OLPC | Arduino | Trim-Slice | vDSO | Elmer | MPD

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community

NOVEMBER 2011 | ISSUE 211 |
www.linuxjournal.com

ONE LAPTOP PER CHILD STATUS REPORT

USE YOUR PHONE TO LOCK YOUR SCREEN AUTOMATICALLY

HACK THIS

BUILD AN ULTRA-LOW-POWER SERVER WITH THE TRIM-SLICE

+

HOW TO: ADD A CUSTOM vDSO TO A LINUX KERNEL

Set Up a Music System That Follows You Around

Get Started Programming the Arduino

Create a Unified Boot Environment on a USB Drive

Understanding Drupal
Administering Drupal
Site Building with Drupal
Learning CCK for Drupal
Learning Views for Drupal
Theming Basics for Drupal
Advanced Theming for Drupal
Introduction to jQuery
jQuery & JavaScript in Drupal
Drupal Module Development

Cut Execution Time by >50% with WhisperStation-GPU

Delivered ready to run new GPU-enabled applications:

Design	Simulation		BioTech
3ds Max	ANSYS Mechanical	MATLAB	AMBER
Bunkspeed	Autodesk Moldflow	ACUSIM AcuSolve	GROMACS
Shot	Mathematica	Tech-X GPULib	NAMD, VMD
Adobe CS5			TeraChem

WhisperStation with 4 Tesla Fermi GPUs

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

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.

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

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

NVIDIA TESLA

intel Channel Partner Premier 2010

AMD Fusion Partner Premier

GSA

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

Microway®
Technology you can count on℠

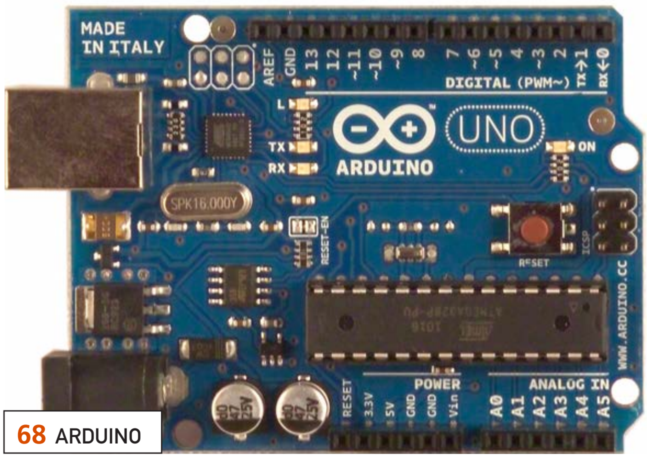
CONTENTS

NOVEMBER 2011
ISSUE 211

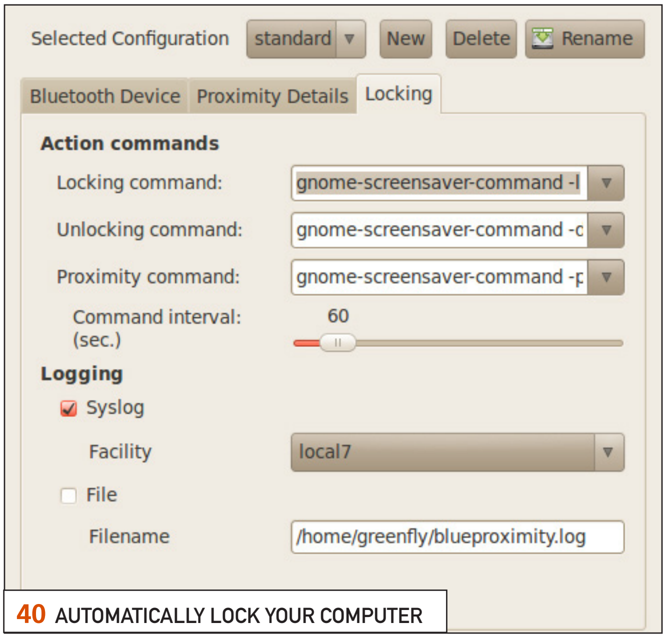
HACK THIS

FEATURES

58 TRIM-SLICE



68 ARDUINO



Selected Configuration: standard

Action commands

Locking command: gnome-screensaver-command -l

Unlocking command: gnome-screensaver-command -c

Proximity command: gnome-screensaver-command -p

Command interval: (sec.) 60

Logging

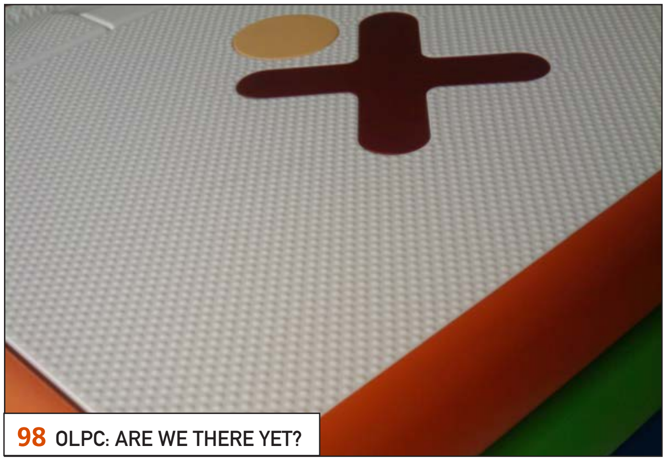
☑ Syslog

Facility: local7

□ File

Filename: /home/greenfly/blueproximity.log

40 AUTOMATICALLY LOCK YOUR COMPUTER



98 OLPC: ARE WE THERE YET?

ON THE COVER

- One Laptop Per Child Status Report, p. 98
- Use Your Phone to Lock Your Screen Automatically, p. 40
- Build an Ultra-Low-Power Server with the Trim-Slice, p. 58
- How-To: Add a Custom vDSO to a Linux Kernel, p. 106
- Set Up a Music System That Follows You Around, p. 82
- Get Started Programming the Arduino, p. 68
- Create a Unified Boot Environment on a USB Drive, p. 88

COVER IMAGE: © Can Stock Photo Inc. / zphoto

LINUX JOURNAL (ISSN 1075-3583) is published monthly by Belltown Media, Inc., 2121 Sage Road, Ste. 310, Houston, TX 77056 USA. Subscription rate is $29.50/year. Subscriptions start with the next issue.

LINUX JOURNAL

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

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

Proofreader	Geri Gale

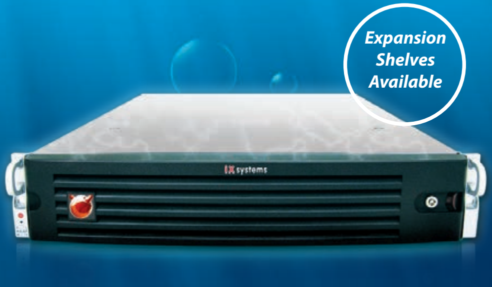
Publisher	Carlie Fairchild publisher@linuxjournal.com
Advertising Sales Manager	Rebecca Cassity rebecca@linuxjournal.com
Associate Publisher	Mark Irgang mark@linuxjournal.com
Webmistress	Katherine Druckman webmistress@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

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

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

TrueNAS™ 2U Appliance: You Are the Cloud

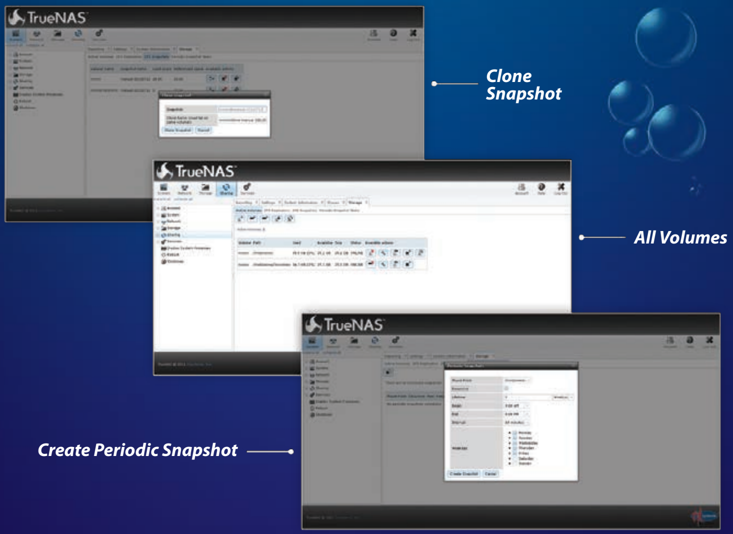


FreeNAS 8 POWERED

Storage. Speed. Stability.

With a rock-solid FreeBSD® base, Zettabyte File System (ZFS) support, and a powerful Web GUI, TrueNAS™ pairs easy-to-manage FreeNAS™ software with world-class hardware and support for an unbeatable storage solution. In order to achieve maximum performance, the TrueNAS™ 2U System, equipped with the Intel® Xeon® Processor 5600 Series, supports Fusion-io's Flash Memory Cards and 10 GbE Network Cards. Titan TrueNAS™ 2U Appliances are an excellent storage solution for video streaming, file hosting, virtualization, and more. Paired with optional JBOD expansion units, the TrueNAS™ System offers excellent capacity at an affordable price.

For more information on the **TrueNAS™ 2U System**, or to request a quote, visit: **http://www.iXsystems.com/TrueNAS**.



KEY FEATURES:

- Supports One or Two Quad-Core or Six-Core, Intel® Xeon® Processor 5600 Series
- 12 Hot-Swap Drive Bays - Up to 36TB of Data Storage Capacity*
- Periodic Snapshots Feature Allows You to Restore Data from a Previously Generated Snapshot
- Remote Replication Allows You to Copy a Snapshot to an Offsite Server, for Maximum Data Security
- Up to 4.48TB of Fusion-io Flash Memory
- 2 x 1GbE Network interface (Onboard) + Up to 4 Additional 1GbE Ports or Single/Dual Port 10 GbE Network Cards

JBOD expansion is available on the 2U System

* 2.5" drive options available; please consult with your Account Manager

iX systems®

intel® Xeon® inside™

Powerful. Intelligent.

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.

SHAWN POWERS

To Hack, or Not To Hack

Of course, here at *Linux Journal*, we choose "hack" every time. My wife doesn't always agree with my world-modding attitude, however. She seems to think switches, and not cron jobs, should be used for turning lights on and off. She doesn't understand why a person would want to mount a broken laptop on the wall as a digital picture frame. And, of course, I'm not allowed to use the ultimate adhesive, duct tape, on anything our guests might see. Because she's also the reason I'm dressed, fed, shaved and showered most days, I'm really not complaining, it's probably for the best.

This issue is all about hacks. If you're smart and ambitious, you usually can make an awesome thing even better by hacking it a bit. Reuven M. Lerner starts off with a book review for 2011. Reading is one of the best ways to get better at hacking, whether it's hardware hacking or programming. Reuven reviews his favorite programming books and lets us know what he thinks. Dave Taylor follows him up with part II of his series on working with images within scripts. Hacking the command line is cool, but hacking graphics *with* the command line? Even cooler.

As one of my favorite television characters, Sheldon Cooper, says, "Everything is better with Bluetooth." Kyle Rankin and Michael Nugent show us some interesting Bluetooth hacks this month. Kyle addresses the problem of forgetting to lock your workstation when you walk away from it. In a home office, that's no big deal, but in cubicle-land, it can be an open invitation for pranks. Kyle shows how to lock your workstations automatically by setting up a dæmon to monitor when you (or more specifically, the Bluetooth-enabled phone in your pocket) wander away. Michael Nugent takes the opposite approach, and using Bluetooth proximity, he explains how to make your media PCs play music to certain speakers based on your location. It's a little like having the party follow

If you're smart and ambitious, you usually can make an awesome thing even better by hacking it a bit.

you wherever you go!

Daniel Bartholomew takes a hack at his power bill this month and writes about the Trim-Slice server. If you like the power savings a NAS device offers, but want the power of a full-blown Linux server, Daniel's article is right up your alley. He manages to run his new server with only 15% of the power his last one took. As someone with a server rack in my home office, I'm seriously considering scaling back the power.

No hacking issue would be complete without some talk about Arduino. Amit Saha teaches how to program the little buggers and bend them to your will. Arduino boards come with USB nowadays, which makes interfacing with them a snap. And, while we're talking about USB, it's the perfect time to mention Adrian Hannah's article on USB Flash drives. Adrian manages to hack multiple Linux distributions into a single USB drive, while providing persistent storage as well. Along the way, you'll become a GRUB expert to boot. He'll make you Lord of the keyRings, even if you're not a hobbit!

A few years back, the One Laptop Per Child Project took the world by storm with its aggressively priced laptops designed for kids in areas that are lacking in technology. The OLPC Project has taken its share of hits along the way, but it's still going strong. Sameer Verma provides an update on the OLPC initiative and describes the new hardware. Sugar is still alive and well on the OLPC computers, and, of course, so is Linux.

Don't think we just hacked this issue together though; we have the full lineup of Linux and open-source-related goodies you've come to love. Matt Davis describes how to create a vDSO (Virtual Dynamic Shared Object). John Knight demonstrates PdfMasher, a tool for converting PDFs into HTML, and James Coyner shows how to tweak the Android memory management system. Add to that countless other geeky goodies, and you have an issue that's bound to please. Oh, and if you want to use duct tape in your hacks? Don't worry, I 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 IRC channel on Freenode.net.

letters

jQuery | Gauger | Moose | Qt4 Designer | GNU Awk | jEdit
LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
PROGRAMMING
Make Utility Primer
What's New in GNU Awk 4.0
WaveMaker for Rapid Application Development
Modern Development with Perl and Moose
DEVELOP GUIs with Qt4 Designer and Eclipse
MULTIPLATFORM DEVELOPMENT Using GNU Libraries and Tools
USE GAUGER for Performance Regression Testing
PLUS: GETTING STARTED WITH jEDIT

I'll Be in My Hole

As a regular reader (and now subscriber) to *LJ*, I think Shawn Powers' Current_issue.tar.gz is always a great setup to the issue. His column in the August 2011 issue is a particular case in point. I read the opening paragraph to my wife. Her reply to Shawn's "basement-dwellers" comment was "or upstairs" (which happens to be the home of my toys).

I would like to thank you and all the contributors to your magazine. I came to Linux as an antique of 57 (now 63)—a retired "mainframe" developer unable to afford the constant upgrades to Windows development tools. Your magazine has helped me immeasurably get over the Linux learning curve as well as the fear factor. Though I don't consider myself agile in the environment, Windows is nowhere to be seen on my desktop, and tools like the Qt SDK have permitted me to continue to explore C++ and learn about genetic algorithms and programming.

One question: with so many more-experienced and knowledgeable Linux/open-source developers, how does someone with my limited experience contribute back to the community? I doubt there is much I can offer. Thank you again for *Linux Journal*.

—Pete R.

Thanks for your kind words! As hard as it might be to believe, I try to tone down my silliness in Current_Issue.tar.gz so as not to confuse non-native English speakers. In real life, I'm much weirder. (I'm sure the rest of the Linux Journal *staff would agree!)*

When it comes to contributing, the standard answer is "documentation". That said, I've never once been able to contribute documentation to a project, so perhaps the standard answer needs to be reconsidered. I think just using Linux, and enjoying it, and letting other people see you enjoy it will really do a lot for the community. And, if you're near a town with a Linux Users Group, try to attend meetings. They might have a need only you can fill!—Shawn

What Day Is It?

I know Dave Taylor must be really tired of everybody telling him how to better parse the output of `cal` to get the day of the week (see Dave Taylor's Work the Shell column in the June–September 2011 issues of *LJ*), but I figured I'd go a slightly different route. Many years ago, I was searching for an algorithm that would allow me to calculate the correct day of the week for a given date, and I found a great one. I no longer can remember where I found it, or who wrote it originally, but I have converted it into pretty much every language I use, and this last time was into a bash script:

```
#!/bin/bash

declare -a DAYS=(0 3 2 5 0 3 5 1 4 6 2 4)
declare -a DAYS_IN_WEEK=('Sunday' 'Monday' 'Tuesday'
 ➥'Wednesday' 'Thursday' 'Friday' 'Saturday')

get_dow() {
       # $1 is month, $2 is day, $3 is year
       # Strip off leading zeros from month and day:
       MONTH=${1#0}
       DAY=${2#0}
       YEAR=$3

       if [ $MONTH -lt 3 ]; then
               YEAR=$((YEAR-1))
       fi

       MONTH=$((MONTH-1))
       DOW=$(echo "($YEAR + ($YEAR/4) - ($YEAR/100)
       ➥+ ($YEAR/400) + ${DAYS[$MONTH]} + $DAY) % 7" | bc)
}

if [ "$1" == "" ]; then
       echo "dow.sh - Prints the day of the week,
       ➥for a given date"
       echo "  Where the date is in the format of mm/dd/yyyy
       ➥or mm-dd-yyyy"
       echo "usage:"
       echo "  dow.sh 01/01/1970"
       echo "or"
       echo "  dow.sh 01-01-1970"
       exit
fi

# Replace slashes or dashes on the input
# with spaces:
INTEXT=$(echo $1 | sed -e 's/[\/-]/ /g')
# Use the 'set' command to transfer to $1, $2, $3
set $INTEXT
# Call get_dow function with date to check:
get_dow "$1" "$2" "$3"
# Print day of week date falls on:
echo "${DAYS_IN_WEEK[$DOW]}"
```

Sorry, I don't have a ton of error-detection code in here, but I didn't want to overcomplicate it.

By the way, I am an avid reader of Dave's Work the Shell column, along with the rest of *Linux Journal*!

—Howard Pepper

Dave Taylor replies: *Thanks for this, Howard. A few people had told me that there is a formula for calculating the day of the week for any given date in the Julian calendar, but try as I might, I couldn't find one that seemed to be reliable with leap years and such. Now we have one, and it's a terrific shell script too.*

More on Calculating Day of the Week

This is regarding Dave Taylor's column in the August 2011 issue. Despite Dave's remark that his solution may be more complicated than necessary and that he'll stick to it, I'd like to say why not use the %w option of the date command? The following command does the job just fine:

```
date +%w -d <datespec>
```

—Mayuresh Warunjikar

Dave Taylor replies: *Well sure, if you have that version of the* date *command, but not every distro of Linux has all those advanced capabilities.*

Thanks for the New All-Digital Format

I just want to take a moment to congratulate and thank you on your transition to the new electronic format. The new format renders perfectly on my Sony PRS-600, and it has made the publication an absolute joy to read! I've been an avid reader for the past ten years.
—Joel Coulson

Thanks! I really enjoy reading on my PRS-300, the pocket-size version of your e-reader. My favorite part is how well the Sony device handles the table of contents.—Ed.

More Thanks for Going Digital-Only

Thanks for making your magazine digital only. I remember the demise of *TUX* magazine, which I really liked, but not many, including me, subscribed to it. And yes, I have subscribed to *LJ* for a year. I like the convenience of digital subscriptions and being able to click on articles and advertisers for more information.

I've already learned something new from your magazine today about JEdit (see Adrian Klaver's "JEdit: a Text Editor and More" in the September 2011 issue of *LJ*) and discovered it in my OpenSUSE repo. I wish you luck and success.
—Robert Smits

That's great to hear, Robert. I too was a TUX *subscriber long before I was on staff here at* Linux Journal. *I enjoyed it back then, and now with digital reading devices, I like the* LJ *electronic version even more!—Ed.*

Why Portrait Mode?

Like countless others (I'm sure), I was surprised by your recent "digital-only" announcement. Time will tell if this was a great or bad decision. I'm hoping it works out! Over the years, *LJ* has been a wonderful spokesperson and learning tool for the Linux community.

However, I don't understand why when a magazine goes digital, it still clings to the old print format. Yes, most magazines and books are printed in "portrait" mode. But, the majority of computers are stuck in "landscape". Formatting in portrait forces me to page up/down constantly, especially when you continue having multiple columns. Maybe it'll work when I get my 60" hi-res monitor? Best of luck with the new venture!
—Bob van der Poel

The layout part is a complicated endeavor.

Yes, landscape would make sense if it were viewed as a PDF on computer screens only. (If you remember TUX *magazine, that is exactly how it was done.) But,* Linux Journal *now can be read on e-readers, which are mainly in portrait mode. Then, there are the devices that happily switch from portrait to landscape, so we need to take that into account as well.*

We are trying to offer the same content in as many forms as possible. The PDF looks like a traditional magazine, the epub/mobi versions have flowing text and graphics specifically designed for e-readers, and our on-line interactive version was made for Web browsers. On top of that, we have the Android app, etc., etc. Ideally, with all those options, we can cover everyone's preferences for the most part.—Ed.

Why, Why All-Digital?

I'm kind of pissed off at you guys. An all-digital publication? No more driving home from the post office, glancing at the cover of the *LJ* on the seat beside me, anticipating a good read of *Linux Journal* next to the fire when I get home or over coffee and a croissant tomorrow morning? No more digging an old issue out of the stack to show somebody some awesome article on Java or Linux on Mars? I really hate reading anything of any length on my laptop; it's a major PITA. Now I have to think about getting some kind of e-reader, and I can't think of one I like. I actually write code to produce epub- and mobi-formatted e-books, and I can tell you there isn't an e-reader on the market today that's worth the box it comes in. The Nook comes closest, except it uses Adobe's e-reader SDK, which is almost as bad as the software in the Kindle. Man I am really pissed!

I know you guys are between a rock and a hard place. I work for a publisher, and I can see the writing on the wall, but "all-digital"? Jeesh.

—Scott Derrick

Scott, I know the switch is frustrating. Like you said, working for a publisher, I'm sure you weren't surprised, but that doesn't mean you have to be happy about it! I miss the old version as well, but I'm hoping as a tech magazine, we can push through this awkward time in publishing history and come out the other side stronger and wiser. Our readers are mostly geeks, and I think this transition is actually easier for us than it would be for a knitting magazine. If technology magazines can get over this hump, perhaps the non-tech magazines will be able to follow our blazed trail. Big plans, yes, but heck, we're the operating system that wants to take over the world!—Ed.

It Could Have Been Worse

I am perhaps an unusual subscriber. I am 79 years old and am not an administrator or a programmer. My wife and I have used Linux since Red Hat 5.2 (1999) and haven't used Windows for most of that time. I have about six years remaining on my subscription. I enjoy sitting in my easy chair in the evening and reading *Linux Journal*. I guess that is to be no more.

Still, it could have been worse. The magazine could have folded completely. At least I will have the digital copy; I just won't be able to sit in the recliner.

By the way, Dave Taylor and his Work the Shell column is my favorite.

—Bruce Bales

Bruce, hopefully e-reader prices will continue to fall, and you'll be able to read Linux Journal *from your recliner once again. Amazon seems to be doing a decent job with its $79 Kindle. If nothing else, you could move the easy chair over to the computer desk. I'm just teasing of course, but thank you for sticking with us.—Ed.*

Wow, Podcasts, Facebook and More?

I was extremely pleased to get the August 2011 issue of *LJ*. I recently released a social network site around podcasts (**http://weHEARus.com**), so I was excited to see Charles Olsen's article on podcasting as a way to share Linux knowledge. The Git article by Henry Van Styn was also a good read, since I currently am using SVN and have considered Git, but I was surprised you didn't really go into its "distributedness" aspect and how that works. Also, although I already have implemented Facebook's "like" button, the rest of Mike Diehl's "Facebook Application Development" article was very informative. A mobile Facebook app is something that is planned for the future. Do you have any plans to delve further into the Facebook API and publish articles on this? There was so much appropriately timed good stuff in the August 2011 issue. Thanks for yet another awesome one!

—Malcolm Beaulieu

Malcolm, I kind of feel like we should "high five" or something! I'm really glad you enjoyed the issue. Hopefully, this one is as useful for you too.—Ed.

Suggestions

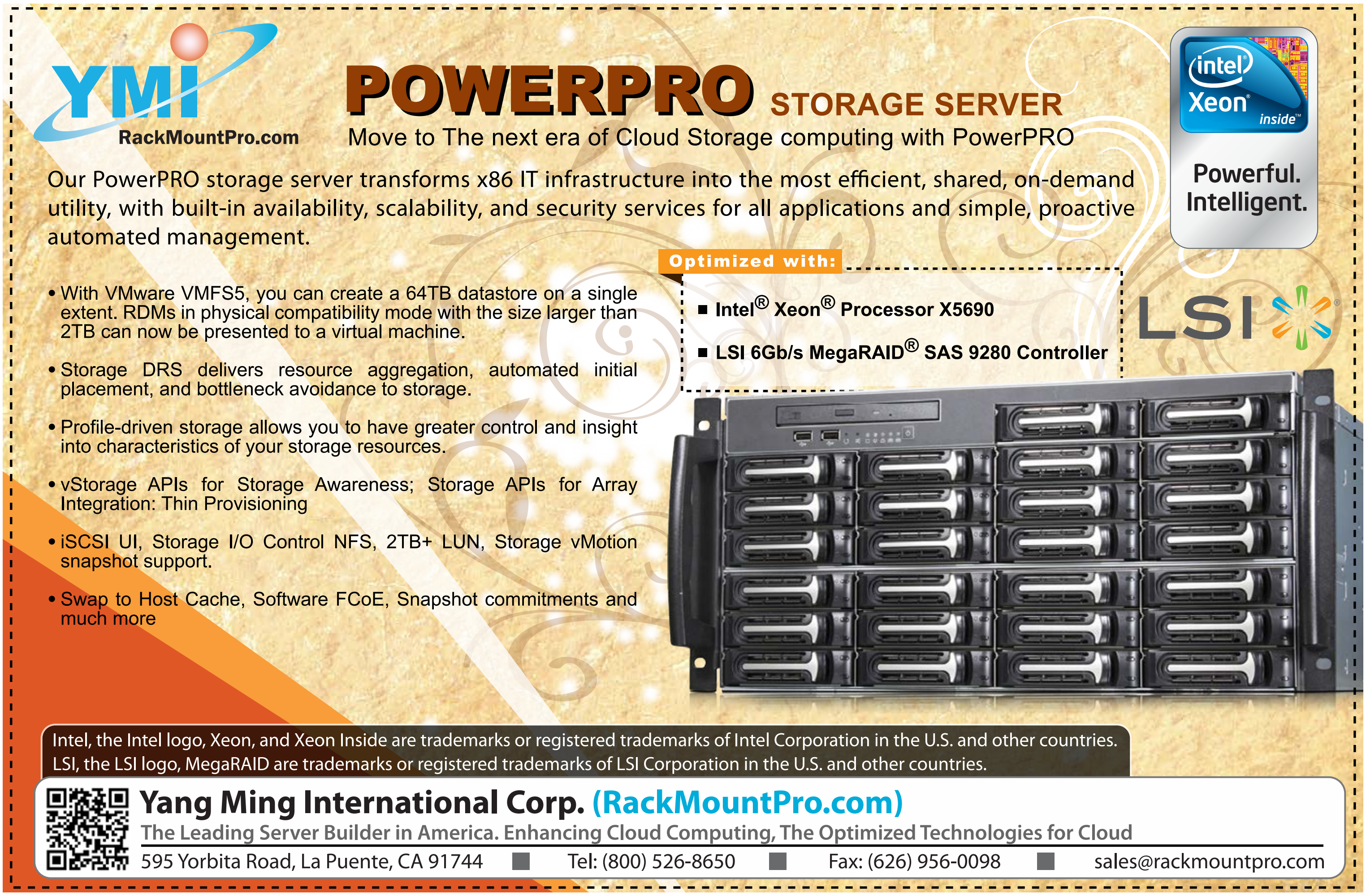
The diff -u (kernel development) column by Zack Brown is my favorite by far (my second favorite is Doc Searls' EOF). I like getting "the inside scoop" on some projects I don't follow closely. Is there any chance of having a full two-page spread with kernel development as well as some monthly rotating FOSS project development news besides the kernel? Even just a quick summary of "important" discussions on their respective mailing lists would be nice.

—Matthew Brush

Matthew, that's an interesting idea; we'll have to talk about it and see what we come up with. Thanks!—Ed.

Digital Television FCC Rules

Regarding R. E. Mackey's letter in the August 2011 issue of *Linux Journal* about digital TV and the FCC: although not really on the subject of Linux, many people do not realize that the FCC's regulations are



openly accessible to anyone, if you know where to look and what to look for.

You could say this is the broadcast equivalent of what "use the Source, Luke" would be, or the four-letter equivalent. The relevant rules are found in the Code of Federal Regulations Title 47 (47 CFR). For broadcast TV, the relevant part is 73 (47 CFR 73—broadcast as a whole is assigned parts 70–79). And, note that the CFR really is written in code. There are specific definitions of specific terms, and, to use an INTERCALism, COME-FROMs abound. But code it is. And, well, there are bugs in the code.

I mentioned "open"; see **http://www.gpo.gov/fdsys** to see how open. Browse into the Code of Federal Regulations, and drill down to, say, 47CFR73.616 to see how interference protection, post-DTV transition is worked out. (Anyone who can follow LKML should have no trouble finding it.)

The GPO is the Government Printing Office, and it is in charge of publication of rules and regulations. To see the law as passed by Congress, you want the United States Code. For the FCC, the relevant part is Title 47 USC. Well, that's an oversimplification. To use a revision control system's terminology, the USC is the trunk, and the Public Laws as passed by Congress are the commit logs (and the Federal Register is the equivalent to LKML), and all are freely available in electronic form, at the GPO.

I also meant "open" in another way. The Media Bureau, which is over broadcast stations, has quite a bit of open-source code out there, and it was a pioneer in government open source. Drill down into Bureaus from the main FCC page and find out for yourself.

Do note that the FCC.GOV site is huge. It is easy to get lost in there, much like github (and there is an FCC github).

—L. Owen

What to Do without Paper?

I understand that you have stopped (paper) publishing your magazine—yet another American (good) Linux magazine that is falling off circulation. It is too bad. Our society has found a more effective manner to publish. The reality is that you can have more up-to-date and timely content—that's true, but Shawn, an important question here: uh, uhm, what am I to read during any important "business" daily meeting? Well, I guess I'll have to take my laptop in with me from now on!

—Bob Wooden

Bob, you bring up the very reason I never borrow someone else's smartphone. And, now I know to never borrow your laptop.—Shawn

Rip-off?

Your simplistic conversion of print subscriptions to electronic subscriptions is a *rip-off*. I paid for a print subscription and have two years remaining.

Let's review: Paid. For. A. Print. Subscription. Get It?

Why? I can get a boat-load of on-line Linux news for *free*. I *paid* for being able to read it off-line. I won't be renewing.

—MJD

The switch was sudden and frustrating for everyone—we totally understand that. Unfortunately, we didn't have a choice. We are trying to offer as many digital options as we can (PDF, epub, mobi, interactive on-line, Android app and iPhone and iPad apps), but the print format just wasn't possible any longer. I am truly sorry you feel slighted. Hopefully, you'll find our digital offerings worthwhile at least during the remainder of your subscription.—Ed.

Dream Come True

Linux Journal as digital-edition-only? Oh no, I thought, when I was thinking about reading a PDF file on my Notebook or my Kindle during my train ride to/from work. What a surprise after I've realized that epub and mobi versions also are ready for download. Now I'm able to read my favourite Linux magazine comfortably on my Kindle. Thank you guys, a dream came true!

—Thomas Balzer

Thomas, I'm really excited about the epub/mobi versions as well. The PDF worked on my KindleDX, but on my smaller Kindle and my Sony PRS-300, it really wasn't easy to read. The epub/mobi version is very convenient for me, and I'm happy to see they are for you as well. Our Art Director, Garrick Antikajian, really gets the credit for the epub layout—three cheers for Garrick!—Ed.

Correction

Regarding my boto article "Python in the Clouds" in the October 2011 issue: Mitchell Garnaat, the author of boto, works for Eucalyptus Systems, not Canonical. I apologize for the error.

