

DJANGO | OPENSWAN | ALPHAMAIL | PLAYSTATION 3 | ICECAST

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community
AUGUST 2007 | ISSUE 160



Ices and Icecast
Streaming Audio

Embedded Linux SOP

Intro to LMMS, inotail,
Karmen and GAMGI

Next-Generation
Residential Gateway

BUILD IT YOURSELF ARCADE AND OTHER COOL PROJECTS

NOKIA E61 and
OPENSWAN VPN

PLAYSTATION 3
SUPERCOMPUTING
CLUSTER

MAGNATUNE
is NOT EVIL

www.linuxjournal.com



+ Create
Firefox Extensions

..... **Exploit**
64-Bit Linux



Enterprise and High-Performance Computing Under Your Control

Appro is **leading 4P x86 computing** by combining the latest technology that meets the demands of the enterprise HPC market.

4-Way XtremeWorkstation™

- AMD Opteron™ 8000 Series processors
- Up to 128GB of DDR2 533/667 memory
- Up to 6.0TB SATA or 2.4TB SAS
- Hot-swappable drives
- 2 PCI-E x16 slots for high-end graphics card
- Windows® or Linux OS



4-Way 3U XtremeServer™

- AMD Opteron™ 8000 Series processors
- Up to 128GB of DDR2 533/667 memory
- Up to 4.5TB SATA or 1.8TB SAS
- 2 PCI-E x16 and 3 PCI-X slots
- Hot-swappable drives
- Redundant power supplies & fans
- ServerDome Management – IPMI 2.0
- Windows® or Linux OS



For more information, please visit www.appro.com
or call Appro Sales at 800-927-5464

AMD Opteron™
Processors:

- Quad-Core Ready - increase capacity without altering datacenter infrastructure
- Best performance per-watt with energy-efficient DDR2
- Optimized system performance with Direct Connect Architecture



Manage Any Data Center. Anytime. Anywhere.



Avocent builds hardware and software to access, manage and control any IT asset in your data center, online or offline, keeping it, and your business, "always on".

Visit us on our Remote Control Tour. For locations near you, go to www.avocent.com/remotecomtrol.



Avocent[®]
The Power of Being There[®]

Avocent, the Avocent logo and The Power of Being There, are registered trademarks of Avocent Corporation. ©2007 Avocent Corporation.

CONTENTS

AUGUST 2007
Issue 160



ILLUSTRATION ©ISTOCKPHOTO.COM/CYRO PINTOS

FEATURES

38 BUILD YOUR OWN ARCADE GAME PLAYER AND RELIVE THE '80S!

Donkey Kong won't steal your lunch money.

Shawn Powers

44 CREATE A LINUX VPN FOR A NOKIA E61 WITH OPENSWAN

Want your personal phone VPN?

Ben Martin

50 MAGNATUNE AN OPEN CHOICE; ITUNES AN EXPENSIVE CHOICE

Trade iTunes for your tunes.

James Lees

54 THE BEST GAME IN TOWN

Clustering game machines?

James Gray

ON THE COVER

- Ices and Icecast Streaming Audio, p. 84
- Embedded Linux SOP, p. 88
- Intro to LMMS, inotail, Karmen and GAMGI, p. 68
- Next-Generation Residential Gateway, p. 58
- Built It Yourself Arcade, p. 38
- Nokia E61 and Openswan VPN, p. 44
- PlayStation 3 Supercomputing Cluster, p. 54
- Magnatune Is Not Evil, p. 50
- Create Firefox Extensions, p. 74
- Exploit 64-Bit Linux, p. 62



Keep it simple. Rest easy.

High technology is exciting. But all too often that includes pointless complexity.

Not with Coyote Point. From local to global load balancing, application acceleration or ultimate network manageability, Coyote Point leads the pack. We take the guesswork out of application traffic management to deliver reliable solutions. You won't find anything faster, smarter or more affordable out there.

Find out why more than 2,000 businesses save time, money and mental energy with Coyote Point. Write info@coyotepoint.com or call 1-877-367-2696.

A coyote is a creature of elegant simplicity. Your network should be, too.



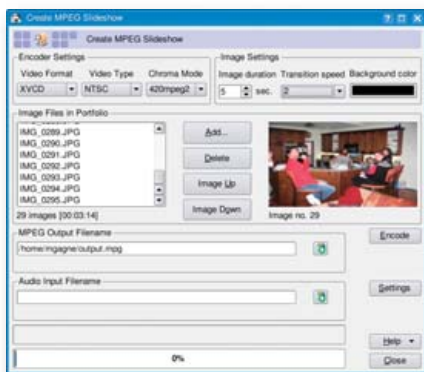
CONTENTS AUGUST 2007

Issue 160

COLUMNS

20 REUVEN M. LERNER'S
AT THE FORGE
Django Views and Templates

24 MARCEL GAGNÉ'S
COOKING WITH LINUX
It's Like Déjà Vu, but at a
Higher Resolution



28 DAVE TAYLOR'S
WORK THE SHELL
007's Favorite Game: *Baccarat?*

30 JON "MADDOG" HALL'S
BEACHHEAD
Cool Change

32 DOC SEARLS'
LINUX FOR SUITS
Work to Be Done

96 NICHOLAS PETRELEY'S
/VAR/OPINION
The Benevolent Racketeer

IN EVERY ISSUE

8 LETTERS
12 UPFRONT
36 NEW PRODUCTS
81 ADVERTISERS INDEX

INDEPTH

58 BUILDING A NEXT-
GENERATION RESIDENTIAL
GATEWAY
So you want a residential gateway?
Alexander Sirotkin

62 EXPLOITING 64-BIT LINUX
Remember when 640K was enough
RAM for anyone?
Steve Munroe

68 TAKE A PEEK AT SOME OF
THE FRESHEST PROJECTS
AROUND
Some cool apps on the radar.
John Knight

74 BUILDING FIREFOX
EXTENSIONS
Customizing Firefox at this level is
geek paradise.
Justin Huff

84 STREAMING AUDIO WITH
ICES AND ICECAST
How W0ZWY put announcements
on the Net.
Brian Matherly

88 STANDARD OPERATING
PROCEDURES FOR
EMBEDDED LINUX SYSTEMS
Need some embedded Linux
guidelines?
**Chi-Hung Chou, Tsung-Hsien Yang,
Shih-Chiang Tsao and Ying-Dar Lin**

92 ALPHAMAIL IS SCALABLE
AND ACCESSIBLE WEB MAIL
Middleware improves performance.
Tony Kay



50 MAGNATUNE



36 NEW PRODUCTS

Next Month

ULTIMATE LINUX BOX

We've got the design for an Ultimate Linux Box next month: a balance of do-it-yourself effort and price that delivers a powerful punch in performance and features. We also examine some alternative configurations in case your priorities lean more toward price or more toward performance. And, we've chosen the Ultimate Linux Box in notebooks too, for those who prefer power and portability. Plus, we also compare the performance several powerful \$7,000 servers.

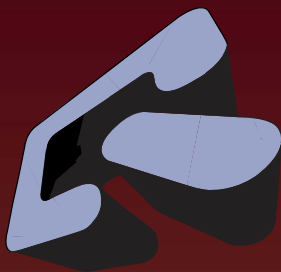
As always, there's much more. We've got the scoop on the evolution of Linux authentication, namely Fedora Directory Server, and hacking cell phones using Bluetooth tools.



Don't feel bad, Our servers won't go down on you either.

We've all known disappointment. And few things are more disappointing than undependable, expensive servers that don't satisfy your needs. To avoid the heartbreak of lost expectations, QSOL provides dependable, server solutions for virtually any popular operating system at a price that will blow your mind not your budget.

QSOL.com Server Appliances come in a variety of configurations to meet the most specific needs of our users. Whether you're into 1U with 8/16 Cores, or 5U with nearly 40+ Terabytes, we can provide a server that can perform under the most demanding conditions. If your server isn't giving you what you want, visit QSOL. (Don't worry, it's a sure thing!)



QSOL.COM Server Appliances

Serving Servers since 1999

LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

Digital Edition Now Available!

Read it first

Get the latest issue before it
hits the newsstand

Keyword searchable

Find a topic or name
in seconds

Paperless archives

Download to your computer for
convenient offline reading

Same great magazine

Read each issue in
high-quality PDF

Try a Sample Issue!

www.linuxjournal.com/digital



LINUX JOURNAL

Editor in Chief

Nick Petreley, ljeditor@linuxjournal.com

Executive Editor	Jill Franklin jill@linuxjournal.com
Senior Editor	Doc Searls doc@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
Chef Français	Marcel Gagné mggagne@salmar.com
Security Editor	Mick Bauer mick@visi.com

Contributing Editors

David A. Bandel • Greg Kroah-Hartman • Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti • Ludovic Marcotte • Paul Barry • Paul McKenney • Dave Taylor

Proofreader Geri Gale

Publisher	Carlie Fairchild publisher@linuxjournal.com
General Manager	Rebecca Cassity rebecca@linuxjournal.com
Director of Sales	Laura Whiteman laura@linuxjournal.com
Regional Sales Manager	Joseph Krack joseph@linuxjournal.com
Regional Sales Manager	Kathleen Boyle kathleen@linuxjournal.com
Circulation Director	Mark Irgang mark@linuxjournal.com
Marketing Coordinator	Lana Newlander mktg@linuxjournal.com
System Administrator	Mitch Frazier sysadm@linuxjournal.com
Webmaster	Keith Daniels webmaster@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

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

Editorial Advisory Board

Daniel Frye, Director, IBM Linux Technology Center
Jon "maddog" Hall, President, Linux International
Lawrence Lessig, Professor of Law, Stanford University
Ransom Love, Director of Strategic Relationships, Family and Church History Department,
Church of Jesus Christ of Latter-day Saints
Sam Ockman, CEO, Penguin Computing
Bruce Perens
Bdale Garbee, Linux CTO, HP
Danese Cooper, Open Source Diva, Intel Corporation

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: +1 713-589-3503
FAX: +1 713-589-2677
TOLL-FREE: 1-888-66-LINUX
MAIL: PO Box 980985, Houston, TX 77098 USA
Please allow 4-6 weeks for processing address changes and orders
PRINTED IN USA

LINUX is a registered trademark of Linus Torvalds.

THE NEW 43LB BULLY.

**THE NEW QUAD-CORE READY PERFORMANCEWARE 1680:
FOUR DUAL OR QUAD CORE OPTERONS, 128GB DDR2/667 RAM IN A TRIM 1U BODY**

Unbeatably tough on your most challenging applications, but easy on the budget (you may even get to keep your lunch money).

Extremely compact and powerful, the PerformanceWare 1680 is designed to take on complex database applications, scientific calculations, and the latest virtual machine technology. AMD Opteron Quad-Core ready, 128GB of fast DDR2 667 RAM make the PerformanceWare 1680 the server to beat.

Be the first on your block to harness the power of this bully. Call now for more details.

www.pogolinux.com



**Experience, Imagination, and Support.
Pogo Linux is Hardware Solutions Built for Linux, Built for You.**

To get started, contact us at 888.828.POGO or inquiries10@pogolinux.com
Pogo Linux, Inc. 701 Fifth Ave. Suite 6850, Seattle, WA 98104



AMD, Athlon, Athlon XP, and the AMD logo are registered trademarks of the Advanced Micro Devices or its subsidiaries in the United States and other countries.
For additional terms and conditions please visit www.pogolinux.com

letters



No Agape for Latin

Agape is not Latin. It is Greek. What does this say about the rest of his article? [See Jon "maddog" Hall's Beachhead in the June 2007 issue of *Linux Journal*.]

--
Tobin

Jon "maddog" Hall's heart was in the right place, but his research was defective. Agape is Greek, not Latin. I submit this small criticism in the highest sense of the word.

--
Daryl

A note for Jon "maddog" Hall: I was always taught (and Wikipedia confirms) that agape is of Greek origin, not Latin. Otherwise, it was an interesting stroll down memory lane. What gives?

--
John E. Young

Maddog, I enjoyed your article, but I almost didn't make it past the second paragraph; agape is a transliteration from Greek not Latin—a compiler error I presume.

I can assure you assembly language is still kicking—the last assembly language (IBM 360 type, now z/Arch type) program I wrote was yesterday.

--
Richard Pace

One can only hope that Jon "maddog" Hall learned his programming languages better than he learned his Latin. In his June 2007 column, he says that he named his boat *Agape* from the Latin word meaning "the highest form of love". The word agape is not Latin, as Hall suggests, but Greek, and being one of several Classical Greek words for different types of love, it generally is understood to mean something like "brotherly love" or charity. Maybe maddog needs to debug his Latin.

--
Max E. Klinger

The origin of the word agape is Greek, but Latin assimilated it. As Jon's teacher taught Latin, not Greek, she was correct in teaching agape as a Latin word. One other note: agape is often translated as charity, but philia is the Greek for brotherly love.—Ed.

More FORTRAN before Interface

A heartfelt thanks to Jon "maddog" Hall for both his article on languages in the June 2007 issue and his dedication of that issue to John Backus. While my career diverges wildly on details from Jon's, there was enough commonality to draw a hearty "Amen, Brother Jon" at the end of his column. I also started with FORTRAN II (on CDC Big Iron) in the late 1960s as an astronomy student, and was drawn into a computer systems class that was mostly a class on CDC's assembly language and CDC computer architecture. Knowing how all those 0s and 1s work together to get all that hardware to do what you want is definitely an essential knowledge to anyone writing serious code. I've not written anything at the machine level in decades, but, as Jon said, what I learned in programming "kindergarten" is in the back of my mind any time I write any code, be it in FORTRAN (still my primary language), Java, PostScript, Python or whatever language I need to use.

--
Jim Secan

Never Fight a Land War in Asia

First off, love the magazine. I look forward to each issue even with a large portion of it being over my head.

I often spend time telling people of the merits of Linux and that we are winning—much like Doc Searls' "Picking New Fights" article in the June 2007 issue. But I never fail to run into the standard, "what about obscure hardware or newer hardware that doesn't run on Linux? With Windows, it just works."

I then have to explain that, no, it doesn't "just work". The manufacturers simply write drivers under Windows, and if they would get pressure to write for Linux, it would "just work" there as well. Our answer is that the manufacturers are the ones that should spend time testing for compatibility. That would, in turn, also free up Linux programmers to write other useful stuff.

Nick Petreley, in */var/opinion* [June 2007], fell victim to one of the classic blunders. The first of which is never get involved in a land war in Asia. The second, which is only slightly less known, is never go in against a Sicilian when death is on the line, and the third (even slightlier less known) is always blame the manufacturer for incompatibility.

We blame ATI for crummy Linux drivers, and not Linux itself, right? ATI, the manufacturer, we say, should be responsible for basic support and contribution to the Linux community to support its hardware, correct? So why doesn't jEdit create an Ubuntu package and submit it?

I realize it wasn't a specific complaint against Ubuntu, but the same assumption is there—that it's Ubuntu's responsibility to go out and do the legwork. Better yet, how about Nick making an Ubuntu package and submitting it?

You guys are great. I just wanted to share with you my initial impression. I know that one day everything will hopefully just gel together. If you have a program you like, ask to make it available for your favorite distro, etc.

--
Kermit Jones

I love the references to Princess Bride,

SEARCHING FOR THE WAY FORWARD WHEN DEBUGGING MULTI-CORE APPLICATIONS?

Search no longer. Industry leaders are finding the path via the TotalView Technologies Multi-Core Debugging Framework — a well-defined, logical approach for organizing and managing the complex parallel debugging process. Brought to you by the leading provider of products that enable quick, easy, and effective debugging of UNIX, Linux, and Mac OS X applications — the framework was specifically designed to address the ever increasing complexities involved in working with multi-core, multi-thread, multi-processor applications. **Find your way by downloading our new white paper — *Debugging in the Multi-Core Age.***

Visit www.totalviewtech.com/whitepaper.



TOTALVIEW TECHNOLOGIES MULTI-CORE DEBUGGING FRAMEWORK

Specifically designed to address the ever increasing complexities involved in working with multi-core, multi-thread, multi-processor applications.



thanks. If I had time to create packages for Ubuntu, I would.—Ed.

To Doc from Doc

I just read your June 2007 column and felt compelled to write about a few issues. [See Doc Searls' "Picking New Fights".]

I am a Chiropractor, running a small office in Sandy, Utah. Perhaps I have overlooked the buzz of Linux enthusiasts coding for healthcare projects. Any ideas on this?

Another area I find lacking...is small business. I am a regular in several channels on irc.freenode.net as bonez39, in #utah and #debian. I have many friends there, who are talented coders, well versed in Linux and a host of tools and languages. Where it becomes disheartening for me, running a small business, is when I seek out other small-business owners, in such channels.

It seems that I find only programmer/coder types, and further, it seems that people running small businesses are too busy running the business to mingle in such groups. Do you know of any groups devoted to small-business owners, who also hope to run things efficiently and effectively with Linux?

Amen to the fight against the telcos. I recently switched over at my office from Qwest, where 1.5 was the maximum rate for broadband, up to 4+Mb with Comcast. This is at the base business rate, so I can keep my overhead down.

What can I, as one business owner, do to motivate patients, friends and family to further the open mobile and freedom movement with telcos?

You've got a new devoted reader here! Thanks for a great piece this month.

--
Dr. Scott S. Jones

I Never Metaprogramming I Didn't Like

I was really pleased to see the article on metaprogramming in the June 2007 issue. [See Ariel Ortiz's "An Introduction to Metaprogramming".] After more than a decade of writing Fortran programs that write PostScript programs, and shell scripts that write Fortran, and awk scripts that make the PostScript frames that a shell script turns into a GIF animation, I finally came across the word metaprogramming a couple of years ago. Certainly a major motivation for this kind of work is to make the machine produce the boilerplate for some programming language. I liked the article, but I was surprised to see bash ignored, except for an oblique mention that "Some languages have a facility called eval."

Finally, after all the interesting articles, there's Nick Petreley's opinion column. Is it just me, or has *LJ* become a lot more interesting since he took the helm? Thanks to you all, in any case.

--
Andrew T. Young

Who Owns OpenGL?

In the editor's comment to the "Miniature OpenGL Development System" letter in the June 2007 issue, you claim that "Microsoft owns the patent to OpenGL".

That is not true, and it's a bad example of spreading FUD that I wouldn't have expected from you.

Microsoft bought some patents from Silicon Graphics in 2002 that may be applicable to parts of OpenGL. (And very little has ever surfaced again, but then again, with patents you never know.)

--
Ulrich Hertlein

I sit partially corrected. Microsoft has staked claims on important portions of OpenGL, not the entire thing.—Ed.

LINUX JOURNAL

At Your Service

MAGAZINE

PRINT SUBSCRIPTIONS: Renewing your subscription, changing your address, paying your invoice, viewing your account details or other subscription inquiries can instantly be done on-line, www.linuxjournal.com/subs. Alternatively, within the U.S. and Canada, you may call us toll-free 1-888-66-LINUX (54689), or internationally +1-713-589-2677. E-mail us at subs@linuxjournal.com or reach us via postal mail, Linux Journal, PO Box 980985, Houston, TX 77098-0985 USA. Please remember to include your complete name and address when contacting us.

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

LETTERS TO THE EDITOR: We welcome your letters and encourage you to submit them to ljeditor@linuxjournal.com or mail them to Linux Journal, 1752 NW Market Street, #200, Seattle, WA 98107 USA. Letters may be edited for space and clarity.

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

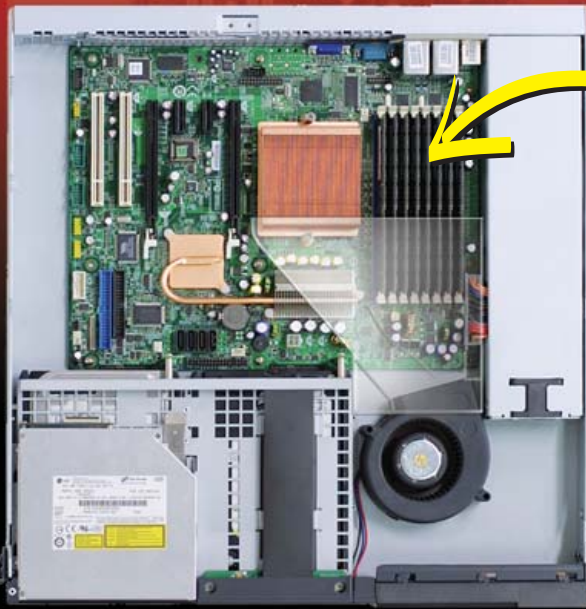
ADVERTISING: *Linux Journal* is a great resource for readers and advertisers alike. Request a media kit, view our current editorial calendar and advertising due dates, or learn more about other advertising and marketing opportunities by visiting us on-line, www.linuxjournal.com/advertising. Contact us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

ON-LINE

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

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

Maximize **AMD Opteron™ Quad-Core** Performance with **UniServer**



Up to 32GB memory on 8 DIMM sockets

1U UniServer with 1 Socket

- Dual/Quad-Core AMD Opteron™ 2000 series
- Up to 32GB of memory
- 2 hot swap SATA 3.5" HDDs
- MCP55V Pro chipset
- Slim CD/DVD-ROM
- 1 PCI-Express x16 slot
- Remote management-IPMI 2.0

Find Full line of Innovative AMD Opteron™ Servers and Workstation Platforms

1U Server: 1 & 2 Sockets, up to 64GB memory

2U Server: 2 & 4 Sockets, up to 128GB memory

3U Server: 2 & 4 Sockets, up to 128GB memory

Workstation: 1, 2 & 4 Sockets, up to 128GB memory



www.bellmicro.com
1-800-291-2070



www.avnet.com
1-888-300-8277



www.synnex.com
1-888-756-4888



www.uniwide.com
1-877-520-0071



Uniwide is an official AMD Validated Server Program Partner

Copyright © 2007 Uniwide Technologies, Inc. All right reserved. Specifications are subject to change without notice.

diff -u WHAT'S NEW IN KERNEL DEVELOPMENT

Linus Torvalds recently said that the whole software suspend debate between Pavel Machek's `swsusp` and Nigel Cunningham's `suspend2` was entirely misguided. Linus said he no longer has faith that either project can succeed. His hope now, he said, was that some new person would step in and take the whole concept in an entirely new direction. He also said that he believes software suspend is indeed a kernel issue and not something that should be entirely in user space. If he finds a solution he likes, he'll incorporate it into the kernel.

It also seems that Linus himself has solved some of the basic development issues, and various people are now telling him his ideas are crazy. Typically when this happens, folks will hurl insults at Linus for weeks or months, while he'll just continue to explain his ideas. Eventually, some kind of tipping point is reached, and a general perspective shift occurs. At this point, one or more folks will rush off to code up the new idea.

Alan Hourihane of Tungsten Graphics has been working on an `FBDev driver` for Intel's `LE80578 chipset`. Intel actually had been funding the work through Tungsten, and Alan posted his code recently to the `linux-kernel` mailing list. In response to various technical suggestions, Alan also posted updated patches addressing those issues—typically a sign that a piece of code is likely to be included in the official kernel sooner rather than later.

The question of what will become of the `Reiser4 filesystem` is still open. It could be adopted into the official kernel; it could transform from a primarily corporate project to a primarily volunteer one; it could become unmaintained and disappear completely. There are many possibilities. But, the fact of Hans Reiser's murder trial does not change most of the issues surrounding whether to include the code in the kernel.

One thing holding the project back is that kernel developers are still reluctant to give significant feedback on the code. Hans' habit of rejecting all commentary and engaging in ad hominem attacks on the commentators has left a mark that even his absence on murder charges has not eroded. A lot of folks who had put in a significant amount of time on giving analysis of Reiser4 code simply have moved on to other projects, and no one has

arisen to take their place.

Without reviewers, the likelihood of Reiser4 going into the kernel diminishes greatly. A number of relatively large technical issues are standing in the way of inclusion, and reviews by kernel developers were one of the key ways the Reiser4 developers could understand those issues.

It seems possible that Andrew Morton might try to jump-start some kind of review process by accepting the patches into his `-mm` tree, with the implied threat/promise of submitting them to Linus Torvalds. This would certainly have a number of kernel developers up in arms and cause them to take a much closer look at the code. But, it also is a fairly drastic measure for Andrew to take.

It's also possible that any sort of review process might take too long. The `Namesys engineers` who maintain Reiser4 are having to confront a diminishing amount of funds available to support the project. At least one `Namesys` engineer, Edward Shishkin, has said he would still devote about 25% of his time to Reiser4 on a volunteer basis if his employment at `Namesys` went away.

As usually happens when issues like these are brought out in the open, recent discussions on `linux-kernel` have resulted in several folks showing interest in reviewing the Reiser4 code. None of the reviewers from the old days seem very interested, but maybe this new set of reviewers will be able to provide enough commentary to smooth the path a little bit.

Adrian Burk continues his long-standing efforts to clean up the kernel. A lot of kernel code hangs around in the kernel for a long time, even after a better alternative comes along, and even when that better alternative makes its way into the official source tree.

In the current case, `X86_ACPI_CPUFREQ` is the new code, allowing users to change the clock speed of their CPUs on the fly, which can amount to a large power savings. The old code, `X86_SPEEDSTEP_CENTRINO_ACPI`, has been deprecated for a long time, and now its time has come.

Adrian posted a patch removing the older code, and there was a small bit of discussion—whether to include the patch in 2.6.21 or 2.6.22, and how to communicate the change to users who might rely on the older code. Bill Davidson was particularly worried about this latter possibility, until Adrian reminded him that the older code had been marked as deprecated for a long time, and the `feature-removal-schedule.txt` file contained a clear explanation of the replacement.

—ZACK BROWN

1. Position of Linux-based Rackspace among Netcraft's most reliable hosting companies for April 2007: 1

2. Number of Linux-based hosting providers among Netcraft's top 10 for April 2007: 3

3. Number of open-source hosting providers among Netcraft's top 10 for April 2007: 6

4. Millions of dollars in 2006 revenues for Rackspace: 224

5. Rackspace revenue growth percentage over 2005: 61

6. Number of open-source venture capital deals in 2004: 36

7. Millions of dollars invested by VCs in open-source startups in 2004: 297

8. Number of open-source venture capital deals in 2005: 41

9. Millions of dollars invested by VCs in open-source startups in 2005: 306

10. Number of open-source venture capital deals in 2006: 48

11. Millions of dollars invested by VCs in open-source startups in 2006: 481

12. Billions of dollars invested by VCs in open-source startups since 2000: 1.9

13. Number of five-year road maps for the Linux kernel: 0

14. Number of operating systems that support more hardware than Linux: 0

15. Millions of digital cameras: 400

16. Millions of cell phones with cameras: 600

17. Millions of digital music players: 550

18. Millions of computers: 900

19. Millions of plasma TVs sold worldwide: 70

20. Year by which the amount of data to be stored exceeds the capacity of data storage devices: 2007

Sources: 1–5: Netcraft.com

6–12, Matt Asay, Robin Vasan and Matthew Aslett
13, 14: Jonathan Corbett

15–20: IDC, via *Freedom's Phoenix*

—Doc Searls



Are you Shocked

by the high cost of iSCSI & Fibre Channel SAN storage?

AoE is the answer!

ATA-over-Ethernet = **Fast, Reliable, Simple** storage.

www.coraid.com



EtherDrive® SRxxxx

- Fast & Flexible RAID appliances with slots for hot swap SATA disks
- Check out our full line of EtherDrive® Storage and VirtualStorage Appliances and NAS Gateways



1. Fast 10 Gigabit Ethernet Storage without the TCP/IP overhead!
2. Unlimited expandability, at the lowest possible price point!!
3. You want more storage...you just buy more disks – it's that simple!!!

Visit us at www.coraid.com



1.706.548.7200

The Linux Storage People

www.coraid.com

Here's a Hitch

The Linux Foundation now has a travel fund for developers who need to attend technical conferences but can't make it on their own. The fund is open to "elite community developers with a proven track record of open-source development achievements and who don't otherwise have access to funding for attending technical events". In other words, it's "for the rock stars of the open-source world", as Jim Zemlin, Executive Director of the Linux Foundation, puts it.

The fund covers travel to "LF



Collaboration Summits, the Linux Foundation's Japan Symposiums, the Kernel Summit, Ottawa Linux Symposium, Linux.conf.au, desktop conferences, such as Guadec and Akademy, and other technical conferences where true collaboration takes place". More pointedly, it "does not offer sponsorships to tradeshows or nontechnical conferences".

If this is for you, or if you wish it were, visit www.linux-foundation.org/en/Travelfund.

—DOC SEARLS

Dude! It's an Ubuntu!

Last month, we visited the overwhelming pro-Linux response to Dell's IdeaStorm site ("New Product Design: Now You Decide!", www.ideastorm.com). As we go to press, there is much talk about Dell's plans to market PCs with Ubuntu pre-installed. That talk has been fueled by the biography page of company founder and namesake Michael Dell. At its top, above all the other computers he uses, is this:



—DOC SEARLS

GLOBE ILLUSTRATION © IStockphoto.COM/MARK STAY

JavaFX Mobile: It Only Starts with the Phones

First there was the Qtopia Greenphone (www.trolltech.com/products/qtopia/greenphone) from Trolltech: "a Linux mobile development device open for unlimited software innovation". Then there was OpenMoko (wiki.openmoko.org/wiki/Introduction): "the world's first integrated open-source mobile communications platform".

Now there's JavaFX Mobile: "a complete mobile operating and application environment built around Java and Linux open-source technologies". Not coincidentally, the phone Sun used to demonstrate JavaFX Mobile at JavaOne in May 2007 was the OpenMoko design (known as Neo1973) that we covered in the February 2007 issue of *Linux Journal*.

Sun and Java have been around the cell-phone business for a long time. And, the platform has come a long way since the time I joked to Sun CEO Jonathan Schwartz that Java to me was "the logo I had to stare at for 16 seconds after I hit the wrong button on my cell phone".

But, this is a huge new move, for five reasons. First, Java is being GPL'd (which was the cover story of our June 2007 issue). Second,

Sun wants to fill the world with mobile phones that are open to development by players other than the phone builders and their carrier partners. Third, Sun actually has clout with phone makers. Fourth, it's not stopping at phones, but looking to put this in all kinds of embedded devices. Fifth, but hardly last, Schwartz & Co. have a bone to pick with Steve Jobs, who has dissed Java while getting huge PR for Apple's new (and closed) iPhone.

There are lots of dots to connect here.

There's SavaJe, the source of the code base for JavaFX Mobile, recently acquired by Sun. There are connections between SavaJe and longtime Linux hardware developer Tadpole Technology.

There's Ed Zander, the Motorola CEO who used to be Sun's President and CEO, and who can't be happy to have seen Motorola's phone "partnership" with Apple go south. Would Motorola be willing to go zig in the open direction while Apple zags in the closed? Good chance.

There's Eric Schmidt, the former Java development chief at Sun who is now CEO of Google. He's on Apple's board, but still,

it's a connection.

Then, there's what Jonathan Schwartz says on his blog (blogs.sun.com/jonathan/entry/when_not_where): he wants the world to have an open platform that allows "any consumer electronics manufacturer to accelerate the delivery of Java/Linux-based devices, from phones to set tops and dashboards and everything else imaginable, without fear of format lock-in or disintermediation from a competitor". All this from "a company with no agenda to disintermediate content owners and every interest in propelling the Open Source community (every portion of the content Sun contributes to the JavaFX product and community will be via the GPL license, at the core of Java and GNU/Linux)".

He also wants this to make phones cheap as well as open, to "play a central role in bringing the Internet to the planet" and to "be the software to build the devices to bridge the digital divide"—like the One Laptop Per Child, only half the price or less.

Encouraging stuff. It'll be interesting to see what happens.

—DOC SEARLS

SUPERMICR

UIO

Universal I/O



Maximized I/O Expandability & Flexibility

The Power of 3

Three Add-On Cards in 1U

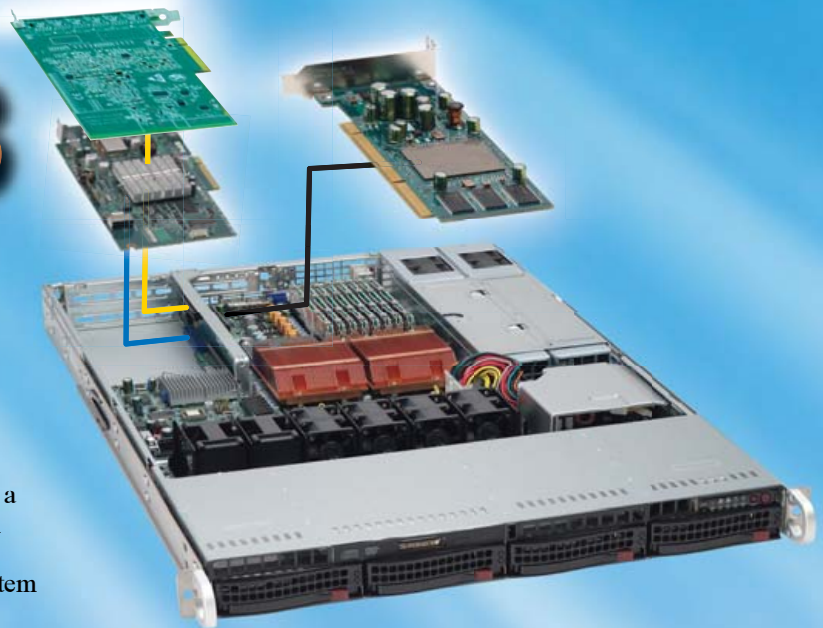
Gives you the freedom to build exactly what you need!

