, in general, start with a header of this format, describing what each case file consists of. In this particular file, you can define sections containing vertices, blocks, edges or patches. Once you have created this file, you can run the `blockMesh` utility to process this dictionary file and generate the actual mesh data that will be used in the computation.

The next step is to set your initial conditions. This is done by creating a subdirectory called `0` and placing the relevant case files here. These case files would contain the initial values and boundary conditions for the fields of interest. For example, if you were looking at a fluid flowing through a cavity, you may be interested in the pressure and

velocity. So, you would have two files in the `0` subdirectory describing these two fields. You also need to set any important physical properties in files stored in the `Dictionaries` subdirectory. These files end with `Properties`. This is where you would define items like viscosity. The last step in pre-processing is to set up the control file. This file is named `controlDict` and is located in the `system` subdirectory. As an example, let's say you wanted to start at $t=0$, run for 10 seconds with a timestep of 0.1 seconds. This section of the control file would look like this:

```
startFrom      startTime;
startTime      0;
stopAt         stopTime;
stopTime       10;
deltaT         0.1;
```

You also set output parameters in this control file. You can set how often OpenFOAM writes output with the `writeControl` keyword. So, let's say you want to write out the results every 10 timesteps. That section of the control file would look like this:

```
writeControl    timeStep;
writeInterval   10
```

This tells OpenFOAM to write out results every 10 timesteps into separate subdirectories for each write. These subdirectories would be labeled with the timestep.

You also can set the file format, the precision of results and file compression, among many other options.

Before you actually start the solver step, it probably is a good idea to check the mesh to be sure that it looks right. There is a post-processing tool called `paraFoam` that you can use. If you are in your case directory, calling `paraFoam` will load it up. Or, you can specify another directory location with the command-line argument `-case xxx`, where `xxx` is the case directory you are interested in.

The next step is to run a solver on your model and see what you get. Solvers tend to be specialized, in order to solve one particular class of problem efficiently. So OpenFOAM comes with a rather large set of standard solvers out of the box. If you are interested in solving a laminar flow problem for an incompressible fluid, you likely would use `icoFoam`. Or, if you are interested in looking at cavitation, you could use `cavitatingFoam`. Or, you may want a solver for internal combustion engines

(engineFoam). Take a look at the documentation to see what is available. Once you know which solver you are going to use, you can run it by simply executing the relevant binary from within your case directory, or you can use the `-case` command-line argument to point to another case directory.

The last step is the post-processing step, where you actually look at the results and see what happened inside your model. Here, you can use the supplied utility `paraFoam`. It can accept a `-case` command-line argument, just like all of the other utilities I've discussed. You then can manipulate the data and look at various time slices. You can generate isosurface and contour plots, vector plots or streamline plots. You even can create animations, although not directly. You can make `paraFoam` output image files representing movie frames. So they would be saved with filenames of the form:

```
fileroot_imagenum.fileext
```

Then, you can take this sequence of images and run them through something like the `convert` utility from ImageMagick to bundle them together into a single movie file.

As a final comment, in both the pre- and post-processing steps, there are utilities that allow you to convert to and from file formats used by other CFD packages, including `fluent`, `CFX` and `gambit`, just to name a few. This comes in handy if you are collaborating with other people who happen to be using one of these other packages. If you are working with someone who is using `fluent`, you would use `fluentMeshToFoam` in your pre-processing step and `foamMeshToFluent` in your post-processing step.

This short article provides only a small taste of OpenFOAM. If you are interested in fluid dynamics, you definitely should take a look at OpenFOAM. The home page includes links to great documentation and several tutorials. The installation package also includes several example cases, so you can see how standard problem sets are handled. You usually can use one of these examples as a jumping-off point for your own problem. Check it out and see if it can help you in your work.

—JOEY BERNARD

They Said It

If at first you don't succeed, call it version 1.0.

—Unknown

WE APOLOGIZE FOR THE INCONVENIENCE—God

—Douglas Adams, *So Long and Thanks for All the Fish*

All Your Base Are Belong To Us

—Zero Wing (Nintendo Game)

Now I am become Death, the destroyer of worlds.

—J. Robert Oppenheimer

Klaatu barada nikto.

From 1951's *The Day the Earth Stood Still*



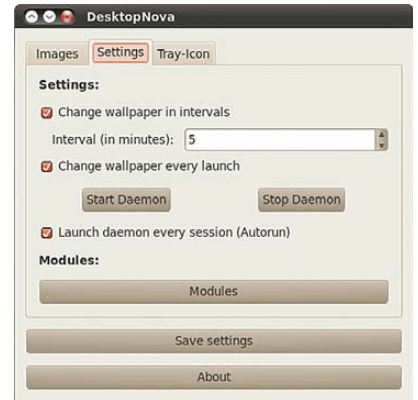
READERS' CHOICE AWARDS 2011

Vote for your Linux and open-source favorites in this year's Readers' Choice Awards at www.linuxjournal.com/rc11. Polls will be open August 2–22, 2011, and we'll announce the winners in the December 2011 issue of *Linux Journal*.

LINUXJOURNAL.COM

This month's issue is all about community. The Linux and Open Source community is what keeps all of us motivated and having fun with what we do. These communities range in size and scope from a small LUG in your hometown, to the distributed development teams on the largest Linux distributions, to other Open Source communities that exist to support projects like Drupal. They are all part of the larger ecosystem that is open-source software. We also like to think of LinuxJournal.com as its own community, and we hope you'll join us.

Visit www.linuxjournal.com/participate to connect with *Linux Journal* staff and others, check out our forums and learn about community events. We encourage you to chime in with your comments, and let your opinions be known. It's a great way to give and get feedback from our writers and readers. We also invite you to visit us on Freenode on our IRC channel, #linuxjournal. There, you'll find *Linux Journal* staff and readers who always are eager to share stories and sometimes lend a hand. See you there!—KATHERINE DRUCKMAN



DesktopNova has many options to set, and a convenient method to set them.

DesktopNova

One of the things that frustrates me about GNOME is the lack of wallpaper rotation integration. Thankfully, several tools are available to remedy that shortcoming. My current favorite solution is DesktopNova by Stefan Haller. DesktopNova not only lets you add multiple folders to your wallpaper rotation group, it also optionally allows you to show a tray icon to interact with it.

DesktopNova comes precompiled for several distributions, and it already may be in your repository. When combined with the hundreds of space photos I've downloaded from NASA, DesktopNova makes my computing experience out of this world! (Yes, I think that counts as the cheesiest thing I've written all year.)

Check it out at sites.google.com/site/haliner/desktopnova.

—SHAWN POWERS



REUVEN M. LERNER

Introducing CoffeeScript

An attractive, new language that makes JavaScript programming fun and easy.

For many years, JavaScript had a bad reputation among developers. Sure, we would use it when we needed to do something dynamic in the browser, but we also would avoid it as much as possible. As I (and others) have written numerous times in the past year, however, JavaScript is going through a multidimensional renaissance. It executes quickly, and it increasingly is standardized and stable across browsers and implementations. You can use it in the traditional role of browser-side applications, either by itself or using a framework, such as jQuery. You even can build entire MVC applications in the browser using systems like Backbone.js, as I described in this space the past two months. And on the server, you can use node.js to create high-performance applications.

It's great that JavaScript has improved in many ways. At the same time, the language contains many legacy issues—not in terms of capabilities, but in terms of the syntax and grammar. You can do great things with JavaScript, but it's easy to write code that has unexpected side effects, whose variables don't have the scope you expect, or whose functions operate just differently enough from your intention to cause problems.

(Another thing that hasn't improved is its name and the fact that although everyone calls it "JavaScript", it's officially known as "ECMAScript", named for the standards body that approved it.)

One of the more interesting solutions to this problem has been the introduction of languages that are supersets of JavaScript. One of the best-known examples is Objective-J, which bolts an Objective-C-like syntax, object structure and class library onto JavaScript. The Cappuccino framework for in-browser MVC applications is written in Objective-J and provides a programming experience similar to that of the Cocoa platform for the Macintosh. Another solution has been to run a preprocessor on JavaScript code, as in the case of Google Closure. Closure compiles JavaScript into JavaScript, but in so doing, optimizes and minimizes your JavaScript code.

Another approach, taken by Jeremy Ashkenas (who, incidentally, also created the Backbone.js framework) was to create an entirely new language that compiles into JavaScript. This language, which he called CoffeeScript, is not designed to be run directly (although you can do so, with the right tools). Rather, CoffeeScript code is compiled into JavaScript code,

which then is executed.

Why would you want to compile a program into JavaScript? First, because modern JavaScript engines are quite speedy. Compiling to JavaScript ensures that the execution will be fast, and that it will work on a wide variety of platforms and in many contexts—similar in many ways to the growing number of languages (for example, Clojure and Scala) that compile to JVM bytecodes. Second, if you can compile into JavaScript, you can take advantage of the libraries and frameworks that already work with JavaScript.

The other reason to use a language other than JavaScript, but that compiles into JavaScript, is that you can use a syntax that's more accessible and less prone to errors. I'm a longtime user of both Ruby and Python, and I found that CoffeeScript incorporated many of my favorite features from both languages to create a language with easy-to-understand syntax and powerful features. If you know Ruby, Python and JavaScript, you'll likely be able to learn this language in a relatively short time period.

Now, it's true that CoffeeScript has been around for about 18 months at the time of this writing, and that it already has attracted a number of fans. But, several events have made it even more interesting during the past few months. First, Brendan Eich, Mozilla's CTO and the inventor of JavaScript, said earlier this year that CoffeeScript may well provide some useful syntax ideas for the next version of JavaScript, known as "Harmony". Douglas Crockford, well known for his ideas, writing and lectures about JavaScript, apparently has lent his support to CoffeeScript as well.

Even more significant, from my perspective, is that the Ruby on Rails core team has announced that CoffeeScript will be a standard part of Rails, starting with version 3.1. (Version 3.1 also will feature jQuery as the standard framework, replacing Prototype, and the incorporation of SASS, a CSS macros language.) I'm not going to explore the details of CoffeeScript and Rails here, but the fact that a large, growing, active and influential Web framework is adopting CoffeeScript will give JavaScript language architects like Eich a chance to see how it works "in the wild" and gather feedback from developers before deciding what will (and won't) go into the next version of JavaScript.

So, this month, I want to introduce some of the basics of CoffeeScript. I must admit I was skeptical of learning yet another new language, especially one that compiles into JavaScript. But after a bit of playing and experimentation, I see why people are so excited about it. Just where CoffeeScript will be in another few years remains to be seen, but it's certainly a useful tool to have right now, particularly if you're working on Web applications whose code has become increasingly twisted and complex with JavaScript functions.

Installing and Running

You can download and install CoffeeScript in a number of ways. One method is to download and compile the source from the GitHub account on which it is stored and developed. Just say:

```
git clone http://jashkenas.github.com/coffee-script/
```

and you will have the source on your machine. The CoffeeScript compiler depends on node.js, so it's not surprising that you also can download and install it using the node.js package manager, npm:

```
npm install coffee-script
```

Finally, some Linux packages (of various flavors) exist for CoffeeScript, for those who would prefer to install it alongside other packages, with dependencies taken care of.

Once you've installed it, you can involve the interactive command-line prompt, and start coding at the `coffee>` prompt.

You can print something immediately, using the built-in `console.log` function:

```
coffee> console.log "Hello, world"
```

What's `console.log`? It's a built-in function on the "console" object—for example:

```
coffee> console.log.toString()
console.log.toString()
function () {
  process.stdout.write(format.apply(this, arguments) + '\n');
}
```

As you can see, this lets you take a look at the JavaScript behind the scenes. If you do the same thing with the "console" object, you see something that looks even more like the JavaScript you know and love (or not):

```
coffee> console.toString()
console.toString()
[object Object]
```

In some ways, I haven't yet done anything different

from standard JavaScript. JavaScript, after all, is all about functions—those that you can pass as parameters, and those that you can execute with parentheses, much like in Python. You can start to see the differences as soon as you define a function though:

```
coffee> hello_world = (name) -> console.log "Hello there, #{name}!"
```

First and foremost, you can see that defining a function in CoffeeScript is really a variable assignment. The function-definition syntax is still a bit weird by my standards, using `->` to separate the parameters and body of an anonymous function. Parameters are named in parentheses, with commas separating them if necessary.

In this particular function body, I'm also using Ruby-style string interpolation to print the user's name. If you're tired of using `+` or a third-party library just to interpolate string values, this is likely to be a useful feature for you.

Basic Syntax

To execute the function, just give the CoffeeScript REPL (read-eval-print loop) the name of the function, followed by parentheses (that is, execution):

```
coffee> hello_world()
```

CoffeeScript functions return their final value, much as Ruby methods do. In the case of this `hello_world` function, that means the function is returning "undefined". You can make that a bit better by having the function return the string, rather than print it:

```
coffee> hello_world = (name) -> "Hello there, #{name}!"
```

Then, you can say:

```
coffee> console.log hello_world('Reuven')
Hello there, Reuven!
```

or:

```
coffee> hello_world('Reuven')
'Hello there, Reuven!'
```

depending on whether you want to get the string back or just print it to the console.

CoffeeScript provides the basic JavaScript data types, but it adds a great deal of syntactic sugar, as well as additional methods, that make those data types easier to deal with. For example, CoffeeScript uses Python-style triple-quoted strings.

The "existential" operator, a question mark (?), allows you to determine whether a variable contains something other than null or undefined values. This is better than just checking for the truth of a value, which will lead to (literal!) false negatives if a value contains

0 or the empty string. For example:

```
v = 0

if v
  console.log "yes"
else
  console.log "no"

if v?
  console.log "yes"
else
  console.log "no"
```

The above code example shows how CoffeeScript implements if/else control blocks. There also is an “unless” operator, which (as you might expect) inverts the output from if. CoffeeScript also provides postfix “if” and “unless” statements—something you might recognize from Perl and Ruby.

You also can see another element of CoffeeScript above, this one taken from Python. Whitespace and indentation are significant, and allow you to remove much of the curly braces and begin/end of many other

languages. Unlike Python, however, colons aren’t necessary after the “if” and “else” statements.

CoffeeScript arrays are like JavaScript arrays, but with all sorts of nice syntax added. Want a ten-element array? Just use this:

```
a = [0..9]
```

and you’ll get it, using the built-in “range” operator. (This doesn’t work for letters though.) You can retrieve ranges as well:

```
a[5..7]
```

You can assign to a range, without worrying about the precise length of what you’re splicing in:

```
a[5..7] = ['a', 'b']
```

You also can retrieve substrings using this syntax:

```
coffee> alphabet[5..10]
alphabet[5..10]
'fghijk'
```



visit us at www.siliconmechanics.com
or call us toll free at 866-352-1173

Silicon Mechanics is proud to announce their sponsorship of Wave Vidmar and the Ocean Row Solo North Atlantic Challenge 2011, a solo row from west-to-east across the North Atlantic.

Expedition

Departure: July 2011, Cape Cod
Destination: England
Duration: 140 days (estimated)

Philosophy

“It’s not just one man rowing across an ocean, it’s hundreds or thousands of people rowing across an ocean. It’s a cooperative effort, that helps push the boundaries and limits, giving real-life examples to others in what they too can possibly achieve.”

Equipment

“We only work with the best of the best. When you trust your life to equipment and/or services, you can’t afford to have second-best. We only accept sponsors that have superior product, value, and policies.”

Goals

Education and research: Wave’s Science Education Adventure Program reaches hundreds of thousands of students around the world.

Boat

Liberty is 24 feet long, with a 6.5 foot beam, and weighs 1400 pounds fully laden. It is the most advanced, technologically equipped ocean row boat in the world. Shaped and designed using 3D modeling programs, the boat was tested under simulated extreme conditions long before it was ever constructed.

Mission Control

Silicon Mechanics and Intel have partnered to donate the Rackform iServ workstation that will run mission control functions for the expedition. The Intel® Xeon® Processor E3-1200 Series is at the heart of his customized high-performance system.



For more information about our
Intel Xeon Processor-based servers
visit www.siliconmechanics.com/xeon

For more information about Wave Vidmar and Ocean Row Solo,
visit www.oceanrowsolo.com.
To track the expedition, visit www.siliconmechanics.com/ors.

CoffeeScript strings, like JavaScript strings, are immutable. Thus, you cannot assign to a substring:

```
coffee> alphabet[5..10] = [5]
alphabet[5..10] = [5]
[ 5 ]
```

```
coffee> alphabet
alphabet
'abcdefghijklmnopqrstuvwxyzz'
```

JavaScript objects, which can be used like hashes (aka “dictionaries” or “associative arrays”) work just as they do in JavaScript, but with some easier syntax. For example, in JavaScript, you would write:

```
person = {
  first_name: 'Reuven',
  last_name: 'Lerner'
}
```

In CoffeeScript, you can say:

```
person =
```

```
  first_name: 'Reuven'
  last_name: 'Lerner'
```

Once again, using Python-style indentation instead of curly braces makes it more compact but no less readable.

Loops and Comprehensions

You can loop through arrays with `for..in`:

```
for number in a
  console.log number
```

I should point out that this will print each number on a line by itself and will return an array of undefined values (because `console.log` doesn't return a value, and “for” returns an array).

Objects can be accessed using the similar `for..of` loop:

```
for key, value of person
  console.log "key = '#{value}'"
```

Note that `for..of` loops work on any JavaScript object. Because every object can contain properties, and because arrays are objects, you even can assign

When you partner with Silicon Mechanics, you get more than affordable, high-quality HPC — you get a team of Experts pulling right along with you.



**Powerful.
Intelligent.**

Expert included.

Silicon Mechanics and the Silicon Mechanics logo are registered trademarks of Silicon Mechanics, Inc. Intel, the Intel logo, Xeon, and Xeon Inside, are trademarks or registered trademarks of Intel Corporation in the US and other countries.

properties to arrays:

```
a.foo = 'bar'
```

Using a `for..in` loop, you'll continue to see the values in the "a" array. But using a `for..of` loop, you'll get not only the array elements (whose keys are the indexes), but also "foo". Additionally, note that if you're interested only in the properties defined on an object, rather than on its prototype, you can add the "own" keyword:

```
for own key, value of person
  console.log "key = '#{value}'"
```

In this particular case, there will be no difference. But if "person" were to have a prototype, and if the prototype were to have keys of its own, you would see the prototype's keys and values, and not just those for the "person" object.

But the need for these `for..in` and `for..of` loops is reduced dramatically in CoffeeScript because of comprehensions, an idea that comes from Python that takes some getting used to, but that offers a great deal of power and simplicity once you do so, allowing you to combine `map`, `reduce` and `filter` in a single statement. For example, you can take a list of numbers:

```
coffee> a = [100..110]
[ 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110 ]
```

And you then can multiply each number by 5, much as you would do with "map" in Ruby or Python, but using "comprehension" syntax:

```
coffee> i*5 for i in a
[ 500, 505, 510, 515, 520, 525, 530, 535, 540, 545, 550 ]
```

You also can select out some values. Suppose, for example, you want to operate only on those elements that are odd. You can say:

```
coffee> i*5 for i in a when i%2
[ 505, 515, 525, 535, 545 ]
```

Objects

As I wrote above, everything in JavaScript (and, thus, in CoffeeScript) is an object, and every object has properties. (Another way to think about this is to say that everything in JavaScript is a hash, with key-value pairs.) Understanding and working with this can be a bit confusing at first, particularly because of the way the "this" keyword operates. CoffeeScript offers several solutions to this problem, as well as the scoping and variable issues that drive JavaScript programmers crazy.

First, scoping of all variables is lexical and is obvious from the indentation. You can attach variables to the global object, or other objects, but you will end up

doing this only when you want, not by mistakenly forgetting to add a "var" somewhere.

Second, properties can be accessed and assigned to as if they were variables, using Ruby-style `@varname` syntax. So, when you say `x=5` in a CoffeeScript program, you're assigning the value 5 to the lexical value `x`. When you say `@x=5` though, you're assigning the value 5 to the property `x` on the current object.

Finally, "this" is scoped dynamically in JavaScript, changing value to reflect the object to which the current function is attached. This means that if you aren't careful, you can write callback functions that reference "this", but accidentally end up referring to the wrong object. CoffeeScript lets you define functions not only with `->` (the "thin arrow"), but also with `=>` (the "fat arrow"). The difference is that when you define functions with `=>`, they are bound to the value of "this" when it was defined, allowing access to the defining context's properties using `@proprname` syntax.

All of these things come together in CoffeeScript's object model, which extends JavaScript's such that you can work with something resembling a traditional class-instance model, rather than the built-in, JavaScript-style prototype-based model. Prototypes still exist and work—and indeed, CoffeeScript's classes compile into JavaScript prototypes. But, you can get beyond prototype-style inheritance by declaring classes with a constructor. Here is a simple example of the sort of thing you can do:

```
class Person
  constructor: (firstName='NoFirst', lastName='NoLast') ->
    @firstName = firstName
    @lastName = lastName
    Person.count++

  @count: 0

p1 = new Person()
console.log p1
console.log Person.count

p2 = new Person('Reuven')
console.log p2
console.log Person.count

p3 = new Person('Reuven', 'Lerner')
console.log p3
console.log Person.count
```

When you run the above file, you get the following output:

```
{ firstName: 'NoFirst', lastName: 'NoLast' }
1
{ firstName: 'Reuven', lastName: 'NoLast' }
2
```

```
{ firstName: 'Reuven', lastName: 'Lerner' }  
3
```

This not only shows how you can use CoffeeScript classes in a way similar to traditional classes, but also that you can have default values for function parameters by declaring their values before the -> sign. You also can see how naturally the use of the @propname syntax fits into this object model. The above constructor looks almost like a Ruby method, rather than a JavaScript one, with a clear distinction between local variables and instance variables.

Conclusion

CoffeeScript is an attractive, new language that makes JavaScript programming fun and easy. You can think of it as JavaScript with a new syntax or as an easy-to-learn language that integrates into JavaScript applications, with many ideas taken from Python and Ruby. It removes many of the syntactic bloat and issues associated with traditional JavaScript, simultaneously providing a language that's easier to write and maintain, but also faster to execute, than raw, unoptimized JavaScript. CoffeeScript is a language we all should watch in the coming months and years. It probably won't replace JavaScript altogether, but it definitely will make it easier to accomplish certain tasks.

Next month, I'll look at how CoffeeScript can be integrated into your browser-side Web applications, especially those using jQuery. That's the direction the Ruby on Rails community seems to be moving toward with its 3.1 release, and even if your favorite framework doesn't adopt CoffeeScript, understanding how they work together probably will be useful. ■

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

Resources

The home page for CoffeeScript, including documentation, quick references, FAQs and annotated source code, is at jashkenas.github.com/coffee-script. There is an active and growing community of CoffeeScript users, with an IRC channel (#coffeescript) and Wiki at GitHub.

For a good introduction to CoffeeScript, see the presentation written by Jacques Crocker, available at coffeescript-seattlejs.herokuapp.com.

Finally, the Pragmatic Programmers have released (at the time of this writing) an excellent pre-release "beta book", written by active CoffeeScript user Trevor Burnham. If you're interested in learning more about this interesting little language, I highly recommend this book. It's mostly aimed at beginners, but given the limited number of advanced CoffeeScript programmers out there, this should not bother you.



Linux - FreeBSD - OpenSolaris - etc.

Proven **Technology**.

Proven **Reliability**.

When you can't afford to take chances with your business data or productivity, rely on a Genstor server customized to your specifications.

POWER

PERFORMANCE

Fly into the Cloud with Genstor Systems



- Up to 48 cores in a 1U.
- AMD Opteron 6100 series.
- Single high-efficiency power supply.
- Up to 512GB DDR3 memory.
- Ideal as front end processing servers.

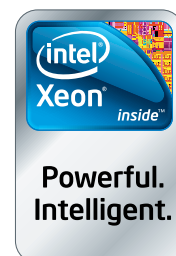


- Up to 12 cores in a 2U
- Dual redundant high efficiency power.
- Up to 96GB DDR3 memory.
- Server Power Capping via Intel Intelligent Power Node Manager.
- Ideal as front end processing and/or storage.



- Up to 4 GPU cards.
- Dual redundant high efficiency power.
- Up to 192GB DDR3 memory
- Up 24 cores using 2 CPUs.
- Up to 8 3.5" disks.

Genstor Systems, Inc.



1501 Space Park Drive
Santa Clara, CA 95054
www.genstor.com
E-mail: sales@genstor.com
Phone: 877-25 SERVER
408-980-0121

Intel®, the Intel® logo, Intel® Xeon®, and Xeon® Inside® are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.



DAVE TAYLOR

Calculating Day of the Week

Working on the nuances of this day-of-the-week shell script.

For those of you playing along at home, you'll recall that our intrepid hero is working on a shell script that can tell you the most recent year that a specific date occurred on a specified day of the week—for example, the most recent year when Christmas occurred on a Thursday.

There are, as usual, nuances and edge cases that make this calculation a bit tricky, including the need to recognize when the specified date has already passed during the current year, because if it's July and we're searching for the most recent May 1st that was on a Sunday, we'd miss 2011 if we just started in the previous year.

In fact, as any software developer knows, the core logic of your program is often quite easy to assemble. It's all those darn corner cases, those odd, improbable

The solution I'm going to work with is likely more complicated than necessary, but it's mine and I'm sticking with it.

situations where the program needs to recognize and respond properly that makes programming a detail-oriented challenge. It can be fun, but then again, it can be exhausting and take weeks of debugging to ensure excellent coverage.

That's where we are with this script too. On months where the first day of the month is a Sunday, we're already set. Give me a numeric date, and I can tell you very quickly what day of the week it is. Unfortunately, that's only 1/7th of the possible month configurations.

What DOW Is That DOM?

For purposes of this discussion, let's introduce two acronyms: DOM is Day Of Month, and DOW is Day Of Week. May 3, 2011, has DOM=3 and DOW=3, as it's a Tuesday.

The cal utility shows this month like this:

```

    May 2011
Su Mo Tu We Th Fr Sa
1  2  3  4  5  6  7

```

```

 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

```

Look! A perfectly formed month, so it's easy to figure out the DOW for once. But, that's not really fair for our testing, so let's move forward a month to June and look at June 3 instead. That's DOM=3, DOW=6 (Friday):

```

    June 2011
Su Mo Tu We Th Fr Sa
           1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30

```

The solution I'm going to work with is likely more complicated than necessary, but it's mine and I'm sticking with it.

Here's the idea. As awk goes through the lines, it easily can ascertain NF (number of fields). If NF < 7, we have a month where the first day starts on a DOW other than Sunday. Any matching date for the first week of June 2011, for example, would have NF = 4.

Look back at June though, because it's important to recognize that the last week of the month has a problem too. It has NF=5. Because any match in that line must have DOM > 7, however, we can address this nuance later. Stay tuned, as they say.

The formula we can use to calculate day of week for the first week of a month, however, given all this information and with i the day of the month is DOW=i+(7-NF). A few test cases verify that it works:

```

June 3 = i=3, NF=4      DOW=(7-4)+3 = 6
July 1 = i=1, NF=2      DOW=(7-2)+1 = 6
May 2 = i=2, NF=7       DOW=(7-7)+2 = 2

```

For any date that doesn't occur on that first week, however, we can ignore all these complicated calculations and simply get the day of the week.

How do you tell if it's in the first week? Another

test. Search for the matching DOM and then look at the matching line number. If it's not line 1, we have to calculate the day of week from the matching cal output line:

```
awk "/$expr/ { for (i=1;i<=NF;i++)
  { if (\$i~/${day}/) { print i } } }"
```

In my previous columns, I was creating this overly complicated regular expression to match all the edge cases (literally, the cases when the match was the first or last day of a week). Instead, here's a new plan that's faster and less complicated. We'll use sed to pad each calendar with leading and trailing spaces:

```
cal june 2011 | sed 's/^/ /;s$/ /'
```

Now our regular expression to match just the specified date and no others is easy:

```
[^0-9]DATEINQUESTION[^0-9]
```

Further, awk easily can give us that NF value too, so here's a rough skeleton of the DOW function for a given day of the month, month and year:

```
figureDOM()
{
  day=$1; caldate="$2 $3"
  expr="[^0-9]${day}[^0-9]"
  NFval=$(cal $caldate | sed 's/^/ /;s$/ /' | \
    awk "/$expr/ { print NF }")
  DOW="$(( $day + ( 7 - $NFval ) ))"
}
```

That works if we search only for matches that are in the first week of the month, but that, of course, is unrealistic, so here's a better, more robust script:

```
figureDOW()
{
  day=$1; caldate="$2 $3"
  expr="[^0-9]${day}[^0-9]"
  cal $caldate | sed 's/^/ /;s$/ /' > $temp
  NRval=$(cat $temp | awk "/$expr/ { print NR }")
  NFval=$(cat $temp | awk "/$expr/ { print NF }")
  if [ $NRval -eq 3 ]; then
    DOW="$(( $day + ( 7 - $NFval ) ))"
  else
    DOW=$(cat $temp | awk "/$expr/
  { for (i=1;i<=NF;i++) { if (\$i~/${day}/) { print i } } }"
  fi
  /bin/rm -f $temp
}
```

A few quick tests:

```
DOW of 3 june 2011 = 6
DOW of 1 july 2011 = 6
```

How do you tell if it's in the first week? Another test. Search for the matching DOM and then look at the matching line number.

```
DOW of 2 may 2011 = 2
DOW of 16 may 2011 = 2
```

Looks good!