—Adrian Klaver

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

LINUX JOURNAL

At Your Service

SUBSCRIPTIONS: *Linux Journal* is available in a variety of digital formats, including PDF, .epub, .mobi and an on-line digital edition, as well as apps for iOS and Android devices. Renewing your subscription, changing your e-mail address for issue delivery, paying your invoice, viewing your account details or other subscription inquiries can be done instantly on-line: **http://www.linuxjournal.com/subs**. Alternatively, within the US and Canada, you may call us toll-free at 1-888-66-LINUX (54689) or internationally at +1-818-487-2089. E-mail us at subs@linuxjournal.com or reach us via postal mail at *Linux Journal*, PO Box 16476, North Hollywood, CA 91615-9911 USA. Please remember to include your complete name and address when contacting us.

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

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

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

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

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

diff -u
WHAT'S NEW IN KERNEL DEVELOPMENT

As **Linus Torvalds** released **Linux 3.0** at long last, there followed a cascade of minor breakages related solely to the change in version number. Some of these had to do with the kernel sources themselves; others had to do with user scripts out in the wild. The **git** tree had to be renamed to git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git, because the previous tree had "2.6" in the URL. Then the **kernel.org** page pointed to the wrong git tree, so that had to be sorted out.

The kernel build scripts also expected a three-numbered version number, with an optional fourth (for the stable series), while the 3.0 kernel will have only two, with an optional third. But, it turns out that there's no easy answer, because the kernel is so used to recognizing itself as a three-numbered entity, that it couldn't (for now) be made to identify as 3.0 internally.

A related problem is that some third-party software, including some that exists in binary form only, can't tolerate the version number change. As **Andi Kleen** reported, this means that Linux 3.0 is not fully backward-compatible with Linux 2.6, even though the only cause of that incompatibility is the version number. He posted a patch to cause the kernel to masquerade as a 2.6 kernel, with an additional string tacked onto the version number, indicating the proper 3.0 version. It's about as ugly a solution as you could imagine, but Andi said that there didn't seem to be any better workaround that would still allow the binary-only software to run under the new kernels.

The **OpenRISC architecture** will be included in the Linux kernel. OpenRISC is a really cool hardware project for creating an open-source CPU design. The project has gone through various vicissitudes, including being virtually dead for a while. But, now there are about 25 developers on the project, and they've had a kernel port in development since 2.6.35. **Jonas Bonn** recently submitted that code to Linus Torvalds for inclusion, and it was accepted readily enough, after a small moment of confusion over whether a related patch by **Rusty Russell** would be cleaner to accept before or after Jonas'.

Another architecture also is well on the way to being accepted into the kernel. The **C64x** from **Texas Instruments** recently got a GCC port, which means it's now possible to compile a full system for that chip. The GCC port is actually in a relatively early stage, but it's still promising enough that **Mark Salter** felt the time was right to post his patch introducing the C64x architecture into the kernel. He also set up a developer wiki at **linux-c6x.org**.

An interesting detail about the C64x processors is that they don't support **cache coherency**. In other words, multiple CPU cores are not guaranteed to see the same data when they check a given location in a shared data store. Without cache coherency, supporting **SMP** (Symmetric Multiprocessing) systems is much more difficult. For now, C64x systems may have to remain single-processor only.—ZACK BROWN

LinuxJournal.com

Since this issue is devoted to the fun and exciting world of hacking, consider this a friendly reminder to visit our security section at **http://www.linuxjournal.com/tag/security**.

While most hacking you'll read about here in *Linux Journal* is of the fun and harmless variety, hacking occasionally comes in forms that can become a nuisance or security risk to our dear readers. So, take some time to check out our offerings at LinuxJournal.com, because you never know what threats lurk and what awesome tips you might find for avoiding them. And, if that doesn't convince you, just please think of the children.

Give these a try to get started:

- "A Primer to the OAuth Protocol" by Adrian Hannah: **http://www.linuxjournal.com/article/10965**.
- "Live-Fire Security Testing with Armitage and Metasploit" by Raphael Mudge: **http://www.linuxjournal.com/article/10973**.
- "Linux Distro: Tails—You Can Never Be Too Paranoid" by Michael Reed: **http://www.linuxjournal.com/content/linux-distro-tales-you-can-never-be-too-paranoid**.

—KATHERINE DRUCKMAN

Stop Installing Outdated Ubuntu!

Have you ever installed Linux, only to be greeted immediately with the system update notification applet? If you have a fast Internet connection or, even better, a local Ubuntu mirror (see **http://www.linuxjournal.com/video/mirror-partner-repo-canonical** for some tips on creating your own local mirror), you can use the "Installation Minimal CD" and get a fully up-to-date system the first time it boots.

Another advantage to using the Installation Minimal CD is that it allows you to install any official Ubuntu variant with the same CD. Whether you want Ubuntu, Xubuntu, Kubuntu or even the more specialized *buntus for video editing and such, the Minimal CD can do it. Download a copy at **https://help.ubuntu.com/community/Installation/MinimalCD**. It's less than 20MB! —SHAWN POWERS

Flow, Flow, Flow Your Code

Our recent transition to a fully digital magazine has been exciting, stressful, tumultuous and, thanks to our readers, collaborative. As we were converting our traditional publishing skills into the digital arena, we tried hard to preserve the things that make magazines popular. We didn't want the magazine to become less; we wanted it to become more. Unfortunately, with all our efforts to maintain useful layouts and provide interactive reading apps, we did miss an important consumption model: the traditional e-reader.

We're so thankful for all the feedback we received, and continue to receive, regarding our digital transition. Thanks to that feedback, we now offer an epub edition of *Linux Journal*, complete with flowing text and functional navigation for the vast majority of E Ink devices. Yes, many devices can read PDF files, but they're just not pleasant to read on a small screen. We're excited about this issue, and we hope you are too!—SHAWN POWERS

They Said It

“An empty stomach is not a good political adviser.“—Albert Einstein

“A question that sometimes drives me hazy: am I or are the others crazy?“
—Albert Einstein

“A man should look for what is, and not for what he thinks should be.“
—Albert Einstein

“Any man who can drive safely while kissing a pretty girl is simply not giving the kiss the attention it deserves.“—Albert Einstein

“Common sense is the collection of prejudices acquired by age eighteen.“
—Albert Einstein

“Education is what remains after one has forgotten what one has learned in school.“—Albert Einstein

Non-Linux FOSS

Whether you’re trying to use Google Voice and a free SIP service to get a free telephone system or trying to connect to your company’s internal phone system, if you’re on a Mac, you need a SIP client. A few free options exist, but none are as simple as Alexei Kuznetsov’s Telephone.

Telephone even integrates with OS X’s Contacts program, so you can make a SIP call directly from there. It’s free, open source, and it’s even available in the Apple App Store. If you need to make a VoIP call on your Macintosh, be sure to check out Telephone: **http://www.tlphn.com**.

—**SHAWN POWERS**

Tweak Your Android Memory Management System

In my travels, I've discovered that not only does the Linux community have a robust following, but so does the Android hacking community. My journey began two years ago with the release of the Original Droid. I had come from a Windows Mobile world and didn't know that phone hacking even existed. Now hacking around on my Droid is almost an everyday task.

Most of you own Android phones now, and many of you already are running hacked versions of Android, and many of you already heavily tweak your builds. This article is not for you. It's an introduction to the world of Android customization, intended for Android users who don't yet know how to tweak their devices.

You've probably heard of Cyanogen and CVPCS if you are at all involved in the community. They are, in fact, a couple of the most influential developers of CyanogenMod, an open-source customized version of Android—a distribution of sorts. With this distro, you get the kernel for your device as well as the latest version of Android.

Before I begin, the most important requirement is that you are rooted. You can find many guides on-line to do this. Once you obtain root, let's begin with the tweaks.

Android comes with the Linux kernel, which, if you have not already inferred, means that many of the mainline kernel tweaks can be applied to the Android version of the kernel through the use of various configuration files. For this guide, you're going to edit build.prop, which is located under /system, and the local.prop, which is under /data.

On Linux, you can edit these files with vim or nano, but Android is slightly different. On Android, you need a file browser that is able to run and edit files as the superuser or root. There are many root-level browsers on the Android market, but I have been quite successful with an application called Root Explorer, so I use that for the remainder of this guide, but any root browser with text editor support will suffice.

Now that you have rooted your device and have the version of Android installed that you want to tweak, you can begin tweaking. The first thing I normally do is optimize my TCP stack for faster 3G download speeds. To do this, open your file browser of choice (I used Root Explorer), navigate to /system/build.prop or /data/local.prop, and open it in the text editor.

Every Android device has a build.prop file, but not all will have the local.prop file. If the local.prop file exists, it's generally safer to edit this file rather than the build.prop file. So edit whichever you feel safer editing, and copy the code below into your file of choice:

```
ro.ril.hsxpa=2
ro.ril.gprsclass=12
ro.ril.hep=1
ro.ril.enable.dtm=1
ro.ril.hsdpa.category=28
ro.ril.enable.a53=1
ro.ril.enable.a52=1
ro.ril.enable.3g.prefix=1
ro.ril.htcmaskw1.bitmask=4294967295
ro.ril.htcmaskw1=14449
ro.ril.hsupa.category=9
ro.ril.def.agps.mode=2
ro.ril.def.agps.feature=1
ro.ril.enable.sdr=1
ro.ril.enable.gea3=1
ro.ril.enable.fd.plmn.
prefix=23402,23410,23411
```

I generally tend to add my tweaks of preference to the local.prop file, but after you edit it, reboot your device and enjoy the faster download and browsing speeds. This method is generally the more difficult of the two methods that are available. The easiest way to apply tweaks is to do it with a script. To run a script on Android, you can, in fact, run your script of choice exactly the same as you do on your Linux desktop, through the terminal.

Two scripts that I now use to accomplish the above are the V8/V6 Supercharge and 3G Turbocharger scripts. The V8/V6 script modifies how Android manages the available system memory as well as various kernel tweaks to provide a smoother and more fluid Android experience. A simple Web search will help you find both of these scripts. You can run these scripts in either of two ways: through the terminal application or a script manager. To run the script from the terminal, `su` to root, then navigate to where you downloaded it to your SD card and run it. To run it from a script manager, again navigate to the directory where it is located, and run it as root. It will ask you which scrolling speed you prefer, so select one. Next, the V8/V6 script will present you with 14 different options from which to choose. I like to multitask without my applications forcibly closing, so I choose option 7, but there also are options for gaming, aggressive memory management and balanced management.

It's that simple. Happy tweaking!

—JAMES COYNER

Elmer for PDEs

Elmer is an open-source multiphysics simulation software that has been around since 1995. It is developed by the CSC — IT Center for Science Ltd. This group is administered by the Ministry of Education, Science and Culture in Finland. With Elmer, you can model physical systems, such as fluid dynamics, electromagnetism, heat transfer and acoustics. All of those problem areas have one thing in common—they are all described by partial differential equations (PDEs).

A PDE is a differential equation that depends on more than one variable, usually more than one space variable. For example, an equation that describes how heat gets transferred across a metal plate, where a is the thermal diffusivity of the metal plate, would look like the formula shown in Figure 1.

$$\frac{du}{dt} = a\left(\frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}\right)$$

Figure 1. Sample Equation

Very simple PDEs have exact solutions, but anything more complex that describes more physical situations just can't be solved exactly. This is where numerical solutions come into play. There are three widely used methods: finite element, finite volume and finite difference methods. Elmer uses the finite element method. The basic idea is to break up the problem space into some kind of a mesh of smaller spaces, and then either eliminate the PDE completely, by reworking it into a steady state problem on the mesh, or approximating the PDE with a system of ordinary differential equations that then are integrated using standard methods, like Euler's method or Runge-Kutta.

Elmer is distributed under the GPL, so you always can download the source and compile your very own version. Binaries are available for all those poor users still stuck on Windows. Packages should be available for most Linux distributions. For Ubuntu-based distros, you simply can use the following to install Elmer and all the necessary libraries:

```
sudo apt-get install elmer
```

Elmer is broken into several parts. ElmerSolver is the part that actually takes the input files and does the calculations to evolve the PDE you are interested in. ElmerPost is the visualization and post-processing tool. With this tool, you actually can see the results of your calculations. ElmerGrid is the meshing tool that can create simple 1-D, 2-D and 3-D meshes that will be used to evolve your PDE. It also can be used to import meshes generated using other software packages, such as those created using the Ansys tools.

There also is a GUI interface called ElmerGUI. ElmerGUI allows you to define and generate input files for ElmerSolver, as well as import external meshes from other software (Figure 2).

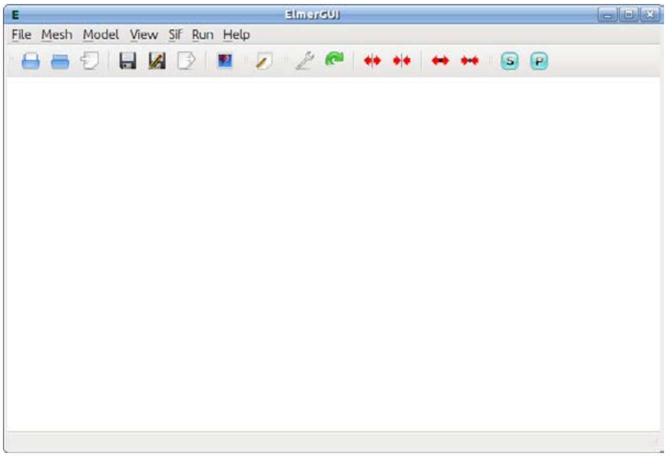
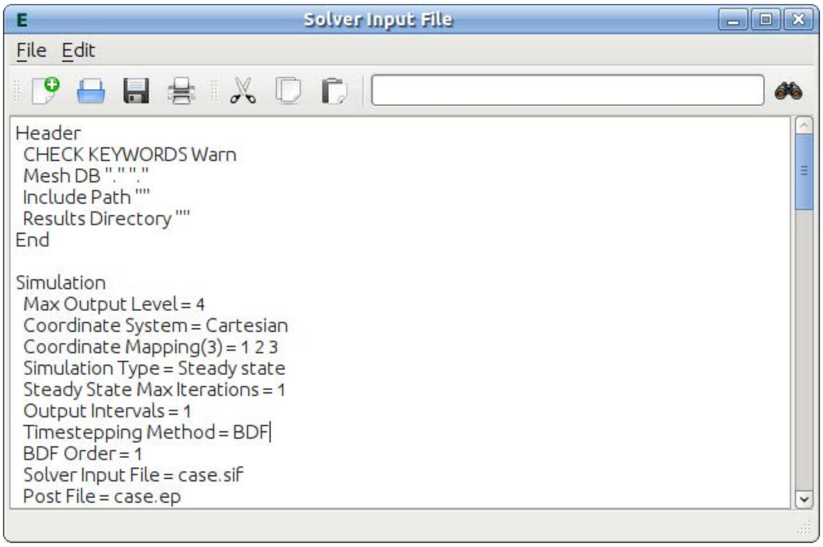


Figure 2. ElmerGUI

Looking at all of this, you may be thinking, "This is perfect!" So, I also should mention some of Elmer's downsides. Elmer actually is a suite of tools, and those tools are not always at the same development level. The documentation tends to lag behind the newest features, and ElmerGUI may be missing some of the more esoteric methods and models that ElmerSolver can use. Elmer is a huge system, and like any large software package, it has a steep learning curve. The tools available (ElmerGrid) can handle generating only relatively simple meshes. This means that if you have a very complicated system, you likely will need to use some other tool to generate the mesh and then import it into Elmer. If you can work around these possible deficiencies, Elmer may be exactly the right solution for you.

ElmerGUI is likely to be what most people prefer using. You can import element mesh files in various formats, generate element partitionings for geometry input files, set up the PDE systems you want to solve, and export model data to hand in to ElmerSolver. Most conveniently, ElmerGui also provides an interface to the parallel version of the solver, ElmerSolver.mpi. There also is a built-in post-processor that allows you to view your results right there from ElmerGUI. The menus in ElmerGUI are fully programmable as well, so you can tune the interface to match your specific problem area better.

A user forum, wiki and other resources are available at **http://www.elmerfem.org**, and a very good tutorial is available from the main Elmer Web site (**http://www.csc.fi/english/pages/elmer**). Along with the tutorial documentation, sample files are available for each of the cases. Those files can be great starting points for your own calculations and help you as you are learning the system. As an example, let's look at the sample "Computation of fringe capacitance". Once you have downloaded the sample files and unpacked them, you can load the project with File→Load Project... from the menu. Once it is loaded, all the input data also is loaded. A project is simply a directory containing all the files required to do a computation. The actual mesh is defined by the files mesh.header, mesh.nodes, mesh.elements



```
Header
  CHECK KEYWORDS Warn
  Mesh DB "." "."
  Include Path ""
  Results Directory ""
End

Simulation
  Max Output Level = 4
  Coordinate System = Cartesian
  Coordinate Mapping(3) = 1 2 3
  Simulation Type = Steady state
  Steady State Max Iterations = 1
  Output Intervals = 1
  Timestepping Method = BDF
  BDF Order = 1
  Solver Input File = case.sif
  Post File = case.ep
```

Figure 3. Solver Input File

and mesh.boundary. The current settings and state are stored in the file egproject.xml. There also is a solver input file called case.sif that is handed in to ElmerSolver when you actually are ready to do your calculations. You can edit this file via Sif→Edit... (Figure 3).

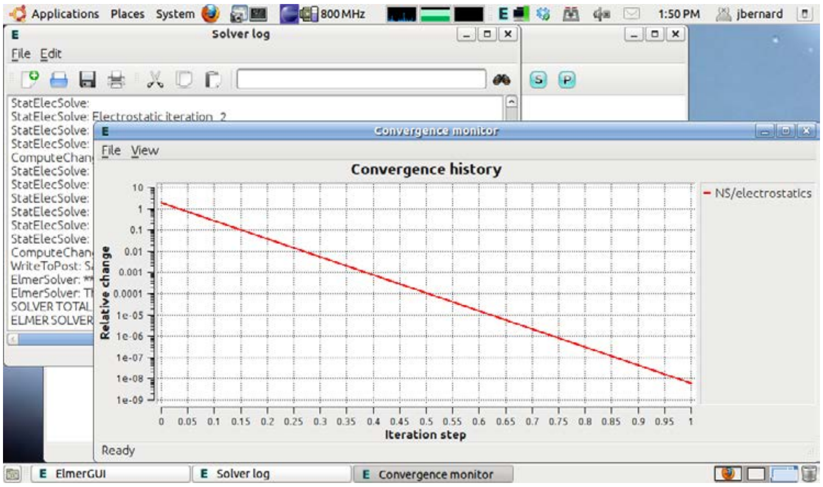
You can edit the model details (like physical constants) via Model→Setup.... When you are ready to run your calculation, you can use Run→Start Solver. This opens a log window and shows its progress. A convergence monitor shows how quickly Elmer converges on the results (Figure 4).

This creates a new file named case.ep in your project directory containing the results of your calculation. You can view it using either Run→Start Postprocessor or Run→Postprocessor (VTK)... (Figure 5).

As you can see, several tools are available to help you visualize the results of your calculation.

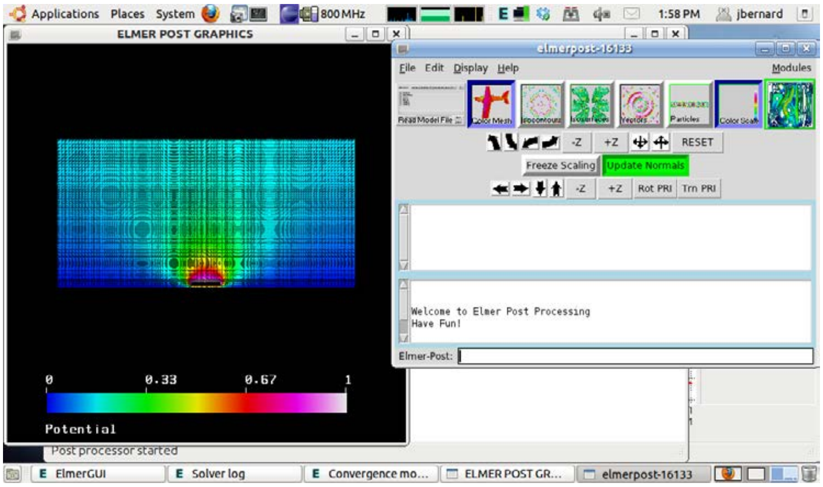
Now that you have seen a basic example of running one of the tutorials, what else can Elmer do for you? The solver can handle solving linear systems. It can do this by using direct methods, through the LAPACK library, for example. You can use a set of Krylov subspace methods to do iterative solutions. In order to get rapid convergence though, you usually need to use some form of preconditioning. A class of iterative methods called multilevel methods are used for large linear systems. Elmer provides two options: geometric multigrid and algebraic multigrid.

More complex, and hence more



```
StatElecSolve:
StatElecSolve: Electrostatic iteration 2
StatElecSolve:
StatElecSolve:
ComputeChan
StatElecSolve:
StatElecSolve:
StatElecSolve:
StatElecSolve:
StatElecSolve:
StatElecSolve:
ComputeChan
WriteToPost: S
ElmerSolver: **
ElmerSolver: T
SOLVER TOTAL
ELMER SOLVER
```

Figure 4. Convergence Monitor



```
Welcome to Elmer Post Processing
Have Fun!
```

```
Post processor started
```

Figure 5. Viewing the Results

physically accurate, problems tend to be nonlinear. This nonlinearity may be as simple as what you see in the full equation for pendulum motion to the Navier-Stokes equations for fluid flow to the equations of General Relativity. Elmer deals with nonlinear systems by first linearizing the equations at each iteration step. How the equations are linearized depends on exactly which solver method is being used. For example, the Navier-Stokes solver can use either the Picard linearization or the Newton linearization.

There are methods for solving time-dependent systems. First-order time derivatives can be discretized using either the Crank-Nicolson method or the Backward Differences Formulae. You also can solve eigenvalue problems with Elmer. These tend to crop up in structural analysis problems, including factors like elasticity and damping.

For really large problems, you likely will want to look into running your computation in parallel. Elmer uses MPI as the parallelization technique, along with domain decomposition as the method of dividing up the work. The first step is to take the mesh and subdivide it into chunks or partitions, which, depending on the actual calculation to perform, will divide the load evenly across all the CPUs. These chunks then are sent out to individual CPUs, and the calculation is done. At the end of the run, the results are combined back into a single result. Because of the work involved in partitioning and so forth, most users likely will take advantage of ElmerGUI's parallelization tool.

The last of Elmer's selling points is its modular nature. The solver is written in FORTRAN 90. This means if you want to add your own user functions or a complete solver, it is simply a matter of writing a FORTRAN module and including it in Elmer. The main Elmer site provides good documentation covering the steps involved.

Hopefully, this introduction has given you some ideas of what you can do with Elmer. If you are studying multiphysics problems, Elmer probably is a very good tool to learn. It also might be a good tool to introduce in a numerical physics course, because you can model so much. If you do end up using it in your research or studies, I would love to hear about it.

—JOEY BERNARD

SILICON MECHANICS

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

intel® Xeon® inside™

Powerful. Intelligent.

The Science of Ocean Row Solo

Wave Vidmar is committed to using his expeditions to gather data on the ocean, his boat, and the physical and psychological effects of the challenge. He will have a host of sensors with him on the row. A partial list of the instruments on board includes

G-Force sensors (2) mounted in the boat

Professional weather station

Body monitors to record about 20 different body functions, including activity, calories burned, resting time, sleep time, and temperature

Rowing performance monitor to track strokes taken, power per arm for each stroke, duration of stroke, depth of stroke, catch, drive, recovery, and speed

GPS devices for real-time tracking, and marine navigation software

Silicon Mechanics and Intel are proud to sponsor the Ocean Row Solo North Atlantic Challenge 2011 expedition. They partnered to donatethe 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 workstation.

Silicon Mechanics will be maintaining a dashboard, updated as information comes in, so that we can track his progress throughout the voyage.

Expedition Update

Wave Vidmar remains on the East Coast with his team. They report that they weathered the recent storms and are awaiting a few final pieces of equipment. They are working energetically to complete their launch preparations.

For more information about Wave Vidmar and Ocean Row Solo, visit www.oceanrowsolo.com.

To track the expedition, visit www.siliconmechanics.com/ors.

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

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.

Expert included.

REUVEN M. LERNER

2011 Book Roundup

Reuven's annual list of recommended books and podcasts for open-source developers.

Each year, I like to step back and write an article describing and recommending some of the computer-related books I have most enjoyed during the previous 12 months. I'm fortunate to come across a very wide variety of books, as well as journals and podcasts, and I enjoy sharing the best of what I've found. I believe that everyone in our industry should try to keep their skills up to speed not only by practicing software development, but also by learning more about it.

My rules for these reviews are pretty simple. If I discovered and read a book during the past 12 months or so, it's fair game for me to mention it. It's possible that some of the books I discuss here were written and published long ago, but if it's new to me, I include it. I don't promise to be comprehensive. I can judge only the books that came my way. And, I obviously have my favorite technologies, which tend to get more of my attention.

Ruby and Rails

Ruby continues to be my favorite programming language, one I use every day, and it pleasantly continues to surprise me. Much of my work in Ruby is as a Web developer, using the Ruby on Rails framework. I do share the concerns some have expressed—that Rails is becoming too complicated for newcomers, in part because of its adoption of many new, unusual or Rails-specific technologies in recent versions. However, I still think that if you're writing a Web application, you would be wise at least to consider Ruby on Rails.

I'm not the only person who feels this way, which is why a torrent of Ruby books continues to be published. One of the best I've seen in the past year is *Eloquent Ruby* by Russ Olsen (Addison-Wesley, ISBN 978-0321584106). This book is aimed at intermediate-to-advanced Ruby programmers who want to write code that's not only functional, but also takes advantage of the language's best characteristics. Some of the techniques are a bit advanced, but I'm all in favor of learning such techniques, even if I never use them, because they deepen my understanding of the language. This book has deservedly received a great deal of praise.

For people new to Ruby and who want to learn how to develop Web applications using Rails, Michael Hartl has produced a book, "Ruby on Rails 3 Tutorial" (Addison-Wesley, ISBN 978-0321743121) and an accompanying video. I must admit I haven't yet watched the video, but the book is a great introduction to the environment, spirit and techniques of Rails development. The book has the attitude that you can learn the Ruby language while you're also learning Rails, and although I think that a better foundation probably is appropriate before jumping into Web development, it's far from a fatal flaw. The book, which is aimed explicitly at beginners, is full of useful tidbits and hints on Ruby, includes hosting an application on Heroku, and also is about Rails and Web development in general.

One of the first books about Ruby to be published is *The Ruby Way* by Hal Fulton, and a second edition was published several years ago. In the wake of this success, Obie Fernandez published *The Rails Way*, an almost encyclopedic description of Ruby on Rails and the way it can (and should) be used. This past year, Fernandez published an updated version of this book, *The Rails 3 Way* (Addison-Wesley, ISBN 978-0321601667), and although there are many improvements and interesting tidbits in this book, it's no longer as extensive and encyclopedic as the previous edition. Now, that's not necessarily a bad thing, given that Fernandez is well known in the Rails community, and he's trying to show what technologies he uses, as opposed to the ones that others might use. That said, I learn something new every time I read through this book, typically saving me time in the future. So, I recommend that Rails developers read this book, but only after they've had sufficient grounding in Rails.

Two more books, both from the Pragmatic Programmers, are worth a look for Rails developers: *Continuous Testing with Ruby* by Ben Rady and Rod Coffin (ISBN 978-1934356708) gives a step-by-step introduction to the world of automated testing and how you can incorporate it into your Ruby development environment. Meanwhile, Jose Valim, author of the popular "Devise" gem for authentication in Rails, has written *Crafting Rails Applications* (ISBN 978-1934356739), which is less an introduction to Rails applications than a tour of interesting projects you can do with Rails, using advanced techniques that might well come in handy.

Finally, I read at least two nice short, independently published books during the past year. *Rails 3 Upgrade Handbook* by Jeremy McAnally (**http://www.railsupgradehandbook.com**) is a short but extremely useful guide to moving your Rails applications from version

It seems I'm not the only programmer whose feelings about JavaScript have changed dramatically during the years.

2 to version 3.

If you haven't yet made this change, McAnally walks you through many of the necessary steps to get it done. Separately, Avdi Grimm, an independent developer, blogger and podcaster, published a great little book about exceptions and how to use them called *Exceptional Ruby* (**http://exceptionalruby.com**). Unless you're thoroughly schooled in the way Ruby exceptions work, this is a great book to read through, understand and then put into practice in your own projects.

JavaScript

It seems I'm not the only programmer whose feelings about JavaScript have changed dramatically during the years. First, I dismissed it as a toy language that wasn't good for much. Next, I cursed it every time I had to deal with it, because of the strange syntax and conditions, and then I cursed the implementations of the language, which were slow, buggy and incompatible with one another. Then, almost overnight, JavaScript changed—or at least, my attitude toward it did. The differences between browsers were easily ignored, thanks to libraries like Prototype and jQuery. Bugs were fixed, and JavaScript engines were tuned and re-tuned, making it a very fast-executing language. And libraries grew over time, such that you now can do amazing things with JavaScript. I still find the language to be a bit verbose and annoying in some ways, but there's no doubt that JavaScript development is an integral part of any modern Web developer's toolbox, and understanding the way it works is essential.

Two great books that can help push you forward are *Eloquent JavaScript* by Marijn Haverbeke (ISBN 978-1593272821, and at **http://eloquentjavascript.net**), which introduces programming at a basic level, but does so by showing many powerful and interesting techniques JavaScript has to offer. Even if you're a seasoned JavaScript hacker, it's worth looking at this book. A nice thing about the on-line edition, available for free at the URL listed above, is that you can run the code right in your browser. I've also seen many mentions of the *JavaScript Cookbook* by Shelley Powers (O'Reilly, ISBN 978-0596806132). It doesn't aim to be a JavaScript tutorial, but rather a collection of short techniques, ideas and articles describing common tasks that JavaScript coders might need. If you're using

JavaScript—and nearly all Web developers are doing so nowadays—you probably will get a lot out of this book.

Now that JavaScript executes so quickly and is universally available in browsers, we've seen the beginning of two new trends: Web frameworks that operate within the browser, using only JavaScript, and languages that compile into JavaScript, rather than into byte code or binary executables. For now, the only reference to Backbone.js that I have seen is a screencast from Peepcode (**https://peepcode.com/products/backbone-js**), which I enjoyed and from which I learned quite a bit. Meanwhile, if you're interested in trying CoffeeScript—a language that compiles into JavaScript, which is mandatory in Rails 3.1, and which has taken the JavaScript world by storm—you probably should read Trevor Burnham's *CoffeeScript: Accelerated JavaScript Development* (Pragmatic Programmers, ISBN 978-1934356784), which introduces the language and describes how you can use it.

Other Technologies

Although my main language is Ruby, I still like and appreciate Python quite a bit, and I still get to use it in some of my projects. Mark Lutz, who has written many editions of *Learning Python* and *Programming Python* for O'Reilly has come out with his latest book, *Programming Python* 4th edition (ISBN 978-0-596-15810-1). If you're using Python, you probably want to have this book on your shelf, so that you can refer to it on a regular basis. More-advanced Python programmers would do well to read *Pro Python* (Apress, ISBN 978-1430227571), which describes the behind-the-scenes implementation of many Python idioms, such as metaclasses and common protocols, which really can come in handy.

HTML and CSS form the foundation of the Web, and HTML5 includes so many new features that learning them can take some time. From the Pragmatic Programmers comes *HTML5 and CSS3* by Brian Hogan. This book is written well, providing (I think) a gentler introduction to the subject than many other books. Another great introduction, but one that focuses on other aspects of HTML5, is Mark Pilgrim's on-line *Dive into HTML5* (**http://diveintohtml5.org**), also available as a print book, *HTML5: Up and Running* (O'Reilly, 978-0596806026).