Supermicro UIO servers allow users to select from a wide range of I/O options to provide the ultimate in storage and networking flexibility. The UIO card becomes a part of the serverboard, allowing the system to retain all of its PCI Express and PCI X slots for expansion cards. As a result, future upgrades can be achieved by replacing the UIO card and/or expansion cards instead of replacing the entire system. This versatility helps to minimize the amount of different servers that customers need to operate their business.

- ✓ 3/7 Add-on Cards for 1U/2U
- ✓ Highly Upgradeable
- ✓ High Efficiency Power (up to 90%+)
- ✓ Multiple Expansion Card Options (SAS RAID 5, 10-G, IB...up to 20 choices)
- ✓ Simplify Your Inventory Management

For more information visit us at www.supermicro.com

* Our new generation power supply efficiency measures 90%+ or more under a typical loading operation.



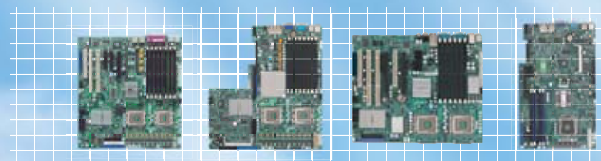
Optimized Chassis Solutions



SC815TQ-R650U

SC825TQ-R700U

UIO Motherboard Options



X7DBN

X7DBU

X7DAL-E+/X7DAL-E

PDSMU

AMAX
1-800-800-6328
www.amax.com

Arrow Electronics
1-888-427-2250
www.arrownacp.com

ASI
1-800-2000-ASI
www.asipartner.com

Bell Micro
1-800-232-9920
www.bellmicro.com

Ingram Micro
1-800-456-8000
www.ingrammicro.com

MA LABS
1-408-941-0808
www.malabs.com

Synnex
1-800-756-5974
www.synnex.com

Tech Data
1-800-237-8931
www.techdata.com

Thin Clients Rock

When you introduce Linux to a group of people unfamiliar with the idea, it's a lot like giving cough syrup to a five-year-old. Average users tend to lose sight of the things Linux can do for them and focus on the rough edges. As Linux "people", our job is often just as political as it is technical, because let's face it, the software is generally the easiest part of the equation.

In 2003, Michigan's educational funding started a downward trend that continues into present day (2007). Every school in our state feels it, and every school is competing to keep students. In an unfortunate twist, 2003 also was the year we needed to change our student management system from our old, now defunct, Macintosh-based system to a Microsoft Windows-based solution. My suggested method to accomplish that was Linux—specifically, Linux thin clients.

I created a menu of proposals for the school board. One described the cost to switch entirely to a Microsoft-based infrastructure. This included workstations, servers, licensing, staff training and additional third-party needs, such as antivirus software and Microsoft Office. The other option required purchasing a handful of servers and using donated computers as thin clients. We would base the system on the Linux Terminal Server Project (LTSP). The first proposal came in at about \$500,000 for the first year, and the latter came in at about \$150,000. I didn't have to do much more selling—"Linux" was the school board's new favorite word.

It's important to note that I easily could have quoted the cost at \$50,000 and still had a functional network. One of the biggest mistakes we as OSS advocates make is not spending enough money. Don't scrimp on hardware when you are saving so much on software. It's important to find a balance between savings and functionality. For example, the bare minimum I needed to convert our district to thin-client technology was:

- A fast LTSP server per 50–75 thin clients to handle the booting.
- A Windows Terminal Server and appropriate licensing.
- Actual thin clients (we were a Macintosh

school, and at that point, old Mac hardware couldn't be used for thin clients, so we had to acquire new ones).

However, my quote also included the following additional purchases:

- Two more fast servers to support additional thin clients and/or load balancing.
- A complete network upgrade (switches and so on) to a 1Gb backbone.
- Microsoft Office licenses to ease the transition to OSS for naysayers.
- New 17" monitors instead of recycled 15" monitors.

The board approved the transition to Linux. We ordered the hardware and jumped feet first into an exciting summer.

LTSP is a system that works on top of basically any Linux distribution. I've always been rather fond of Debian, but for this huge install, I wanted to use something that had been tried and tested. The folks over at K12LTSP have taken Fedora and created an installer that integrates LTSP and scores of tweaks that really make the thin clients shine. The support from their mailing list is usually quick and always provides more help than any phone support I've ever used. In fact, the developers usually are available in real time on IRC if you have a mission-critical question. Unless you have a real aversion to Fedora-based distributions, there's really not any good reason to go elsewhere.

One of the more difficult parts of implementing a wide-scale thin-client rollout is to estimate server requirements. There is no simple way to determine how many thin clients a given server can handle. The requirements vary enormously depending on the applications that will run from the thin clients. A best guideline, based on discussion with the LTSP developers, is the following:

- Get the fastest processors you can afford. Dual/quad is better than single.
- Hard drives, as long as file storage is done remotely, really don't have to be super fast.

- Reserve 512MB of RAM for the server and 100MB per thin client.

- 50 clients is a lot on a server. For more than that, consider multiple servers.

- Gigabit Ethernet on a switched network is a must. Dual gigabit is better.

- File serving for a large network is best done from a separate NFS server, which has the fastest drives you can afford.

Following those guidelines, I figured in order to support the 75 thin clients I needed, I should get three LTSP servers, one file server and one Windows Terminal Server (for the Windows application we needed).

During the installation, one of my biggest worries was not having enough horsepower to run all the thin clients. The first concern was network bandwidth. We did purchase new switches throughout the district, but I wasn't familiar with Ethernet channel bonding, and I didn't want to experiment with the idea on the first day of school. I decided to split the network traffic into two segments. All traffic between the LTSP servers and the thin clients traveled over eth0 on our main network, and all file transfers between the LTSP servers and the NFS server flowed through a separate, gigabit switch over eth1. That effectively isolated the two main bandwidth usages and avoided the need for channel bonding. The solution was very effective.

I was still concerned about the performance issue and didn't want my ignorance to make LTSP or Linux look bad on day one. To make sure bandwidth and processor usage wouldn't be a problem, I decided to give the users a very spartan desktop when they logged in. The software I chose for a minimalistic Linux experience was:

- XDM as the login manager, because it's so minimal and easy to lock down.
- IceWM for the windows manager.
- Nautilus for file management, without the desktop feature enabled (no background or desktop icons).

- A custom, very limited, “Start” menu, with only the programs needed available.

The school year started, and from a technical standpoint, the system performed better than I anticipated. The servers were minimally loaded, and I never even had to bring the second and third LTSP servers on-line. That first day was great, but my excitement was short-lived.

Our teachers *hated* the thin clients. There were nasty complaints, grievances and even personal attacks pointed at me. In hindsight, I easily could have avoided the problems. Currently, four years later, almost the entire staff loves the thin clients. I have more requests for thin clients in the classroom than I have the hardware to handle. Now, four years later, I see the results I expected that first day of school in 2003.

The following are a few ways to introduce Linux successfully, specifically thin clients, to an organization, while avoiding some of the errors we made along the way.

1. Go with a Grass-Roots Campaign

I started testing LTSP with a willing teacher back in 2001 when it was very new. That 4th-grade classroom found a ton of bugs, but worked through them and really loved the technology. It was the right idea to start in such a way, but my focus was too narrow. Multiple pilot programs, with staff demonstrations, are a very effective way to get people talking positively about Linux. With LTSP, pilot programs are easy, because a standard workstation class computer easily can act as a server, supporting 4–6 thin clients. A few small-scale labs really showcase the power of thin clients, and they usually are almost free to implement. When staff members see firsthand the benefits OSS can offer, especially when money is tight, they are much more willing to try something new.

2. Don't Try to Oversell Thin Clients, or Linux

Linux is great. Thin clients are amazing. It's not necessary to claim thin clients are better in every way than traditional workstations. I made the mistake of talking about all the good things thin-client technology meant and failed to point out the shortcomings. As Linux users, we're very familiar with the reasons to use OSS, because we've already learned that the benefits outweigh the drawbacks. It's

important to let new users judge those things for themselves and come to a decision on their own (see the sidebar outlining some pros and cons of thin clients). It's vital to be up front and informative on both sides. In fact, stressing the shortcomings sometimes pleasantly surprises people when they realize how unimportant those faults really are.

3. Get Outside Help

We hired a consultant. In our case, it was late in the game, and it was much more responsive than proactive. Hiring a professional consultant will be money well spent. I don't mean it's necessary to hire a consultant due to lack of personal ability, but rather as a political move with the advantage of another hired brain. The fact of the matter is, consultants actually do very little of the work. We hired a well-known educational technology consulting company in Michigan—Trimble Consulting. If I could do it over again, I happily would have traded one of my servers for a single day of their help and wisdom. That said, it is important to hire wisely. Consultants should be able to do the following:

- Understand Linux, or at least not be afraid of it.
- Know they're there to help the integration process, not determine whether it's wise.
- Be the bad guys. They should be willing to take blame. Consultants are paid a lot and are used to taking the heat off local technicians.
- Communicate very well and translate technical lingo into lay terms.

Hiring consultants is different from hiring helpers. As I said, on the job work-wise, they do very little. The benefits they offer behind the scenes are incredible. The credibility they add to a project is priceless.

4. Learn to Explain Complicated Technology with Simple, Mundane Explanations

Here are some example explanations:

- A thin client is basically a remote keyboard, mouse and monitor that hooks to a giant, powerful computer. Several people plug in to the same giant server and

PROS:

- Cost: thin clients are cheap to buy and cheap to replace.
- Consistency: because everyone is logging in to the same server remotely, the user experience is the same, regardless of the location.
- Quick repair: if something goes wrong with hardware, a quick replacement offers the exact same user experience, and the tech-support people can repair the hardware at their leisure.
- Quiet, cool and low power consumption.
- They are fast! Because thin clients perform at the speed of the server, the end user gets to experience the responsiveness of a \$10,000 server instead of a low-end workstation.

CONS:

- Proprietary software is not as widely available for Linux. Connecting to a Windows Terminal Server helps a bit with that, but at a significant additional cost.
- Floppy, CD and USB Flash drive support is fairly good now, but the experience is different from having a local drive.
- Multimedia is the real downside. Sound works for some things but not for others. Video streaming can work for short clips, but not nearly as nicely as on a full workstation.
- It's different, and different is uncomfortable.

Even though it sounds backwards, my advice is to point out the things LTSP can't do. If you educate users regarding those faults, all that's left to discover are the wonderful things thin clients can do.

share the power.

- An Ethernet network is a lot like a garden hose. Every thin client's hose hooks to a big valve, so the giant server needs a really big hose in order to get the information to all the thin clients.
- A USB Flash drive is like a floppy drive that holds a lot more information. It's also easier to carry and less likely to lose data.

You get the idea. Communication is key, and being able to communicate very complex ideas simply is a huge asset. In fact, looking back over the past four years, our biggest difficulties were not technical, but social. In general, people are not keen to change, so helping them understand the process makes the change less foreign. It's impossible to over-communicate.

5. People Like Shiny Things

Our thin clients were donated Pentium 133 computers. Even in 2003, Pentium 133s were ancient. The big problem is that they looked as old as they were. In fact, most of them had been marked with permanent markers, and many of them had CD drawers broken off the front. They looked terrible.

So on the first day of school, teachers didn't notice the speed that a dual Xeon 3.2GHz server with 6GB of RAM gives a person. The brand-new gigabit backbone in the closet didn't impress them. The deafening sounds of cooling fans in the server room didn't make them feel special. They saw old, junky computers on the first day of school. In fact, the computers looked worse than the Macintosh 5400s they left behind the year before. I did get new 17" monitors, and brand-new optical mice (which were pretty new technology then), but in the end, they just saw crud-

dy old desktops. When staff members found out the "new" computers couldn't even play audio CDs, they were sure they'd been sold out.

In retrospect, for an additional 10% on the original purchase, I could have gotten everyone an actual thin-client device, such as the type sold by the LTSP developers at www.disklessworkstations.com. If that number would have been 15% more than the original, they could have had thin-client devices with LCD displays. Never underestimate the power of fancy new toys to win over hearts.

As I mentioned earlier, our four-year transition to thin clients has been a bumpy road. In the end, OSS was powerful enough to overcome even my short-sightedness. Our staff and our student body love Linux. We've expanded our number of thin clients from the original 75 to a couple hundred. This year, we introduced our first 24-station thin-client lab in the middle school. In fact, this year was the first year I needed all three of those original servers. I'm ashamed to admit that for three years, I had two brand-new servers unplugged. Thankfully, that's no longer the case.

—SHAWN POWERS

Resources

K12LTSP: www.k12ltsp.org

LTSP: www.ltsp.org

Trimble Consulting:
www.trimbleconsulting.net

Diskless Workstations:
www.disklessworkstations.com

They Said It

Support open spectrum. Demand real competition. Ask for investigations of past spectrum auctions and the return of frequencies obtained through fraud. Demand that systems be built-out. It will take political pressure to create the kind of competitive wireless market in which something like a \$30 Internet phone can be made available. Time to bring it.

—Dana Blankenhorn,
blogs.zdnet.com/open-source/?p=1042

I don't know who discovered water, but I'm pretty sure it wasn't a fish.

—Marshall McLuhan, multiple places on the Net

We're continuing to nurse along a few basically 15-year-old filesystems while we do have the brains, manpower and processes to implement a new, really great one.

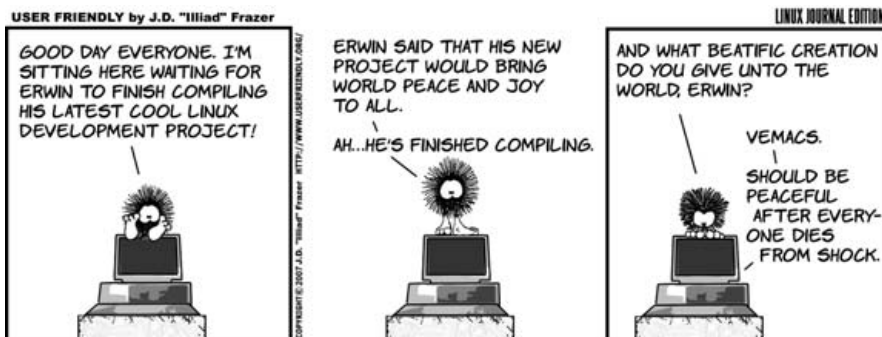
—Andrew Morton, source: Jonathan Corbett, presentation at CELF, April 2007

It's so hard to write a graphics driver that open-sourcing it will not help.

—Andrew Fear, NVIDIA Software Product Manager, source: Jonathan Corbett, presentation at CELF, April 2007

Linux cannot give in to binary-only drivers. That way leads to the end of our free system.

—Jonathan Corbett, presentation at CELF, April 2007



EmperorLinux

...where Linux & laptops converge



Portable

Since 1999, EmperorLinux has provided pre-installed Linux laptops to universities, corporations, government labs, and individual Linux enthusiasts. Our laptops range from full-featured ultra-portables to desktop replacements. All systems come with one year of Linux technical support by phone and e-mail, and full manufacturers' warranties apply.

Toucan T61/T61ws

ThinkPad T61/T61ws by Lenovo

- Up to 15.4" WSXGA+ w/ X@1680x1050
- NVidia Quadro NVS 140M graphics
- 1.8–2.4 GHz Core 2 Duo
- 512 MB–4 GB RAM
- 80–160 GB hard drive
- CDRW/DVD or DVD±RW
- 5.2–6.0 pounds
- 10/100/1000 Mbps ethernet
- 802.11a/b/g (54Mbps) WiFi
- **Starts at \$1650**



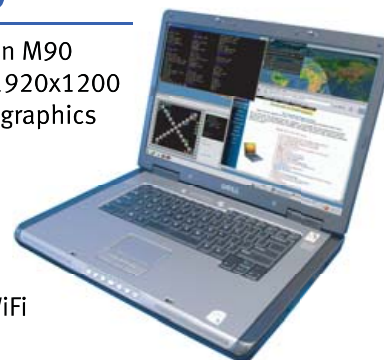
Powerful

EmperorLinux specializes in the installation of Linux on a wide range of the finest laptops made by IBM, Lenovo, Dell, Sony, and Panasonic. We customize your choice of Linux distribution to your laptop and provide support for: ethernet, wireless, X-server, ACPI power management, USB, EVDO, PCMCIA, FireWire, CD/DVD/CDRW, sound, and more.

Rhino D830/M90

Dell Latitude D830/Precision M90

- Up to 17" WUXGA w/ X@1920x1200
- NVidia Quadro FX 3500M graphics
- 1.8–2.4 GHz Core 2 Duo
- 512 MB–4 GB RAM
- 60–160 GB hard drive
- CDRW/DVD or DVD±RW
- 6.3–8.6 pounds
- 802.11a/b/g (54Mbps) WiFi
- ExpressCard/EVDO
- **Starts at \$1445**



Unique

EmperorLinux offers Linux laptops with unique features. Ruggedized Panasonic laptops are designed for harsh environments: drops, vibrations, sand, rain, and other extremes. ThinkPad tablet PCs are like other laptops, with an LCD digitizer for pen-based input both as a mouse and with pressure sensitivity for writing and drawing on-screen.

Raven X61 Tablet

ThinkPad X61 Tablet by Lenovo

- 12.1" SXGA+ w/ X@1400x1050
- 1.6 GHz Core 2 Duo
- 1–4 GB RAM
- 80–120 GB hard drive
- 3.8 pounds
- Pen/stylus input to screen
- Dynamic screen rotation
- Handwriting recognition
- X60s laptops available
- **Starts at \$2300**



www.EmperorLinux.com

1-888-651-6686

Model prices, specifications, and availability may vary. All trademarks are the property of their respective owners.



REUVEN M. LERNER

Django Views and Templates

Getting started with Django views and templates, with an eye to the Django way of working.

Last month, we began looking at Django, a popular framework for creating Web applications in Python. Django has the reputation of being the “Ruby on Rails of the Python world”, and there is some justification for that description. Even though Django was developed in parallel with Rails and has many unique features that distinguish it from its Ruby counterpart, it’s difficult to avoid comparing the two.

In the introduction to Django last month, we saw a number of subtle, but important, differences between the two systems. Django handles a “project” at a time, where each project may contain one or more applications. This contrasts with the Rails approach, in which there is no real equivalent to Django’s projects, because everything is an application.

We also saw that Django comes with a built-in Web-based administrative system. It takes a tiny bit of fiddling with a configuration file to activate this administrative system, but it provides a great number of benefits to any system that uses it.

Despite their differences, both Django and Rails use an approach that is best known as MVC (model, view, controller) developed in the Smalltalk community but adopted by many other languages and frameworks since then. These terms, used verbatim in the Rails world, are called models, templates and views in the Django world, and they form the bulk of a Django-based site.

This month, we work with templates and views, ignoring models and the database API until next month. But, don’t worry; Django’s templates are quite powerful, and they provide an interesting lesson in restraint.

Creating an Application

Our first step involves creating a new application within our project. For a number of reasons, including the simplicity with which we can create a page, our example application is a very simple blog program. We create our blog application by switching into our project directory, and then by using our management program, `manage.py`:

```
$ cd /opt/atf/mysite
$ python manage.py startapp blog
```

If this executes successfully, Django won’t print any messages. Rather, we can get a listing of the current directory, which should now include a blog subdirectory:

```
$ ls blog
__init__.py  models.py  views.py
```

Django is smart enough to stop us from creating an application twice, if we try to do so:

```
$ python manage.py startapp blog
Error: [Errno 17] File exists: '/opt/atf/mysite/blog'
```

Now, let’s create a simple “Hello, world” view, just so we can demonstrate that things are working. Thus, we open `blog/views.py`, which starts off empty except for a comment indicating that this is the file in which views are defined.

The simplest possible “Hello, world” view will be called `index`, acting sort of like the `index` method in Rails or the `index.html` file on a Web site—providing the default content when no specific method is invoked for an application. We also have to import the Python module in which Django’s HTTP response capabilities are defined. When we’re done, the entire `views.py` file looks like this:

```
from django.http import HttpResponseRedirect

def index(request):
    return HttpResponseRedirect("Hello, world.")
```

If our server isn’t already running, we can start it with:

```
python manage.py runserver
```

Or, if we want to run on a publicly accessible IP address, rather than `127.0.0.1` (“localhost”), we can do something like this:

```
python manage.py runserver 69.55.232.87:8000
```

URLConf

To see the output from our view, let’s point our browser to the following URL, expecting to see “Hello, world”: `http://69.55.232.87:8000/blog/`. Instead, we get an error message, indicating that Django has no idea what we’re talking about. The good news is that this error message, which is turned on only when we are developing an application, tells us what we did wrong—namely, that our URLConf definition failed to have any entry for `blog` in it.

This is a fundamental difference between Django and Rails, and it reflects the different philosophies of the two communities. In Rails, you expect the system to do the right thing by default; you should have to say something

only when it deviates from that norm. In contrast, Django assumes that you want to make everything explicit.

Among the things that you need to make explicit is the way in which URLs are translated into method calls. The Django system for doing this is called URLConf, and it is a set of regular expressions defined for the entire project. (It functions something like routes in Rails.) If you are familiar with regular expressions, it should be pretty easy to add or modify URLConf.

For example, let's say we want the `/blog/` URL to go to our blog application. So, we would open the URLConf file, which for the mysite project will be in `mysite/urls.py`. (We already modified this last month, when we added administrative capabilities to our Django site.) We then add a line that looks like the following:

```
(r'^blog/$', 'mysite.blog.views.index')
```

In other words, if the system sees a URL that begins with `blog` and ends with `/`, it should invoke the `index` method within `views.py` in our blog application. And, sure enough, as soon as we save `urls.py` to disk, we can reload our URL `http://69.55.232.87:8000/blog/`, and in our browser, we see "Hello, world". Our Django application is starting to come together.

More Complex URLs

Things are not that interesting if all we have is a single method and if it always does the same thing. For a blog application, we'll presumably want to be able to read a particular posting, or postings, on a particular day. And although we'll get into the model for our blog application next month, it stands to reason that this means we'll need to be able to request blog postings by an individual ID or by date.

Our current URLConf doesn't handle such situations. Indeed, Django requires that we explicitly indicate each possible URL that a user might request and how that URL should be handled. So although we have taken care of `blog/`, we need to handle such URLs as:

```
^blog/ID
```

Luckily, it should be pretty straightforward for us to set up a regular expression that captures such functionality. If we open `urls.py` once again, we can add another statement:

```
(r'^blog/(?P<post_id>\d+)/$', 'mysite.blog.views.view_one_posting')
```

Here, we see something a bit strange and different, namely the use of capturing parentheses, along with `?P` and names inside angle brackets. We tell URLConf that whenever we receive a URL that looks like `/blog/NUMBER`, with one or more digits, we should invoke `view_one_posting`. Moreover, because we have captured the digits (`\d+`, in regular expressions) with

the name `post_id`, `view_one_posting` will be invoked with an additional parameter of `post_id`.

In other words, once we've modified `urls.py` to include the above mapping, we now can go back to `views.py` and say the following:

```
def view_one_posting(request, post_id):  
    return HttpResponse("You asked for post '%s'" % post_id)
```

Then, we can go to `http://69.55.232.87:8000/blog/5`, and in our browser, we get the following response:

```
You asked for post '5'
```

Notice that in the `view_one_posting` method, we used `%s` (for strings) to render `post_id`, rather than `%d` (for integers). This is because parameters and URLs are passed as strings, rather than integers or other data types. We could, of course, get around this problem by converting the string to an integer, but for our purposes right now, this is enough.

Perhaps this goes without saying, but each method in our Django application is a Python method whose output just happens to be sent to the user's Web browser. You can use any number of Python libraries that you want, and even access databases, filesystems and remote computers. So long as the output is returned as a legitimate HTTP response, your method will work. The application we are building in these examples represents the tip of the iceberg, as far as implementation complexity is concerned.

Templates

If we wanted to, we could create a whole Web site using nothing more than the tools we've already seen. However, it quickly would become difficult, and even tedious, to do so, putting all of our output strings as parameters to `HttpResponse`. The real solution is to put the HTML in external files, interpolating variable values as necessary.

There are a lot of different templating solutions out there, and each has its advantages and disadvantages. The most common type is that used by ASP, PHP, JSP and ERb—the last of which is part of Ruby on Rails—in which the code is interspersed with the HTML. This type of template can be extremely easy for programmers to work with, but it can cause trouble when nonprogramming designers are involved, or if you just want to have a complete separation between code and design.

Such templates can be used by Django, but they are not the default. Rather, Django uses templates that are similar in many ways to the Smarty system for PHP, which avoids the use of actual code in the template by introducing its own, deliberately limited language. So long as a view can pass values to a template, there's no real need for a full-blown programming language inside of a template. It's probably enough to have `if/then` statements and some basic loops.

Despite their differences, both Django and Rails use an approach that is best known as MVC (model, view, controller) developed in the Smalltalk community but adopted by many other languages and frameworks since then.

You can use any number of Python libraries that you want, and even access databases, filesystems and remote computers.

This is what Django's default template system provides. In order to use templates, we first modify the settings.py file that sits in the project's main directory. We want to set the TEMPLATE_DIRS variable, giving it a list of one or more directories in which our templates might be found. For example, we could set it to:

```
TEMPLATE_DIRS = (
    "/opt/atf/mysite/templates"
)
```

With this in place, now we create the appropriate directories:

```
$ mkdir -p /opt/atf/mysite/templates/blog
```

And, then we create, in that blog subdirectory, a new HTML file, which we call view_one_posting.html:

```
<html>
<head>
    <title>One post</title>
</head>

<body>
    <p>This is the "view_one_posting" template.</p>
</body>
</html>
```

Now, we modify views.py, such that our view_one_posting method can invoke this template. We add the following import statement at the top of the file:

```
from django.template import Context, loader
```

Then, we modify view_one_posting to be:

```
def view_one_posting(request, post_id):
    t = loader.get_template('blog/view_one_posting.html')
    c = Context({})
    return HttpResponse(t.render(c))
```

Our template is loaded into the variable t, and the context—that is, the variable values we want to pass to our template—is bound to the variable c. We then tell the template to render itself within the context c. In this particular example, we aren't passing any variables in the context, so it is represented by an empty dictionary.

And, if we reload the URL /blog/5 on our site, we should see the following in the browser window:

```
This is the "view_one_posting" template.
```

That's certainly better than what we had before, in that the contents of the template are easier to handle (by programmers and designers alike) than a string inside of a view method. But, how do we pass variables to be interpolated?

The answer is quite simple. In the view method, we can pass any variable we like to the template using the context dictionary, in which the keys are the passed variable names, and the values are the passed variable values. So, we can say:

```
def view_one_posting(request, post_id):
    t = loader.get_template('blog/view_one_posting.html')
    c = Context({'post_id': post_id})
    return HttpResponse(t.render(c))
```

If we want to see this value in our template, we can view it in double curly braces:

```
<html>
<head>
    <title>One post</title>
</head>

<body>
    <p>This is the "view_one_posting" template.</p>
    <p>The post_id is {{post_id}}.</p>
</body>
</html>
```

And, sure enough, when we render this template, we get:

```
This is the "view_one_posting" template.
```

```
The post_id is 5.
```

Conclusion

Django, like Rails and many other Web frameworks, uses MVC to divide the work between models, views and templates. This month, we saw how to connect a URL to a view, how to pass one or more URL parameters to a view and how to invoke a template from a view.

Next month, we will see how to integrate databases and data models into a Django application. ■

Reuven M. Lerner, a longtime Web/database consultant, is a PhD candidate in Learning Sciences at Northwestern University in Evanston, Illinois. He currently lives with his wife and three children in Skokie, Illinois. You can read his Weblog at allneuland.lerner.co.il.

Resources

The main Django site is at www.djangoproject.com. The site contains a great deal of documentation, including tutorials and pointers to mailing lists.

A prerelease copy of the forthcoming Django book (to be published by Apress) is at www.djangobook.com/en/beta, and although the book is still unfinished in many places, it is well written and includes many examples.



One

PGI Unified Binary™

Now, PGI® compilers can generate a single PGI Unified Binary executable fully optimized for both Intel EM64T and AMD64 processors, delivering all the benefits of a single x64 platform while enabling you to leverage the latest innovations from both Intel and AMD. PGI Fortran, C, and C++ compilers deliver world-class performance and a uniform development environment across Linux and Windows as part of an integrated suite of multi-core capable software development tools. Visit www.pgroup.com to see why the leading independent software vendors in structural analysis, computational chemistry, computational fluid dynamics and automotive crash testing choose PGI compilers and tools to build and optimize their 64-bit applications.



The Portland Group™
www.pgroup.com ++ 01 (503) 682-2806



MARCEL GAGNÉ

It's Like Déjà Vu, but at a Higher Resolution

The coolest Linux and open-source applications may be ones you already have.

One of my favorite photo management applications and one of the finest available, is digiKam.

How can our main server be out of disk space, François? Last week, we had enough disk left to last us into the next decade. What happened? No, *mon ami*, I am not accusing you of anything. I'm just curious if you, perhaps, might have an idea. *Quoi?* You were downloading a few packages for the cool apps issue? Let me see. *Mon Dieu!* There are thousands of source packages here! I know you were trying to help, François, but you didn't need to download every application on SourceForge. I see some ISO images here as well. I guess you wanted to find out which was the coolest. How many did you get? All of them! Sigh...we'll talk about this later. For now, we need to get ready for our guests' arrival. They will be here any moment.

Too late, François. Look sharp! Welcome everyone, to *Chez Marcel*, home to exquisite Linux and open-source software served with the finest wines and enjoyed by the greatest of clientele. Your tables are waiting. Please, make yourselves comfortable, and my faithful waiter, François, will fetch the wine. To the cellar, *mon ami*. I think the 2002 Vinedos de las Vientos Tannat will surprise many of our guests. The Tannat grape is a varietal from Uruguay, and as such, it's unusual for many palates. Drink and enjoy!

While we await the wine, let's look at tonight's menu of cool applications. Contrary to François' well-meaning efforts, you don't need to spend hours scouring the Internet looking for the coolest software. All you need to do is spend a little time exploring the applications you already have, and you'll discover wonders you didn't know existed, right there in front of you. Speaking of my faithful

waiter, here he is. François, please pour the wine for our guests. While you enjoy that first sip, *mes amis*, allow me to reintroduce you to a program you already may be using.

One of my favorite photo management applications and one of the finest available, is digiKam. digiKam is a complete digital photo man-

agement system. Using digiKam, you can create collections and organize your photos (by date, type and more) in easily searchable albums (Figure 1). Selecting and downloading images from your camera is easy and fun. You can add comments to pictures, adjust light levels, fix red-eye and more. It basically has everything you would expect from a photo management suite, plus a few additional options that make it a contender for the coolness crown.

As you can see from Figure 1, digiKam automatically organizes your photos by date, but there are other clever ways of organizing them. One way is to create named tags. For instance, say you had a couple hundred travel pictures, of which 30 or so are from Rome. You could assign all vacation pictures a Travel tag and further tag those Rome pictures with the word Rome. Next time you want to hunt down the shots of your Roman holiday, simply click the Rome tag. If one of those images looks like something you want to share with a friend, select the image, click Image on the menu bar, select Email Images, and a dialog appears that lets you add more images to an e-mail message or just send out the one you chose.

There's a problem with having a lot of images though. Okay, there may be more than one problem, but if you are like me, you have a tendency to copy your photos more than once. digiKam has a cool feature that lets you locate duplicate images. Simply click Tools on the menu bar, and select Find Duplicate Images.

Speaking of multiple images, wouldn't it be cool to create an MPEG slideshow—a movie of your photos? Yes, indeed, digiKam does that too. Select a group of images, click Tools on the menu bar, and select Create MPEG Slideshow. A window appears (Figure 2) from which you can choose the amount of time a picture is displayed, a fade time (if you want one), the video output format and more. You also can organize the order in which the pictures are displayed by moving them up or down in the list.

You can do many exciting things with digiKam, which I will let you explore on your own (look under the Image and Tools menu for a start), but before we move on, I do want to share one more cool feature. It's not unusual, at Christmas time, to share cards with pictures of the family. Using digiKam, you can go one step further. How about a calendar for the coming year with great color pictures selected from your collection?

To create a calendar with digiKam, click Tools, then

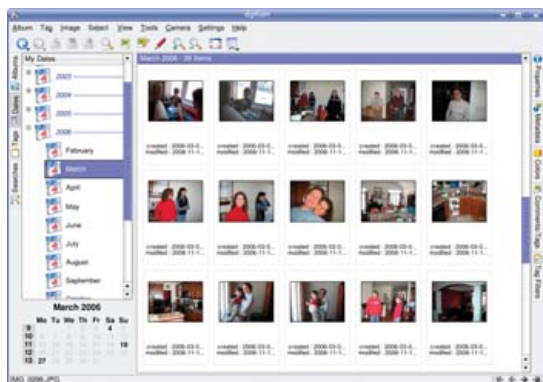


Figure 1. digiKam is a superb digital photo management suite with features you probably hadn't imagined.