Next month, we'll tie this all together. We have a function that calculates day of week for a given date, we already have figured out how to parse user input to get a desired day of week for a specified month/day pair, and we know how to figure out if the starting point for our backward date search is the current year (for example, whether we're past that point in the year already).■

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

Open Frame Panel PC

PPC-E7+

- Fanless ARM9 400MHz CPU
- 4 Serial Ports & SPI
- Open Frame Design
- 10/100 BaseT Ethernet
- SSC-I2S Audio Interface
- 2 High Speed USB 2.0 Ports
- SD/MMC Flash Card Interface
- Battery Backed Real Time Clock
- Up to 1 GB Flash & 256 MB RAM
- Linux with Eclipse IDE or WinCE 6.0
- JTAG for Debugging with Real-Time Trace
- WVGA (800 x 480) Resolution with 2D Accelerated Video
- Wide Input Voltage from +12 Vdc to +26 Vdc





Setting up a Panel PC can be a puzzling experience. However, the PPC-E7+ Compact Panel PC comes ready to run with the Operating System installed on Flash Disk. Apply power and watch either the Linux X Windows or the Windows CE User Interface appear on the vivid color LCD. Interact with the PPC-E7+ using the responsive integrated touch-screen. Everything works out of the box, allowing you to concentrate on your application, rather than building and configuring device drivers. Just Write-It and Run-It.

For more info visit: www.emacinc.com/panel_pc/ppc_e7.htm

Since 1985
OVER
25
YEARS OF
SINGLE BOARD
SOLUTIONS



EMAC, inc.
EQUIPMENT MONITOR AND CONTROL

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



KYLE RANKIN

Nagging Notifications

Why let your e-mail do all the nagging? Let screen alert you to Nagios issues.

In the February 2011 issue, I wrote about screen, the console window manager, and how I configure its hardstatus line to show notifications along the bottom of my terminal window. Although some people like their desktop environment to fire up notifications when they have a new e-mail or IM, because I spend a good deal of my time within screen, it has my focus, and it makes sense to put important notifications there. In that February 2011 article (see Resources), I introduced how to set up the hardstatus line and demonstrated a custom script I use to show when I have new e-mail.

For this article, I expand on the topic of screen notifications with a new notification script I've found incredibly useful. Ever since I've had more than a handful of servers, I've relied on monitoring programs like Nagios to keep track of server health. Although monitoring software has its own method of notifications via e-mail or SMS, I've found it valuable to have my current

Although monitoring software has its own method of notifications via e-mail or SMS, I've found it valuable to have my current Nagios health right there in my screen session.

Nagios health right there in my screen session. It not only provides a backup to my mail notifications, it also saves me from having a Nagios window open in my browser all the time.

If you are new to screen and haven't set up a custom hardstatus line, check out my February 2011 article first to get up to speed. Instead of revisiting how to configure a .screenrc file from scratch, I'm assuming you already have a basic .screenrc set up, and instead, I'm skipping ahead to how to add this Nagios script to your existing screen session.

Screen Scraping for Screen

When I set about writing this script, I realized there are a number of different ways to capture the current health of Nagios. Although I didn't spend a lot of time looking into it, I imagine there are lower-level APIs I could query, but honestly, all I really wanted was to know if Nagios was all green (okay) or had any warnings or critical alerts (yellow or red), and if so, how many. To accomplish that, I decided the simplest method was to scrape one of the Nagios status pages

for the information I needed. Honestly, this same method should work pretty well for just about any monitoring program you might use, as long as it has a Web interface and you have enough regex-fu to parse the HTML for the data you need.

I originally wrote the script so that if the Nagios status was okay, it would print that, and if there were any critical or warning alerts, it would output those statistics instead. I realized that I wanted screen to print okay in green, warnings in yellow and critical alerts in red. That way, I might notice problems even if I wasn't looking directly at my terminal at the time. To accomplish this, I actually needed to run the script three different times within screen.

The script below takes just two arguments: the Nagios host to poll (with an optional user name and password if you use one) and the type of status to report. I chose the color codes green, yellow and red to represent okay, warning and critical statuses, respectively. I found the `http://nagioshostname/cgi-bin/nagios3/tac.cgi` page was the simplest to scrape and had all of the information I needed for the script:

```
#!/usr/bin/perl

# usage: nagios_scraper.pl [user:password@]nagios_host STATUS
# where STATUS is green, red, yellow, or all

$ Nagios_host=shift;
$ show=shift;

open TAC, "wget --timeout=2 -q -O -
  http://$Nagios_host/cgi-bin/nagios3/tac.cgi |"; @tac = <TAC>;
close TAC;

foreach $line (@tac){
    if ($line =~ /\(d+\) Down/){ $hosts_down = $1; }
    elsif($line =~ /\(d+\) Unreachable/){ $hosts_unreachable = $1; }
    elsif($line =~ /\(d+\) Up/){ $hosts_up = $1; }
    elsif($line =~ /\(d+\) Pending/){ $hosts_pending = $1; }
    elsif($line =~ /\(d+\) Critical/){ $services_critical = $1; }
    elsif($line =~ /\(d+\) Warning/){ $services_warning = $1; }
    elsif($line =~ /\(d+\) Unknown/){ $services_unknown = $1; }
    elsif($line =~ /\(d+\) Ok/){ $services_ok = $1; }
    elsif($line =~ /\(d+\) Pending/){ $services_pending = $1; }
}

# remove the username and password from the output
$ Nagios_host = s/.*\@//;
```

```

if($show eq "green" && ($hosts_down == 0 && $services_critical == 0
->&& $services_warning == 0)){
    print "$nagios_host: OK";
}
elseif($show eq "red" && ($hosts_down > 0 || $services_critical > 0)){
    print "$nagios_host: ${hosts_down}D ${services_critical}C ";
}
elseif($show eq "yellow" && $services_warning > 0){
    print "$nagios_host: ${services_warning}W ";
}
elseif($show eq "all"){
    print "${hosts_down}D ${hosts_up}U ${services_critical}C
->${services_warning}W ${services_ok}OK";
}

```

As you can see, I actually collect a lot more statistics than I ultimately use, just in case I want to refer to them later. The important thing to note in this script is that in each of the green, red and yellow statuses, I print something only if there's something of that status to print. This is crucial, because I don't want to clutter my hardstatus line, and I want to see yellow or red text

only if it truly needs my attention.

Name this script nagios_scraper.pl, put it either in /usr/local/bin for everyone to use or your user's ~/bin/ directory, make sure it is executable, and then test it against your Nagios server to make sure you have the syntax right. For instance, if you had no user name or password set up for Nagios, and your Nagios server was named naggyhost, you would type the following command to test if everything was okay:

```
$ /usr/local/bin/nagios_scraper.pl naggyhost green
```

Type the following to test for critical alerts:

```
$ /usr/local/bin/nagios_scraper.pl naggyhost red
```

Or, type the following to test see all statuses:

```
$ /usr/local/bin/nagios_scraper.pl naggyhost all
```

I do recommend that you set up a user name and password for your Nagios Web access if you haven't already. Because the user name and password you use



General Parallel File System

Compute Nodes



Compute Network Switch



I/O Server



I/O Server



I/O Server



Optimized with:

- Intel® Xeon® Processor X5690
- LSI 6Gb/s MegaRAID® SAS 9280 Controller

- Enterprise cluster storage file system
- Seamless capacity expansion
- Policy-driven ILM over tiered storage
- Scales vertically for more data
- Scales horizontally for more concurrent users
- High performance, reliability, and availability
- Exploit parallelism from all nodes to all disks
- Data replication increases availability to withstand media failures/errors
- Multiple paths to data to withstand communication failures/errors
- File system logging to withstand system failures/errors
- Snapshots to withstand user failures/errors
- Manage cluster operations from any node
- Supports multiple versions within the same cluster
- Supports rolling upgrades for zero downtime



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



Yang Ming International Corp. (RackMountPro.com)

The Leading Server Builder in America. Enhancing Cloud Computing, The Optimized Technologies for Cloud

595 Yorbita Road, La Puente, CA 91744

Tel: (800) 526-8650

Fax: (626) 956-0098

sales@rackmountpro.com

for this script ultimately will end up in plain text, I recommend setting up an account for the Nagios Web interface that can log in but can see only the Nagios status and can't submit any changes (like maintenance modes and acknowledgements). Let's assume I set up an account called readonly with a password of n0wr1t3 on naggyhost. I would call the script like this:

```
$ /usr/local/bin/nagios_scraper.pl readonly:n0wr1t3@naggyhost red
```

Again, if the script doesn't provide any output in one of the modes, it could just mean that the status doesn't currently apply. If you want to test that for sure, run the script with the all argument instead of green, yellow or red to see the full status.

We Will, We Will, Nag You

Once you have tested the script and have it working, the next step is to add it to your ~/.screenrc. The first step is to add three new backtick configuration lines to ~/.screenrc that will call nagios_scraper.pl each with green, red and yellow statuses. In my case, I assume you might have a few backtick commands defined, so I start with command 110:

```
backtick 110 27 27 /usr/local/bin/nagios_scraper.pl
↳readonly:n0wr1te@naggyhost red
backtick 111 61 61 /usr/local/bin/nagios_scraper.pl
↳readonly:n0wr1te@naggyhost yellow
backtick 112 73 73 /usr/local/bin/nagios_scraper.pl
↳readonly:n0wr1te@naggyhost green
```

I've set each of these commands to run at different intervals. I want to check for critical alerts more frequently than warnings or when everything is okay, so I run the command with the red argument every 27 seconds. I then run it with yellow and green every 61 and 73 seconds. Note that I set these intervals to be at odd times. I've realized the value in staggering my screen notification scripts so they don't risk all running at the same time, so to help with that I try to choose odd intervals.

Once you have defined the backtick lines, the next step is to add them to your hardstatus string so they show up in green, yellow and red. In my case I pasted in:

```
%{+b r}%110`%{+b y}%111`%{= g}%112`
```

so that my hardstatus string modified from my previous article would be:

```
hardstatus string '%{= w}%Y-%m-%d %c | %l | %101`'
↳%{+b r}%110`%{+b y}%111`%{= g}%112`'
```

Now save your changes to your ~/.screenrc, and either start up a new screen session or type Ctrl-A : and type source ~/.screenrc to load these changes

```
greenfly@moses:--$
2011-05-02 21:50 | 0.72 0.69 0.65 | naggyhost: OK
```

Figure 1. Everything is okay.

```
greenfly@moses:--$
2011-05-02 21:52 | 0.72 0.70 0.65 | naggyhost: 1D 5C
```

Figure 2. One host is down alert with five critical services.

into your existing screen session. Figures 1 and 2 show what the hardstatus line will look like either when the status is okay or when there are critical alerts.

What amazes me the most the more I dig in to screen notifications is just how simple it is to add new scripts to the list once you get the hang of it. Even if you don't use screen, it wouldn't be too difficult to modify the script so that it outputs to a desktop notification instead (see my December 2009 column for details).■

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.

Resources

"Status Messages in Screen" by Kyle Rankin, *LJ*, February 2011: www.linuxjournal.com/article/10950

"Message for You Sir" by Kyle Rankin, *LJ*, December 2009: www.linuxjournal.com/article/10612

10 years of Open Source success and adoption by an increasing number of high profile sites, demand for Drupal talent is high.

NOW IS A GREAT TIME TO BE A DRUPAL PRO

@drupalcon



DRUPALCON LONDON

22-26 August 2011, Fairfield Halls

**DON'T MISS
OUT. REGISTER
NOW!**

london2011.drupal.org

Come to DrupalCon London...

Meet thousands of Drupal users, developers, designers and business people

Develop your Drupal skills by attending sessions, talks and code sprints

Experience our amazing community and have loads of fun



Compuware's Gomez Platform

Compuware's integrated application performance management (APM) solution not only has been upgraded but also rechristened the Compuware Gomez Platform. Gomez, says Compuware, "optimizes mobile and Web application performance across the entire application delivery chain, from data centers, through the cloud, to the edge of the Internet". This update includes new functionality in both Compuware's on-premise APM products (formerly called Vantage) and its software as a service (SaaS) APM products, now all operating under the Gomez product brand name. New features and capabilities in the new Gomez release include real-user monitoring, mobile-readiness assessment, real-user transaction analytics, integration with Google Page Speed and Internet health map with last-mile data.



www.compuware.com

NOBLE SYSTEMS

Noble Harmony

Noble Systems has released the next generation (that is, version 2) of Noble Harmony, a browser access tool for remote and mobile contact center management. Version 2 builds on the earlier edition's flexible, browser-agnostic design, adding improved user-interface layout and customization. The net effect is that managers and supervisors can operate effectively without being tethered to an individual station by allowing them to monitor real-time activities with mobile devices and tablets. More specifically, they now can arrange how KPI information is displayed on the dashboard, speed access to frequently used screens, query a vastly expanded array of variables, set alerts to be triggered on specific activities and parameters, find and analyze individual campaigns and agents, and manage list assignments.

www.noblesys.com

Protecode's Developer Assistant

The folks at Protecode have expanded the reach of Developer Assistant, a tool that the company calls "the industry's first solution for real-time management of open-source licensing and copyrights". Previously available only for the Eclipse IDE, the tool now supports all operating systems and all software development tools worldwide. Developer Assistant, says Protecode, operates unobtrusively in the background and requires no training for developers, allowing them to concentrate on development rather than open-source license management. Operating directly at the developer workstation, it uses the code analysis services of Protecode's Enterprise Server and compares the code structure of a file to signatures of millions of files stored in Protecode Global IP Signatures. There, it identifies the licensing and copyright obligations of the file and provides instant feedback to the developer as the code is put together or brought in from the Internet, an external storage device or a corporate repository.

www.protecode.com

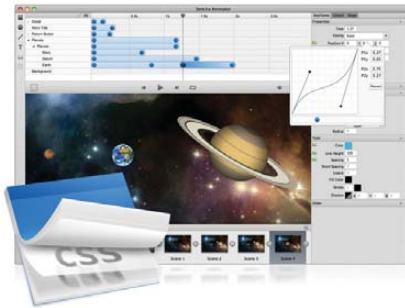
protecode



Lucid Imagination's Lucid Certified Developer Exam

If your daily grind involves development with Apache Lucene/Solr technology, surf on over to Lucid Imagination to find out more about its new Lucid Certified Developer Exam. The company calls the exam "the industry's only standardized benchmark of open-source search knowledge, skills and experience". Created by some of the most esteemed experts and contributors for the Lucene/Solr Project, the Lucid Certified Developer Exam provides developers with a way to validate their expertise and differentiate themselves in the competitive job market, as well as helps employers select new hires with more confidence and more accurately assess knowledge gaps in the current employee base.

www.lucidimagination.com/certification



Sencha Animator

Sencha is encouraging interactive designers to switch to its new Sencha Animator v.1, “the industry’s first purpose-built tool for creating CSS3-based animations that deliver plug-in-free rich-media experiences”. Designed for interactive designers and advertising creatives who want to create cross-platform animations for playback on WebKit browsers and touchscreen mobile devices, Sencha Animator brings static Web pages to life and ensures high-fidelity playback on all leading mobile devices and desktop browsers. Key product features include full CSS3 support, interactive timeline and intuitive object property controls, an intuitive timeline interface and exporting of animations to readable HTML.

www.sencha.com

InMage Scout

From the backup and recovery news desk comes breaking news from InMage on version 6 of InMage Scout—a single, easy-to-use solution for local backup and replication of physical and virtual environments that is based on continuous data protection (CDP) technology. InMage claims that the upgraded InMage Scout becomes the only unified product on the market featuring CDP-based backup and remote disaster recovery support for most major enterprise platforms, including Linux, AIX, Windows, VMware, Solaris, XenServer and Hyper-V. The new Scout also features both asynchronous IP and synchronous FC-based data protection without imposing software-based storage virtualization on customers. With this launch, InMage Scout 6.0 also introduces the PureSplit architecture, which allows customers to use their existing storage without having to virtualize it through server appliances.

www.inmage.com



Quepixon QueCloud

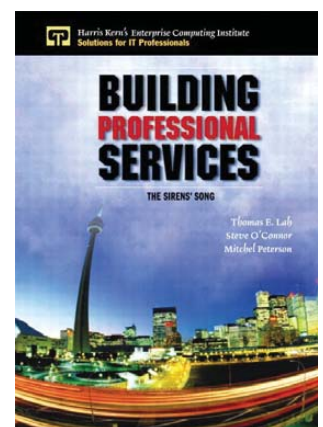
Integrate your disparate cloud-based, SaaS and on-premise applications with Quepixon's new QueCloud, easily and without coding. This can be done using QueCloud, says Quepixon, at low cost and with a great deal of automation while avoiding common drawbacks, such as long service engagements, manual processes and the constant wholesale movement of entire datasets. QueCloud cites its Application Software Blades technology that provides automatic connection to data sources, complemented by data virtualization that brings all of the structure of the data sources into the system instantly. From there, the entire data integration process merely involves checking off boxes to align the data and the business rules desired. Once applications are integrated, QueCloud keeps critical customer, product and financial data consistent between applications. Absent from the process are typical elements of ETL technology, such as coding, SQL, complex widgets and wire diagrams.

www.quepixon.com

Thomas Lah, Steve O'Connor and Mitchel Peterson's *Building Professional Services* (Prentice Hall)

Ready to take your world-class skills out of the cubicle and into the big, brutal world? Before you take that jump, sit down long enough to digest the new book *Building Professional Services: The Sirens' Song* from Thomas Lah, Steve O'Connor and Mitchel Peterson. There certainly is money to be made, but the pitfalls are also deep and treacherous, say the authors. These leading experts present a comprehensive guide to creating professional services organizations, managing them to maturity and delivering quality services while earning superior margins. Readers will learn how to adopt a complete, practical framework for delivering a full spectrum of professional services—from support and education services to managed, consulting and productized services. Besides emphasizing the key factors that drive success—revenue, references and repeatability—the book covers frameworks for organization, project delivery, solutions development and operational infrastructure.

www.informit.com/store



Please send information about releases of Linux-related products to newproducts@linuxjournal.com or New Products c/o Linux Journal, PO Box 980985, Houston, TX 77098. Submissions are edited for length and content.

Fresh from the Labs

Blu-ray Special

This month's article is in a different format from usual; normal service will resume next month. I recently scraped together some cash to buy a Blu-ray drive, and like every other user, I found there was no immediate way to play the disc other than reboot into an OS I won't mention. Therefore, this month, I cover Blu-ray-specific projects, highlighting individual parts of what later will become a whole in basic playback of Blu-ray discs—I hope.

libbluray—Blu-ray Playback Library

www.videlolan.org/developers/libbluray.html

According to the libbluray Web site: "libbluray is an open-source library designed for Blu-ray discs playback for media players, like VLC or MPlayer. This research project is developed by an international team of developers from Doom9."

The Web site lists the project's features as follows:

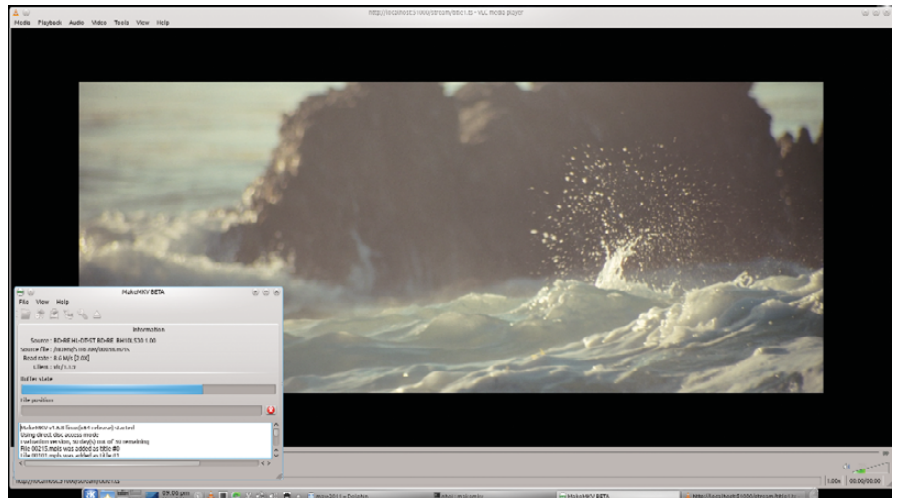
- Portability: currently supported platforms are GNU/Linux, Windows and Mac OS X. Dependencies are very limited.
- Freedom: libbluray is released under a Free Software license, ensuring it will stay free.
- Features: the library integrates navigation, playlist parsing, menus and BD-J.
- Legal: libbluray is DRM-circumvention-free, and thus, safe to integrate in your software.
- Fun: libbluray is a fun-to-hack-on project in its starting phase.

Installation To install libbluray, you need to grab a binary from somewhere (like the Ubuntu Personal Package Archive, for instance), or you need to grab the source. To get the source, you need to have git installed. Once you have git, enter the following command:

```
$ git clone
git://git.videlolan.org/libbluray.git
```

Once that downloads, change into the new directory and compile like so:

```
$ cd libbluray
$ ./bootstrap
$ ./configure
$ make
```



I'm watching a Blu-ray movie under Linux. How? Read through to the end, but I guarantee you won't like it.

If your distro uses sudo:

```
$ sudo make install
```

If your distro uses root:

```
$ su
# make install
```

The main note here is that "Most commercial Blu-ray are protected by AACS or BD+ technologies, and this library is not enough to play back those discs". This is simply a library, and you'll need a player like MPlayer or VLC compiled against it. This is the first of several libraries I'm going to cover, so let's move on to the next.

libaacs—AACS Reading Library

www.videlolan.org/developers/libaacs.html

To quote the libaacs Web site:

libaacs is a research project to implement the Advanced Access Content System specification. This

research project provides, through an open-source library, a way to understand how the AACS works. This research project is mainly developed by an international team of developers from Doom9. NB: this project doesn't offer any key or certificate that could be used to decode encrypted copyrighted material.

According to the project page, libaacs features the following:

- Portability: currently supported platforms are GNU/Linux, Windows, Mac OS X. The main dependency is libgcrypt for all cryptographic functions.
- Freedom: libaacs is released under a Free Software license, ensuring it will stay free.
- Legal: libaacs does not include any key or certificate and respects copyright.
- Fun: libaacs is a fun-to-hack-on project

in its starting phase.

Installation The installation follows the same procedure as that for libbluray. If you're grabbing the source, enter:

```
$ git clone
git://git.videolan.org/libaacs.git
```

Change into the new directory with the following command:

```
$ cd libaacs
```

Then, repeat the next few lines from my libbluray installation instructions.

libbdplus—Purpose Unconfirmed (URL Unknown)

Perhaps the missing link in basic Blu-ray playback, I translated this from a French Web site: "libbdplus is a library for decrypting protected BD + Blu-ray discs. Currently in development, the project will certainly be hosted by the VideoLAN team if legal clarification is achieved."

However, finding this library seems to be a mystery in itself, which I'm guessing may be kept under wraps for legal reasons. On the first page of Googling, I came across this e-mail from the MPlayer archives: "Can you send me a link to the hidden libbdplus git? Or the source snapshot? Thanks."

Indeed, I've seen references on other mailing lists referring to this trio of libraries being the key to basic playback. Although I couldn't confirm it with my one-and-only Blu-ray disc (and meager download limit), it seems that unencrypted discs can be played back with the first library, and these last two are needed for encrypted discs.

DumpHD—HD Decrypter forum.doom9.org/showthread.php?t=123111

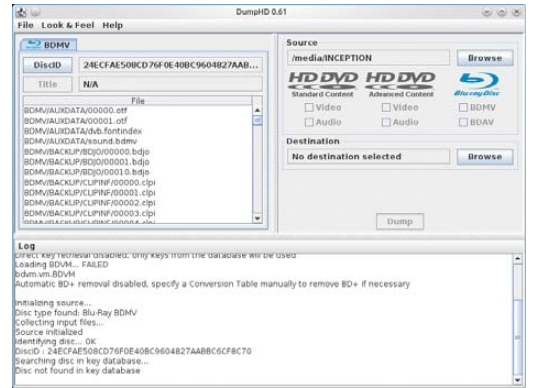
While I must stress from the outset that I couldn't get this to rip my only disc, it looks to be invaluable nonetheless. Also developed by the Doom9 folks, like libbluray and libaacs, DumpHD is a sleek GUI program for decrypting HD-DVD and Blu-ray discs and dumping the files onto your hard disk. According to the documentation: "DumpHD is a HD-DVD/Blu-ray decrypter based on the idea of BackupHDDVD by muslix64. The goal of this program is to make a perfect backup without any

traces of AACs."

Also according to the documentation, here are its key features so far:

- Dual-Core supported decryption of EVO/M2TS files (for hard-disk-to-hard-disk speed records).
- Support for every pack type of an EVO (including in-place decryption of ADV_PCKs, excluding Sequence Key Sections).
- Decryption of every ARF protection type.
- Multiple files (currently CLI only) or complete disc mode.
- Usage of a key database to get the decryption keys or direct retrieval of the keys off the source disc.
- Supports HD-DVDs for Standard/Advanced Content (but not both on the same disc), Blu-ray ROM BDMV.
- Experimental Blu-ray recordable support (with multiple CPS Units, BDMV, BDAV with Aux Directories and Thumbnails).
- Automatic BD+ removal using the BDVM Debugger or manually by supplying a correct Conversion Table (currently CLI only).
- Streaming output of EVO/M2TS files to stdout.
- Very much console output for free.
- GUI.

Although I don't have space to cover the installation properly, I did find this program in the Ubuntu repository, and as this is a trickier program, I hope it'll be in your distro archive too. Using DumpHD at least is easy. The only real input required from the user is to browse for the disc's



This snazzy-looking program is DumpHD. It decrypts HD DVD and Blu-ray discs onto your hard drive.

path (/media/INCEPTION, in my case), as well as to browse for a destination. From here, you need to check only whether you want audio, video or both, and then click Dump.

Part of my problem appeared to be something to do with external libraries and helper applications, and their installation and pathing. So far, the documentation available is strewn haphazardly across README files, forums, man pages and so on, all as separate elements that are hard to track down. The same applies for the actual program files themselves, which is not fun. If anyone wants to help these guys with packaging, hosting, documentation (anything to make this a cohesive package), I swear I will cover this program again but in proper detail.

MakeMKV

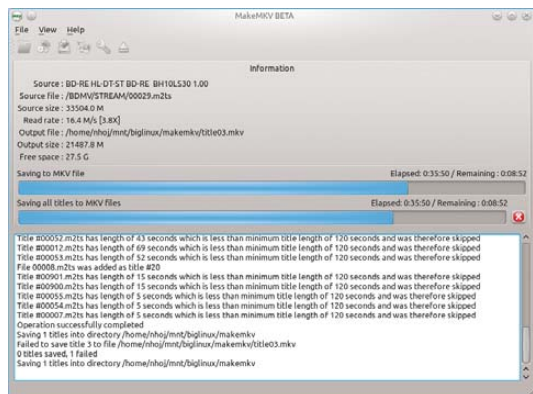
www.makemkv.com

If you've reached this point of the article filled with despair and just want to watch a film with your new drive, there is one last resort: MakeMKV.

Before I continue, this is shareware—that horrid software model from the 1980s and 1990s. Nevertheless, installation is quite easy, and its GUI integration is simply excellent (it managed to include clever program choices when I right-clicked on the titlebar).