Regular readers of my articles know that I have long used PostgreSQL for all of my relational database needs. Although I believe that PostgreSQL is a better fit for most projects than MySQL and offers a powerful host of features that is a serious match for open-source and commercial databases alike, it's a sad fact that there aren't very many books about PostgreSQL on the market. Fortunately, Packt Press has released two very high-quality books in the past year: *PostgreSQL 9 Administration Cookbook* by Simon Riggs and Hannu

Krosing (ISBN 978-1849510288) and *PostgreSQL 9.0 High Performance* by Gregory Smith (ISBN 978-1849510301). I personally refer to these books all the time when working on clients' projects, and I always learn something from them.

Other Languages

Lisp is one of those languages that people either swoon over and dream about or despise with a passion. I'm strongly in the first category, and, thus, I continue to enjoy the fact that Lisp books still are published. A hilarious and excellent introduction to Lisp, *Land of Lisp*, teaches Lisp programming and concepts by writing games. I really enjoyed reading this book, and I think it's worth the time even if you don't write programs or plan to develop in Lisp.

I'm still working my way through it, but I want to mention *Learn You a Haskell for Great Good!* (No Starch Press, ISBN 978-1593272838), which is both funny and interesting, and it finally has managed to keep my attention for a while on learning Haskell. I'm not sure if I'll ever use Haskell in a real project, but I feel like at least I'm in a better position to understand its strengths and weaknesses.

R is a statistics language that I've been using for many years, and it seems to be getting more popular over time. O'Reilly has published the *R Cookbook* (ISBN 978-0596809157), which, in contrast to the many free books available on-line, takes a very practical approach to doing things in R, whose data types and functions are a bit different from other languages I've used.

Podcasts

I've become quite a fan of podcasts during the past few years, in part because I tend to sit on the train a great deal, as well as walk to clients' offices from the nearest train stop, which isn't always nearby! Fortunately, many high-quality, interesting podcasts exist—far more than I have time to listen to. I'm listing here those that have to do with open-source and Web development; obviously many others are available on-line on a variety of other interesting topics.

Two of my staples are done by the pair of Jason Seifer and Peter Cooper: "The Ruby Show" (**http://rubyshow.com**) and its younger sibling "The JavaScript Show" (**http://javascriptshow.com**) are half-hour, weekly news updates on the state of each of these languages and the development frameworks built on them. I imagine their humor is not for everyone, but I find Jason and Peter to be entertaining as well as informative, and regret when they (or I) miss a week.

Another good open-source podcast is the "Changelog" (**http://thechangelog.com**), a weekly hour-long interview podcast hosted by Adam Stacoviak and Wynn Netherland.

Finally, one of the most thought-provoking and interesting podcasts I've started to listen to is "Ruby Rogues" hosted by Charles

Max Wood (**http://rubyrogues.com**) with a panel of distinguished Rubyists every week. Now, it's true that the panel comes from the Ruby world, and the discussion centers around the Ruby language, but the panelists are serious and clever software engineers who take pride in their craft and always are learning how to do things better, which means the discussion is deeper than most other blogs and podcasts I follow. If you're interested in hearing how truly great software engineers think, this is a great weekly one-hour dose.

Non-Computer Books

It's true that I tend to read a great many computer-related books during the year, but I'm happy to report that I read about other subjects as well. Two of the most interesting books I've read in the past year are *Country Driving* by Peter Hessler (ISBN 978-0061804090), the former China correspondent for the *New Yorker*, who provides an interesting and nuanced picture of modern China.

Another book about that area of the world, which was haunting and disturbing more than anything else, was *Nothing to Envy* by Barbara Demick (ISBN 978-0385523912), in which she describes what life is like in North Korea. The stories she tells in this book, based on interviews with North Koreans who escaped, are beautifully told but horrifying to read all the same.

I also read two books by Gail Collins, the *New York Times* columnist, about the history of women in America: *When Everything Changed* (covering the 1960s through the present) and *America's Women* (ISBN 978-0060185107), describing the history of the United States from colonial times through the 1960s, but from a woman's perspective.

Finally, I'm currently in the middle of *How Rome Fell* by Adrian Goldsworthy (ISBN 978-0300137194). My high-school history classes didn't go into much detail on most subjects, and I'm learning through this book just how much I don't know about an empire that ruled the part of the world in which I currently live, and whose influence still can be felt today by a very large number of people. Although the book is about how the Roman empire fell, I'm using it as a way to better understand how it worked when it did, and I'm enjoying having that opportunity.

I'm sure I could recommend many other books, but these were the first that came to mind. My next article will return to our regularly scheduled program, looking more at open-source Web technologies that can help us put the ideas from these books into practice for ourselves and our clients.■

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.

DAVE TAYLOR

Working with Image Files, Part II

Adding caption generation to the image-inclusion script.

If you're a faithful reader of my *LJ* column with a good memory, you'll have realized this is the second time I've written a set of columns talking about my image-scaling script. My last article presented a major revision to the script that added the capability of scaling the image dimensions in the resultant HTML code and also warning if it produces a significantly smaller image than the existing file specifies.

One thing I pay close attention to with my Web sites is their search-engine friendliness. After all, why put all the effort into producing good content and then omit the last step or two that can help maximize search-engine results placement (SERPs, in the biz)?

It turns out that if your page loads slowly, you're less likely to nab a really great spot in the search results than if it's lightning fast. So, if you are loading a 73KB image and scaling it down 33%, for example, it'd be faster to scale the image file itself (even if you end up with multiple versions of the image at different sizes) and have the 39KB version or similar.

In this article, I want to turn our attention to something else, generating attractive captions. There are two ways that captions are specified from the command line: `-c` tells the script to use the filename as the caption, substituting spaces for dash (-) or underscore (_) characters, and `-C xx` tells the script to use the user-specified value `xx` for the caption.

The latter is more accurate, but more work, so I typically use `-c`, particularly if I'm generating the image-inclusion HTML for a group or collection of images en masse. In that case, the filename is typically something like "mac-pages-hyphenation-control-1.png".

Converting Filenames to Image Captions

The easy way to create a caption is to axe the filename suffix and replace all dashes or underscores with spaces. For the filename mentioned above, that'd give us "mac pages hyphenation control 1", which isn't too bad. However, it would be better to fix the capitalization so the caption looks more like proper English.

The problem is, case-transliteration

utilities in Linux are designed to be all-or-nothing enterprises, so translating "pages" into "PAGES" is easy, but producing "Pages" is a bit trickier.

To do that, the script breaks the all-lowercase caption into separate words, then breaks each word into its first letter and subsequent letters:

```
firstletter=$(echo $word | cut -c1 |
   tr '[[:lower:]]' '[[:upper:]]')
otherletters=$(echo $word | cut -c2-)
newcaption="$newcaption$firstletter$otherletters "
```

Wrapped in the following for loop:

```
for word in $nicecaption
```

Ideally though, the sentence cap function should be smart enough to know that certain words shouldn't be capitalized, like "the", "of" and "or". That I solve as generically as possible:

```
if [ $wordcount -gt 0 ] ; then
  case $word in
   the|and|or|a|an|of|in) newcaption="$newcaption$word ";
        Continue;  ;;
  esac
Fi
```

Do you know why I check the word count in the resultant properly capitalized caption? Because if it's the first word in the caption, it should be capitalized. For example, "The Black and the Blue" is correct, not "the Black and the Blue".

One problem needs to be fixed due to how I reassemble the sentence in the script: the removal of the final trailing space. There are a bunch of ways to do that, but I really like using `rev` twice and cutting off the very first character:

```
rev | cut -c1- | rev
```

The entire function is neatly wrapped in a shell function:

```
FixCaption()
{
  newcaption="" ; wordcount=0
  for word in $nicecaption
  do
    if [ $wordcount -gt 0 ] ; then
      case $word in
       the|and|or|a|an|of|in) newcaption="$newcaption$word ";
          continue; ;;
      esac
    fi
    firstletter=$(echo $word | cut -c1 | tr '[[:lower:]]'
        '[[:upper:]]')
    otherletters=$(echo $word | cut -c2-)
    newcaption="$newcaption$firstletter$otherletters "
    wordcount=$(( $wordcount + 1 ))
  done
  nicecaption=$(echo $newcaption | rev | cut -c1- | rev)
}
```

It's complicated, but hopefully,

still understandable!

Producing Good HTML Code

Are you wondering what we've created? Here's how this now gives us a nice, readable caption based on the well-named file:

```
$ scale -c 1 facebook-upload-photo-computer-1.png
<center><img
src="http://www.askdavetaylor.com/pics/
➥facebook-upload-photo-computer-1.png"
alt="facebook upload photo computer 1" border="0"
 ➥width="604" height="204"/><div style="font-size:
➥80%;color:#777;">Facebook Upload Photo Computer
 ➥1</div></center>
```

I'll unwrap that so you can see the HTML with less headache:

```
<center>

<img
src="http://www.askdavetaylor.com/pics/
➥facebook-upload-photo-computer-1.png"
alt="facebook upload photo computer 1" border="0"
 ➥width="604" height="204"
/>

<div style="font-size:80%;color:#777;">

Facebook Upload Photo Computer 1

</div>

</center>
```

If you know HTML, you might be tempted to have some better code, where the entire image + caption are in a single div container. For that matter, there's a "caption" attribute to the "img" tag in modern HTML, but I don't use it because I like to have more control over how the actual text is positioned and rendered on the page—old-school, I guess.

I'm going to stop hacking the script here because it's already almost 200 lines, and I have to say that if a script is getting to be more than 100 lines or so, it might be time to consider moving the functionality into a Perl script or another programming language like C or C++. Not always, but shell scripts are really good until a certain point, then you're just wrestling with limitations rather than expanding your capabilities.

That's it for this month. Do you have a scripting challenge you'd like some help with or just an idea for a fun or interesting project we can tackle here in *Linux Journal*? If so, get off your duff and send me a message about it!■

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 http://www.DaveTaylorOnline.com.

The Intel® Xeon® processor 5600 series increases performance, cuts operating costs, and helps deliver ROI. That's intelligent performance.[1] Check out the intelligent features of the Xeon 5600 at intel.com/itcenter.

intel Xeon inside™

Powerful.
Intelligent.

Performance:

Your processor gets the job done.

Intelligent performance:

Your processor gets the job done with 40% more performance by adapting to your workload.[2]

Servers from Advanced Clustering Technologies feature the Intel® Xeon® processor 5600 series.

advanced clustering
technologies, inc.

advancedclustering.com/lj/1BX5502/
Call toll free: 866.802.8222

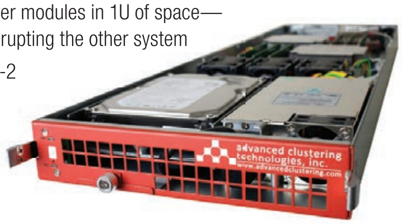
The Pinnacle 1BX5502 blade server delivers full-size performance in half the space. Our 1U chassis-based solution holds two complete blade servers to give you the high performance and density you need. Contact our experts to find your perfect Pinnacle server.

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

Pinnacle 1BX5502 1U Dual System Blade Server

- Intel® Xeon® processor 5600 series
- High performance, energy-efficient design for HPC clusters
- Two complete independent server modules in 1U of space—easy module access without disrupting the other system
- Available with built-in ConnectX-2 QDR InfiniBand

Starting at **$1,700**


advanced clustering technologies, inc.
www.advancedclustering.com

[1] Increased performance tested when comparing to the previous generations of Intel® Xeon® processors. Performance tests measure approximate performance of Intel® products on specific computer systems; any difference in hardware, software, or configuration may affect actual performance. For more information, visit www.intel.com/performance/server.

[2] When compared to the previous generations with servers based on Intel® 32nm microarchitecture. Based on results on a server side Java* benchmark in conjunction with power consumption across a load line. Intel internal measurement (Jan. 15, 2010). Configuration details: server side Java benchmark in conjunction with power consumption across a load line.

KYLE RANKIN

Automatically Lock Your Computer

Your phone can do much more than make calls and install apps that make obnoxious noises—learn how you can use it to lock your screen automatically.

If you've ever worked with pranksters, you've probably come across this classic office prank. First, the unsuspecting victim leaves his computer and goes to lunch or a long meeting and doesn't lock his screen. The prankster then takes a screenshot of his current desktop, hides all the desktop icons and any taskbars, and sets the background to be the screenshot the prankster just took. When the victim gets back to his computer, none of the icons work, and no programs will open, yet when he inevitably reboots, even that doesn't fix it. Around the time he calls the desktop support team, the prank is revealed and everyone has a good laugh about it—except maybe the victim, who vows from that day on always to lock his screen.

Good Locks Make Good Neighbors

I don't know about you, but even though I've never been the victim of a prank like that, I've always tried to be good about locking my screen when I'm away. I imagine some of this just stems from trying to be thoughtful about security, but also, if you are a sysadmin, you often have access to a lot of sensitive systems and data. Although you'd like to extend trust to your coworkers, an unlocked sysadmin computer would be a great source for sensitive information if you were a disgruntled employee. In this article, I discuss a simple program I've found that works great to lock your screen automatically when you walk away. At the end of the article, I also discuss some ideas for how to hack in to a system protected by this software.

Traditionally on any desktop environment I've used, I would configure a Ctrl-Alt L keybinding that locked my screen. It was so committed to muscle memory, that the moment I was about to stand up, my fingers automatically would lock my desktop without thinking about it. Although this method works great, I've recently discovered just how easy it is to set up your desktop to lock automatically when you leave, and even better, to unlock automatically when you return.

The solution to proximity-based desktop locking has been around for a while. I remember *Linux Journal*'s own Bill Childers demonstrating such a thing to me with Bluetooth on a Mac more than six years ago, but I've never felt like messing around with the Bluez tools on Linux to write my own script. It turns out these days there's a simple program you can use to lock your screen via Bluetooth that takes only a minute to set up: BlueProximity.

How BlueProximity Works

Essentially, BlueProximity works on the notion that most of us carry around a Bluetooth device with us—our cell phone. Because Bluetooth works only over a limited range, with a bit of tuning, you can make some assumptions about how close a device is to your computer based on the strength of its signal. BlueProximity works via these assumptions to decide when to lock or unlock your screen. When you leave your computer, the Bluetooth signal gets weaker until it crosses a threshold, and your desktop is locked. When you return to your computer, the signal gets stronger until it crosses a different threshold, and the computer automatically unlocks.

Install BlueProximity

You install BlueProximity much like any other Linux program: with your package manager. In my case, the package simply was called blueproximity. Alternatively, if your distribution doesn't package it, you can download and build the software from the source available at the BlueProximity SourceForge page. Once installed, you either can type `blueproximity` in a terminal to launch the program or click on it in your applications menu (on my system, it showed up under the Accessories category).

Once you start the program, you will get a new icon in your panel and also see a window much like the one shown in Figure 1. BlueProximity requires that the device you set up already has been paired with your computer, so if you haven't yet paired your

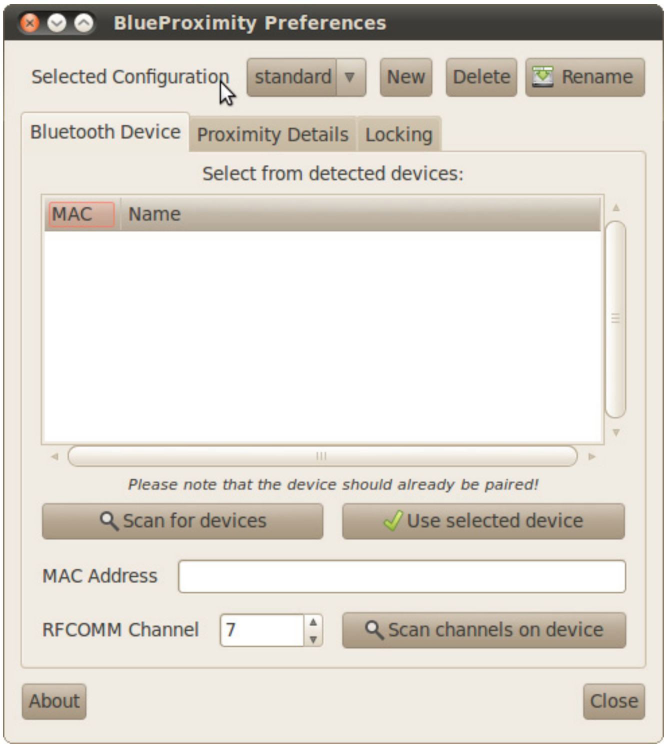


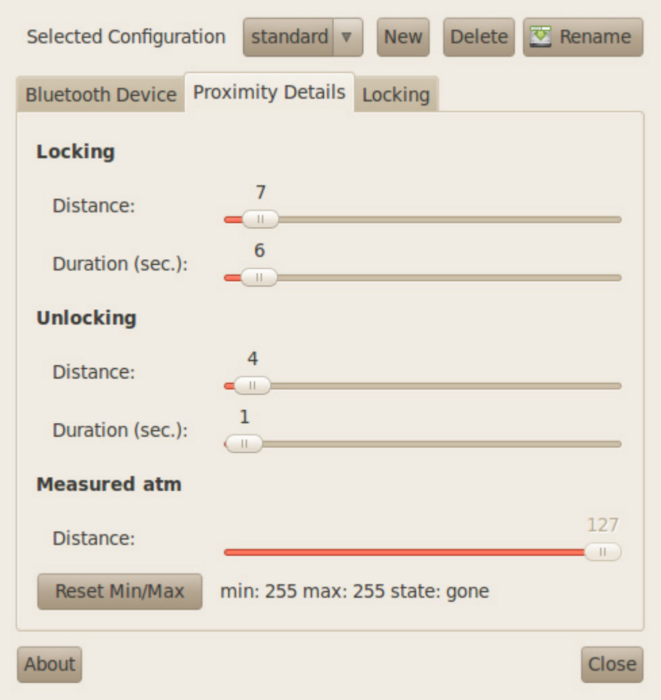
Figure 1. The Default BlueProximity Tab

phone (or other Bluetooth device you wish to use), you need to go through the steps to pair your Bluetooth device first. In my case (on an Ubuntu system), I first made sure Bluetooth was enabled and visible on my phone, next I clicked System→Preferences→Bluetooth, then clicked Set up new device, and after that I just followed through the wizard that appeared.

After you pair the Bluetooth device, go back to the main BlueProximity screen and click on Scan for devices. It may take a bit for the scan to complete, but afterward, you should see your Bluetooth device on the list. At that point, just select it, click Use selected device, and your device will be configured and ready to use. You can safely click the close button at the bottom of the window, as BlueProximity will still stay in your panel.

Fine-Tune Your Settings

Out of the box, BlueProximity tries to use reasonable settings to determine when to lock your screen; however, each Bluetooth device is different, and I've personally seen that different Bluetooth devices can have much different antenna strengths. If you notice that your screen doesn't lock as soon as you'd like, or worse, if it locks while you are still at your desk, click on the Proximity Details tab where you can edit thresholds (Figure 2). In this window, you will find a number of different sliders. The Distance slider for both Locking and Unlocking lets you control how close you need to be to the computer. The Measured atm slider updates every second or so and shows the current distance BlueProximity reads, so you can get a sense of how sensitive your equipment is. The Duration sliders let you control how long the device needs to be at a certain distance before it locks or unlocks the screen. This is useful in particular when tuning the Locking section so that one bad reading doesn't lock your screen when you don't want it to.



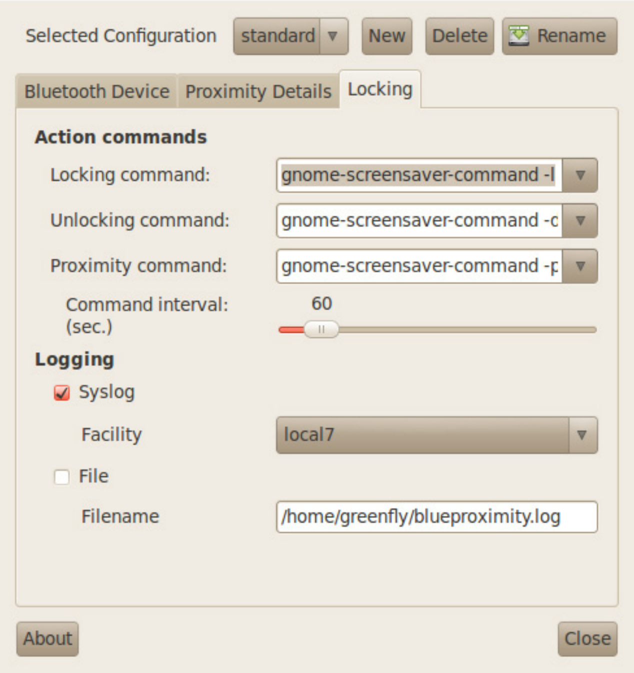
Locking
Distance: 7
Duration (sec.): 6
Unlocking
Distance: 4
Duration (sec.): 1
Measured atm
Distance: 127
min: 255 max: 255 state: gone

Figure 2. The Proximity Details Tab

If you do notice that the screen locks on you often while you are at your desk, increase the Distance slider for locking. Alternatively, if you notice when you come back to your desk you have to sit there for some time before it unlocks, you may want to increase the Distance slider for Unlocking and decrease the Duration slider if it's higher than one. Just make sure that your unlocking distance isn't higher than your locking distance.

Use Your Own Locking Program

By default, BlueProximity is configured to trigger gnome-screensaver commands. If you use GNOME, that might be fine, but if you use another desktop environment, click the Locking tab (Figure 3) where you can change what command gets run to lock and unlock the screen. By default, you can choose between gnome-screensaver and xscreensaver from the drop-down list, but you could really put any script you want in here. For instance, you might want to create a locking bash script that not only calls gnome-screensaver but also uses a DBUS signal to pause your music. Then you could write a



Action commands

Locking command: gnome-screensaver-command -l

Unlocking command: gnome-screensaver-command -c

Proximity command: gnome-screensaver-command -p

Command interval: (sec.) 60

Logging

☑ Syslog

Facility: local7

□ File

Filename: /home/greenfly/blueproximity.log

Figure 3. The Locking Tab

It is this convenience that may possibly open you up to an attack.

similar script to unpause your music and unlock the screen when you return.

Hack BlueProximity

Security is often a trade-off for convenience. BlueProximity is interesting in that it tries to increase your security (by automatically locking your screen), and it also tries to increase your convenience (by automatically unlocking your screen when you return). It is this convenience that may possibly open you up to an attack. That said, BlueProximity and the Bluetooth protocol do make attacking this method rather difficult, because BlueProximity allows only one device to unlock it at a time. You may think you could hack this setup simply by cloning the MAC address of a person's phone on your Bluetooth device. Although this is possible, unfortunately, during the pairing process, keys are shared between the two devices that they will use for secure communication later on, so even if you can clone a phone's MAC address, it still will not necessarily accept your device, because you don't have the shared secret. That having been said, Andrew Y. Lindell released a paper for Blackhat 2008 demonstrating that if you could sniff the pairing procedure, the password potentially could be leaked to you. Even if you weren't there when the devices were paired, the paper demonstrated how you might be able to get a device to re-run the pairing procedure.

Of course, such an attack is quite sophisticated and might be difficult to pull off. An easier approach simply would be to borrow the person's phone to make a phone call or check out an app when he isn't at his desk and then sneak over there. Along with that attack, BlueProximity naturally would be vulnerable to knocking the person out and stealing the phone, but then we are talking about office security here, and if you have to worry about coworkers beating you over the head and robbing you, maybe you should look into another line of work.■

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.

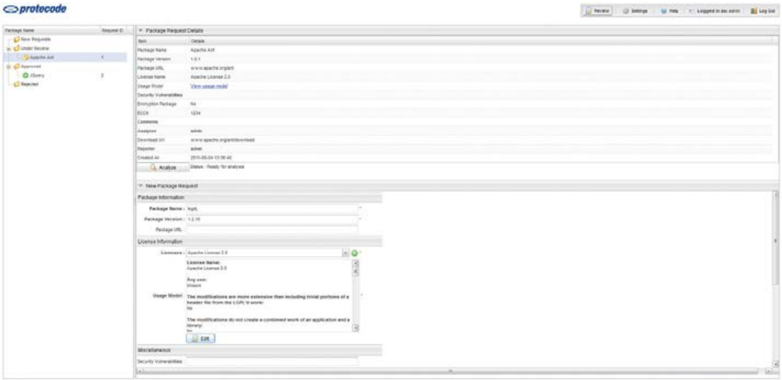
Resources

The BlueProximity Project Page: **http://sourceforge.net/projects/blueproximity**

Blackhat 2008 paper on Bluetooth Hacking by Andrew Y. Lindell: **http://www.blackhat.com/presentations/bh-usa-08/Lindell/BH_US_08_Lindell_Bluetooth_2.1_New_Vulnerabilities.pdf**

jQuery | Gauger | Moose | Qt4 Designer | GNU Awk | jEdit
LINUX JOURNAL
Since 1994: The Original Magazine of the Linux Community
SEPTEMBER 2011 | ISSUE 209 | www.linuxjournal.com
PROGRAMMING
Make Utility Primer
What's New in GNU Awk 4.0
WaveMaker for Rapid Application Development
DEVELOP GUIs with Qt4 Designer and Eclipse
MULTIPLATFORM DEVELOPMENT Using GNU Libraries and Tools
USE GAUGER for Performance Regression Testing
PLUS:
GETTING STARTED WITH jEDIT
FREE for Subscribers

Protecode System 4



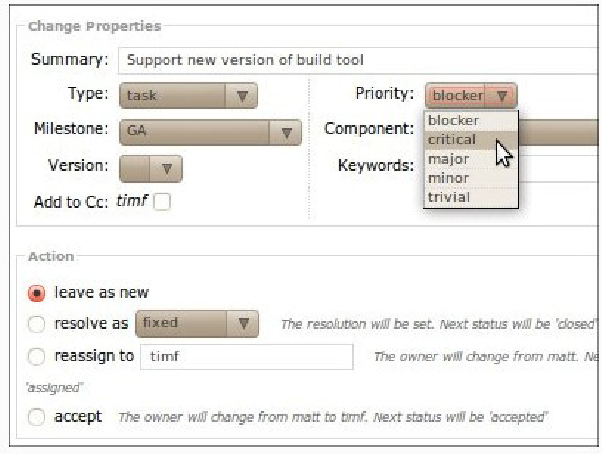
The most noteworthy enhancement in the upgraded Protecode System 4, Protecode Inc.'s open-source license management solution, is full support for the Software Package Data Exchange (SPDX) standard. Sponsored by the Linux Foundation, SPDX is an effort to create a standard format for communicating the components, licenses and copyrights associated with a software package. Protecode says that System 4's ability to read and generate SPDX information eases license information exchange across the software supply chain and allows for a simpler license compliance process. System 4 now also includes Code Administrator, an application that facilitates requesting, analysis and approval of open-source and other third-party software within an organization. It ensures that only a well-understood, preapproved set of software packages of specific versions with defined use cases are deployed.

http://www.protecode.com

Nimbula Director

Nimbula has unveiled the new version 1.5 of Nimbula Director, which the company says is the first cloud operating system that's capable of supporting a geographically distributed cloud. Nimbula Director helps enterprises and service providers build powerful private, hybrid and public cloud infrastructures. Nimbula Director abstracts the underlying technology to present a coherent view of a completely automated compute and storage cloud. This one-stop virtual data-center management solution isolates customers from the operational and hardware complexity associated with deploying a private or public cloud. The software can manage many geographic locations of a multi-site cloud from a single view, which allows end users to access any resource worldwide and deploy their workloads to any site in a self-service manner with a single login.

http://www.nimbula.com



Change Properties

Summary: Support new version of build tool

Type: task ▼

Milestone: GA ▼

Version: ▼

Add to Cc: *timf* □

Priority: blocker ▼

Component:

Keywords:

Action

● leave as new

○ resolve as fixed ▼ *The resolution will be set. Next status will be 'closed'*

○ reassign to timf *The owner will change from matt. Ne*

'assigned'

○ accept *The owner will change from matt to timf. Next status will be 'accepted'*

Akiri Solutions' DevBox-EZ

Cheap is good, but free is better, and the latter is what you'll pay for Akiri Solutions' DevBox-EZ, a new virtual appliance that offers all the tools necessary for implementing software development projects as well as simplifying and accelerating the migration to Git. The free version is modeled on Akiri's popular DevBox solution. DevBox-EZ, says its developer, "is up and running within just a few minutes and provides shared Git repositories, bug tracking, automated backup, file sharing and project-based built-in wikis". Integrated bug and issue tracking eliminates the need for the hassle of a separate tool. The automated backup can be scheduled at any time with a click of the mouse, and project-based built-in wikis that help ensure important project knowledge can be captured easily. The file-sharing capability simplifies the painstaking process of adding or deleting users and managing permissions versus using FTP sites or shared file servers.

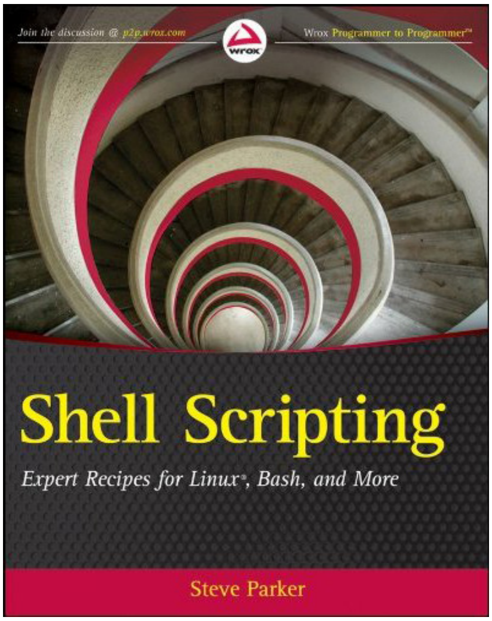
http://www.akirisolutions.com

Samba

Messrs. Tridgell, Allison and the rest of the dedicated Samba team have been forging ahead with a slew of new features and functionality, culminating in a recent major new version 3.6 release. Samba, of course, is the famed Windows interoperability suite of programs for Linux and UNIX. Most notable in Samba 3.6 is the first free software implementation of Microsoft's new SMB2 file-serving protocol. The brand-new asynchronous server architecture allows Samba's new SMB2 server to double the performance of some network applications when run with Microsoft Windows 7 clients. In addition, the print subsystem has been rewritten completely to use automatically generated remote procedure calls and provides greater compatibility with the Windows SPOOLSS print subsystem architecture. Other features include improved clustered file server support, simplified identity mapping and greater reliability, among others.

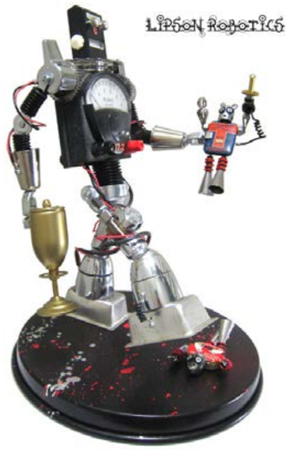
http://www.samba.org

Steve Parker's *Shell Scripting: Expert Recipes for Linux, Bash, and More* (Wrox)


Join the discussion @ p2p.wrox.com
Wrox Programmer to Programmer™
Shell Scripting
Expert Recipes for Linux®, Bash, and More
Steve Parker

A Linux geek's reference shelf without a book on shell scripting is painfully lonely. If your books are experiencing this loneliness, consider picking up Steve Parker's new book *Shell Scripting: Expert Recipes for Linux, Bash, and More*. This Wrox title is billed as "a compendium of shell scripting recipes that can immediately be used, adjusted and applied". With this book, Linux expert Steve Parker shares a collection of shell scripting recipes that can be used "plug and play" or easily modified for a variety of environments or situations to avoid re-inventing the wheel. The book covers shell programming, with a focus on Linux and the Bash shell; it provides credible, real-world relevance, as well as including the flexible tools to get started immediately.