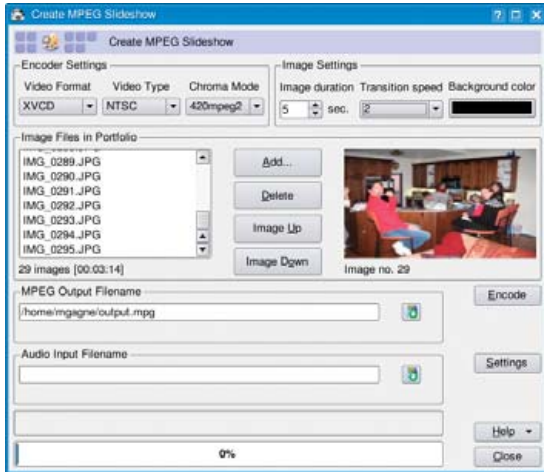


Figure 2. digiKam makes it easy to create a movie from your digital photos.

Select Calendar. At this time, the Create Calendar dialog appears (Figure 3) from which you can create a personalized calendar in only a few steps. To begin, choose a paper size (US or A4) and the location of the image on the page. The latter selection also defines the orientation of the page. At this point, you also can choose a font and decide whether you want lines surrounding the calendar dates.

Click Next to continue on to the photo selection screen (Figure 4). This is where the real fun takes place. You've got 12 months and 12 possible images for your calendar. I suggested family photos earlier, but you could use anything. Instead of buying a calendar of artwork or travel photos, how about creating your own? Those Roman ruins would probably make a nice addition to your office wall. Click on each month to add images, or drag them from digiKam onto the month icons. To clear an icon and select a different picture, right-click the month. When you have chosen something for all 12 months, click Next. The final screen displays a

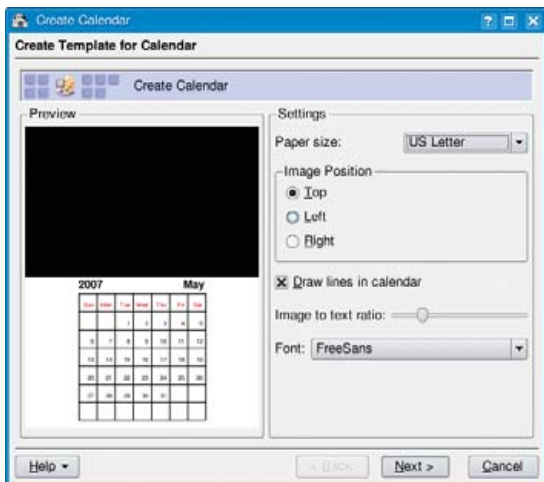


Figure 3. Create a custom calendar using your own digital photos.



Figure 4. Half the fun of creating your own calendar is deciding what pictures will represent each month.

status as each page is generated and an overview of the total. When the job is done, click Finish to close the dialog.

Another favorite application of mine is the amazing Amarok, an impressive multimedia player with a host of cool features that makes it one of the best multimedia players, regardless of operating system. Sadly for a few of your friends, you can't get Amarok on the Redmond OS, but because you run Linux, that's not a problem. When you are hard at work in front of the computer, it's nice to listen to some of your favorite tunes. Depending on where you work, you may even have the luxury of singing along with the music. But what if you don't know the words?

Click the Lyrics tab. If this is the first time you've looked at this feature, odds are the script manager won't yet be running. Click the Run Script Manager button to bring up the Script Manager. You don't need to be listening to a song to do this. At any time, you can click Tools on the menu bar and select Script Manager from there. When the Script Manager window appears (Figure 5), you'll see a list of scripts, categorized on the left. Clicking the plus sign beside each category opens the list.

I invite you to highlight a script and click the About button to discover what each one does. You also can click the Get More Scripts button to discover additional extensions and scripts for Amarok. You'll find screensavers based on your current music, voting scripts, scripts that print album covers and playlists and more. And, of course, there are lyric scripts. One already comes with Amarok, but it isn't necessarily turned on. Click on Lyrc (under Lyrics), and then click the Run button. This activates a script that queries the lyric server at lyrc.com.ar. Then, while listening to your music, you can click the Lyric tab, and Amarok displays the words to the song, turning you into a diva (or divo).

Sometimes, you need to take your music with you. Music devices, such as iPods, are supported using a plugin architecture. What are these plugins? The popular iPod is supported, as are iRiver, Creative Nomad and several other generic audio devices. Depending on your player, you may find that it works with a plugin other than the one you

Another favorite application of mine is the amazing Amarok, an impressive multimedia player with a host of cool features that makes it one of the best multimedia players, regardless of operating system.

Did you know that K3b, your CD/DVD burning application also is a great tool for ripping audio tracks from your CDs?

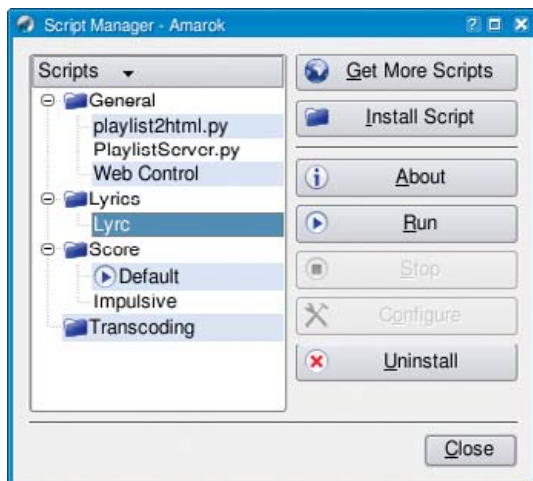


Figure 5. Amarok's feature set can be extended with scripts.

expect, so keep trying. For instance, I have an iRiver T10, which works using the MTP Media Device plugin. All this is to say that if your particular player isn't listed or doesn't work immediately, don't give up; try one of the other plugins first.

To use a player with Amarok, click Tools→Configure Amarok. When the configuration window appears (Figure 6), look at the left-hand sidebar, and select Media Devices. If you haven't done so, make sure your music player (iPod, iRiver or whatever) is already connected and powered on. Depending on the type of MP3 player you are connecting, Amarok already may have detected the device, in which case you'll see the device identified at the top of the configuration window with the Plugin already chosen for you.

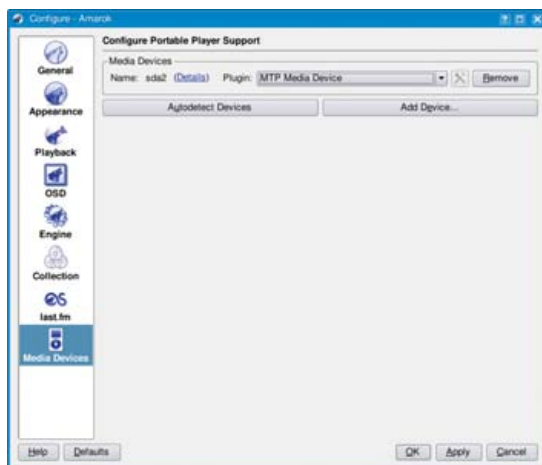


Figure 6. Amarok should be able to detect your music player, but if it doesn't, it's easy to add it manually.

If your player hasn't been detected, you can add a device manually by clicking the Add Device button. From the pop-up window that appears, select a plugin, provide your new device with a name (whatever you like), then identify its mountpoint on the system (for example,

/mnt/ipod). Click Apply to close the window and return to Amarok. Now, to put music on your player, click the Devices tab on the far left. You should see your player listed at the top of the left-hand pane. Directly above the device name is a button labeled Connect. Make sure your device is connected through its USB cable, and click Connect.

If you already have tracks loaded on your music player, they will appear in Amarok's left-hand pane, sorted by artist. Incidentally, you can use this list to delete songs from the player by right-clicking on a track and selecting Delete from device. On the other hand, if your player is empty, you'll be looking at a plain window, waiting to be populated. To add songs, simply drag them from your Amarok playlist (the main window to the left) into the device window. Once you do, a Transfer Queue appears in the lower half of the media browser window on the left. At the very bottom of the media browser, you'll see a graphical bar showing the amount of storage space on your player, the amount available and the space in your transfer queue (Figure 7). When you are happy with the playlist in your transfer queue, click the Transfer button at the top of the media browser window. A progress bar appears at the lower left, updating as each track is copied to your player.

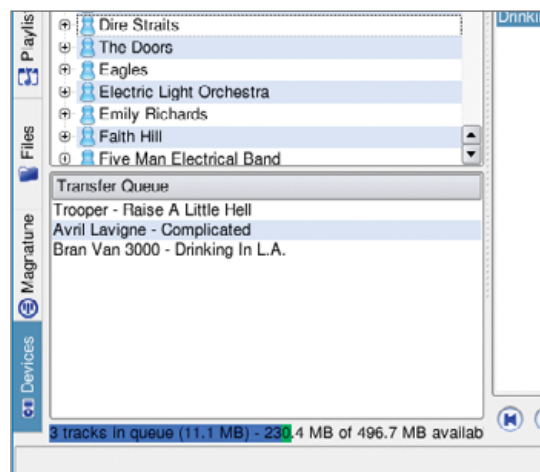


Figure 7. Add your tracks, but keep an eye on the available space.

That's it. You're ready to take your music with you. Monsieur Shakespeare said that music was the food of love. Now all you need is a little more wine.

I wish I could spend more time showing you the wild and unusual side of what you may have come to think of as ordinary, but the hour grows late. Did you know that K3b, your CD/DVD burning application also is a great tool for ripping audio tracks from your CDs? Simply click Tools and select Rip Audio CD. You also can rip a video DVD—click Tools and select Rip Video DVD.

Then there's Konqueror, the all-around file manager, Web browser combination. When using it as a file manager, you can split the screen left and right by pressing Shift-Ctrl-L. Pressing Shift-Ctrl-T splits the screen vertically. Select a location in each screen (including remote FTP or secure copy connections), then simply drag and

Note:

Some older players use memory cards to store music in the same way that digital cameras do. You still can use these with Amarok by creating a generic device with a defined mount-point, such as /mnt/mp3player.

drop from one window to the next. You even can split the already-split windows.

Konsole, your KDE terminal application, can transform to a transparent Konsole, allowing your desktop artwork

to shine through. Click Settings→Schema, and then select Transparent Konsole (or one of the many other looks).

As you can see, *mes amis*, although it's tempting (and exciting, as temptation can be) always to be on the lookout for the latest, greatest and coolest applications, your search need not always be a long one. Sometimes the coolest applications really are the ones you already use. You just need to spend a little time getting to know them better. Uncover their secrets and your existing friendship could turn into something beautiful. *Mon Dieu*, I think my mind may be wandering to places other than software. If so, *mes amis*, surely we can blame the wine. François, please, refill our guests' glasses a final time. Raise your glasses, *mes amis*, and let us all drink to one another's health. *A votre santé! Bon appétit!* ■

Resources

digiKam: digikam.org

Amarok: amarok.kde.org

K3b: extragear.kde.org/apps/k3b

Konqueror: www.konqueror.org

Konsole: konsole.kde.org

Marcel's Web Site: www.marcelgagne.com

The WFTL-LUG, Marcel's Online Linux User Group:
www.marcelgagne.com/wftllugform.html

Marcel Gagné is an award-winning writer living in Waterloo, Ontario. He is the author of the all-new *Moving to Free Software*, his sixth book from Addison-Wesley. He also makes regular television appearances as Call for Help's Linux guy. Marcel is also a pilot, a past Top-40 disc jockey, writes science fiction and fantasy, and folds a mean Origami T-Rex. He can be reached via e-mail at mggagne@salmar.com. You can discover lots of other things (including great wine links) from his Web site at www.marcelgagne.com.

Hurricane Electric Internet Services... **Speed and Reliability** You Can Depend On!

Flat Rate
Gigabit Ethernet

1,000 Mbps of IP

\$13,000/month*

Full 100 Mbps
Port

Full Duplex

\$2,000/month


Colocation Full
Cabinet

Holds up to 42 1U
servers

\$400/month

Order Today!

email sales@he.net or call 510.580.4190

 he.net

* Available at PAIX in Palo Alto, CA; Equinix in Ashburn, VA; Equinix in Chicago, IL; Equinix in Dallas, TX; Equinix in Los Angeles, CA; Equinix in San Jose, CA; Telehouse in New York, NY; Telehouse in Los Angeles, CA; Telehouse in London, UK; NIKHEF in Amsterdam, NL; Hurricane I and Hurricane II in Fremont, CA, and Hurricane in San Jose, CA



DAVE TAYLOR

007's Favorite Game: *Baccarat*?

Create a shell script to get a taste of being Bond, James Bond.

Well, I can't create a casino as a shell script, and I certainly can't create either a secret agent or a gorgeous female sidekick, but I can create a *Baccarat* game as a shell script. Heck, it's probably the first time anyone's even attempted it!

If you've been a faithful reader of this column since the beginning, you'll know that almost two years ago we started out by writing a *Blackjack* game as a shell script. It was a long-winded affair (I didn't just say that, did I?), but as part of the project, we created a simple way to emulate a deck of cards, "shuffle" the cards (that is, put them in quasi-random order) and even convert a numeric 1–52 value into a suit and rank.

We'll use that as the starting point for creating our *Baccarat* game, so we can focus on the complicated rules. Let's start there.

I was recently watching *Casino Royale* and thinking about the James Bond series, particularly how Sean Connery was so much more sophisticated as Bond than Daniel Craig. Connery was more debonair, and one of the ways he'd demonstrate it was by playing a mysterious high-stakes game called *Baccarat* while surrounded by gorgeous women in casinos in Monte Carlo.

Well, I can't create a casino as a shell script, and I certainly can't create either a secret agent or a gorgeous female sidekick, but I can create a *Baccarat* game as a shell script. Heck, it's probably the first time anyone's even attempted it!

If you've been a faithful reader of this column since the beginning, you'll know that almost two years ago we started out by writing a *Blackjack* game as a shell script. It was a long-winded affair (I didn't just say that, did I?), but as part of the project, we created a simple way to emulate a deck of cards, "shuffle" the cards (that is, put them in quasi-random order) and even convert a numeric 1–52 value into a suit and rank.

We'll use that as the starting point for creating our *Baccarat* game, so we can focus on the complicated rules. Let's start there.

The Basics of *Baccarat*

Baccarat has been around since the mid-1400s, and the variation I'll be coding, *Punto Banco*, is completely rule-based with no skill involved. Two cards are dealt to both the player and banker and, depending on those cards, a third might be dealt to one or both. Face cards are worth zero, and numeric cards are worth face value. You add the value of a hand and its final value is that value modulo 10. The higher point value wins, and if they're identical, it's a tie.

For example, if the player was dealt a 7H and a 3C, that'd be worth zero ($7 + 3 = 10 \pmod{10} = 0$). A 6S and 2D is better though; it's worth 8. And, finally, a 9 + 3 + J = 2. Got it? The best possible hand is worth 9 points.

If either the banker or player have 8 or 9 points, no further cards are dealt to either, and the game ends with the dealer or player winning or in a tie. If the player has an initial total of 0–5 points, the player can draw one additional card.

The banker's play at this point is sufficiently complicated that I'll defer explaining it until next issue. For now, let's just look at how to code these rudiments of *Baccarat*.

Ready, Mr Bond?

Capturing the Basics of *Baccarat*

The first piece of the puzzle is pretty straightforward—a shell function that returns a *Baccarat* value for a given sequence of cards (integer 1–52):

```
function handValue
{
    handvalue=0 # initialize
    for cardvalue
    do
        if [ $cardvalue -ge 0 ] ; then
            rankvalue=$(( $cardvalue % 13 ))
            case $rankvalue in
                0|11|12 ) rankvalue=0 ;;
                1 ) rankvalue=11 ;;
            esac

            handvalue=$(( $handvalue + $rankvalue ))
        fi
    done

    handvalue=$(( $handvalue % 10 ))
}
```

This function makes it easy to calculate the value of a hand—whether it will have two or three cards. Here's a typical invocation:

```
handValue ${player[1]} ${player[2]}
```

The result is returned as the global variable `handvalue`, which is calculated by summing up the individual `rankvalue` of each card.

Dealing the cards is accomplished by initializing things:

```
initializeDeck
shuffleDeck
```

Then, here's actually dealing out the cards from `newdeck` into the player and dealer arrays:

```
player[1]={newdeck[1]}
player[2]={newdeck[3]}
nextplayercard=3

dealer[1]={newdeck[2]}
```

```
dealer[2]={newdeck[4]}
nextdealercard=3
```

Realistically, if it's a shuffled deck, it would be an identical result to have the first two cards go to the player and the next two dealt to the dealer, but since we're trying to emulate the actual sequence of events at a *Baccarat* game, I'm dealing cards #1 and #3 to the player and #2 and #4 to the dealer.

The next step is to calculate the value of both the player and dealer hands, which can be done with the handValue function:

```
handValue ${player[1]} ${player[2]}
playerhandvalue=$handvalue
handValue ${dealer[1]} ${dealer[2]}
dealerhandvalue=$handvalue
```

Now, let's test to see if we're done with the hand because either player or banker has a hand value of 8 or 9:

```
if [ $playerhandvalue -ge 8 -o $dealerhandvalue -ge 8 ] ; then
  echo -n "Play is complete. "
  showResult
  exit 0
fi
```

The showResult function simply calculates (and displays) who won:

```
function showResult
{
  if [ $dealerhandvalue -gt $playerhandvalue ] ; then
    echo "Dealer wins"
    result=1
  elif [ $dealerhandvalue -lt $playerhandvalue ] ; then
    echo "Player wins"
    result=2
  else
    echo "Tie"
    result=3
  fi
}
```

I'll stop here, but next column, I'll pick up the task by examining how to test whether the player should get a third card. Then, we'll really dig into the rules for the banker and start running some games!

Note: I have leaned heavily on Wikipedia's information on *Baccarat* for the rules and history of the game (en.wikipedia.org/wiki/Baccarat). I'm focused on what's called *Punto Banco*, the so-called North-American-rules *Baccarat*. If you, like Bond, prefer *Baccarat Chemin de Fer* or *Baccarat Banque*, you can tweak things as necessary. ■

Dave Taylor is a 26-year veteran of UNIX, creator of The Elm Mail System, and most recently author of both the best-selling *Wicked Cool Shell Scripts* and *Teach Yourself Unix in 24 Hours*, among his 16 technical books. His main Web site is at www.intuitive.com, and he also offers up tech support at AskDaveTaylor.com.



Want your business to be more productive?
The ASA Servers powered by the Intel Xeon Processor provide the quality and dependability to keep up with your growing business.

Hardware Systems for the Open Source Community Since 1989
(Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MS, etc.)

1U Woodcrest/Clovertown Storage Server Starts at - \$1,741



- 1TB Storage installed. Max - 3TB.
- 1U Dual core 5030 CPU (Qty-1). Max - 2 CPUs.
- 1GB 667MGZ FBDIMMs Installed.
- Supports 16GB FBDIMM.
- 4X250GB hswap SATA-II Drives installed.
- 4 port SATA-II RAID controller.
- 2X10/100/1000 LAN onboard.

2U Woodcrest/Clovertown Storage Server Starts at - \$3,991

- 4TB Storage installed. Max - 12TB.
- 3U Dual core 5050 CPU.
- 1GB 667MGZ FBDIMMs Installed.
- Supports 16GB FBDIMM.
- 16 port SATA-II RAID controller.
- 16X250GB hswap SATA-II Drives installed.
- 2X10/100/1000 LAN onboard.
- 800w Red PS.



3U Woodcrest/Clovertown Storage Server Starts at - \$3,991



- 4TB Storage installed. Max - 12TB.
- 3U Dual core 5050 CPU.
- 1GB 667MGZ FBDIMMs Installed.
- Supports 16GB FBDIMM.
- 16X250GB hswap SATA-II Drives installed.
- 16 port SATA-II RAID controller.
- 2X10/100/1000 LAN onboard.
- 800w Red PS.

5U Woodcrest/Clovertown Storage Server Starts at - \$6,691

- 6TB Storage installed. Max - 18TB.
- 5U Dual core 5050 CPU.
- 4GB 667MGZ FBDIMMs Installed.
- Supports 16GB FBDIMM.
- 24X250GB hswap SATA-II Drives installed.
- 24 port SATA-II RAID, CARD/BBU.
- 2X10/100/1000 LAN onboard.
- 930w Red PS.



8U Woodcrest/Clovertown Storage server Starts at - \$11,191



- 10TB Storage installed. Max - 30TB.
- 8U Dual core 5050 CPU.
- 2X5050 installed.
- 1GB 667MGZ FBDIMMs.
- Supports 32GB FBDIMM.
- 40X250GB hswap SATA-II Drives installed.
- 2X12 Port SATA-II Multilane RAID controller.
- 1X16 Port SATA-II Multilane RAID controller.
- 2X10/100/1000 LAN onboard.
- 1850 W Red Ps.

All systems installed and tested with user's choice of Linux distribution (free). ASA Collocation—\$75 per month



2354 Calle Del Mundo,
Santa Clara, CA 95054
www.asacomputers.com
Email: sales@asacomputers.com
P: 1-800-REAL-PCS | FAX: 408-654-2910



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

Prices and availability subject to change without notice. Not responsible for typographical errors.



JON "MADDOG" HALL

Cool Change

Here are some ideas for cool projects; send us some of yours.

Nick Petreley, our fearless Editor in Chief, had joined me at my favorite restaurant and watering hole, *Alideia dos Piratas* for an evening out. As the music played and the young folk danced the night away, Nick and I talked about the Linux community and all the interesting things that have happened there.

Nick had declared this month as Cool Projects month for the magazine. Traditionally, the focus had been Home Projects, but he rationalized that although a lot of home projects are cool, the name Home Projects was too limiting. Robotics and wearable computers, for example, are cool, but not necessarily home.

Even before I knew that this month would be Cool Projects month, I had started thinking about what makes a software project "cool".

My "cool" thought processes started with an article by Kathy Sierra called "Professional vs Passionate" at the site headrush.typepad.com/creating_passionate_users/2006/11/two_simple_word.html.

Although this particular article focuses on how we sometimes discard the raw passion people feel when they shout out words not normally found in polite society and how companies change when they trade in their sandals and T-shirts for the suits and ties of the Professional, the article also touched on the concept of what is cool. One example given was: "kid-at-airshow-seeing-an-F16-on-afterburners-rip-by-so-close-it-makes-your..."

Some of you may have been a "kid at the airshow" and had that feeling of cool as the plane made the ground shake.

Now, this coincided with some thought about computer conferences and tradeshows and why people do or do not go to them. Certainly, some people go to conferences and tradeshows to find existing solutions to existing problems. Other people go because they are curious about what is going to be a solution in the next one or two years. Still others go to find out new and novel things (cool stuff)—

things that may never turn into a product or service or things that may be suitable only for such a small group of people that no one sees a commercial opportunity for them. Nevertheless, they're cool.

Lately, some of the conferences I've attended seemed to be concentrating on the existing solutions, and less on the forward thinking and cool stuff.

Of course, cool is in the eye of the beholder, and it seems to change over time. In my early days of programming, choosing a new algorithm to solve a problem and having the program take half the time to get the same answer was cool. Today, this has less of the feeling of the F-16, but it is still important.

So, I went out to my user group and asked, "what's cool?" I received several responses before the mailing list moved away from the original question onto a discussion of one topic that someone thought was cool.

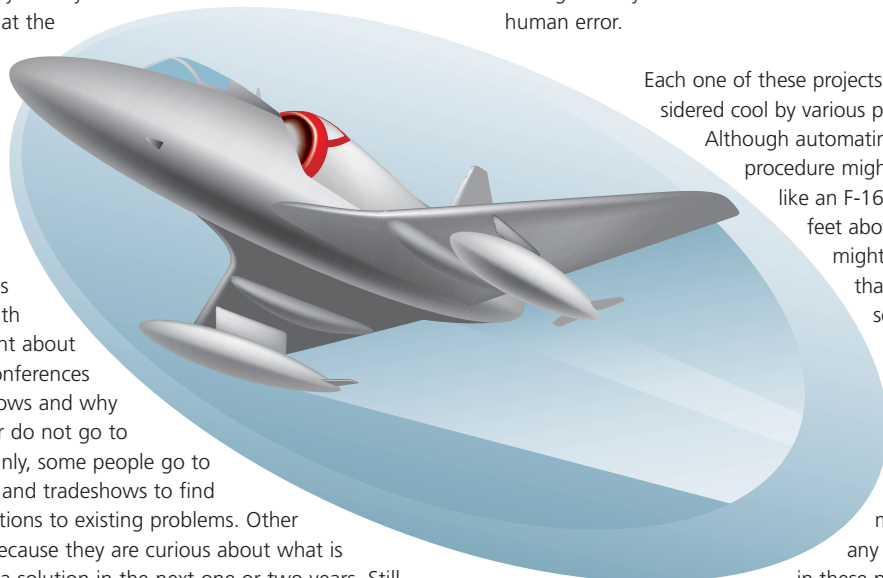
Some of the ideas were:

- NCID: a system for obtaining, displaying and even speaking caller-ID information.
- MythTV, Plutohome and LinuxMCE: various systems for recording, storing and playing back TV programs, video, music, pictures and home automation.
- Automation of testing procedures to cut production testing time by a factor of four and to cut down on human error.

Each one of these projects might be considered cool by various people.

Although automating a testing procedure might not sound like an F-16 flying at 200 feet above you, it might be a project that would get some people excited and try to implement that solution right away. Other people may not have any interest at all in these projects.

Just as I was about to turn away from the responses to



Some of you may have been a "kid at the airshow" and had that feeling of cool as the plane made the ground shake.

my query, someone mentioned the site of C Data Solutions, Ltd. (www.cdatas.com/index1.html), where a computer was being built out of CompactFlash Modules. Now I admit to being a gearhead, particularly when it comes to very small systems, but when I saw this, an F-16 flew right over my head.

A couple of months ago, I mentioned the Gumstix folks (www.gumstix.com) in an article. Both Gumstix and the C Data Solutions systems share some of the same characteristics—they're tiny and low power, yet programmable through our favorite operating system. I see them, and I get excited over what they could do. However, I don't remember seeing any projects at any of the recent conferences I attended using these types of systems, and I think that is a shame.

At a tradeshow recently, I saw a very nice sponsored area where high-school and college students were trying to get their robots to maneuver a maze and do some simple tasks. The students were all in teams of three or four people, had their own colors, flags and work areas. The teams were changing their programs and even doing hardware changes to their robots right at the event. This event was not about selling some product or even solving some

problem. It was about watching students work together in teams and solve problems as teams. It was an exciting time, and it was cool, both for the students and for the spectators.

Realizing that you, my readers, and I may never agree on what is cool or why we think it is cool, I will make a request. Write to me at cool@linuxjournal.com, and tell me about existing FOSS projects that you think are really cool (and why), or what would make a really cool project. If you have ideas for a really cool contest or conference activity, send those too. I will roll them up and post them on the *Linux Journal* Web site.

But, as you formulate your idea, please make sure to duck, so the F-16 will miss your head. ■

Jon "maddog" Hall is the Executive Director of Linux International (www.li.org), a nonprofit association of end users who wish to support and promote the Linux operating system. During his career in commercial computing, which started in 1969, Mr Hall has been a programmer, systems designer, systems administrator, product manager, technical marketing manager and educator. He has worked for such companies as Western Electric Corporation, Aetna Life and Casualty, Bell Laboratories, Digital Equipment Corporation, VA Linux Systems and SGI. He is now an independent consultant in Free and Open Source Software (FOSS) Business and Technical issues.

Expert Included.

Shane's customers are always pushing the limits of technology. That's why he is a fan of the Rackform iServ R2020, an innovative, 1U, two-compute-node system designed to increase computing density while reducing cost, energy, and space requirements. With support for two Quad-Core Intel® Xeon® Processors 5300 Series per compute node, the iServ R2020 combines Intel's proven reliability with industry-leading 16-core-per-1U density. Additionally, 8 Fully Buffered DIMM sockets, 2 hot-swap SATA hard drives, and a PCI-Express slot in each compute node convince Shane that the iServ R2020 provides the density, flexibility, and cost effectiveness needed to tackle even the most demanding computing challenges.

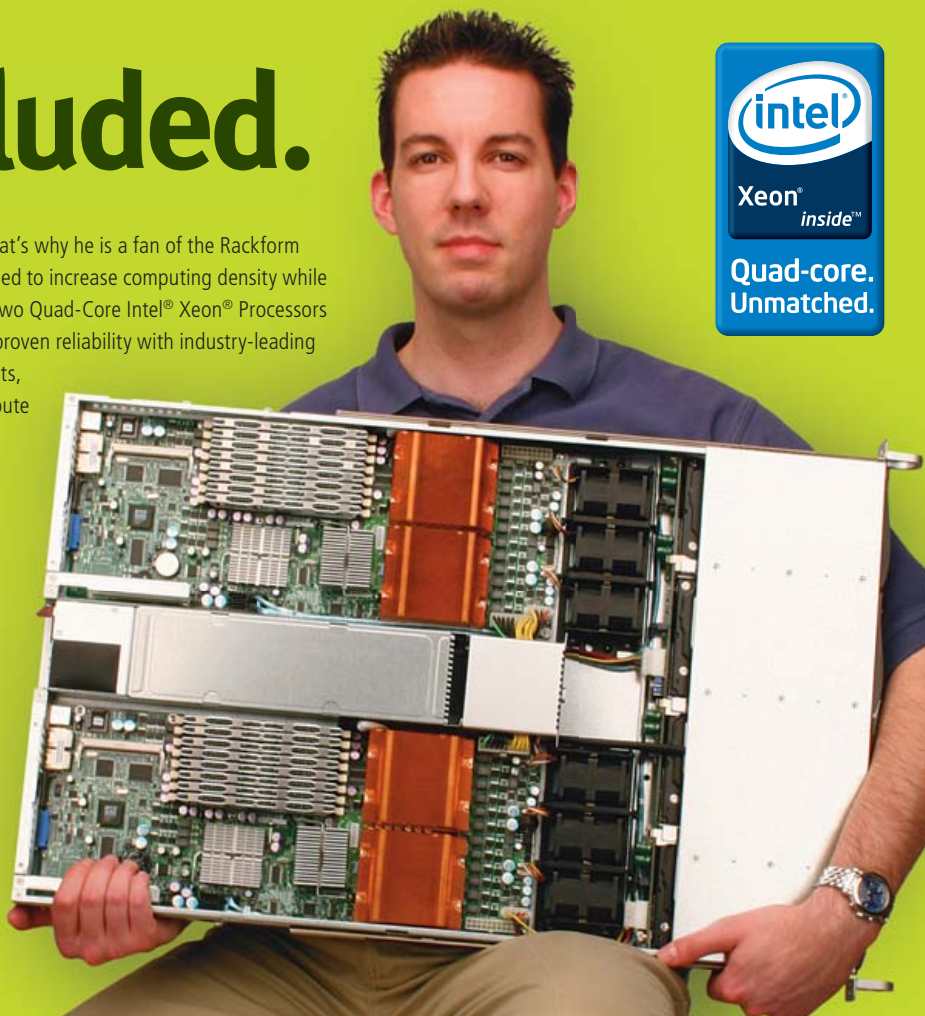
When you partner with Silicon Mechanics, you get more than a powerful Intel Solution — you get an expert like Shane.



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

Silicon Mechanics and the Silicon Mechanics logo are registered trademarks of Silicon Mechanics, Inc.

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





DOC SEARLS

There was a constant digital dance between analysis and synthesis—and in the midst of it all, the increasingly obvious need for tools that are open and free.

Work to Be Done

Break down the knowledge barriers with demotic character, relative impermanence, dialogic imagination and other picture postcards.

We were flying from the Coliseum north above Figueroa into downtown Los Angeles, about five stories above the road surface. After crossing the Harbor Freeway, we passed the Bob Hope Patriotic Hall on our right, then barely cleared ten lanes of elevated traffic on the Santa Monica Freeway. After zooming past the Convention Center and the Staples Center, we veered to avoid hitting a large building facing south across Olympic that featured monstrous portraits of Kobe Bryant and two other Lakers. From here north, the street became a slot between rows of buildings on both sides. When a break showed up at 5th, we took a hard right to head east a couple blocks before taking another right on Hill. Below to our right appeared Pershing Square. Suddenly, we dove down to the ground and into the escalators leading down into the subway. Here we paused on the platform to watch a train go by, then punched out through the roof and into the air above, where we took Hill North to a right on 4th and then across Broadway and through the arched doorway into the Bradbury Building, a hollow shrine to ironwork and exposed elevator mechanics, best known for the noir scenes it provided for the movies *Blade Runner* and *Chinatown*. From here, we flew out to the north and then east on Cesar Chavez to look at plans for turning the concrete trough of the Los Angeles River into a green park of some kind. Suddenly, the park itself appeared, the river blue to the brim.

Our pilot and urban simulation god was Zachary Rynew, a grad student in Architecture at UCLA. The flight was a demonstration of Virtual Los Angeles, a simulacrum of downtown LA. One can fly through and explore Virtual LA in surreal time—that is, right now or at times in the past or future. Virtual LA is designed to facilitate urban planning, emergency response, architectural development, education and a mess of other specialties and combinations thereof. It combines 3-D geometric modeling with street-level and aerial photography. Some of the visuals are so detailed, you can read graffiti and small signs in shop windows. The system originally was developed on SGI systems, but it was moved to Linux several years ago. Like all open-source projects, and like the subject it explores, it is under development. Work will continue on it as long as the project is useful.

Virtual Los Angeles was one among many visualizations at the Digital Innovation Day Showcase at UCLA in May 2007. Others included the Roman Forum Project, 800 Years of Berlin and the Qumran Visualization Project. In UCLA's Visualization Portal, we were treated to a tour of Santiago de Compostela, a 13th-century cathedral in Spain, as three projectors blended images seamlessly on a curved 180° screen surrounding seating for 40. In each case, the experts talked about how the

work of virtual construction (or reconstruction) resulted in better insights and understandings of settings both long gone and not yet built. There was a constant digital dance between analysis and synthesis—and in the midst of it all, the increasingly obvious need for tools that are open and free.