Still, you get only a trial period for Blu-ray functionality, at which point you have to pay for the program. Okay, you're thinking,

While I must stress from the outset that I couldn't get this to rip my only disc, it looks to be invaluable nonetheless.



MakeMKV: this expensive bit of shareware always gets the job done in ripping Blu-ray discs to hard drive, but at quite a price.

that doesn't sound too bad. How much is it? It costs 50 euros. Last time I looked, that's about eleven-billion in US dollars.

I might pay 50 euros for a commercial software player, *maybe*, but for a ripper, no, sorry. Anyway, I have 30 days of trial usage, and I'm desperate just to watch a film with my new drive, so let's push on.

Head to the Web site's Download page, where at the top, it has a link to a forum for Linux users. There you'll find all the information with excellent instructions for installation, with links to two tarballs you'll need. The installation went perfectly for me, and all I had to do was download and extract the tarballs, open a terminal in each folder, and copy and paste some commands.

The GUI is intuitive and largely self-explanatory, but I urge you to make sure the output folder is on a partition with enough free space to accommodate these

Hopefully, we as a community can focus our efforts in whatever form we can—hosting, coding, documentation and so on—and reach the same point of maturity as projects like libdvdcss.

massive files. Ripping something that's 30Gb makes something that's almost 30Gb as well. I guess this is to make the quality as 1:1 as possible, but I couldn't find anything to lower the bitrate for storage (half the point of ripping).

Maybe I missed something in the GUI, but I don't think so.

When you get to the ripping stage, you need to choose which title(s) to rip, and be sure you choose the right ones on the first go. After a big wait, your MKV should be ready to watch. I hope you chose the right title!

Still, this isn't watching Blu-ray. It's ripping it. And, it takes some time too. "Can we watch the film at your place, honey?" "Certainly. I'll just rip it first and guess at which titles are the film and the special features, and hope I get it right on the first go." Less than cool.

Note: just after writing that section, I found a guide on how to stream from MakeMKV to VideoLAN. This saves on ripping time, but it also has jittery performance, and when I tried it with MPlayer, it was very glitchy. For those wanting to try their luck with this alternative, the guide

is available at themediaviking.com/software/bluray-linux.

And, that's about as much coverage as I'm willing to give a shareware program.

Conclusion

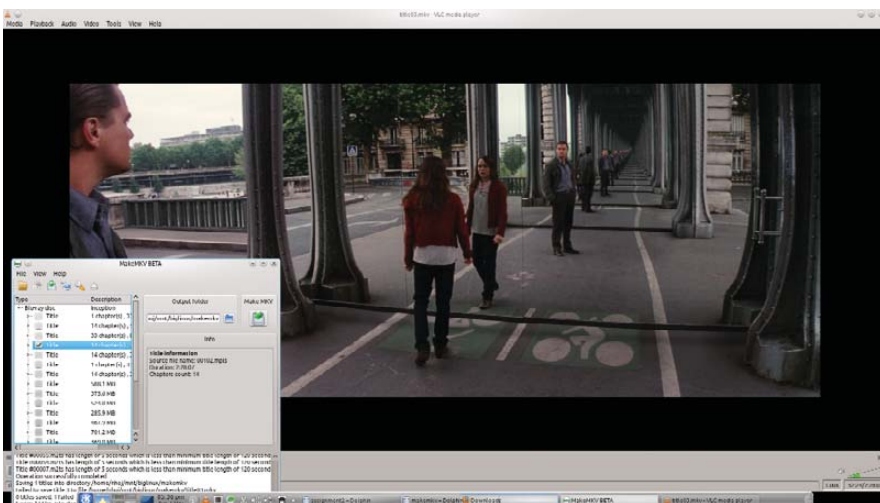
Linux and Blu-ray may be in a bit of a sorry state for now, but (encrypted) DVD also had many of the same problems in its first few years. When I first tried DVDs under Linux (longer ago than I care to remember), projects like xine were in their infancies. Only unencrypted DVDs (of which there weren't many around) could be played, and playback methods were primitive. No menus or navigation, no chapters and individual .vob files were played one at a time. The Next Chapter button in xine would skip to the next .vob file (if memory serves correctly).

Now we take Linux DVD playback for granted—at least for those of us in countries where libdvdcss isn't a problem. And, unless I'm using something OSS like VideoLAN, playing DVDs under Windows nowadays is actually more of a hassle than when I'm in my native Linux. So what about Blu-ray?

Currently, the projects for decrypting and playing Blu-ray discs are spread around the Net, disparate and unconnected. The streaming method mentioned at the end of the last section of this article shows it is at least possible to play encrypted discs, albeit in a bastardized form. Hopefully, we as a community can focus our efforts in whatever form we can—hosting, coding, documentation and so on—and reach the same point of maturity as projects like libdvdcss.

If I've made any mistakes along the way, I implore you to write in and correct me! I'd rather provide some kind of help with this issue than contribute misinformation. ■

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



The final product: a 27-gig mkv rip that I can finally watch. Worth it? Not sure.

Brewing something fresh, innovative or mind-bending?
Send e-mail to newprojects@linuxjournal.com.

The newly updated **LINUX JOURNAL ARCHIVE** is here!



The archive includes **all 200 issues of *Linux Journal***, from the premiere issue in March 1994 through December 2010. In easy-to-use HTML format, the fully searchable, space-saving archive offers immediate access to an essential resource for the Linux enthusiast: *Linux Journal*.

**A
CONFERENCE
PRIMER
WITH
GARETH
GREENAWAY**

**Can't make it to a LinuxFest this year?
Host your own!**

SHAWN POWERS



FEATURE A Conference Primer with Gareth Greenaway

Every time I attend a Linux conference, I'm reminded of one thing: I'm so glad I'm not in charge. I have a difficult time getting my family out the door to go to school on time every morning. The thought of arranging dozens of sessions, hundreds of volunteers and thousands of attendees is just overwhelming. I've always been quite certain every conference has a "nervous-breakdown room" where the people in charge go to rock back and forth in a fetal position a few times a day.

Although that might be true for many people, as it would certainly be for me, my opinion changed when I attended my first Southern California Linux Expo (SCALE) a couple years ago. There certainly was plenty of chaos to be had, but in the hallway outside the vendor area, I met Gareth Greenaway. Gareth was casually talking to the group of us from *Linux Journal*, sipping on a cup of coffee and wearing flip-flops. When I learned that he was one of the people in charge of SCALE, I figured he already had cracked and had just come from a fetal-ball-rocking session in the nervous-breakdown room. As it turns out, he's just really good at what he does. When we decided to do an issue on Community, I thought he would be the perfect person to interview about running a Linux conference. Thankfully, in his typically calm and casual way, he agreed.

SP: What is your job, specifically, when it comes to SCALE every year?

GG: Currently, I fill two roles within the SCALE organization. The first role is Conference Operations. This role is the primary contact between the venue and the SCALE "team". In this capacity, I also deal with several of the vendors that make the show happen, such as the booth decorators.

The second role is that of Community Relations. This role is responsible for working with the community to get the word out about the show, working directly with the speakers' chairs to draw speaker submissions from community groups and projects, and directing the Community Relations committee's efforts to recruit dotORG exhibitors from the Free and Open Source community.

SP: Given a one-year calendar, could you describe the milestones and activities that take place to prepare for SCALE? If some things are planned more than a year in advance, mention those as well.

GG: The big milestones for the show include things like the dates for the Call for Presentations, when the Call for Presentations should open and when it should close. Other milestones are when

attendee registration for the show should open, and when the "early-bird" registration should end and the ticket price should increase. We also have several milestones for behind-the-scenes tasks, such as the various committees submitting their budgets for the show, determining the placement of booths on the show floor, signing various contracts and so on.

Planning for the show usually begins around May or June. Around September is when we typically open up the Call for Presentations. In December, we open up the registration system, with the early-bird ticket prices going until around early January. The Call for Presentations closes near the end of December or early January. Around this time, things really get busy for most of the committees. [Note: the conference is usually scheduled to take place toward the end of February each year.]

SP: Most conferences charge an entry fee, accept sponsorships and get money from vendors for having booths on-site. Without divulging specific dollar amounts, what percentage of funding comes from what sources?

GG: Generally, a good portion of the funds

that we need to host the show comes from our generous sponsors. Several of them return each year and have a regular presence on our show floor, and we greatly appreciate their support. We end up providing various discounts for our attendees to attend the show—these range from student discounts to discounts for various user groups and open-source projects.

SP: What is the most expensive part of putting on a Linux conference?

GG: There are a few aspects that could easily qualify for the most expensive part. The first is the show network, which is utilized by our exhibit floor, our show registration system, our show wireless networks and audio/video equipment used for monitoring talks. Each of these areas, including each booth on the show floor, ends up being a separate managed network. The downside is that providing this kind of network infrastructure can be very expensive, while the upside is that SCALE ends up with a rock-solid network.

The other aspect is our audio and visual equipment. We made the decision a few years back to handle the audio and visual equipment ourselves, realizing how expensive renting it directly from the venue can be. Luckily, we had some very talented individuals step forward and fill the need. Although we're handling this element ourselves, it still can end up being quite an expense, especially with the number of speaker sessions that we need to accommodate now. At the SCALE 9x Friday sessions, we had nine concurrent sessions running, each room with its own full A/V setup.

SP: What is the most difficult aspect of planning SCALE? And, what's the most rewarding?

GG: For me, the most difficult aspect is that the show lasts only a few days. We plan things out for almost a full year, and it often feels like it's over too quickly.

A few years ago, I jokingly said that it reminded me of a scene from one of my favorite movies, *Brewster's Millions*. The main character inherits millions of dollars and has to spend it all to get his full inheritance. Shortly after he gets his first inheritance, he has an interior designer design him an office, but it's never quite right. He explains to her that he wants to walk in and say "I want to die in this office!" Finally in one of the final scenes, after he's spent all his money and has

“WE’VE PUT TOGETHER A LIST OF THOSE SPEAKERS WHO ARE TYPICALLY LATE, AND AS THE DEADLINE GETS CLOSER WE SEND OUR GOON SQUADS TO THEIR HOMES, FORCING THEM TO GO THROUGH THE SUBMISSION PROCESS!”

nothing, Brewster walks into the office and exclaims “I want to die in this office.” Immediately after that, all the rented furniture is taken back and the office is torn down. This is always how I feel when SCALE is being packed up.

There are a few aspects about organizing SCALE that I consider especially rewarding. The first being the opportunity to help spread the word about many of the free and open-source projects that aren’t able to be as vocal as some of the others. A few years ago, we invited the creators of the QIMO Project to attend SCALE as a dotORG exhibitor, and they ended up being one of the more-popular exhibitor booths during the show and gained quite a bit of exposure because of being at SCALE.

We’ve also given quite a few people who have never spoken at a conference the opportunity to speak. One of my favorite sessions that we’ve had at SCALE was a session that took place during our Friday sessions a few years back. The daughter of one of our volunteers, Larry Cafiero, and two daughters of Fedora/Red Hat contributor Karsten Wade, gave a talk on what they liked and disliked about free and open-source software. It was a great moment as these three young girls were explaining to a room full of adults, many of whom were developers, what the software should and shouldn’t do.

SP: As someone who speaks at conferences, I know I’m horrible at getting talks submitted in time. I can’t imagine I’m the only speaker with that tendency. Do you find it difficult to arrange quality sessions in a timely manner?

LINUX PRESENTER TIPS

- People like to look at pictures and listen to speakers. I try to keep my PowerPoint presentations simple and with few words. Not everyone presents that way, but it works for me.
- Be excited about your topic, even if you’re not. I just gave a very lively and exciting talk at Penguicon about backups, pretty much the most boring topic on the planet.
- Give 110% to your audience, whether it’s 2,000 people or two people. They came to see you, likely at great expense; give them the respect they deserve.
- Leave some time for questions if possible.
- Leave the room when your time slot is over. If there is a crowd, lead them into the hallway to be respectful of the next speaker.
- Make bullet points short, and elaborate on them when you talk. If you just read your bullet points to the crowd, the crowd can (and will) do it without you. This needs to be bold, highlighted and repeated. Your talk elaborates your bullet points. If you simply read the presentation, it makes for a horribly boring session.
- Don’t be arrogant. Your audience is full of your peers, not your students. Always assume there are people in the audience who know more about the topic than you do (there usually are).
- It’s important to take questions from the audience, but don’t let the questions (or one audience member) dominate the talk. If questions start to get out of hand, tell the audience you’ll take further questions at the end or after the talk.
- Fact-check your presentation. Geeks *love* to correct a presenter, so be careful with generalizations and look for places in your talk where you might be inaccurate. Make sure any sources you use are authoritative and up to date.
- Avoid demos unless absolutely necessary. Screenshots, screen-scrape videos and example commands usually can make your point and are less likely to break in the middle of your talk. Nothing’s worse for a presenter’s reputation or an audience’s patience than a broken demo. If you must use a demo, test it thoroughly and have a backup plan in case it fails. Preparing for things to go completely wrong is something that you *have* to do. I gave a talk on virtualization at Penguicon a couple years ago. I prepared for the talk and ran through the demo at least four times. But on the day of the presentation, the virtual networking I needed for the talk failed to run. I played it off, spent a couple minutes getting comments and tips from the peanut gallery in the crowd on how to troubleshoot it, but then abandoned it, and just went off to talk about what was left in the slide. The talk wasn’t as good as I’d hoped, and it wasn’t what people came to see, but I think it wasn’t a total wreck either, simply because I had enough material and was prepared in the event that it went sideways.
- Practice your talk at a LUG. Linux Users Groups usually are quite grateful to have speakers and are a great place to practice your talk in front of a similar audience before the “main event”.
- Try not to take yourself too seriously. There will be mistakes, flubs and people doing silly stuff during your talk. Be ready to blow off mistakes or joke about them. You can turn a bad moment into something that will make the crowd laugh. They won’t remember the flub, and you come off as very entertaining.

TIPS FOR ATTENDING A LINUX EVENT

- Always plan a backup talk to attend. Sometimes a talk's summary is written months before the presentation, and often presenters will completely rework a talk. Other times, the talk just isn't what you were expecting. Usually you can tell in the first five minutes whether a talk is worth the time, and if you already have a backup talk planned, you'll be able to get there before you miss too much of it.
- Attend the "hallway track". Some of the most rewarding time you'll spend at a conference is the hallway track or the conversations you have with geeks in the hallway outside formal talks. It's definitely worth it to get over any social anxiety and strike up a conversation with fellow geeks to find out what cool things other people are up to.
- All things being equal, choose an expert. Sometimes you'll have a few different talks you want to attend at the same time. If it's difficult to decide, add extra points if one of the talks is being given by either a known good presenter or the absolute expert on the topic.
- Don't be "that guy". There's always the person in the audience who wants to prove he or she is smarter than the presenter and tries to dominate the talk. You'll get better answers and a better presentation from presenters if you actually let them present. If you have detailed questions, most presenters are more than happy to field questions after the talk is over.
- Always throw in at least one "wild card". It's easy if you are a programmer to stay in the developer track, or if you are a sysadmin to stay in the sysadmin track—after all, that's why tracks were invented. Be open to mixing it up though, and choose at least one wild-card talk in a subject you may have only indirect interest in.
- Silence your phone. This goes for your computer too. It's really distracting for everyone to hear that Ubuntu startup sound, a ringtone or the *Angry Birds* noises in the middle of a talk.
- Get your schedule figured out, and plan which talks you want to go to. If you have a smartphone or PDA, put them in your calendar, along with what room they're in. That way, you're not scrambling trying to figure out where you need to be and when.
- If you're on social media, find the event's hashtag and Tweet/Facebook-post with the hashtag. Also, search for that hashtag. You'll wind up finding lots of sidebar events and meeting a whole bunch more folks that way. It's the "hallway track", but virtualized.
- If it's a multi-day event, bring half the clothes and twice the money. Hotel food is expensive, and drink is doubly so.
- Pack a power strip.
- Be ready to have a *lot* of fun.

GG: We've put together a list of those speakers who are typically late, and as the deadline gets closer we send our goon squads to their homes, forcing them to go through the submission process! In all seriousness, it can be difficult, and we've had to resort to extending the call for papers a few times. Usually this works to our benefit though, as we see a flood of submissions for really great talks. Unfortunately, the flood of talks also leaves us with more talks than we can accommodate in the main conference. Fortunately, many of those talks end up being great candidates for the SCALE Friday sessions or the UpSCALE, our version of Ignite.

SP: Who is your dream speaker, and why?

GG: I really have two answers to this one. An obvious choice would be Linus Torvalds. I believe there is an unwritten rule somewhere that your event really can't be considered a true open-source event until you've had Linus speak. I'd imagine he's got a whole blessing he does too, a big ceremony and probably a closing dance number as well.

My other dream speaker isn't anyone in particular but all the people who don't think they have what it takes to speak at a conference. One of the things I love seeing at events is an overflowing room for an unknown speaker.

SP: What is the most stressful part of the planning process?

GG: The most stressful part is the last few weeks leading up to the show—wondering if everything is going to come together and wondering if anyone will show up. This is also when I start dreaming about all the things that could possibly go wrong. Usually the scenarios end up being pretty ridiculous though. One of those scenarios had the show taking place on a farm, with speakers giving talks in a chicken coop.

SP: How many people are involved in putting on SCALE (not including guest speakers)?

GG: There are roughly ten core members that work on the show each year, but that number is growing as more and more people get involved, which is really great to see, because new members bring new ideas and make sure that the show stays fresh and fun. It works very much like an open-source project—someone will see a missing or existing element that needs to be added or needs attention, then he or she jumps in. We've had several people start off simply as volunteers during the show, who are not core members of the team.

SP: How did SCALE start?

GG: Several years ago, I proposed to some

members of the local Linux Users Group that we host a one-day event, with the idea of it being a gathering of the local LUGs in the area. We had speakers from the local LUGs giving presentations on a variety of topics, and exhibitors included LUG members showcasing their favorite open-source projects and a handful of local commercial companies with Linux-based products. We called the event LUGFest and ended up hosting the event four times, every six months.

While planning the fifth event, a member of one of the other local LUGs told me that some students at University of Southern California and University of California Los Angeles and members of the USCLUG and UCLALUG, respectively, were looking into the possibility of hosting a conference on the USC campus. I made contact with that team, and we ended up meeting, then planning out the first SCALE event. The first show was held at the Davidson conference center on the USC campus, and there were roughly 20 exhibitors and two speaker tracks.

Since then, we've hosted the show at various locations, expanded the speaker track to five concurrent sessions, added specialty Friday sessions and expanded the exhibitor floor to more than 100 exhibitor booths.

SP: You must be passionate about Linux to take on something like this. What's your story?

GG: I actually got involved with free and open-source software while looking for a job. I was hunting for a job as a computer programmer and came across one that sounded interesting. I ended up asking about their environment and was told that they used UNIX. I had never heard of UNIX, so I started doing some research and came across two free UNIXes, Linux and FreeBSD. I don't remember exactly why, but I ended up going the Linux route, and I ended up doing a Slackware install. After writing 40+ floppy disks, I had a Linux machine. That definitely was a different time. I can vividly remember the first time I set up an X graphical session—

crouching under my desk, reaching up to the keyboard to hit the Enter key, because I had read horror stories of people blowing up their monitors. I wouldn't say I necessarily miss those days, but I do think it's important to realize and appreciate how far things have come.

Free and open-source software definitely has come a long way. It's given so many people so many opportunities to use and work with a variety of software that they otherwise wouldn't have had access to.

SP: How has the economy dip in recent years affected the turnout of both guests and vendors?

GG: When the economy dipped a few years ago, we definitely saw that reflected in many aspects of the show. For obvious reasons, many sponsors and commercial exhibitors were more reluctant to fund the show and have their employees host a booth. We saw fewer companies out at the show recruiting versus other years as well. One aspect that remains a constant, however, is our attendance. Many of us feared



**VESA-Mountable
NVIDIA® ION2 System**
Fully configured with HDD, RAM.
Features GeForce® Graphics, flexible
storage options, and Wi-Fi.



**Fanless, Rugged
Intel® Core™ Platform**
No fans, no moving parts. Just
quiet, reliable performance.
Full featured; no compromises.

Chris
Assembly Team Manager

Genuine expertise.

www.logicsupply.com/linux

LOGIC
SUPPLY®

LINUX COMMUNITY EVENTS

CHECK TO SEE IF THERE'S A COMMUNITY-ORGANIZED EVENT IN YOUR REGION:

- Flourish—Chicago, Illinois: www.flourishconf.com
- LinuxFest Northwest—Bellingham, Washington: linuxfestnorthwest.org
- Northeast GNU/Linux Fest—Worcester, Massachusetts: www.northeastlinuxfest.org
- Ohio Linux Fest—Columbus, Ohio: ohiolinux.org
- Penguicon—Troy, Michigan: penguicon.org
- SCALE—Los Angeles, California: www.socallinuxexpo.org
- SouthEast LinuxFest—Spartanburg, South Carolina: southeastlinuxfest.org
- Florida Linux Show: www.floridalinuxshow.com

For a more complete list of Linux-related events, including popular corporate-organized events as well as international events, visit www.linuxjournal.com/events.

that with the economy in the position it was that we would see a downturn in the number of people who attended. While it didn't shoot up at all, it certainly didn't plummet, which was a pleasant surprise.

In recent years, as a hopeful sign that the economy is heading back to a more healthier place, we've seen an uptick in both sponsorship and attendance. This past SCALE (SCALE 9x), we saw quite a jump in the number of attendees that came out for the show as well as the number of exhibitors.

SP: Many are wondering if there's a place for a major national show again—a LinuxWorld circa 1998—or if the trend is toward regional community-driven events like SCALE. What are your thoughts?

GG: I think the time for commercially driven events has passed, and now it's the time for community events. That being said, I think having a governing organization for national or international events could be an interesting and important addition. The community events are so successful because of the passion for the subject that the organizers bring to the table. There definitely is a different dynamic when people work on something for a paycheck vs. for the love of it. I'm not against people being paid for their work on free and open-source

software or events, but it shouldn't just be about the paycheck.

SP: How do you arrange the session schedule? Is it your fault when two sessions I want to go to occur at the same time? That said, is there a certain methodology or software that helps arrange tracks to minimize topic overlap?

GG: We do our best to arrange the talks in a way that two talks we think will be popular don't overlap. Unfortunately, we're not always right. We've definitely had some talks that were wildly popular, with attendees spilling out into the hallways and sitting in the aisles. That being said, it's absolutely my fault when there are two sessions that you want to go to occur at the same time.

SP: Share some experiences or details that would be surprising for those not familiar with planning and running a Linux conference.

GG: One thing that always surprises everyone about SCALE is that everyone involved is a volunteer; no one is paid to work on the show.

Advice I would give to others wanting to plan a similar event would be to start small and definitely not try to do everything yourself. When dealing with vendors and any venues, make sure to

get everything in writing. And, if you're holding an event at a hotel, be sure to account for the 20% service charge that gets charged to everything!

One of the most interesting experiences that I've had planning SCALE was the year the show ended up being the same weekend that a certain awards show takes place in Southern California. Before the show began, while hunting around the venue for a missing package for one of sponsors, I ended up in the room where the awards show tickets were being stored. Following the show, buildings surrounding the venue were topped with snipers as a form of security for the awards show.

SP: What advice can you give to someone who is (or readers who are?) considering starting a regional event themselves?

GG: Start small, don't try to do too much the first year.

Make sure roles and responsibilities are well defined and well communicated. Nothing is worse than having too many cooks in the kitchen all trying to do the same thing.

Take chances, and allow others to make mistakes. The event doesn't need to be absolutely perfect, and it won't be.

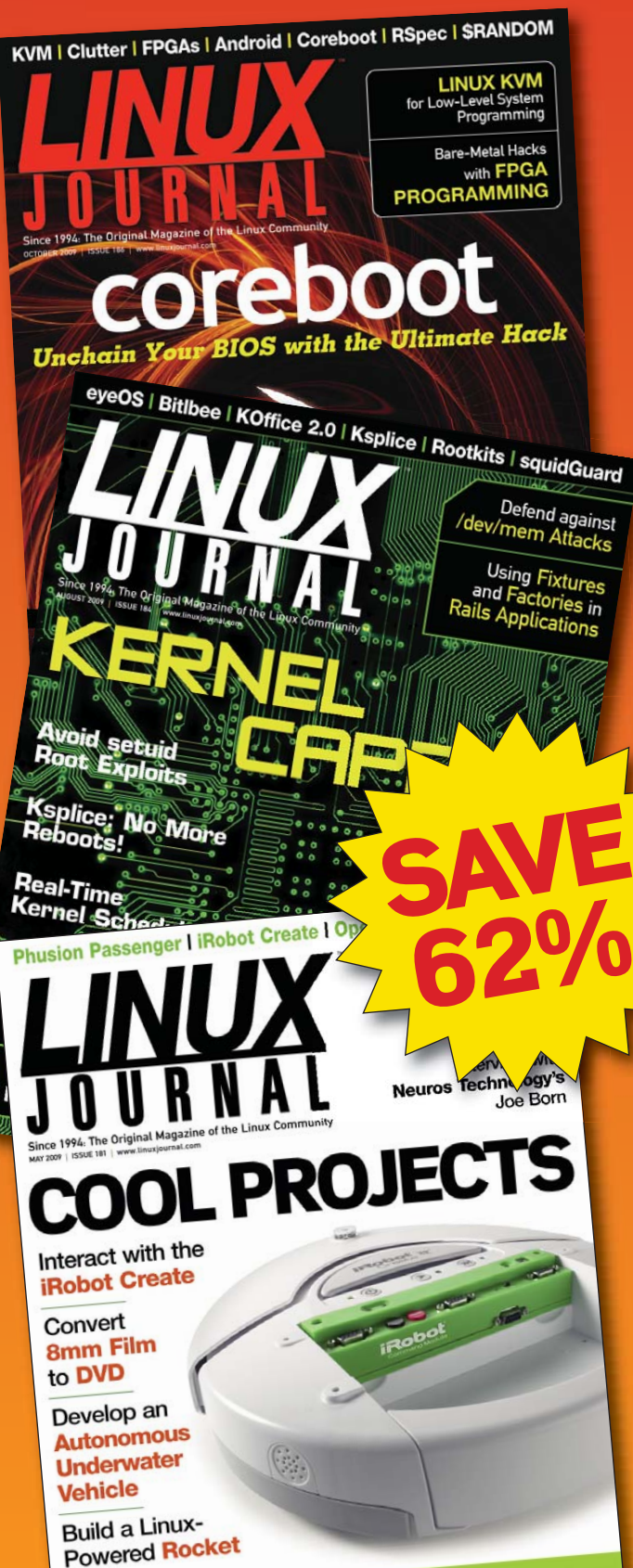
Don't take things personally. People are going to complain; it's just a fact of life. Be polite, and listen to what they have to say. There likely will be some useful information in the complaints.

Open communication is key. It definitely should be a team effort.

SP: As always, Gareth, speaking with you was great. Thanks for taking the time to talk to us. And, to *LJ* readers, if you've ever considered starting a community event, I'd highly recommend attending one of the regional events in your area. Talk to the people who make conferences happen; they even might be willing to give you some advice. And, if you're ever in Southern California in late February, be sure to come to SCALE! Tell Gareth we sent you. If you're planning to speak at or attend a Linux event in the future, see the sidebar to this article for some tips from myself and editors Kyle Rankin and Bill Childers. ■

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.

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



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

Get *Linux Journal* delivered to your door monthly for 1 year for only \$29.50! Plus, you will receive a free gift with your subscription.

SUBSCRIBE NOW AT:
WWW.LINUXJOURNAL.COM/SUBSCRIBE

Offer valid in US only. Newsstand price per issue is \$5.99 USD; Canada/Mexico annual price is \$39.50 USD; International annual price is \$69.50. Free gift valued at \$5.99. Prepaid in US funds. First issue will arrive in 4-6 weeks. Sign up for, renew, or manage your subscription on-line, www.linuxjournal.com/subscribe.

Facebook Application Development

Facebook development, for friends and profit. / Mike Diehl

Do you have a Facebook page? If you do, you are among the 600 million users who actively use the social-networking service. I'll admit I initially resisted the temptation to join Facebook. After all, I've got e-mail, instant messaging and VoIP. What do I need with yet another form of communication? Eventually, temptation got the better of me, and I gave it a try. For the record, I also experimented with MySpace and Twitter, but those sites just didn't seem to do anything for me. These days, every time I go to my computer, I check my servers, my e-mail and my Facebook.