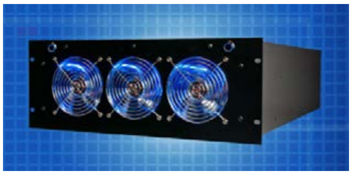
http://www.wrox.com



Lipson Robotics' Robot Sculptures

All right geeks, let's drop everything and talk Christmas presents! Have you dropped any gift hints yet? Good, because good things (sometimes) come to those who procrastinate. We suggest requesting the gift of art, specifically art that a computer nerd can love and appreciate. We recently stumbled upon the whimsical robot sculptures from New York City artist David Lipson. Lipson assembles these one-of-a-kind robots from random new and old objects. All are inspired by Lipson's love of tin toy robots, sci-fi movies and retro design. Each artwork is sold exclusively at Animazing Gallery in New York City, as well as via the gallery's Web site.

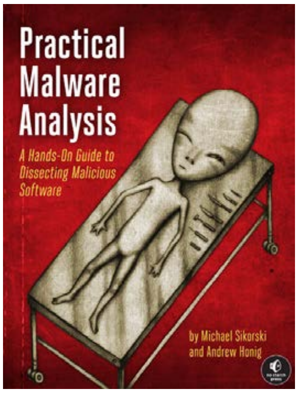
http://www.lipsonrobotics.com and **http://www.animazing.com**



BlackSky Computing's Apollo Enterprise HPC Storage System

It's easy being green with BlackSky Computing's Apollo Enterprise HPC Storage System, a massively scalable storage architecture that packs 180TB in a 4U chassis while maintaining a light footprint. Apollo can be expanded to support a single filesystem of more than 2 Petabytes "with no degradation in performance", says BlackSky, and it can work with HPC data sets up to 80Gb/s, making the system reportedly "four times faster the I/O and one-fourth the price of the leading competitors". BlackSky built Apollo because all others failed its own internal needs, which include the following requirements: tremendous capacity at low-price/TB, proportional network bandwidth in and out of the device that will cripple HPC applications if not done correctly and expandability under one filesystem to very large volumes. Other features include full redundancy and a Web-based UI.
http://www.blackskycomputing.com

Michael Sikorski and Andrew Honig's *Practical Malware Analysis* (No Starch)


Practical
Malware
Analysis
A Hands-On Guide to Dissecting Malicious Software
by Michael Sikorski and Andrew Honig

Get smart and ward off malware by thoroughly understanding it and those who make it by reading *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. The new book, authored by Michael Sikorski and Andrew Honig and published by No Starch Press, provides a rapid introduction to the tools and methods used to dissect malicious software, showing how to analyze, debug and disassemble these threats safely. The book goes on to examine how to overcome the evasive techniques—stealth, code obfuscation, anti-debugging, encryption, packing and others—that malware authors employ to thwart analysis attempts. Key chapters include a lab at the end, which reviews important concepts from the discussion in the context of real-world malware examples. Both beginners and veterans alike will find the book useful.
http://www.nostarch.com

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

Fresh from the Labs

PdfMasher—E-Book Conversion

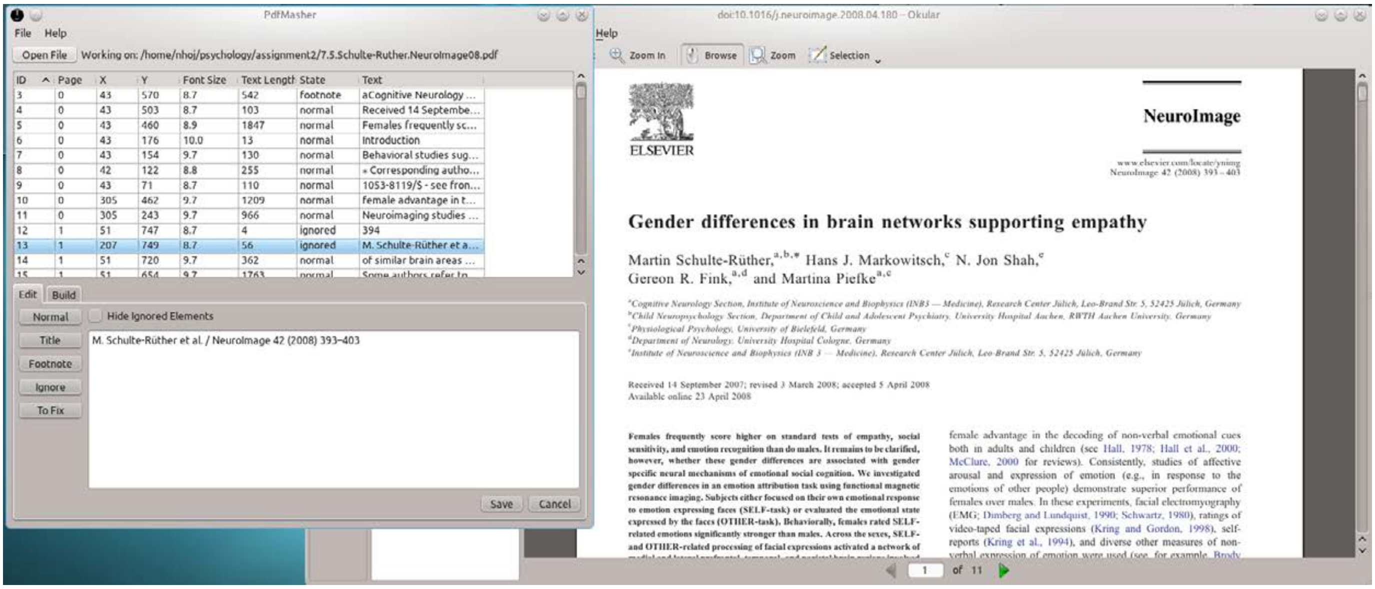
http://www.hardcoded.net/pdfmasher

If you've had problems reading PDF files on various devices (like mobile phones), PdfMasher may be just what you're looking for. According to the Web site:

> PdfMasher is a tool to convert PDF files containing text in ready-for-e-book HTML files. Most e-book readers support PDF files natively, but it's often a real pain to read those documents, because we don't have font-size control over the document like we have with native e-books. In many cases, we have to use the zooming feature, and it's just a pain. Another drawback of PDFs on e-book readers is that annotations are not supported.
>
> There are already tools to convert PDFs to e-books, like Calibre, but what they do is try to guess the role of each piece of text in the PDF (and that's if you're lucky). I think that in all but the simplest cases, it's a mistake to think that anything short of an AI can do that kind of guessing.

Installation If you can install this with a binary, by all means do so. Available on the site are 32- and 64-bit Linux .deb packages for the ubiquitous Intel x86 architecture. For masochists, or those who don't have an Intel-based CPU, there is the obligatory source.

In order to grab the latest source, first you need to install hg, which was under the package name "mercurial" on my Kubuntu system. Once that's installed, grab the



Working on: /home/nhoj/psychology/assignment2/7.5.Schulte-Ruther.NeuroImage08.pdf

ID	Page	X	Y	Font Size	Text Length	State	Text
3	0	43	570	8.7	542	footnote	aCognitive Neurology ...
4	0	43	503	8.7	103	normal	Received 14 Septembe...
5	0	43	460	8.9	1847	normal	Females frequently sc...
6	0	43	176	10.0	13	normal	Introduction
7	0	43	154	9.7	130	normal	Behavioral studies sug...
8	0	42	122	8.8	255	normal	∗ Corresponding autho...
9	0	43	71	8.7	110	normal	1053-8119/$ - see fron...
10	0	305	462	9.7	1209	normal	female advantage in t...
11	0	305	243	9.7	966	normal	Neuroimaging studies ...
12	1	51	747	8.7	4	ignored	394
13	1	207	749	8.7	56	ignored	M. Schulte-Rüther et a...
14	1	51	720	9.7	362	normal	of similar brain areas ...
15	1	51	654	9.7	1763	normal	Some authors refer to

☐ Hide Ignored Elements

M. Schulte-Rüther et al. / NeuroImage 42 (2008) 393–403

ELSEVIER

NeuroImage

www.elsevier.com/locate/ynimg
NeuroImage 42 (2008) 393–403

Gender differences in brain networks supporting empathy

Martin Schulte-Rüther,[a,b,*] Hans J. Markowitsch,[c] N. Jon Shah,[e]
Gereon R. Fink,[a,d] and Martina Piefke[a,c]

[a] Cognitive Neurology Section, Institute of Neuroscience and Biophysics (INB3 — Medicine), Research Center Jülich, Leo-Brand Str. 5, 52425 Jülich, Germany
[b] Child Neuropsychology Section, Department of Child and Adolescent Psychiatry, University Hospital Aachen, RWTH Aachen University, Germany
[c] Physiological Psychology, University of Bielefeld, Germany
[d] Department of Neurology, University Hospital Cologne, Germany
[e] Institute of Neuroscience and Biophysics (INB 3 — Medicine), Research Center Jülich, Leo-Brand Str. 5, 52425 Jülich, Germany

Received 14 September 2007; revised 3 March 2008; accepted 5 April 2008
Available online 23 April 2008

Females frequently score higher on standard tests of empathy, social sensitivity, and emotion recognition than do males. It remains to be clarified, however, whether these gender differences are associated with gender specific neural mechanisms of emotional social cognition. We investigated gender differences in an emotion attribution task using functional magnetic resonance imaging. Subjects either focused on their own emotional response to emotion expressing faces (SELF-task) or evaluated the emotional state expressed by the faces (OTHER-task). Behaviorally, females rated SELF-related emotions significantly stronger than males. Across the sexes, SELF- and OTHER-related processing of facial expressions activated a network of

female advantage in the decoding of non-verbal emotional cues both in adults and children (see Hall, 1978; Hall et al., 2000; McClure, 2000 for reviews). Consistently, studies of affective arousal and expression of emotion (e.g., in response to the emotions of other people) demonstrate superior performance of females over males. In these experiments, facial electromyography (EMG; Dimberg and Lundquist, 1990; Schwartz, 1980), ratings of video-taped facial expressions (Kring and Gordon, 1998), self-reports (Kring et al., 1994), and diverse other measures of non-verbal expression of emotion were used (see, for example, Brody

Using PdfMasher, PDF files like these can be manipulated manually for conversion into other formats.

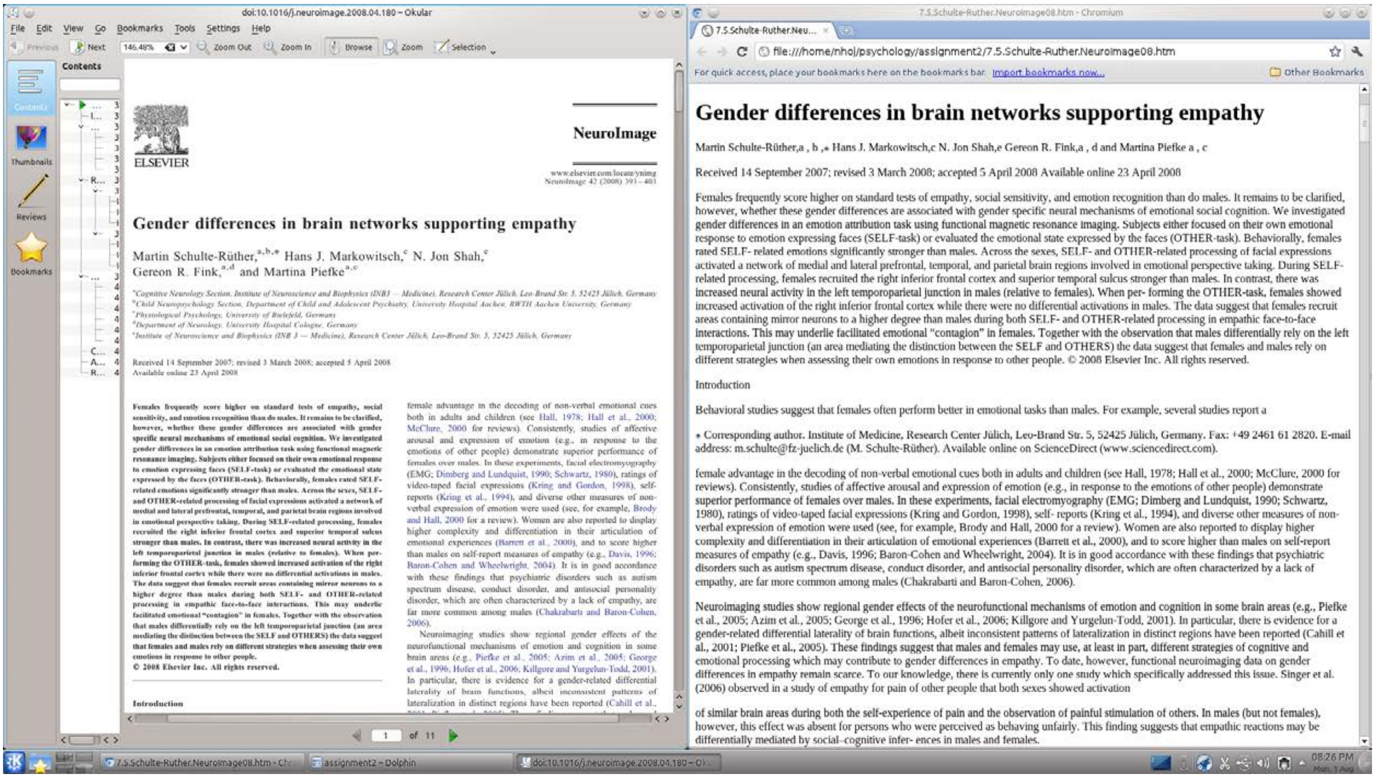
latest source by entering the command:

```
$ hg clone https://bitbucket.org/hsoft/pdfmasher
```

Once that has finished downloading, keep this terminal open where it is, because next you'll need to sort out the library requirements, and then you'll return to this terminal and continue the installation. As far as dependencies are concerned, the documentation lists the following:

- Python 3.2 (**http://www.python.org**)
- pdfminer3k (**http://hg.hardcoded.net/pdfminer3k**)
- jobprogress 1.0.0 (**http://hg.hardcoded.net/jobprogress**)
- Sphinx 1.0.7 (**http://sphinx.pocoo.org**)
- pytest 2.0.3 to run unit tests (**http://pytest.org**)
- Markdown 2.0.3 (**http://www.freewisdom.org/projects/python-markdown**)
- PyQt 4.7.5 (**http://www.riverbankcomputing.co.uk/news**)

With the dependencies out of the way,


doi:10.1016/j.neuroimage.2008.04.180 – Okular
File Edit View Go Bookmarks Tools Settings Help
Previous Next 146.48% Zoom Out Zoom In Browse Zoom Selection
Contents
Thumbnails
Reviews
Bookmarks
1 of 11
7.5.Schulte-Ruther.NeuroImage08.htm - Chromium
7.5.Schulte-Ruther.Neu...
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
Other Bookmarks
7.5.Schulte-Ruther.NeuroImage08.htm - Chr...
assignment2 – Dolphin
doi:10.1016/j.neuroimage.2008.04.180 – Ok...
08:26 PM

file:///home/nhoj/psychology/assignment2/7.5.Schulte-Ruther.NeuroImage08.htm

ELSEVIER

NeuroImage

www.elsevier.com/locate/ynimg
NeuroImage 42 (2008) 393–401

Gender differences in brain networks supporting empathy

Martin Schulte-Rüther,[a,b,*] Hans J. Markowitsch,[c] N. Jon Shah,[e] Gereon R. Fink,[a,d] and Martina Piefke[a,c]

[a] Cognitive Neurology Section, Institute of Neuroscience and Biophysics (INB3 — Medicine), Research Center Jülich, Leo-Brand Str. 5, 52425 Jülich, Germany
[b] Child Neuropsychology Section, Department of Child and Adolescent Psychiatry, University Hospital Aachen, RWTH Aachen University, Germany
[c] Physiological Psychology, University of Bielefeld, Germany
[d] Department of Neurology, University Hospital Cologne, Germany
[e] Institute of Neuroscience and Biophysics (INB 3 — Medicine), Research Center Jülich, Leo-Brand Str. 5, 52425 Jülich, Germany

Received 14 September 2007; revised 3 March 2008; accepted 5 April 2008
Available online 23 April 2008

Gender differences in brain networks supporting empathy

Martin Schulte-Rüther,a , b ,* Hans J. Markowitsch,c N. Jon Shah,e Gereon R. Fink,a , d and Martina Piefke a , c

Received 14 September 2007; revised 3 March 2008; accepted 5 April 2008 Available online 23 April 2008

Females frequently score higher on standard tests of empathy, social sensitivity, and emotion recognition than do males. It remains to be clarified, however, whether these gender differences are associated with gender specific neural mechanisms of emotional social cognition. We investigated gender differences in an emotion attribution task using functional magnetic resonance imaging. Subjects either focused on their own emotional response to emotion expressing faces (SELF-task) or evaluated the emotional state expressed by the faces (OTHER-task). Behaviorally, females rated SELF- related emotions significantly stronger than males. Across the sexes, SELF- and OTHER-related processing of facial expressions activated a network of medial and lateral prefrontal, temporal, and parietal brain regions involved in emotional perspective taking. During SELF-related processing, females recruited the right inferior frontal cortex and superior temporal sulcus stronger than males. In contrast, there was increased neural activity in the left temporoparietal junction in males (relative to females). When per- forming the OTHER-task, females showed increased activation of the right inferior frontal cortex while there were no differential activations in males. The data suggest that females recruit areas containing mirror neurons to a higher degree than males during both SELF- and OTHER-related processing in empathic face-to-face interactions. This may underlie facilitated emotional "contagion" in females. Together with the observation that males differentially rely on the left temporoparietal junction (an area mediating the distinction between the SELF and OTHERS) the data suggest that females and males rely on different strategies when assessing their own emotions in response to other people. © 2008 Elsevier Inc. All rights reserved.

Introduction

Behavioral studies suggest that females often perform better in emotional tasks than males. For example, several studies report a

* Corresponding author. Institute of Medicine, Research Center Jülich, Leo-Brand Str. 5, 52425 Jülich, Germany. Fax: +49 2461 61 2820. E-mail address: m.schulte@fz-juelich.de (M. Schulte-Rüther). Available online on ScienceDirect (www.sciencedirect.com).

female advantage in the decoding of non-verbal emotional cues both in adults and children (see Hall, 1978; Hall et al., 2000; McClure, 2000 for reviews). Consistently, studies of affective arousal and expression of emotion (e.g., in response to the emotions of other people) demonstrate superior performance of females over males. In these experiments, facial electromyography (EMG; Dimberg and Lundquist, 1990; Schwartz, 1980), ratings of video-taped facial expressions (Kring and Gordon, 1998), self- reports (Kring et al., 1994), and diverse other measures of non-verbal expression of emotion were used (see, for example, Brody and Hall, 2000 for a review). Women are also reported to display higher complexity and differentiation in their articulation of emotional experiences (Barrett et al., 2000), and to score higher than males on self-report measures of empathy (e.g., Davis, 1996; Baron-Cohen and Wheelwright, 2004). It is in good accordance with these findings that psychiatric disorders such as autism spectrum disease, conduct disorder, and antisocial personality disorder, which are often characterized by a lack of empathy, are far more common among males (Chakrabarti and Baron-Cohen, 2006).

Neuroimaging studies show regional gender effects of the neurofunctional mechanisms of emotion and cognition in some brain areas (e.g., Piefke et al., 2005; Azim et al., 2005; George et al., 1996; Hofer et al., 2006; Killgore and Yurgelun-Todd, 2001). In particular, there is evidence for a gender-related differential laterality of brain functions, albeit inconsistent patterns of lateralization in distinct regions have been reported (Cahill et al., 2001; Piefke et al., 2005). These findings suggest that males and females may use, at least in part, different strategies of cognitive and emotional processing which may contribute to gender differences in empathy. To date, however, functional neuroimaging data on gender differences in empathy remain scarce. To our knowledge, there is currently only one study which specifically addressed this issue. Singer et al. (2006) observed in a study of empathy for pain of other people that both sexes showed activation

of similar brain areas during both the self-experience of pain and the observation of painful stimulation of others. In males (but not females), however, this effect was absent for persons who were perceived as behaving unfairly. This finding suggests that empathic reactions may be differentially mediated by social–cognitive infer- ences in males and females.

With the original PDF on the left and outputted HTML on the right, this e-book now can be read on any device without readability woes.

re-open the terminal from before and enter the following commands:

```
$ cd pdfmasher
$ python configure.py
$ python build.py
```

Then, run the program with:

```
$ python run.py
```

If you're lucky enough to have the binary installed, you simply can run the program with the command:

```
$ pdfmasher
```

Usage Before I try to explain how to use PdfMasher myself, I should include the following from the Web site:

> PdfMasher asks the user about the role of each piece of text, and does it in an efficient manner. Your PDF has a header on each page, and you don't want them to litter your text? Sort text elements by Y-position (thus grouping them all together); Shift-select the elements and flag them as ignored. They will not appear on your final HTML. Your PDF has footnotes on many pages? Sort your elements by text content (thus grouping all elements with the text starting with a number together) and flag them as footnotes. They will be moved to the end of the document, and PdfMasher will try to create hyperlinks to footnote references.

Before changing things under PdfMasher, I recommend having your PDF open to one side in another program so you can cross-check bits of text as you're culling sections. When you're ready to start, click on Open File and choose the PDF you want to "mash".

Once open, the pane below fills up in a manner that at first glance is overwhelming and incomprehensible. However, on a very basic level, each line is a section of text in your PDF. If you explore each line, you can check which part of the PDF is being examined, and if it's redundant, you can choose to ignore it in the conversion.

Looking at these PdfMasher lines in detail, each line has an X and Y axis reference, as well as font size, text length and page number. Whenever you click a line, the full text content of its section in the PDF is shown in the pane below.

If you've decided on which sections to remove, click Ignore to cut out the text from the final product. Click Normal to reinstate the text for inclusion. Depending on which device you'll be reading the resulting e-book, the header and footer information may be something you want to cut out of the page.

For example, in the screenshot, I'm removing the beginning references and page headers in a psychology paper that otherwise would leave a hard-to-navigate, garbled mess if I translated it into something I could read on my phone.

However, if what you're preparing is intended to be something like a public Web page instead of a trimmed-down e-book, you might want to use the Title and Footnote buttons. Title will result in an H1 title header in the outputted HTML. The Footnote button will move the text to the bottom of the document, and PdfMasher will try to make one of the cool hyperlinks mentioned earlier.

Once you've finished editing your document, click on the Build tab below, and then click on the Generate Markdown button. A raw text file will be generated in the same folder as the original PDF. Click on Reveal Markdown, and the source folder will be opened in your default file manager. Edit Markdown will open the actual text file in your default text editor, and View HTML will show the end product in a Web browser.

If you've made any errors, the output will reveal them quickly, and you can go back and simply start the Build process again. From here, you either can leave your output as is or convert your files into specific e-book formats.

Either way, PdfMasher uses some very simple methods to create something very clever and is a must-have for any regular e-book reader.

Free Poker DB—Advanced On-line Poker Database

http://fpdb.wiki.sourceforge.net

According to its Freshmeat entry:

> Fpdb is a free/open-source tracker/HUD for use with on-line poker. The intent is to make fpdb capable of supporting all games on all sites.
>
> Fpdb currently supports flop games (Hold 'em, Omaha, Omaha hi/low), stud games (7 card stud, Stud 8 and Razz) and draw games (2–7 Lowball single and triple draw, Badugi and 5 card draw).
>
> Cash games are fully supported, and tournament support is improving all the time.
>
> Currently supported sites include PokerStars, Full Tilt Poker, the Everleaf network, the Boss Media network and others; see Features for a full list. Additional poker sites can be supported by writing a plugin to parse the site's hand history files. Several additional plugins are under development and in the development tree.

Installation Binaries may be available in your repository (called python-fpdb on my Kubuntu system). Source also is available, and it doesn't seem to require any awkward compiling. To get an idea of the library requirements, the documentation noted the following successful package combination for Ubuntu 9.10:

- Python 2.6
- GTK+ 2.18.3
- PyGTK 2.16.0
- matplotlib 0.99.0

- numpy 1.3.0
- sqlite3 2.4.1
- sqlite 3.6.16
- database mysql

Note the SQL dependencies, as they are particularly important. The fpdb wiki's installation section has a very involved section regarding MySQL under Gentoo, so hopefully a combination of the provided Ubuntu and Gentoo instructions will point you in the right direction for your system.

For those running with source, once you have the library requirements out of the way, you either can grab a source tarball or set up a local repository with git. For the git option, enter the command:

```
$ git clone git://git.assembla.com/free_poker_tools.git
```

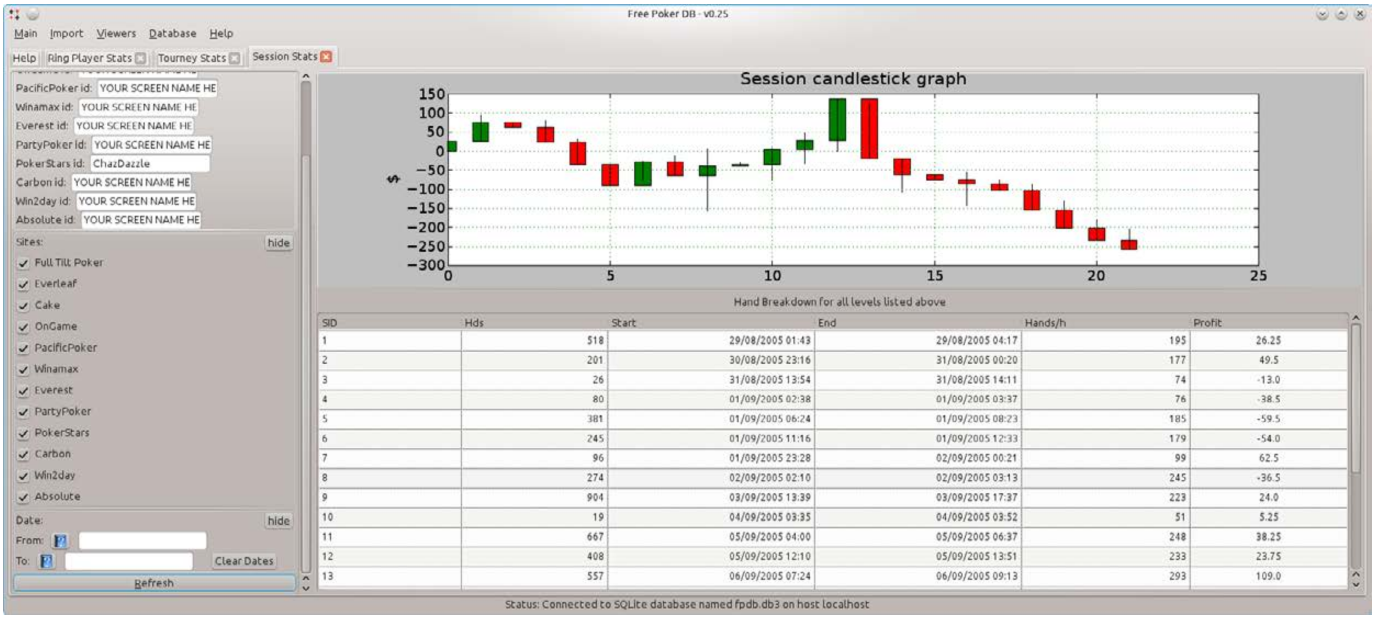
For those wanting to use the source tarball, grab the latest tarball from the Downloads page and extract it. Open a terminal in the new folder, and you should be able to run the program simply by entering:

```
$ ./run_fpdb.py
```

Whereas my Kubuntu binary ran with the command:

```
$ fpdb
```

Usage Before you start using fpdb, you obviously have to play some poker on one of the supported games, building up an account with some dealt hands against other players and whatnot. Once you've done that, you need to locate the local



PacificPoker id: YOUR SCREEN NAME HE
Winamax id: YOUR SCREEN NAME HE
Everest id: YOUR SCREEN NAME HE
PartyPoker id: YOUR SCREEN NAME HE
PokerStars id: ChazDazzle
Carbon id: YOUR SCREEN NAME HE
Win2day id: YOUR SCREEN NAME HE
Absolute id: YOUR SCREEN NAME HE

Sites:
- ✓ Full Tilt Poker
- ✓ Everleaf
- ✓ Cake
- ✓ OnGame
- ✓ PacificPoker
- ✓ Winamax
- ✓ Everest
- ✓ PartyPoker
- ✓ PokerStars
- ✓ Carbon
- ✓ Win2day
- ✓ Absolute

Date:
From:
To:

Hand Breakdown for all levels listed above

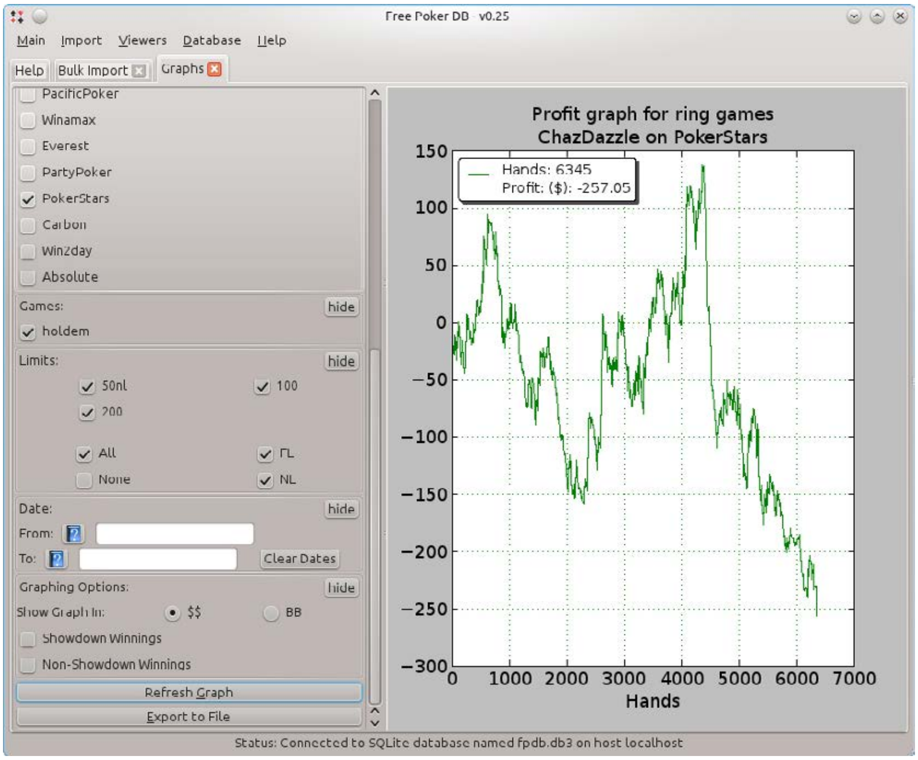
SID	Hds	Start	End	Hands/h	Profit
1	518	29/08/2005 01:43	29/08/2005 04:17	195	26.25
2	201	30/08/2005 23:16	31/08/2005 00:20	177	49.5
3	26	31/08/2005 13:54	31/08/2005 14:11	74	-13.0
4	80	01/09/2005 02:38	01/09/2005 03:37	76	-38.5
5	381	01/09/2005 06:24	01/09/2005 08:23	185	-59.5
6	245	01/09/2005 11:16	01/09/2005 12:33	179	-54.0
7	96	01/09/2005 23:28	02/09/2005 00:21	99	62.5
8	274	02/09/2005 02:10	02/09/2005 03:13	245	-36.5
9	904	03/09/2005 13:39	03/09/2005 17:37	223	24.0
10	19	04/09/2005 03:35	04/09/2005 03:52	51	5.25
11	667	05/09/2005 04:00	05/09/2005 06:37	248	38.25
12	408	05/09/2005 12:10	05/09/2005 13:51	233	23.75
13	557	06/09/2005 07:24	06/09/2005 09:13	293	109.0

Status: Connected to SQLite database named fpdb.db3 on host localhost

Fpdb provides impressive player statistics to give you the edge in on-line poker.

account files for this game so fpdb can find it later.