All this work grew alongside and out of a relatively new discipline called digital humanities. I had first heard of the field from Joseph Vaughan, Assistant Director of the UCLA Center for Digital Humanities, in an e-mail responding to a Linux for Suits report on the first Desktop Linux Summit, back in 2003. In that e-mail, Joe reported on gradual progress in getting more Linux and open-source software put to use on digital humanities projects. I ran into Joe at the showcase, and he said that, in fact, a lot of progress had been made during the four years since he sent that e-mail, with much more to go. The same could be said of digital humanities, and for that matter, of all digital technologies.

This was one of the first points made by the keynote speaker, Dr Willard McCarty of the Centre for Computing in the Humanities, Kings College London. Dr McCarty's talk was titled "What's going on?", and he began by telling us that what's happening in computing is by nature perpetually protean. "In the history of inventions", he said, "computing is in its infancy, its products incunabular. The point I am making is that it and they always will be, however progressively better they get."

But, he does note some trends:

About 30 years ago, Northrop Frye noted a striking technological change in scholarly resources, from the "portly...tomes" of his childhood, communicating in their physical stature "immense and definitive authority", to the "paperback revolution" of his adulthood, which better matched the speed of scholarship and gave wing to the diversification of publishing. The demotic character and relative impermanence communicated by these paperbacks also implied the undermining of authority I just mentioned, in this case a weakening of the barrier between author and reader. Running in parallel if not cognate with this physically mediated change came theoretical changes in ideas of textuality, for example, Mikhail Bakhtin's "dialogic imagination", reader-response theory and, more recently, in anthropological linguistic studies of context. Meanwhile, various parts of computer science have developed congruently, from design of black-boxed, batch-orientated systems of former times to toolkits and

BLADE KILLER 2 SERVERS IN 1U



*84 nodes
in 42U*



Ideal for Virtual Servers

Features and benefits:

- **Higher Density than Blades** (Yes, really) — Two dual quad-core servers in a single 1U. Up to 84 nodes / 672 processor cores in a 42U rack.
- **Lower Power** — Superior power utilization. Increased power supply efficiency.
- **Redundancy** — High redundancy clusters. Independent servers instead of single points of failure found in blade solutions.
- **Standardization** — Industry standard architecture. No proprietary blade backplane/architecture.
- **Cooling** — Individually cooled 1U chassis vs. several blades in a single large chassis.
- **Complete Customization** — Every node is considerably more customizable than any blade.
- **Hot Swappable** — Swap an entire dual server without interrupting network flow or data array access.
- **Lower Cost** — Substantially lower cost than comparable blade solutions.

IBM BLADECENTER® H

ABERDEEN STIRLING 122

9U Blade Solution with 14 Blades

7U Server Solution with 14 Nodes

	IBM BLADECENTER® H	ABERDEEN STIRLING 122
	9U Blade Solution with 14 Blades	7U Server Solution with 14 Nodes
Description	9U IBM BladeCenter H chassis with 14 IBM BladeCenter HS21 blade servers	7 1U Aberdeen Stirling 122 Servers, each with two server nodes
Processors per blade / node	Two Quad-Core Intel® Xeon® processors E5310 1.60 GHz with 8 MB cache and 1066 MHz FSB	Two Quad-Core Intel® Xeon® processors E5310 1.60 GHz with 8 MB cache and 1066 MHz FSB
Memory per blade / node	2 GB ECC DDR2 667 MHz memory (max 32 GB / 4 DIMM)	2 GB ECC DDR2 667 MHz memory (max 32 GB / 8 DIMM)
Hard drives per blade / node	Two 36 GB 10,000 rpm internal 2.5" SAS drives (max 146 GB without optional SIO blade)	Two 36.7 GB 10,000 rpm hot-swap 3.5" SATA drives (max 1.5 TB)
Maximum blades / processors per 42U rack	56 blades / 112 processors (448 cores) in 4 x 9U chassis (6U empty)	84 server nodes / 168 processors (672 cores) in 42 x 1U chassis (0U empty)
Warranty	3 year on-site limited warranty for parts and labor	5 year limited warranty for parts and labor
Total price for 14 blade / node solution	\$70,161	\$36,495
Price per blade / node	\$5,012	\$2,607



implementations of “interaction design”. Computing has become literally and figuratively conversational.

This applies not only to the rise of independent and autonomous development by individuals and groups, but also to participation by everybody willing and able to weigh in, including users, which McCarty places now at the start rather than at the end of things:

...it makes less and less sense to be thinking in terms of “end users” and to be creating knowledge-juke-boxes for them. It makes more and more sense to be designing for “end makers” and giving them the scholarly equivalent of Tinker Toys. But, we must beware not to be taking away with one hand what we have given with the other. To use Clifford Geertz’ vivid phrase, we need rigorous “intellectual weed control” against the Taylorian notions that keep users in their place—notions of knowledge “delivery”, scholarly “impact”, learning “outcomes” and all the

rest of the tiresome cant we are submerged in these days. The whole promise of computing for our time—here is my historical thesis—is directly contrary to the obsolete 19th-century cosmology implicit in such talk.

McCarty’s proximal context is the academy, particularly the graduate schools and programs where specialties tend to be well-tended and guarded spaces. About these, he says:

Particularly since the advent of the Web, our attention and energy have been involved with the exponential growth of digitization. The benefits for scholarship here are unarguably great. But as ever larger amounts of searchable and otherwise computable material become available, we don’t simply have more evidence for this or that business as usual. We have massively greater ecological diversity to take account of, and so can expect inherited ways of construing reality and of working, alone and with each other, to need basic renovation. Here is work to be done. It’s not a matter of breaking down disciplinary boundaries—the more we concentrate on breaking these down, the more they are needed for the breaking down. Rather the point is the reconfiguration of disciplinarity. From computing’s prospect at least, the feudal metaphor of turf and the medieval tree of knowledge in its formal garden of learning make no sense. We need other metaphors. Here is work to be done.

The challenge McCarty lays out over and over, with those four words—“work to be done”—is not just for academics alone, or academics in cahoots with programmers and other technologists. It’s work to be done by everybody.

This comes home for me with ProjectVRM, which I’m heading up as a fellow with the Berkman Center for Internet and Society at Harvard University. Here, I’m working to drive development of tools that fix marketplaces by giving customers both independence from vendor entrapments and better ways of engaging with vendors on an equal power footing. These tools of independence and engagement don’t exist yet—though their parts surely do, amid the growing portfolio of free and open-source software and standards that are lying around in the world.

The problem is, the only code I know is Morse. A few years ago, that would have disqualified me as a project leader. Now, it doesn’t. Because now, it’s easier than ever for people who see problems worth solving to find the people and tools that will help those problems get solved. That’s what’s happening with ProjectVRM. It’s what happened with the user-centric identity movement out of which VRM grew as a specialty. And, it’s what will lead Linux and other open-source projects off of their own turf and out into a larger world where users are at the start of things, and not just the end. ■

Doc Searls is Senior Editor of *Linux Journal*. He is also a Visiting Scholar at the University of California at Santa Barbara and a Fellow with the Berkman Center for Internet and Society at Harvard University.

Resources

Virtual Los Angeles:

digitalinnovations.ucla.edu/2007/ccc/projects/Jepson.htm

Digital Innovation Day Showcase: www.ucla.edu/spotlight/07/digital-innovation.html

Roman Forum Project: www.digitalinnovations.ucla.edu/2007/ccc/projects/Favro.htm

800 Years of Berlin:

www.digitalinnovations.ucla.edu/2007/ccc/projects/Presner.htm

Qumran Visualization Project: digitalinnovations.ucla.edu/2007/ccc/projects/Schniedewind.htm

UCLA’s Visualization Portal: www.ats.ucla.edu/portal/default.htm

Santiago de Compostela Tour: digitalinnovations.ucla.edu/2007/ccc/projects/Dagenais.htm

Digital Humanities: en.wikipedia.org/wiki/Digital_Humanities

Dr Willard McCarty: staff.cch.kcl.ac.uk/%7Ewmccarty

Centre for Computing in the Humanities: www.kcl.ac.uk/schools/humanities/cch/index.html

Kings College London: kcl.ac.uk

ProjectVRM: projectvrm.org

Berkman Center for Internet and Society at Harvard University: cyber.law.harvard.edu

Polywell's Ultimate Linux Systems

More Choices, Excellent Support Service, Great Value!

1U Value Servers Starts at \$399, Up to 4GB RAM



(custom config. available)

\$399 1U-485Ax Sempron 3000+, 1GB DDR2, 80GB HD
(For Volume Purchase Only)

\$499 1U-485Ax Athlon64 3500+, 2GB DDR2, 80GB HD
(For Volume Purchase Only)

\$699 1U-690GA Athlon64 X2 Dual-Core 3800+, 4GB DDR2,
2x80GB HD (Dual LAN +\$45)

\$1299 1U-690GA Athlon64 X2 Dual-Core 4200+, 8GB DDR2,
2x80GB HD (Dual GigaLAN +\$45)

Linux Appliance Starts at \$299



Poly 485Ax Sempron 3000+, 512M DDR2,
ATI X1100 Graphics, 100Mbit LAN, DVD,
160G HD **\$299** (For Volume Purchase Only)

Poly 690GA Athlon64 X2 Dual-Core 3800+
1G DDR2 ATI X1250 DVI+VGA, GigaLAN,
DVD-RW, 500GB HD **\$499**

(OEM /ODM Service Available)

1U Advanced Servers, Up to 64GB RAM, 4TB HD



\$1,999 1U-690S4 Athlon64 X2 Dual-Core 500+, 8GB DDR2
2TB 4x500GB HD, Dual Gigabit LAN

\$2,399 1U-1000SL Opteron 1210 Dual-Core, 8GB DDR2 ECC
2TB 4x500GB HD, Dual Gigabit LAN

\$2,999 1U-2500A16 2 x Opteron 2212 Dual-Core, 16GB ECC DDR2,
1TB 4x250GB HD, Dual Gigabit LAN

\$5,999 1U-2500A16 2 x Opteron 2216 Dual-Core, 32GB ECC DDR2,
2TB 4x500GB HD, Dual Gigabit LAN (Option: 64GB+4TB HD)



Low-Cost NAS Storage 2.0TB Starts at \$999



\$999 Netdisk 4000 2.0TB 4x500G

\$1,550 Netdisk 6000 3TB 6x500G (2U)

1U Twin Servers, 1U 8-Way Quad Opteron



\$4,999 1U-Twin 2 x 2 Dual-Core Processors, 2 x 8GB ECC DDR2,
2 x Dual 250GB HD, 2 x Dual Gigabit LAN

\$5,999 1U-8415A 4 x Opteron 8212 Dual-Core, 16GB ECC DDR2,
1.5TB 3x500GB HD, Dual Gigabit LAN

\$7,999 1U-8415SS 4 x Opteron 8212 Dual-Core, 32GB ECC DDR2,
2x74GB 15K RPM SAS HD, 3x Gigabit LAN

High-Density Multi-Processor Servers



1U 8-Way, 5U 16-Way Servers, Up to 128G RAM



\$12,999 1U-8450SS 4 x Opteron 8212 Dual-Core, 64GB ECC DDR2,
2x74GB 15K RPM SAS HD, 3x Gigabit LAN

\$39,999 1U-8450SS 4 x Opteron 8214 Dual-Core, 128GB ECC DDR2,
2x74GB 15K RPM SAS HD, 3x Gigabit LAN

\$18,500 5U-8850T5U 8 x Opteron 8212 Dual-Core, 64GB RAM,
2TB 4x500GB HD, 3 x Gigabit LAN

\$46,999 5U-8850T5U 8 x Opteron 8214 Dual-Core, 128GB RAM,
4TB 8x500GB HD, 3 x Gigabit LAN

6TB 2U Storage Server 2012SC-2055A

4TB 8x500G, Opteron **\$2,999**
6TB 12x500G, Opteron **\$3,999**



Blade Servers - 10 Dual or Quad Processors Blades



\$13,999 8U 10 x 2055A Blades
10 x (Dual Opteron 2210 Dual-Core, 4GB RAM, 80G HD)

\$24,999 8U 10 x 2500M Blades
10 x (Dual Opteron 2212 Dual-Core, 16GB RAM, 80G HD)

\$36,999 8U 10 x 8450A Blades
10 x (Quad Opteron 8212 Dual-Core, 16GB RAM, 80G HD)

\$66,999 8U 10 x 8450A Blades
10 x (Quad Opteron 2214 Dual-Core, 32GB RAM, 80G HD)

18TB 4U Storage Server 4024AIS-2500A16

12TB 24x500G, Opteron **\$7,500**
18TB 24x750G, Opteron **\$11,999**



AMD Dual-Core technology enables one platform to meet the needs of multi-tasking and multi-threaded environments; provides platform longevity

Polywell OEM Services, Your Virtual Manufacturer

Prototype Development with Linux/FreeBSD Support
Small Scale to Mass Production Manufacturing
Fulfillment, Shipping and RMA Repairs



- 20 Years of Customer Satisfaction
- 5-Year Warranty, Industry's Longest
- First Class Customer Service

888.765.9686

www.Polywell.com/us/LJ

Polywell Computers, Inc 1461 San Mateo Ave. South San Francisco, CA 94080 650.583.7222 Fax: 650.583.1974

Opteron, Sempron and ATHLON are trademarks of Advanced Micro Devices, Inc., Quadro, nForce and NVidia are trademarks of NVIDIA Corporation, All other brands, names are trademarks of their respective companies.



Motorola's RAZR² V8

Motorola started tinkering with Linux on mobile phones as an experiment in China, which was supported, interestingly, by the Chinese government. Because the company was so impressed with Linux, our beloved OS is becoming ever-more common on its mobile devices. One of the latest Linux-powered options is the RAZR² V8, a next-generation edition of this popular handset line. The RAZR² V8 features up to 500 minutes of talk time, quad-band connectivity, up to 2GB of memory, a two-megapixel camera, Opera Web browser, independent speech recognition and music management based on Windows Media Player 11. Motorola was coy about the latter feature, divulging only that it has licensed audio codecs, DRM and transfer protocols from Microsoft and integrated them into its Linux-Java platform. Still, it calms the Linux-questioning soul to know that one can intentionally go out and get a butt-kicking Linux-based phone!

www.hellomoto.com



Drew Technologies' DashDAQ

As the Detroit automakers obsessively perfect the cupholder, Drew Technologies seeks to sneak a slick Linux-based device, the DashDAQ, on board your new ride. The DashDAQ, says DrewTech, is "a cross between an automotive gauge, dashboard, navigation system, data acquisition system, trip recorder, diagnostic tool and a handheld computer." The device was designed for use as an automotive display, includes OBD2 communications protocols and runs on Linux (yesss!). The software that is included with DashDAQ allows users to create their own themes and automotive gauge skins. Other features include dual-ARM architecture, 4" (QWVGA) 24-bit TFT color display, a touchscreen and 64MB of RAM. The product is available through resellers, distributors and directly from DrewTech.

www.dashdaq.com

Axigen Mail Server

Axigen continues to add to the feature set of its Mail Server messaging solution, with version 4.0 now shipping. The key new feature in version 4.0 is a Personal Organizer module offering features such as calendaring, tasks, journal, notes and collaborative support. The feature is available via Axigen's WebMail interface and Outlook clients. In addition, the product now contains the Axigen Outlook Connector, which implements most Exchange-like features, such as server-side search folders. Axigen Mail Server comes in three additions: ISP/HSP, Business and a free Office edition.

www.axigen.com



Kyle Wheeler's *Qmail Quickstarter* (Packt Publishing)

If qmail's home page is too unwieldy for you, pick up Kyle Wheeler's *Qmail Quickstarter*, a new book for folks familiar with Linux/UNIX and DNS servers who desire to set up a qmail mail server. Starting with the basics, *Qmail Quickstarter* moves on to getting e-mail messages in and out of the queue, along with storing, retrieving and authenticating them. The book also covers virtualisation of domains and user management, filtering spam, SSL encryption and mailing lists. Packt says that the book's style focuses on practical examples that system administrators can use right away, but that it also explains the rationale behind every example.

www.packtpub.com



Ulrik Pilegaard and Mike Dooley's *Forbidden LEGO* (No Starch Press)



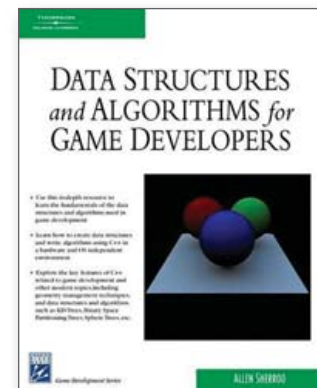
Book publishers seem to release their books in waves, and presently we find ourselves in the midst of a geek-book tsunami. The ever-eclectic No Starch Press continues to pack its LEGO Mindstorms series with fun titles, the latest being *Forbidden LEGO* by Ulrik Pilegaard and Mike Dooley, subtitled cheekily as "Build the Models Your Parents Warned You Against". Presto, rebellion accomplished! *Forbidden LEGO* focuses on "free-style building" and shows you how to make zany models, such as "a toy gun that shoots LEGO plates, a candy catapult, a high-voltage LEGO vehicle, a continuous-fire ping-pong ball launcher and other useless but incredibly fun inventions." A word to the wise: stock up on LEGOs in advance!

www.nostarch.com

Data Structures and Algorithms for Game Developers by Allen Sherrod (Charles River Media)

Grand Theft Auto ain't *Pong*, which is why you'll need this book—*Data Structures and Algorithms for Game Developers* by Allen Sherrod—for your next game undertaking. Published by Charles River Media, this title focuses on teaching game developers the fundamentals of data structures and algorithms using C++ and its alternative options, such as C++ STL. The book also covers many topics that today's game and graphics programmers must know to be successful, including geometry management techniques, KD-Trees, binary space partitioning trees, sphere trees and so on. Furthermore, the code that's included is not platform- or OS-dependent, which is just as we Linuxers like it!

www.charlesriver.com



ActiveState
Dynamic Tools for Dynamic Languages

ActiveState's Tcl Dev Kit

The latest news from ActiveState is the release of Tcl Dev Kit (TDK) 4.0, a multiplatform toolkit for creating and deploying Tcl applications. In addition to providing essential tools for building self-contained, installation-free executables, TDK includes a graphical debugger, profiling and analysis tools for code optimization, a pre-compiler and a Windows services manager. New features in TDK 4.0 include checker enhancements for finding errors in Tcl scripts, enhanced TEA package management, compiler support for Tcl 8.5, GUI facelifts and support for Mac OS X and Solaris x86. Other supported platforms include Linux, Windows, AIX and HP-UX.

www.activestate.com

Please send information about releases of Linux-related products to James Gray at newproducts@linuxjournal.com or New Products c/o *Linux Journal*, 1752 NW Market Street, #200, Seattle, WA 98107. Submissions are edited for length and content.

BUILD YOUR OWN ARCADE GAME PLAYER AND RELIVE THE '80S!

Save your quarters;
you won't need
them when you
build your vintage
arcade player.

Shawn Powers



I grow weary of hearing how bad Linux is for gaming. In this article, I buck that stereotype and show how to use penguin power to relive the '80s. In this cool project, I describe how to construct a fully functional arcade cabinet. When complete, you'll be able to play all the old coin-op games from your childhood in the coin-free luxury of your living room (or garage—depending on the tolerance of individual spouses).

The system uses software based on the MAME (Multi-Arcade Machine Emulator) Project to play the original classics emulated in Linux. MAME uses software to emulate the original arcade hardware. This article explains how to connect the original joysticks and buttons, and even gives some pointers regarding purchasing rights to the original games.

The Biggest Case Mod You'll Ever See

First off, it's the arcade case that makes this project truly awesome. If you don't have an actual arcade cabinet, with original arcade controls, you might as well go back to playing *Zork* on your TSR-80 for your dose of nostalgia. The cabinet is what makes this cool, period. It's up to you to decide whether to make a cabinet or buy a used one. If you make the whole thing from scratch, the design is completely up to you. The downside is that doing this requires some wood-working skills. I have no such skills. In fact, to me, "saw" is a type of waveform on an oscilloscope. So, I opted to purchase an old arcade machine and gut it out. Unless you are a woodworker, this is the way I'd suggest doing it. As a bonus, I got all the buttons and controls I needed for free, and the case even smelled a little like a pizza parlor. (Okay, that might be my imagination.)

To find used cabinets, look up "coin op" or "arcade" in the phone book. It was my particular fortune to find a man willing to give me two used cabinets for \$30. I did have to listen to a litany on the downfall of



Figure 1. You can see the cabinet won't win any beauty contests, but it's just dripping with nostalgia.

the arcade generation and how kids these days don't appreciate the classic games. I think he threw in the second case for me when I mentioned that I was rebuilding it because I wanted to relive my youth. He liked that idea almost as much as I did. Keep in mind that if you happen to get an arcade monitor, it's possible actually to use it, but it requires more work. For the purpose of this article, I assume you're using a standard computer monitor.

After I bought my wife flowers and took her out to dinner, I was allowed to bring one of the cases into the house and start the fun. My particular case was from an old *Neo-Geo* game, and I chose to leave the somewhat damaged decal on the side (Figure 1).

If you are, in fact, a woodworker, you may want to purchase the particle board and make a truly custom cabinet. There are designs and original measurement specifications for some common arcade machines on www.arcadecontrols.com. In fact, that site has tons of information on constructing a MAME machine from the ground up, and it's an invaluable Web site. To be honest though, even if I were able to start from scratch, I

actually prefer the used cabinet. There's something special about knowing it's the real deal, even if the insides are all different.

The most difficult part of construction, for me, was getting the monitor properly mounted. I used an old 17" Apple monitor that was destined for the trash. As you can see in Figure 2, the monitor shelf has to be at the right angle so that the face of the screen is flush with the Plexiglas front. I'm sure there are wood-working shortcuts for figuring out how to get the angle correct, but I ended up with some really complicated geometry equations involving SINE and COSINE. Yep, I'm a geek.

The shelf below is



Figure 2. I most likely could have gotten just as close with a guess, but I must confess a lot of math went into the angle calculation for this shelf!



Figure 3. I really was lucky with the free buttons I got with the cabinets. I couldn't have planned the colors any better if I tried. You'll see evidence of my woodworking shortcomings by the extra hole I drilled.

to hold the computer. All I did there was line up the shelf with the coin door in the front, so I could reach in to power the computer on and off. (The coin mechanism actually can be wired and used, but mine was broken, so I never installed it. Maybe next time.)

The last bit of woodworking is the most complicated. You must decide what button layout you desire. I'm the kind of person who spends 20 minutes looking at a dinner menu, so a decision like this was bordering on impossible for me to make. I read, I Googled, I compared and I surveyed. My suggestion is that you avoid all that and go with the layout I chose. Why? Because it works for most games, and you can always change it later. Figure 3 shows the button layout I chose, which happens to be a standard

OPTIONS

SOFTWARE: AdvanceCD, custom Linux install.

- **PROS:** creates a custom CD or bootable USB key that boots directly into the AdvanceMENU front end.
- **CONS:** really tough to configure if you don't use the standard button layouts. If your hardware isn't detected properly, it's also hard to fix. Also it's kind of slow, especially if booting from USB 1.1.

SOFTWARE: LinCade, based on Gentoo.

- **PROS:** also uses the nice AdvanceMENU front end, and it's a little more configurable, with the option to add and modify games after the install.
- **CONS:** still very difficult to configure if you don't use the standard MAME button layout.

SOFTWARE: KnoppixMame, based on Knoppix.

- **PROS:** great hardware detection.
- **CONS:** really not designed for an arcade cabinet, but rather for someone with a keyboard and mouse.

SOFTWARE: standard Ubuntu and WahCade front end.

- **PROS:** very easy to configure. The Ubuntu install handles all the hardware detection. This is the option I chose.
- **CONS:** not specifically designed for an arcade cabinet.

Street Fighter II layout.

You'll notice an extra joystick in the middle, which is not part of the *Street Fighter* layout. It's an original *Ms. Pac-Man* joystick that my wife got as a belated high-school graduation gift. (We're an odd family.) It's wired directly to the player-one joystick, so don't worry if you don't have such a beast, it's not really additional, just an alternate.

The joystick and button holes will vary in size, but standard buttons use 1 1/8" holes. Feel free to lay the buttons out however you like. I went out of my way to make sure the buttons were perfectly placed, but there aren't rules for these things. Don't forget a button for inserting coins! Mine is sticking out of the front of the machine, about where you'd put in a quarter if it were an actual arcade.

Now you should have a mostly empty case with a monitor shelf, computer shelf and a wooden control panel full of holes. If you are doing this work in the house, and you happen to be married, take this time to vacuum. You can thank me later.

If you are the artistic type and want to paint the control panel and cabinet, go ahead and do so. I couldn't wait, and to this day my cabinet has a raw-wood control panel.

Goodbye Keyboard, Hello Solder

This is where the magic happens. If you're anything like me, you've already tried MAME on your computer and played the classic games with your keyboard. It's fun, but it's definitely not the same as the original. In this step, we connect the real arcade controls to the computer. First, if you weren't lucky enough to get used joysticks and buttons on the cabinet, you need to buy some. There are a few places to buy arcade controls, but one of the most popular is Happ Controls (www.happcontrols.com). If you check out the controls section of the Arcade Controls Web site mentioned previously, you'll find a plethora of vendors for such things.

Once you have the actual buttons and joysticks, you need to decide whether you want to buy a keyboard encoder or hack a keyboard yourself. The former option is easier and more reliable, but the latter offers a certain geeky charm. I chose the latter. There are very in-depth instructions on the Arcade Controls Web site regarding how to figure out button layouts properly, but see the Keyboard Hacking sidebar for a quick description, so you can decide for yourself which way to go. If you do

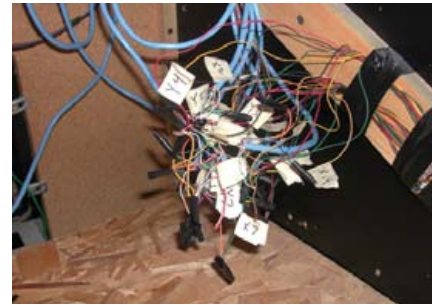


Figure 4. This photo pretty much speaks for itself. It's a great example of how not to organize wires.



Figure 5. My goal was to avoid soldering. The spade plugs look great, but they honestly don't make a very good connection—another example of what not to do.

choose to hack a keyboard, I encourage you to come up with a better way to connect the wires than I did. Figure 4 shows my horrible rat's nest of wires. It's ugly, but it works.

Whether you choose to create your own keyboard controller or purchase one from a place like Hagstrom Electronics, you still have to connect the wires to the arcade controls. Because I'm terrible at woodworking, you'd think there would be a certain balance in the universe that would suggest I could solder well. Let me assure you that balance does not exist. I thought it would be really great to avoid soldering to all the buttons. I bought some spade connectors that would slide right onto the terminals (Figure 5). Although the idea still sounds great, it turns out that the microswitches in the buttons and joysticks really need a solid connection. Unless you want to take apart your control panel to wiggle wires every so often (like me), I highly suggest you solder everything. Invite a friend, because I'm under the opinion that soldering takes three hands. Perhaps your two hands are better than mine, but at least the friend can hold a light.

OBTAINING LEGAL ROMS

MAME ROMs are easily located for download on the Internet, but the legality of using those ROMs are questionable. There are a couple ways to ensure that you have the legal right to your ROMs.

1. Buy the actual motherboard from the original game. This may sound far fetched, but it's not too difficult to purchase broken arcade boards from either local coin-op vendors or on-line from eBay. Because you're just looking for the legal right to use the downloaded ROM, it doesn't really matter if the board is broken.
2. You can buy the X-Arcade keyboard, which comes with a handful of classic ROMs.
3. Check out arcadecontrols.com. There are some free, noncommercial games available for download and some more ideas regarding obtaining legal ROMs. It often is difficult to find whom to buy the rights from, even if you want to spend the cash.

How Many GHz to Emulate 2.5MHz?

So now, you should have a cabinet, mounted controls, some sort of keyboard interface and an empty shelf waiting for a computer. The last step is to put in a computer and install the software. The horsepower you'll need really depends on what games you want to play. If you are planning to play the classics, like *Pac-Man* and *Asteroids*, a Pentium II should suffice. If, however, you want to play *Street Fighter II*, or any newer games, you should get a Pentium III or 4. I have a mid-

dle-of-the-road Pentium 4. If you want my advice, I'd say to use what you've got. If you feel the need to invest more later, you always can do so.

It's Penguin Time!

To run MAME in an arcade cabinet, you must configure two separate programs: MAME itself and a front end. Because MAME is a command-line program, you need a front end to launch the games with the joysticks and buttons. This last step is really my favorite part. If you thought my



Figure 6. Here's the main WahCade screen. This screenshot was provided by Andy Balcombe, the WahCade developer. Thanks Andy!

Linux Laptops

Starting at \$799



Linux Desktops

Starting at \$375



Linux Servers

Starting at \$899



DON'T BE SQUARE!
GET CUBED!



R³ Technologies
Working Hard to Bring Technologies to Life

309.34.CUBED
shopcubed.com

KEYBOARD HACKING

Hacking a keyboard in four easy steps:

1. Take apart the keyboard, making sure not to damage the PCB.
2. Find the two sets of connectors on the PCB, and on a chart, label them X and Y.
3. Trace the very small, fine lines on the flexible plastic sheets to determine the X and Y coordinates for every key. No, really, I'm not kidding.
4. Solder wires onto each X and Y terminal, and use those wires to connect to the arcade controls.

indecisiveness regarding the button layout was excessive, you don't want to know how long I spent determining what software to run. Obviously, I chose to run Linux. Check out the Options sidebar for some comparisons on a few common choices regarding distros and front ends.

I chose to install a stock Ubuntu distribution. Really any distro would have worked, but I just happen to like Ubuntu's single CD install. Also, the WahCade front end has a really nice .deb installer, so Ubuntu's Debian roots really made sense. First, install Ubuntu like a normal desktop user. I chose to install

The system uses software based on the MAME (Multi-Arcade Machine Emulator) Project to play the original classics emulated in Linux.

Feisty Fawn, because it's the most recent release, and I happened to have a CD already burned. I'd highly suggest running the installation using the same monitor you plan to use in the arcade cabinet. It will save a lot of hassle later on.

After Ubuntu is installed, open up a terminal window, and install xMame (the X Window System version of MAME):

```
# sudo apt-get install xmame-x
```

Then, go to www.anti-particle.com and get the latest version of WahCade. You'll want to get the .deb file. To install it, type:

```
# sudo dpkg -i wahcade-xxx.deb
```

Now, you're ready to configure things. The installation instructions at the WahCade home page are pretty clear, but make sure to follow them carefully. You'll have to do a few command-line things and figure out where you want to store ROMs and such. There is a really nice GUI configuration tool, in which you can assign the proper keys (from your joysticks) to control the interface. Once you have the front end configured and ROMs in place, which admittedly will take a while, go ahead and fire up WahCade. Make sure to add the `-fullscreen` and `-skip_disclaimer` options for xmame, the latter because it's difficult to type OK with just a joystick. (MAME displays a disclaimer by default that forces you to type OK in order to continue.)

If everything is configured correctly, you should see the WahCade program and be able to select and launch games accordingly (Figure 6). I hate to admit this, but I spent *hours* trying to figure out how to configure the buttons for xmame. I tried to edit config files, but it was too complicated for me to figure out. In the end, it turns out MAME has a built-in method for reassigning keys. When I realized how simple it really was, I could have kicked myself. I still might. Simply press the Tab key once you're inside a game, and you get a nice configuration screen where you can configure either the default keys or the game-specific keys.

One of the beauties of MAME is that it's possible to assign a command to a combination keystroke. For example, I didn't want to install a button just to exit a game, so I mapped "exit game" to pressing Player 1 and Player 2 simultaneously. It really limits the amount of buttons needed. MAME also has an option to pause, which is something

the original arcade never had, but it's very nice for home use. The yellow button on my control panel is mapped to pause.

Once MAME and WahCade are working, you need to automate the process. Set the GNOME login manager to log in the user account automatically, and create an .xsession file that avoids the GNOME environment and launches WahCade directly. Here's an example of how to create an .xsession file:

```
# cat > ~/.xsession
# wahcade
# ^D (actually press control-D)
#
# chmod +x ~/.xsession (this makes
it executable)
```

So there you have it, an arcade system, complete with real controls and original games. You can customize your system to fit your needs. Perhaps you want a headphone jack, so you can play without disturbing others. Maybe you want to add music support and have your arcade machine double as a jukebox. Because the cabinet has a full computer with Linux under the hood, the possibilities are really endless. Have fun! ■

Shawn Powers is a geeky Technology Director for a small school in northern Michigan. He did manage to find a wife to love him and has three wonderful daughters. His wife even watches *Star Trek* with him, but he suspects it's just because she loves him. Send him e-mail at shawn@brainofshawn.com.