I never thought it would get to this point, but it turns out that Facebook is a surprisingly powerful piece of technology. So far, I've connected with friends from church, college, high school,

previous jobs and some of my son's activities. I've managed to reconnect with my best friend from high school after 20 years. I've even managed to reconnect with my best friend from the third grade! Since I'm 43 years old, I think that's quite an accomplishment. Like I said, it's very powerful technology.

But for me, there's more. I also get links to blogs and news articles from like-minded friends as well as pointers to breaking news that's of interest to me. Many times, I've heard about events on Facebook before I saw them on TV or read about them in the newspaper. Lately, I've been investigating ways to use Facebook to promote my new business, and I'm sharing some of what I've found in this article.

First, let's take care of some of the easy stuff. Everyone's probably been to a Web site and seen a Facebook "Like" button. When users click on this button, they are asked to log in to Facebook, if they're not already. Then, the button click causes a brief message to be added to users' Facebook pages, indicating that they like the particular Web site. All of the users' Facebook friends then can see this message. Assuming that many of one's friends share the same interests, this is a very simple way to get a message out to potentially interested people.

Adding a Like button to a Web site is as easy as embedding a little HTML code into your page. Facebook even has a wizard to help customize the button: <https://developers.facebook.com/docs/reference/plugins/like>.

When I played with this wizard, I was given code that looked like the code shown in Listing 1. This code results in a Web browser displaying something that resembles Figure 1.

Facebook also provides methods that allow a Web site to post to users' Facebook pages as well as add a person or page to their friends list. These methods are documented at <https://developers.facebook.com/docs/reference/dialogs>.

Before I go any further, let me pass on a little tidbit of information I came across while researching this article. Facebook supports a primitive chat function that allows users to exchange interactive messages with their friends. This chat function is based on the XMPP protocol, which is the same protocol that Google Talk and Jabber use. The details are documented at <https://developers.facebook.com/docs/chat>. Essentially, you

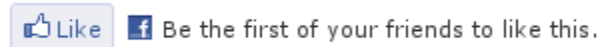


Figure 1. Like Button

the user's full name, you would use something like this:

```
select name from user where uid=1690577964
```

The resulting URL would like like this:

```
https://api.facebook.com/method/fql.query?query=select
  name from user where uid=1690577964
```

Amazingly, your results look like this:

```
<fql_query_response list="true">
  <user>
    <name>Mike Diehl</name>
  </user>
</fql_query_response>
```

FQL requires that you specifically list each of the fields in which you are interested. That means you can't simply use `select * from...` as you would be tempted to do in a language like SQL. So, it's important to know what tables

Adding a Like button to a Web site is as easy as embedding a little HTML code into your page.

need to know your Facebook user name and password. Then, you prepend your user name to `@chat.facebook.com` to form a Jabber ID. When I plugged this information in to Kopete, I was able to chat with my Facebook friends using Kopete. I also was able to get real-time updates when my friends logged in and out of Facebook.

Facebook also provides a powerful method of querying its database for various types of information. Using a language called Facebook Query Language (FQL), you can ask for all kinds of information about a Facebook user, subject to access restrictions. The easiest way to do this is to use the HTTP GET access method. This method allows you to execute a query using a standard HTTP GET. The results are returned in XML format. All you have to do is append your query to the end of this URL: [https://api.facebook.com/method/fql.query?query=.](https://api.facebook.com/method/fql.query?query=)

For example, if you had a user ID number and wanted to find

Listing 1. Like Button Code

```
<iframe src="http://www.facebook.com/plugins/like.php?href=
  www.linuxjournal.com&send=true&layout=
  standard&width=450&show_faces=true&action=
  like&colorscheme=light&font=height=80"
  scrolling="no" frameborder="0" style="border:none;
  overflow:hidden; width:450px; height:80px;"
  allowTransparency="true"></iframe>
```

and columns are available to you. This is well documented at: <https://developers.facebook.com/docs/reference/fql>.

At the risk of duplicating what's already on the Web site, here's a list of a few of the more interesting tables that you can query (from the Web site):

- `comment`: query this table to obtain comments associated with one or more feed comment XIDs.
- `connectioncomment`: query this table to return a user's friends and the Facebook pages to which the user is connected.
- `friend`: query this table to determine whether two users are linked together as friends.
- `like`: query this table to return the user IDs of users who like a given Facebook object (video, note, link, photo or photo album).
- `notification`: query this table to get the notifications for the current session user—that is, any notification that appears on <http://www.facebook.com/notifications.php>.
- `profile`: query this table to return certain (typically publicly) viewable information from a user's profile or Facebook page that is displayed in a story.
- `status`: query this table to return one or more of a user's statuses.

FEATURE Facebook Application Development

- stream: query this table to return posts from a user's stream or the user's profile.
- user: query this table to return detailed information from a user's profile.

You can query against quite a few other tables, and I've not fully explored what information is available. I can tell you that e-mail addresses and phone numbers aren't readily available using this query method. Also, users are able to set permissions that determine what can be retrieved.

Listing 2. Facebook JavaScript Demonstration

```
1 #!/usr/bin/perl
2
3
4 print <<EOF
5 Content-Type: text/html; charset=ISO-8859-1
6
7
8 <http>
9 <head>
10 <title>test page</title>
11 </head>
12 <body>
13
14 <script src="http://connect.facebook.net/en_US/all.js"></script>
15
16 <div id="fb-root">
17 </div>
18
19 <hr><span id="who"></span>
20 </hr>
21 <p>
22 <div id=result>
23 </div>
24
25
26 <script>
27 var user_id;
28
29 var r;
30
31 FB.init (
32 {
33   appId : '206854775749',
34   status : true,
35   cookie : true,
36   xfbml : true
37 }
38 );
39
40 FB.getLoginStatus(t, false);
41
42 function t (response) {
43
44   if (response.session) {
45     r = response;
46
47     var query = FB.Data.query('select name, uid from user
48     ↳where uid={0}', r.session.uid);
49     query.wait(
50
51     function(rows) {
52
53     });
54
55     var query2 = FB.Data.query('select uid1, uid2 from
56     ↳friend where uid1={0} order by uid2', r.session.uid);
57     query2.wait(
58     function(rows) {
59       var i = 0;
60       var friends = "";
61
62       while (rows[i]) {
63         var friend = "<img width='20px' height='20px'
64         ↳src=http://graph.facebook.com/" + rows[i].uid2
65         ↳+ "/picture> ";
66         friends = friends + friend;
67
68         if ((i%20) == 19) { friends = friends + "<br>";
69         i++;
70       }
71
72       document.getElementById("result").innerHTML = friends;
73     }
74   );
75 } else {
76   FB.login(function(response) {
77     if (response.session) {
78       // user successfully logged in
79     } else {
80       // user canceled login
81     }
82   });
83   r = null;
84 }
85
86 </script>
87
88 <p>
89 <fb:like href="http://apps.facebook.com/mikestestpage/"
90 ↳width="450" height="80"/>
91 </body>
92 </http>
93 EOF
94 ;
```

That said, you could use these methods in a server-side program to get all kinds of information about a user. But, the real power comes from being able to perform these queries on the client's computer via JavaScript. (I'll show you how that's done, shortly.)

Before you can do anything even remotely fun with Facebook's JavaScript API, you need to become a developer. Becoming a developer is a simple matter of adding the "developers" application to your Facebook profile and filling out a form, which is available at <https://www.facebook.com/developers>. Once you've filled out the form and agreed to the Facebook terms of service, you can start adding and configuring your new applications. You'll be issued an application key for each application you define. (I'll explain how to use that key shortly.)

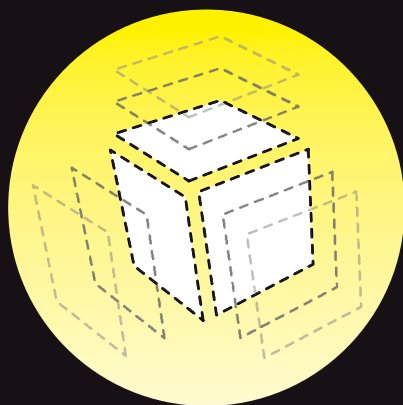
Facebook has received a lot of bad publicity lately because of its often-changing and sometimes poorly implemented privacy policies. But, from reading its terms of service, which all developers have to agree to abide by, it appears that Facebook is really trying to do the right thing. Some items from the terms of service, for example, stipulate that you can't spam your users. That is, you can't create pre-canned messages and blast them out to multiple users. You always must provide an "opt-out" for any action your application performs. You're not allowed to store personal data for use outside your application and so on. Facebook's terms of service all seem pretty rational and are

Figure 2. Create New Application

documented at <https://www.facebook.com/terms.php>.

Now that all the formalities are out of the way, let's define our first application. Start the process by going to <https://www.facebook.com/developers/createapp.php>, and create a new application (Figure 2). Once you've created your new application, you have to configure it.

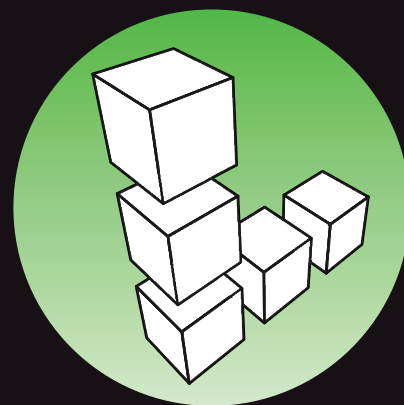
There are a half-dozen pages of configurable options, so I won't go into too much detail with them. Most of them are fairly intuitive and well documented. As long as you understand the process that Facebook performs when a user accesses your application, you won't find this very difficult. Because understanding the process is critical, I'll discuss it a bit before looking at some more code.



Develop.



Deploy.



Scale.

Full root access on your own virtual server for as little as \$19.95/mo

Multiple Linux distributions to choose from • Web-based deployment • Five geographically diverse data centers • Dedicated IP address • Premium bandwidth providers • 4 core SMP Xen instances • Out of band console access • Private back-end network for clustering • IP fail-over support for high availability • Easily upgrade or add additional Linodes • Free managed DNS

For more information visit www.linode.com or call us at 609-593-7103



linode.com

FEATURE Facebook Application Development

The key concept is that all Facebook applications occupy a space on the Facebook interface known as a canvas. The canvas is where your application's content will be rendered and is embedded inside the rest of the Facebook look and feel. Essentially, a Facebook application developer simply uses the Facebook JavaScript API and server-side code to perform the application's function. Thus, you can write your server-side code in any language you like.

When users access your application, they use a URL that looks like `http://apps.facebook.com/APP_NAME`, where `APP_NAME` is the name given to your application. So, if I created an application called `miketestpage`, it would be accessible as `http://apps.facebook.com/miketestpage`. You'll notice that this URL points to a Facebook page. When the user's browser loads this page, Facebook sends a page back to the user, as expected. However, Facebook also embeds the content of your application's canvas code. The end result is a seamless integration between the Facebook user experience and your program code.

To keep this from getting too abstract, let's look at the code shown in Listing 2.

This is a simple Perl script, but it's meant to be as language-agnostic as possible. All it does is output some static HTML. Lines 1–13 are simple boilerplate that produce a valid HTTP and HTML header. Line 13 loads the Facebook JavaScript API. I discovered that my application didn't work unless I included the code in lines 16 and 17; Facebook must use that DOM container internally. The rest of the code is the fun part.

Lines 19–23 create the simple application's layout by creating a `who` and a `result` DOM container. The rest of the code is devoted to populating those containers. It's all done in the JavaScript starting on line 27.

The key concept is that all Facebook applications occupy a space on the Facebook interface known as a canvas.

Lines 31–38 are where you initialize the Facebook API by passing it an object that contains, among other things, your application's unique identifier. Line 40 calls the `getLoginStatus` method and passes it two parameters. The first parameter is a callback function that gets called when this method returns. The second parameter has to do with authentication caching; I found that if I set it to `True`, my callback was called twice, which didn't work very well.

The main code is in the callback function in lines 42–83. Line 44 checks to see if the user is logged in. If not, the `else` statement block on lines 73–82 is executed. In this simple application, users will be given the opportunity to log in and then must reload the page. More sophisticated applications would reload the page for them, of course.

That leaves lines 45–71. Line 47 creates an FQL query, as discussed earlier. All this query does is get the user's name. Lines 49–53 use the `wait` method to get the results and process them using a callback function that you pass implicitly as a parameter. Here, all you're doing is creating a string value with the query results and placing it inside the `who` DOM container. The fact that this method is called `wait` is a bit of a misnomer. This is *not* a blocking call. This function will be called asynchronously as the results are returned to the client.

In lines 55 and 56, you set up another query and process the results in another callback function. Only this time, the query selects all of the current user's friends by UID number. The callback function,

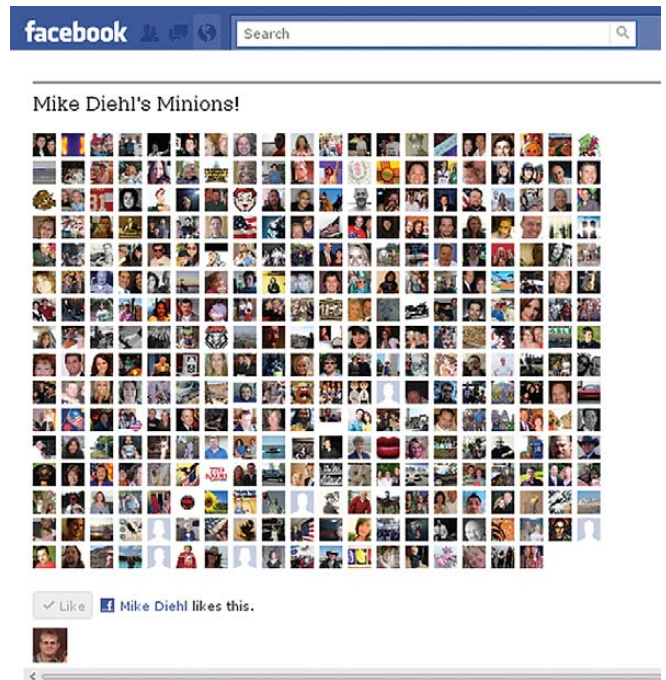


Figure 3. My Minions

in lines 57–70 loops over that result set. Each time through the loop, you build up an `IMG` tag that points to the thumbnail picture of each of the user's friends. On line 65, you ensure that you start a new row

of pictures after every 20 pictures are rendered. Finally, you send all of the image tags, en masse, to the `result` DOM container. The browser then begins to load the images.

And, that's the bulk of the application, except for that little piece of magic on line 89. The `<fb:like>` tag is a tag that Facebook interprets before it sends the results to your application's canvas. In this case, you're asking Facebook to provide you with a Like button that your users can press to indicate to their friends that your program is worth trying. There is an entire class of such tags, known as the Extensible Facebook Markup Language (XFBML), and I've barely scratched the surface of what's available. This article was meant as a teaser to let you know that this functionality is available.

Figure 3 shows the results of this simple application.

As you can see, Facebook has a huge user base and provides many powerful ways to interact with those users. I've demonstrated a couple neat tricks you can use with your own Web pages, as well as provided an introduction to the Facebook Javascript API. I've also briefly shown how to integrate your application into the Facebook framework. I hope this has been fun, and I hope it piques your interest in Facebook application development. ■

Mike Diehl operates Diehlnet Communications, LLC, a small IP phone company. Mike lives in Albuquerque, New Mexico, with his wife and three sons. He can be reached at mdiehl@diehlnet.com.

20th USENIX SECURITY SYMPOSIUM

San Francisco, CA • August 8–12, 2011

Join us for a 5-day tutorial and refereed technical program for researchers, practitioners, system administrators, system programmers, and others interested in the latest advances in the security of computer systems and networks.

The Program Begins with 4 Days of Training Taught by Industry Leaders, Including:

- Richard Bejtlich on TCP/IP Weapons School 3.0 (2 Day Class)
- Rik Farrow on SELinux—Security Enhanced Linux
- Jim DelGrosso on an Overview of Threat Modeling
- SANS on Hacker Detection for Systems Administrators (2 Day Class)

And Continues with a 3-Day Technical Program Including:

Keynote Address

Charles Stross, Hugo award-winning author, on “Network Security in the Medium Term: 2061–2561 AD”

Paper Presentations

35 refereed papers that present new research in a variety of subject areas, including securing smart phones, understanding the underground economy, and privacy- and freedom-enhancing technologies

Plus:

- Invited Talks
- Panel Discussions
- Rump Session
- Poster Session
- Birds-of-a-Feather Sessions (BoFs)

Co-Located Workshops Include:

EVT/WOTE '11: 2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, August 8–9, 2011

CSET '11: 4th Workshop on Cyber Security Experimentation and Test, August 8, 2011

FOCI '11: USENIX Workshop on Free and Open Communication on the Internet, August 8, 2011

WOOT '11: 5th USENIX Workshop on Offensive Technologies, August 8, 2011

HealthSec '11: 2nd USENIX Workshop on Health Security and Privacy, August 9, 2011

HotSec '11: 6th USENIX Workshop on Hot Topics in Security, August 9, 2011

MetriCon 6.0: Sixth Workshop on Security Metrics, August 9, 2011

Stay Connected...

 <http://www.usenix.org/facebook>

 [#SEC11](http://twitter.com/usenix)



Register by July 18 and Save!

www.usenix.org/sec11/lj

Linux Standard Base

State of Affairs

An overview of the LSB, the state of LSB specifications, tools, compliance process and lessons learned.

**JEFF LICQUIA,
STEW BENEDICT and
VLADIMIR RUBANOV**

Linux is a dynamic and exciting platform, and its presence in the marketplace is now an undeniable fact. But, to an outsider, it can be tempting to ask “which Linux?” Given the several leading implementations and several variations throughout the world, it’s easy to wonder how the platform maintains any consistency.

This is the problem that motivated the creation of the Linux Standard Base (LSB). By reducing the differences between individual Linux distributions from the application developers’ point of view, the LSB greatly reduces the costs involved with porting applications to different distributions, as well as lowers the cost and effort involved in after-market support of those applications.

For many independent software vendors (ISVs), supporting Linux also becomes a question of “which one?” Either they choose a particular Linux vendor/release and build, test and support that one (or several). Or, they build against an older distribution, hoping to pick up only “stable” interfaces, shipping their own builds of known problem libraries and wish for the best. By defining a known, consistent set of interfaces, the LSB gives ISVs a better chance of building an application that will work across multiple distributions.

The LSB consists of the following key components:

- Specification: a set of requirements that compliant applications and distributions must meet.
- Tools: a set of tools, tests and informational systems that help develop the LSB in a collaborative manner as well as help automate the certification process for applications and distributions.
- Certification: a process to ensure the two-way promise of platform/application compatibility using the tools and the specification.

The LSB is a core standard for the Linux operating system that encourages interoperability between applications and the platform. Figure 1 shows the components and processes of the LSB.

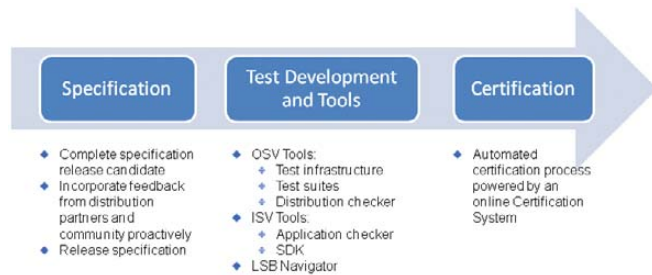


Figure 1. LSB Components and Processes

The LSB specification defines a common set of elements across multiple Linux distributions that covers the following:

- Packaging and installing guidelines.
- Shared libraries and their interfaces.
- Configuration files.
- File placement (Filesystem Hierarchy Standard = FHS).
- System commands.

Because it is a binary specification, the LSB is divided into both general and processor-specific components. The following computing architectures now are supported:

- x86 (IA32).
- x86-64 (AMD64/EM64T).
- Intel IA64 (Itanium).
- IBM PPC 32.

- IBM PPC 64.
- IBM 31-bit S/390.
- IBM 64-bit zSeries.

By providing a number of guarantees about a typical Linux distribution, LSB eases the burden for ISVs wanting to write applications for Linux and support users on their preferred distributions.

LSB Tools

The LSB workgroup provides a number of tools to support the specification. The most important ones are discussed below, and Figure 2 presents a general overview of the tools.

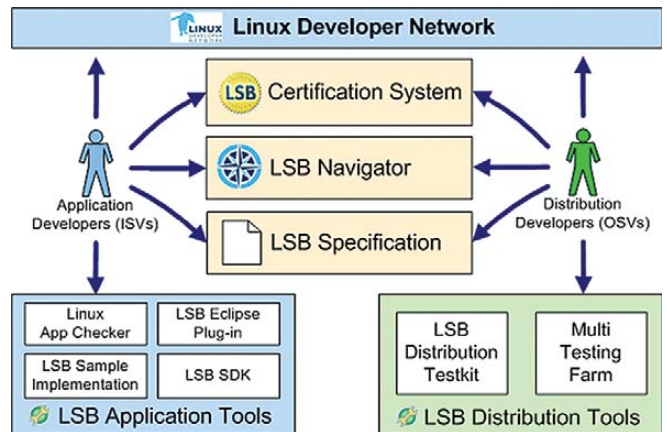


Figure 2. LSB Tools

The **LSB SDK** allows application developers to build their applications in a special environment to ensure that all external dependencies of the resulting binaries meet the LSB requirements. The LSB SDK includes special header files that contain only LSB-defined functions and data types, stub libraries that export only LSB-defined symbols, and compiler wrappers `lsbcc` and `lsbcc++` that launch underlying GCC components with proper settings to use the special headers and stub libraries. Using the LSB SDK is easy—one simply should replace calls to GCC with calls to the compiler wrappers (for example, by changing `CC/CCX` variables). Special `.pc` files are provided for those developers using `pkg-config`. Also, integration of the SDK with Eclipse is supported via the LSB Eclipse Plugin, which adds additional project types (LSB executable, LSB shared and static libraries) to the Eclipse environment and allows one to manage all the settings in a visual way.

The **Linux App Checker** is used by ISVs and application developers to check their applications for portability across multiple Linux distributions. The tool is not limited to checking just LSB requirements; it also enables general portability analysis based on an internal knowledge base of distribution contents. Currently, the knowledge base contains information about 70 different distribution versions. App Checker accepts as input a set of components comprising an application’s package: binaries and `.sos`, possibly in various directories or/and packed in `.rpm/tar.gz` (arbitrary mix) files. Results of the analysis are visualized in interlinked HTML reports. For example, external dependencies (libraries and interfaces) of the application as a whole (internal dependencies between components are excluded) are visualized with the info of the “portability

FEATURE Linux Standard Base: State of Affairs

degree” for each dependency and recommendations on how to improve portability. An interesting aspect is that the tool can differentiate between required libraries (registered as DT_NEEDED records in ELF) and libraries actually used by the application.

The **LSB Distribution Checker** is a framework for distribution developers and OSVs (operating system vendors) to check their systems’ conformance to the LSB. It is actually a whole stack of software components. The bottom layer consists of actual tests that check specific requirements for particular components in the target system under test. The majority of these tests are runtime unit tests, but there also are special checkers, including static verification mechanisms. The middle layer is the framework to control test execution and results collection. And finally, the top layer is the user interface over all the components. The browser-based interface supports “one button” execution of all certifications tests. Alternatively, the user can select specific tests to run, which can be useful for upstream developers interested in testing only particular components. Note that the LSB Distribution Checker can be used not only for LSB compliance testing, but also as a general QA testing framework. During development of the tests and their regular execution, the LSB workgroup has identified hundreds of bugs in various distributions and upstream components, which were reported and acknowledged by the upstream and distribution developers.

The **LSB Navigator** is an integrated on-line informational system over a corresponding database that holds structured information about LSB elements, about the Linux ecosystem, as well as service data used in the standardization analysis and decision making. Correspondingly, one can distinguish three parts of the Navigator:

1) Interactive on-line version of the LSB specification, which might be much more useful in a practical perspective compared to the plain specification text. It provides searchable and browseable information about standardized and non-standardized Linux modules, libraries, interfaces, types and so on. For example, given an interface name, one can find out the following in two clicks:

- Whether the interface is in LSB.
- Recommendations on using the interface or any of its alternatives.
- Direct links to the interface’s documentation.
- What libraries in which distributions provide the interface.
- How many registered applications use the interface.
- Which open-source tests check the interface (can be used as interface usage examples).

2) Analytical data about popular Linux distributions and applications, for example:

- Interface “popularity” lists—how many applications use each interface (separately for LSB and non-LSB interfaces).
- See what external libraries and interfaces are required and actually used by popular applications.
- See which elements modern Linux distributions consist of (particular versions on various hardware platforms).

- Analyze and compare distributions (statistics on Linux elements in each distribution, which distributions provide/miss a particular version of a component, library, command, interface and so on).

3) LSB workgroup services for structuring standardization process and decision support, for example:

- Statistics on LSB evolution (which element appeared/disappeared in which LSB version).
- Analytical data on which popular elements (used by numerous applications and provided by majority of distributions) are not yet in LSB—this serves as a basis for expanding the standard.
- Service data like the test coverage and documentation coverage of particular libraries.

All the tools are integrated—there are links in the Application and Distribution checkers to pages in the Navigator. Also, the checkers are integrated with the Certification system and allow one to submit test data to the Linux Foundation for certification, right from within the checkers.

LSB Certification

LSB certification is a two-way promise of platform/application compatibility. For distributions, being certified is a guarantee that all required LSB commands and interfaces are available on the platform. For ISVs, it is a guarantee that their applications do not rely on anything not provided by either the LSB or the ISV itself in its application package or dependent packages. Both distributions and applications can be certified.

The vendor submits the test results to the Certification Authority (the Linux Foundation). The Linux Foundation reviews the results and either issues the certification or requests the vendor to correct issues and/or resubmit tests with questionable results. The vendor will need to sign a Trademark License Agreement to use the LSB trademark and pay any necessary fees.

The best way to get started with certification is to use the checkers—the LSB Distribution Checker for OSVs and the Linux

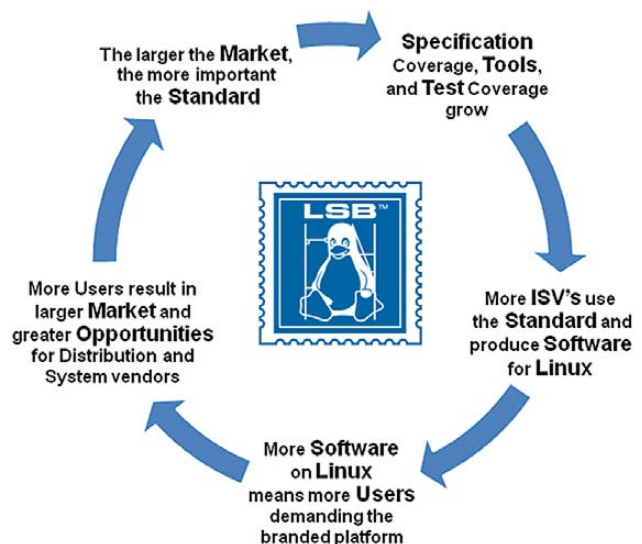


Figure 3. Auto-Catalytic Effect in the Linux Ecosystem

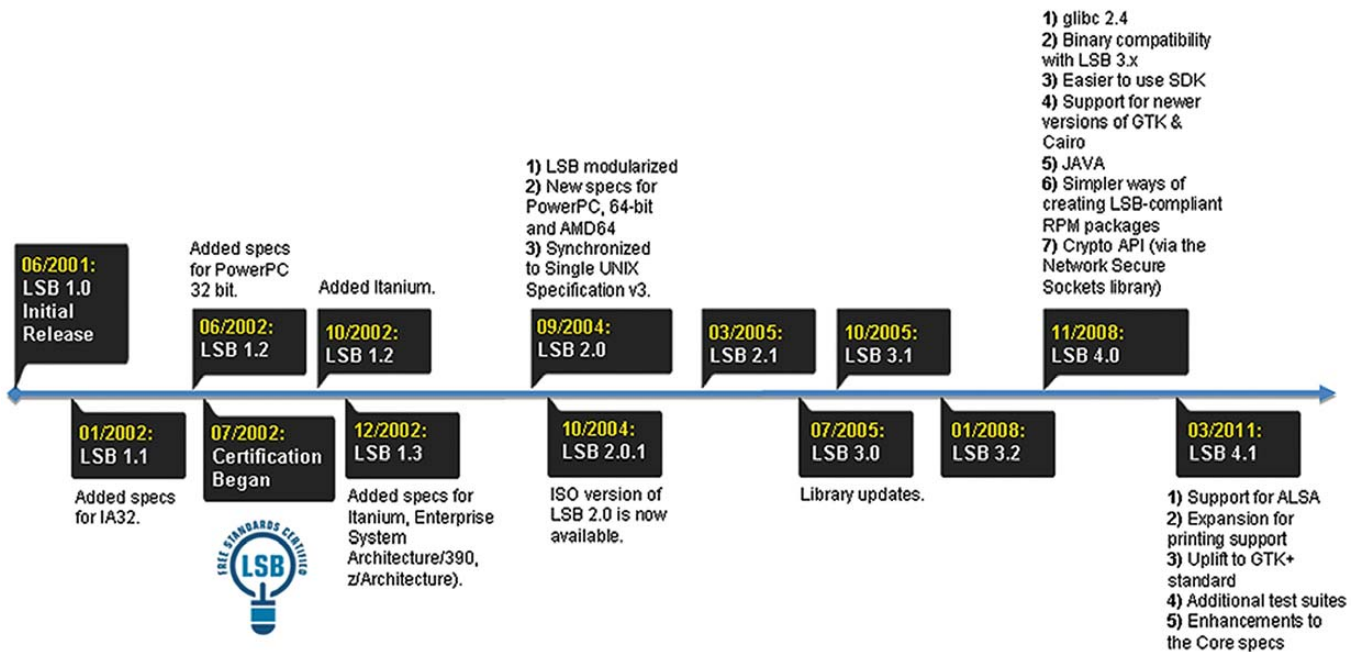


Figure 4. LSB Release Timeline

Application Checker for ISVs. Once the vendor has addressed any issues found by the tools, the checkers can upload the test data to the Certification system to begin the certification process.