The best supported commercial site for Linux is PokerStars, as it runs almost perfectly under Wine. As a religious guy, I can't actually play for money, but I can vouch for PokerStars as it does have a "Play Money" mode for people like me. However, the developers did make it clear to me that fpdb is focused on real money games, so play money support isn't well tested but should work for PokerStars' cash games.



□ PacificPoker
□ Winamax
□ Everest
□ PartyPoker
☑ PokerStars
□ Carbon
□ Winzday
□ Absolute

Games:
☑ holdem

Limits:
☑ 50nl ☑ 100
☑ 200
☑ All ☑ FL
□ None ☑ NL

Date:
From:
To: Clear Dates

Graphing Options:
Show Graph In: ● $$ ○ BB
□ Showdown Winnings
□ Non-Showdown Winnings

Status: Connected to SQLite database named fpdb.db3 on host localhost

Fpdb's profit graph: looks like ChazDazzle had a bad weekend!

Moving back to fpdb, once you have some data ready to go, click on the Import menu and choose Bulk Import. Browse for the file(s) of your poker site below, and choose your game from the Site filter drop-down box. Now, click Bulk Import and wait a moment for your data to be processed.

I can take you through only a few basic steps, but it should be enough to get you started, after which you should pick up things pretty easily. Looking at the graphs first, click on the Viewers menu and choose Graphs. Find your game(s) below, and enter your user ID. Now in the Sites pane, check/uncheck the games you want to display, and choose Refresh Graph at the bottom. If all went well, a profit graph should display in the panel on the right.

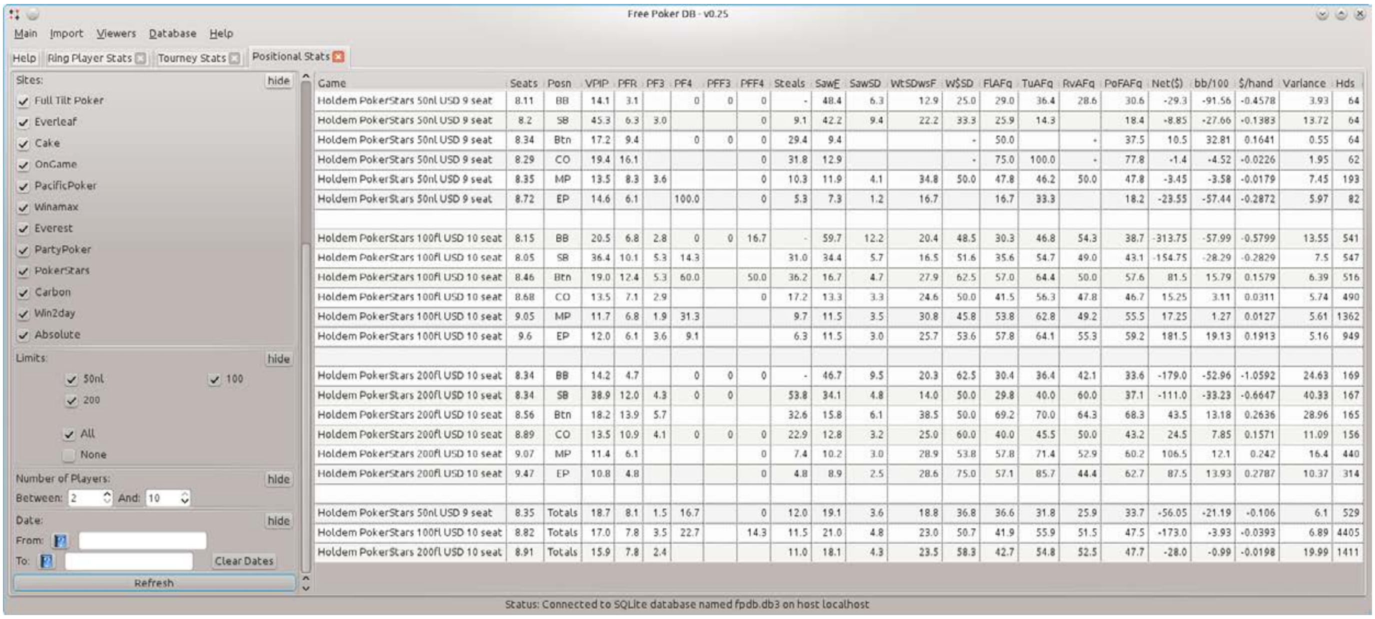
This last step really shows you how to use all of the other viewers as well, so feel free to explore the other features to your heart's content—Ring Player Stats, Tourney Stats, Positional Stats, it's all there. And don't feel restricted to your own account either. You also can see the stats of other players, which, when you think of it, is really the whole point of this program!

Ultimately, Free Poker DB will give a genuine edge to any serious on-line poker players, particularly those taking part in tournaments and the like. However, I'd like to end this month on a slightly different note—an unsolicited comment from co-developer "Chaz" on some heart-warming realities of OSS:

I got involved in the project about a year ago after leaving my job as a management consultant in Washington, DC, to start Pokeit. Pokeit is a similar product to fpdb, except it's a Web application and a commercial venture. On the face of it, Pokeit's collaboration with fpdb might seem a bit odd, given that we're trying to charge for something that fpdb gives away for free. In practice, it feels perfectly natural, and really it should. It almost goes without saying that you can't launch a business today without depending on open-source software—whether that be databases, such as MySQL or Postgres, free development tools, languages, add-on modules or niche libraries. Likewise for us, any tools that track and analyze hands of Internet poker require a set of core functionality for reading and storing data. Developing such a core function from scratch would have been a monumental waste of time for us when fpdb already had a two-year head start and strong foundation already built. So instead of going it alone, we decided early on to collaborate with fpdb on developing the codebase in as many ways as it made sense.

Let's hope his example catches on.■

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.



Free Poker DB - v0.25
Main Import Viewers Database Help
Help | Ring Player Stats | Tourney Stats | Positional Stats

Sites:

☑ Full Tilt Poker
☑ Everleaf
☑ Cake
☑ OnGame
☑ PacificPoker
☑ Winamax
☑ Everest
☑ PartyPoker
☑ PokerStars
☑ Carbon
☑ Win2day
☑ Absolute

Limits:

☑ 50nl ☑ 100
☑ 200
☑ All
□ None

Number of Players:

Between: 2 And: 10

Date:

From:

To:

hide
hide
hide
hide
Clear Dates
Refresh

Game	Seats	Posn	VPIP	PFR	PF3	PF4	PFF3	PFF4	Steals	SawF	SawSD	WtSDwsF	W$SD	FlAFq	TuAFq	RvAFq	PoFAFq	Net($)	bb/100	$/hand	Variance	Hds
Holdem PokerStars 50nl USD 9 seat	8.11	BB	14.1	3.1		0	0	0	-	48.4	6.3	12.9	25.0	29.0	36.4	28.6	30.6	-29.3	-91.56	-0.4578	3.93	64
Holdem PokerStars 50nl USD 9 seat	8.2	SB	45.3	6.3	3.0			0	9.1	42.2	9.4	22.2	33.3	25.9	14.3		18.4	-8.85	-27.66	-0.1383	13.72	64
Holdem PokerStars 50nl USD 9 seat	8.34	Btn	17.2	9.4		0	0	0	29.4	9.4			-	50.0		-	37.5	10.5	32.81	0.1641	0.55	64
Holdem PokerStars 50nl USD 9 seat	8.29	CO	19.4	16.1				0	31.8	12.9			-	75.0	100.0	-	77.8	-1.4	-4.52	-0.0226	1.95	62
Holdem PokerStars 50nl USD 9 seat	8.35	MP	13.5	8.3	3.6			0	10.3	11.9	4.1	34.8	50.0	47.8	46.2	50.0	47.8	-3.45	-3.58	-0.0179	7.45	193
Holdem PokerStars 50nl USD 9 seat	8.72	EP	14.6	6.1		100.0		0	5.3	7.3	1.2	16.7		16.7	33.3		18.2	-23.55	-57.44	-0.2872	5.97	82
Holdem PokerStars 100fl USD 10 seat	8.15	BB	20.5	6.8	2.8	0	0	16.7	-	59.7	12.2	20.4	48.5	30.3	46.8	54.3	38.7	-313.75	-57.99	-0.5799	13.55	541
Holdem PokerStars 100fl USD 10 seat	8.05	SB	36.4	10.1	5.3	14.3			31.0	34.4	5.7	16.5	51.6	35.6	54.7	49.0	43.1	-154.75	-28.29	-0.2829	7.5	547
Holdem PokerStars 100fl USD 10 seat	8.46	Btn	19.0	12.4	5.3	60.0		50.0	36.2	16.7	4.7	27.9	62.5	57.0	64.4	50.0	57.6	81.5	15.79	0.1579	6.39	516
Holdem PokerStars 100fl USD 10 seat	8.68	CO	13.5	7.1	2.9			0	17.2	13.3	3.3	24.6	50.0	41.5	56.3	47.8	46.7	15.25	3.11	0.0311	5.74	490
Holdem PokerStars 100fl USD 10 seat	9.05	MP	11.7	6.8	1.9	31.3			9.7	11.5	3.5	30.8	45.8	53.8	62.8	49.2	55.5	17.25	1.27	0.0127	5.61	1362
Holdem PokerStars 100fl USD 10 seat	9.6	EP	12.0	6.1	3.6	9.1			6.3	11.5	3.0	25.7	53.6	57.8	64.1	55.3	59.2	181.5	19.13	0.1913	5.16	949
Holdem PokerStars 200fl USD 10 seat	8.34	BB	14.2	4.7		0	0	0	-	46.7	9.5	20.3	62.5	30.4	36.4	42.1	33.6	-179.0	-52.96	-1.0592	24.63	169
Holdem PokerStars 200fl USD 10 seat	8.34	SB	38.9	12.0	4.3	0	0		53.8	34.1	4.8	14.0	50.0	29.8	40.0	60.0	37.1	-111.0	-33.23	-0.6647	40.33	167
Holdem PokerStars 200fl USD 10 seat	8.56	Btn	18.2	13.9	5.7				32.6	15.8	6.1	38.5	50.0	69.2	70.0	64.3	68.3	43.5	13.18	0.2636	28.96	165
Holdem PokerStars 200fl USD 10 seat	8.89	CO	13.5	10.9	4.1	0	0	0	22.9	12.8	3.2	25.0	60.0	40.0	45.5	50.0	43.2	24.5	7.85	0.1571	11.09	156
Holdem PokerStars 200fl USD 10 seat	9.07	MP	11.4	6.1				0	7.4	10.2	3.0	28.9	53.8	57.8	71.4	52.9	60.2	106.5	12.1	0.242	16.4	440
Holdem PokerStars 200fl USD 10 seat	9.47	EP	10.8	4.8				0	4.8	8.9	2.5	28.6	75.0	57.1	85.7	44.4	62.7	87.5	13.93	0.2787	10.37	314
Holdem PokerStars 50nl USD 9 seat	8.35	Totals	18.7	8.1	1.5	16.7		0	12.0	19.1	3.6	18.8	36.8	36.6	31.8	25.9	33.7	-56.05	-21.19	-0.106	6.1	529
Holdem PokerStars 100fl USD 10 seat	8.82	Totals	17.0	7.8	3.5	22.7		14.3	11.5	21.0	4.8	23.0	50.7	41.9	55.9	51.5	47.5	-173.0	-3.93	-0.0393	6.89	4405
Holdem PokerStars 200fl USD 10 seat	8.91	Totals	15.9	7.8	2.4				11.0	18.1	4.3	23.5	58.3	42.7	54.8	52.5	47.7	-28.0	-0.99	-0.0198	19.99	1411

Status: Connected to SQLite database named fpdb.db3 on host localhost

The Positional Stats are comprehensive to say the least.

BREWING SOMETHING FRESH, INNOVATIVE OR MIND-BENDING? Send e-mail to newprojects@linuxjournal.com.

POLICE PUBLIC CALL BOX

BUILDING AN ULTRA-LOW-POWER FILE SERVER WITH THE TRIM-SLICE

HOW-TO: replace a large, inefficient custom-built file server with the Trim-Slice.

DANIEL BARTHOLOMEW

For the past several years, I've used a custom-built file server at my house. I've upgraded it many times, but it began life, as near as I can recall, in April 2000. When I say "upgraded", I mean the internals have been swapped completely on at least two occasions among other things. The most-recent major upgrade was in 2006 (or thereabouts) when I added a software RAID5 with three 500GB hard drives (later expanded to six). It has chugged along merrily for years, but lately it has begun showing its age. For starters, two terabytes of space isn't all that much anymore. Also, it's not as efficient power-wise as I would like (in my measurements, it draws between 1.8 and 2.0 amps continuously, depending on load). Finally, the case for this server takes up way too much space (it's a full tower).

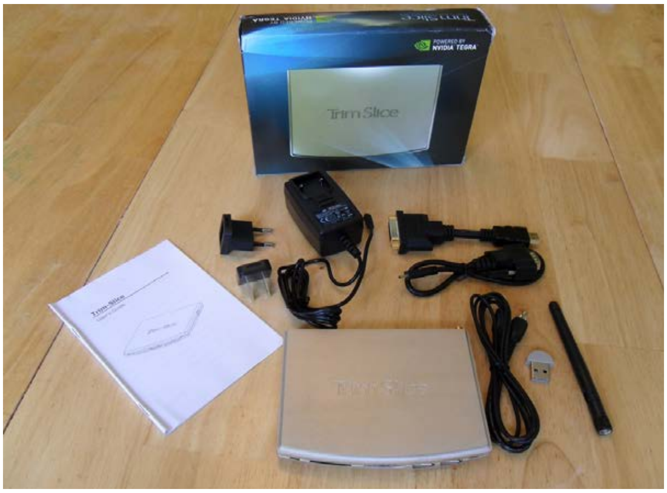


Figure 1. The Trim-Slice and Everything That Comes in the Box

As an experiment, and finally to get rid of that large, inefficient and ugly tower case, I decided to use the new Trim-Slice as the base for an ultra-low-power, ultra-small replacement file server. The Trim-Slice is built on the NVIDIA Tegra 2 platform, and the specific model I purchased features a 1GHz dual-core ARM Cortex A9 processor, 1GB of RAM and a 32GB SATA SSD. Did I mention that it's really small? You know, teeny—like, I-can't-believe-this-is-a-full-computer small. The dimensions are 130mm x 95mm x 15mm. For comparison purposes, a standard 3.5" hard drive has dimensions of approximately 146mm x 102mm x 25mm.

On the outside, it has an RS232 serial port, SD and microSD card slots (both SDHC-compatible), four USB ports, HDMI and DVI-D video out ports, 802.11n and a Gigabit Ethernet port. Inside, it comes with Ubuntu pre-installed on the SSD (10.10 "Maverick" was installed on the one I received, but there is now an update to 11.04 "Natty", which I applied and which I expect is now shipping on newly ordered units).

The full Ubuntu Linux inside is what set this solution above alternatives like the Drobo FS, ReadyNAS and others, at least in my mind. The price is in the same ballpark too. The model I ordered, complete with shipping from Israel, came to $335.

The main downside for my purposes is that there is no place to connect internal hard drives. I have to make do with external USB drives instead. I don't like the thought of running a software RAID over USB, so I further decided simply to use multiple large external USB drives (each with at least one corresponding backup drive).

To start with, my goal was to replace the old tower server, which just requires the Trim-Slice and two 2TB external USB hard drives. Yes, a single hard drive, especially a USB drive, is not as reliable or nearly as fast as a RAID5 array, but it's a compromise

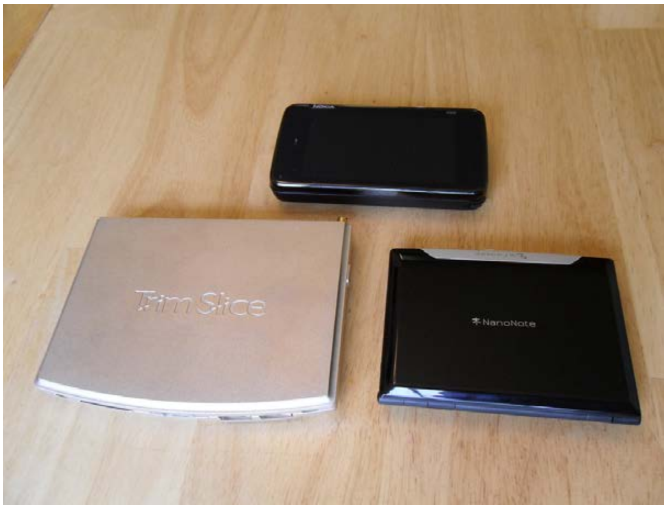


Figure 2. Size Comparison: the Trim-Slice Next to a Nokia N900 and the Ben NanoNote

I'm willing to make for the power, space and noise savings. USB is plenty fast for my needs, and besides, with two drives, I have a backup.

Anatomy of a File Server

The purpose of a file server is to serve files over a network. There are many ways to do this, but I focus on the most common ones here.

First, there is classic "file server" software: NFS and Samba. These systems don't care what your data is. All they see are files, and no file is any different from the next (apart from size and permissions).

The new kids are content-aware file servers like UPnP and DAAP. This type of file server software does care about content types, and it serves up metadata about your files along with the files themselves. It will refuse to serve files it doesn't recognize or support. But, it can do some tricks that NFS and Samba can't, like alter data on the fly for clients who can't read the original data. So, they're more fussy than classic file server software, both to set up and run, but they do have advantages.

UPnP and DAAP are designed specifically for serving audio, video and image files. DAAP is built-in, or available as a plugin, for many popular audio jukebox apps, such as Rythmbox, Amarok and Banshee, but there also are standalone server applications available. UPnP Media Server support is built in to various consumer devices, such as the PlayStation 3, Xbox 360 and various handheld and set-top media players.

NFS

NFS is the classic Network File System, and it is has been in use for decades on Linux and UNIX. The Popcorn Hour media player connected to my TV supports NFS, and I don't have any Windows computers, so NFS is really the only classic file-serving protocol I need (or want) on my network. NFS has very limited security, so it's not ideal for everyone, but it's lightweight and easy to configure. In my opinion, if you have a device that supports NFS and SMB, go with NFS.

On Ubuntu, the NFS server I use is called nfs-kernel-server, and you can install it with the following:

```
sudo apt-get install nfs-kernel-server
```

To create an nfs share, edit the /etc/exports file, and add the directory you want to export. Here is an example:

```
/mnt/disk01 popcorn(ro,sync,root_squash,no_subtree_check)
```

The above line exports the /mnt/disk01 directory to my Popcorn Hour, with the following flags:

- `ro` — read only: in other words, don't allow anything that could change the filesystem. The Popcorn Hour has the ability to delete items, but I don't want to let my kids delete things arbitrarily or accidentally with the remote.

- `sync` — reply to requests only after the changes have been committed to stable storage.

- `root_squash` — map requests from uid/gid 0 to the anonymous uid/gid. This makes things a little more secure.

- `no_subtree_check` — from the man page: "This option disables subtree checking, which has mild security implications, but can improve reliability in some circumstances." See the man page for more information (`man exports`).

With the line in place, I run the `sudo exportfs -ra` command to refresh the exports. Then on the Popcorn Hour, I can mount the exported directory, and away I go. There are several other options you can use in the /etc/exports file. See the exports man page for details.

The example entry above can't be mounted on any other host, but to permit other hosts to do so, I either can change `popcorn` to the IP address and netmask of the network I want to share it with (for example, 192.168.10.0/24 for every host with an IP address starting with 192.168.10.), or I can add additional host definitions to the end of the line.

With the exports file updated and refreshed, I can mount the export with something like this:

```
sudo mount -t nfs trimslice:/mnt/trimslice/disk01
 ➥/mnt/trimslice/disk01
```

Or, I could add an entry like the following to my /etc/fstab file:

```
trimslice:/mnt/trimslice/disk01  /mnt/trimslice/disk01
 ➥nfs  defaults  0 0
```

and the NFS share always would be mounted at boot time.

Samba SMB/CIFS

Samba, aka SMB/CIFS, is how you go about sharing files with computers running Windows. If I had a Windows machine or two, using Samba would be a given. I don't, but I'll go ahead and describe the process here. For starters, Samba is installed on the Trim-Slice with the following:

```
sudo apt-get install samba
```

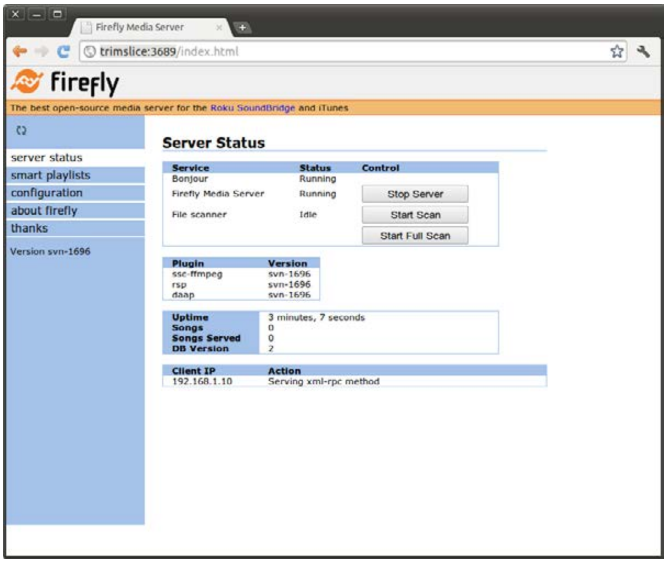
After installation, edit the /etc/samba/smb.conf file to set up your shares (add them to the end of the file). A read-only share equivalent to the NFS one described above is:

```
[disk01]
    comment = trimslice disk01
    path = /mnt/disk01
    browsable = yes
    guest ok = yes
    read only = yes
```

Add the above to the end of the smb.conf file, and the share will pop into existence on the network. With Samba, there is no need to restart the service or run a command after editing the smb.conf file; any changes are applied automatically as soon as the file is saved.

It's a good idea to uncomment the `security = user` line in the smb.conf file to add some security (and if you do want security, you should set `guest ok` in the above example to `no`). And, if you have a proper Windows network, you should change the `workgroup` name in the smb.conf file to the actual name of your Windows workgroup.

As with NFS, you can enter a lot more settings in the smb.conf file to tweak things just the way you want them. The default file is filled with examples, and the Samba documentation goes into even greater detail.



The best open-source media server for the Roku SoundBridge and iTunes

Version svn-1696

Server Status

Service	Status	Control
Bonjour	Running	
Firefly Media Server	Running	Stop Server
File scanner	Idle	Start Scan
		Start Full Scan

Plugin	Version
ssc-ffmpeg	svn-1696
rsp	svn-1696
daap	svn-1696

Uptime	3 minutes, 7 seconds
Songs	0
Songs Served	0
DB Version	2

Client IP	Action
192.168.1.10	Serving xml-rpc method

Figure 3. mt-daapd, aka the Firefly Media Server

DAAP

DAAP, in case you are interested, stands for Digital Audio Access Protocol. An older, but serviceable, standalone DAAP server for Linux is mt-daapd, also known as the Firefly Media Server. Unfortunately, it is not under active development. Some forks are in the works (which aren't in the Ubuntu repositories yet), so maybe the situation will improve in the future. To install it, do the following:

```
sudo apt-get install mt-daapd
```

After installing mt-daapd, set the password for the admin account in the /etc/mt-daapd.conf file. Technically, the password already is set, but it's good practice to change it. You can tweak other settings in the file, but the GUI is easier.

After changing the password, restart mt-daapd with:

```
sudo /etc/init.d/mt-daapd restart
```

Then, go to the Web interface to configure it: http://trimslice:3689 (replace "trimslice" in the URL with the correct IP address or name).

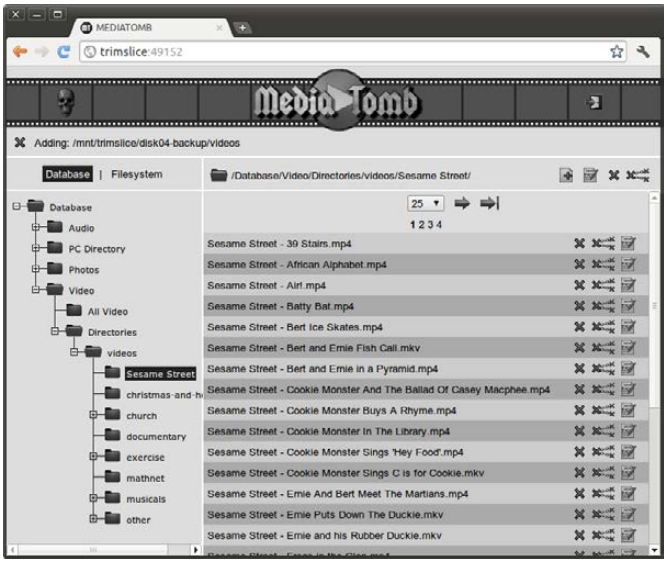
The configuration page is simple and self-explanatory. You can set the name, change the admin password and set a password for listening to the music (in case you don't want to share your collection of classic Dr. Who music with everyone on your network). You also set which folder or folders contain your music (multiple folders can be specified). Finally, you can configure how often to have mt-daapd rescan your music folder(s).

Once the changes are to your liking, pressing the Save button saves the settings to the /etc/mt-daapd.conf file. But, the GUI is there so you might as well use it.

All should be well and good at this point. Unfortunately, mt-daapd, as packaged in the repository the Trim-Slice uses, does not support FLAC files. If your collection is mostly MP3 files, that won't be an issue. If it is an issue, your options are to compile your own, live with the limitation or find an alternative.

UPnP

For serving up video files to my PS3, I use the MediaTomb UPnP media server. Or at least, I would, if I didn't have the Popcorn Hour. MediaTomb, like mt-daapd, is a nice piece of software and works just fine for what it does, but devices like the PS3 can be very picky about what file types they will support. On-the-fly transcoding (supported by both mt-daapd and MediaTomb) can eliminate some of these issues, especially with audio files (for example, by transcoding a FLAC file to WAV while it is being transferred, so that iTunes can play it). Transcoding isn't practical for video files though. It can be done, but the CPU requirements are hefty to say the least, especially when you start talking about 720p and larger video files.



Adding: /mnt/trimslice/disk04-backup/videos

/Database/Video/Directories/videos/Sesame Street/

Sesame Street - 39 Stairs.mp4
Sesame Street - African Alphabet.mp4
Sesame Street - Airl.mp4
Sesame Street - Batty Bat.mp4
Sesame Street - Bert Ice Skates.mp4
Sesame Street - Bert and Ernie Fish Call.mkv
Sesame Street - Bert and Ernie in a Pyramid.mp4
Sesame Street - Cookie Monster And The Ballad Of Casey Macphee.mp4
Sesame Street - Cookie Monster Buys A Rhyme.mp4
Sesame Street - Cookie Monster In The Library.mp4
Sesame Street - Cookie Monster Sings 'Hey Food'.mp4
Sesame Street - Cookie Monster Sings C is for Cookie.mkv
Sesame Street - Ernie And Bert Meet The Martians.mp4
Sesame Street - Ernie Puts Down The Duckie.mkv
Sesame Street - Ernie and his Rubber Duckie.mkv

Figure 4. The Media Tomb File Browser

Limitations aside, installing and configuring MediaTomb is similar to mt-daapd. First, enter the following:

```
sudo apt-get install mediatomb-daemon
```

After the install completes, edit the /etc/mediatomb/config.xml file to enable the graphical user interface (GUI) and set the default user and password. To do this, change `enabled="no"` in the following lines to `enabled="yes"` (both of them), and set the password to something more secure:

```
<ui enabled="no" show-tooltips="yes">
  <accounts enabled="no" session-timeout="30">
    <account user="mediatomb" password="mediatomb"/>
```

The above lines should be near the top of the file. Save the file, and restart the server with:

```
sudo /etc/init.d/mediatomb stop
sudo /etc/init.d/mediatomb start
```

Once restarted, connect to http://trimslice:49152/. Enter the user name and password, and you will be in the GUI. To add a folder, click the Filesystem link, and browse to the folder you want MediaTomb to index. With the correct folder selected in the left pane, click on the plus, or plus-with-a-circle icon, to have MediaTomb scan the contents of the folder. The plus-with-a-circle icon adds the folder as an autoscan folder, meaning it will rescan the folder periodically looking for new files.

For PS3 support, a couple lines need to be changed in the config.xml file; they are commented and easy to find if you search for "PS3".

Backups

With this new file server, I lose the protection of RAID, so backups are more important. RAID, of course, does not eliminate the need for backups; it just makes the primary filesystem more reliable. Because I already needed backups with my old setup, I had a backup system in place.

The "system" itself is a custom rsync backup shell script. The backup drive contains several directories: one named current and then 14 others named 01, 02, 03 and so on, up to 14. The basic

flow of the script is:

```
rm -rf '14'
mv '13' '14'
mv '12' '13'
...
mv '01' '02'
cp -al 'current' '01'
rsync drive-to-back-up to 'current'
```

The `-al` part of the copy command above is important. It tells the command to operate in archive mode, which preserves attributes and copies directories recursively, and to create hard links instead of actually copying the files.

When rsync comes upon a changed file, it will de-link the file before updating it, so the combination of rsync and the cp command gives me 14 days of backups (assuming the script is run once a day).

The extra space required for these backups is low, so I can use the same size drive for the backups that I do for the primary. Once a backup drive starts nearing its limit, the primary drive likely will be close to its limit too, and it will be time to shop for an additional pair of drives.

I've used variations of this script for years, and if I had to start from scratch today, I might use it or something else.

Lots of excellent backup programs are available for Linux. The point is to *make* backups, as many as necessary.

Conclusion

The Trim-Slice has worked out very well as a file server. The built-in serial port lets me operate it completely without a monitor, and the hardware has so far been more than adequate for my household's modest file-serving needs.

With dual 2TB external USB disk drives (and more to come), the Trim-Slice is even more energy-efficient than I thought it would be. My consumer-grade "Kill-a-Watt" power meter (which I admit is probably not very accurate) shows an average power draw of 0.28 amps, which, completely accurate or not, is much better than the 1.8 to 2.0 amps the old server was pulling. The power draw of the Trim-Slice by itself is an astounding 0.08 amps.

Power is just one benefit. I also like the smaller footprint. Noise is much better too. The old case needed several fans, but the Trim-Slice is passively cooled. There are fans in the external drive enclosures, but they don't come on very often, and when they do, I don't notice them at all. The Trim-Slice does get a bit hot to the touch, but I suppose that's what you get when you make the outer casing a heat sink.

The jury is still out on how long this new setup will last. I consider USB drives to be less reliable, and although the build quality of the Trim-Slice appears high, it's a new product with no history. To mitigate this, I am going to be very careful to make sure important stuff is copied across all future drives. Despite my worries, I must admit, I did replace a few of the drives in my RAID5 over the years, and I don't imagine the difference in reliability will be so great as to cause any huge problems.■

Daniel Bartholomew works for Monty Program (http://montyprogram.com) as a technical writer and system administrator. He lives with his wife and children in North Carolina and often can be found hanging out in #maria on Freenode IRC (he occasionally pokes his head into #linuxjournal too).

Resources

The Trim-Slice: **http://trimslice.com**

The Trim-Slice User Manual: **http://trimslice.com/download/documentation/trim-slice-user-guide.pdf**

Linux NFS-HOWTO: **http://nfs.sourceforge.net/nfs-howto**

Samba: **http://www.samba.org**

MediaTomb: **http://mediatomb.cc**

Firefly Media Server: **http://en.wikipedia.org/wiki/Firefly_Media_Server**

The Popcorn Hour: **http://www.popcornhour.com**

OSDC 11

Nov. 16–18;
MiniConfs Nov. 14–15

For developers.
By developers.

8th annual Open Source Developers' Conference
this year at the Australian National University,
Canberra, Australia.