Resources

MAME Arcade Emulator:
www.mame.net

WahCade Front End:
www.anti-particle.com

Arcade Controls (Cabinet Building Info):
www.arcadecontrols.com

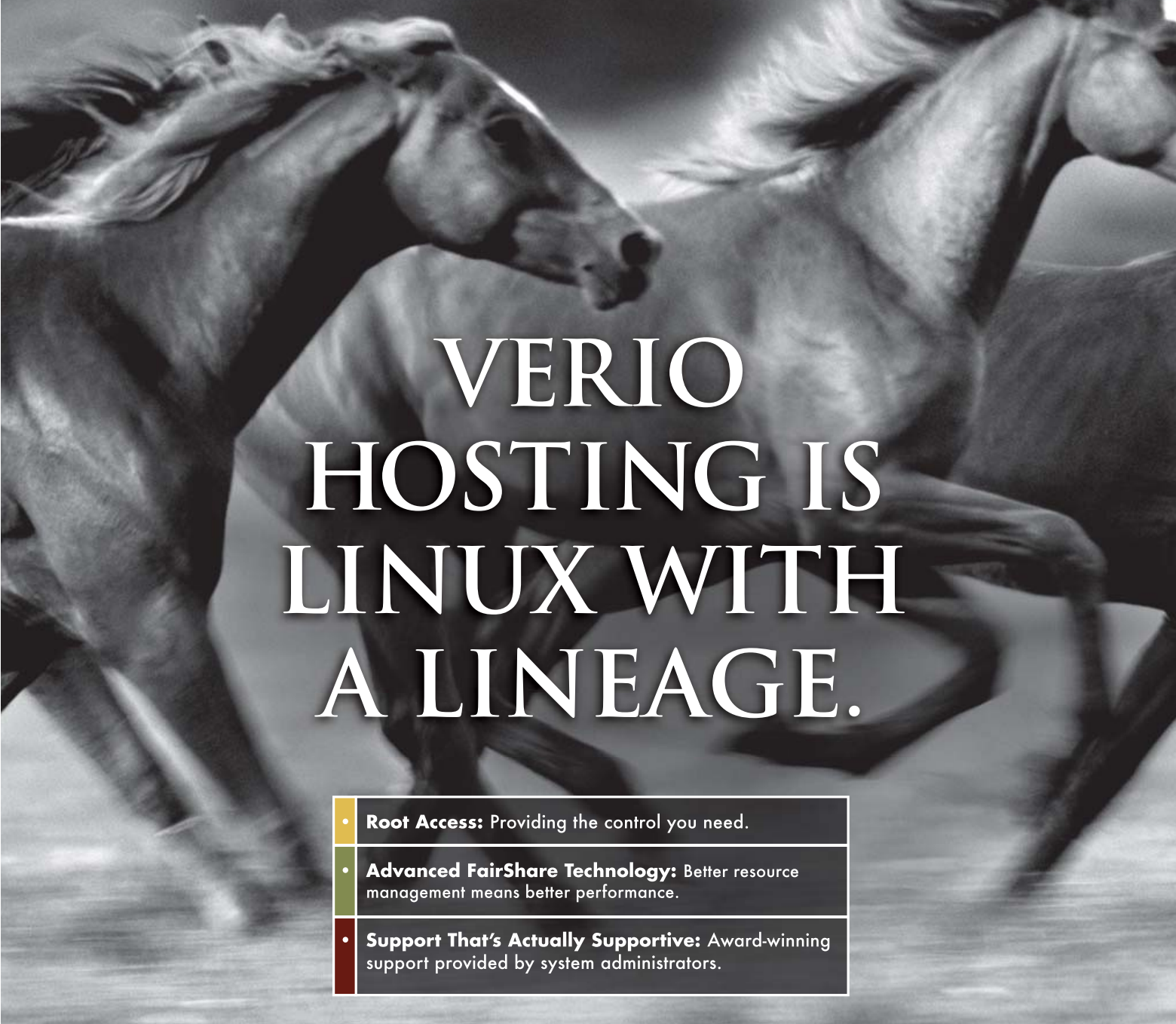
Happ Controls:
www.happcontrols.com

Hagstrom Electronics:
www.hagstromelectronics.com

Ubuntu Linux: www.ubuntulinux.com

AdvanceMAME/AdvanceMENU:
advancemame.sf.net

LinCade: www.pc2jamma.org



VERIO HOSTING IS LINUX WITH A LINEAGE.

- **Root Access:** Providing the control you need.
- **Advanced FairShare Technology:** Better resource management means better performance.
- **Support That's Actually Supportive:** Award-winning support provided by system administrators.

Verio Linux® VPS and MPS: Best of Breed.

At Verio, we have a long-running commitment to open source, dating back to our early work with FreeBSD. Now, as the pioneer in virtual private server (VPS) technology and as a hosting provider backed by the financial resources of the world's largest telecommunications company, we bring something extra to Linux: reliability.

To learn more about Verio Linux VPS or Verio Linux MPS, or to register for a free test-drive, call 1-877-837-4654 or visit www.verio.com/linuxlineage.

Verio and the Verio logo are trademarks and/or service marks of Verio Inc. in the United States and other countries. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries. The mark FreeBSD is a registered trademark of The FreeBSD Foundation and is used by Verio Inc. with the permission of The FreeBSD Foundation. All other names are trademarks or registered marks of their respective owners. ©2007 Verio Inc. All rights reserved.

Build on us.
VERIO
An NTT Communications Company

CREATE A LINUX VPN

for a Nokia E61
with **Openswan**



Create a virtual private network between your Nokia E61 phone and a Linux gateway. Ben Martin

A VIRTUAL PRIVATE NETWORK (VPN) allows you to send traffic across an untrusted network without exposing the content of that traffic. Conceptually, this is done by creating a pipe between two hosts where all network traffic transferred is protected by cryptography.

The example in this article is connecting a Nokia E61 device to a home network through a VPN over the Internet. The Nokia E61 is a smartphone that has Wi-Fi support as well as a VPN client. A similar procedure might work for other phone models using the same VPN client software, though the hardware was not on hand to test this. The Linux side was run on Fedora Linux 6; other distros might have slight path and package name changes.

The VPN support on the Nokia E61 uses IP security (IPSec). Openswan is an IPSec server that is configured on the Linux machine to provide the other end of the virtual network.

I should mention one caveat up front: I've been unable to configure the VPN client on the phone to connect to a server that does not have a static IP address.

To keep notation simple, I refer to the phone as e61 and the server running Openswan as vserv. The IP address of the e61 is irrelevant to

the article, as you likely will be moving around to different Wi-Fi hotspots with the phone. When a VPN is set up, the e61 gets another IP address, which the e61 refers to as the virtual IP address. Once the VPN is set up, this virtual IP address is where all traffic to and from the e61 is sent. For this article, I use a 192.168.x.x IP address for this e61 VPN address. As the non-VPN IP address of the e61 is mostly irrelevant, unless I explicitly mention otherwise, the e61 IP address will be this non-Internet-routable IP address.

Unlike the other network settings on the phone, you cannot configure the VPN manually using the e61 itself. You have to create a package containing all the information about the VPN and install that package on the phone. These packages are the SIS files. A VPN SIS file also must be digitally signed before the e61 will allow you to install it. Signed SIS files normally have an six extension. The most difficult part of setting up the e61 to talk to Openswan is in creating the six file to install on the phone.

The SIS file still must be digitally signed, even if you have set the configuration parameter Software installation to All in App Mgr/Options/Settings.

The Contents of the sixs Package

The sixs package is composed of three files. Two of these are boilerplate-type package metadata (the VPN.pin and VPN.pkg files).

Getting the boilerplate files out of the way, the VPN.pin file is mostly uninteresting and is shown in Listing 1, and the VPN.pkg file is shown in Listing 2. Both files should work fine without any changes. Note that the paths shown in Listing 2 are to be interpreted relative to the phone itself and should not be changed.

Listing 1. Some Very Basic Package Metadata

```
[POLICYNAME]
VPN public
[POLICYDESCRIPTION]
VPN public
[POLICYVERSION]
1.1
[ISSUENAME]
Do not edit
[CONTACTINFO]
Do not edit
```

The VPN.pol file shown in Listing 3 defines the meat of how to connect to the VPN and what key to use for authentication.

Some things need to be changed in VPN.pol before using it. The main changes are the static IP address of the Openswan server (192.168.0.1) and the password to use to connect. The server's IP address appears more than once in the configuration file. To avoid any confusion about virtual IP addresses mentioned above, this IP address is the one from which vserv can be reached publicly from the Internet. The

password is in the last field: the KEY. The number is the string length of the key that follows after a space.

If USE_XAUTH is set to true, when establishing the VPN connection the e61 prompts you for a user name and password with which to connect. This provides an additional level of security. In the event that the e61 is stolen, the thief will have to know your user name and password in order to access your VPN.

Openswan can use either PAM or a separate config file to test the user name and password on the server (more on this later).

Creating the sixs VPN Configuration File

Three Windows executables are used to create a signed SIS file. DevCertRequest.exe is used once to create a certificate to sign the SIS file; makesis.exe and signsis.exe then are used to create the package and sign it. The last two commands are part of the S60 SDK available for free from Nokia's Web site. All of these Windows executables can be run in Wine, though you need to have MFC42.DLL and MSVCP60.dll available to run DevCertRequest.

It's best to get the certificate in order to begin; register for free at sybiansigned.com, and download the DevCertRequest executable. Registration requires

your name, e-mail address, organization, address and phone number.

DevCertRequest is used only to input a few settings and generate a key and a certificate sign request (.csr file). Unfortunately, the DevCertRequest executable is actually an installer, so you

Listing 2. Package manifest and description about the application type. You can change the "VPN Policy" string that is right before 0x1000597E to something else.

```
:"VPN public"
&EN
%{"VPN public"}

;
; A VPN POLICY PACKAGE
;

; LANGUAGES
; - None (English only by default)

; INSTALLATION HEADER
; - Only one component name is needed
; to support English only
; - UID is the UID of the
; VPN Policy Installer application
#{"VPN Policy"},(0x1000597E),1,0,1,TYPE = SA

; LIST OF FILES

; Policy file
"VPN.pol"- "C:\System\Data\Security\Install\VPN.pol"

; Policy-information file
; - NOTE: The policy-information file
; MUST be the last file in this list!

; - FM (FILEMIME) passes the file to the
; respective MIME handler
; (in this case, the VPN Policy Installer
; application).
"VPN.pin"- "C:\System\Data\Security\Install\VPN.pin",
FM, "application/x-ipsec-policy-info"

; REQUIRED FILES
; - The VPN Policy Installer application
(0x1000597E), 1, 0, 0, {"VPN Policy Installer"}

[0x101F7961], 0, 0, 0, {"S60ProductID"}
```

have to install this application and then execute it (Figure 1). For this article, DevCertRequest_30_10_2006_v2.0.exe was used.

After all the pain of installing DevCertRequest, using it consists of five simple steps, and it isn't needed again afterward. You give the location for the new csr file (monkeyiq.csr); the location for your new private key (monkeyiq-private-key.key) and

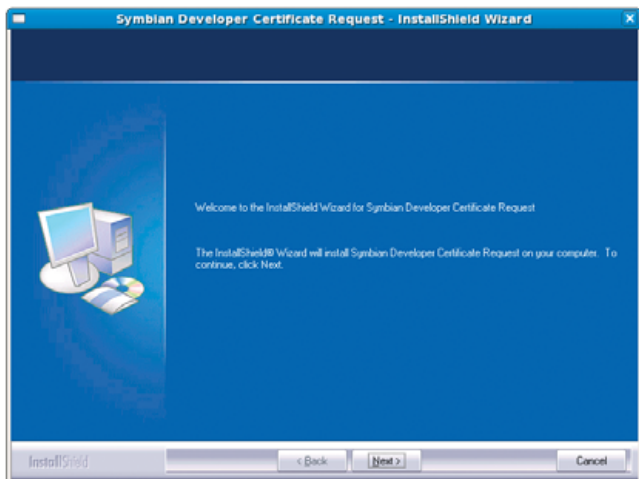


Figure 1. Installing the DevCertRequest Application in Wine

Listing 3. Policy for the VPN: Where to Connect, How to Do It and How to Tell It's Me

```
SECURITY_FILE_VERSION: 3
[INFO]
VPN
[POLICY]
sa ipsec_1 = {
    esp
    encrypt_alg 12
    max_encrypt_bits 256
    auth_alg 3
    identity_remote 0.0.0.0/0
    src_specific
    hard_lifetime_bytes 0
    hard_lifetime_addtime 3600
    hard_lifetime_usetime 3600
    soft_lifetime_bytes 0
    soft_lifetime_addtime 3600
    soft_lifetime_usetime 3600
}

remote 0.0.0.0 0.0.0.0 = { ipsec_1(192.168.0.1) }
inbound = { }
outbound = { }

[IKE]
ADDR: 192.168.0.1 255.255.255.255
MODE: Main
SEND_NOTIFICATION: TRUE
ID_TYPE: 11
FQDN: MobileGroup
GROUP_DESCRIPTION_II: MODP_1536
USE_COMMIT: FALSE
IPSEC_EXPIRE: FALSE
SEND_CERT: FALSE
INITIAL_CONTACT: FALSE
RESPONDER_LIFETIME: TRUE
REPLAY_STATUS: TRUE
USE_INTERNAL_ADDR: FALSE
USE_NAT_PROBE: FALSE
ESP_UDP_PORT: 0
NAT_KEEPALIVE: 60
USE_XAUTH: TRUE
USE_MODE_CFG: TRUE
REKEYING_THRESHOLD: 90
PROPOSALS: 1
ENC_ALG: AES256-CBC
AUTH_METHOD: PRE-SHARED
HASH_ALG: SHA1
GROUP_DESCRIPTION: MODP_1536
GROUP_TYPE: DEFAULT
LIFETIME_KBYTES: 0
LIFETIME_SECONDS: 28800
PRF: NONE
PRESHARED_KEYS:
FORMAT: STRING_FORMAT
KEY: 3 foo
```

a password for it; your country, state and company; the IMEI of your phone (as DevCertRequest tells you, keying *#06# on the phone will show it) and which capabilities you want for your certificate; and a confirmation that the information is correct.

To create the certificate itself, you have to return to **sybiansigned.com** and upload the csr file. First log in, and then select the My Symbian Signed tab. In the side panel, the Developer Certificates option has the Request sub-option. At the bottom of this page, you can upload the csr file (Figure 2). The next page allows you to download your certificate (Figure 3).

The makesis.exe and signsis.exe files can be extracted from the "S60 Platform for Symbian OS" SDK, as shown in Listing 4.

With the certificate (monkeyiq.csr) file, you now can roll the sixx file with the code shown in Listing 5. Make sure the three files that make up the package use the carriage return plus new line combination to terminate each line instead of the standard Linux new line only; see unix2dos(1). These three files are the pol, pkg and pin files shown in Listings 3, 2 and 1, respectively.

Installing the VPN sixx and Final e61 Setup

Any method can be used to transfer the sixx file to the e61. I've used Bluetooth push, in which case it can be installed on the e61 directly from the incoming messages list. As this sixx file contains a password, it is better to transfer it to the phone using a wired method.

Using a mini-SD card in the e61 and plugging in the USB connection cable to the phone and a Linux machine likely will bring up a file browser for the mounted SD card on the e61. Copy the file to a convenient location, such as Documents/vpn on the e61, and eject or unmount the SD card to

Listing 4. Extracting the makesis and signsis Executables

```
$ mkdir /tmp/e61-sdk
$ cd /tmp/e61-sdk
$ unzip /tmp/S30_3rded_f0__S60-SDK-0548-3.0.zip
$ unshield x data2.cab
$ cd Epc32/tools
$ mkdir -p ~/e61tools
$ cp makesis.exe signsis.exe ~/e61tools
```

Listing 5. Rolling the Signed Package File

```
wine makesis.exe VPN.pkg VPN.sis
wine signsis VPN.sis VPN.sisx monkeyiq.cer \
monkeyiq-private-key.key my-certificate-password
```



Figure 2. Uploading the Certificate Request

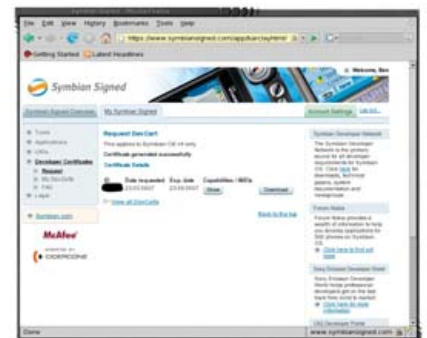


Figure 3. Download the Certificate Needed to Sign SIS Files

force a disk sync before removing the cable (Figure 4).

Once the sixx file is copied to the e61 memory card, the Menu/Office/File Manager on the e61 lets you navigate to the VPN directory on your memory card. When you click the joystick on the VPN sixx file, the phone asks if

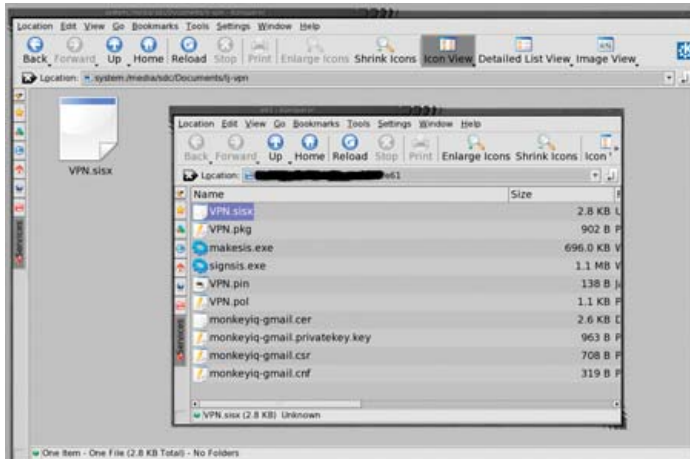


Figure 4. Copy the VPN file to a subdirectory of Documents on the memory card.

Directing all Web traffic to the VPN has the added bonus that the Wi-Fi hotspot you are using isn't able to record the Web sites you visit.



Figure 5. Starting the VPN Policy sisx Install

you want to install it. Right after clicking on the sisx file, you should see something like that shown in Figure 5. After inspecting some metadata, you'll see the ominous-looking screen shown in Figure 6. As you have just created the package from a bunch of text files and you've looked over them for nasties, this security warning shouldn't really be much of a problem to ignore.

The VPN sisx file can be prepared for use by going to Menu→Tools→Settings→Connection→VPN. Select VPN access points and Options→New access point. Set the connection name to something memorable, and set the policy name and access point. A convenient access point is EasyWLAN. You also might want to set the proxy server address and port. It's nice to be able to surf the



Figure 6. As we made the package, we trust that this is not really a security issue.

Internet and get to Intranet servers over the VPN. Directing all Web traffic to the VPN. Directing all Web traffic to the VPN has the added bonus that the Wi-Fi hotspot you are using isn't able to record the Web sites you visit. The final stage is shown in Figure 7.

If you are already using WEP to connect locally and want to continue to do so and be able to test the VPN locally, define another VPN access point, setting its Internet access point to your WEP access point. Having the second VPN config means you won't be prompted for the WEP key when connecting locally. There is little gain in doing VPN over WEP except for not having to loosen the security on your wireless access point.

Setting Up the Linux Openswan End

Packages are shown at rpmseek.com for Fedora, Mandriva and SUSE Linux. Debian.org also lists an Openswan package. On a Fedora Linux machine, Openswan can



Figure 7. The final setup—give a name and Wi-Fi access point to the policy, and maybe set a proxy server too.

be installed simply by using `yum install openswan`. As mentioned previously, I used a Fedora Linux machine for this article; other distributions may have subtle differences.

The two main areas for configuring Openswan are the `/etc/ipsec.conf` file and a handful of files in `/etc/ipsec.d`. The main config file can be left as it stands. A few settings that might be of interest are `forwardcontrol=yes` to turn packet forwarding on and off when Openswan is started and stopped. The other interesting option is the `interfaces` setting, allowing you to control which IPSec interface is bound to which network interface—for example, `interfaces="%default route ipsec2=eth1 ipsec3=ppp1"`. If no `interfaces` parameter is specified, Openswan works on the network interface that has the default route. For Internet VPN connections, this is fine.

Another parameter that might come in handy in the `ipsec.conf` file is setting

Listing 6. The Openswan Configuration File to Allow the e61 to Connect

```
conn e61
    # Key exchange
    ike=aes256-sha1-modp1536
    # Data exchange
    esp=aes256-sha1
    # Authentication method PSK
    authby=secret
    auto=add
    keyingtries=3
    # Modeconfig setting
    modecfgpull=yes
    pfs=no
    rekey=no
    leftid=@monkeyiq.example.org
    left=%defaultroute
    leftsubnet=192.168.0.1/0
    leftsasigkey=none
    leftmodecfgserver=yes
    leftxauthserver=yes
    rightsasigkey=none
    right=%any
    rightxauthclient=yes
    rightmodecfgclient=yes
    rightsourceip=192.168.6.252
    rightsubnet=192.168.6.252/32
```

Listing 7. Private Key for the VPN

```
: PSK "foo"
```

plutodebug=all, and reading your syslog files if you can't connect.

To describe a connection to Openswan for the e61, create a config file `/etc/ipsec.d/e61.conf`, as shown in Listing 6. The `pfs` setting is for perfect forward security. Unfortunately, I've had no luck using this option and connecting from the e61. As shown in the VPN config for the e61, I've listed the left value as `%defaultroute`, so Openswan will substitute the IP address of the network interface to which the default route points. As the default route is to the Internet, this works well. I've also used the DNS name of the `vserv` as `leftid`; this should be optional. You need to substitute your DNS name for `monkeyiq.example.org` in the config file. The `rightsouceip` is the virtual IP address that the e61 will use when talking over the VPN. For the firewall rules (shown later), I have assigned the hostname for the e61 to `192.168.6.252` in `/etc/hosts`.

The same private key that was specified in the `KEY` field of the VPN policy above should be placed into the `/etc/ipsec.d/e61.secrets` file,

shown in Listing 7.

Finally, the `USE_XAUTH` option in the VPN policy needs Openswan to have a user name and password lookup for this connection. The Openswan `README.XAUTH` file recommends against using PAM for this. The password file can be created using `htpasswd` from the Apache package, as shown in Listing 8. A sample of the `passwd` file is shown in Listing 9.

A Hole in the Wall—iptables

Trying to debug packet logs on the machine that is running as the IPsec server can be a little difficult. Packets that arrive encrypted are decrypted and put onto the network interface to appear as though they have arrived without any encryption. Part of a packet log is shown in Listing 10, showing a packet that appears to have come from the e61's IP address without any encryption. The log is further complicated because the WEP setup gives the e61 the same IP address as the VPN does. For traffic from the Internet, the top two packets will have a random IP address instead of the e61. To get a clearer picture of the packet data, connecting the e61 through a laptop to `vserv` allows proper packet snooping on the laptop. The `imaps` packet in Listing 10 will not be seen by the laptop—only a bidirectional stream of ESP packets.

The `iptables` commands shown in Listing 11 provide a base to allow the e61 to connect to the IPsec server from the Internet.

The packet filtering rules are quite simple. Allow Internet Security Association and Key Management Protocol (ISAKMP) packets to enter and exit the server and allow any Encapsulating Security Payload (ESP) traffic, assuming that the IPsec server will be responsible for sorting out fraudulent packets. VPN traffic itself is sent through the ESP packets; the ISAKMP is used at the VPN session startup,

Listing 8. Making the initial `passwd` file with `htpasswd`. The `-c` option creates the `passwd` file if it doesn't exist or replaces it if it does. Use the `-c` only once. Make sure your `umask` is set first, as some distributions have lax defaults.

```
umask 0027
htpasswd -b -c passwd ben your-password-here
htpasswd -b passwd chidori her-password-here
```

Listing 9. The XAUTH user name and password lookup file—the password itself is `linuxjournal`.

```
ben:n0ta0uj0p2G4g:e61
chidori:.....e61
```

Listing 10. Partial Traffic Log for `eth0` on the IPsec Server

```
15:58:07.n IP ipsecserv > e61: ESP(spi=...), len...
15:58:07.n IP e61 > ipsecserv: ESP(spi=...), len...
15:58:07.n IP e61.57397 > ipsecserv.imaps: . ack...
```

Listing 11. Punching a little hole in the firewall. Note that the e61 is set in `/etc/hosts` to `192.168.6.252`.

```
iptables -X REMOTEVPN_INPUT 2>/dev/null
iptables -X REMOTEVPN_OUTPUT 2>/dev/null
iptables -N REMOTEVPN_INPUT
iptables -N REMOTEVPN_OUTPUT

iptables -I INPUT -j REMOTEVPN_INPUT
iptables -I OUTPUT -j REMOTEVPN_OUTPUT

iptables -A REMOTEVPN_INPUT -p esp -j ACCEPT
iptables -A REMOTEVPN_INPUT -m udp -p udp \
    --dport isakmp -j LOG \
    --log-prefix "incoming-ipsec-key "
iptables -A REMOTEVPN_INPUT --src e61 \
    -p tcp --dport imaps -j LOG \
    --log-prefix "incoming-imaps "
iptables -A REMOTEVPN_INPUT -m udp -p udp \
    --dport isakmp -j ACCEPT
iptables -A REMOTEVPN_INPUT --src e61 -p tcp \
    --dport imaps -j ACCEPT
iptables -A REMOTEVPN_INPUT --src e61 -p tcp \
    --dport smtp -j ACCEPT
iptables -A REMOTEVPN_INPUT --src e61 -p tcp \
    --dport squid -j ACCEPT
iptables -A REMOTEVPN_INPUT --src e61 \
    -j LOG --log-prefix "e61-strange "

iptables -A REMOTEVPN_OUTPUT -p esp -j ACCEPT
iptables -A REMOTEVPN_OUTPUT -m udp -p udp \
    --sport isakmp -j LOG \
    --log-prefix "outgoing-ipsec-key "
iptables -A REMOTEVPN_OUTPUT -m udp -p udp \
    --sport isakmp -j ACCEPT
```


Listing 12. Remove the e61 access.

```
iptables -D INPUT -j REMOTEVPN_INPUT
iptables -D OUTPUT -j REMOTEVPN_OUTPUT
```

and those packets also are logged, so that the syslog monitor can alert you of strange connection attempts or door knocking.

Because the non-encrypted traffic is placed on the network interface when Openswan is done with it, the rules have to allow e-mail and squid connections from the e61 IP address. I need to have these rules here because normally no connections can be initiated on the network interface connected to the Internet. If you filter outward traffic too, you have to allow packets from these services to the e61 to be sent to the Internet network interface (to be encrypted by Openswan before being sent to the proper Internet address).

The firewall rules are designed to be used with a default policy of drop. The logging commands can be added or removed to help debugging by searching for the relevant log prefix in `/var/log/messages` to see which packets are moving around before the firewall may drop them. The script shown in Listing 12 undoes what was done by Listing 11 to disable remote access again.

You also might want to consider using a Single Packet Authorization (SPA) client on the e61 and setting up the server to open the firewall ISAKMP port only after a successful SPA. See Michael Rash's "Single Packet Authorization" article in the April 2007 issue of *Linux Journal* for more information on SPA.

Connecting!

One more complication exists for using some of the publicly available Wi-Fi hotspots. Depending on where on the globe you are, many of these hotspots follow the pattern that when you try to open a Web site, they redirect you to their Wi-Fi login page, you authenticate to them, and then you can use the Internet. If you simply open up a VPN access point on the e61 that is set to use the EasyWLAN as its Internet access point, things will not work. The e61 will start the Wi-Fi connection and immediately try to send data to set up a VPN connection. As you have to authenticate with the Wi-Fi hotspot before this, it will let traffic through, but then things will come crashing down.

A way to get around this is to open the Web browser and directly connect just using EasyWLAN without any VPN whatsoever. Once you have authenticated to the hotspot, leave

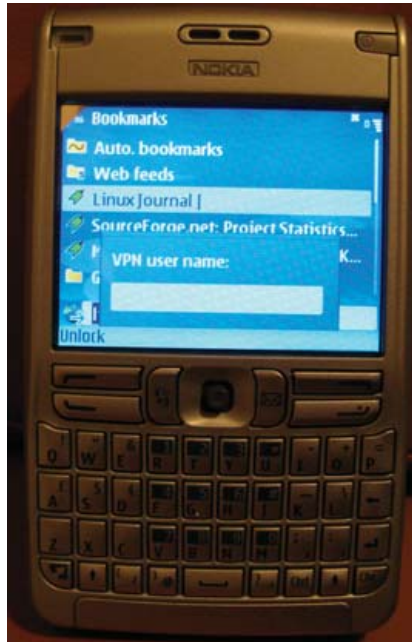


Figure 8. Once the VPN is set up, XAUTH user name and password verification starts.

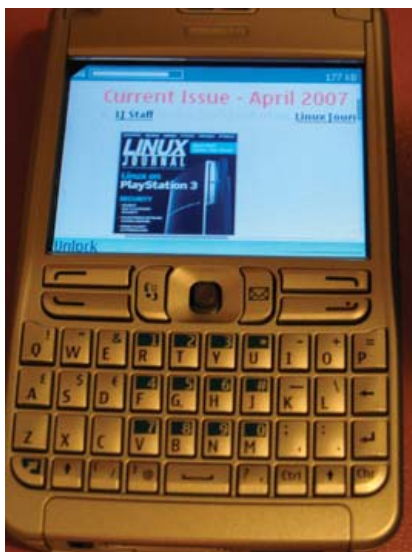


Figure 9. Success! We can read *Linux Journal* through the VPN link.

the browser running and use the menu key to get back to the main menu, and then open the e-mail client. For the access point this time, use the VPN that has EasyWLAN set as its Internet access point. The existing Wi-Fi connection is reused, and the VPN is layered on top. To get secure Web browsing, you can then leave the e-mail program by holding the menu key and going back to the browser. Exit the browser, and the still-running e-mail program holds the VPN open. Start the browser again, and select the VPN as your access point.

Of course, if the Wi-Fi network you are

connecting to allows connections without this preamble, opening any application that wants a data connection should allow you to select the new VPN as your access point. Also, if the Wi-Fi hotspot remembers your MAC address and allows reconnection without explicitly having to log in each time, you can start the VPN directly on subsequent connections.

Once the VPN has connected to vserv, the e61 prompts you for the user name and password to use for XAUTH verification (Figure 8).

After XAUTH verification, you should be able to use the VPN without noticing it. In this case, I can browse the Internet using my LANs proxy server to fetch the data (Figure 9).

What's Next

Being able to use a DNS name in the e61 VPN policy would be wonderful for folks who don't have cheap access to static IP addresses. I'm still investigating how to connect using public key cryptography instead of the preshared key as shown in this article. For connecting a single e61 to the network, using a large enough preshared key should still be quite secure.

The information in the article comes with no guarantee of being correct, secure or suitable for anything; use it at your own risk and discretion. ■

Ben Martin has been working on filesystems for more than ten years. He is currently working toward a PhD combining Semantic Filesystems with Formal Concept Analysis to improve human-filesystem interaction.

Resources

Symbian Signed: symbiansigned.com

Symbian SDK on Unix HOWTO:
www.koeniglich.de/sdk2unix/symbian_sdk_on_unix.html

Symbian SDKs:
www.forum.nokia.com/main/resources/tools_and_sdks/index.html

Nokia VPN Client How-To: pipip.de/index.php?section=other&sub=nokia_vpn_en

Wine and MFC42.DLL:
bugs.winehq.org/show_bug.cgi?id=4461

"Single Packet Authorization" by Michael Rash (*Linux Journal*, April 2007):
www.linuxjournal.com/article/9565

Magnatune

an Open Choice;

iTunes

an Expensive Choice



Finally, a company that understands Internet media distribution. JAMES LEES

On April 2, 2007, Apple CEO Steve Jobs along with EMI CEO Eric Nicoli announced DRM-free downloads at a new cost of \$1.29 US—a fee that's \$.30 more than the current DRM-formatted music. Shouldn't music be DRM-free anyway and not come at an extra cost? That's not what iTunes is saying or showing, as it still is going to sell DRM music for \$.99 and DRM-free music for even more. Why put up with these over-priced songs, not only from iTunes but also from other on-line stores, such as Microsoft's MSN Music that sells over-priced music, and like iTunes, the music comes with DRM.

The new DRM-free songs on iTunes will be in MP3 and WMA formats and supposedly will be of "superior quality". That mention of superior quality, however, is merely to cover up the extra price and save iTunes from the UK's fair trading pressure of infringement on European trading regulations. Why should you have to deal with a company that makes you pay more for freedom? Freedom is not something you should have to pay for. Why pay for freedom when you can choose an on-line site that gives you options the other record-label companies don't offer?

Such a site exists, and it gives you many choices, including how you want to receive an album, such as a downloaded DRM-free format (MP3, WAV, OGG, FLAC or AAC) or a purchased CD that will be mailed to you, all without the extra "freedom fee". This site also lets you choose to pay what

you think a particular album is worth, and it gives you the option of re-downloading a purchased song if you lose it due to a faulty hard drive or misplaced file. This site also gives you the option to back up music on another device, without restrictions on the number of devices on which you can store your music—something iTunes does not allow, and it's something I know I want to be able to do with music I purchase. And, the best thing of all, this site allows you the freedom to share an album and give the gift of music to your friends. If you buy an album, you can share it with up to three friends (based on the honor system).

What if I tell you it's all possible with Magnatune—the company that says "We Are Not Evil", and it's just that, not evil. Magnatune allows customers to do everything mentioned above—plus more. It's not only an independent, on-line record label, but also a company that wants to change how music is distributed and open up possibilities for less-known bands and artists. The artists and bands on Magnatune are not highly rated with albums or songs in the Top 100, but they still are great to listen to and download. All Magnatune needs is only one top artist to switch over to put it on the map as a mainstream label company. Magnatune takes big strides to label great-sounding artists, and one out of 300 artists gets signed.