Certification puts the application or distribution on the Linux Foundation's official certification list, and it provides marketing materials for developers to advertise their certification. You also will be allowed to use the LSB trademark/branding in your packaging/advertising.

LSB certification secures a number of benefits for the whole Linux ecosystem:

- App vendors need to do only one port.
- Distro vendors get more apps ported to the Linux platform.
- End users can choose their distro and their apps without vendor lock-in .

There is an auto-catalytic effect driven by the LSB (Figure 3).

LSB Releases

Figure 4 shows the history of LSB releases. LSB 4.0, introduced a number of major improvements:

- Expanded coverage: seven completely new libraries and around 1,000 new interfaces were added as compared to its predecessor, LSB 3.2. Four completely new test suites and many new tests in the existing test suites expanded the test coverage significantly.
- Version independence in tools: with the advent of backward-compatible standards starting with LSB 3.0, the proliferation of different LSB versions and the collection of corresponding tools and tests to support them, things were becoming confusing. LSB 4.0 introduced version-independent tools and tests that support all LSB versions starting from 3.0, which means users can select which LSB version they target when using the new tools, making it easy to move back and forth on the version stack as needed without the need for re-installation.

- Best-effort dynamic linking: perhaps the largest difference between an LSB-compliant application and a native Linux application involves the use of the LSB dynamic linker. This has been used in the past to resolve difficult compatibility problems, allowing the native environment to differ from the LSB while still supporting LSB applications. LSB 4.0 debuted a new build strategy that allows an LSB application to run using the native environment when the LSB dynamic linker is not present.

Figure 5 shows the list of currently certified LSB 4.0 distributions.

LSB version 4.1, released in March 2011, has the following main changes from LSB 4.0:

- Additional symbols: a number of additional symbols have been added based on ISV demand. These are primarily symbols from glibc and related libraries, with a few new symbols brought in by uplifts of GTK and Cairo.
- ALSA: ALSA has been a trial-use module in previous releases. For 4.1, the workgroup has promoted ALSA to a required component and included both shallow and normal-level tests for the ALSA interfaces.
- Enhancements to printing: printing enhancements include the addition of ipp/HTTP interfaces from CUPS, as well as a GTK uplift that includes the common printing dialog.

Lessons Learned

The LSB workgroup has been working on the standard since the early 2000s, moving together with the evolving Linux ecosystem. We've learned a number of things along the way that are worth sharing with the community:

- Need for upstream documentation: good standards should rely on a good documentation for the elements they define. In the case of LSB, the most burning point is poor upstream documentation of library interfaces. A number of good interfaces could not be included in the standard due to this reason. We would like to encourage upstream developers to pay attention

FEATURE Linux Standard Base: State of Affairs

to the documentation quality for their components as it helps develop consistent applications.

- Test what you document: upstream components should have good functional tests that can check correctness of the component's implementation against the requirements stated in the documentation. Ideal conformance tests should have an explicit linkage between particular checks in the tests and particular requirements in the documentation, thus enabling traceability during testing.

- New methods for developing interface standards: LSB is the biggest interface standard in the world. It embraces dozens

of thousands of elements. Previous interface standardization efforts counted far fewer elements (for example, POSIX includes just about 1,800 interfaces). The huge number of elements makes it mandatory to use special methods and tools for developing the standard; otherwise, it is impossible to manage the complexity. The LSB workgroup, jointly with ISPRAS (Institute for System Russian Academy of Sciences), has developed a number of leading-edge methods and supporting tools to make development of the standard systematic and manageable. These include automatic generation of some parts of the LSB specification and tools, a systematic process and supporting tools for identifying candidates for inclusion into LSB and moving them through the standardization process stages.

- Continuous testing—find problems early: our approach to testing has changed with time. In the past, our focus on testing has been limited to ad hoc QA on our own software and the auditing of results from customer tests. At some point, the LSB workgroup started running its own regular tests of both the enterprise and community distributions, as well as development versions of those distributions, and it has been filing bugs with distributions and upstream projects for found problems. These have resulted in quick deployment of fixes before they reach the public, both of our tests and of the products being tested.

LSB and the Community

The relationship between the LSB and the greater Linux community













Date of Certification	LSB 4.0 - Product Certified	Architectures	Company	
06/16/2009	Ubuntu 8.04	x84, x86-64	Canonical	
06/18/2009	SUSE Linux Enterprise 11	x86, x86-64, IA64, PPC32, PPC64, S390, S390X	SUSE LINUX Products GmbH	
07/16/2009	Ubuntu 9.04	x86, x86-64	Canonical	
08/07/2009	Mandriva Enterprise Server 5	x86, x86-64	Mandriva	
10/16/2009	Oracle Enterprise Linux Server 5.3	x86, x86-64	Oracle	
01/19/2010	ATOS LFS LC6	x86_64	ATOS Worldwide	
02/02/2010	Linpus Lite 1.3	x86	Linpus Technologies Inc.	
02/05/2010	Red Flag Linux Desktop 6.0	x86	Red Flag Software Co., Ltd.	
04/14/2010	NeoShine Linux Desktop 4	x86	China Standard Software Co., Ltd.	
12/08/2010	BOSS 4.0	x86	Centre for Development of Advanced Computing	
06/28/2010	Kylin Server 3.1	x86-64	Hunan Kylin Information Engineering Technology Co., Ltd.	
10/18/2010	Red Hat Enterprise Linux 6.0	x86, x86-64, PPC64, S390X	Red Hat Inc.	
11/22/2010	Oracle Linux 5 Update 5	x86-64	Oracle	
12/17/2010	Wind River Linux 4.0	x86, x86_64, ppc32	Wind River	
02/07/2011	Oracle Linux 5 Update 6	x86_64	Oracle	
02/12/2011	Red Hat Enterprise Linux 5.6	x86, x86_64, ia64, ppc32, ppc64, s390x	Red Hat Inc.	
3/21/2011	Asianux Server 4	X86, x86_64	Asianux	
1/18/2011	ATOS LFS LC6	x86_64	Atos Worldline	
3/21/2011	Oracle Linux 6	X86, x86_64	Oracle	

Figure 5. LSB Certified Distributions

has been defined largely by the LSB's role as a trailing standard; we document and test for the behavior designed and developed by the community, rather than guiding the community to the behavior we design. On occasion, we act as a clearinghouse when different parties introduce incompatible behavior, but we intentionally avoid "kingmaker" roles even then, preferring to encourage the parties involved to come to a solution on their own.

Another of LSB's community benefits is our recently accelerated testing strategy, which has resulted in the discovery of numerous compatibility bugs at an early stage, allowing them to be fixed before they can make it into the wider world. To our knowledge, we are the only group performing wide multi-distro compatibility testing in the Linux community.

The challenge, going forward, is to communicate the value of the LSB to the Open Source Development community more clearly. To this end, we have been looking at making better use of social media, such as Facebook and Twitter, and in generating more statistics about our contributions.

A number of items on the rolling project plan page could be candidates for future versions of LSB. The workgroup generally finalizes the plan for the next release at a face-to-face session, weighing the demand/return for the development work to add new modules with the available resources we have to work on the LSB. Contributions to LSB are always welcome, and if a contributor was to provide a complete module with specifications and tests, it certainly would increase the chances of being added to the next version of LSB.

Conclusion

The LSB provides a standardized and tested common set of interfaces for ISVs to target, thus addressing the platform fragmentation problem. LSB's goal is to make the ISV's incremental cost of targeting each additional distro approach zero. The LSB Certification mark is available to distros and apps alike. By enabling ISVs to target the LSB rather than any one distro, the LSB enables portability across distributions. Also, the LSB serves as a place where distros, upstream maintainers and ISVs can come together to solve problems and prevent regressions.

Resources

LSB Workgroup Home Page www.linuxfoundation.org/collaborate/workgroups/lsb

LSB Downloads: www.linuxfoundation.org/collaborate/workgroups/lsb/download

LSB Navigator: linuxfoundation.org/navigator

LSB Certification System: www.linuxfoundation.org/lsb-cert

LSB Project Plan: www.linuxfoundation.org/en/ProjectPlan42

Interoperability often is more about having a forum to resolve differences rather than any one approach being correct.

LSB has gone through a long evolution together with the Linux ecosystem. Looking forward to advancing the standard further, the LSB workgroup appeals to the community to participate and contribute to the effort to help secure the success of the Linux platform as a whole.

Acknowledgements

The authors would like to acknowledge all companies, developers, community members, Linux Foundation members and employees, the ISPRAS team and everyone else who contributed to the LSB efforts. Such an industry-wide initiative as LSB would be impossible without the wide support and talented people involved. ■

Jeff Licquia is a software developer at the Linux Foundation. His Linux experience dates to the early days of Linux, and he has extensive experience bringing Linux into the workplace effectively. He has been working on the LSB since 2005.

Stew Benedict is a member of the technical staff at the Linux Foundation. He previously worked as a distribution developer for Mandrake/Mandriva, working on PPC/IA64 ports and various security initiatives. He has been involved with LSB, first as a distribution representative and later as a developer, for several years.

Vladimir Rubanov is Head of Department at the Institute for System Programming of the Russian Academy of Sciences (ISPRAS) and the Director of the LinuxTesting.org Verification Center. He and his team have been actively involved in the LSB workgroup since 2005. Vladimir holds MSc and PhD degrees in Computer Science from MIPT University ("Russian MIT").



For more information visit
www.siliconmechanics.com/R350,
www.siliconmechanics.com/A350,
or call us toll free at 866-352-1173.



There's a lot of heavy lifting going on these days in the world of data processing. Just ask Jason, a Silicon Mechanics Sales Expert who fields questions every day about the best way to address a vast range of computing workloads. One solution finding enthusiastic adoption is hybrid CPU / GPU computing, such as with the Silicon Mechanics Rackform iServ R350 and the Rackform nServ A350.

We start with your choice of two state-of-the-art processors, for fast, reliable, energy-efficient processing. Then we add two NVIDIA® Tesla GPUs, to dramatically accelerate parallel processing for applications like ray tracing and finite element analysis. Load it up with DDR3 memory and you have herculean capabilities and an 80 PLUS Gold Certified power supply, all in the space of a 1U server.

When you partner with Silicon Mechanics, you get more than breakout technologies that will carry the weight of your workload—you get an expert like Jason.

Expert included.



▲ **In 2005, after just two weeks, Linus Torvalds completed the first version of Git, an open-source version control system.**

Unlike typical centralized systems, Git is based on a distributed model. It is extremely flexible and guarantees data integrity while being powerful, fast and efficient. With widespread and growing rates of adoption, and the increasing popularity of services like GitHub, many consider Git to be the best version control tool ever created.

Surprisingly, Linus had little interest in writing a version control tool before this endeavor. He created Git out of necessity and frustration. The Linux Kernel Project needed an open-source tool to manage its massively distributed development effectively, and no existing tools were up to the task.

Many aspects of Git's design are radical departures from the approach of tools like CVS and Subversion, and they even

differ significantly from more modern tools like Mercurial. This is one of the reasons Git is intimidating to many prospective users. But, if you throw away your assumptions of how version control should work, you'll find that Git is actually simpler than most systems, but capable of more.

In this article, I cover some of the fundamentals of how Git works and stores data before moving on to discuss basic usage and work flow. I found that knowing what is going on behind the scenes makes it much easier to understand Git's many features and capabilities. Certain parts of Git that I previously had found complicated suddenly were easy and straightforward after spending a little time learning how it worked.

I find Git's design to be fascinating in and of itself. I peered behind the curtain, expecting to find a massively complex machine, and instead saw only a little hamster running in a wheel. Then I realized a complicated design not only wasn't needed, but also wouldn't add any value.

Git Object Repository

Git, at its core, is a simple indexed name/value database. It stores pieces of data (values) in “objects” with unique names. But, it does this somewhat differently from most systems. Git operates on the principle of “content-addressed storage”, which means the names are derived from the values. An object’s name is chosen automatically by its content’s SHA1 checksum—a 40-character string like this:

```
1da177e4c3f41524e886b7f1b8a0c1fc7321cac2
```

SHA1 is cryptographically strong, which guarantees a different checksum for different data (the actual risk of two different pieces of data sharing the same SHA1 checksum is infinitesimally small). The same chunk of data always will have the same SHA1 checksum, which always will identify only that chunk of data. Because object names are SHA1 checksums, they identify the object’s content while being truly globally unique—not just to one repository, but to all repositories everywhere, forever.

To put this into perspective, the example SHA1 listed above happens to be the ID of the first commit of the Linux kernel into a Git repository by Linus Torvalds in 2005 (2.6.12-rc2). This is a lot more useful than some arbitrary revision number with no real meaning. Nothing except that commit ever will have the same ID, and you can use those 40 characters to verify the data in every file throughout that version of Linux. Pretty cool, huh?

Git stores all the data for a repository in four types of objects: blobs, trees, commits and tags. They are all just objects with an SHA1 name and some content. The only difference between them is the type of information they contain.

Blobs and Trees

A blob stores the raw data content of a file. This is the simplest of the four object types.

A tree stores the contents of a directory. This is a flat list of file/directory names, each with a corresponding SHA1 representing its content. These SHA1s are the names of other objects in the repository. This referencing technique is used throughout Git to link all kinds of information together. For file entries, the referenced object is a blob. For directory entries, the referenced object is a tree that can contain more directory entries, in turn referencing more trees to define a complete and potentially unlimited hierarchy.

It’s important to recognize that blobs and trees are not themselves files and directories; they are just the contents of files and directories. They don’t know about anything outside their own content, including the existence of any references in other objects that point to them. References are one-way only.

In the example shown in Figure 1, I’m assuming that the files MyApp.pm and MyApp1.pm have the same contents, and so by definition, they must reference the same blob object. This behavior is implicit in Git because of its content-addressable design and works equally well for directories with the same content.

As you can see, directory structures are defined by chains of references stored in trees. A tree is able to represent all of the data in the files and directories under it even though it contains

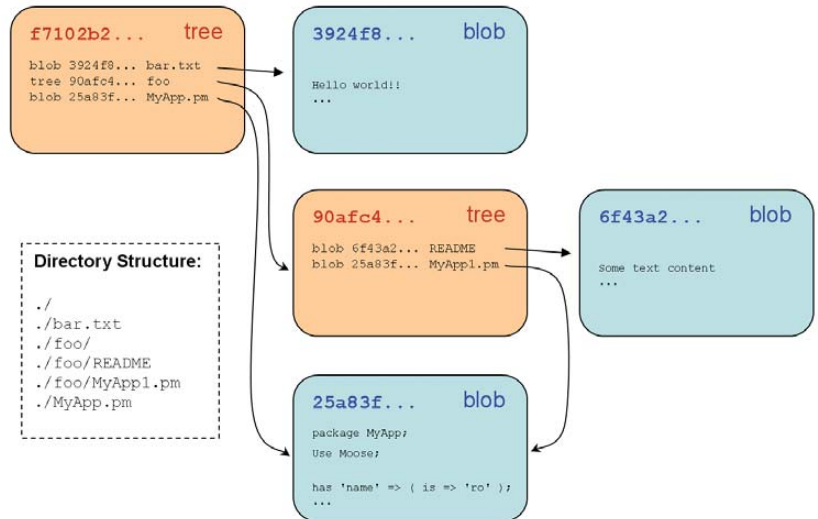


Figure 1. An example directory structure and how it might be stored in Git as tree and blob objects (I truncated the SHA1 names to six characters for readability).

only one level of names and references. Because SHA1s of the referenced objects are within its content, a tree’s SHA1 exactly identifies and verifies the data throughout the structure; a checksum resulting from a series of checksums verifies all the underlying data regardless of the number of levels.

Consider storing a change to the file README illustrated in Figure 1. When committed, this would create a new blob (with a new SHA1), which would require a new tree to represent “foo” (with a new SHA1), which would require a new tree for the top directory (with a new SHA1).

While creating three new objects to store one change might seem inefficient, keep in mind that aside from the critical path of tree objects from changed file to root, every other object in the hierarchy remains identical. If you have a gigantic hierarchy of 10,000 files and you change the text of one file ten directories deep, 11 new objects allow you to describe both the old and the new state of the tree.

Note:

One potential problem of the content-addressed design is that two large files with minor differences must be stored as different objects. However, Git optimizes these cases by using deltas to eliminate duplicate data between objects wherever possible. The size-reduced data is stored in a highly efficient manner in “pack files”, which also are further compressed. This operates transparently underneath the object repository layer.

Commits

A commit is meant to record a set of changes introduced to a project. What it really does is associate a tree object—representing a complete snapshot of a directory structure at a moment in time—with contextual information about it, such as who made the change and when, a description, and its parent commit(s).

A commit doesn’t actually store a list of changes (a “diff”) directly, but it doesn’t need to. What changed can be calculated on-demand by comparing the current commit’s tree to that of its parent. Comparing two trees is a lightweight operation, so there is no need to store this information. Because there actually is

FEATURE Git

nothing special about the parent commit other than chronology, one commit can be compared to any other just as easily regardless of how many commits are in between.

All commits should have a parent except the first one. Commits usually have a single parent, but they will have more if they are the result of a merge (I explain branching and merging later in this article). A commit from a merge still is just a snapshot in time like any other, but its history has more than one lineage.

By following the chain of parent references backward from the current commit, the entire history of a project can be reconstructed and browsed all the way back to the first commit.

A commit is expanded recursively into a project history in exactly the same manner as a tree is expanded into a directory structure. More important, just as the SHA1 of a tree is a fingerprint of all the data in all the trees and blobs below it, the SHA1 of a commit is a fingerprint of all the data in its tree, as well as all of the data in all the commits that preceded it.

This happens automatically because references are part of an object's overall content. The SHA1 of each object is computed, in part, from the SHA1s of any objects it references, which in turn were computed from the SHA1s they referenced and so on.

Tags

A tag is just a named reference to an object—usually a commit. Tags typically are used to associate a particular version number with a commit. The 40-character SHA1 names are many things, but human-friendly isn't one of them. Tags solve this problem by letting you give an object an additional name.

There are two types of tags: object tags and lightweight tags. Lightweight tags are not objects in the repository, but instead are simple refs like branches, except that they don't change. (I explain branches in more detail in the Branching and Merging section below.)

Setting Up Git

If you don't already have Git on your system, install it with your package manager. Because Git is primarily a simple command-line tool, installing it is quick and easy under any modern distro.

You'll want to set the name and e-mail address that will be recorded in new commits:

```
git config --global user.name "John Doe"
git config --global user.email john@example.com
```

This just sets these parameters in the config file `~/.gitconfig`. The config has a simple syntax and could be edited by hand just as easily.

User Interface

Git's interface consists of the "working copy" (the files you directly interact with when working on the project), a local repository stored in a hidden `.git` subdirectory at the root of the working copy, and commands to move data back and forth between them, or between remote repositories.

The advantages of this design are many, but right away you'll notice that there aren't pesky version control files scattered throughout the working copy, and that you can work off-line without any loss of features. In fact, Git doesn't have any concept of a central authority, so you always are "working off-line" unless you specifically ask Git to exchange commits with your peers.

The repository is made up of files that are manipulated by

invoking the `git` command from within the working copy. There is no special server process or extra overhead, and you can have as many repositories on your system as you like.

You can turn any directory into a working copy/repository just by running this command from within it:

```
git init
```

Next, add all the files within the working copy to be tracked and commit them:

```
git add .
git commit -m "My first commit"
```

You can commit additional changes as frequently or infrequently as you like by calling `git add` followed by `git commit` after each modification you want to record.

If you're new to Git, you may be wondering why you need to call `git add` each time. It has to do with the process of "staging" a set of changes before committing them, and it's one of the most common sources of confusion. When you call `git add` on one or more files, they are added to the Index. The files in the Index—not the working copy—are what get committed when you call `git commit`.

Think of the Index as what will become the next commit. It simply provides an extra layer of granularity and control in the commit process. It allows you to commit some of the differences in your working copy, but not others, which is useful in many situations.

You don't have to take advantage of the Index if you don't want to, and you're not doing anything "wrong" if you don't. If you want to pretend it doesn't exist, just remember to call `git add .` from the root of the working copy (which will update the Index to match) each time and immediately before `git commit`. You also can use the `-a` option with `git commit` to add changes automatically; however, it will not add new files, only changes to existing files. Running `git add .` always will add everything.

The exact work flow and specific style of commands largely are left up to you as long as you follow the basic rules.

The `git status` command shows you all the differences between your working copy and the Index, and the Index and the most recent commit (the current HEAD):

```
git status
```

This lets you see pending changes easily at any given time, and it even reminds you of relevant commands like `git add` to stage pending changes into the Index, or `git reset HEAD <file>` to remove (unstage) changes that were added previously.

Branching and Merging

The work you do in Git is specific to the current branch. A branch is simply a moving reference to a commit (SHA1 object name). Every time you create a new commit, the reference is updated to point to it—this is how Git knows where to find the most recent commit, which is also known as the tip, or head, of the branch.

By default, there is only one branch ("master"), but you can have as many as you want. You create branches with `git branch` and switch between them with `git checkout`. This may seem odd at first, but the reason it's called "checkout" is that you are "checking out" the head of that branch into your working copy. This alters the files in your working copy to match the commit at

the head of the branch.

Branches are super-fast and easy, and they're a great way to try out new ideas, even for trivial things. If you are used to other systems like CVS/SVN, you might have negative thoughts associated with branches—forget all that. Branching and merging are free in Git and can be used without a second thought.

Run the following commands to create and switch to a new local branch named "myidea":

```
git branch myidea
git checkout myidea
```

All commits now will be tracked in the new branch until you switch to another. You can work on more than one branch at a time by switching back and forth between them with `git checkout`.

Branches are really useful only because they can be merged back together later. If you decide that you like the changes in myidea, you can merge them back into master:

```
git checkout master
git merge myidea
```

Unless there are conflicts, this operation will merge all the changes from myidea into your working copy and automatically commit the result to master in one fell swoop. The new commit will have the previous commits from both myidea and master listed as parents.

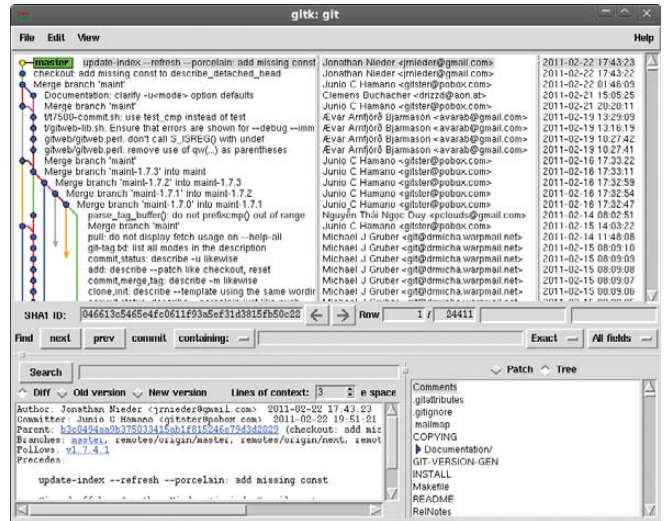


Figure 2. gitk

However, if there are conflicts—places where the same part of a file was changed differently in each branch—Git will warn you and update the affected files with "conflict markers" and not commit the merge automatically. When this happens, it's up to you to edit the files by hand, make decisions between the versions from each branch, and then remove the conflict markers. To

Small Form Factor PCs

Cost Effective Embedded PC For Appliances

650.871.3930
info@OEMproduction.com
800-664-8917



ITX-10A Fanless 1.6GHz \$129



ITX-10B ION2 DualCore \$195
ITX-100A2 Fanless DualCore \$168



ITX-500 Series



ITX-30A Atom with PCI Riser \$135



ITX-30G with NVIDIA® ION™ Graphics
Barebone system \$199



ITX-7025A-E32 \$155
1.8GHz Athlon™ II + Nvidia® Graphics



ITX-H6700 1155pin Core i5 \$199



Full Height Riser or Low-Profile
Add-on Slot
up to 8 x 2.5" or 4 x 3.5" HD



ITX-1000C with 4LAN
and WiFi Option



VESA / Wallmount option

Over 250 Mini-PC Models Available:

- Intel® 1155pin Core™ i5 Systems
- AMD® AM3 Athlon™/Phenom™ Platform
- PCI, PCIe, MiniPCIe Slot for TV Tuner or Industrial Add-on
- Custom Design Chassis for Small to Mid Size OEM Project

www.OEMproduction.com

- Small White Box Solution
- Custom Design, Flexible Configuration
- Excellent Service with Over 23 Year Experiences

OEM Production

1461-2 San Mateo Ave. South San Francisco, CA 94080

Tel: 650.871.3930 Fax: 650.523.8636

NVIDIA, ION are trademarks of NVIDIA Corporation. Other names are for informational purposes only and may be trademarks of their respective owners.

FEATURE Git

complete the merge, use `git add` on each formerly conflicted file, and then `git commit`.

After you merge from a branch, you don't need it anymore and can delete it:

```
git branch -d myidea
```

If you decide you want to throw `myidea` away without merging it, use an uppercase `-D` instead of a lowercase `-d` as listed above. As a safety feature, the lowercase switch won't let you delete a branch that hasn't been merged.

To list all local branches, simply run:

```
git branch
```

Viewing Changes

Git provides a number of tools to examine the history and differences between commits and branches. Use `git log` to view commit histories and `git diff` to view the differences between specific commits.

These are text-based tools, but graphical tools also are available, such as the `gitk` repository browser, which essentially is a GUI version of `git log --graph` to visualize branch history. See Figure 2 for a screenshot.

Remote Repositories

Git can merge from a branch in a remote repository simply by transferring needed objects and then running a local merge. Thanks to the content-addressed storage design, Git knows which objects to transfer based on which object names in the new commit are missing from the local repository.

The `git pull` command performs both the transfer step (the "fetch") and the merge step together. It accepts the URL of the remote repository (the "Git URL") and a branch name (or a full "refspec") as arguments. The Git URL can be a local filesystem path, or an SSH, HTTP, rsync or Git-specific URL. For instance, this would perform a pull using SSH:

```
git pull user@host:/some/repo/path master
```

Git provides some useful mechanisms for setting up relationships with remote repositories and their branches so you don't have to type them out each time. A saved URL of a remote repository is called a "remote", which can be configured along with "tracking branches" to map the remote branches into the local repository.