43
Speakers

6
MiniConfs

5
Days

with keynotes by **Sen. Kate Lundy**, **Damian Conway**, **Brian Catto**, and **Tony Beal**.

schedule & registration at

osdc.com.au

Learning to Program the Arduino

Want to learn how to program a microcontroller and set up a home automation system? Read on to learn the basics of Arduino so you can get started.

Amit Saha

This article should acquaint you with basic Arduino programming and show you how to write programs that interact with objects in the real world. (A mandatory disclaimer: the last time I really studied electronics was way back in high school, so this article focuses more on the programming aspects, rather than the electronic side of things.)

Physical Computing

Before I start talking about this really cool thing called Arduino (Italian for "good friend"), let me say a few things about the fascinating subject of physical computing. Physical computing has been defined in various ways, but the central idea seems to be the same: physical computing is concerned with developing software that interacts with the world beyond the host computer through a combination of hardware and software—it's aware of the world, so to speak. Such awareness makes these applications capable of sensing external events and responding to them in a predefined way. This is accomplished by the use of sensors and actuators (which I describe next).

Actuators and Sensors

The Arduino interacts with the world through actuators and sensors. Sensors are electronic components that describe the world to your application. One common way sensors work is that their electrical properties change (in a mathematically known way) as an effect of the changes in the conditions in which it's operating. For example, the resistance of a photo-resistor changes when the intensity of light incident on it changes. Thermistors are another example of such sensors whose resistance changes when the operating temperature changes. A flex sensor is a different category of sensor, where the resistance changes depending on the extent of the flex or "bend". Such changes can be read as electrical signals on the Arduino's input pin. Depending on the kind of sensor, the signal either can be digital (on or off) or analog (a continuous stream of values). The latter part of this article shows how to work with analog sensors.

Actuators, on the other hand, are electronic components that are used to react to an external event. For example, when it gets dark, the light should be switched on. Sensors and actuators, thus, are used to achieve complementary objectives: one senses, and the other reacts. Examples of actuators are solenoids and servos. Later in this article, I explain how to control a servo using Arduino.

Arduino

The Arduino is an open-source electronics prototyping platform composed of two major parts: the Arduino board (hardware) and the Arduino IDE (software). The Arduino IDE is used to write the program that will interact with your Arduino and the devices connected to it. In the Arduino world, such a program is called a sketch, which has its origin in its mother language, Processing (see Resources).

The Arduino board is a small-form microcontroller circuit board. At the time of this writing, a number of Arduino boards exist: Arduino UNO, Nano, Mega, Mini, Pro and others (see Resources for a complete list). The Arduino UNO (Figure 1) is the latest version of the basic Arduino board, and you need one of these to follow this article (see Resources for the UNO's detailed specifications).

Besides the UNO, you need the following hardware to work through this article:

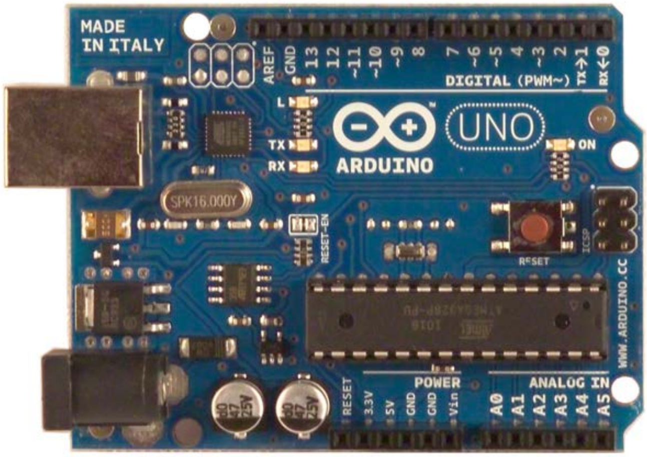


Figure 1. Arduino UNO
(Courtesy of http://arduino.cc/en/Main/ArduinoBoardUno)

1. Breadboard to set up the circuit.

2. Some LEDs.

3. Resistors: 330 Ohm (at least as many as LEDs), 10 kOhm resistors.

4. Continuous rotation servo (SpringRC SM-S4303R continuous rotation servo: **http://www.robotgear.com.au/Product.aspx/Details/482**).

5. Flex sensor.

6. Linear potentiometer.

7. Connecting wires.

One excellent way to get started with Arduino is the Arduino starter kit from Sparkfun. This starter kit contains all the hardware and more that you need to follow this article (except the servo).

If you haven't already opened up your Arduino and plugged it in to your USB port, plug it in. For the purposes of this article, it will be sufficient to use the power supply via the USB connection. If you connect more devices, you will need to connect an external supply.

You will program the Arduino in a language that looks very similar to C and is based on Processing. You can download the Arduino IDE from the Arduino Project Web site.

Arduino IDE

As you might guess, the IDE is as always the front end. The real pieces are the compilers, linkers and libraries that need to be present to communicate and program the AVR microcontroller-based Arduino. Depending on your Linux distribution, the exact names of the packages will vary, so I just list the software by name here:

- The GNU C and C++ compiler for AVR.

- AVR binutils.

- AVR libc.

- avrdude (a program for uploading code to the microcontroller board).

- rxtx (for serial communication).

Once these packages are installed, fire up your Arduino IDE. Take a moment to explore the IDE. The buttons for compiling (verifying) and uploading the sketch are the important ones.

The communication between your computer and the Arduino will be via the USB cable that has been packaged with your Arduino board. Once you plug the USB cable in to your computer (with the other end plugged in to the Arduino board), it should show up in the Arduino IDE under Tools→Serial Port as /dev/ttyACMx. If you have more than one USB serial device communicated, be careful to select the correct one. You need to set up user permissions correctly to access the serial port (see Resources for distribution-specific instructions).

Blink Those LEDs

For the first sketch, let's blink an LED and then extend it to blink multiple LEDs alternately. Before the software part, let's first set up the circuit to connect the LED to the Arduino. The

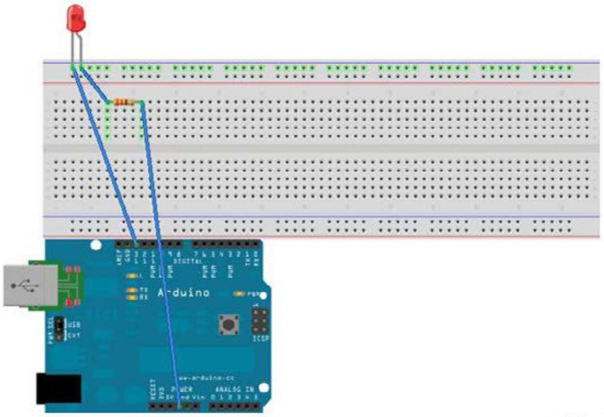


Figure 2. Circuit for a Single LED Blink

Listing 1. Simple LED Blink Sketch

```
/*
 Blink
 Turns on an LED on for one second,
 then off for one second, repeatedly.
 This example code is in the public domain.
 */

void setup() {
 // initialize the digital pin as an output.
 // Pin 13 has an LED connected on most Arduino boards:
 pinMode(13, OUTPUT);
}

void loop() {
 digitalWrite(13, HIGH);   // set the LED on
 delay(1000);              // wait for a second
 digitalWrite(13, LOW);    // set the LED off
 delay(1000);              // wait for a second
}
```

completed circuit should look like Figure 2.

Next, in your Arduino IDE, open the sketch in Examples→Basics→Blink, which should look like Listing 1.

As you can see from that sketch and the circuit diagram, the LED is connected to the digital pin 13 of the Arduino. Once you verify (compile) the sketch and upload it to the Arduino board, you should see the blinking LED.

Because this is your first sketch, take some time to understand the general framework of an Arduino sketch. If you are familiar with C or C++, you will notice that you have two functions in this sketch: setup() and loop(). The code that you write in setup() is meant for initialization and is executed once the sketch is uploaded to the board. The code in loop() is executed repeatedly as long as the power to the Arduino is supplied. Even if you power off

Listing 2. Multiple LED Blink Sketch

```
/* Multiple LED Blinking program
  Amit Saha

*/

// constants won't change. Used here to
// set pin numbers:
const int numPins = 3;
const int ledPin [] =  {11,12,13};   // the number of LED pins

int interval = 100;    // interval at which to blink (milliseconds)

void setup() {
  // Iterate over each of the pins and set them as output
  for(int i=0;i<numPins;i++)
    pinMode(ledPin[i], OUTPUT);
}

/* Loop until death */
void loop()
{
  for(int i=0;i<numPins;i++)
  {
    digitalWrite(ledPin[i],HIGH);
    delay(interval);
    digitalWrite(ledPin[i],LOW);
    delay(interval);
  }
}
```

the Arduino and power it back on, the sketch still resides until you overwrite it with another sketch. An Arduino sketch should be saved in a file with the extension pde and is stored in a directory of the same name.

For the next sketch, let's connect multiple LEDs and blink them alternately to create a cool rippling effect. (You might want to

Listing 3. Fade an LED On/Off Using a Linear Potentiometer

```
/* Using potentiometer to fade on/off an LED
*  Original code notice below:
* ------------------------------
* Demonstrates analog input by reading an analog sensor on
* analog pin 0 and turning on and off a light emitting
* diode (LED) connected to digital pin 13.
* The amount of time the LED will be on and off
* depends on the value obtained by analogREAD().
* Created by David Cuartielles
* Modified 16 Jun 2009
* By Tom Igoe
* http://arduino.cc/en/Tutorial/AnalogInput
*/

int sensorPin = 0;
int ledPin = 13;
int sensorValue = 0;

void setup() {
     //declare the ledPin as an OUTPUT:
     pinMode(ledPin, OUTPUT);
     Serial.begin(9600);
}

void loop() {
        sensorValue = analogRead(sensorPin);//

        digitalWrite(ledPin, HIGH);

        delay(sensorValue);

        digitalWrite(ledPin, LOW);

        delay(sensorValue);
}
```

Listing 4. Fade an LED On/Off Using a Flex Sensor

```
/* Flex sensor + LED
/* Analog Input
* Demonstrates analog input by reading an analog sensor
* on analog pin 0 and turning on and off a light emitting
* diode (LED) connected to digital pin 13.
* The amount of time the LED will be on and off depends
* on the value obtained by analogREAD().
* Created by David Cuartielles
* Modified 16 Jun 2009
* By Tom Igoe
* http://arduino.cc/en/Tutorial/AnalogInput
* Modified 16 July, 2011
* By Amit Saha
* Current code was tested with a Flex sensor
*/
int sensorPin = 0; /*Flex sensor pin */
int ledPin = 13; /* LED pin*/

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {

  int loop=1;
  float sensorValues=0.0,delaytime;
  for(loop=1;loop <=10 ;loop++)
  {
     sensorValues = sensorValues + analogRead(sensorPin);
  }

  //Use the average as a delay value
  delaytime = sensorValues/10;

  digitalWrite(ledPin, HIGH);
  delay(delaytime);
  digitalWrite(ledPin, LOW);

delay(delaytime);
}
```

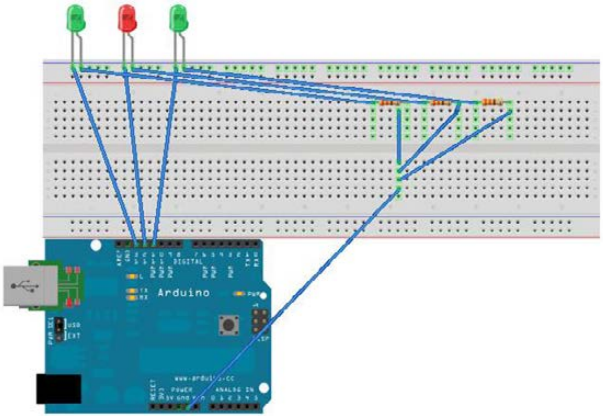


Figure 3. Circuit for Multiple Blinking LEDs

connect more than three LEDs of different colors.) The circuit for this sketch is shown in Figure 3.

Once you have uploaded this sketch to your Arduino UNO board, your LEDs should put on a colorful performance.

Analog Sensors

In the first sketch, you turned the LED on and off by writing to the Arduino's digital pin. What if you wanted something in between—say, to fade on and off? Say hello to a tiny little device called a potentiometer. To set up the circuit, connect the central pin of the potentiometer to the analog pin 0, and the other two pins to the +5V supply and ground, respectively. The LED should be connected as in the first sketch.

In Listing 3, you can see that the reading from the potentiometer is used to control the delay time, so the effect of fading in and out is produced.

Next, let's use an electronic component called a flex sensor to (you guessed it) control an LED. Basically, let's use the flex sensor in place of the potentiometer of the earlier circuit. A flex sensor is an analog sensor whose resistance changes with its bending angle. You will see that the resistance changes as you bend on either side—increasing on one side and decreasing on the other. The Arduino sketch is shown in Listing 4.

The flex sensor has two pins: one end should be connected to the +5V input, and the other pin should be connected to the analog pin 0 and farther on to the ground via a 10 kOhm resistor.

Listing 5. Example of a Server Sketch

```
/* Serial communication demo: +1*/

/* Server program
Amit Saha*/


int number;

void setup()
{
  // Open the serial connection, 9600 baud
  Serial.begin(9600);
 }

void loop()
{
  // Get the data "packet"
  // Wait for some data to arrive
  if (Serial.available()>1) {
      //operation=Serial.read();
      number=Serial.read();

      Serial.println(number + 1);
    }

}
```

Serial Communication

All the sketches so far have made use of the Arduino library calls to read and write to the Arduino pins. You haven't directly made use of the serial communication between the host computer and the Arduino board. First, I'll describe how to write a basic client/server style program.

The "server" program is a sketch that lives on the Arduino board waiting for serial data to be available (an integer in this case) and sends back the number by adding 1 to it to the "client".

The client side of the program is written in Processing using its serial library (Listing 6).

To run the client, download the processing IDE and create a new sketch with the code as shown in Listing 5. Run this code after you have uploaded the server sketch (Listing 5) to the Arduino.

Controlling a Servo

Now, let's make things move with your Arduino code. For this example, let's control the movement of a servo—that is, start and stop the servo and control its speed and rotation. Three wires protrude out of the servo: control (white/yellow), power (red) and ground (black/brown). First, set up the circuit such that the the control wire is connected to the Arduino's digital pin 2, the red wire is connected to the Arduino's 5V input and the black wire to the ground. Now, upload the sketch (Listing 7) to your Arduino. You also optionally may connect an LED to the digital pin 13 (in the same way you connected a single LED earlier), which will turn on or off depending on whether the servo is rotating.

Once the sketch is uploaded, open a serial communication channel using `screen` (feel free to substitute it with your favorite terminal communication program). Type `screen /dev/tttyACM0 9600`, and you should see the "servo prompt" at your service:

```
Arduino Serial Servo Control

Keys:'(s)lower' or '(f)aster', spacebar to center and o to stop
```

Pressing the keys to the servo should produce the desired behavior. If you see the code listing for the servo mechanism, you will see that the key to controlling the speed is basically the duration of the delay (variable pulseWidth) between sending a HIGH and LOW signal to the servo. Here, we are simulating an analog behavior using the important technique called Pulse Width Modulation, which you can read about elsewhere.

If you have gotten this servo example up and running, you also might want to check out the other example sketches for working with servos in the Arduino IDE under File→Examples→Servo.

A Peek under the Hood

I'm drawing toward the close of this article, and I hope you have had a lot of fun. However, if you are like me, you already have started wondering what's

Listing 6. Example of a Serial Client Sketch

```
/*Client for Serial communication*/
/* Amit Saha */

import processing.serial.*;

Serial myPort; // The serial port

// initial variables:
int i = 1; // counter
char inData;

void setup () {

size(400, 350); // window size

// List all the available serial ports
println(Serial.list());

// Pick up the first port, since I usually have
// just the Arduino connected.
// Make sure the correct port is selected here.
myPort = new Serial(this, Serial.list()[0], 9600);
myPort.clear();

// set initial background:

background(255); }

void draw () {

  myPort.write(4);
  //myPort.write(5);

  if (myPort.available() > 0) {
    inData = (char)myPort.read(); // Typecast it to the corresponding
                                  // character for the
                                  // ASCII value
    serialEvent();
   }

}

void serialEvent () {
  System.out.println(inData);
}
```

Listing 7. Servo Control Sketch

```
/*
 * servo1: servo1.pde

 * Servo control from the Serial port
 *
 * Slower, faster, Center and Stop a Servo with an LED Blinky
 * Created: 1 June, 2011, Amit Saha (http://echorand.me)
 * Adapted from http://principialabs.com/arduino-serial-servo-control/
 */

/** Adjust these values for your servo and setup, if necessary **/
int servoPin     =  2;    // control pin for servo motor
int minPulse     =  500;  // minimum servo position
int maxPulse     =  3000; // maximum servo position
int turnRate     =  10;  // servo turn rate increment (larger value,
                            faster rate)
int refreshTime  =  20;   // time (ms) between pulses (50Hz)

int OFF=0; // This variable will be used to get/set the status of the servo

/** The Arduino will calculate these values for you **/
int centerServo;         // center servo position
int pulseWidth;          // servo pulse width
int moveServo;           // raw user input
long lastPulse   = 0;    // recorded time (ms) of the last pulse

/* LED setup*/
int ledPin=13;


void setup() {
  pinMode(ledPin, OUTPUT); // LED Blink
  pinMode(servoPin, OUTPUT);  // Set servo pin as an output pin
  centerServo = maxPulse - ((maxPulse - minPulse)/2);
  pulseWidth = 0;
  Serial.begin(9600);
  Serial.println("      Arduino Serial Servo Control");
  Serial.println("Keys:'(s)lower' or '(f)aster', spacebar to center
and o to stop");
  Serial.println();

  moveServo = 60;
}
```

```
void loop() {
  // wait for serial input
  if (Serial.available() > 0) {
    // read the incoming byte:
    moveServo = Serial.read();

    // ASCII 's' is 115, ASCII 'f' is 102, 'o' is 111, 'spacebar' is 32
    if (moveServo == 115) { pulseWidth = pulseWidth - turnRate; OFF=0;}
//slower
    if (moveServo == 102) { pulseWidth = pulseWidth + turnRate;OFF=0; }
//faster
    if (moveServo == 32) { pulseWidth = centerServo; OFF=0;} //center
    if (moveServo == 111) { OFF= 1;} //STOP

    // limit the servo pulse at min and max
    if (pulseWidth > maxPulse) { pulseWidth = maxPulse; }
    if (pulseWidth < minPulse) { pulseWidth = minPulse; }

  }

  // pulse the servo every 20 ms (refreshTime) with current pulseWidth
  // this will hold the servo's position if unchanged, or move it if
  // changed
  if (OFF == 0)
  {
    /* Turn ON the LED*/
    digitalWrite(ledPin,HIGH);

    if (millis() - lastPulse >= refreshTime) {
      digitalWrite(servoPin, HIGH);   // start the pulse
      delayMicroseconds(pulseWidth);  // pulse width
      digitalWrite(servoPin, LOW);    // stop the pulse
      lastPulse = millis();           // save the time of the last pulse
    }
  }
  else
  {
     /* Turn OFF the LED*/
    digitalWrite(ledPin,LOW);

    //Stop the servo
    digitalWrite(servoPin, LOW);
  }
}
```

going on behind the scenes—the journey of the processing sketch to the bytes getting executed on the Arduino board.

The real work behind the scenes is done by the GNU C/C++ compilers, linkers and libraries for the AVR microcontroller. If you hold the Shift key when you compile your sketch, you will see that the commands - `avr-g++`, `avr-g++`, `avr-ar` and `avr-objcopy` are invoked. First, your Arduino sketch is converted to a suitable C++ file (with the .cpp extension), which then is compiled, linked and finally converted to the hex file that is uploaded to the Arduino board. You can see all these intermediate files in the /tmp/build*.tmp directory. Knowledge of this build process can enable you to bypass the IDE for your Arduino development by writing an appropriate Makefile. See the "Command-line Arduino development" article listed in Resources for an example.

Conclusion

I've described a few simple but cool things that can be done with an Arduino, but this article barely scratches the surface. A number of excellent books are available that list a great number of Arduino projects you can build for fun and profit. These are, of course, in addition to all the excellent on-line resources available. During exploring Arduino purely from the various blog posts on the Web and a trial-and-error-based approach to learning, I discovered many great projects in the Arduino ecosystem. Be sure to see the Resources for this article for some of the most interesting Arduino books, articles and projects.

Also note that wires are good, but only when you want to limit yourself to the confines of your tabletop or even your room. If you want your Arduino to be out there, decoupled from your host machine, you need to explore ways of wireless communication. Say hello to XBee modules, which allow communication with the ZigBee communication standard.

And before I end, you might face issues with erratic serial communication. Especially during extended periods of experimenting with serial communication, I found that the serial ports would remain locked or just be plainly not accessible from the host computer. My advice is to be patient. Unplug and plug back in a few times, and try killing the lock file manually. Now, you should be good to go.

Acknowledgements

Thanks to the awesome Arduino community members for their documentation and numerous other bloggers on the Web. The Arduino circuit diagrams were drawn with Fritzing (**http://fritzing.org**).■

Amit Saha is currently a PhD research student in the area of Evolutionary Algorithms and Optimization. Like his random echoes show (http://echorand.me), he has been writing on myriad Linux and open-source technologies during the past five years. Overall, he loves playing around with a bit of this and a bit of that. He welcomes comments on this article and beyond at amitsaha.in@gmail.com.

Resources

Author's Code for This Article:
https://bitbucket.org/amitksaha/articles_code/src/6bed469945fd/arduino_article

Plumbing for the Arduino: **http://www.concurrency.cc/book**

PureData (Audio Processing and Visualization): **http://puredata.info**

Firefly Experiments (bridging the gap between Grasshopper, the Arduino microcontroller, the Internet and beyond): **http://www.fireflyexperiments.com**

Arduino: **http://arduino.cc**

Processing Language: **http://processing.org**

Arduino Hardware (I/O Boards): **http://www.arduino.cc/en/Main/Hardware**

Arduino Uno: **http://arduino.cc/en/Main/ArduinoBoardUno**

Sparkfun's Starter Kit for the Arduino: **http://www.sparkfun.com/products/10174**

Installing Arduino on Linux (for different distributions): **http://www.arduino.cc/playground/Learning/Linux**

Command-Line Arduino Development: **http://shallowsky.com/software/arduino/arduino-cmdline.html**

Getting Started with Arduino by Massimo Banzi, O'Reilly Media/Make: **http://shop.oreilly.com/product/9780596155520.do**

Programming Interactivity: A Designer's Guide to Processing, Arduino, and openFrameworks by Joshua Noble, O'Reilly: **http://shop.oreilly.com/product/9780596154158.do**

Arduino Cookbook by Michael Margolis, O'Reilly: **http://shop.oreilly.com/product/0636920022244.do**

Arduino Language Reference: **http://www.arduino.cc/en/Reference/HomePage**

Tom Igoe's Physical Computing Page: **http://www.tigoe.net/pcomp**

Experimenter's Guide for Arduino: **http://ardx.org/src/guide/2/ARDX-EG-ADAF-WEB.pdf**

Arduino Tutorials (tronixstuff): **http://tronixstuff.wordpress.com**

Principia Labs, Arduino Serial Servo Control: **http://principialabs.com/arduino-serial-servo-control**

Make's Arduino Web Site: **http://blog.makezine.com/arduino**

Arduino Tutorial on Lady Ada: **http://www.ladyada.net/learn/arduino**

ROAMING MEDIA

PORTABLE MUSIC DOESN'T NEED TO BE RESTRICTED TO HEADPHONES. HERE'S A STEP-BY-STEP HOW-TO ON SETTING UP A MUSIC SYSTEM THAT FOLLOWS YOU AROUND THE HOUSE LIKE A PUPPY.

Michael Nugent

Like many of you, I store all my music digitally on a central server in my home. The problem is when I walk from room to room, my music doesn't come with me. I could carry around an iThingy or put it on my phone, but I'd rather not have to wear headphones, and no matter how awesome they are in their class, tiny phone speakers still are tiny phone speakers. Fortunately, I have a lot of computer hardware lying around from past upgrades, so it was fairly easy for me to come up with some small, older systems for each area of my house. Now, instead of listening to music on a little device, I use a device to tag my location and have the music follow me wherever I go.

The system is easy to build and uses mostly off-the-shelf open-source programs. In addition to the player and control system, you need a way of tagging your location in the house. I use the Bluetooth radio on my phone, but you also could use RFID tags, Webcams with motion detection or facial recognition, or pretty much anything else that will let the system know where you are. For this setup though, I'm assuming you're using a Bluetooth device.

The central piece to this project is a server-based music player. I am using the Music Player Dæmon (MPD), a wonderful server-based system released under the GNU General Public License and available from the repositories of most Linux distributions. Install the software with your favorite package management system. In addition to this player, you need to set up a streaming system. Icecast fulfills this requirement and also is widely available. Install it as well.

THE SYSTEM IS EASY TO BUILD AND USES MOSTLY OFF-THE-SHELF OPEN-SOURCE PROGRAMS.

Configuring MPD is fairly straightforward. The default file for your distribution is probably very similar to the example below, but you may need to change a few things. The `music_directory` entry should point to the directory that contains the music files and one `bind_to_address` should contain the non-loopback name or address of the server. If it binds only to 127.0.0.1, outside boxes may have trouble connecting to it.

The `audio_output` section defines where the music goes when it plays. In this case, you want to transcode it to Ogg format and send it to the Icecast server on the same host (although you could run these on different systems if it makes sense for your setup). Within this section, the port and password must match the configuration for the Icecast server, and the mount will define the portion of the URI after the server name. For simplicity, I've left it at /. If you start MPD after this configuration file is set up, it will be missing the Icecast socket to play to, so you need to set that up next. I've added some comments to the configuration below to help document the options there:

```
music_directory     "/data/arown/files/audio"  #Point this at
the top level of your music directory. If you have more than
one, a directory of symbolic links may help you.
playlist_directory  "/var/lib/mpd/playlists" #The following
files must be writable. I suggest making the directory owned
by the user running MPD.
db_file             "/var/lib/mpd/database"
log_file            "/var/lib/mpd/log"
state_file          "/var/lib/mpd/state"

user                "mpd" #This is the user that the MPD
```

```
program runs under. I highly suggest making this a nonroot user.
bind_to_address     "templar" #Place your machine name here, not
"localhost" or "127.0.0.1" if you want to reach the MPD
server from another machine.

bind_to_address     "/var/lib/mpd/socket"
port                "6600" #This is the client control port for
starting and stopping the MPD player as well as building
playlists and changing server side volume.
log_level           "verbose"

input {
        plugin "curl"
}

audio_output {
        type        "shout"
        encoding    "ogg" #This is the media type for the stream,
if your player wants MP3, use encoding "mp3" and be sure you
have the proper tools to transcode to MP3 on your box.
        name        "MusicPuppy"
        host        "localhost" #The name of the box
that the Icecast server is hosted on.
        port        "8000" #The port of the
Icecast server.
        mount       "/"  #This is the part
of the URI after the hostname. I have left it to "/" for
simplicity, but often things like "/music.ogg" are used
to make it clearer to the user.
        password    "passthis"  #The source password to the
Icecast server. You probably should change this to something
complex because you'll never have to type it in.
        bitrate     "128" #The bitrate to transcode to. You
may want to raise or lower this based on your CPU, bandwidth
or your quality preference.
        format      "44100:24:2"  #This is three variables:
44,100 represents the sample frequency, 24 is the bitwidth,
and 2 is the number of audio channels. Experimenting here
may give you better or worse quality, depending on your audio
hardware and number of speakers.
}
filesystem_charset "UTF-8"
```

In order to finish off the server portion, you need to configure Icecast to stream the music it receives from MPD. The sample Icecast configuration file is quite long, but I've cut out the areas that I've changed.

Make sure that the source-password here matches the password from the MPD server. This allows the MPD server authorization to stream music to Icecast:

```
<authentication>
    <!-- Sources log in with username 'source' -->
    <!-- This password must match the MPD password above -->
    <source-password>passthis</source-password>
    <!-- Relays log in username 'relay' -->
    <relay-password>passthis</relay-password>

    <!-- Admin logs in with the username given below -->
    <admin-user>admin</admin-user>
    <admin-password>passthis</admin-password>
</authentication>
```

Be sure that the bind addresses and ports are correct. The bind address can be localhost if the MPD server is on the same box. The port must match the port from the MPD configuration above:

```
<listen-socket>
    <port>8000</port>
    <bind-address>127.0.0.1</bind-address>
```

```
</listen-socket>
```

While you're in this configuration file, change the default passwords from "hackme" and "hackmemore" to something more secure, and check the location of the logs for debugging. Mine are in /var/log/icecast, but yours may vary.

After you start Icecast, the server configuration should be complete. At this point, start MPD as well. It should hook itself up to the Icecast port, and the logs should be free of errors. Next, an MPD client is needed in order to set up playlists on the server. I use MPDroid for this, an Android variant I use to control playlists from my phone, but there are clients for a wide variety of platforms available via your package manager or from **http://mpd.wikia.com/wiki/Clients**. Fire up a client, and add a few songs to a playlist for a test and tell it to play. The Icecast access log (not the MPD logs) should show that a SOURCE has connected, and it should show a 200 return. For example:

```
127.0.0.1 - - [20/Jul/2011:01:15:03 -0700] "SOURCE /
➥HTTP/1.0" 200 19 "-" "MPD" 424
```

This shows the incoming connection from MPD on the local box, the current date and time and the fact that this is a SOURCE request as opposed to a player request. It shows the directory accessed "/", the protocol used and the return code "200". This is a great source of verification that the connection between MPD and Icecast is up and working properly, though it does not tell you that any data actually is being passed. For that, you'll need to test it end to end.

To test it, you need a music player that supports Ogg format (or MP3 if you've gone that route). I've chosen Rhythmbox for this example, because it supports Ogg, is popular across a wide range of distributions and has a command-line control system that you can use to start and stop the music as well as set the server URI. Install and launch Rhythmbox (or fire up your favorite player with these features) and set it to play from the Icecast URI. For example, mine is at http://templar:8000/. If you have changed the `mount` directive in the MPD configuration file to `"/music.ogg"`, the URI would then be http://templar:8000/music.ogg. Either way, when the music player is pointed at this URI, the Icecast access.log file should show something like this:

```
mj-12 - - [01/Aug/2011:15:28:52 -0700] "GET /
➥HTTP/1.1" 200 1194382 "-" "-" 70
```

The format for this line is very similar to the one above. The only real difference is that this is a GET request instead of a SOURCE request. Seeing this line in the access logs without a corresponding error in the error log shows you that the media player is requesting the music stream from the Icecast server properly.

If you don't hear music at this time, go back over the setup and check the log files. Check the volume level within MPD, and make sure that the MPD client says a song is playing. Many MPD clients also let you stream music directly, so you can verify that it's working as well.

The next step is to set up your Bluetooth token. In my case, this is my Android phone, but pretty much any Bluetooth device will work. Ranges vary, so placement of the Bluetooth receivers is important to avoid overlap or gaps in the area you're trying to cover.

The Linux package for Bluetooth support is called Bluez. It is widely available and comes as part of most distributions. Install this package if it is not installed on your system already. You don't need to make any configuration changes, because all you need to do is identify that the Bluetooth device is in range. You don't need to pair with it or transfer data between devices. After installing the Bluez package, start the software. Your logs should show that the software started correctly and that it identified your Bluetooth hardware properly.

In order to find the Bluetooth token, it needs to be put into discoverable mode temporarily. Turn that on, and run the following scan command from the command line:

```
mike@templar:~$ hcitool scan
Scanning ...
        D4:E9:C0:37:00:0D       eris
```

Make a note of the Bluetooth ID, and be sure that the name field is not blank. It can be anything, but it has to be something. After this step, you can turn off the discoverable mode on the Bluetooth device for increased security. You now have all the information that you need.

The following script checks whether the Bluetooth device is in range and stops or starts the music player based on the result.

THE FOLLOWING SCRIPT CHECKS WHETHER THE BLUETOOTH DEVICE IS IN RANGE AND STOPS OR STARTS THE MUSIC PLAYER BASED ON THE RESULT.