What Magnatune Believes

This is straight from Magnatune's site:

- All music should be shareware. Just as with software, you want to preview, evaluate and pass along good music to others—in the process of buying it.
- Find a way of getting music from musicians to audiences that's inexpensive and supports musicians. Otherwise, musical diversity will continue to suffer greatly under the current system where only mega-hits make money.
- Musicians need to be in control and enjoy the process of having their music released. The systematic destruction of musicians' lives is unacceptable: musicians are very close to staging a revolution (and some already have).
- Creativity needs to be encouraged; today's copyright system of "all rights reserved" is too strict. We support the Creative Commons "some rights reserved" system, which allows derivative works, sampling and no-cost noncommercial use.

Magnatune in the Open Source Community and Becoming a Freedom Fighter for Music and Media

Magnatune recently has taken part in the Linux community, allowing you to search and listen to its podcast and music directories without purchase from the KDE AmaroK 1.4.4 player. This try-before-you-buy philosophy gives you the opportunity to listen to not just one song from an album, but the full album, before deciding how much you are willing to pay for it—if you choose to purchase it at all. The great thing with Magnatune's search-and-buy feature being put into the AmaroK player is that it allows you to purchase an album from the player instead of going to the site, making it easy to search Magnatune's massive album

"Listen to more than 500 hand-picked complete albums. If you like what you hear, download an album for as little as \$5 (you choose the price), or buy a real CD, or license our music for commercial use. You'll get MP3s and WAVs, and no copy protection (DRM), ever." (From magnatune.com.)

collection for specific genres, such as Classical, Electronica, Jazz and Blues, Metal and Punk, New Age and Rock. Once your purchase is complete, the song or album can be put into any type of music player, such as an iPod or any other MP3 player. As I said before, Magnatune lets you pay what you think the album is worth. It does have a starting price of \$5, with a maximum of \$18 (the typical purchasing price for an album is \$8). Each song is DRM-free (at the price of about \$.50 a song) and can be downloaded in many formats to work on just about any music player and device. The best part is that 50% of the sale price of each album goes directly to the artist.

Another great thing about Magnatune that blows iTunes out of the water is its daily free song give-away, compared to iTunes weekly song give-away. Some people hear the word free and think something must be wrong or of poor quality, but people who use or develop open-source and free software know that's not the case. What makes Magnatune's daily DRM-free music give-away special is not only freedom, but also that the songs are great to listen to and have amazing sound quality. But, that's not all, Magnatune changes genres every day for the give-away. I plan to visit the site on a daily basis to download the free song and browse for other songs.

Magnatune also has joined and supports the partly open-source Second Life, which has become an on-line phenomenon that allows you to live a second on-line virtual life, while making money on-line and joining groups and meeting new people. If you're a member of the Second Life world, Magnatune allows you to choose a genre to listen to while you're in the game or in Magnatune's furnished tents, giving you a relaxing area to hang out with other Magnatune users, making it a better community yet again. When you teleport to Magnatune's hang-out location, don't forget to get a free Magnatune T-shirt for your avatar. Best of all, Magnatune considers Second Life's playing of its streams to be promotional use of its music and grants you a license to use that music for free. Magnatune also says it will grant you a collecting society waiver, so that if ASCAP or BMI ever asks you to pay for the music you're using on Second Life, you can point them to magnatune.com/info/second_life.

Open Music has become one of the largest projects that's made Magnatune so well known in the music industry and Open Source community. Open Music is music that



Figure 1. The AmaroK Media Player Playing a Magnatune-Provided Preview



Figure 2. "We are not evil."

is shareable, available in "source code" form, allows derivative works and is free of cost for noncommercial use. It is the concept of open-source computer software applied to music. Think of Open Music as open-source software, allowing you not only to edit and sell it, but also to share the wealth with the artist and Magnatune, or just give it back to the community so others can then edit what you edited to make it even better. And, it's all under the Creative Commons, including Magnatune's main site. One example of this Open Music Project is the new AmaroK Live CD Project. Magnatune allowed the AmaroK Project members and community to bundle tracks from the Magnatune store that were licensed under the attribution/noncommercial/share-alike version of the well-known Creative Commons license.

The Business Model behind Magnatune

Besides joining the Second Life community and starting the Open Music Project, Magnatune now has a MySpace site with more than 1,300 friends. Another on-line community site Magnatune uses for advertising and marketing is YouTube. Yes, Magnatune has a YouTube page that has, at the time of this writing, 17 videos that include information on Magnatune. And, you can watch and listen to artists from Magnatune on YouTube, over and over again, without having to purchase or pay a royalty to watch the video. Because those free services are used for marketing, the company and

"This is very exciting news about Magnatune. This is precisely the kind of innovation that will solve the current crisis in music."—Lawrence Lessig, founder and chairman of Creative Commons

artists save money that they would normally need for getting their names and labels out to the public. You also can be a part of the Magnatune's marketing team by requesting free recruiting cards or printing Magnatune mini-posters and handing them out.

Magnatune's founder John Buckman's idea was to target people who listen to music in the background while they do other work (while writing this article, I'm listening to Rocket City Riot, *Last Of The Pleasure Seekers* from Magnatune) or music that gets little radio airplay or major record distribution, but that has a fairly large audience. Targeting those audiences is what keeps Magnatune's business going in the long run.

One of the biggest ways Magnatune stays in business is the commercial licensing it offers for the music it sells. It allows the music to be



Figure 3. Magnatune's Web Site

used in films and television, on the Internet and for presentations. And, the license contracts are royalty-free, meaning you don't have to pay more if a project is successful. The license and the cost to use the music depend on the project. Magnatune gives you a price quote in a matter of seconds from the site, and the prices are a lot better than other commercial licensing fees and are 30% lower than industry standard.

Music is not the only way Magnatune makes money. It sells other merchandise, such as posters, clothing and mugs. As with the

albums, 50% of those sales go to the artist.

Running on Open Source

Magnatune does not just support open source, it also runs on it. Its servers run Linux with the most widely available HTTP server on the Internet—Apache 2 supporting and using PHP and OpenSSL. Magnatune uses a MySQL database to store and log purchase information, and it's also used for searches on the main site, such as for an artist or track. When the search is done, a Perl script creates the track listings and playlists so transactions can be made.

Conclusion

So, if you're an open-source advocate and like to listen to music that's not mainstream while supporting artists instead of putting money in the label companies' pockets, visit Magnatune's Web site and sign up for the free song of the day and maybe buy some albums while you're there. And, if you're an artist, Magnatune is just the place to check out, especially if you want to get your feet wet in the music industry. Magnatune is there to help artists reach audiences and receive a profit (note that albums must have at least 40 minutes of music). Then, you can listen to your album on Amarok's open-source media player while making an average of \$1,500 US (according to Magnatune) if your album is accepted. So let's forget about iTunes and make the migration to a community-based project with a great business model. ■

James Lees currently is a Web site developer, and he's about to go to college in Network Engineering. He's also a computer hobbyist and die-hard Linux user who wants to help promote freeware and open-source software. And, let's not forget, he's also a longtime *Linux Journal* reader.

Resources

Magnatune's Main Site:
www.magnatune.com

Magnatune in Second Life:
www.magnatune.com/info/second_life

Magnatune's Open Music Section:
www.magnatune.com/info/openmusic

Magnatune's MySpace Site:
myspace.com/magnatune

Creative Commons:
creativecommons.org

Second Life's Main Site: secondlife.com

A large advertisement for Carinet Linux servers. The background is blue with white and yellow text. At the top, it says 'DEDICATED SERVERS' in large green letters and 'Total Linux Support' in blue. Below this are several Linux distribution logos: Fedora, SUSE, Ubuntu, CentOS, and Debian. The Carinet logo is prominently displayed in the center. Below the logo, it lists server specifications: 'STARTING AT 1GB DDR400 RAM — 160GB SATA2 HDD INTEL BOARDS & CPUS 100MBPS DEDICATED CISCO PORT 1300GB THROUGHPUT INCLUDED'. The price '60\$' is shown in large white and blue numbers. At the bottom, it says 'CARI.NET/LJ' and '888.221.5902'.

MORE HOURS IN THE DAY?

NO. BUT WITH THE DUAL-CORE INTEL® XEON® PROCESSOR INSIDE YOUR SERVERSDIRECT SYSTEM, YOU GET THE NEXT BEST THING: THE POWER TO DO MORE IN A SINGLE SERVER.

1U Twin™ Innovation



- * High Density Computing Technology
- * Reducing Cost, Energy and Space Requirements
- * Support up to **16 processor cores** Quad Xeon 5300 Series

SDR-6015T-TB 1U Data Center Clustering Server

Two systems (nodes) in a 1U form factor. Each node supports the following:

- * Dual-processor Quad & Dual Core Intel® 64-bit Xeon® Support
- * Up to 32GB DDR2 667 & 533 SDRAM Fully Buffered DIMM (FB-DIMM)
- * 2x Intel® (ESB2/Gilgal) 82563EB Dual port Gigabit Ethernet Controller
- * 2x Hot-swap SATA Drive Bays
- * 900/980W High-efficiency Power Supply



STARTING PRICE **\$959.99**

SDR-1105T 1U ENTRY LEVEL SERVER

Excellent general purpose server for organizations with the need for a low, entry level price

- * 1U Rackmount Chassis with 520W power supply
- * Supermicro X7DVL-L Server Board with Intel® 5000V (Blackford VS) Chipset
- * Intel® Dual-Core Xeon Processor 5050 3.0GHZ 667 MHz
- * Total 512MB, 2pcs x 256MB Kingston DDR2 533Mhz FB-DIMM ECC
- * Seagate SATAII 80GB 7200 RPM 8MB Cache SATA 3.0Gb/s Hard Drive
- * 4 x 1" Hot-swap SATA Drive Bays
- * Two Intel® 82563EB Dual-port Gigabit Ethernet Controller
- * Intel® ESB2 SATA 3.0Gbps Controller RAID 0, 1, 5, 10 support



STARTING PRICE **\$1,199**

SDR-2503T 2U APPLICATION SERVER

Highest performing with Dual Core/ Quad Core Xeon CPU based. Excellent with general purpose applications and provide the most power.

- * 2U Rackmount Chassis with 650W power supply
- * Supermicro X7DVL-E Server Board with Intel® 5000V (Blackford VS) Chipset
- * Intel® Dual-Core Xeon Processor 5050 3.0GHZ 667 MHz
- * Total 512MB, 2pcs x 256MB Kingston DDR2 533Mhz FB-DIMM ECC
- * Seagate SATAII 250GB 7200 RPM 8MB Cache SATA 3.0Gb/s Hard Drive
- * 6 x 1" Hot-swap SATA Drive Bays
- * Intel® (ESB2/Gilgal) 82563EB Dual-port Gigabit Ethernet Controller
- * Intel® ESB2 SATA 3.0Gbps Controller RAID 0, 1, 5, 10 support



STARTING PRICE **\$2,199**

SDR-3500T DATABASE SERVER

Easily Scalable storage solution with hot-swap functionality for growing businesses

- * 3U Rackmount chassis with Redundant 800W power supply
- * Supermicro X7DBE+ Server Board with Intel® 5000P (Blackford) Chipset
- * Intel Quad-Core Xeon Processor E5310 1.6GHZ
- * Total 1024MB, 2pcs x 512MB Kingston DDR2 533MHz FB-DIMM ECC
- * Seagate SATAII 500GB 7200 RPM 16MB Cache SATA 3.0Gb/s Hard Drive
- * 16 x 1" Hot-swap SATA Drive Bays
- * Dual-port Gigabit Ethernet Controller
- * Intel SATA 3.0Gbps 6-PORT Controller RAID 0, 1, 10 support



STARTING PRICE **\$3,299**

SDR-5111T 5U ADVANCED STORAGE SERVER

Quad Core dual Xeon CPU based, with 24 hot-swap hard disk bays suitable for 18TB of pure data Storage capacity

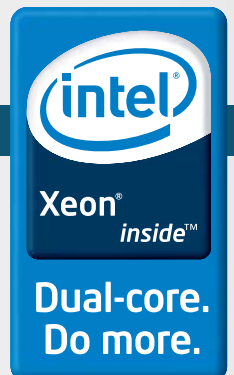
- * 5U Rackmount chassis with Redundant 1350W power supply
- * Supermicro X7DBE Server Board with Intel® 5000P (Blackford) Chipset
- * Intel Quad-Core Xeon Processor E5310 1.6GHZ
- * Total 1024MB, 2pcs x 512MB Kingston DDR2 667MHz FB-DIMM ECC
- * Seagate 750GB 7200 RPM 16MB Cache SATA 3.0Gb/s Hard Drive
- * 24 x 1" Hot-swap Drive Bays
- * Intel® (ESB2/Gilgal) 82563EB Dual-port Gigabit Ethernet Controller
- * Intel ESB2 SATA 3.0Gbps Controller RAID 0, 1, 5, 10 support

SERVERS DIRECT CAN HELP YOU CONFIGURE YOUR NEXT HIGH PERFORMANCE SERVER SYSTEM - CALL US TODAY!

Our flexible on-line products configurator allows you to source a custom solution, or call and our product experts are standing by to help you assemble systems that require a little extra. Servers Direct - your direct source for scalable, cost effective server solutions.

1.877.727.7887 | www.ServersDirect.com

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, Pentium, and Pentium III Xeon are trademarks of Intel Corporation or it's subsidiaries in the United States and other countries.



THE BEST GAME IN TOWN

Kai Staats discusses the power and challenges of a PlayStation 3 supercomputing cluster. JAMES GRAY

In October 2006, Terra Soft announced its plan to build the world's first supercomputing cluster using the Sony PlayStation 3 (PS3), which utilizes the IBM Cell Broadband Engine and the Linux operating system. The idea emerged when Sony Computer Entertainment came knocking on Terra Soft's door, interested in showing that the PS3 is more than merely a game box. After building a 3,000-sq-ft supercomputing facility, located at Terra Soft's headquarters, and adding a heavy dose of good old-fashioned tinkering, the cluster is well underway. Terra Soft's CEO Kai Staats called the building of the PS3 cluster a "highlight of [his] time in this industry". We caught up with Kai recently for an insider's view on the PS3 cluster.

LJ: Thank you for agreeing to talk with us, Kai. Tell us, why did Sony come to Terra Soft to build this cluster?

KS: Terra Soft has, for eight years, dedicated itself to the Power architecture, providing a leading Linux OS for systems built upon the IBM and Freescale CPUs, such as Apple's PowerPC product line. This experience and expertise gave Sony the confidence that Terra Soft would provide a high-quality end-user experience with professional support.

LJ: The PS3 cluster you have created together with Sony is an interesting application of what is marketed primarily as a home-entertainment machine.

The PS3 is really a flexible, powerful machine, isn't it?

KS: Yes, the PS3 is both. I believe we are experiencing an interesting paradigm shift, from three decades of personal computers competing with dedicated game boxes to the industry's first game box offering true personal computer functionality.

Sony recognizes that, with its Cell Processor, the PS3 is not just another image processing engine, but a full-featured, fully capable home computer and lightweight development workstation. This is a tremendous market differentiator.

At home, the PS3 elegantly consolidates the CD, DVD, MP3 player and home computer into a single "appliance". In supercomputing, the PS3 offers an inexpensive, lightweight compute node. Not designed to compete with the Mercury and IBM Cell blades, the PS3 enables individuals and labs to develop and optimize code for this new nine-core architecture within a limited budget. The same code seamlessly migrates to the high-performance Cell products.

LJ: We're curious to know more about the significance of the Cell Processor.

KS: The PS3 is built upon the Cell Broadband Engine, a nine-core CPU designed by Sony, Toshiba and IBM (the STI consortium). It provides an exceptional front-side bus performance.

LJ: Does the Cell's 1+8 multicore processing environment behave like a true eight-core processor, or is there a significant difference?

KS: The first core is the PPU, an IBM 970-compatible unit. This means any Linux application designed to function for the Apple G5 or IBM JS21 (for example) will operate seamlessly on this core. The additional eight SPEs (Synergetic Processing Engines) provide eight additional cores that may be addressed as CPUs (as compared to DSPs), enabling a unique and powerful single-chip processing environment. By keeping the code on the front side (as compared to dropping down to the north bridge as with historic, multiple CPU configurations), the performance is maximized.

LJ: A critical part of realizing the PS3 cluster appears to be Y-HPC, your cluster-construction suite. What are



Aaron Johnson (left) and Kai Staats (right) of Terra Soft

Terra Soft's innovations there?

KS: Simply stated, the Y-HPC cluster-construction suite delivers node images to compute nodes. But, around this core function is the means to manage multiple, unique node images and node "personalities" that modify any given node image to perform various designated tasks. Nodes may be deployed as an NFS server, storage server or compute node (for example), based upon the personality configuration.

Y-HPC integrates a full command-line syntax as well as a graphical user interface. And, Y-HPC may be deployed, both server and nodes, on x86, x86-64 and Power (G3, G4, G5, IBM JS20/21, p5 and Cell). Furthermore, Y-HPC is the first and only cluster-construction suite for IBM, Mercury and Sony Cell.

Although Y-HPC does incorporate some basic cluster node monitoring tools, it is not designed to replace Cluster Resources Moab. Instead, it is designed to integrate with Torque and Moab for a complete build-to-run solution.

Currently in a beta v2.0 release, Y-HPC is being shipped to key customers in addition to its deployment on our internal PS3 cluster.

LJ: How does Yellow Dog Linux play into other Cell-based systems?

KS: In the fall of 2005, Mercury Computer engaged Terra Soft to develop and maintain a commercial Linux OS for its Cell-based systems. This was first announced at SC2005, Seattle (www.terrasoftsolutions.com/news/2005/2005-11-15.shtml).

Mercury began shipping IBM BladeCenter form-factor Cell blades in January 2006 with Yellow Dog Linux pre-installed. Terra Soft continues to maintain and develop Yellow Dog Linux for Mercury's Cell-based products, with forthcoming support for the PCI form-factor CAB and 1U-rackmount form-factor "pizza box" node (mc.com/products/boards.cfm).

LJ: You'll be using this cluster for bioinformatics. Can you explain for our readers what bioinformatics is? Why does this particular cluster lend itself to this application?

KS: Wikipedia explains, "The terms bioinformatics and computational biology are often used interchangeably. However, bioinformatics more properly refers to the creation and advancement of algorithms, computational and statistical techniques, and theory to solve formal and practical problems posed by or inspired from the management and analysis

of biological data."

I would add, at a more basic level, bioinformatics includes the comparison of gene sequences between two or more organisms. For instance, as a bacteria or virus mutates, one or more of its genes differs from the previous strand. Once sequenced (the process by which the genes are identified, labeled and placed into a database), bioinformatics offers a means by which the pre-mutation and post-mutation gene sequences can be compared and better understood.

Y-Bio and the cluster offer a means by which thousands of gene sequences may be compared on a daily basis, in addition to other applications that will be introduced by the Department of Energy and "dot-edu" researchers.

LJ: Originally, Sony contracted with you to build two clusters based on the PS3 platform: a test cluster playfully dubbed E.coli and a production cluster called Amoeba. Is this what finally transpired?

KS: This original plan was to build both a test and production cluster from "beta" PS3 units, the 2U-rackmounts that Sony provided to the game developers prior to shipping hardware. Last fall, Sony determined it

would prefer to use shipping PlayStation 3 units, the same as those found in retail stores worldwide.

This renewed effort was put in motion in January with the first 20 of a slated 128 nodes having arrived a little more than two weeks ago.

The first 20 nodes are on the rack shelves now. Two weekends ago, we created a reduced-footprint Yellow Dog Linux node image of roughly 680MB, just the essentials for a functional, flexible compute node. We are now updating our Y-HPC cluster-construction suite, with the folks from Cluster Resources applying Torque Resource Manager and the Moab Cluster Suite.

LJ: What has your experience been with the cluster thus far?

KS: The only true challenge was in working with a new bootloader (kboot) and the associated ramdisk image for netbooting. An afternoon or two of tweaking and breaking things before we found the magic combination of settings, and the PS3s were up and running as NFS-booting cluster nodes.

LJ: Can you give us an idea of what kind of performance improvements you achieve with the nine-core Cell Processor vs. other CPUs?

KS: During our Hack-a-Thon, we witnessed some interesting advances in code performance on the Cell Processor. In particular, the Mesa library experienced an 80x increase over the published performance on an Intel Woodcrest. Some folks from IBM have been working on BLAST for Cell with a noted 10–20x performance improvement.

LJ: Did you have to do anything special to the Linux kernel to exploit the Cell Processor's multicore architecture and support chips?

KS: We do not modify special kernel interfaces for accessing the Cell's SPEs, which have been included in the Linux kernel for some time now (I can't remember when Cell support was first added). We also include the Cell SDK that allows you to build, execute, run and debug applications that utilize the Cell's SPEs. [This question was answered with support from Owen Stampflee, Lead Developer of Yellow Dog Linux.]

PS3 has just under 256MB of user-space RAM, where the IBM and Mercury Cell blades currently offer 512MB per CPU for a combined, shared 1GB of RAM on a dual-Cell (18-core) board, and far less RAM than

“ I believe we are experiencing an interesting paradigm shift, from three decades of personal computers competing with dedicated game boxes to the industry's first game box offering true personal computer functionality. ”

what we have come to expect of modern 32-bit desktops and 64-bit workstations.

This limited RAM is constraining and, of course, things are not quite as snappy as with more. But, the 3.2GHz CPU and fast front-side bus do compensate well, as the desktop is quite usable. Even large footprint apps, such as OpenOffice.org, are functional. MythTV is impressive. But, with a very large image, The GIMP certainly would take a hit.

When optimizing code for the Cell SPEs, independent of the PS3 or IBM implementation, it is imperative that the algorithms themselves be reworked to enable pipelining, the continuous, steady-streaming of code and data through the relatively small cache. A stall in this pipeline, and performance is lost.

This may be an afternoon of rework, a few weeks or more, depending on the complexity of the code. But when successful, the end result is phenomenal with 32-bit floats taking advantage of both the AltiVec unit onboard each SPE and then the eight-way multicore spread.

LJ: It is impressive to see that you are co-developing and open-sourcing a range of life-sciences applications in

tandem with universities and national labs, such as Lawrence Berkeley, Los Alamos and Oak Ridge. What progress have you made here?

KS: Everything we house in our on-campus, 3,000-sq-ft server room is given to the members of our HPC Consortium (www.hpc-consortium.net) free of charge. This currently includes seven IBM Cell blades (used for Cell code development and optimization) and the growing PS3 cluster interconnected via GbE to two G5s, a devel box and head node. Access is granted via a dedicated port on our fibre drop, which currently sits at 10Mb and scales in minutes to as high as 100Mb, if needed.

We are expecting to receive additional Cell pSeries, perhaps some GPU systems in the not-so-distant future.

All Consortium technical members (those whose proposals for Cell development have been accepted) are granted an account on all in-house systems.

The Consortium is an experiment in leveling the playing field, bringing together developers from all programming backgrounds and engaging them on the same hardware, in the same lists—advanced programmers from IBM working with newbies from universities, DOE lab and commercial employees collaborating during the Hack-a-Thon and then assisting each other with ongoing development.

LJ: Are there other applications besides bioinformatics that you will be targeting?

KS: Absolutely. At the Hack-a-Thon, there were projects to optimize the kernel, to build a Cell development toolset for Windows, the optimization of Mesa (mentioned earlier), visualization libraries and more. There are new projects in motion to bring multimedia apps to Cell, CFD libraries and film rendering. The potential is limited only by the determination of the individuals and teams that work with Cell, and the Consortium by no means limits the effort to a single field of study.

LJ: Thank you for sharing information about this innovative project, Kai. Good luck to you!

KS: Thank you, too! ■

James Gray is *Linux Journal* Products Editor and a graduate student in environmental science and management at Michigan State University. A Linux enthusiast since Slack 1.0 in 1993, he currently lives in Lansing, Michigan, with his wife and Kitty.

MAXIMIZE PROCESSING PERFORMANCE AND MAXIMIZE RESPONSIVENESS.



COST-EFFECTIVE 2U SERVER



STARTING AT:
\$1,599

- Intel® Server System SR2520SAXNA
- Quad-Core Intel® Xeon® Processor 5310
- 1024 MB FB-DIMM ECC Memory
- 400 GB 7200 RPM 16 MB Drive

1U CLUSTER SERVER



STARTING AT:
\$1,999

- Two Nodes in 1U
- Dual-Core Intel® Xeon® Processor 5050
- 1024 MB FB-DIMM ECC Memory
- 250 GB 7200 RPM 16 MB Drive

1-800-576-7931
servers **DIRECT**
GO TO THE SOURCE

ServersDirect can help you configure your next high-performance server system—contact us today!

ServersDirect
1-800-576-7931
www.ServersDirect.com

Our flexible online product configurator allows you to source a custom solution, or call and our product experts are standing by to help you assemble systems that require a little extra. ServersDirect—your direct source for scalable, cost-effective server solutions.



Building a Next-Generation Residential Gateway

You may not need as much as you think to build a residential gateway. ALEXANDER SIROTKIN

Before embedded Linux became the de facto standard for networking devices, building a residential gateway (RG) or similar appliance used to be an expensive and nontrivial task. Real-time operating systems (RTOSes) used on these kinds of devices, such as VxWorks or pSOS, are relatively expensive and lack many features needed for an RG, which has to be purchased separately. VxWorks 5.5, for example, comes with a TCP/IP stack that has performance problems and does not even implement an L2 bridge, not to mention a firewall. This situation created a business opportunity for a number of companies, such as Ashley Laurent and Jungo, who sell RG software stacks for various OSes. The advent of embedded Linux distributions, such as uClinux, reduced the process of building an RG to choosing the right hardware, writing the drivers for peripherals and adding some kind of Web-based configuration utility. Embedded Linux reduced the development and cost of RG devices not only by providing many important features that either were missing or expensive in RTOSes, but also by easing the integration and debugging of drivers and applications, thanks to the clear POSIX driver model and kernel/user-mode separation.

However, Linux dominance in the embedded networking market is now challenged by new technologies. In the past, RG devices with ADSL, DOCSIS and 802.11 peripherals rarely were required to pass throughput beyond 10Mbps. New technologies, such as 802.11n and PON, make 100Mbps throughput a reality, challenging embedded engineers with the task of creating RGs capable of handling much higher traffic on the same or at least similarly priced hardware.

Anatomy of the Residential Gateway

Before designing your own residential gateway, it may be worth taking a closer look at one of these devices.

If you decide to take a look inside your

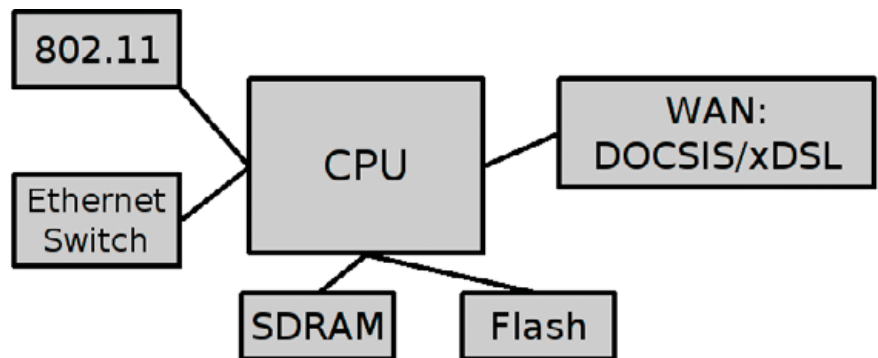


Figure 1. RG Hardware Block Diagram

802.11 access point or ADSL gateway, which probably will void your warranty, you will find an embedded board with a CPU (probably some MIPS variant), Ethernet switch, ADSL/DOCSIS and 802.11 chips, as illustrated in Figure 1.

Some (or even all) of the above can be integrated into one piece of silicon to form a System-on-Chip (SoC). On the other hand, you also may find the same stock mini-PCI (mPCI) 802.11 card connected to an embedded board as the one you will find in your laptop.

Chances are, your home gateway runs Linux. If you'd like to know for sure or maybe hack a bit on it, you have to solder the UART connector, which is relatively easy (most boards come with UART enabled, but without a connector). The software stack usually includes some version of Linux (2.4.x kernels are still pretty common in the embedded world), the usual user-space utilities and libraries, peripheral drivers and a configuration utility—either command-line interface (CLI), Web or both, as shown in Figure 2.

Software

Now that you have a general picture of what makes a residential gateway, and assuming you have working hardware (more details on hardware later in this article), let's see what pieces of software you'll need. At the bare minimum, you need kernel sources, some

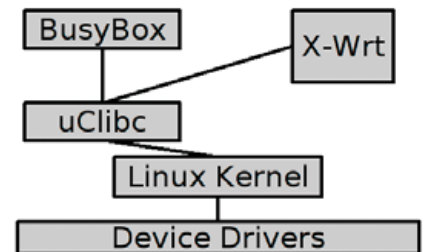


Figure 2. RG Software Block Diagram

variant of the libc library, basic user-mode utilities (at least a shell) and a cross-compiler toolchain to build them all. In short, you need an embedded Linux distribution. uClinux is a good choice, as it supports a wide range of CPUs (not necessarily CPUs without MMU, as the name might imply), comes with many useful user-mode utilities and has an easy-to-use kernel-style configuration system for all the packages. As always, the choice is not limited only to one distribution. There are a variety of free and commercial Linux distributions from which to choose.

Assuming your board (not just the CPU) is supported by uClinux, the process of building a working image boils down to downloading the distribution itself and the cross-compiler toolchain for your CPU and then simply following the build instructions. If your board is not supported by uClinux,

chances are the board manufacturer will provide a board support package (BSP)—that is, adapt uClinux (or some other Linux version) to that hardware. If this is not the case, you have to write the BSP yourself, which is beyond the scope of this article.

The major components of the uClinux distribution are, of course, the kernel itself, the uClibc library and BusyBox. The uClinux kernel has support for MMU-less CPUs, but this feature is of minor importance nowadays. Because the cost of adding the MMU is so small, I expect that most, if not all, embedded CPUs used on an RG will have it. Nevertheless, uClinux is a great distribution, even if you don't exploit the MMU-less CPU support. The same goes for uClibc. It originally was created to support MMU-less systems, which, for instance, cannot have a fork(2) system call. But, even if you don't need this functionality, it is a great alternative for glibc on embedded systems, as glibc has much larger RAM requirements. BusyBox is a collection of standard UNIX utilities, optimized for embedded systems with low RAM. It comes with uClinux, and as with uClibc, you usually will prefer it over full-featured standard utilities, unless you have a system with enough RAM (typically above 16MB).

You need two important pieces of software that are not a part of uClinux. The first piece is a bootloader, which is software that usually resides in ROM (at least partially) and is responsible for loading the Linux image from Flash to RAM and performing some hardware initializations. Unfortunately, there is no standard bootloader for uClinux. In fact, there are no standards for bootloaders in the embedded world, and you cannot use PC bootloaders, such as GRUB or LILO. Your hardware manufacturer almost certainly will provide a bootloader, and I strongly suggest using it. If it does not support Linux, it usually is easier to adopt it than to port a different bootloader to your hardware. If you still choose to port the bootloader, Das U-Boot is a good choice, as are many others.

The second missing piece is a Web-based graphical user interface (GUI), which most users come to expect from RGs. Most, if not all, currently available RGs have a Web interface written from scratch. However, it does not have to be like this anymore with the introduction of the X-Wrt Project, one of the components of the OpenWrt Project, which you almost certainly will want to look at if you are building an RG. OpenWrt is a Linux

**Chip manufacturing,
warehouse automation,
and other throughput-intensive
systems require**

**high
speed**
processing of
c-tree technology

**FairCom database
technology makes
it possible.**



FairCom

www.faircom.com/go/?speed

distribution for the Linksys WRT routers (and not only those).

Add to this a bit of init script writing, and you are done, except for the peripheral (802.11, ADSL and so on) drivers, which is a topic for another article.

Hardware

Now to the real fun—choosing (or even better, designing) your hardware. MIPS is the most commonly used CPU architecture in these kinds of devices, with older devices using the MIPS32 4K cores, and newer devices probably using the MIPS32 24K cores. As MIPS 24K is about 3% slower than MIPS 4K at the same clock rate (because of the longer pipeline), it makes sense to use it only if you really need clock rates of about 400MHz. However, benchmarks show that networking devices are memory- and I/O-bound, not CPU-bound (after all, there is not much number crunching in forwarding packets, with a few exceptions discussed later), so you probably will need these high clock rates only if you are going to use DDR RAM and not SDRAM (which is limited to 133MHz).

Another option is the new (at least in the embedded world) multithreading (MT) technology of the MIPS32 34K, which can have a configurable number of Virtual Processing Elements (VPEs) and Thread Contexts (TCs). Without going into too much detail, this technology is a more flexible equivalent of Intel Hyper-Threading, which helps the CPU exploit parallelism at the process and thread level (compared to parallelism at the instruction level, which superscalar processors can exploit quite well). This can boost performance in some cases, especially when there are a few concurrent processes or threads that are I/O- or memory-bound, but benchmarks show that this may not always be the case. In addition, MT has an overhead—MIPS 34K is slightly slower than MIPS 24K at the same clock rate, and the Linux kernel becomes slightly slower when compiled with symmetric multiprocessing (SMP) enabled.