A remote named "origin" is configured automatically when a repository is created using `git clone`. Consider a clone of Linus Torvald's Kernel Tree mirrored on GitHub:

```
git clone https://github.com/mirrors/linux-2.6.git
```

If you look inside the new repository's config file (`.git/config`), you'll see these lines set up:

```
[remote "origin"]
  fetch = +refs/heads/*:refs/remotes/origin/*
  url = https://github.com/mirrors/linux-2.6.git
[branch "master"]
  remote = origin
  merge = refs/heads/master
```

The fetch line above defines the remote tracking branches. This "refspec" specifies that all branches in the remote repository under "refs/heads" (the default path for branches) should be transferred to the local repository under "refs/remotes/origin". For example, the remote branch named "master" will become a tracking branch named "origin/master" in the local repository.

The lines under the branch section provide defaults—specific to the master branch in this example—so that `git pull` can be called with no arguments to fetch and merge from the remote master branch into the local master branch.

The `git pull` command is actually a combination of the `git fetch` and `git merge` commands. If you do a `git fetch` instead, the tracking branches will be updated and you can compare them to see what changed. Then you can merge as a separate step:

```
git merge origin/master
```

Git also provides the `git push` command for uploading to a remote repository. The push operation is essentially the inverse of the pull operation, but since it won't do a remote "checkout" operation, it is usually used with "bare" repositories. A bare repository is just the git database without a working copy. It is most useful for servers where there is no reason to have editable files checked out.

For safety, `git push` will allow only a "fast-forward" merge where the local commits derive from the remote head. If the local head and remote head have both changed, you must perform a full merge (which will create a new commit deriving from both heads). Full merges must be done locally, so all this really means is you must call `git pull` before `git push` if someone else committed something first.

Conclusion

This article is meant only to provide an introduction to some of Git's most basic features and usage. Git is incredibly powerful and has a lot more capabilities beyond what I had space to cover here. But, once you realize all the features are based on the same core concepts, it becomes straightforward to learn the rest.

Check out the Resources section for some sites where you can learn more. Also, don't forget to read the git man page. ■

Henry Van Styn is the founder of IntelliTree Solutions, an IT consulting and software development firm located in Cincinnati, Ohio. Henry has been developing software and solutions for more than ten years, ranging from sophisticated Web applications to low-level network and system utilities. He is the author of Strong Branch Linux, an in-house server distribution based on Gentoo. Henry can be contacted at www.intellitree.com.

Resources

Git Home Page: git-scm.com

Git Community Book: book.git-scm.com

Why Git Is Better Than X: whygitisbetterthanx.com

Google Tech Talk: Linus Torvalds on Git:
www.youtube.com/watch?v=4XpnKHJAok8

OHIO LINUX FEST

September 9-11, 2011

The Greater Columbus Convention Center



Register Now!

OHIO.LINUX.ORG

Podcasting

If you want to contribute to the Linux community but would rather talk than develop software or write documentation, podcasting may be the ideal approach for you.

CHARLES OLSEN

If you like to talk and have something interesting to say, you may enjoy podcasting. A podcast is simply an audio file in MP3 or Ogg format that people can download and listen to on their own schedule.

A few years ago, I created a podcast called *mintCast*, which was designed to help new users get started with Linux. I had been listening to several podcasts, but I knew nothing about producing one myself. This article summarizes what I learned during the following two years. It should save you some time and pain, if you decide to create a podcast.

Entire books have been written about podcasting, and you should read some of them if you want to study the subject in depth. To keep this article to a manageable length, I focus on what I learned while producing *mintCast*.

Plan

The first step is to choose your topic and target audience. If your topic is too broad or too narrow, it will be hard to build an audience.

Going Linux has a great example. The home page describes the target audience (“New to Linux, upgrading from Windows to Linux, or just thinking about moving to Linux?”) and what it provides (“practical, day-to-day advice on how to use Linux and its applications”). Potential listeners can see at a glance if this is a podcast they want to listen to.

A podcast episode usually has multiple segments. Typical segments might include an introduction, personal news, industry news, opinions and commentary, interviews, tutorials, tips, recommendations of software or Web sites, and listener feedback. Whichever segments you decide to use, it’s a good idea to stay consistent.

That doesn’t mean you need to be absolutely rigid. A typical episode of *mintCast* would have these segments: introduction, personal news, Linux news, main topic, Web site recommendation and listener feedback. The main topic would be a tutorial, interview or discussion about a new distro release. This approach gave us a consistent format, with enough variety to keep it fresh and interesting.

You should write an outline before you start recording. Include the topics you want to talk about and the points you want to make. If your show has multiple hosts, Google Documents is an effective way to share your notes, because all the hosts can read and even edit the document.

Do not write a script of exactly what you want to say, and do not read directly from your notes. It’s very obvious to listeners when you do this, and it sounds bad. It’s acceptable to read a *brief* quotation, for example, when you’re reading from a news item. Otherwise, just write out the points you want to make, and use the list only as a reminder as you’re speaking.

Will you work alone, or with other hosts? I always found it easier to record a show with one or two other hosts. Having

additional hosts means that you don’t have to know everything, and you don’t have to do all the talking.

In the first few episodes of *mintCast*, each host recorded separately, then we edited the recordings together. In later episodes, we recorded at the same time, allowing us to converse and play off each other. Doing it that way made the shows more interesting, and it was easier for the hosts to keep going. It did, however, add complexity to the process of recording the show.

Recording

There are many ways to record a podcast. Sometimes I used a small handheld recorder that saved the recordings in MP3 format and had a USB port that allowed me to copy the files over to my PC.

Most of the time, I used Audacity to record the show. Audacity is an open-source program that can record, edit and play audio files. You can install it from your distro’s repositories.

A good quality microphone makes a big difference when you’re recording a podcast. It doesn’t need to be expensive, but do some research and read reviews before choosing a microphone.

A headset may seem to be a good choice, and that’s what I worked with. It causes problems, though—unless you’re very disciplined about always keeping your hands away from your head, it’s very easy to bump the headset, making an ugly noise on the recording. A microphone boom can help avoid this.

Although Audacity has a lot of power and features, it’s very easy to use. The default settings are fine for recording spoken voice. You can tweak the input settings by watching the meter while speaking a few words, then click the big red Record button when you’re ready to start.

It becomes more complicated if you have two hosts and you’re not sitting in the same room. On *mintCast*, we solved this problem by talking on the telephone, and each of us recorded our own audio in Audacity. We could then assemble the recordings as side-by-side tracks in Audacity. This gave us a very high recording quality.

When more hosts joined the team, or when we did user interviews, we needed to get three or four people on the call at the same time. We did not have the ability to do conference calls from our personal phones, and we didn’t want to ask our interview subjects to try to use Audacity.

We tried Skype and Skype Call Recorder, but never managed to get that working. I know other podcasts use this software, and I certainly would try it again if I were doing a podcast now.

Some podcasts use the Talkshoe Web site. Once you’ve created a free Talkshoe account, you can create a call series and invite others to the calls. You can allow them to participate fully or just listen. Calls can be recorded and then are available to listen to or download at any time.

You can listen to Talkshoe calls on your computer or telephone. You can participate by calling in from a telephone or by using Skype or Ekiga. Talkshoe also sets up a chat room, allowing listeners on a live show to interact with the hosts.

Talkshoe can be your start-to-finish solution to podcasting. It allows you to promote, broadcast, record and publish your recordings. Why, then, should you bother with any of the other tools and processes in this article?

For mintCast, I could not bring myself to publish our shows without a considerable amount of editing. Each of the hosts had a verbal tic—for example, frequently saying “uh”, “you know” or “basically”—that was too embarrassing to leave in the recording. In Talkshoe, the audio is published exactly as it is recorded.

We also wanted to edit in theme music at the beginning and end of the show. We couldn't do that with Talkshoe.

Still, we often used Talkshoe to record the show. I then would download the recording and do the necessary editing before publishing it to our own Web site.

The mintCast team now is using Mumble to record shows. Mumble is a free and open-source voice chat program. It uses a nonstandard protocol, so you need to set up a Mumble server to which the Mumble clients can connect. With Mumble server and client version 1.2.3, you have the ability to record the Mumble call. You then can load the recording into Audacity for editing.

Recording Tips

Before making your first recording, it's a good idea to do a test recording for at least five to ten minutes. Make sure your recording levels are good, and that it sounds like you expect. Listen for background noise—things that you normally don't even notice, like ceiling fans and air conditioners, might interfere.

Even low-quality microphones seem to pick up everything you don't want on the recording. If you're eating, drinking or typing, listeners will know it.

Make sure you know where your mute button is. If your microphone doesn't have a mute button, leave the sound control panel open on your PC and find the mute control. A lot of things can happen that you don't want on the recording, and the mute button can save you a lot of tedious editing.

Don't worry about making a perfect recording. Unless you're using Talkshoe to publish the podcast exactly as recorded, you can edit out most mistakes. We often found ourselves rambling or stumbling over our words, and then we would take a moment to collect our thoughts and then go back and start over at the beginning of the sentence or paragraph. These “do-overs” helped us produce a better show.

Editing

Once you have the show recorded, the best tool for editing is Audacity. It's in the repo of every major distro, and it's very easy to use, even if you have no previous experience editing audio.

If you recorded in Audacity, you're ready to start editing. If you recorded with some other tool, Audacity can open files in any audio format. If you have each host's audio in a separate recording, you can add new audio tracks and paste each recording into a separate track.

I also like to add a separate track for music. We had a theme

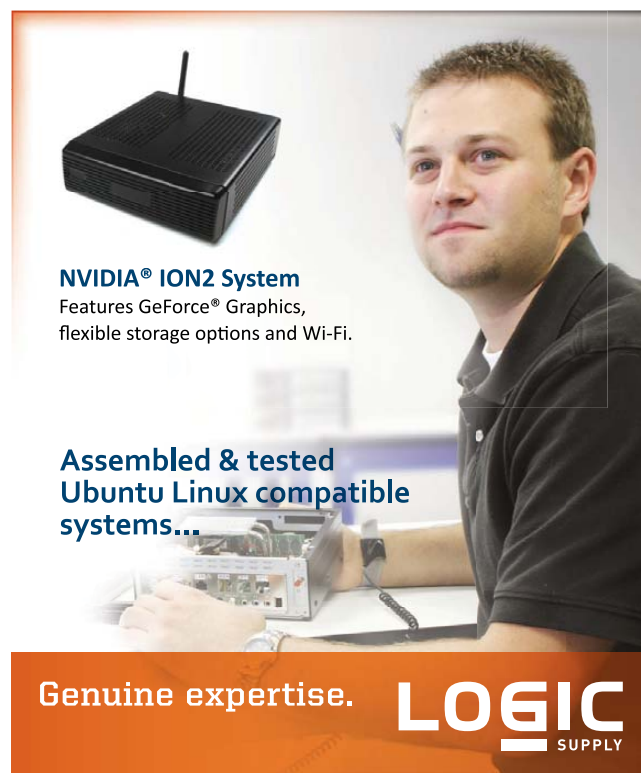
song at the beginning of the show, a longer version of the theme for the end of the show, and a five-second clip from the theme that we used as a bumper between segments. I put all of those clips into the audio track.

Audacity allows you to select part of the audio, then cut or copy and paste it. A scrolling display shows a waveform of your audio for each track, making it easy to select exactly the part of the clip you want to work with. You can zoom in or out on the waveform display. After doing a few shows, I could recognize my “uhs” from the waveform before I even heard them.

If your audio is in multiple tracks, be careful about deleting material. If you delete a clip from one track, you've now changed the timing for the rest of that track, and it will be out of sync with the other tracks. If you delete a clip, be sure to select and delete that time frame from all tracks so that they will stay in sync. Audacity allows you to click and drag across all of the tracks if you need to.

But, if you need to remove unneeded noise from one track while a host is talking on another track, you don't want to delete it. Just select the noise in the waveform, then click the Generate menu and choose Silence. This will silence that part of that track, without affecting that track's timing compared to the other tracks.

Audacity has several useful tools on the Generate and Effect menus. If you want to remove a sound without silencing it, you



NVIDIA® ION2 System
Features GeForce® Graphics,
flexible storage options and Wi-Fi.

**Assembled & tested
Ubuntu Linux compatible
systems...**

Genuine expertise. LOGIC SUPPLY

www.logicsupply.com/linux

© 2011 Logic Supply, Inc. All products and company names listed are trademarks or trade names of their respective companies.

also can select the clip and Generate a tone, a chirp or a noise that sounds like static.

It was not unusual for our initial recording levels to come out way too low. The Effect menu includes an Amplify command, allowing us to select an entire track (or any portion of it) and amplify the volume.

If the volume varies too much in different parts of the recording, the Compressor command on the Effect menu can bring the levels closer together. The volume will be increased for the quiet parts and reduced for the loud parts.

If you do have a consistent noise in the background, such as an air conditioner, the Noise Removal command on the Effect menu often can remove it. You need a few seconds of audio where you're not talking, where the only sound is the background noise you want to remove. When you click the Noise Removal command, a dialog box pops up explaining how to use it.

Save often as you work. Although crashes were not frequent, Audacity did crash more than any other program I've used in Linux.

Finally, Audacity can export the finished file to the appropriate audio format. Podcasts are typically MP3, though many also are offered in Ogg format.

Music

Remember that all those CDs and MP3s you've purchased over the years are copyrighted—you can't use those songs in your show. However, there are sources where you can get music without spending thousands of dollars.

The Podcast Themes Web site has music specifically created for podcasts. A few songs are available for free, and many others are available at a low cost. Various licenses allow you to buy exclusive or non-exclusive rights to the music.

Unique Tracks is another Web site that offers royalty-free music. They offer individual songs and albums in many different styles: classical, uplifting, rock, heavy metal, hip-hop, New Age and many others. Many of the songs include the full-length song as well as shorter versions, and some even include a loop of five to ten seconds, which is ideal for a bumper.

Publishing

Once you have a completed recording, you need to publish it so that others can download it. If you recorded in Talkshoe, it's already published. Even if you recorded in Talkshoe, you still can publish it to your own site.

Any Web-hosting company will offer a template for publishing a podcast. If your site uses WordPress, you'll need the PodPress plugin. In any case, you'll have a Web location to upload the finished files. Then, create a blog entry on your site and link the audio files.

Although your site probably already offers a feed, you can use Google FeedBurner to get information allowing you to analyze your traffic. Once you're set up in FeedBurner, it's a good idea to submit your podcast to the iTunes store, the Zune Marketplace and Podcast Alley. All of those will make it easier for people to find your podcast.

Feedback

Once you have listeners, some will give you feedback about the show. It's best if your Web site is set up with a forum where

Linux Podcasts

Going Linux (goinglinux.com): for computer users who just want to use Linux to get things done.

Linux Action Show (www.jupiterbroadcasting.com/show/linuxactionshow): video broadcast with Linux news and reviews.

Linux Outlaws (www.linuxoutlaws.com): two pragmatic Linux users talk about the latest developments in free and open software and culture (*warning*: explicit language).

Linux Reality (linuxreality.com): an excellent podcast, no longer actively produced but still available for download.

mintCast (www.mintcast.org): generally focuses on helping users get started with Linux.

The Linux Link Tech Show (www.tllts.org): a live weekly Webcasted audio show dealing with the Linux computer operating system and any other geeky subjects that the hosts find interesting at the time.

TuxRadar (tuxradar.com/podcast): news, reviews, rants, raves, chit-chat, geekspeak and more.

Ubuntu UK (podcast.ubuntu-uk.org): all the latest news and issues facing Ubuntu Linux users and free software fans in general.

Many more podcasts about Linux, or that frequently cover Linux topics, exist. You can find dozens more podcasts at the Linux Link: www.thelinuxlink.net.

listeners can discuss the show. On simpler sites, they may be able to leave comments only on the blog post. If nothing else, at least get a Gmail account and publish the address.

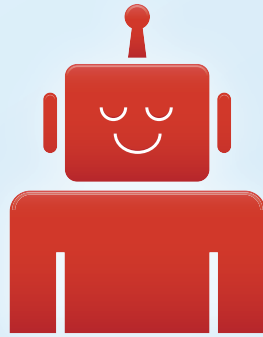
Read the comments and consider them seriously, especially the negative comments. I know it's painful to read criticism after you've put so much work into the podcast, but those criticisms often can help you make a better podcast. Encourage listeners to write in with comments and suggestions, and thank them for their feedback.

Conclusion

When I first started using Linux, the podcasts I listened to really helped me a lot. I was reading books and magazines, of course, but listening to a friendly voice really helps make a new topic accessible. It's also nice to be able to learn new things while driving to and from work.

If you like to talk and know a thing or two about Linux, why not help others increase their mastery of Linux? ■

Charles Olsen has been working in IT help desk and technical training for more years than he will admit. For two years, he was lead host of mintCast, a podcast by the Linux Mint community for all users of Linux. You can find mintCast at www.mintcast.org or, if you must, in iTunes.



Lullabot™

Learn Drupal & jQuery

FROM THE COMFORT OF
YOUR LIVING ROOM



The Lullabot Learning Series includes everything you need to become a Drupal & jQuery expert from the comfort of your living room! The videos are available in both DVD format and high-definition video download.

Purchase the videos at <http://store.lullabot.com>

How to Build a Beowulf HPC System Using the FedoraLiveCD Project

Build a Beowulf cluster without disks to optimize cost and reliability, and simplify software maintenance. HOWARD POWELL

The **FedoraLiveCD Project** allows anyone to create a custom bootable CD or PXE boot image with little effort. For large HPC systems, this greatly simplifies the creation of diskless compute nodes, leading to higher reliability and lower costs when designing your cluster environment. The network and CPU overhead for a diskless setup are minimal, and the compute

nodes will run entirely from an initial ramdisk, so they will exhibit very good I/O for normal OS disk operations.

The cluster I've designed is set up for MPI-based computation. The master node runs a queue system where jobs can be submitted and farmed out to the compute nodes to run within the allotted resources. Because my compute nodes are diskless, the goal is to

Listing 1. Example dhcpd.conf File

```
#
# DHCP Server Configuration file.
# see /usr/share/doc/dhcp*/dhcpd.conf.sample
#
ddns-update-style interim;
allow booting;
allow bootp;
option dns-domain-search-list code 119 = string;

subnet 10.0.0.0 netmask 255.255.0.0 {
    default-lease-time 604800;
    max-lease-time 1209600;
    option routers 10.0.0.1;
    option ip-forwarding off;
    option subnet-mask 255.255.0.0;
    range dynamic-bootp 10.0.0.100 10.0.0.254;
}

subnet 10.1.0.0 netmask 255.255.0.0 {
    default-lease-time 604800;
    max-lease-time 1209600;
    option routers 10.1.0.1;
    option ip-forwarding off;
    option ntp-servers 10.1.0.1;
    option subnet-mask 255.255.0.0;
    option domain-name-servers 10.1.0.1;
    option time-offset -5;
    option domain-name "cluster";
    option interface-mtu 9000;
}

class "pxeclients" {
    match if substring(option vendor-class-identifier, 0, 9) =
        "PXEClient";
    next-server 10.1.0.1;
    filename "pxelinux.0";
}

host c0 {
    hardware ethernet A4:BA:DB:1E:71:2D;
    fixed-address 10.1.0.254;
    option host-name "c0";
}

host c1 {
    hardware ethernet A4:BA:DB:1E:71:3A;
    fixed-address 10.1.0.253;
    option host-name "c1";
}

host c2 {
    hardware ethernet A4:BA:DB:1E:71:47;
    fixed-address 10.1.0.252;
    option host-name "c2";
}

host c3 {
    hardware ethernet A4:BA:DB:1E:71:54;
    fixed-address 10.1.0.251;
    option host-name "c3";
}
```

produce a simple and streamlined operating system with as few libraries and utilities as necessary to get the nodes to interact with the master job scheduler. Software that is needed by jobs (such as the MPI libraries) can be shared via NFS from the master node. The compute nodes simply have a kernel and the basic libraries needed to start a job. User account information can be shared via a local LDAP service running on the master node or by any method you already may have available in your environment.

To prepare a diskless cluster, your master node will need some amount of reasonably fast local disk storage and at least 10/100 Ethernet, preferably gigabit Ethernet. Your diskless nodes will need Ethernet hardware that can PXE boot from a network interface; most modern hardware supports this. These nodes will need to be on the same physical subnet, or you will have to configure your dhcpd service to respond or relay between subnets. Your diskless nodes also should have sufficient physical memory (RAM) to hold the OS image plus have enough room to run your programs—a few gigabytes of RAM should be sufficient if you keep your OS image simple.

For the rest of this article, I assume your cluster is based on a Red Hat-derived distribution, as this is based on a Fedora-specific tool. I'm going to demonstrate an environment where all of the cluster nodes can communicate with the master on a private Ethernet subnet.

Your boot server needs to run just two services for diskless booting: DHCP and TFTP. DNSMasq can be substituted for DHCP and TFTP, but I demonstrate using separate DHCP and TFTP services because that's how I set up my own cluster. For convenience, you may choose to install bind or some other DNS to make communication between nodes more friendly. To deploy custom rpm files quickly, you may want to have access to a local repository shared via Apache or another Web service. Local rpm repositories also are a viable method to deploy custom rpm files.

First, install DHCP via yum:

```
yum -y install dhcp tftp-server syslinux
```

The file /etc/dhcpd.conf should be created, and in this config file, you need to define your subnet and a pxelclients class that simply locates the bootable pxelinux image on disk. You also need

Listing 2. Example tftp File

```
service tftp
{
    socket_type    = dgram
    protocol      = udp
    wait          = yes
    user          = root
    server        = /usr/sbin/in.tftpd
    server_args   = -s /tftpboot
    disable       = no
    bind          = 10.1.0.1
    per_source    = 11
    cps           = 100 2
    flags        = IPv4
}
```

Advertiser Index

CHECK OUT OUR BUYER'S GUIDE ON-LINE.

Go to www.linuxjournal.com/buyersguide where you can learn more about our advertisers or link directly to their Web sites.

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

Advertiser	URL	Page #
1&1 INTERNET INC.	www.oneandone.com	1
ABERDEEN, LLC	www.aberdeeninc.com	C3
ARCHIE MCPHEE	www.mcphee.com	79
DIGI-KEY CORPORATION	www.digi-key.com	79
DRUPALCON	london2011.drupal.org	27
EMAC, INC.	www.emacinc.com	23
GENSTOR SYSTEMS, INC.	www.genstor.com	21
HOSTINGCON/INET INTERACTIVE	www.hostingcon.com	9
IXSYSTEMS, INC.	www.ixsystems.com	C2, 3
LINODE, LLC	www.linode.com	45
LINUX JOURNAL STORE	www.linuxjournalstore.com	33
LOGIC SUPPLY, INC.	www.logicsupply.com	39, 61
LULLABOT	www.lullabot.com	7, 63
MICROWAY, INC.	www.microway.com	C4, 5
OEM PRODUCTION	www.polywell.com	57
OHIO LINUX FEST	www.ohiolinux.org	59
POLYWELL COMPUTERS, INC.	www.polywell.com	79
RACKMOUNTPRO	www.rackmountpro.com	25
SILICON MECHANICS	www.siliconmechanics.com	18, 19, 53
TECHNOLOGIC SYSTEMS	www.embeddedx86.com	13
USENIX SECURITY SYMPOSIUM	www.usenix.org/sec11/lj	47
UTILIKILTS	www.utilikilts.com	79

ATTENTION ADVERTISERS

November 2011 Issue #211 Deadlines

Space Close: August 22; Material Close: August 30

Theme: Hack This

BONUS DISTRIBUTIONS:

Utah Open Source, USENIX OSDI, SHAREPOINT

Contact **Joseph Krack**, +1-713-344-1956 ext. 118, joseph@linuxjournal.com

Listing 3. Example nodes-ks.cfg

```

### System language
lang en_US.UTF-8

### System keyboard
keyboard us

### System timezone
timezone America/New_York

### Root password
rootpw abcd1234

### System authorization information
auth --useshadow --enablecache

### Firewall configuration
# Firewalls are not necessary in a cluster, usually
firewall --disabled

### Disables Selinux
selinux --disable

### Repositories
repo --name=Your-Custom-Repo --baseurl=
    http://your.custom.repo/
repo --name=base --baseurl=
    http://mirror.centos.org/centos/5/os/$basearch/
repo --name=newrepo --baseurl=file:///tmp/localrepo

### Enable and disable some services
services --enabled=gpm,ipmi,ntpd --disabled=nfs

### Package install information
%packages
bash
kernel
syslinux
passwd
policycoreutils
chkconfig
authconfig
rootfiles
comps-extras

xkeyboard-config
nscd
nss_ldap
autofs
gpm
ntp
compat-gcc-34-g77
compat-libf2c-34
compat-libstdc++-296
compat-libstdc++-33
dapl
dapl-utils
dhcp
dmidecode
hwloc
iscsi-initiator-utils
libXinerama
libXmu
libXpm
libXp
libXt
man
mesa-libGL
nfs-utils
openssh
openssh-clients
openssh-server
pciutils
sysklogd
tvflash
vim-minimal
vim-enhanced

### Pre-install scripts

### Post-install scripts
%post

### Here you can run any shell commands you wish to
### further customize your nodes.

### Sets up DHCP networking on the compute nodes
cat << EOF > ifcfg-eth0
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
MTU=1500
EOF

mv ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-eth0

```

If you have multiple network interfaces on your master node, you can choose to bind TFTP to one interface by using the bind command.

to define the diskless hosts definition for each node by associating the bootable MAC address of each node with a static IP that you define for that node. I also chose to include the host-name option, so that my diskless hosts will know a name other than localhost.localdomain once they are booted.

Next, you need to enable the TFTP daemon. Red Hat systems launch TFTP via xinetd—I simply needed to enable the `/etc/xinetd.d/tftp` config file and start xinetd. If you have multiple network interfaces on your master node, you can choose to bind TFTP to one interface by using the bind command.

Once configured, both services should be added to the default runlevel and started:

```
chkconfig dhcpd on
chkconfig xinetd on
service dhcpd start
service xinetd start
```

Now for the fun part—creating the OS image. RPMForge hosts a version of the `livecd-tools` package, which can be installed via yum:

```
yum install livecd-tools
```

The live CD tools require a Red Hat kickstart file—templates can be found via Google and as part of the `livecd-tools` package. A template kickstart is generated by anaconda on any freshly installed system in the root home directory as `/root/anaconda-ks.cfg`.

Of particular interest here are the `%packages` and the `%post` sections. In `%packages`, you can choose exactly which programs you need or want installed on the initial ramdisk image and available to the OS at boot. I recommend choosing as little as you can in order to keep the `initrd` small and streamlined. In `%post`, you can add any shell commands you need in order to customize your compute nodes further—for example, by editing config files for needed services. The example kickstart provided here works with a RHEL- or CentOS 5.5-based distribution.

If you review my example kickstart file, you'll notice that I've specified DHCP as the boot protocol for the network on each of the compute nodes. Because the `dhcpd` service already knows about the Ethernet MAC address of my diskless compute nodes, the nodes simply will re-request an IP address during boot and be reassigned the same one. Remember that no unique information is stored on the node's OS image, so using DHCP is the easiest way to assign IPs to each diskless node.

One special situation to note: because the compute nodes are diskless, each time SSH starts on a node, it generates a new set of host keys. When the node reboots, it generates a new set of different keys, leading to an impossible-to-maintain situation for SSH users. To

Listing 4. Example `cluster-ssh-keys.spec`

```
%define name      cluster-ssh-keys
%define version   1.0
%define release   1

Summary: ssh keys for cluster compute nodes
Name: %{name}
Version: %{version}
Release: %{release}
Group: System Environment/Base
License: GPL
BuildArch: noarch
BuildRoot: %{_builddir}
URL: http://your.custom.url
Distribution: whatever
Vendor: You
Packager: your email

%description
This provides the ssh keys necessary for compute
nodes on a diskless cluster.

%prep
exit 0

%build
exit 0

%install
exit 0

%clean
exit 0

%files
%defattr(-,root,root)
/etc/ssh
```

solve this, I have generated a template host key that I then deploy copies of to each of my diskless compute nodes via an rpm file. To build your own version of this rpm, you need to create a spec file (see the example) and copy the host keys from `/etc/ssh` to the location specified by `BuildRoot` in the spec file. The `rpmbuild` command generates the rpm, and this rpm can be included in a local yum repository by specifying its name to the `%packages` section of your kickstart:

```
rpmbuild -bb sshkeys.spec
```

By setting up SSH with the same host key on each node, I've defeated some of the security of SSH by allowing the possibility of man-in-the-middle attacks between my master node and compute nodes. However, in my cluster environment where compute nodes communicate on a private and dedicated channel and do not have a direct connection to the outside

Listing 5. Example mknodes.sh

```
#!/bin/bash

/etc/init.d/nscd stop

cd /local-disk/nodes/

livecd-creator --config=/local/nodes/nodes-ks.cfg \
  --fslabel=cluster -t /local-disk/nodes/

livecd-iso-to-pxeboot /local-disk/nodes/cluster.iso

rsync -av /local-disk/nodes/tftpboot/ /tftpboot/

rm /local-disk/nodes/cluster.iso
rm -rf /local-disk/nodes/tftpboot

/etc/init.d/nscd start
```

world, this shouldn't be a problem.