Replace the `SERVER_URI` variable with your MPD/Icecast server and the BTADDR variable with your device's Bluetooth ID (this ID comes from the `hcitool scan` command above):

```
#!/bin/bash
SERVER_URI="http://templar:8000/"
BTADDR="D4:E9:C0:37:00:0D"

DBUSADDR=`grep -z DBUS_SESSION_BUS_ADDRESS /proc/*/environ 2>
 ➥/dev/null|
sed 's/DBUS/\nDBUS/g' | tail -n 1`

if [ "x$DBUSADDR" != "x" ]; then
```

```
  export $DBUSADDR
else
  echo "Cannot find DBUS Session for Rhythmbox. Please
be sure the application is running"
  exit 1
fi

NAME=`hcitool name $BTADDR`

if [ -z "$NAME" ] ; then
    `rhythmbox-client --pause`
else
    `rhythmbox-client --play-uri=$SERVER_URI`
fi
```

Save this script as /usr/local/bin/musiccontrol.sh. Next, add the script as a crontab entry that runs every minute. This entry must be run as the same user as the user that owns the Rhythmbox process.

Edit the crontab from the correct user:

```
mike@templar:~$ crontab -e
```

Add the following line, and then save and exit:

```
* * * * * /usr/local/bin/musiccontrol.sh
```

Now, turn on the Bluetooth device (it does not need to be discoverable this time because you already have the address). At the turn of the next minute, the cron script will see the Bluetooth device and then tell Rhythmbox to start playing the music from the MPD/Icecast server. If you move the Bluetooth device out of range, the cron script will no longer see the Bluetooth device and will stop the music.

Rhythmbox, Bluetooth and this cron script must be set up on every machine that you intend to play music for you. If you do it on only one box, only that box will start and stop music as you enter or leave range. If you set up the system on multiple pieces of hardware, it will transition the music for you. When moving out of the range of one server and into the range of another, the music automatically will stop in the room you were in before and start in the room you are in now.

This is just a simple setup for moving media around automatically. Multiple Bluetooth devices could be set up for different members of the house, and a priority system could be put in place. Motion detection using the "motion" package could be set up to differentiate areas of your home further with overlapping Bluetooth. You even could use facial recognition with help from the OpenCV Project. There are many places you can go from here.■

Michael Nugent has spent a good deal of his time designing large-scale solutions to fit into a tiny budget and leveraging Linux to fulfill the roles that typically would be filled by large commercial appliances. Recently, Michael has been working to design map-reduce clusters and elastic cloud systems for growing startups in the Silicon Valley area. When not building systems, he likes sailing, cooking and making things out of other things. Michael can be reached at michael@michaelnugent.org, and code and notes for this article can be found at https://github.com/michaelnugent/MusicPuppy.

ONE KEY TO RULE THEM ALL:

GRUB, USB and a Multiboot Environment

Do you have a spindle of CDs, each of which with a live environment you use for a specific purpose? Read this article to learn how to combine them all into one unified boot environment on a USB drive.

ADRIAN HANNAH

It's inevitable. If you work with Linux professionally, you will end up with a bevy of CDs or USB drives, each with their own live environment, each with a specific purpose. You likely have some haphazard system for keeping track of which device has what on it (my personal method was masking tape and pencil). But, there's no reason to keep doing this! Improvements in bootloaders (particularly GRUB) let you have live, persistent operating environments installed on USB drives as well as allowing for multiboot environments. Put this all together, and you get a single USB drive that replaces the gaggle of devices you used to have.

GRUB

GRUB, short for Grand Unified Bootloader, is the default bootloader for a vast majority of Linux distributions. The bootloader is a tiny program that is loaded into memory immediately when the computer is powered on that knows how to load data from a nonvolatile source (HDD, USB drive, optical device and so on) and, in turn, loads an operating system from the specified device.

Setting Up GRUB on USB in Linux

Thanks to the Linux community's prolific use of GRUB as the default bootloader, installing GRUB to the Master Boot Record for a USB device is incredibly easy. Assume that you want to call your new volume "Multipass" and that your target device resides at /dev/sdb1. First, you need to create the filesystem:

```
sudo mkfs.vfat -n Multipass /dev/sdb1
```

Next, you need to mount the filesystem and install GRUB:

```
mount /dev/sdb1 /mnt/
grub-install --no-floppy --root-directory=/mnt /dev/sdb
```

Now you can add .cfg files to grub.d/ in order to add/edit/delete menu items to the boot menu or customize the boot menu.

Setting Up GRUB on USB in Windows

Installing GRUB on a USB device is markedly different in an environment that doesn't natively have GRUB utilities, but it isn't impossible. Tools like GRUB4DOS and WinGRUB are available that give you access to the same tools in Windows that are available in Linux. Basically, you need to find tools to initialize the drive as a bootable drive and install GRUB to the device. PeToUSB, GRUB4DOS and grubinst are the most widely used set of tools to accomplish this. PeToUSB uses a simple GUI interface to initialize the USB device as a bootable device. grubinst is a GUI installer that installs GRUB to the USB device. The only reason you need to download GRUB4DOS is because you need to copy "grldr" to the USB device. You can find more detailed instructions on how to use these tools on-line. But, as Windows installations are outside the purview of this publication, I digress.

Adding Distributions

Once you have GRUB set up, you need to add something to boot. Download your favorite live CD or app CD, and extract it using something like the GNOME Archive Manager. Do *not* extract this to your USB drive! Instead, extract it to a temporary location. Once you have performed this procedure several times, you will notice that almost all Linux distributions have a boot directory, and the files within this directory are the kernel image and vary from distribution to distribution. Then, rename the boot directory to something apropos to the distribution (for instance, “boot-ubuntu”), and copy it to the USB device. Aside from the boot folder, there typically is another folder in these ISOs that contains the boot image for the OS (for instance, BackTrack4’s image is stored in a directory called bt4). Copy this folder to the USB device as well. You then need to add an entry in the configuration file so that this distribution will show up in the boot menu.

This is where a major schism occurs, and a little more background knowledge of GRUB is required. There are two versions of GRUB, initially referred to as GRUB and GRUB2, they now are called GRUB Legacy and GRUB, respectively (GRUB4DOS is a derivative of GRUB Legacy). There are a significant number of differences between the two, but the one that is important to us right now is that GRUB Legacy uses the menu.lst file for boot menu configuration, and GRUB uses grub.cfg. So depending on the source of the GRUB installation on your USB device, you will use a different configuration file.

Some distributions are distributed as a floppy disk image. Those are the best—all you have to do is copy the file to the root of the USB device and add a one-line entry to the config file.

Even though this takes time, you’re better off testing each distribution after installing a new distribution. That way, if you find an error, it’ll be easier to troubleshoot the problem. Trust me, installations can be finicky when you do things this way, and they can break for what seems like no good reason.

Adding a Menu Item in grub.cfg

At its most common and basic form, each menu entry will provide:

1. A user-friendly title.

2. The root filesystem.

3. A kernel image.

4. A boot image or initial ramdisk.

For example:

```
menuentry "Made Up Distro" { #user-friendly title
    set root=(hd0,1) #root filesystem
    linux /boot-madeup/vmlinuz0 #the kernel image
    initrd /madeup/initrd0.img #the boot image
}
```

Some distributions take some tweaking to get them to load properly. For instance, some live CDs need to be booted into 16-bit mode, and in that case, you would use `linux16` and `initrd16` instead. You can look in the GRUB manual on-line for all sorts of boot parameters and tweaks you can use on the menu item to make it work, but your best resource is the grub.cfg that was on the original ISO for the distribution. Some entries can become ridiculously complex in order to work right, like for Fedora:

```
linux /fedora1/isolinux/vmlinuz0 live_locale=en_US.UTF-8
➥live_keytable=us live_dir=/fedora1 root=UUID=A716-9810
➥rootfstype=auto ro liveimg quiet rhgb
➥rd_NO_LUKS rd_NO_MD noiswmd
```

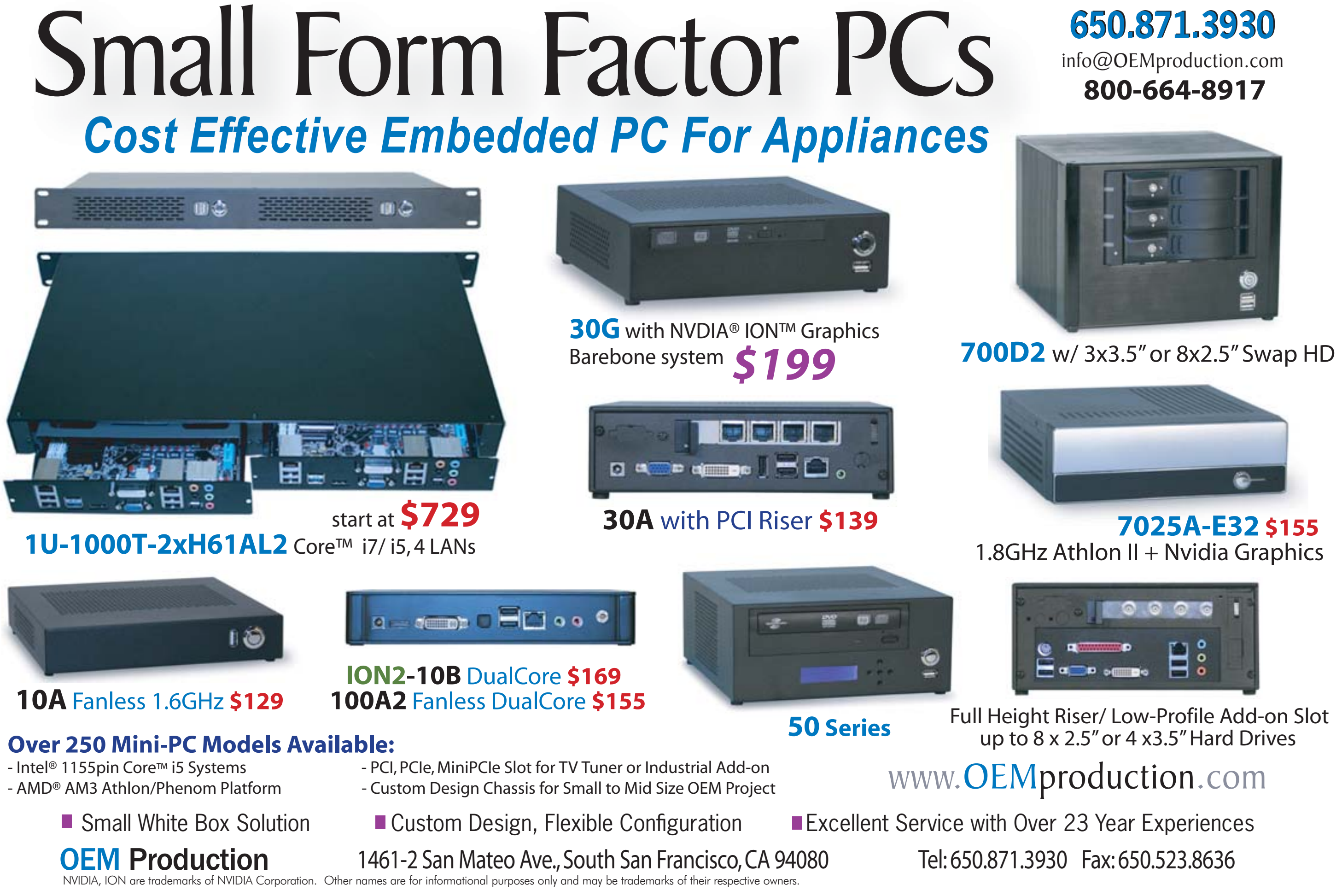
This is the actual kernel image command from my own multipass device. It is long and tedious, but it works.

Adding a Menu Item to menu.lst

The menu.lst file, with regard to menu items, is quite similar to the grub.cfg entries. You still specify the user-friendly title, the root filesystem, and the kernel and boot images, but the syntax is a little different. An entry in menu.lst would look something like this:

```
title Made Up Distro
root (hd0,1)
kernel /boot-madeup/vmlinuz0
initrd /madeup/initrd0.img
boot
```

As with the grub.cfg entries, you can use



Listing 1. Sample grub.cfg File

```
menuentry "Ubuntu Live 11.04 64bit" {
        loopback loop /boot/iso/ubuntu-11.04-desktop-amd64.iso
        linux (loop)/casper/vmlinuz boot=casper
        ➥iso-scan/filename=/boot/iso/
        ➥ubuntu-11.04-desktop-amd64.iso noeject noprompt--
        initrd (loop)/casper/initrd.lz
}

menuentry "Ubuntu Live 9.10 32bit" {
 loopback loop /boot/iso/ubuntu-9.10-desktop-i386.iso
 linux (loop)/casper/vmlinuz boot=casper
iso-scan/filename=/boot/iso/ubuntu-9.10-desktop-i386.iso
 ➥noeject noprompt--
 initrd (loop)/casper/initrd.lz
}

menuentry "Ubuntu Live 9.10 64bit" {
 loopback loop /boot/iso/ubuntu-9.10-desktop-amd64.iso
 linux (loop)/casper/vmlinuz boot=casper
iso-scan/filename=/boot/iso/ubuntu-9.10-desktop-amd64.iso
 ➥noeject noprompt--
 initrd (loop)/casper/initrd.lz
}

menuentry "Grml small 2009.10" {
  loopback loop /boot/iso/grml-small_2009.10.iso
  linux (loop)/boot/grmlsmall/linux26 findiso=/boot/
➥iso/grml-small_2009.10.iso apm=power-off lang=us vga=791
➥boot=live nomce noeject noprompt --
  initrd (loop)/boot/grmlsmall/initrd.gz
}

menuentry "tinycore" {
 loopback loop /boot/iso/tinycore_2.3.1.iso
 linux (loop)/boot/bzImage --
 initrd (loop)/boot/tinycore.gz
}

menuentry "Netinstall 32 preseed" {
  loopback loop /boot/iso/mini.iso
  linux (loop)/linux auto
url=http://www.panticz.de/pxe/preseed/preseed.seed locale=en_US
➥console-setup/layoutcode=de
netcfg/choose_interface=eth0 debconf/priority=critical --
  initrd (loop)/initrd.gz
}

menuentry "debian-installer-amd64.iso" {
  loopback loop /boot/iso/debian-installer-amd64.iso
  linux (loop)/linux vga=normal --
  initrd (loop)/initrd.gz
}

menuentry "BackTrack 4" {
 linux /boot/bt4/boot/vmlinuz BOOT=casper boot=casper
 ➥nopersistent rw vga=0x317 --
 initrd /boot/bt4/boot/initrd.gz
}

menuentry "Memory test (memtest86+)" {
 linux16 /boot/img/memtest86+.bin
}

menuentry "BackTrack ERR" {
 loopback loop /boot/iso/bt4-pre-final.iso
 linux (loop)/boot/vmlinuz find_iso/filename=/boot/iso/
 ➥bt4-pre-final.iso BOOT=casper boot=casper
 ➥nopersistent rw vga=0x317--
 initrd (loop)/boot/initrd.gz
}

menuentry "XBMC ERR" {
 loopback loop /boot/iso/XBMCLive.iso
 linux (loop)/vmlinuz boot=cd isofrom=/dev/sda1/boot/iso/
 ➥XBMCLive.iso xbmc=nvidia,nodiskmount,tempfs,setvolume
 ➥loglevel=0 --
 initrd (loop)/initrd0.img
}

menuentry "netboot.me" {
 loopback loop /boot/iso/netbootme.iso
 linux16 (loop)/GPXE.KRN
}

menuentry "debian installer amd64 netboot XEN pressed" {
  linux /boot/debian/linux auto
preseed/url=http://www.panticz.de/pxe/preseed/xen.seed
 ➥locale=en_US console-setup/layoutcode=de netcfg/
 ➥choose_interface=eth0 debconf/priority=critical --
  initrd /boot/debian/initrd.gz
}
```

many more commands than you can use in menu.lst, and you can make a lot of tweaks to make a nonfunctional entry suddenly work.

ISO Loopback

It all seems pretty daunting, no? Don't you wish there were a way to dump all your ISO files onto one disk and then mount them all? Your wish is granted! With GRUB2, we were introduced to the loopback boot option. This allows you to use GRUB to mount an ISO file and boot from it as if it were a physical piece of media. All you have to do for the menu entries is add a `loopback` command and then adjust the `linux` and `initrd` commands:

```
menuentry "Made Up Distro" { #user-friendly title
    set root=(hd0,1) #root filesystem
    loopback loop "/madeup.iso"
    linux (loop)/boot/vmlinuz0 #the kernel image
    initrd (loop)/initrd0.img #the boot image
}
```

GRUB then mounts the ISO file and boots accordingly. Note that you still may need to add some arguments to either command to get it to boot properly. Unfortunately, this doesn't work for all ISO files. If you can't get the loopback to work for an ISO, you have to load it the old-fashioned way as described above.

Customizing the Boot Menu

What good would a fancy tool like this be if you couldn't customize it to your liking? The default boot menu is white text on black background, which, let's face it, is *boring*. In your config file, you can specify a background image or color, text color, highlight color, alternate text for your menu entries, and the list goes on.

Set the Background Image

First, if you are using GRUB4DOS, the background picture has to be converted to 640x480 and 14-color, so you need to make sure that whatever you're using will look good under those constraints. GRUB2 has a bit more-advanced rendering capability, and it can handle JPG-, TGA- and PNG-formatted pictures as long as they are in RGB mode and not indexed mode.

The basic process for doing this is to scale and crop the image to your liking, limit the number of colors in the image palette to 14, save the image as an xpm file, and gzip it. For a more in-depth explanation of this process, I will use GIMP, although there are a number of other ways to do this.

After opening the image in GIMP, use a combination of scaling the image (Image→Scale Image) and the crop tool to make the image 640x480. To drop the number of palette colors in the image, click Image→Mode→Indexed, and choose "Generate optimum palette" with 14 colors. That's all there is to it. You can do other things to the image to make it look better in 14 colors:

- Filters→Blur→Selective Gaussian Blur: this smooths out the image without losing detail.

- Colors→Posterize: this turns subtle differences in color into a large field of a single color. This is done smoothly, which makes the image look better in 14 colors.

- Colors→Levels→Auto: this increases the contrast of the image, lessening the number of colors used in the image.

- Colors→Map→Gradient Map: maps the existing colors in your image into a gradient, essentially minimizing the colors used.

Once you've got your image the way you want it, save it as an XPM image file, then gzip the XPM file. You'll end up with a file named something like image.xpm.gz. Move this file to your USB device.

If you are running GRUB4DOS, look in menu.lst for a line something similar to this:

```
splashimage splash.xpm.gz
```

If you find it, replace the splash.xpm.gz with the path and filename of your file. Otherwise, add the line.

If you are running GRUB2, open grub.cfg and find a line that looks something like this:

```
GRUB_BACKGROUND=splash.xpm.gz
```

If it exists, replace the image name with the path and filename for your image. Otherwise, add the line, then run:

```
update-grub
```

Set the Menu and Highlight Color

You also can adjust the color of the menu text itself. You can adjust two separate values: the normal color (consisting of most of the text and the menu border) and the highlight color (the colors for the item currently selected in the menu list). Each of these values consists of two colors: the foreground and the background. Colors you can choose from include black, dark-gray, light-gray, white, brown, yellow, red, light-red, blue, light-blue, green, light-green, cyan, light-cyan, magenta and light-magenta. If black is chosen as the background color, it will show as transparent.

The command to adjust the colors in menu.lst for GRUB4DOS is `color`, followed by the normal color and then the highlight color. Specify the foreground and background colors by using a forward slash. For example:

```
color white/black white/blue
```

The above signifies white text on a transparent background, and the highlighted text is white on a blue background.

GRUB2 uses two variables in grub.cfg to accomplish the same thing. These variables are `COLOR_NORMAL` and `COLOR_HIGHLIGHT`. Again, use the forward slash to separate foreground color from background color:

```
COLOR_NORMAL="white/black"
COLOR_HIGHLIGHT="white/blue"
```

Change the Text Font

As far as I've been able to discern, you can change the font used in the menu only if you are using GRUB2. To accomplish this, you need to utilize one of the many nifty tools that come with grub: `grub-mkfont`. Assuming you already have the font and you know where the font file resides, you simply need to run `grub-mkfont`, and specify the location of the font file, the destination for the grub font file and the size you want the text to be in points. For example:

```
sudo grub-mkfont --output=/path/to/usb_device/comicsans.pf2 --size=14

/usr/share/fonts/comicsans.ttf
```

Then, in your grub.cfg file, add the variable GRUB_FONT and set it to the path and name of the grub font file you just created:

```
GRUB_FONT=/path/to/usb_device/comicsans.pf2
```

Applications to Automate the Process

Take everything I've taught you up to this point and throw it out the window! There are a number of different applications for all the major operating systems that easily will replicate the previously outlined procedure with little to no fuss. YUMI and SARDU are the top picks for generating a multipass in Windows, and MultiSystem is the tool of choice in Linux. Each of these applications has a handy GUI that walks you through adding all the images you want and customizing the menu to your heart's content.

I was first introduced to the concept of a multiboot USB device several years ago when the only way to set it up was manually with a lot of guessing and checking. I don't mind telling you, it was a royal pain to get anything working, let alone be elegant. As time goes by, utilities to help you set them up have emerged, and they've gotten better. What took me several hours to set up two years ago, took me only about 15 minutes this time around (and that includes the time to download the ISOs).■

Adrian Hannah has spent the last 15 years bashing keyboards to make computers do what he tells them. He currently is working as a system administrator for the federal government. He is a jack of all trades and a master of none. Find out more at http://about.me/adrianhannah.

Resources

Windows Resources for GRUB: **http://sourceforge.net/projects/grub4dos/files** and **http://www.gocoding.com/page.php?al=petousb**

Multisystem (original site in French): **http://liveusb.info/dotclear**

Multisystem (via Pendrivelinux.com): **http://www.pendrivelinux.com/multiboot-create-a-multiboot-usb-from-linux**

YUMI: **http://www.pendrivelinux.com/yumi-multiboot-usb-creator**

SARDU: **http://www.sarducd.it**

OLPC: Are We There Yet?

An update on the One Laptop Per Child Project. SAMEER VERMA

Just the other day, I ran into someone in the parking lot, who looked at the silly-looking green laptop slung across my shoulder and said "Didn't the founder of that project ditch and disappear?" News travels fast, no matter where it comes from, and no matter how wrong it is. I'm amused.

Of course, I am talking about the One Laptop per Child (OLPC) Project, founded by Professor Nicholas Negroponte in 2005. Conceived as a learning project at MIT back in the early 1980s (with Professor Seymour Papert), it morphed into a laptop project in Cambodia in 1999 and eventually into the current OLPC Project in 2005. OLPC currently has more than two million laptops in the hands of children in more than 25 languages in more than 40 countries.

I got involved in July 2007, when I first saw the XO laptop at OSCON (thanks to Rob Savoye) and instantly fell in love with it. I quickly signed up with the developer program and got myself a beta machine. However, unlike my short-lived affairs with other gadgets, I've hung onto this one for more than four years now. Along the way, I became the de facto organizer for the OLPC San Francisco volunteer community. OLPC also has found a spot on my research agenda. I now run projects in Jamaica, India and Tuva (yes, that Tuva, of Richard Feynman fame), and I help out with others in Armenia, Madagascar, Morocco and a whole bunch of other places. Making technology work in remote places has been a welcome challenge. Making it work for children the world over has been a fulfilling experience.

Hardware

Speaking of laptops, the lean, green children's machine is quite the icon. It gathers crowds with little effort. It brings forth the inner child in many a tough adult. Let's take a peek into what's under the plastic. Table 1 shows the different incarnations of the OLPC XO laptop.

The XO has many cool features. It is rugged, solar-chargeable, has a sunlight-readable screen, supports mesh networking and has a very aggressive suspend-resume cycle, but the importance of these features supersedes their coolness when deployed in the field. One of my projects is in Bhagmalpur,

Table 1. XO Laptop Hardware Evolution

MACHINE	CPU	RAM	STORAGE	FORM FACTOR
XO-1	AMD Geode LX 700 (x86) at 433MHz	256MB	1024MB NAND Flash JFFS2 filesystem	Laptop (green, with membrane keyboard
XO-1	AMD Geode LX 700 (x86) at 433 MHz	256MB	2048MB NAND Flash JFFS2 filesystem	Laptop (red, with membrane keyboard)
XO-1.5	VIA C7-M (x86) at 1GHz (variable)	1024MB	4GB microSD Ext4 filesystem	Laptop (green, with membrane keyboard)
XO-1.5HS	VIA C7-M (x86) at 1GHz (variable)	1024MB	4GB microSD Ext4 filesystem	Laptop (blue, with Netbook keyboard)
XO-1.75	Marvell Armada 610 (ARM) SoC at 1GHz (variable)	512MB	Undecided (still in testing)	Laptop
XO-3	ARM (proposed)	Undecided	Undecided	Tablet

Figure 1. Red-Green-Blue Spectrum of XOs

a rural village in northern India. Bhagmalpur is close to my heart. A part of my family lives there, and keeping up with the village helps me stay grounded in their realities. I have learned much from them (I can milk a cow!), and I am glad to be able to give back. Bhagmalpur is beautiful, but it has its own problems. Electricity is elusive at best. There's no Wi-Fi. Internet access rarely works, and when it does, it trickles. There are no classrooms; the kids sit under a tree. This is where we see the real benefits of sunlight-readable screens, solar chargers, under-a-tree mesh-networking technology and aggressive suspend-resume cycles that conserve every fraction of a watt. All of this made possible by a combination of free and open-source software.

Internet in a Box

Another neat setup we have in Bhagmalpur is an off-line server. Remember SneakerNet from the days of tapes and floppies? SneakerNet is a wonderful concept. You can think of it as a kind of mirroring with a very high latency. Given that about 70% of the world's population still is not on the public Internet, SneakerNet approaches become important stopgap measures. OLPC has a Fedora-based image (called the XS) that is designed to run an off-line server at the local school, village or library. Think of it as an "Internet in a box". Yes, it's not the entire Internet, but storage is cheap these days. You can pack plenty of content on a 2TB drive, be it Khan Academy videos, TED Talks, books from Project Gutenberg, the Internet Archive, Wikipedia and so on. This approach acts as a stopgap, so that we can get the kids access to some information at least for now. Eventually, when the Internet gets there 24/7, this server can continue to function as a conduit to the outside.

The school server runs on x86 machines (ours in Bhagmalpur draws 8 watts) and provides a plethora of services. The design of this server revolves around Moodle, a learning management system. Some of the administrative capabilities also are plugged in to Moodle, so teachers don't have to drop to a command line to administer the server. Other services that run on the XS are Apache, PHP and PostgreSQL (to run Moodle), ejabberd (to provide XMPP-based collaborative services), Squid, DHCP, DNS and a few others. In the Bhagmalpur setup, we also use Munin to keep an eye on server health and OpenVPN to tunnel back to San Francisco to pull logs and to push content via rsync. The server not only serves up content and courses, but it also seamlessly backs up each laptop's datastore, pushes both OS and application updates, manages theft deterrence, network access and more.

The XS is designed to follow a curricular model, complete with courses, teachers, students, enrollment and assessment all managed via Moodle. However, in some instances, a curricular model isn't suitable. Sometimes we simply don't have a functional school in the neighborhood. In such cases, a reference library is more apt. There's a parallel effort to fill the need for an off-line digital library, but without a specific curricular underpinning. Pathagar (in Bengali, path=lesson, agar=repository) is a Django-based digital library that serves up an XML catalog of content using the Open Publication Distribution System (OPDS). Content can be searched and downloaded using a browser or custom reader. Uruguay has been using Pathagar in production for some time now. We are testing it for Bhagmalpur (using

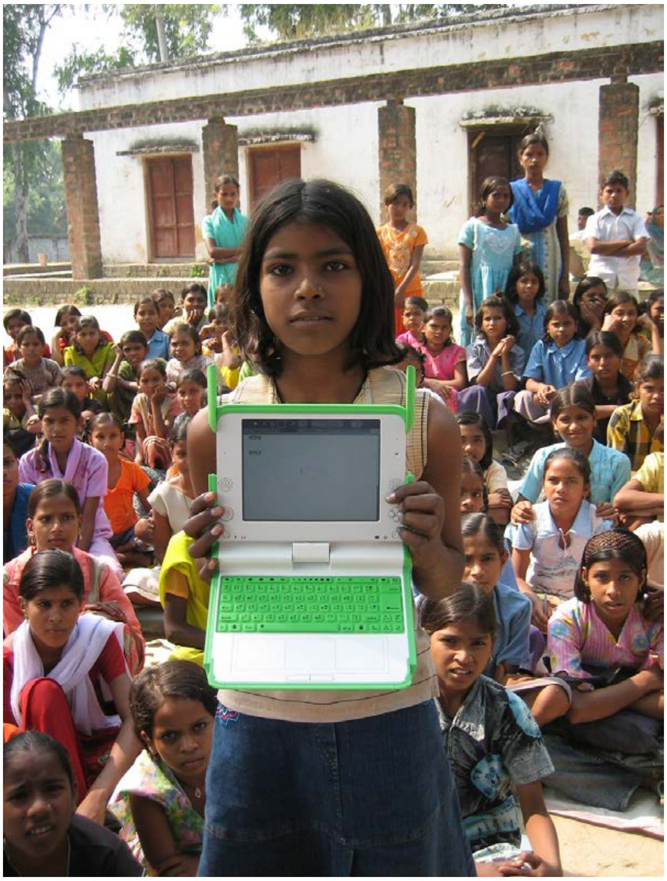


Figure 2. Garima in Bhagmalpur

Debian Squeeze on a SheevaPlug) right now and hope to put it into production soon.

Deployments

OLPC projects come in all sizes. Projects that implement laptops in the field are called deployments. There are large, top-down OLPC-managed deployments and small third-party deployments managed by NGOs and volunteer communities. As Professor Negroponte puts it, uppercase OLPC is the "official" effort, whereas lowercase olpc is the community effort. Of course, one without the other wouldn't get very far. Uppercase deployments, like Peru, are the largest with almost a million machines. Uruguay follows closely with 100% saturation at the primary school level, and Rwanda recently crossed the 100,000 laptop mark. In the lowercase olpc realm there are several projects with much less volume, but nevertheless, a strong intent. Some are in the hundreds, like my projects with the University of the West Indies in Jamaica. Others are in the tens, like the effort in Bhagmalpur. We even have some efforts with a single eBay-bought XO laptop lighting up someone's world somewhere. It's a spark that holds the potential to ignite.

Keeping track of smaller deployments is difficult. We have two planet feeds to keep track of various deployments (see Resources). We also have an effort to map the community at large—who they are, where they live, what they do and whom they work with. olpcMAP.net maps out deployments, projects and volunteers from Armenia to Zambia, with language translation efforts from Amazigh to Zulu.

Kids Like Sugar

Looking past the hardware, one of the key features of OLPC deployments is its software—more precisely, its software as a learning platform. Sugar is a learning platform designed for children. It originally was designed for the OLPC XO laptop, but since then it has grown to run on most Netbooks and PCs. Sugar is used by more than two million students in Peru, Uruguay, Rwanda, Nepal, the United States and more than 40 other countries.

Sugar is distributed as free software, released under GPL v2. It is available in more than 25 languages, although the translation infrastructure has 132 languages in various stages of completion. Sugar is a collaborative learning platform that promotes collaborative learning. Sugar also can be viewed as a graphical user interface and a collection of software applications (called Activities) that give children a rich environment for learning.

The Sugar user interface is very simple. It has a "low floor", so that even children as young as ages two and three can start to use it. At the same time, it puts no limit on what they can achieve. Children can use Sugar to reach very complex ideas. They are not bound by its simplicity; rather, they use its simplicity as a catalyst for growth. Sugar presents no "ceiling" to the learner. In fact, each application ships with its source code (usually Python) and can be viewed and modified in much the same way as with HTML.