Another CPU you probably should consider is MIPS' closest competitor—ARM. It is used less frequently than MIPS for these particular devices, mostly for historical reasons. ARM7 is very popular, small and has a low power core; however, it is clearly too slow for an RG. If you decide to go with ARM, you should consider one of the ARM9 or ARM10 families of processors, with the major differences, as far as an RG application is concerned, being perfor-

mance, power consumption and cost.

RAM and Flash choice is obvious; you need at least 8MB of RAM and 2MB of Flash. However, as memory prices are dropping, I expect that future devices will have 64MB of RAM and 4MB of Flash. For the same reason, you probably will have to go with DDR and not SDRAM—as SDRAM actually becomes more expensive than DDR.

Assuming you are going to use one of the reference design boards provided by a manufacturer (board design is beyond the scope of this article anyway), you are ready to test your brand-new RG. Unfortunately, you are in for a big surprise. Your CPU is going to choke at not so unreasonable bandwidths.

One possible solution is to use a more powerful (and significantly more expensive) CPU, such as the Intel IXP or Freescale PowerQUICC network processors. For instance, the 533MHz IXP425 will have no problem sustaining that kind of traffic. Unfortunately, in order to stay competitive, RG manufacturers cannot afford these high-end processors, so a more creative solution is required.

Optimization

This is the challenge of next-generation RGs—achieve high throughput using low-end hardware. One possible solution is to offload the whole data path of the RG to hardware, the way it works on high-end core routers, which usually employ a general-purpose CPU only for management and control. There are a few chips that can do this for the low-end RG market, such as the Realtek RTL8650B/RTL8651B, which can do routing, NAT and firewall in hardware. Of course, the implementation is limited compared to the Linux TCP/IP stack, but the hardware can be configured to trap the CPU in case it encounters a packet it cannot handle, so that most of the packets will be forwarded from one interface to another without interrupting the CPU. However, this approach has a number of problems, the most serious one being the fact that the hardware TCP/IP stack is limited to a fixed interface (MII in case of RTL8650). It would be difficult and, in some cases, impossible, to connect other interfaces, such as 802.11, DOCSIS and xDSL, to that logic. Therefore, I believe that although this approach can work in some specific cases, it is a wrong idea in general.

Typically, it is a good idea to keep to the software as much as possible, because it's easier to develop and debug. Another opti-

mization approach is based on the observation that RGs (and networking applications in general) are memory-bound, so it would be extremely beneficial to improve memory access. Let's separate data and code for the sake of this discussion. As far as the code is concerned, we want to keep it in cache as much as possible. Ideally, we want the whole critical path routines—that is, starting from one driver's receive function via the TCP/IP stack and to the other driver's send function to stay in cache all the time. This is not possible with most embedded processors, which have only a 32K cache. However, it can be shown that the Linux 2.6 critical path—that is, functions used 95% of the time, under firewall and NAT configuration, including Ethernet drivers' send and receive routines, can fit into a 64K cache, and there are embedded processors with 64K on the market. If your CPU does not have that much cache, but instead has scratchpad SRAM, you can modify the Linux linker script to place certain routines in the SRAM memory region.

If you want to test the above observation (or calculate how much cache your particular application needs), use OProfile, which is a system-wide profiler for Linux that allows you to profile user-mode applications, kernel and drivers, and supports many embedded architectures along with objdump (or any other utility) that will show you how much memory each routine requires.

As for the data, it is absolutely necessary to ensure that all network drivers follow zero-copy methodology, and it may be wise to place frequently accessed data structures in a scratchpad SRAM.

Yet another optimization approach is the mixture of the above two—using profiler, find the most CPU-intensive pieces of code and offload this particular functionality to the hardware. As it turns out, the two most CPU-intensive tasks related to networking are IPSec and the UDP/TCP checksum calculation. It is very convenient (and not very surprising) that both have a well-defined architecture for hardware offloading. UDP and TCP checksum offloading is extremely beneficial, because if it is checked in the hardware on receive and calculated in the hardware on transmit, the CPU will never have to bring the whole packet into the cache, significantly reducing the number of memory accesses. IPSec, on the other hand, is less useful, as an RG is rarely an IPSec termination point—usually IPSec (VPN) traffic is passed through and terminated on the PC.

Another approach that I definitely do not recommend, but one being taken by some manufacturers because it is actually cheaper than the ones mentioned above, is to "optimize" Linux by creating various types of "fast paths". For instance, if the L2 bridge performance is not satisfactory, it is possible to pass packets from one network driver directly to the other, eliminating at least one context switch and some other code, resulting in performance gain. Although it will not work in general cases, it does work for an RG, where the manufacturer controls the whole system, including all the drivers and the kernel itself. The biggest problem with this approach is that it actually cripples the stock Linux kernel, limiting functionality and introducing bugs. These modifications rarely are submitted to the Linux-kernel mailing list, and even if they were, they never would be accepted. But, they do go into some products you can find in stores.

Conclusion

Using the steps described above, you should be able to build a Linux-based RG with relatively little effort. If performance becomes an issue, which almost certainly will be the case if you cannot use high-end processors, follow the optimizations guidelines outlined above. And, it's always a good idea to run a profiler on your particular system to discover additional bottlenecks.

Although this article discusses RGs, most of the conclusions and guidelines are true for any embedded networking system.

The issue of Linux optimization for RG systems actually leads to a much bigger and more controversial topic. There seems to be a significant communication problem between the Open Source community and embedded developers working for commercial companies. On one hand, features added to the kernel sometimes hurt performance on small embedded systems. On the other hand, Linux improvements done by some companies do not always find their way back to the main kernel tree, often because they are not done properly. One good example of this miscommunication is the 2.6 kernel itself, which included many important improvements for embedded systems, but suffered some performance degradation. As a result, a significant number of embedded systems still run the 2.4 kernel. The reason for this miscommunication is probably the fact that semiconductor companies that usually do embedded software

development find it hard to embrace the idea of open source, but it also may be due to the fact that the Open Source community is less interested in embedded systems, because they are harder to hack than a PC. I do believe that the first problem eventually will go away, as semiconductor companies understand how they can benefit from open source, and I try to do my share of explaining wherever I can. As for the second problem, one of the messages of this article is that it's easy and pretty cool to hack embedded systems, and you actually may have the hardware already. ■

Alexander Sirotkin works for Metalink Broadband as a software architect. Metalink Ltd. (NASDAQ: MTLK) is a leading provider of high-performance wireless and wireline broadband communication silicon solutions. Alexander has more than ten years' experience in software, operating systems and networking, and he holds MSc and BSc degrees in Applied Statistics, Computer Science and Physics from Tel-Aviv University.

Resources

OProfile: oprofile.sourceforge.net

uClinux: www.uclinux.org

DENX: www.denx.de

Das U-Boot: sourceforge.net/projects/u-boot

uClibc: www.uclibc.org

BusyBox: busybox.net

OpenWrt: openwrt.org

X-Wrt: x-wrt.org

Linux/MIPS: www.linux-mips.org

ARM Linux: www.arm.linux.org.uk

"An Introduction to Embedded Linux Development, Part 1", by Richard A. Sevenich: www.linuxjournal.com/article/7848

"An Introduction to Embedded Linux Development, Part 2", by Richard A. Sevenich: www.linuxjournal.com/article/7911

"An Introduction to Embedded Linux Development, Part 3" by Richard A Sevenich: www.linuxjournal.com/article/8001

Boots Linux in 1.69 seconds!

TS-7300 High Security Linux FPGA Computer



\$219 qty 1 **\$189** qty 100

200 MHz CPU

- Fanless, no heat sink, < 2 Watts
- User-programmable Altera 2C8 Cyclone II FPGA
- PC/104 expansion bus
- 10 COM ports, 55 DIO
- 2 USB ports, 2 SD Card sockets
- 2 10/100 Ethernets
- Matrix keypad & LCD ports
- VGA video 800x600
- 1.69 s Linux-based bootloader
- Runs Debian, supports Real-Time
- One piece metal enclosure setup provides 4 high-current GPIO, 4 ADC, 2 DAC, 1 CAN for **\$160**

Design your solution with one of our engineers

- Over 20 years in business
- Never discontinued a product
- Engineers on Tech Support
- Open Source Vision
- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products stocked and available for next day shipping

See our website for options, peripherals and x86 SBCs



We use our stuff.

visit our TS-7200 powered website at
www.embeddedARM.com
 (480) 837-5200

Exploiting 64-Bit Linux

How a big Mandelbrot can illustrate the advantage of using 64-bit Linux. STEVE MUNROE

Wide availability of 64-bit systems offers an opportunity to simplify programming. By leveraging large virtual addresses and memory mapped files, we combine programming and file persistence into a single activity. This simplifies programming when persistence and sharing of large complex data structures are required. It also improves the efficiency of data access and sharing, as multiple programs can access data directly (operate in place) from the single real page copy. This eliminates the need to copy the data through multiple layers of buffering. When all programs share data at the same virtual address, there are opportunities for the kernel to manage the memory map to avoid aliases and share MMU resources across applications.

As a proof of concept, I have a small user library I call SPHDE (Shared Persistent Heap/Data Environment). This library manages a portion of the application's address space to provide blocks of persistent and shared storage. The library also provides some locking primitives and some "utility objects" that provide finer-grained storage allocation and indexing.

As a demonstration, I wrote a gigapixel Mandelbrot (GTK2) GUI program. This program computes each point in the Mandelbrot set once and saves the resulting image as a quadtree in SPHDE storage. This quadtree representation can be shared and displayed by multiple instances of the program. Program instances can cooperate to fill in the next level of detail.

Resources and Design

System design and programming models are (initially) driven by the need to manage scarce resources. In the early days of programming, both address spaces and physical memory were small. IO Subsystems evolved to manage the buffering of serial media, such as tape and punch cards. Even the introduction of random access disk and virtual memory did not change the basic paradigm much. The programming paradigms we use and teach today have far outlived the original scarcity that caused them. The separation of programming from persistent data is a prime example. It creates inefficiencies in both programming effort and runtime execution.

Opportunities

With the prevalence of 64-bit microprocessors, the original (addressing/memory) scarcity is gone. We can leverage virtual memory to access and share massive data structures directly using "operate in place" and "implicit persistence" patterns. This simplifies programming, because any data structure is implicitly persistent by where it is allocated. This improves operational efficiency, because it eliminates layers of buffer management and the OS has more opportunities to share a single physical copy. There are additional opportunities for improving address translation efficiency by eliminating address aliasing and exploiting hardware features, such as large pages.

Some Alternatives Tried

There have been attempts to unify programming and persistence before. Persistence frameworks abound, but they simply add layers of buffering

and data conversions to the top of the IO Subsystem stack. Operational efficiency is lost to obtain a modest reduction in programming effort.

Alternatively, Single-Level Stores have been a research topic for years. Although the technology works (simplified programming and data efficiency), the requirement for specialized hardware or unique OS environments has limited acceptance.

This picture changes again with widely available 64-bit microprocessors and standardized POSIX memory management APIs. The addition of a small runtime library allows adventurous programmers to use operate in place and implicit persistence across numerous Oses and 64-bit hardware platforms. This hybrid Single-Level Store approach allows for incremental exploration and adoption of a new programming paradigm.

A High-Level Design

Simple memory-mapped files is not enough. We need a design for how the address space and backing files are managed. The design should balance simplicity and efficiency in the management of the virtual (address space) and physical (real memory and disk space) resources.

Here are some principles I believe are important to include in the design. The runtime should be simple to use and compatible with UNIX programming and POSIX APIs. It should use resources (address space) that are not normally used in existing applications. It should appear to the program like a large shared persistent heap (SPH). Access to data in the SPH should use standard C pointer semantics. Setup to access to data in the SPH should be minimum. The design evolved to the concepts of regions, stores, segment and blocks, all with power of two sizes.

The region is simply the range of virtual addresses that the application wants the runtime to manage on its behalf. The region should be in a range of virtual addresses not normally used by the application. We want to ensure that the region does not interfere with the normal program usage of (local temporary) heap and dynamic libraries. On Linux, this means the area above the TASK_UNMAPPED_BASE and below the main stack. Leaving lots of room below the stack and above TASK_UNMAPPED_BASE still allows for the use of up to half the program's address space for SPH.

Next, we need to choose a segment size. The segment is the unit size for allocating backing files and memory mapping those files into the region. This should be large enough so that we don't do frequent kernel calls and small enough so that we don't waste file space.

A store is a directory that contains one or more files used to back SPH segments. A system can have multiple SPH stores by allocating the backing files in different directories. For now (also to keep things simple), a program can bind to only one region/store at a time.

Finally, a block is the unit of space allocation within an SPH region. It is normally between a page and a segment in size, but that is not a hard limit. The only hard limit is that the block fits with the remaining space at the time. If the block is allocated to a portion of the SPH region that does not currently have a backing file allocated, the SPH runtime will implicitly

Looking for hardware as killer as your next app?



GEMINI - Intel® Core™ 2 Duo



SBC-GX533 - AMD Geode™ GX



ZEUS - PXA270



VIPER - PXA255



VULCAN - IXP425



...look to Arcom
embedded boards

- Intel XScale, Core 2 Duo, AMD and X86 technology
- Fanless solutions
- Display controller for TFT and STN displays
- Serial ports for legacy communications
- USB, SDIO, CompactFlash, PC/104 & PCI Expansion
- Complete operating system and application development environment



888-941-2224
www.arcom.com

 **Arcom**
A MEMBER OF EUROTECH GROUP

Listing 1. A List of Memory Blocks and Their Usage

```

-----address-----  --size--
0, 0x400000000000, 1024KB
Total in use          1024KB
Total free            0KB
0, 0x400000100000, 1024KB
1, 0x400000200000, 2048KB
2, 0x400000400000, 4096KB
3, 0x400000800000, 8192KB
Total Uncommitted 15360KB
0, 0x40001000000, 16384KB
1, 0x40002000000, 32768KB
2, 0x40004000000, 65536KB
3, 0x40008000000, 131072KB
4, 0x40010000000, 262144KB
5, 0x40020000000, 524288KB
6, 0x40040000000, 1048576KB
7, 0x40080000000, 2097152KB
8, 0x40100000000, 4194304KB
9, 0x40200000000, 8388608KB
10, 0x40400000000, 16777216KB
11, 0x40800000000, 33554432KB
12, 0x40100000000, 67108864KB
13, 0x40200000000, 134217728KB
14, 0x40400000000, 268435456KB
15, 0x40800000000, 536870912KB
16, 0x41000000000, 1073741824KB
17, 0x42000000000, 2147483648KB
18, 0x44000000000, 4294967296KB
19, 0x48000000000, 8589934592KB
20, 0x50000000000, 17179869184KB
Total Region free 34359721984KB
0, 0x40000000000, 16384KB
Total Region used 16384KB

```

create the file(s) in the store directory associated with the region.

For overall simplicity, we use power of two sizes for blocks and segments. This allows a power of two buddy system to be used to track all aspects of SPH storage. The buddy system reduces fragmentation and simplifies recombining smaller blocks into larger blocks when they are freed.

We need some low-level utility functions to help manage the buddy lists. First, we need a simple heap manager to sub-allocate blocks for smaller control blocks and lists. The lists need to be ordered and will be searched frequently, so we use a simple binary tree. Algorithms for the heap manager and binary tree can be found in Donald E. Knuth's *The Art of Computer Programming* (see Resources).

Finally, we need a place to store and maintain this information. The anchor block is the first block in the region. The anchor block starts with a block header, which includes signature words (eye-catchers initialized to special values), type info, pointer to the start of the local free list (heap) and a set of pointers for the various binary trees that manage space in the SPH region. The remainder of the anchor block is initialized as free heap space. This internal heap is used to allocate

additional nodes required for the binary trees and any other internal control blocks needed to manage the SPH region.

After an SPH region is initialized, we have mostly an empty region with one segment (backing file in the store directory) and one block (the anchor block) allocated from that segment. The various lists have been initialized to depict this state. For example, the region free and used lists contain entries for the unallocated (no backing files) and allocated (with backing files) portion of the region. An entry for the anchor block is added to the "used" list. The "uncommitted" list contains entries for the current unused portions of the first segment. And, the free list is empty. Now we are ready to allocate more blocks.

The Lock Manager

We choose a hash table-based lock manager using the virtual address as the lock ID. The addresses of data within SPH are unique, and active locks can be found quickly in a hash table.

Storage for the lock tables has to be shared but not persistent, so we allocate IPC shared memory for that. This memory is initialized with a block header and associated heap, which is used to allocate storage for the lock hash table and lock lists. IPC semaphores also are allocated and used to block threads waiting on contended locks.

Utility Objects

So far, we have blocks of shared persistent storage and an address-based lock manager. Blocks are useful for storing large uniform arrays but are awkward to use for complex structures, such as link lists and trees. The SPH runtime includes utility objects that allocate and manage blocks for finer-grained allocations and complex lists and trees.

Utility objects all start with a block header and provision for an internal heap (using the same heap manager as the anchor block). The same signature words are used, but each utility object has a unique type. The type values define a simple type system for runtime checks. The signature words and the fact that blocks are all powers of two sizes simplifies finding the block header for any utility object from any address within a block. This is the trick supporting the new-near allocation scheme discussed later.

The simplest utility object is SPHSimpleStack. The SimpleHeap is simply a block header and internal heap. A CompoundHeap is a heap manager that allocates SimpleHeaps. The block header links multiple CompoundHeap blocks together to form an expandable CompoundHeap. This "heap of heaps" structure, combined with the "new-near" mechanism is useful for maintaining locality of reference for large complex list and tree structures.

The CompoundHeap is a framework (think superclass) for the SPHStringBTree, SPHIndex and SPHContext utility objects. The SimpleHeap is the framework for the BTree nodes internal to SPHStringBTree and SPHIndex. An SPHStringBTree maps a string (name) to an address. An SPHIndex maps an arbitrary binary value to an address. An SPHContext defines a two-way mapping between one or more strings and an address. These utility objects are useful for creating naming structures and content-addressable memories.

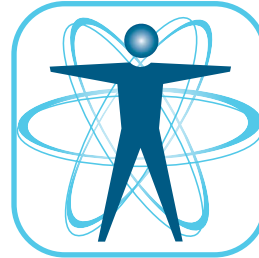
Handling Gigapixel Images

I needed a test for Shared Persistent Heap runtime and thought storing and processing large images would be interesting. In a previous personal project, I implemented a fast Mandelbrot Fractal (see Resources) display based on breaking the image in quadrants and storing the image in a quadtree structure. The Mandelbrot set

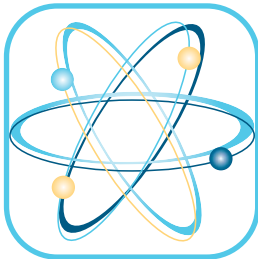
SD BEST PRACTICES

2007 CONFERENCE & EXPO

September 18-21, 2007
Hynes Convention Center
Boston, MA



Early Bird Discount
Register by August 17
SAVE UP TO \$300



the future of software development

Over 175 Classes and tutorials in 8 tracks

- Build & Deploy
- C++
- Design & Architecture
- People, Projects & Teams
- Process & Methods
- Requirements & Analysis
- Testing & Quality
- Web Services / SOA

PLUS Secure Design mini track

Register today at www.sdexpo.com

Plus:

Expo, Case Studies, Birds-of-a-Feathers, Roundtables, Parties and Special Events

Keynotes Include:

- Ivar Jacobson
- Jim McCarthy



CMP
United Business Media

NEW THIS YEAR!

SD Best Practices will be co-located with Embedded Systems Conference and RFID World

All SD Best Practices attendees will also get access to the RFID World and Embedded Systems Expo floor, industry addresses, keynotes, sponsored sessions and selected special events.

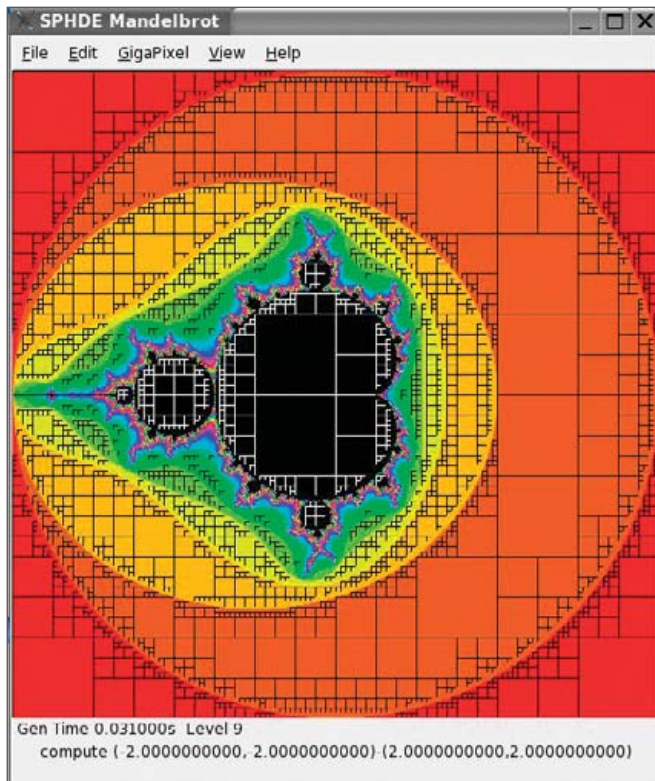


Figure 1. Mandelbrot Set with Quadtrees Outlined

is interesting, because it is “self-similar” and shows detail and any zoom factor.

This program could pan and zoom over color renditions of the Mandelbrot set where most of the image was pre-computed and stored in the quadtree. The algorithm is incremental, so detail is computed and added to the quadtree as needed (for the current display).

At the time, I had no good way to store the resulting quadtree for later use. I did write recursive streaming code to write/read a flattened representation of the quadtree to/from a file. But, as the quadtree image accumulated detail, it slowed down noticeably. Also, I ended up with multiple files with pre-computed details of different areas of the Mandelbrot set, but had no good way to merge them in a single high-resolution image. At this point, the Mandelbrot Project was set aside.

Later, when I was working on the SPHDE Project and was trying to think of a good demo, I remembered the quadtree Mandelbrot Project. The hard part was converting the original Mandelbrot program from Borland OWL to Linux and GTK2. The actual conversion to use SPHDE was much easier.

First, I added SPHJoinRegion and SPHCleanUp calls to the entry and exit of main. Then, I added code to handle first-time setup. This involved allocating a control block to anchor the quadtree and create an expanding SPHCompoundHeap to manage storage of the quadtree. Subsequent use of the program needs to obtain only the address of this control. This pointer can be stored and retrieved from a free slot in the anchor block header.

The next step was to change the quadtree algorithm to use SPHDE

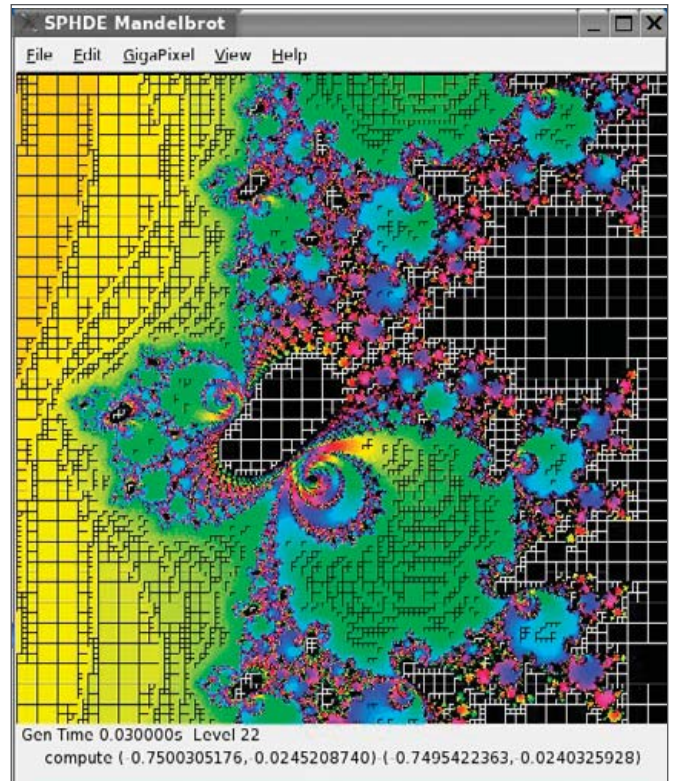


Figure 2. Zoom 8192X with Quadtrees Outlined

storage. This is only slightly complicated by a desire to maintain good locality of reference within the quadtree itself. The SPHCompoundHeap allocates SPHSimpleHeaps from which quadtree nodes are allocated. Allocating adjacent quadtree nodes from the same SPHSimpleHeap ensures physically locality in memory and the backing file. This minimizes the number of pages touched to display the whole Mandelbrot set (the topmost part of the quadtree) or zoom to any part of the set.

The simple call:

```
node = (TQuadTree*) SPHSimpleHeapNearAlloc (near, sizeof (TQuadTree));
```

always will attempt to allocate from the SimpleHeap containing the nearest existing quadtree node. If this allocation fails, we either need to find another SimpleHeap with free space or call SPHCompoundHeapAlloc to allocate another SimpleHeap:

```
if (!node)
{
    temp1 = SASCompoundHeapAlloc (QuadBlockPtr->compoundHeap);
    if (temp1)
    {
        node = (TQuadTree *)
            SASSimpleHeapAlloc (temp1, sizeof (TQuadTree));
        lastAlloc = temp1;
    }
}
```


A simple cache of recently allocated SimpleHeaps and SimpleHeaps where nodes were recently freed served to keep the quadtree relatively dense within the SPHCompoundHeap managed storage. All this logic is contained in 100 lines of code.

Finally, we need to ensure that multiple instances of the program can share and update the quadtree safely. This requires adding SPHLock/SPHUnlock calls around any code with the potential to modify a quadtree node. There are only four such locations in the current algorithm. The utility objects (SimpleHeap, CompondHeap and so on) use appropriate locks internally to ensure consistent behavior in a shared environment.

Conclusion

The result is a program that can compute and store a gigapixel or larger image. Generating a gigapixel (32768x32768) Mandelbrot image takes about five minutes (Apple G5 dual-2.3GHz PowerPC). Once the gigapixel quadtree is generated, exiting and restarting is nearly instantaneous (24 milliseconds to display a 512x512 pixel image). Panning and zooming around the pre-computed quadtree is also fast (12–20 milliseconds). Zooming to a depth beyond the current pre-computed set will slow down (200–500 milliseconds) due to the heavy computation required. Once an area is computed and stored, subsequent displays are fast (12–20 milliseconds) again.

The SPHDE runtime and Mandelbrot demo program run on both 32-bit and 64-bit Linux, including the PowerPC, i386 and x86_64 platforms. 32-bit systems can support a 1–2GB region, which is large enough to store a gigapixel Mandelbrot quadtree (~313MB). Larger (terapixel) images will require the large regions allowed by 64-bit systems. With recent Linux kernels, PowerPC64 can support regions in the 8TB range. The X86_64 platform supports a 128TB address space and a 64TB region. This is more than enough for some interesting projects. ■

Steve Munroe is currently an architect covering toolchain (GCC, GLIBC, binutils, GDB and related performance collection tools) issues in IBM's Linux Technology Center. Steve is an active contributor to open-source software, porting the GNU C library (GLIBC) to 64-bit PowerPC. Steve is an expert in single-level storage, large address space architectures and Java, with an emphasis on object persistence and business applications. He coauthored (with Steven Halter) the book *Enterprise Java Performance*. He also was a performance architect for IBM's San Francisco Project and authored a Java workload from which he helped SPEC (Standard Performance Evaluation Corporation) create SPECjbb2000, an industry-standard workload for Java server performance.

Resources

Enterprise Java Performance, Chapter 14, Steve L. Halter and Steven J. Munroe, Prentice Hall 2001, ISBN 0-13-017296-0.

The Art of Computer Programming, Volume 1, Fundamental Algorithms, 3rd ed., Donald E. Knuth, Addison Wesley 1997, ISBN 0-201-89683-4.

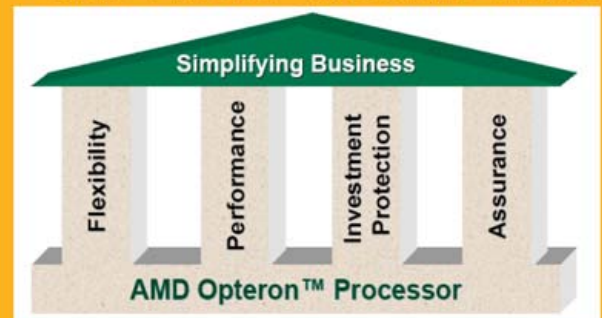
SASOS: www.ira.uka.de/csbib/Os/sasos.html

Mandelbrot Set: en.wikipedia.org/wiki/Mandelbrot_Set

Quadtree: en.wikipedia.org/wiki/Quadtree



Linux - FreeBSD - x86 Solaris - MS etc.



The AMD Opteron™ processor with Direct Connect Architecture scales from 1P/2-core up to 8P/16-core across a single industry-standard platform without external logic, allowing for maximum versatility and lowering overall system cost.

Quad CPU



Up to 8 Cores - in 1U

Ideal for high density clustering in standard 1U form factor. Up to 8 Cores for high CPU needs. Comes pre-configured with OS of your choice.

Features:

- 1U rack-optimized chassis (1.75in.)
- Up to 4 Dual Core AMD Opteron™ processors
- Up to 8 Cores Per 1U rackspace
- Up to 64GB DDR2 667 & 533 Mhz SDRAM
- Dual-port Gigabit Ethernet Per Node
- 3 SATA or SCSI Removable HDD Per Node
- 1 (x8) PCI-Express



Servers :: Storage :: Appliances

Genstor Systems, Inc.

780 Montague Expy. # 604
San Jose, CA 95131

www.genstor.com

Email: sales@genstor.com

Ph: 1-877-25 SERVER or 1-408-383-0120



AMD, AMD Opteron and the AMD logo are trademarks or registered trademarks of Advanced Micro devices, Inc. Product pictures only for reference.

Take a Peek at Some of the Freshest Projects Around

A view to a killer app or four, including LMMS, inotail, Karmen and GAMGI. JOHN KNIGHT

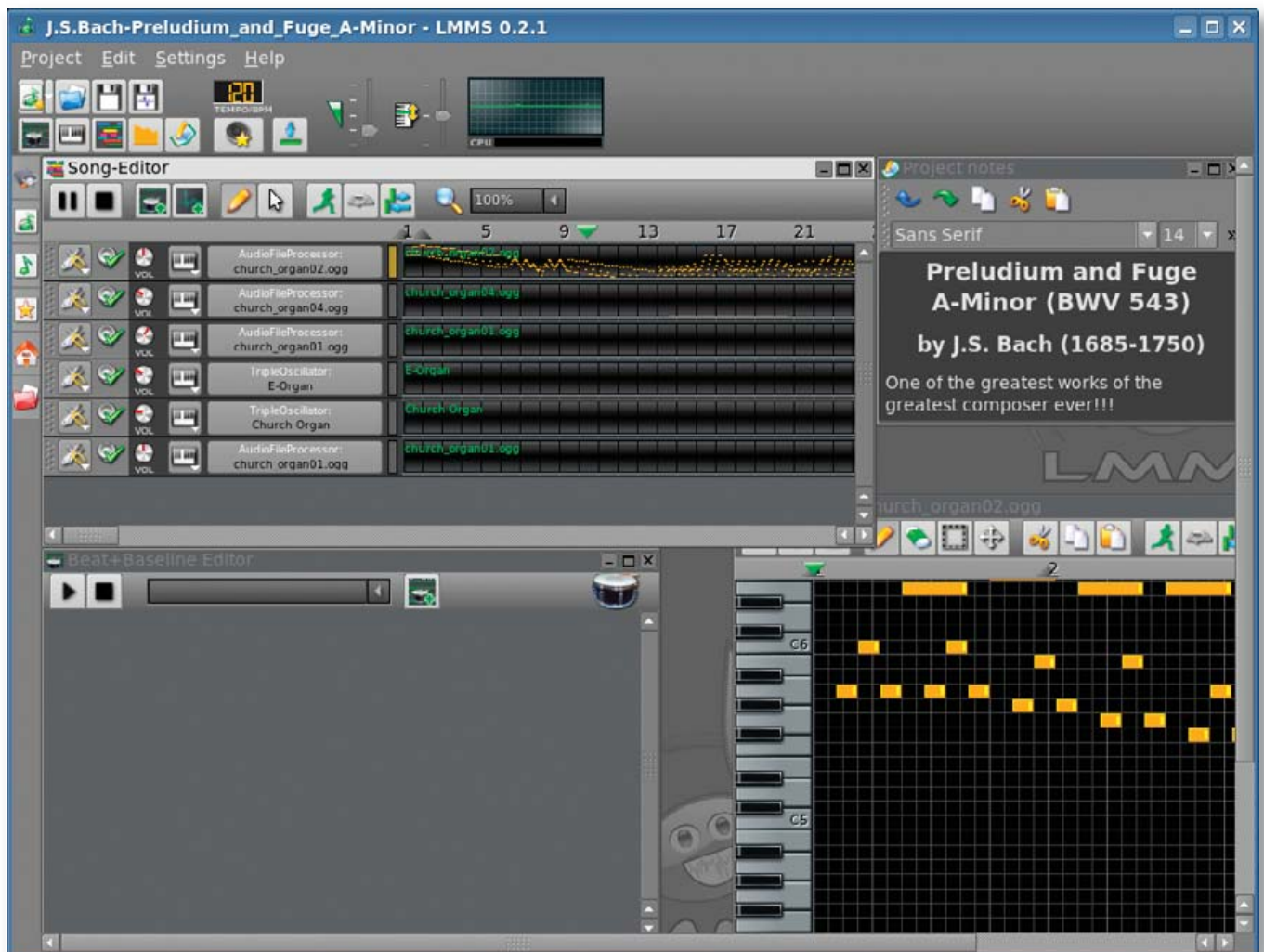
LMMS—Linux MultiMedia Studio

Initial screenshots of LMMS will give you a hint of FruityLoops and the like, but in reality, LMMS takes a completely different tack. LMMS takes on a micro-managed approach, and it probably will appeal to control freaks who want to generate their own samples or instruments from scratch. With the ability to layer instruments and tracks, you can create whole songs from the ground up.