Another idea that might simplify your SSH environment is to consider enabling host-based SSH authentication (so users don't have to generate private and public keys while on your

Listing 6. Example exports File

```
/local-disk 10.0.0.0/255.0.0.0(rw,async)
```

Listing 7. Example fstab File

```
master:/local-disk /local-disk nfs _netdev 0 0
```

creates the image and cleans up any temporary files for me.

Once the files have been copied to tftpboot, it's time to boot a compute node. If all goes well, the diskless client will request a DHCP address, and your DHCP server will respond with an IP and the location of the TFTP server and image to download. The client then should connect to the TFTP server, download the image and launch the OS you just created.

Problems with the PXE boot process can be diagnosed by using any network protocol analyzer, such as Wireshark. Once the image is loaded and the kernel is alive, you should see the normal boot process on the screen of the diskless compute node.

As noted before, specialized user-level software (such as the MPI libraries in my case) can be distributed to your nodes via standard NFS shares. On your NFS server (it can be the same as your master

User home directories can be shared via NFS or via a high-performance, cluster-based filesystem, such as PVFS2 or Lustre.

cluster). The root SSH environment is hardened against SSH host-based authentication, so you'll either have to work around this security measure or set up SSH public/private key-chains for the root account on your new cluster. Normal users should have no problems with host-based SSH authentication, so long as the UIDs are common among the entire cluster.

Once your kickstart has been customized to your liking, the rest of the setup is simple. Just run the livecd-creator script to generate an ISO image, then use the livecd-iso-to-pxe script to convert that into something TFTP can use.

When compiling the OS image, some active daemons may interfere with the build process. Of particular note, SELinux must be permissive or disabled, and if you use the nameserver cache daemon (nscd), you may need to disable it temporarily while the build process runs or else risk a corrupted image:

```
setenforce 0
service nscd stop
livecd-creator --config=nodes-ks.cfg --fslabel=Compute_nodes
livecd-iso-to-pxe Compute_nodes.iso
rsync -av tftpboot/ /tftpboot/
service nscd start
```

I've chosen to write all of this into a handy shell script that

node), simply define a new share in /etc/exports and enable NFS:

```
chkconfig nfs on
service nfs start
```

Your nodes need to add an entry for the NFS server either to their local fstab files or via some other method like autofs.

User home directories can be shared via NFS or via a high-performance, cluster-based filesystem, such as PVFS2 or Lustre. NFS is reliable when disk I/O is not very intensive or mostly read-only, but it breaks down if your code relies heavily on large numbers of files or heavy, simultaneous I/O operations to disk.

Please keep in mind that any customizations of the environment on the diskless nodes are not maintained between reboots. In fact, it's perfectly okay to cold-reset a diskless node; the OS image cannot be corrupted like it could be if it were on a local disk. This simplifies troubleshooting strange node problems. If a reboot doesn't clear the problem (and assuming no other diskless nodes show the same problem), it's almost certainly a hardware bug—this alone can save hours of time when working with a large cluster. ■

Howard Powell is the sole sysadmin at the University of Virginia Astronomy Department. He's built three generations of Linux-based high-performance computing clusters to support the Virginia Institute of Theoretical Astronomical, which are used to study cool things like what's happening around black holes in our universe. He lives near Charlottesville, Virginia.

Radio Dramas in Linux

A little care and some excellent FLOSS software can yield some ear-popping audio goodness. DAN SAWYER

Podcasting and satellite radio have brought back the age of audio fiction. The Golden Age of radio may have run through the 1930s and 1940s, but with the advent of Audible.com, Paudiobooks.com and the hundreds of streaming stations, the true Golden Age of audio fiction is now.

Everywhere you look, people are producing their audio fiction with...Garageband? If Macs are the price of entry, count me out. Excellent though Apple is on a number of points, I've never been able to reconcile myself to paying a premium for the privilege of being a member of a totalitarian-minded gadget-nut club.

But audio fiction has been a dear love of mine since childhood, and at the urging of Scott Sigler, I joined the rising wave four years ago. I've done fairly well for myself, producing my way, on Linux. What follows is a quick-and-dirty guide to making a radio drama that sounds amazing, rather than just adequate, using only FOSS tools.

Pre-Production and Production

I'm gonna rocket through these really quick, because they're not software-intensive, but they *are* important—desperately important. Without good quality audio to start with, you might as well go home and forget the whole thing.

The goal is to get good performances, with good technical quality, in a semi-dead room. Good performances come from a good director and good actors working together, either through a well-annotated script in a remote studio, or through recording with the director and actor in the same studio—work styles vary, as may your mileage. If you're starting this from scratch, experiment to find out what works best for you and your actors.

A semi-dead room is important as well. A number of the qualities of the human voice only manifest in space—a close mic in a completely dead room can rob some voices of their richness, while mic'ing a few inches away in a room with a few diffuse reflective surfaces will help bring out the texture and timbre in your actor's voice as the sound waves play phase games in the space. Put some time into tuning your room and doing experiments, and you'll be well rewarded. Also, a good pop screen (about \$30) and a copy of *The Yamaha Sound Reinforcement Handbook* will save you hours of grief on the far end.

Good technical quality comes through good equipment, clean power and quiet. Ultimately, you want a signal that's free of ground loop buzz with an ambient noise floor of below -60db. Don't try cheating by reducing your recording gain—this won't help you. When you amp it back up in post, you'll just hear all the noise again and some extra besides. Instead, aim for a mean signal ratio of more than -20db for a normal speaking voice, peaking up to 0 for shouting or loud

talking, and dipping down to -40db (at the lowest) for whispers or low voices. Check your work with the spectrum analyzer, and double-check it by listening with a good pair of headphones.

Getting your power cleaned might require a power conditioner (a \$60 Fuhrman will do fine) or a battery-operated recorder like the Zoom H4, and minimizing cable runs to reduce the chance of radio interference. Getting your room quiet enough may require installing foam pads in the windows and removing all computers from the room. A good handbook on setting up a home recording studio wouldn't go amiss.

And, of course, when recording voice, always record at as high a sample rate as possible. For best results, if you're recording on Linux, record in either Audacity or Ardour. (Watch those xruns; they will cause skips, so make sure you have JACK well set up for your equipment if you're using Ardour.) Also, make sure you're working with Planet CCRMA, 64Studio, UbuntuStudio or another "Studio" distribution—you'll need that real-time kernel with all its special patches if you want this to work like a dream, rather than a nightmare.

Cleanup and Editing

Most of the cleanup you'll need to do actually will come in the next step, and you can do that easily with in-line effects in Ardour—but I'll get to that in a moment. First, let's assume that, despite your best engineering-fu, you haven't quite been able to get a clean enough signal. You've got white noise at -50db—no buzz, just some annoying spread-spectrum hiss—so how do you deal with it?

This is the kind of thing the noise removal tool in Audacity was born for—and the one thing it's really good at. Here's how to make the most of it.

Select a short clip of room tone (where your actor isn't talking, and there's no noise other than that which you wish to reduce), bring up the noise removal tool, and click "sample". Then, select the entire area you want to quash (typically the whole raw dialogue track) and open the noise remover again. This time, with your frequency blend set at 130 and your reduction set at -8db (over the years, I've found these settings give the best trade-off of noise reduction/quality preservation for spoken word material), then click OK.

Once you've run this—and, if you've got really bad audio, run it a second time with a new noise sample—export your audio as a .wav file. Do *not* overwrite your original .wav file—if anything's gone wrong that you haven't caught yet, you want to be able to go back to the unaltered file.

When you're done with your cleanup, bring up Ardour and build your project. For the purposes of this article, I'm going to assume you have one room in which the characters are conversing,

that there will be sound FX and some music as well.

Your track layout, then, will look like this: two mono tracks for dialogue, two stereo tracks for sound FX and one stereo track for music. You're also going to need one stereo bus for the reverb rooms—all of which I'll get to in the next section. For now, lay out your tracks, pull in your audio and lay it down.

Now comes the edit—arguably the most arduous part of the process. Select your performance keepers, trim them and lay them in. Adjust their timing, and time them with your audio effects. Ardour's non-destructive slide-edit and automatable faders make this process much quicker and easier than it is in other programs—trust me on this one. In my misspent youth, I edited three full-length films and several short films and corporate videos in destructive editors like Audacity, and that's a kind of hell I wouldn't wish on my worst enemy.

In dramatic terms, the edit is where your emotion is created, sustained and communicated. Get the edit close to correct, then give your ears a break before moving on to the next step—you're going to need yourself in top form to make the magic happen.

Reverb, Panning and Mixing: Where the Magic Happens

My 18-hour full-cast audiobook *Down From Ten* took place entirely in a single house. Since the house and the house's geography are key to the drama, it was essential to make the environment a character, with every room as distinct and sonically interesting as the characters who were performing in it. That sense of place is one of the things mentioned frequently in reviews by listeners—"immersive", they call it.

But I'm not alone in doing this. Dirk Maggs of Above The Title Productions made his name pioneering this sort of deep audio texturing for BBC radio, and it's the kinds of touches and flourishes I'm about to describe that make the difference between a pleasant and listenable production and a full-blown audio movie.

To create that sense of place, start by building a "room". The characteristics of the materials, angles and distances in a space are what makes the difference between a kitchen and a living room, or a cathedral and a park outdoors. For this sample project, let's assume you need a bedroom and a cathedral, because most of us know what each of those sound like.

This is where that stereo bus comes in. Rename it "Cathedral". From your two dialogue tracks, put "sends" in before the faders (in Ardour, everything is inserted by right-clicking above the fader in the mixer window). Route those sends to "Cathedral"—left-to-left, right-to-right. Double-click on each send, and adjust your send level and pan—this is how hard your sound will hit the reverb room and on what side. You can't automate these, but there's rarely a need. Set the send level at -12db or a little lower for now, and leave the pan pot dead center. Now, duplicate these settings on the other dialogue track, and then set up another send on the sound FX track.

Popping over to the Cathedral bus, insert a reverb. Linux has a number of excellent reverbs available in LADSPA, LVR2 and VST flavors (just less than a dozen at my last count). Pick

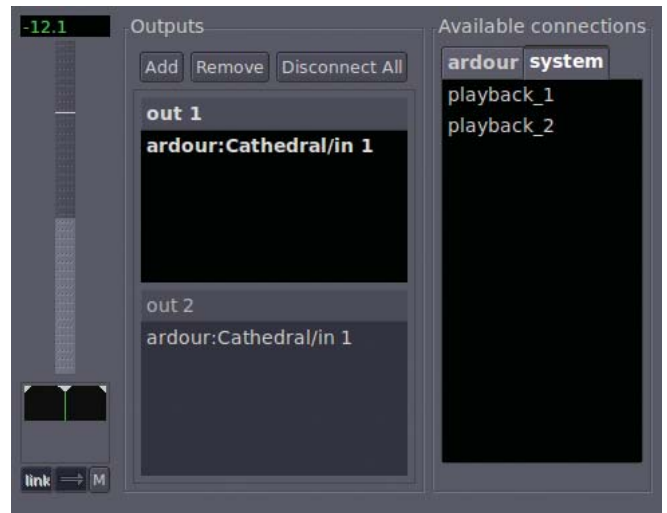


Figure 1. Controlling the intensity with which the audio hits the reverb is key to managing the characteristics of your room.



Figure 2. Piping the Audio through the Reverb Room by Creating a Send

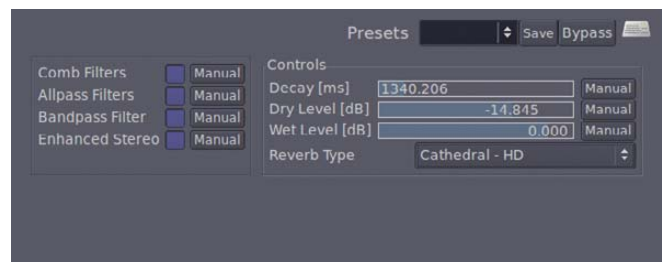


Figure 3. Setting Up the Reverb Room Using the TAP Reverberator

your poison. For my purposes, I'm using the somewhat limited but most excellent TAP Reverberator. Because we're doing this room as a cathedral, we'll use the "Cathedral HD" preset. As you bring it up, you'll notice things like wet/dry balance, decay



Figure 4. Setting the Pan Pot to “Write”

time and other controls—adjusting these will give you an innate feeling of the space you’re in. For these purposes, let’s just use the default setting.

You now have your room. If you click play, you’ll hear everything playing in that room. But, what if it’s too live? The reverb might be too hot in certain frequency ranges—cathedrals tend to sound a bit sharp, because the processor simulates all that stone and vaulting you find in a real cathedral. You can finesse your room further by adding an EQ to the cathedral bus—for example, the LADSPA “Multiband EQ” plugin—and selectively damping some frequencies. For voice in a cathedral, dumping the extreme low end and extreme high end, and subtly damping 800Hz, 1KHz and 2,400Hz can sweeten up the room wonderfully. You can control (and automate) the volume of your reverb with the fader on the bus.

Now, the real magic happens. Let’s say you want to have one character enter the room from the left and approach the other character, who the audience is standing near (in audio space). If the POV character’s voice is on dialogue 1 and the visitor is on dialogue 2, here’s what you do.

Set dialogue 2’s fader to “write”, and start it off at a low volume. Push play, and gradually raise the volume as the character crosses the room to the other character. When he reaches his destination, his fader’s endpoint volume should be the same as the other character’s fader volume level. You’ll notice that you’re already hearing that sense of “approach”, as if the character is walking straight toward you—that’s because as you change the fader volume on the track, you’re changing the wet/dry balance for the reverb. The drier the signal, the closer it seems; the wetter the signal, the farther away it seems.

But, you wanted the character to come in from the left, not walk toward you from deep in the distance. So, set that fader back to “play” and return the transport to the beginning of the walk-in.

Now, set the track’s pan pot to “write”, and drag it all the way off left. Press play, and, in tandem with the increasing volume, slide that pot from extreme left to just left of center,

arriving at a stop just off-center at the same time the volume fader reaches its maximum. Set the pot back to “play” and give it a listen. You’ll hear the character walking into the room (without footsteps), approaching through sonic space. (If you want footsteps, put them on the sound FX track and repeat the same automation moves for them.)

Now that you’ve got a handle on how these things interact when you start moving pots and faders around, it’s time to mix your project. Do it now, adjusting the relative volume levels of everything (and their positions in the room) in “write” mode, then switch back to “play”. Watch the redline while you’re doing it—you don’t want to create a mix that clips on exports. Keep everything below 0db, and you’ll be fine.

Go ahead. I’ll be here when you get back.

Linux News and Headlines Delivered To You

Linux Journal topical RSS feeds available



http://www.linuxjournal.com/rss_feeds

Best Plugins for Spoken Word

- **Tube Amp (overdriven):** for subtle distortion, as of a voice coming over a speaker.
- **Ring Modulate:** for radio interference, mechanization, and alien vocal effects.
- **EQ:** for damping, booming, flattening and sweetening. Good EQ plus a good actor can yield dozens of distinct vocal textures from a single mouth.
- **Reverb (with a light touch):** for a tiny bit of extra resonance to add a sense of authority or gravity. Experiment with all the available FOSS reverbs—they all have subtly different characteristics and are useful in different situations.

EQ, for That Extra-Special Touch

Without a doubt, EQ is the single-most powerful tool in audio-drama engineering. It's through EQ tweaks that you can turn a regular spoken voice into a tinny phone voice, a voice shouting through a wall from a neighboring room or a voice of great authority. For the human voice, a good 15-band parametric EQ (such as the Multiband EQ plugin that comes with almost all Linux distros) will do just fine. As each voice is different, there aren't

Without a doubt, EQ is the single-most powerful tool in audio-drama engineering.

many universal recommendations—even so, here's a few that will apply almost anywhere:

1. Dump everything below 125Hz. Almost all human voices bottom out at around 125. Anything lower is noise, and that noise can screw up your reverb, so just mute the low bands. Ditto for anything above 16KHz.
2. Vocal crispness comes in above 2KHz. Your consonants are up here—the difference between good articulation and bad is the quality of the consonants. Damping these frequencies makes the voice muddy, boosting them makes it crisp. You can tune the performance subtly by tweaking the consonants. Similarly, if your actor has harsh consonants, damping these subtly will take out those nasty sibilants and dentals.
3. Vocal richness and texture comes in the middle of the range: 300–600 for tenors and around 500–800 for altos

and sopranos. These frequencies create vocal warmth when they're higher than the others, and damping them down makes for a cold, emotionally distant sound.

4. Authority comes in the low registers: 150–400 for most men and 250–600 for most women. Boosting these lends more gravity and urgency to the voice, cutting them does the opposite.

But, when it comes to constructing a good drama mix, you'll also want to take a look at your music and sound FX. Because ambient room tone, most music and many sound effects are very strong in the mid-range frequencies, they easily can swamp your voices, even if the music is at a very low volume compared to your voices. Your actors can get lost in the mix, even when you've been sure not to mix the music too hot.

To solve this, slap an EQ on your music and effects channels, and cut those middle frequencies just a bit—5db usually does the trick, and adding another subtle reduction up around 2,500Hz will make room for your consonants. Doing this lets the music and sound FX fill in the sound around your actors, rather than running over them, and the resulting sound will be much cleaner and more pleasing.

Mastering

Once you've finished your editing, mixing, reverb, EQ and all the other various sweetening and tweaking you want to do, it's time to export. Assuming you've got a good mix that doesn't clip—or clips only on one or two occasions—you won't need a compressor. Throw the LADSPA Declipper on the master bus just for safety and export your project.

If you're destined for CD distribution, you may want to pull JAMin into this—it's a bit esoteric if you're new to this game, but well worth learning. The easiest way to use it is by setting up an insert on the master bus (see Dave Phillips' many excellent articles on LinuxJournal.com for instructions on using JAMin).

If you're aiming for MP3 distribution, import your exported .wav into Audacity and use it to master your MP3. This gives you the chance to do a final waveform inspection, looking for pops, skips or dead spots (as can sometimes happen on export with some systems) and to do your tagging as the file spools out.

Wrapping It Up

And that's really about it. Whether you're soundtracking a film, doing a radio drama or an audiobook, the procedure and tools are very similar. A few plugins (far fewer than most people think you need), a little care and some excellent FOSS software can yield some ear-popping audio goodness. Once you learn the basics here, all that's really left is practice. Which reminds me, I have another audiobook due next week. Best get back to it. ■

Dan Sawyer is the founder of ArtisticWhispers Productions, a small audio/video studio in the San Francisco Bay Area where he produces full-cast audiobooks and radio dramas. He is the author of *The Clarke Lantham Mysteries*, the Parsec-nominated *The Antithesis Progression*, *Down From Ten* and several short stories. You can find out more about him and his various flavors of insanity at www.jdsawyer.net.

Unison, Having It Both Ways

Unison is a program for bidirectional synchronization of files using the rsync algorithm.

ADRIAN KLAVER

Unison is a file synchronization tool that supports bidirectional updates of files and directories. It uses the rsync algorithm to limit the size of updates to only what has changed. The program runs on UNIX-based systems as well as Windows machines and can sync between them. It uses SSH as the default transport method.

Unison originated as a research project at the University of Pennsylvania, and although that project has moved on (Harmony/Boomerang), the program continues on. This has led to some misconceptions as to the status of the program that are answered at the URL provided in the Resources section of this article. The short version is that the program is very much alive and in active use by many. In this article, I demonstrate the setup and basic usage of Unison.

The first step is installing Unison on your machine. It is available in package repositories, although probably as an older version. For instance, on my version of Kubuntu, 10.04, the package version is 2.27.57. The latest stable version is 2.40.61, at the time of this writing. A lot of usability, performance and cross-platform improvements have been made between those versions, so for this article, I use 2.40.61. For Windows and Mac machines, current binaries are available from the Web site's download page. For other platforms, it is necessary to build from source.

Complete instructions are available in the manual's Install section (see Resources). Unison can be built as either a text or GUI (GTK) version, assuming the appropriate libraries are available. The GUI version also can be run in text mode via a command-line switch, so it is the most flexible. The text-only version is handy for servers where graphical libraries are not installed. Note: when I built from source, the Quit button did not show up in the GUI toolbar. As you will see later, that is more annoying than fatal.

Unison can be used and customized in a variety of ways, but trying to cover all the bases would deplete the magazine's ink allowance for this issue. Instead, I demonstrate the way I use Unison on a daily basis as an illustration of what is possible. My basic setup consists of Unison on three machines: my home desktop, my laptop and an Amazon EC2 instance.

Before going into my actual setup, bear with me as I explain the basic operating principles behind Unison. The starting point for Unison are two roots. These are just directory paths that will be synced. Both paths can be local, or one can be local and the other remote. The preferred method of connecting local to remote is SSH, and sections in the manual and wiki (Resources) describe how to set up Windows to use SSH.

Unison also has a socket method where an instance of the program is set up as a server listening on a socket. The catch is that the data is transferred unencrypted, so use at your own risk. To keep track of state and configuration information, Unison creates a private directory. In the absence of an ENVIRONMENT variable saying otherwise, that directory is \$HOME/.unison

on UNIX-like systems. On Windows systems, it's either \$USERPROFILE\.unison or \$HOME\.unison or c:\.unison, depending on the settings for USERPROFILE and HOME.

Now, let's move on to the actual setup: desktop <--> EC2 server <--> laptop. The important part of the above is that each <--> represents a different Unison root pair. The desktop Unison program has no knowledge of what is on the laptop, only what it can see on the EC2 server and vice versa. The role of the EC2 Unison instance in this scenario is to keep a particular set of files in sync with either the desktop or laptop, depending on who is asking. In simplest terms, the EC2 Unison is a server to the desktop/laptop Unison clients, although this is not strictly true due to the bidirectional nature of Unison. Either Unison instance in a pairing can serve or receive files. Connection between the desktop/laptop and the EC2 server is done over SSH using public/private key authentication. For the purposes of this article, I am using a subset of the paths I normally keep in sync.

Although it is possible to run Unison by supplying arguments to the program on the command line, there is a better way—that is, to create a profile file for each root pair that is synced. These are *.prf files stored in the ~/.unison directory. When you start Unison in text mode, you can supply it a some_name argument that maps to a some_name.prf file. In GUI mode, Unison searches the ~/.unison directory for *.prf files and presents them as choices (Figure 1). For this article, I am using lj_article.prf (Listing 1).

I use this same profile on both the desktop and laptop machines. Let's go through the lines and detail what they mean.

Listing 1. ~/.unison/lj_article.prf

```
#Preferences available 2.27.57+ except as noted
root = /home/aklaver
root = ssh://alabama/lj_article
path = software_projects/linux_journal_articles
path = software_projects/aws
path = software_projects/track_stocks
path = .unison
ignore = Name *~
# BelowPath in version 2.40.61+
ignore = BelowPath .unison/*
ignorenot = Name *.prf
backup = Name *
backuploc = central
backupdir = unison_backup
maxbackups = 3
# Copy preferences version 2.30.4+
copythreshold = 100
```

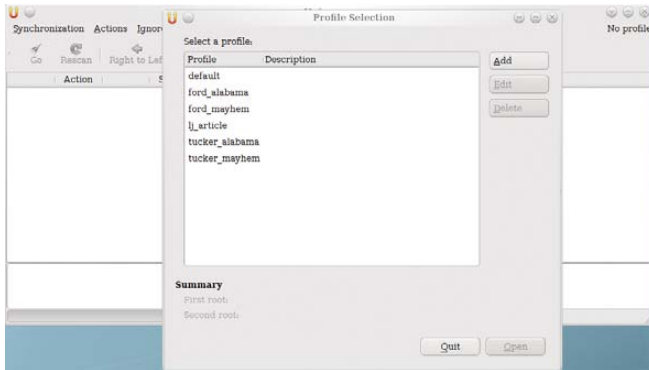


Figure 1. Profile Choices

An aside, what I talk about here are called preferences in the Unison manual. The first two labeled root are the root pair I'll be syncing. The paths can be either absolute or relative to the directory in which Unison is started. The first root is an absolute one defining my home directory. The second takes a little more explaining. The `ssh://` indicates that SSH is used to make the connection to the remote machine. In this case, I have used an SSH alias "alabama" to refer to the remote EC2 host. This expands as `aklaver@host_name` and assumes the default SSH port of 22. Should your machine be running at a different port, the form is `ssh://user@host_name:port_number`. The `/lj_article` portion is the directory path relative to my home directory on alabama. If I wanted to specify an absolute directory path, it would look something like `ssh://alabama//home/aklaver/lj_article` (note the two forward slashes).

With just the two roots specified, running Unison would sync everything in my local home directory with the `lj_article` directory on the remote machine. This is not what I want, so I have included some path preferences. A path preference is exactly that—a path to some part of the root that you want to sync. Presence of a path preference restricts the sync to the paths specified. A path is relative to the root of the replica, and the separator character is the forward slash (it will be converted as needed). The important part is that paths are relative to the root. This means that the data I sync on either my desktop or laptop is relative to `/home/aklaver` and becomes relative to `/home/aklaver/lj_article` on the EC2 server. A path preference can point to a single file, a directory or a symbolic link (on Unixen). The path preference is inclusive, in that if you specify a directory, it syncs everything in that directory and below.

To filter what is synced in a particular path, or in general, ignore preferences are used. The first form used here, Name 'name' is one of four types; the others being Path 'path', Regex 'regex' and BelowPath 'path' (this is new to 2.40.16). There is a lot of power here, explained fully in the Path Specification and Ignoring Paths sections of the manual. The `ignore = Name *` preference uses globbing patterns to ignore any tilde files in the paths. The second ignore preference `ignore=BelowPath .unison/*` is more specific. It causes anything in `.unison` and paths below it to be ignored. This is in

contrast to the regular Path 'path' form that matches only the path specified. I am doing this to avoid syncing the archive files and the backup directory (more on that later).

Now I have a quandary. I have said that I want to sync the `.unison` path with `path=.unison` and ignore it with `ignore = BelowPath .unison/*`. At that point, nothing would be synced, and the preferences would be useless. That is where the `ignorenot` preference comes in. By specifying Name `*.prf`, I am saying "disregard what I said about ignoring what is in `.unison` for the specific case of `*.prf` files". The end result is that I sync only the profile files in `~/unison`.

The next four preferences configure the backup option. The presence of a `backup=` preference signals that backups are to be done. A backup is done when a file is changed or deleted by Unison. The backed-up file is kept on the machine where the change is applied by Unison. So if you make a change to a file on your local root, when it is synced to the remote root and the change is applied to the remote file, the previous version of the remote file will be saved on the remote machine. In this case, using Name `*` means back up everything. It is possible to use the path specifications mentioned above to restrict that. There also is a `backupnot` that operates like `ignorenot`.

The `backuploc` preference specifies where the backups are to be stored. The options are `local` and `central`, where `local` has the backup files being stored alongside the original files, and `central` moves the backup files to a location as defined in `backupdir`. With `backuploc=central` and no `backupdir` preference, the backups will be found in the directory `backup/` in `~/unison`. This is why I have the `ignore=BelowPath .unison/*` preference in the profile. Although in this case, I am using `~/unison_backup` to store backups, I have other profiles using `~/unison/backup`.

The `maxbackups` preference is self-explanatory; it restricts the number of backed-up files to the three most-recent versions. In its absence, the default is two versions. I use the `central` method of backup, because I don't want to sync the backups. Per my article, "Using `rdiff-backup` and `rdiffWeb` to Back Up and Restore" (*LJ*, December 2010), I use `rdiff-backup` to keep versioned backups already. I do like to keep Unison backups close at hand though, as insurance if I make an inappropriate change to a profile and cause a file or files to disappear, or in case I make the wrong decision on which direction to propagate a change. Besides, I am a firm believer that you cannot have too many backups.