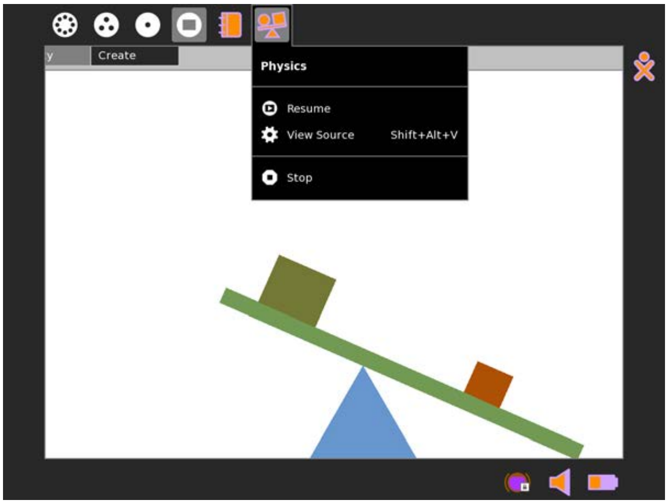


Figure 3. Sugar View Source Option

Sugar has built-in mechanisms for collaboration using an XMPP-based layer that allows for applications to share data across the network. For instance, the Write Activity (based on AbiWord) can "talk" to another child's Write Activity over the network and allow for real-time, collaborative word processing. This is very much like what you may see with Google Docs, but without Google in the middle. It's agnostic of the network layer, so the collaboration can work between two laptops meshed together under a tree or two laptops connected over the Internet reaching across continents.

Sugar maintains a journal, a diary of sorts, of everything a child does. It is a record of both what things learners make and how they made them. This journal runs on top of an XML datastore written in Python and works as a flat, time-based listing, sortable by time, topic, mime-type and so on, and it allows children to resume work where they left off. The journal also allows for reflecting upon one's work by taking notes once each activity is done. Looking at these notes over time should give students a sense of how their work has evolved. In all, Sugar is not about instruction; it is about learning.

Sugar Activities come in many flavors. Some, such as the Browse activity (based on Firefox) rely on the network extensively, while others, such as the Wikipedia bundle, are completely off-line. Then, there's the physical computing kind of activity. Activities like Scratch and Turtle Art interface with the 3-D real world using Arduino boards and LEGO robots. Taking a cue from the Uruguayans who are building Arduino-based XO robots and the Peruvians who are distributing 92,000 LEGO robotics kits to their schools, we have started exploring the use of LEGO WeDo robotics kits in Jamaica. The charge is being led by a ten-year-old Jamaican kid!

Then there is the Measure activity, an oscilloscope of sorts, reading from the microphone port for sensor data. How many overtones do throat singers of Tuva produce? Measure can tell you—think lemon batteries, conductive playdough and electromagnetic

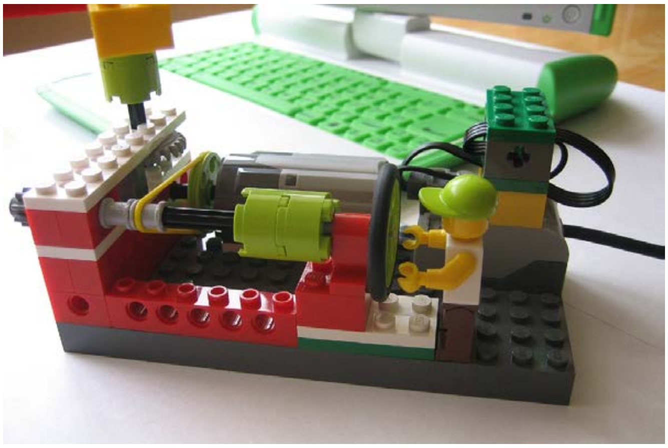


Figure 4. Scratch and LEGO WeDo Robot

speedometers. Now we are talking real education! The list goes on, but you can explore all of these on the Sugar Activities repository.

On the technical side, all of OLPC's XO images are Fedora-based (Fedora 14 at the time of this writing). Sugar runs on top of Fedora (OLPC also packs a GNOME option for older kids). Sugar is primarily written in Python, using PyGTK extensively. Sugar uses an interesting security spec called Bitfrost. Creating Activities in Sugar is easy once you understand Sugar's human interface guidelines. Many existing GTK apps can be "sugarized" easily as well. Although those details are out of the scope of this article, a good place to start is James Simmons' book *Make Your Own Sugar Activities!* (see Resources).

Sugar Labs is the current home of the Sugar Project. Started as a spin-off from OLPC, it is headed by Walter Bender as its Executive Director. Sugar Labs is a volunteer-driven, nonprofit organization, and a member project of the Software Freedom Conservancy. It coordinates volunteers—an international community of teachers, software developers, artists, writers, parents and children—who are passionate about providing educational opportunities to children through the Sugar Learning Platform. Children as young as 12 years old have written Sugar Activities. When a 12-year-old developer walked into one of our OLPC San Francisco meetings, we thought he was lost and looking for his parents!

Conclusion

In all, the project is alive and well. Every single laptop out there counts as one more platform of information—both for access and for creation. Young minds all over are starting to get opportunities to connect the dots with the rest of the world. These are not cheap laptops. These are little Linux-powered revolution machines, and the revolution is on. So, come and join us. Bring meaningful and participatory education to millions. Stand up and make your mark for those who cannot. Help them solve their own problems. Start tiny sparks, so that some day, we may light up the world.

Before I wrap up, I should set the record straight. I have been approached by several people who ask, "Didn't OLPC switch over to Windows?" This myth also is a popular point of contention on sites like Slashdot to which John Waltington (OLPC's VP for Hardware Engineering) replied: "Linux has shipped, and will continue to ship, on *every* XO produced. You can believe random commenters, or you can believe the person who signs off on every SKU produced." Two million+ laptops *all* shipped with Linux. There you have it, straight from the man.

Acknowledgements

I would like to acknowledge the input and suggestions from Walter Bender, Christoph Derndorfer, Adam Holt and Mike Lee.■

Sameer Verma is a professor of Information Systems in the College of Business at San Francisco State University in San Francisco, California. He is also the founder and chief organizer of the OLPC San Francisco volunteer community. All of his computers run Linux—even the one in his car. Track him down at http://verma.sfsu.edu.

Resources

One Laptop Per Child: **http://laptop.org**

Sugar Labs: **http://sugarlabs.org**

OLPC Planet: **http://planet.laptop.org**

Sugar Labs Planet: **http://planet.sugarlabs.org**

Sugar Translation: **http://translate.sugarlabs.org**

Sugar Activities Repository: **http://activities.sugarlabs.org**

OLPC San Francisco: **http://olpcsf.org**

Bhagmalpur Project: **http://bhagmalpur.wordpress.com**

Jamaica Projects: **http://olpcjamaica.org.jm**

Tuva Project: **http://olpctuva.wordpress.com**

Scratch and LEGO WeDo: **http://info.scratch.mit.edu/WeDo**

James Simmons' *Make Your Own Sugar Activities!*: **http://en.flossmanuals.net/make-your-own-sugar-activities**

John Waltington's Comment on Slashdot: **http://hardware.slashdot.org/comments.pl?sid=1941464&cid=34810832**

Creating a vDSO: the Colonel's Other Chicken

vDSO, or virtual dynamic shared object, is a clever feature of the Linux kernel used to speed up certain system calls and access data in separate memory segments efficiently. MATT DAVIS

A vDSO (virtual dynamic shared object) is an alternative to the somewhat cycle-expensive system call interface that the GNU/Linux kernel provides. But, before I explain how to cook up your own vDSO, in this brief jaunt down operating system lane, I cover some basics of vDSOs, what they are and why they are useful. The main purpose of this article is to illustrate how to add a custom vDSO to a Linux kernel and then how to use the fruits of your labor. This is not intended to be a vDSO 101; if you would like more in-depth information, see the links in the Resources section of this article.

vDSO Basics

The traditional mechanism of communication between userland applications and the kernel is something called a system call. Syscalls are implemented as software interrupts providing the userland application with some kernel functionality. For instance, `gettimeofday()` and `fork()` are both system calls. The reason syscalls exist is due to the fact that the Linux kernel is divided into two primary segments of memory: userland and kernel land. Userland is where common programs, including dæmons and servers, execute. Kernel land is where the kernel schedules processes and does all of its nifty kernel-specific magic. This division in memory acts as a safety barrier between user applications and the kernel. The only way a user application even can touch the kernel is via system call communication. Therefore, the robustness and integrity of the kernel is protected by the limited set of routines it provides userland access to, the system calls.

To accomplish a syscall, the kernel must flip-flop memory contexts: storing the userland CPU registers, looking up the syscall in the interrupt vector of syscalls (the syscall vector is initialized at boot time) and then processing the syscall. Once the syscall has been processed in kernel land, the kernel must restore the registers from the previously stored userland context. This completes the syscall; however, as you can imagine, this is not a tax-free series of events. Numerous cycles are spun just to make these special kinds of function calls.

Although this segmentation sounds great for the security world, it does not always provide the most efficient means of communication. Certain functions that do not write any data and merely return a value stored in the kernel, such as `gettimeofday()`, are relatively safe in nature and provide no threat to the kernel from the requesting userland application. Wouldn't it be nice if you could make safe functions not have to do the memory-barrier tango? Well, you can—with vDSO!

You're probably wondering how a vDSO gets placed into a program in the first place, over the traditional syscall. Well, vDSO hooks are provided via the glibc library. The linker will link in the glibc vDSO functionality, provided that such a routine has an accompanying vDSO version, such as `gettimeofday()`. When your program executes, if your kernel does not have vDSO support, a traditional syscall will be made. This test of vDSO functionality is provided by the code linked from glibc. Of course, you don't want to hack up glibc just so you can have your home-brewed vDSO run. The method for creating a vDSO described below does not require modification of glibc; instead it relies on hacking up the kernel, as expected.

Cluck, Cluck...vDSO

These safe syscalls can be implemented on a page of virtual memory that can be mapped into each running process' memory. This implementation is similar to how other dynamically shared objects are mapped into a process, such as shared libraries. In fact, if you were to extract the page from memory and disassemble it, the result is a shared-library ELF. In other words, the vDSO is just a shared library (sorry to blow the magic for you). With this page of safe syscall routines resident to the userland application, a program can make the call and not have to endure the overhead of the memory-hopping between user and kernel segments that a traditional syscall would require. One perfect example is `gettimeofday()`. This routine not only is timing-sensitive, but it often is a routine that is used at a high frequency. Consider that it takes the kernel time to hop memory segments. Once the clock is sampled, cycles must be spent to flip memory segments. The longer this takes, the less accurate the returned time value will be.

Let's Get Frying'

Enough with theory and all that mumbo-jumbo, let's get to what this article is all about—making your own vDSO. This article assumes a 64-bit x86 processor using the 2.6.37 Linux kernel. You'll probably be surprised at how easy this is. It is even less involved than making a traditional syscall. The confusing part comes when trying to share data via variables between kernel and userland.

Let's create a syscall that does something basic—say, produce an integer value of, oh, the number of the beast, 666. For all instructive purposes, let's call this function, number_of_the_beast(). Because I'm not sure that the true number of the beast is static (hey, beasts might change), let's make this function do just that, tell us the number of the beast. (It could be like a president and change every few years.) Create a file in linux-2.6.37/arch/x86/vdso/ called vnumber_of_the_beast.c, and inside there, define your function:

```
#include <asm/linkage.h>

notrace int __vdso_number_of_the_beast(void)
{
    return 0xDEAD - 56339;
}
```

The only interesting/unusual thing here is the `notrace` macro. It is defined in linux-2.6.37/arch/x86/include/asm/linkage.h as being:

```
#define notrace __attribute__((no_instrument_function))
```

The above GNU extension tells the gcc compiler that when it compiles the function to exclude hooks supporting profiling feedback. Profiling feedback can be built in, if the `notrace` macro is removed and if the gcc flag `-finstrument-functions` was passed to the gcc at compile time (see the GCC Manual, listed in Resources).

You also need to tell the compiler to link a userland-accessible function called `number_of_the_beast`, which is also a weak symbol. Weak symbols represent data, such as function calls, that do not resolve until runtime. The word "weak" simply means the symbol can be overridden. If the symbol does not exist, no warnings are issued, as no symbol is acceptable in this case. The alias associates the local `__vdso_number_of_the_beast` to the world-accessible version, `number_of_the_beast`. Add the following piece just after the function previously added:

```
int number_of_the_beast(void)
    __attribute__((weak, alias("__vdso_number_of_the_beast")));
```

Now, you just need to toss in some pieces to the linker script so that when the kernel builds, your code will get built and linked into the `vdso.so` shared object. That is what you will use for your hook when writing code that uses the vDSO. Now, bust out your text editor and modify linux-2.6.37/arch/x86/vdso/vdso.lds.S to add the function names you just added:

```
VERSION {
    LINUX_2.6 {
        global:
            clock_gettime;
            __vdso_clock_gettime;
            gettimeofday;
            __vdso_gettimeofday;
            getcpu;
            __vdso_getcpu;

            /* ADD YOUR VDSO STUFF HERE */
            number_of_the_beast;
            __vdso_number_of_the_beast;
        local: *;
    };
}
```

One more thing, you need to tell the compiler actually to compile the information in vnumber_of_the_beast.c. To do this, just toss some information into the Makefile located in linux-2.6.37/arch/x86/vdso/Makefile. Add the name of the file, with a .o instead of a .c extension. And, through make wizardry and black magic, it will be compiled at compile time. Again, break out the text editor, and add the name to the list of object files for the variable vobjs-y. Your result should look something similar to the following:

```
# files to link into the vdso
vobjs-y := vdso-note.o vclock_gettime.o vgetcpu.o
 ➥vvar.o vnumber_of_the_beast.o
```

And Now Some Special Sauce

If the vDSO is operating in userland, how do you access kernel-land variables? After all, if vDSOs are supposed to provide kernel information, don't they have to trip the userland/kernel-land memory segment? And, wouldn't that flip-flopping of memory context render a vDSO useless? Well, it all depends how the userland version, the vDSO version, accesses the kernel data. For gettimeofday(), a special time variable is mapped into memory where the kernel updates it and the userland (vDSO version) can read it. The kernel merely copies what it knows about time into that variable, and when a vDSO is made, that call just reads the information saving the overhead of crossing memory segments. The addition or access of a kernel variable is fairly involved as compared to a basic vDSO function, but because the purpose of a vDSO is to access kernel information, such as that provided in variables, I probably should give a quick overview of doing that.

For illustrative purposes, let's add a value that lives in kernel land but is read from userland. Sure, I said earlier that this mystical number might change and you should implement a function to return it. Well, you have a function, but all you know now is the value and not what it might change to in the future. Let's make the function return a value, nonconstant. Wow, this use case is becoming really unusual. To elaborate, let's update this variable as the kernel requests. The kernel will update the vDSO variables in the update_vsyscall() function located in linux-2.6.37/arch/x86/kernel.

If you were to declare it const int vnotb = 666;, the value captured there would not be set (more on this later).

Let's define the value to be, in fact, the mysterious number of the beast itself, which I will call vnotb. This number will reside in kernel land, as so many other useful values, such as time, which the efficient gettimeofday() vDSO will obtain. This is where the true magic of vDSOs lie.

Let's remain in linux-2.6.37/arch/x86/vdso and modify all the goodies here. First, declare the variable via the VEXTERN() macro. In vextern.h, add your declaration alongside all the other declarations:

```
VEXTERN(vnotb)
```

This macro will create a variable that is a pointer to the value you care about and is prefixed with `vdso_`. In essence, you have declared vnotb as `int *vdso_vnotb;`.

vextern.h mentions that:

> Any kernel variables used in the vDSO must be exported in the main kernel's vmlinux.lds.S/vsyscall.h/proper__section and put into vextern.h and be referenced as a pointer with vdso prefix. The main kernel later fills in the values (comment in linux-2.6.37/arch/x86/vdso/vextern.h).

Now that you have some of the vDSO code in place, the userland stuff and the kernel-userland mapping, let's make use of it. In the function `vget_number_of_the_beast()`, let's return the value:

```
notrace int __vdso_number_of_the_beast(void)
{
    return *vdso_vnotb;
}
```

Don't forget to add the header that declares that value, `vextern.h` as well as an additional header that will resolve some data referenced by the latter, `vgtod.h`:

```
#include <asm/vgtod.h>
#include "vextern.h"
```

To wrap things up, you need to let the kernel know about this variable so it can pump data into it. You need the kernel to give userland a value. Well, you have it mapped at the address specified above, but that is rather pointless, unless Mr Sanders, the colonel, doesn't push some data into it. You need to go up one directory (yes, this isn't the most trivial of processes). Hop into linux-2.6.37/arch/x86/kernel. You need to let the linker know of this value, so it can map between kernel and userland, so you probably should rock that. Modify vmlinux.lds.S, and add the following after the `vgetcpu_mode` piece (note that adding it after or before `vgetcpu_mode` isn't necessary, but it's an easy place to find things):

```
.vnotb : AT(VLOAD(.vnotb)) {
        *(.vnotb)
}
vnotb = VVIRT(.vnotb);
```

This links the `vnotb` symbol with the variable `vnotb`. This sets up the variable in the address space for kernel land to access and write to. The macros above, `AT`, `VLOAD` and `VVIRT` deal with modifying addresses so that the proper piece of data, at the `vnotb`, is referenced.

Now, you need to declare the value that the kernel land will write to. In linux-2.6.37/arch/x86/include/asm/vsyscall.h declare this puppy and its section that will be inserted via the above linker script entry you most recently added:

```
#define __section_vnotb __attribute__ ((unused,
➥__section__ (".vnotb"), aligned(16)))
```

In this file, as mentioned, you also will declare the kernel-land variable to which the kernel will write. To keep things slightly more readable, associate your variable next to the `vgetcpu_mode` declaration:

```
extern int vnotb;
```

You also will define a value the kernel can read (I don't use this in my example, but if the kernel needs to read the value, this is the variable to read):

```
extern int __vnotb;
```

Now let's put this stuff in code and give it a value. The kernel will write the value via the writable `vnotb`, and you also can read it from the shared memory between kernel and userland via `__vnotb`. You will write the value in the kernel-land version of the variable, which is writable. In linux-2.6.37/arch/x86/kernel/vsyscall_64.c, preferably after all of the #include headers and just after the piece: `int __vgetcpu_mode __section_vgetcpu_mode;`, add the following:

```
int __vnotb __section_vnotb;
```

Remember, you did a trick with the linker setting the value. If you set the value globally, as you would for an extern, you would not get a value, the linker would override it. You need to set this value at runtime and not statically at compile time. To set this value as the kernel updates, modify the update_vsyscall() routine in linux-2.6.37/arch/x86/kernel/vsyscall_64.c with:

```
vnotb = 666;
```

This statement is defining the value declared previously in vsyscall.h.

Compiling, Linking and Running

Wait, is that all there is to adding a vDSO? Um, yes. Of course, if the function was something supported by the C library (glibc, in our case), you can hack that to do the detection of vDSO and then the actual call. However, I mentioned we wouldn't be

Listing 1. Extracting the vDSO from a Running Kernel

```
/* extract_vdso.c */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


int main(int argc, char **argv)
{
    char        buf[256], *mem;
    const char *range_name;
    FILE       *rd, *wr;
    long long   start_addr, end_addr;

    /* Open file for writing the vdso data to */
    if (argc != 3)
    {
        fprintf(stderr,
                "Usage: %s <file> <string>\n"
                "\t<file>:  File to write the vdso data to.\n"
                "\t<string>: Name of the mapped in region, e.g. vdso\n",
                argv[0]);

        abort();
    }

    range_name = argv[2];

    if (!(wr = fopen(argv[1], "w")))
    {
        perror("Error: fopen() - output file");
        abort();
    }

    /* Get this process' memory layout */
    if (!(rd = fopen("/proc/self/maps", "r")))
    {
        perror("Error: fopen() - /proc/self/maps");
        abort();
    }

    /* Find the line in /proc/self/maps that contains
       the substring [vdso] * */
    while (fgets(buf, sizeof(buf), rd))
    {
        if (strstr(buf, range_name))
```

```
          break;
    }

    fclose(rd);

    /* Locate the end memory range for [vdso] */
    end_addr = strtoll((strchr(buf, '-') + 1), NULL, 16);

    /* Terminate the string so we can get the start
       address really easily * */
    *(strchr(buf, '-')) = '\0';
    start_addr = strtoll(buf, NULL, 16);

    /* Open up the memory page and extract the vdso */
    if (!(rd = fopen("/proc/self/mem", "r")))
    {
        perror("Error: fopen() - /proc/self/mem");
        abort();
    }

    /* Hop to the vdso portion */
    fseek(rd, start_addr, SEEK_SET);

    /* Copy the memory locally and then move it to the file */
    mem = malloc(end_addr - start_addr);
    if (!fread(mem, 1, end_addr - start_addr, rd))
    {
        perror("Error: read() - /proc/self/mem");
        abort();
    }

    /* Write the data to the specified output file */
    if (!fwrite(mem, 1, end_addr - start_addr, wr))
    {
        perror("Error: fwrite() - output file");
        abort();
    }

    free(mem);
    fclose(rd);
    fclose(wr);

    printf("Start: %p\nEnd:   %p\nBytes: %d\n",
            (void *)start_addr, (void *)end_addr, (int)(end_addr -
             ➥start_addr));

    return 0;
}
```

hacking glibc. And, you don't need to anyway, because getting the code to work is pretty simple. With the chunks described above all in place, it's time to start building. Just configure and compile your kernel as you typically would:

```
make menuconfig
make bzImage
make modules
make modules_install
```

Now, install and boot your new modified vDSO kernel. Once that is up and running, it's time to test a few things, mainly the vDSO stuff you just added. Let's compile a test case to exercise the vDSO call:

```
/* notb.c */
#include <stdio.h>

int main(void)
{
    int notb = number_of_the_beast();

    printf("His number is %d\n", notb);

    return 0;
}
```

Then, compile the code above as:

```
gcc notb.c -o notb vdso.so
```

The file you link against is vdso.so, which provides the symbol resolution needed to make the kernel call. The kernel version of `number_of_the_beast()` is called, even if the code for that function is completely different in vdso.so. Where is vdso.so located? It's located in the kernel build directory after building the kernel: linux-2.6.37/arch/x86/vdso/vdso.so.

At runtime, when a program executes `number_of_the_beast`, the kernel code is called and not the version of `number_of_the_beast()` in the vdso.so file. If you modify the kernel and, say, have `number_of_the_beast()` return `42`, then unless you load that kernel, you still will get `666`. Even if you compile the test example above with the newer modified-to-42 vdso.so.

Another way of getting the vdso.so file is by writing a program that extracts the vDSO memory from a running executable. Numerous sources on-line explain how to do this, but I briefly describe it here. The vDSO page, which is mapped into the memory of every running process, can be in a non-deterministic memory range of your executing process, thanks to Linux's address space layout randomization (ASLR). To get this address, a running program can find its memory information from the file /proc/self/maps. In there, a line with the text `[vdso]` exists. That line contains the address range in the executing process of the vDSO page. For example, you could run `cat /proc/self/maps`.

Note that running this command multiple times produces different address ranges for `[vdso]` thanks to (if your kernel supports it) address space layout randomization.

The output should look something similar to:

```
...
7fff40d71000-7fff40d72000 r-xp 00000000 00:00 0 [vdso]
...
```

The above range is showing for the `cat` process you just executed that the address range for the vDSO page is located starting at `7fff40d71000` and ending at `7fff40d7200`. Subtracting the start and end range, you get 0x1000 or 4096 bytes. 4096 is the page size often used in the kernel. Listing 1 shows code for extracting the vDSO from a running kernel, and it is based on code from the "Examining the Linux VDSO" article listed in Resources.

A simple dumping of the dynamic object symbols can be conducted via:

```
objdump -T vdso.so
```

Because a shared library is also an elf, the `readelf` tool also can be used on vdso.so.

Security Implication

Anytime you dabble with the kernel, you should consider the security implications. If you think you can "own" someone by creating your own vDSO calls, you might want to think again. Because adding a vDSO requires users to bake their own kernels, the only people they could be compromising is their system and the users on their system. Of course, any dabbling with kernel resources should be done with much consideration. Remember, playing with vDSO goodies occurs in userland; however, your vDSOs can access kernel data. And, your kernel can read vDSO data. That can be a concern, but I'll leave that up to you as an exercise for finding anything exploitable.

Finally, this article is just a little one-two on how to cook up your own vDSO. Now go make yourself a smoking kernel.■

Matt Davis is a software engineer on leave from his job in the US to pursue a PhD from the Computer Science Department at the University of Melbourne, where he is focusing his hackery toward the compiler field. He has been involved in both the fields of modeling and simulation, as well as kernel-level high-precision timing. His interests include coding, compilers, kernels, listening to obnoxious music, consuming vast quantities of coffee and being social with wulfpax and 757.

Resources

GNU/Linux Kernel. 2.6.37: **http://www.kernel.org**

"6.30 Declaring Attributes of Functions" (GCC Manual): **http://gcc.gnu.org/onlinedocs/gcc/Function-Attributes.html**

"Weak Symbol" (Wikipedia): **http://en.wikipedia.org/wiki/Weak_symbol**

"Examining the Linux VDSO" (Truth, Computing and Fail): **http://anomit.com/2010/04/18/examining-the-linux-vdso**

Johan Peterson's "What is linux-gate.so.1?": **http://www.trilithium.com/johan/2005/08/linux-gate**

Matt Davis' "Linux syscall, vsyscall, and vDSO...Oh My!": **http://davisdoesdownunder.blogspot.com/2011/02/linux-syscall-vsyscall-and-vdso-oh-my.html**

Losing Its Way

DOC SEARLS

Why has IBM remained sane while HP's gone nuts? One word: engineers.

When I arrived in Silicon Valley, in August 1985, HP loomed large—literally. My office was in Palo Alto Square, on the corner of El Camino Real and Page Mill Road. Across the street on another corner was a little building preserved as the original post-garage location of Hewlett-Packard. (It's gone now.) The next block up Page Mill was the sprawling headquarters for what Hewlett-Packard had become: the largest and most prestigious company in the whole valley and one of the most desirable places in the world for an engineer to work.

HP in those days was what people called a "Theory Y" company (**http://en.wikipedia.org/wiki/Theory_X#Theory_Y**): a place where work was as enjoyable as play. ("Theory X" referred to the more-common form.) Bill Hewlett and Dave Packard were both alive and active then, and "The HP Way" (**http://www.hpalumni.org/hp_way.htm**) still meant what it said: "trust and respect for individuals...high level of achievement and contribution...uncompromising integrity...teamwork...flexibility...innovation..." and so on.

In his autobiography, *iWoz*, Steve Wozniac calls the old HP "just the most perfect company":

> This was in January of 1973, and for an engineer like me, there was no better place to work in the world. Unlike a lot of technology companies, Hewlett-Packard wasn't totally run by marketing people. It really respected its engineers. And that made sense, because this was a company that had made engineering tools for years—meters, oscilloscopes, power supplies, testers of all types, even medical equipment. It made all the things engineers actually used, and it was a company driven by engineers on the inside so far as what engineers on the outside needed. Man I loved that.

The distinction between marketing people and engineers is the one between today's HP and the one that died along with its founders. You could see it in 1999, when the instrument business—the original heart and soul of HP—was spun off. The name for the new company was Aligent, which sounds like something you squirt up your nose so you won't sneeze. It still exists, but who cares? It's not HP.

Spinning off and anonymizing the soul of the company was the last thing most engineers inside HP would have done, for the simple and obvious reason that no engineers (or anybody) outside the company wanted it. The same was true when HP bought Compaq. Both moves were typical of "marketing people", even as they violated the original function of marketing, which was finding and serving the true wants and needs of customers.

One could see the same disconnect—between marketing fantasy and engineering reality—in HP's purchase of Palm for $1.2 billion in April 2010, and its announced decision to kill what was left of Palm's old business in August 2011. That announcement came at the same time as the company also revealed that its entire PC division—including what it obtained from Compaq (for $25 billion in stock) in 2002—was for sale. (The marketing spin was "Evaluate Strategic Alternatives": **http://www.hp.com/hpinfo/newsroom/press/2011/110818xb.html**.)

At that moment, HP still was the world's #1 PC vendor, with $40.1 billion in revenue and $2 billion in operating income for the most recent fiscal year (*Wall Street Journal* **http://blogs.wsj.com/digits/2011/09/08/h-p's-pc-chief-hopes-for-speedy-spinout**). It had a 17.5% market share, with shipments of 14,888,086 units in Q2 2011 (Gartner: **http://www.gartner.com/it/page.jsp?id=1744216**), ahead of Dell (12.5%), Lenovo (12.0%), Acer (10.9%),

Advertiser Index

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

ADVERTISER	URL	PAGE #
1&1	http://www.1and1.com	27
Advanced Clustering Technologies	http://www.advancedclustering.com	39
Emac, Inc.	http://www.emacinc.com	23
EmperorLinux	http://www.emperorlinux.com	103
iXsystems	http://www.ixsystems.com	7
Logic Supply	http://www.logicsupply.com	43, 111
Lullabot	http://store.lullabot.com	2
Microway	http://www.microway.com	3, 64, 65
OEM Production	http://www.oemproduction.com	91
Open Source Developers Conference	http://2011.osdc.com.au	67
RackMountPro	http://www.rackmountpro.com	15
Silicon Mechanics	http://www.siliconmechanics.com	29
TakeDownCon	http://www.takedowncon.com	97
USENIX Lisa	http://www.usenix.org/lisa11/lj	57

ASUS (5.2%) and Toshiba (5.2%). The more telling difference was in growth. HP's was 3.0%. Dell's was 3.3% and ASUS's was 3.7%, while Acer was down –20.4% and Toshiba was down –1.9%. The only bright spot was Lenovo, which had a whopping 22.5% growth rate.

And, who's surprised? Which would you rather have—a ThinkPad from Lenovo or a whatever-they're-called from one of those other guys? Even given Lenovo's faint-hearted support for Linux, ThinkPads are the least clone-like of all the "compatible" PC laptops. Meanwhile, HP's PCs have long been nearly anonymous but for the company logo they bear. Without that small bit of value, what's their distinction? The answer is none, which is why no other PC maker has bought the division. They already make no-name computers. So, by announcing the hunt for "strategic alternatives", HP put a giant "GOING OUT OF BUSINESS! EVERYTHING MUST GO!" sign in its store window.

Predictably (to all other than HP, apparently), this wacky move has spooked the enterprise customers HP intends to serve with more products, services and devotion. Ray Barnard, chief information officer for Fluor Corp., told the *Wall Street Journal* that his company, which normally spends $25 million per year on hardware and software, has put on hold its plans to buy new high-end computers from HP, because "It appears they are lost right now" (**http://online.wsj.com/article/SB10001424053111903895904576544822297017068.html**). Frank Purdue, vice chancellor for information services at Purdue University Calumet, told the paper that he has to "reassess his tablet plans" and was unsure how to proceed on other fronts with HP. It's not even worth going into the circus around the death, rebirth, re-death and persistent purgatorial state of HP's Android-based TouchPad tablets.

The US National Library of Medicine (**http://www.nlm.nih.gov**) defines psychosis as "a loss of contact with reality, usually including false beliefs about what is taking place or who one is (delusions) and seeing or hearing things that aren't there (hallucinations)". There is no better characterization of HP today, and for the last 12 years or more.

Back at the turn of the Millennium, when IBM announced its love affair with Linux, I wrote this in *Linux Journal* (**http://www.linuxjournal.com/article/4339**): "One of the things I observed about the IBM announcement, when they essentially declared themselves a Linux company, was that this was clearly a company that was now in full compliance with its own engineers."

That's one more reason why the sale of IBM's PC division to Lenovo made sense, while HP's rejection of its PC division did not. IBM remains driven, to the degree it remains sane, by its engineers. HP lost its Way when Dave and Bill died.■

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.