The `copythreshold` preference is one of the performance enhancements in recent versions. Since 2.30.4, it has been possible to tune Unison when doing whole file transfers or when redoing an interrupted transfer. The `rsync` code, as used by Unison, is designed more for doing changes to files than for moving over entire files. Setting a `copythreshold` to a positive integer causes Unison to farm out the job of copying complete files. If set to 0, all whole file transfers will be farmed out. Otherwise, a number greater than 0 refers to a file size in kilobytes above which Unison will use another program. By default, that program is `rsync`, although that can be changed using `copyprog`. Unison also uses `rsync` to resume interrupted large file transfers.

In the same vein, Unison 2.30.4+ will, without any setting

needed, keep track of an interrupted transfer of a new directory. Whatever has been transferred will be stored in a temporary directory away from the intended destination, on the receiving side, and on the next transfer, it resumes with the untransferred files.

At this point, running this profile is anti-climatic. The first run pops up a warning about there not being archive files present for the root pair (Figure 2). The archive files are where Unison stores its information about the roots and files synced. Pressing Enter starts the process of the initial scan and the population of the archive files. After the scan is done, the GUI presents the choices available for file transfers (Figure 3). In this case, because the remote path is empty, all arrows point from local to alabama. There are two options here: arrow down through each choice and press Enter, or type the letter g as a shortcut for the Go button and start the transfer of all the files at once. On subsequent transfers, the choices for each directory/file may well be different, depending on the changes made on each root (Figure 4). The choice presented is not set in stone, and you can change it with the arrow and Skip buttons on the toolbar, dealing with each file as necessary.

For batch actions on the files, use the Actions menu (Figure 5). Also worth mentioning is the Ignore menu item. It is very handy in that it will write an ignore preference to your profile matching the item you select using the ignore type you select. With version 2.40.1+, a profile editor is built in to the GUI interface. So, if you

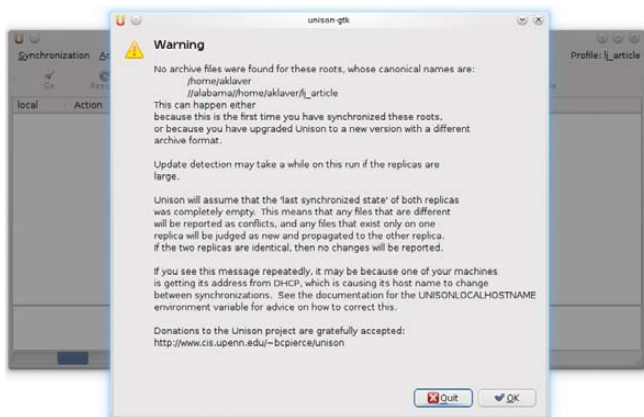


Figure 2. First Sync of Root Pair, Archive Warning

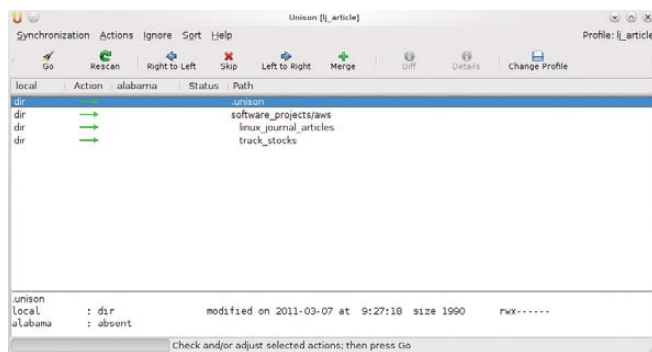


Figure 3. Syncing First Time

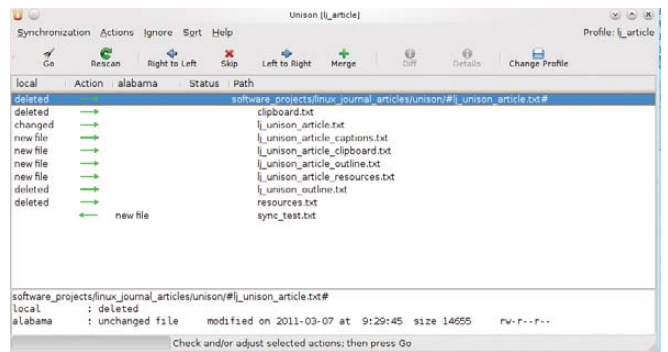


Figure 4. A Sync with Changes to Both Roots

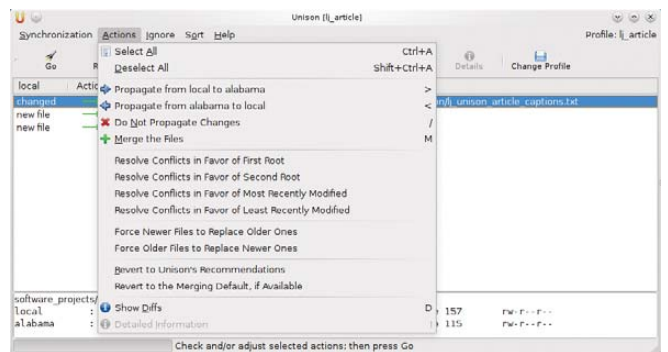


Figure 5. Action Menu Showing Bulk Action Choices

decide you want to undo the ignore preference, simply go to the Synchronization menu and then Change Profile.

As mentioned previously, when I built the GUI, it did not have the Quit button. Typing q as a shortcut still works, or you can go through the Synchronization menu to get to Quit.

So, what happens if a file has been changed on both roots since the last sync? The default action is to skip syncing that particular file, and this is represented by a ? in the file listing (Figure 6). It is left up to the user to decide what to do in that situation. That said, preferences can be set to force Unison's hand (search for "force" in the manual). Should you decide to sync, some tools are available to help you make a decision on what to do with a file. For nonbinary files, you can use the Diff button to look at the diff between a pair of files to help figure things out. There also is the Merge button. This requires some extensive preparation, and I have not used it since I first tried Unison many years ago (at the time there were known issues with the merge code). Since then, considerable work has gone into making merge functional, and it is on my to-do list to give the new code a spin. The manual provides detailed instructions for setting up merge, if you are so inclined.

With all these files flying around, how safe is your data? For a more complete discussion, see the Invariants section of the manual. The essence of that section is that Unison guarantees certain behavior for the paths synced as well as its own information about the process, at any moment in time. For paths, that is that they

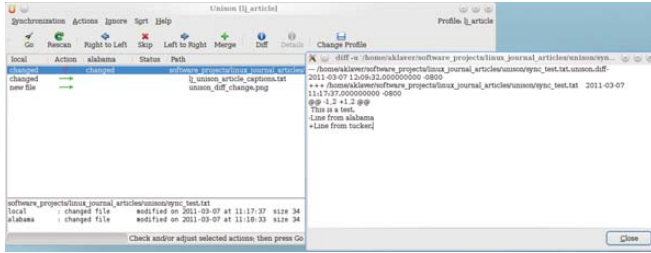


Figure 6. Sync with Changes to Both Roots. Showing Diff Also

are either at their original state or their completed changed state. The same principle applies to Unison's own private data—it is either in its original form or reflects the current successfully completed syncs. My own experience is that Unison is very forgiving of operator error. A sync interrupted either intentionally or by mistake is recoverable.

I use Unison on a daily basis to help me keep track of file changes on my laptop and desktop. Using an EC2 server as an intermediary host, I am able to bounce the changes from either of my personal machines off the cloud and back to the other machine. For many years now, Unison has been a must-have program for me. I hope you find it useful too. ■

Adrian Klaver lost a term paper once, due to no backups. Ever since, he has had a bit of an obsession with backup utilities. When not worrying about the state of his data, he can be found trying to outsmart his digital camera.

Resources

Unison Web Site: www.cis.upenn.edu/~bcpierce/unison

Harmony/Boomerang Project: www.seas.upenn.edu/~harmony

Project Status: www.cis.upenn.edu/~bcpierce/unison/status.html

Manual: www.cis.upenn.edu/~bcpierce/unison/download/releases/stable/unison-manual.html

Wiki: <https://alliance.seas.upenn.edu/~bcpierce/wiki/index.php>

“Using rdiff-backup and rdiffWeb to Back Up and Restore” by Adrian Klaver, *LJ*, December 2010: www.linuxjournal.com/article/10701



LINUX
JOURNAL

SUBSCRIBE TODAY!
WWW.LINUXJOURNAL.COM/SUBSCRIBE

It's Always DNS's Fault!

How do you fix a DNS server that isn't broken? Find out in this episode of Tales from the Server Room.

It's always better to learn from someone else's mistakes than from your own. In this column, Kyle Rankin or Bill Childers tells a story from his years as a systems administrator, and the other chimes in from time to time. It's a win-win: you get to learn from their experiences, and they get to make snide comments to each other. Today's episode is narrated by Bill.

Some Days, You're the Pigeon...

I was suffering, badly. We had just finished an all-night switch migration on our production Storage Area Network while I was hacking up a lung fighting walking pneumonia. Even though I did my part of the all-nighter from home, I was exhausted. So when my pager went off at 9am that morning, allowing me a mere four hours of sleep, I was treading dangerously close to zombie territory.

I looked at the pager and saw that someone had pushed the dreaded "Panic Button", a Web-based tool we'd made that would alert the larger IT team to an unknown high-priority issue. I sat up, reeling and asked my wife to begin the caffeine IV drip that would wake me up while I slowly started banging synapses together, hoping for a spark. According to the report, our DNS infrastructure was timing out on a lot of requests, causing overall site slowdown. I had to re-read that e-mail several times for it to sink into my oxygen-and-sleep-deprived brain. How could DNS be timing out, and why hasn't our internal monitoring caught that? We monitored the DNS servers and service levels internally, and if performance was bad, I should have been the first to know. Something smelled really funny, and it wasn't me, despite the pneumonia-induced fever.

[Kyle: I'll pretend I didn't see the "something smelled funny" comment, as it's too easy. The funny thing here was that we had a long-standing tradition of DNS being blamed whenever there was any sort of networking problem. I've said before that people tend to blame the technology they understand least. This case was one of the first times that it actually seemed (at least on the surface) to be a DNS issue.]

I started checking on things as I dialed in to the conference call for this issue. Our monitoring system said nothing was awry, and response times for DNS were normal. I ran a few nslookups past the DNS server, and it replied in its usual speedy fashion with the expected result. I flipped through the logs as well, and they showed nothing out of the ordinary. What was going on?

At this point, I probably should describe how the

company's DNS infrastructure was set up. It had two main data centers: A and B. Each data center had a load-balanced pair of DNS servers set up as active-passive, and the public virtual IP addresses for each were published as the NS records for each domain we serviced. That would cause each data center to service half the DNS load for any set of requests, and due to each data center having a load-balanced pair of DNS servers, we could tolerate a failure of a DNS server without any degradation in customer-facing service.

[Kyle: The beauty of a system like this is that even though DNS has automatic failover if you have more than one NS record, if a DNS server is down, you generally have to wait the 30 seconds for it to time out. That 30-second delay was too long for our needs, so with this design, we could take down any individual DNS server and the load balancer would just forward requests to the remaining server in the data center.]

Anyway, I continued troubleshooting. On a hunch, I started running nslookups against a few domains from my home—maybe the problem was visible only from the outside. Oddly enough, nslookups succeeded for the most part, except for those pointing to one of our most active sites, which had Akamai as a content delivery network (CDN). Akamai requires that you configure your DNS using CNAME, or alias, records, so that its CDN can spider and cache your content. The CNAME records look something like the following:

- A CNAME pointing `www.ourdomain.com` to `ourdomain.com.edgesuite.net`.
- A CNAME pointing `origin-www.ourdomain.com` to `ourdomain.com`.

Surely enough, external requests that hit data-center A would time out and wind up failing over to data-center B. Typical DNS timeouts put this on the order of 30 seconds, which is unacceptable for any kind of commercial Web site. Since Akamai was involved, and the main site I found that was affected was utilizing Akamai, I made the call to Akamai support for assistance.

[Kyle: I can't count how many times I've used personal servers that are colocated outside a corporate network to troubleshoot problems. It can be invaluable to have a perspective on the health of a network service that's completely detached from your corporate network. Think of it as another reason to keep your home server on 24/7.]



KYLE RANKIN



BILL CHILDERS

...and Some Days You're the Statue.

At this point, I had the Akamai folks looking at the issue from their end, and after a couple hours of back-and-forth troubleshooting, they announced that the problem was *not* with their setup, and it had to be something in our DNS servers. However, all the tests I did within the data center returned correctly and instantly. It was only tests from the outside that timed out and failed to data-center B, and even those were sporadic. By this time it was after noon, and even the caffeine IV drip was wearing off. I was tired, sick, and my brain was not firing on all cylinders.

It was at about this time that I started getting e-mail messages from my pointy-headed boss about how I was “not doing anything” to fix the problem, and he wanted status updates every half-hour on how things were progressing. I replied that I either could update him every half-hour or work on the problem like I had been.

That e-mail message made my cell phone ring, instantly. My boss was on the line, demanding I reboot the DNS server in an attempt to “fix” the problem. However, without any signs of anything wrong on the DNS server other than the failed queries, I was reluctant just to bounce the server, because if the error condition cleared, we'd have no further way to collect information about the problem.

Kyle wound up finding an odd bug in an older version of BIND where a counter rollover caused weird things to happen. That bug supposedly was fixed in the version we were running, but it was a lead to go on, so I made the call, reluctantly, to restart the DNS service on the primary DNS server. Much to my surprise, once we did that, the timeouts stopped occurring. All of a sudden, our DNS infrastructure was back up to 100%, and site performance returned to its normal levels.

[Kyle: It's worth noting that the DNS process on this system had been up and stable for more than a year. Although it was technically possible that an internal uptime counter rollover (like the 498-day uptime rollover on older Linux kernels) could cause strange behavior, it really seemed like grasping at straws, and I was surprised when it seemed to fix the problem. That, of course, brought up other questions though—were we going to have to bounce our DNS service every year?]

My boss called me after the site performance returned. To this day, I don't know if the call was to gloat about his “solution” being correct, or if he called to chastise me for waiting so long to restart the DNS server. I explained that although the issue was no longer occurring, we had zero information as to the root cause of the issue, and that it was not “fixed”. Rebooting things randomly is what Windows admins do when things act up. UNIX system administrators tend to try to reach the heart of the issue. At the end of the call, it was apparent he didn't care about a fix. He simply wanted the site back to its normal performance. I finally passed out, exhausted, yet feeling worried that we had not seen the last of this issue.

Listen to Your Hunches

Fast-forward a couple weeks later. I'm feeling better, the pneumonia's defeated, and I'm back at work. True to my gut feeling, the issue spontaneously re-occurred again. People around the office started panicking, blaming the DNS infrastructure and flailing about in general. Kyle and I immediately set to troubleshooting again, but just like the time before, we couldn't find a single thing wrong with the DNS server. This time though, I was more on the ball,

However, without any signs of anything wrong on the DNS server other than the failed queries, I was reluctant just to bounce the server, because if the error condition cleared, we'd have no further way to collect information about the problem.

and I remembered that the DNS servers were fronted by a load balancer. On a hunch, I asked the network engineer if he had noticed any issues with the load balancer. He investigated and saw weirdness in the log files of the unit. After a little more conversation with him, he agreed there was a problem on the primary load balancer, and the decision was made to fail over to the backup load balancer. Once the failover happened, the DNS issue we were seeing cleared up again. All along, we were fighting a flaky load balancer, not an issue on the DNS server.

Lessons Learned

Several lessons came out of this issue. The biggest one is that it's easy to lose sight of all the technologies that come into play in a modern data center. My team was responsible for the UNIX systems, so we naturally tested and troubleshot the servers, but initially didn't think that the network possibly could be a problem. Always be sure to look outside your realm of responsibility, as the problem may lie there.

[Kyle: It's funny, because I've known people who default to looking outside their realm of responsibility when there is a problem. We should have been clued off when we noticed that internal DNS requests always worked while external ones (ones through the load balancer) were flaky. But like Bill said, once we rebooted the service and the problem disappeared, there wasn't any troubleshooting left to do.]

Another lesson was one I already knew, but it was highlighted that day. Rebooting a server that's misbehaving is an absolute last resort, as you'll wind up losing the problem in the first place, and you'll never figure out what the root cause is.

In all, although this issue did cost the company money that day due to the poor site performance, it was a good learning experience. I think of this incident a lot when designing new infrastructure or when faced with a new and unknown problem. It reminds me to be thorough when troubleshooting, to look at every possibility and not assume anything. ■

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

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

American made Utility Kilts for Everyday Wear

UTILIKILTS.com

Archie McPhee

Supplying the world with impractical weirdness for over 25 years!

mcphee.com

INNOVATION ON THE GO
ORDER YOUR BEAGLE BOARD FROM DIGIKEY.COM

AVAILABLE EXCLUSIVELY AT DIGI-KEY

beagleboard

only \$149⁰⁰

LOW-COST, NO FAN, SINGLE-BOARD COMPUTER

Digi-Key CORPORATION

www.digikey.com

Polywell Solutions

More Choices, Excellent Service, Great Prices!

Quiet Storage NAS/SAN/iSCSI

NetDisk 8074A - 4x Gigabit LAN
72 Bay 144TB \$26,999 - RAID-5, 6, 0, 1, 10
74 Bay 222TB \$36,999 - Hot Swap, Hot Spare
 - Linux, Windows, Mac

5048A 5U-48Bay 144TB \$26,999
 RAID-6, NAS/iSCSI/SAN Storage
 SATA, 4x GigaLAN

9020H 20Bay 60TB \$9,999
 - 4x Gigabit LAN
 - RAID-5, 6, 0, 1, 10
 - Hot Swap, Hot Spare
 - Linux, Windows, Mac
 - E-mail Notification
 - Tower Case

9015H 15Bay 45TB \$7,750
30TB \$4,999
 - Dual Gigabit LAN
 - RAID-5, 6, 0, 1, 10
 - Hot Swap, Hot Spare
 - Linux, Windows, Mac
 - E-mail Notification
 - Tower Case

2U12B 2U-12Bay 36TB \$6,999
 SATA II, RAID-6, 2x GigaLAN
 NAS/iSCSI/SAN Storage

4U24A 4U-24Bay 72TB \$12,950
 RAID-6, NAS/iSCSI/SAN Storage
 Mix SAS/SATA, 4x Giga/10Gbit LAN

Netdisk 8000V Quiet Performance
 - Dual Gigabit LAN
 - RAID-5, 6, 0, 1, 10
 - Hot Swap, Hot Spare
 - E-mail Notification
 - Tower Case

Silent Eco Green PC
 - Intel® / AMD® CPU
 - Energy efficient
 - Quiet and Low Voltage Platform
 starts at **\$199**

LD-001

Polywell OEM Services, Your Virtual Manufacturer

- Prototype Development with Linux/FreeBSD Support
- Small Scale to Mass Production Manufacturing
- Fulfillment, Shipping and RMA Repairs
- 20 Years of Customer Satisfaction
- 5-Year Warranty, Industry's Longest
- First Class Customer Service

888.765.9686
 linuxsales@polywell.com
www.polywell.com/us

Polywell Computers, Inc 1461 San Mateo Ave. South San Francisco, CA 94080 650.583.7222 Fax: 650.583.1974

NVIDIA, ION, nForce, GeForce and combinations thereof are trademarks of NVIDIA Corporation. Other names are for informational purposes only and may be trademarks of their respective owners.

First Brazil, Then the World

A high-leverage business project for free software and low-cost hardware.

DOC SEARLS



Too often we hear a project or an idea dismissed because it tries to “boil the ocean” instead of something more modest, like poaching an egg. But the problem with modesty is that we’ve seen the ocean boiled, many times. Linux is the prime example. Linus himself half jokingly talked about “world domination” in the mid-’90s, long before it actually happened. Other examples go back as far as J.C.R. Licklider, whose writings on “man-computer symbiosis” and the “Intergalactic Computer Network” predated the personal computer and the Internet by several decades. Both free software and open source also were intentional ocean-boiling efforts, which continue to pay off today in degrees of social, economic and technical value beyond calculation.

Watching successful ocean-boiling efforts encourages ambitions on the same scale. It certainly has for me (but we’ll leave that one for another EOF—after we start seeing results). It also has for Jon ‘maddog’ Hall, who has been an accessory to Linux’s success right from the start. But, maddog is a modest guy. These days he’s not trying to boil a whole ocean, but instead putting the heat on just one country: Brazil.

What he’s cooked up is Project Cauã (www.projectcaua.org). Although the Web site’s first paragraph describes it as “a Free and Open Source Software and Hardware (FOSSH) project conceived to make it possible for people to make a living as a Systems Administrator/Entrepreneur using FOSSH as the basis”, its ambitions are larger than that. maddog wants to create a whole new business category with lots of jobs in it, while greening everything up by sucking less power off the grid and making cities more livable.

I got some hang time with maddog when he spoke at an MIT gathering earlier this year, and I like where he’s going with Project Cauã. He looks at a city like São Paulo (one of the world’s largest) and sees “tall buildings and apartment complexes with servers in the basements and thin clients in the rooms” and “large new business opportunities for small entrepreneurs, manufacturers, and the economics that both support.” When people ask, “How can I make money with free software?”, his answer is “By renting out thin clients, server space and services to back both, in large volume and at low costs.”

On the technical side, “The software is done, and the hardware is close.” The University

of São Paulo has agreed to design the thin-client hardware and license the design to manufacturers, who will compete to provide components and finished gear at the best prices. This will provide some needed income to the university as well.

“But the main part of this isn’t technical”, maddog says. “It’s economic. Most of our work will go into running the pilot project and preparing educational and documentation materials. We want to make starting and maintaining a business as easy as possible, for as many people as possible.”

Thin clients and fat servers make equally good business and technical sense in a city like São Paulo—and across the rest of Latin America, where the vast majority of people live in urban areas, especially as both television and computing are moving onto the Net and into “the cloud”. A cheap connected device with a good display (also cheap these days) and a capable server with plenty of low-latency storage is a bargain for customers and a business for entrepreneurs who like to be highly connected (literally and otherwise) to their communities.

With its concentrated urban populations, proactive government, cooperative universities, advanced manufacturing facilities and growing FOSSH-friendly technical community, Brazil is a large lever on the rest of Latin America and the world beyond. “After we get it working in Brazil, we’ll translate the materials into Spanish, then run pilots in other countries. Or help others do it for themselves. We’ll all learn by doing.”

For a new entrepreneur, here’s the drill:

- Train on FOSSH system administration.
- Get certified, licensed and bonded.
- Get letters of intent from prospective customers.
- Get underwritten loan commitments from banks.
- Buy and install the gear.
- Deliver services.
- Get feedback from services.
- Improve documentation and shared knowledge

with others in the same business.

The most recent developments are timely. Says maddog:

The other main driver for Project Cauã, Douglas Conrad of OpenS Tecnologia, has put together a home-theater thin-client solution that we will start selling through the “Entrepreneurs”. This will show people that the SA/E (system administrator/entrepreneur) model can work and that SA/Es can sell things, support clients and make money.

We want to show the prototype at FISL (June 28–July 2nd) in Porto Alegre, Brazil, and have it ready to start being sold by SA/Es shortly after that. We will probably have intensive training on Project Cauã in the November/December time frame at Latinoware in Foz do Iguaçu, Brazil.

Right now, maddog and friends are evangelizing the project and raising funds. maddog says, “We need that funding because volunteers only get us so far. Some things require funding, such as legal work, working with the government, paying fees for registration, typing and clerical work, documentation, buying hardware and other areas. Once we have the money guaranteed, we can start hiring and begin the project in full force. It should take six months from that point until we start a pilot.” His goal for funding at the outset: “Three million US dollars, guaranteed.”

For investors, the “play” isn’t Project Cauã, but the entrepreneurs it bootstraps. For the biggest and most aggressive investors, \$3 million is an ATM withdrawal. My advice for them—and for anybody wanting to heat things up—is to step up to Project Cauã’s table and start placing some good bets.

Postscript: thanks to maddog for his patience while I wrote this up and to podcast.de for some of the quotes. ■

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.

ANYONE INTERESTED IN SAVING MONEY?

Looks like these guys are comfortable overpaying
for enterprise storage. Are You?

“Hewlett-Packard Co. agreed to buy 3Par Inc. for \$2.35 billion” — *Bloomberg.com*

“EMC to Buy Isilon Systems Inc. for \$2.25 Billion” — *Wall Street Journal*

“Dell to Buy Compellent for \$960 Million” — *CNBC*

So what “benefit” will you see by this spending spree, other than higher costs?

The AberSAN Z-Series scalable unified storage platform, featuring the Intel® Xeon® processor 5600 series, brings the simplicity of network attached storage (NAS) to the SAN environment by utilizing the innovative ZFS file system. The AberSAN Z20 is easily found starting under \$20,000.

Who gives you the best bang for the buck?

	3Par InServ F200	Compellent Storage Center Series 30	Isilon NL-Series	Aberdeen AberSAN Z20
Storage Scale-Out	✓	✓	✓	✓
Thin Provisioning	✓	✓	✓	✓
HA Clustering	✓	✓	✓	✓
VMware® Ready Certified	✓	✓	✓	✓
Async / Synchronous Replication	✓	✓	✓	✓
iSCSI / Fibre Channel Target	✓	✓	iSCSI Only	✓
Unlimited Snapshots	✗	✓	✓	✓
Native Unified Storage: NFS, CIFS	✗	✗	✓	✓
Virtualized SAN	✗	✗	✗	✓
Deduplication	✗	✗	✗	✓
Native File System	none	none	OneFS	ZFS 128-bit
RAID Level Support	5 and 6	5 and 6	Up to N+4	5, 6 and Z
Raw Array Capacity (max)	128TB	1280TB	2304TB	Unlimited
Warranty	3 Years	5 Years	3 Years	5 Years
Online Configurator with Pricing	Not Available	Not Available	Not Available	Available



Above specific configurations obtained from the respective websites on Feb. 1, 2011. Intel, Intel Logo, Intel Inside, Intel Inside Logo, Pentium, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. All other trademarks are the property of their respective owners. © 2011 Aberdeen Inc. All rights reserved. For terms and conditions, please see www.aberdeenninc.com page 11 of 11. 11038

888-297-7409
www.aberdeenninc.com/lj038

Cut Execution Time by >50% with WhisperStation-GPU

Delivered ready to run new GPU-enabled applications:

Design

3ds Max
Bunkspeed
Shot
Adobe CS5

Simulation

ANSYS Mechanical
Autodesk Moldflow
Mathematica

MATLAB
ACUSIM AcuSolve
Tech-X GPULib

BioTech

AMBER
GROMACS
NAMD, VMD
TeraChem

Integrating the latest CPUs with NVIDIA Tesla Fermi GPUs, Microway's WhisperStation-GPU delivers 2x-100x the performance of standard workstations. Providing explosive performance, yet quiet, it's custom designed for the power hungry applications you use. Take advantage of existing GPU applications or enable high performance with CUDA C/C++, PGI CUDA FORTRAN, or OpenCL compute kernels.

- ▶ Up to Four Tesla Fermi GPUs, each with: 448 cores, 6 GB GDDR5, 1 TFLOP single and 515 GFLOP double precision performance
- ▶ Up to 24 cores with the newest Intel and AMD Processors, 128 GB memory, 80 PLUS® certified power supply, and eight hard drives
- ▶ Nvidia Quadro for state of the art professional graphics and visualization
- ▶ Ultra-quiet fans, strategically placed baffles, and internal sound-proofing
- ▶ New: Microway CL-IDE™ for OpenCL programming on CPUs and GPUs



WhisperStation with 4 Tesla Fermi GPUs

Microway's Latest Servers for Dense Clustering

- ▶ 4P, 1U nodes with 48 CPU cores, 512 GB and QDR InfiniBand
- ▶ 2P, 1U nodes with 24 CPU cores, 2 Tesla GPUs and QDR InfiniBand
- ▶ 2U Twin² with 4 Hot-Swap MBs, each with 2 Processors + 256 GB
- ▶ 1U S2050 servers with 4 Tesla Fermi GPUs

Microway Puts YOU on the Cutting Edge

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

Configure your next WhisperStation or Cluster today!

microway.com/quickquote or call 508-746-7341

Sign up for technical newsletters and special GPU promotions at microway.com/newsletter



OctoPuter™ 4U Server with up to 8 GPUs and 144 GB memory

1U Node with 2 Tesla Fermi GPUs

2U Twin² Node with 4 Hot-Swap Motherboards
Each with 2 CPUs and 256 GB



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

Microway
Technology you can count on™