Unlike most compositional software under Linux, LMMS takes an alternative approach to midi for many circumstances, by synthesizing through wave instead. This sidesteps the often annoying process of

making midi work under Linux, although midi still can be used and is very much a part of LMMS. By tweaking all the knobs and available options, you can generate entirely original sounds and use them in any way you see fit.

I managed to generate some brilliant effects and samples for my own musical projects. This likely will appeal most to dance-music DJs and also to fans of *Nine Inch Nails* or the more recent *Massive Attack*-style music for those who want unique samples, or it will appeal to fans of retro 1980s synthesized sound for those who want to make whole orchestrations.



LMMS—the Music Suite for the Ultimate Control Freak

Installation is a breeze, as LMMS resides in many distributions' repositories. Under Debian-based distributions, all I had to do was a simple:

```
# apt-get install lmms
```

For those who can't find a binary for their system, however, a simple:

```
$ ./configure
$ make
(as root or sudo)
# make install
```

with the source is all that is required. Thankfully, LMMS doesn't have too many strange dependencies, so it should compile right off the bat for most systems.

Once you dive into the world of LMMS, a steep learning curve presents itself. Some tutorials and demos are included, but these will leave most users in the dark on many issues (but they are worth examining as they do uncover the uses of many GUI items). The demos are indeed worth a look and show off the capabilities of the program. The demos include Bach's "Preludium and Fuge in A Minor", which is impressive when you watch the keys played in real time under each instrument's settings, and other cool demos are included that range from general dance music to the occasional more atmospheric piece.

The best tip for exploring LMMS for the first time is to click on the yellow star on the left whose tab is titled My presets. From here, you can play around with some already-tweaked instrument settings and various GUI buttons and sound settings until you come up with something that sounds interesting and the interface becomes more familiar. Also, be sure to look at the green musical note above the yellow star, titled My samples. This contains the base instruments to begin with and offers a large variety, including (but not limited to) some impressive drum samples, Latin guitars, as well as some amazing string, choir and atmospheric effects.

Despite all of the features and demos, however, it's worth visiting the LMMS Wiki. Documentation is still lacking for this project, but hopefully this improves in the future, particularly in relation to chord compositions and note editing. Documentation issues aside, the program is quite stable at the moment and can at least be used as a good beat sequencer. If you can figure out the interface for instrument composition, whole tracks also will be at your fingertips.

Some other impressive features are JACK and LADSPA support, but best of all is the ability to mix an instrument in surround as opposed to just stereo. When I asked lead developer Tobias Doerffel about the development status, he mentioned that LMMS is still rather unstable on 64-bit platforms, but after this is fixed, it will be in a quite usable state. He also wants to include a mixer that mixes all of the tracks as well as improve the available JACK support.

Currently, this is fairly mature beta software that looks extremely promising, and it is lacking only decent documentation—definitely worth a look for any home musician.

For more information on LMMS, see:

■ Home Page: lmms.sourceforge.net

■ Download Page: lmms.sourceforge.net/download.php

■ Wiki: wiki.mindrules.net

inotail—Inotify Enhanced Tail

For all you system administrators out there keeping track of important log files with tail, this is definitely worth a look. When using the follow mode (`tail -f filename`), tail re-reads a file once a second by default. inotail takes a different approach by making use of a newer kernel feature, the inotify API. Instead of a clumsy cyclical update based purely on time, inotail listens to special events sent by the kernel using the new API.

After testing inotail, I was happy to see its results. I simply took a text file and read it with the command `inotail -f test.txt`. With the text file, I added lines one at a time and saved the document each time as I added a line. As soon as I pressed Save, the inotail screen output instantly updated without a hitch. Okay, so it's not exactly going to impress your mates, but system administrators probably are going to find an instant use for this far more elegant approach to keeping an eye on file updates.

I'm afraid that you still will have to compile this one from source, but it isn't hard. According to developer Tobiad Klausner's Web site, inotail should be in the Debian repository soon, but it isn't there at the time of this writing. To compile and install inotail, simply extract the tarball

COMPACT EMBEDDED SERVER



- x86 200MHz CPU
- 128MB SDRAM On Board
- 512MB CompactFlash™
- 10/100 Base-T Ethernet
- Reliable (No CPU Fan or Disk Drive)
- Two RS-232 & Two USB 1.1 Ports
- Optional Wireless LAN & Hard Drive
- Optional On Board Audio
- Dimensions: 4.5 x 4.5 x 1.375" (115 x 115 x 35mm)

Compact SIB (Server-In-a-Box) Starting at \$170.00 Quantity 1.

- EMAC Linux 2.6 Kernel
- Menu Drive Configuration Utility
- Eclipse Development Environment
- HTTP and FTP Servers
- PPP Dial In/Out Server & Client
- Telnet Server

Since 1985 OVER 22 YEARS OF SINGLE BOARD SOLUTIONS

EMAC, inc.

EQUIPMENT MONITOR AND CONTROL

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

Instead of a clumsy cyclical update based purely on time, inotail listens to special events sent by the kernel using the new API.

(available on the main page of the Web site), and go to the new directory. Enter the commands:

```
$ make
(as root or sudo)
# make install
```

As with most source compilations, this places the executable in `/usr/local/bin` by default. If you would rather place it in `/usr/bin`, enter the command:

```
# make prefix=/usr install
```

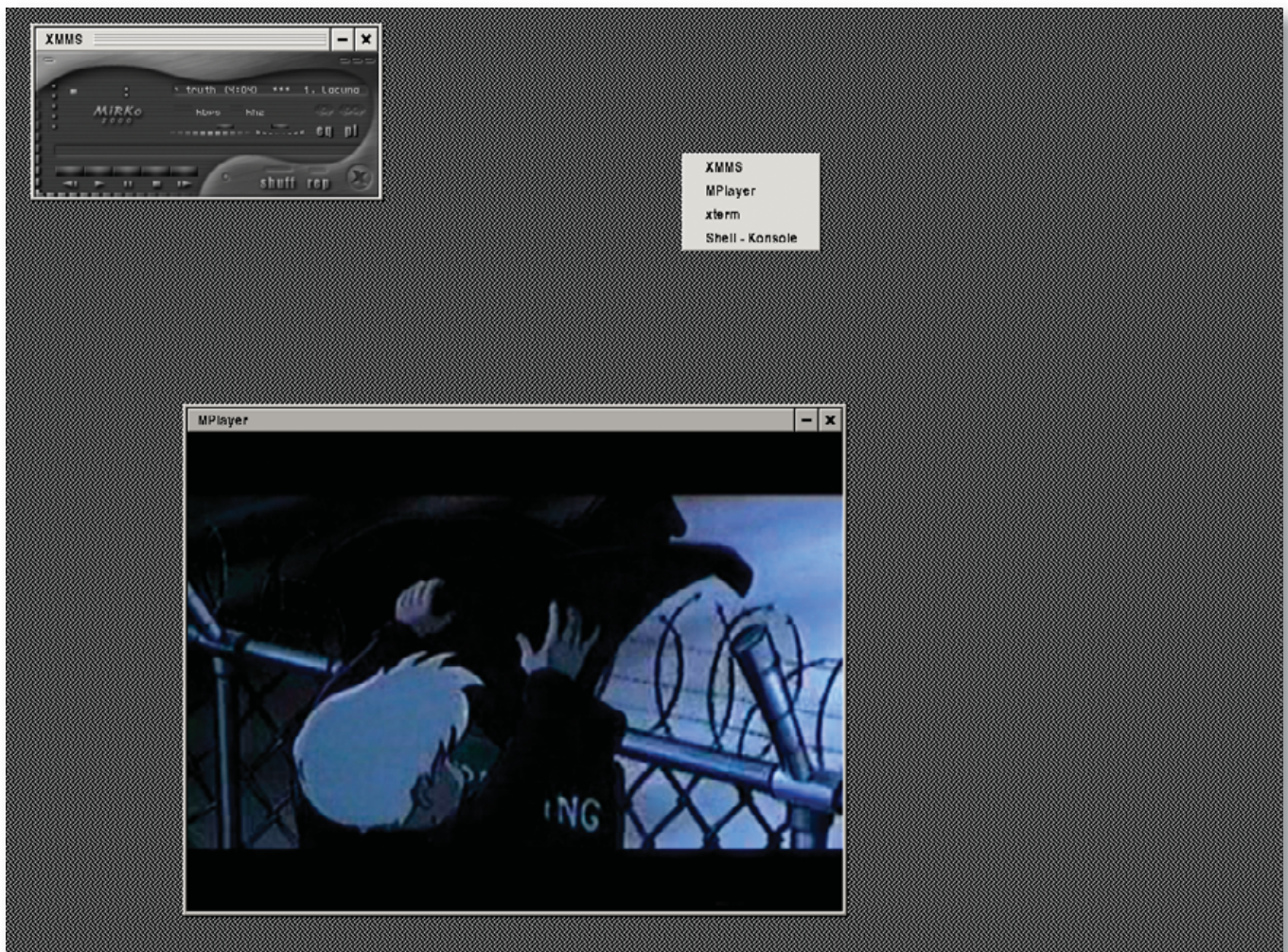
The only real factor hampering this project for now is time. inotail requires at least a 2.6.13 kernel, which still is fairly young for many system administrators who tend to use somewhat older and more mature distributions than the rest of us. After a year or so has passed, when inotail has been accepted into the Debian archive and administrators have upgraded their distributions, inotail should be finding its way into many an admin's toolbox.

Home Page: distanz.ch/inotail

Karmen

For those of you who like minimalist window managers, Karmen may be your future choice. Designed to "just work", it has no dependencies except Xlib and no configuration file to fiddle around with. According to the README file, these are its main goals for full release:

- Intuitive, efficient window management.
- Provide a high-quality look and feel.
- Standards compliance (ICCCM and EWMH/NetWM).



Karmen—for a Clean and Minimalist Environment

- Work well as a standalone window manager.
- Work well with other desktop utilities and environments.
- Focus on window management and let other tools do the rest.

Head to the main Web site to grab the latest tarball. Indeed, installing it was a cinch. Doing a simple:

```
$ ./configure
$ make
(as root or sudo)
# make install
```

was all that was required, and as it says on the tin, there are no weird little dependencies to get in the way. It also happens to be quick and sleek, and each time I turned my head back around to look at the monitor, `configure`, `make` or `make install` was already done!

Once you have the Karmen desktop running, moving windows around is nice and familiar in a KDE/GNOME/Windows way, with clicking to focus and window resizing working in the intuitive way users have come to expect. Rather than the all-too-common annoyance of having to click on a window's titlebar to change focus that plagues

many lightweight window managers, Karmen lets you switch window focus by clicking within a window's body. Maximizing is a different affair though; whereas two obvious minus and close buttons sit at the top right of a window, maximizing requires that you double-click the titlebar—not immediately obvious. For a list of windows (whether active or minimized), right-click on the desktop, and a new menu appears, allowing you to bring to focus any windows currently running.

Another interesting note: Karmen seems to be quite a keyboard-driven window manager. For instance, to tell a window to stay on top of others, click the titlebar with the Shift key held down. Shift-clicking again disables the stay-on-top property of the window. A welcome addition to minimalist window managers, pressing the familiar old Alt-Tab cycles between windows. In fact, most basic GUI functions can be performed via the keyboard, generally by using a combination of Alt and another key.

However, Karmen is quite young in its development, with some strong limitations in its current form. Still lacking is any kind of menu for major functions, such as logging out or choosing a simple xterm. Indeed, Karmen still requires that you kill its process manually to exit—obviously something that will be changed in the future, but a definite indicator of a project that's still in its infancy. Lacking too are startup scripts to start the window manager cleanly; you either have to make your own script or edit `.xinitrc`. For those who can't be bothered with



Don't complicate a simple task

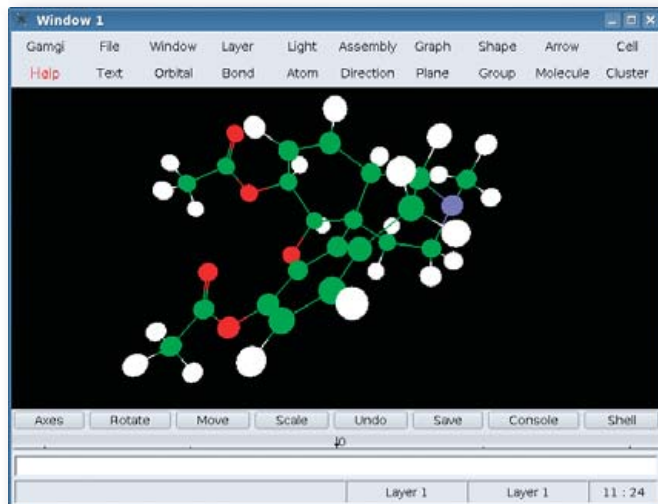
Keep basic tasks just that with handheld, stationary and vehicle-mounted wireless data collection from AML. While others are busy reinventing the wheel, we're keeping things simple, from our products to our personal service. Visit us at www.amld.com or call **1.800.648.4452** for a real live person.



M7100 Wireless Family







GAMGI—for the mad scientist. Can you believe that thing is a heroin molecule?

this, you also could start Karmen from the command line, but you have to add `xterm` specifically after the command. Thankfully, these topics are touched upon in a nice man page—a professional touch at this early stage. Also, for now, Karmen is a strictly one-desktop affair, but hopefully this also will change.

All limitations aside, however, developer Johan Veenhuizen's approach seems to be one of working on individual sections cleanly and then moving on. The coding is indeed very clean and stable—although in its infancy—and looks like it will be a snazzy little desktop once finished. I reckon Karmen definitely will find a home with many niche users once it matures.

Karmen: karmen.sourceforge.net

GAMGI—General Atomistic Modeling Interface

For this last project, I thought I'd throw in something for the mad scientists out there. According to the home page, GAMGI's goals are to provide the community with a free software package to construct, view and analyze atomic structures, and to make it as powerful and as simple to use as possible. GAMGI's developers also are probably the most hardcore free-software advocates I've seen since Stallman himself. Scattered throughout the installation instructions and user manuals are loads of exhortations to harness the power of source and compile it yourself, as well as many other passionate free-software-related endeavors.

Here are GAMGI's stated aims from these obviously passionate developers:

GAMGI aims to be useful for: 1) the scientific community working in atomistic modeling, which needs a graphic interface to build and analyze atomic structures; 2) the scientific community at large, which needs a graphic interface to study atomic structures and to prepare images for presentations; 3) teaching the atomic structure of matter in schools and universities, even inviting students to run GAMGI at home; 4) science promotion, in exhibitions and science museums.

Installation can be a little tricky for the newbie, but anyone comfortable with things such as `make`'s role or symlinking should be okay.

GAMGI is designed to be as standardized as possible in its requirements, and it should work with any standard X11 installation, though it does have one or two esoteric library requirements. Glib and GTK should be fine with most systems, as well as FreeType, but you may have to seek out and install Expat and Gtkglarea.

For those seeking a simple binary, an RPM is usually provided at GAMGI's download page. The RPM is designed to be LSB-compliant and should convert easily with a utility such as `alien` for Debian-based systems. At the time of this writing, however, the link was taken down, so please check the site again at a later date or bug the developers. For those seeking to compile from source (keeping the developers happy), a few more steps than your run-of-the-mill tarball need to be taken. First, it's easiest if GAMGI is extracted and compiled somewhere public, such as `/usr/local/src`, as you probably will want to symlink to the compiled binary later. Grab the tarball from the given link and extract it.

Once extracted, simply open a console and make your way to the `src` directory under the new folder. Now run the command `make`, and if all goes well, GAMGI should compile successfully. If `make` returns errors, check that you have all of the needed libraries installed. If they are, your shared libraries may be installed to another path than what GAMGI's `make` script is expecting. Edit the `make_local` file in the `src` directory and continue. Once compilation has completed, you can run the `gamgi` binary from the `src` directory, or symlink to it for easier access with the command:

```
# ln -s /usr/local/src/gamgi-(version)/src/gamgi
  ➔ /usr/bin/gamgi
```

Once all that nasty stuff is out of the way, the weird science can begin. Some interesting templates and tutorials have been included in GAMGI's `dat` directory. From the menu, choose `File`→`Import` and choose an atomic structure at random. Ever wondered what adrenaline looks like? Try `molecule/drugs/adrenaline.xml`. Presto! An adrenaline molecule appears in the main black window. To play around with the view, press and hold the left, middle or right mouse buttons and move the mouse around. The left button controls rotation, the middle buttons controls the X and Y axes, and the right button controls the zoom function. You have to admit, it is pretty cool to play with!

To explore further, there are a bunch of other menus related to all things atomic, but I haven't the foggiest idea how to use any of them (or what they even do). I think this is best left to the atomic scientists. Check the `doc` directory for further instructions, and have a read through the well-constructed HTML manual pages, or head over to the GAMGI home page and converse with fellow white-coated boffins. Either way, I'm sure I'll sleep easy tonight now that I know what a propane molecule looks like and have confirmed that I am, indeed, a geek!

For more information, see:

- Home page: www.gamgi.org
- Tarball: www.gamgi.org/src/gamgi-all-0.12.5.tar.gz
- Wiki: www.gamgi.org/wiki ■

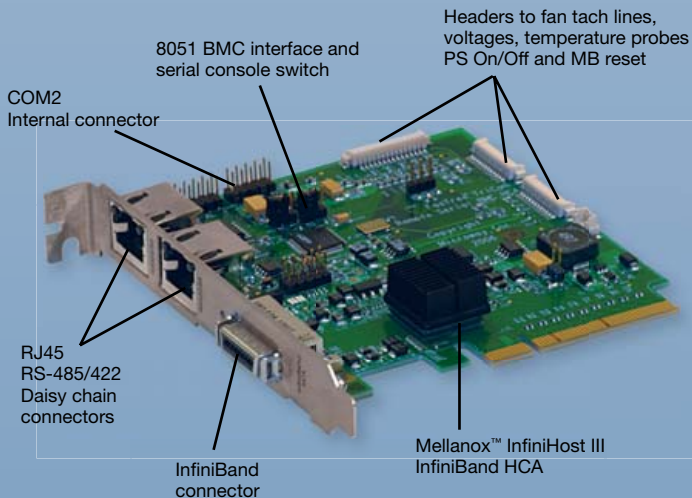
John Knight is a 23-year-old drumming, climbing and music-obsessed maniac who usually can be found squeezing Audacity to within an inch of its life.

Affordable InfiniBand Solutions

4 Great Reasons to Call Microway NOW!

1 TriCom™

- DDR/SDR InfiniBand HCA
- "Switchless" serial console
- NodeWatch web enabled remote monitor and control



2 FasTree™

- DDR InfiniBand switches
- Low latency, modular design
- 24, 36 and 48 port building blocks



3 ServaStor™

- Extensible IB based storage building blocks
- Redundant and scalable
- Parallel file systems
- Open source software
- On-line capacity expansion
- RAID 0,1,1E, 3, 5, 6, 10, 50



4 InfiniScope™

- Monitors ports on HCA's and switches
- Provides real time BW diagnostics
- Finds switch and cable faults
- Lane 15 interface
- Logs all IB errors



Upgrade your current cluster, or let us design your next one using Microway InfiniBand Solutions.

To speak to an HPC expert
call **508 746-7341** and ask
for technical sales or email
sales@microway.com
www.microway.com

 **Microway**
Technology you can count onsm

Building Firefox Extensions

How to create your own Firefox extension. JUSTIN HUFF

Like many *Linux Journal* readers, Firefox is my browser of choice. One of its core strengths is the number of available extensions. Early extensions were focused around merely changing the look of the browser; however, in the past few years, extensions have been used to provide a very rich user experience while straddling the line between desktop and Web applications.

In this article, I explain how easy it is to extend Firefox by building an extension that integrates with a photo editing API provided by Picnik.

Setting Up the Environment

The first thing a new extension developer should do is set up a development profile. Although you can do extension development using your normal Firefox profile, it often is easier to create a new profile dedicated to development. First, start Firefox's profile manager:

```
$ firefox -ProfileManager
```

Next, click the Create Profile button. Once the wizard is loaded, click Next to get started. At this point, you should see a window similar to the one shown in Figure 1. Enter a name for your new profile (I used dev). Make sure you write down the path to the folder where your profile will be stored before clicking Finish. You'll be using that path later.



Figure 1. Creating the Profile Named dev

Now that you have a dedicated profile, you should install some extensions that make development easier. The first one you should install is the Extension Developer. This is a compilation of several handy extensions—all designed to make

Your First Extension

developers' lives easier. See Resources for several other handy extensions. I highly recommend that you install all of them.

At this point, you're ready to start your first extension. Nearly all extensions start with the same basic boilerplate code, so the same person who made the Extension Developer put together the Firefox Extension Wizard to automate this part of the process. You can find its URL in the Resources for this article.

Most of the required fields should make sense. The main one of note is the Extension ID. This is used to identify the extension uniquely for updates and other purposes. In the past, standard practice was to

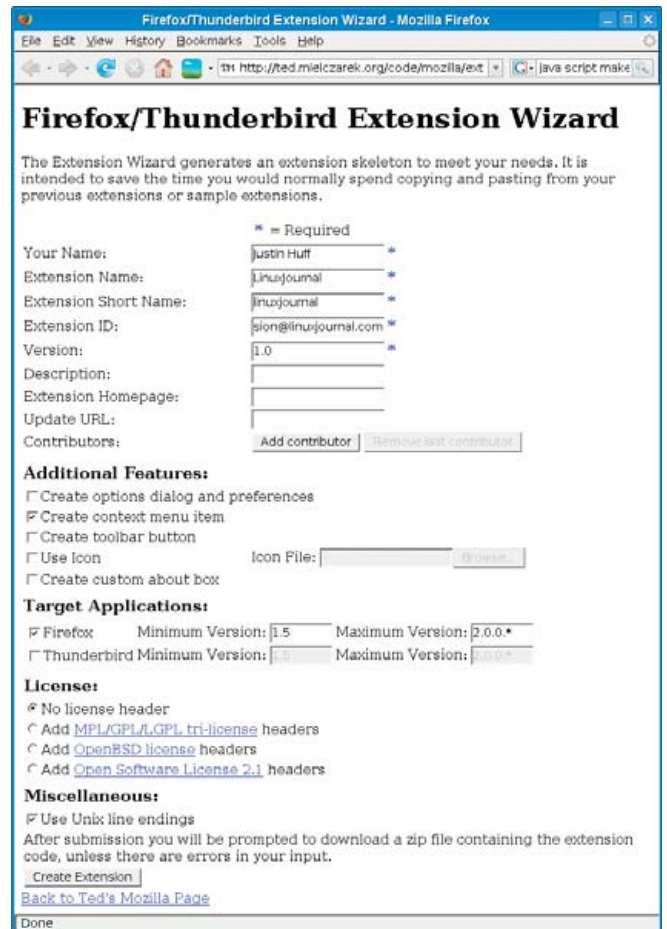


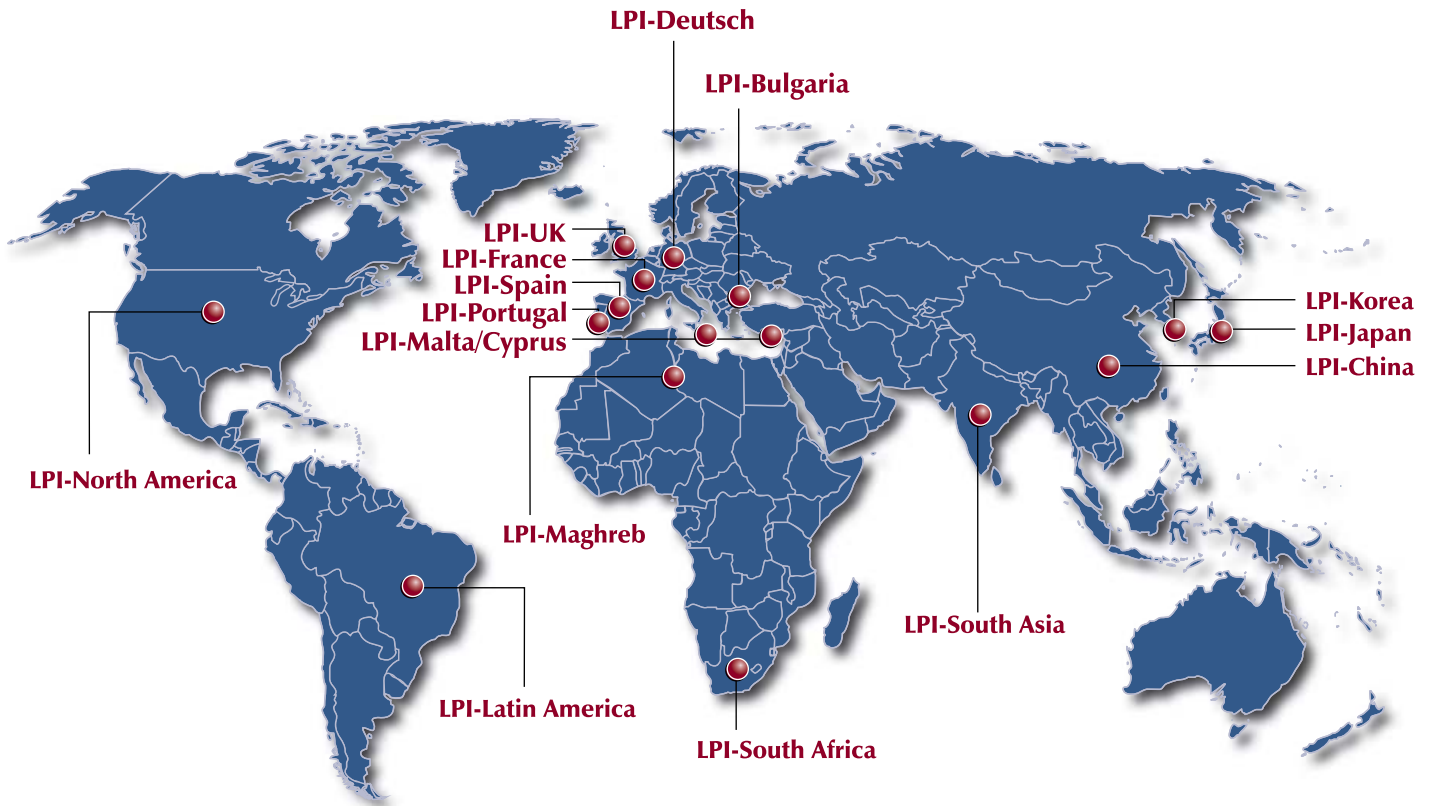
Figure 2. Navigating the Firefox/Thunderbird Extension Wizard

use a GUID (Globally Unique Identifier). Most developers recently have switched to a format that bears resemblance to an e-mail address. For this example, I used extension@linuxjournal.com. I also selected the option to create a context (right-click) menu. Figure 2 shows how I filled in the rest of the fields.

Once you are satisfied with your choices, click the Create Extension button. After a few seconds, your browser should prompt you to download a zip file. Go ahead and extract it:

```
$ unzip linuxjournal.zip
Archive:  linuxjournal.zip
  inflating: linuxjournal/install.rdf
  inflating: linuxjournal/chrome.manifest
  inflating: linuxjournal/readme.txt
  inflating: linuxjournal/content/firefoxOverlay.xul
```

Growing a World of Linux Professionals



We at the Linux Professional Institute believe the best way to spread the adoption of Linux and Open Source software is to grow a world wide supply of talented, qualified and accredited IT professionals.

We realize the importance of providing a global standard of measurement. To assist in this effort, we are launching a Regional Enablement Initiative to ensure we understand, nurture and support the needs of the enterprise, governments, educational institutions and individual contributors around the globe.

We can only achieve this through a network of local "on the ground" partner organizations. Partners who know the sector and understand the needs of the IT work force. Through this active policy of Regional Enablement we are seeking local partners and assisting them in their efforts to promote Linux and Open Source professionalism.

We encourage you to contact our new regional partners listed above.

Together we are growing a world of Linux Professionals.



Stable. Innovative. Growing.


```

inflating: linuxjournal/content/overlay.js
inflating: linuxjournal/skin/overlay.css
inflating: linuxjournal/locale/en-US/linuxjournal.dtd
inflating: linuxjournal/locale/en-US/linuxjournal.properties
inflating: linuxjournal/config_build.sh
inflating: linuxjournal/build.sh

```

Before going into the purpose of all those files, you should install it to see what the auto-generated extension actually looks like. Firefox can use extensions installed in two ways. The normal installation method involves opening the extension's .xpi file in Firefox. This is the way most extensions are distributed and installed. The other method is to create a pointer file that tells Firefox where to find your extension's files. With this method, you don't have to re-install the extension every time you want to test a change; all you have to do is create the pointer file:

```

$ cd linuxjournal
$ pwd > ~jjhuff/.mozilla/firefox/lhn85ppm.dev/extensions/
➔extension\@linuxjournal.com

```

Of course, you'll want to replace ~jjhuff/.mozilla/firefox/lhn85ppm.dev with your Firefox development profile directory.

Now, go ahead and start up Firefox using your development profile:

```
$ firefox -P dev
```

First, check to see that the extension is installed. Select Tools→Add-ons, and verify that LinuxJournal 1.0 is listed. You should see a window like the one shown in Figure 3. While you have the Tools menu open, you probably noticed the new (and red) menu item (Figure 4). Go ahead and right-click in the browser window. You should see a menu similar to the one shown in Figure 5. If everything looks right, your extension is installed properly.

Innards

Before modifying the generated code, you should understand how all the pieces interact. The main file for the extension is install.rdf. It specifies the extension's name, ID and version. The install.rdf file also contains a list of all the compatible applications and their versions. In this example, we specify a single application with an ID of {ec8030f7-c20a-464f-9b0e-13a3a9e97384}, which is the ID for Firefox. We also specify that we're compatible with Firefox versions 1.5 through 2.0.

The second file of interest is chrome.manifest, which tells Firefox what to expect inside the extension. The manifest also includes a list of overlays. (I explain overlays later in this article.)

Most extensions are organized into several directories. The content directory typically contains the bulk of your extensions UI and logic. The skin directory is where CSS and any graphics live. Finally, locale is for locale-specific files, such as translations. (I discuss localization later in this article.)

Chrome

The user interface for Firefox (as well as some other Mozilla projects) is implemented in a file format known as XML User Interface Language (XUL) combined with JavaScript. Collectively, this is known as Chrome. If you installed the Chrome List extension, you

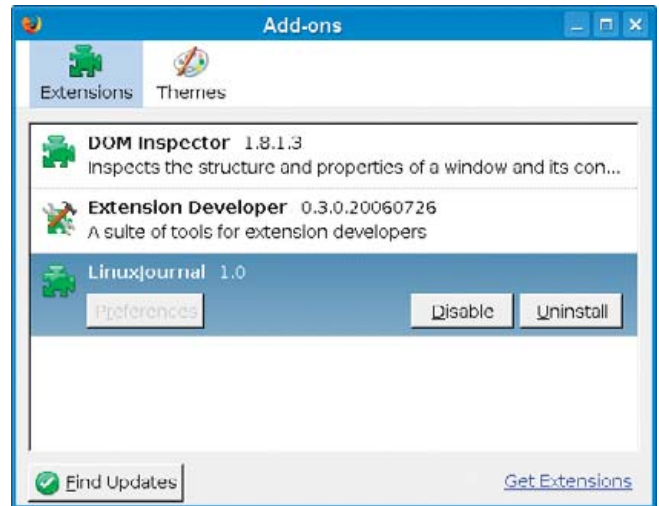


Figure 3. The LinuxJournal extension is installed and ready to go.

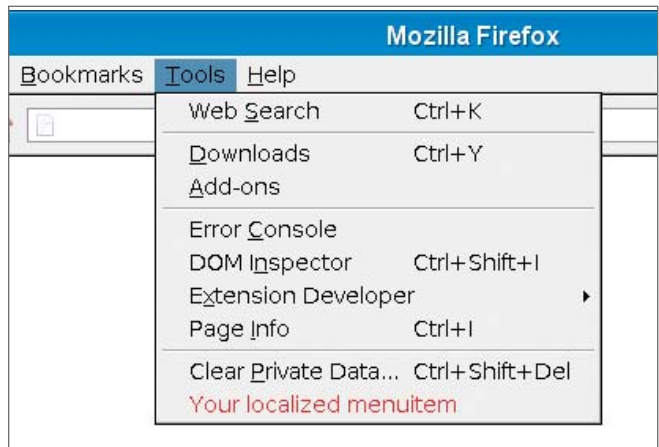


Figure 4. The extension can create a custom menu item under Tools.

easily can view the files that make up your browser and its extensions. For example, the file chrome://browser/content/browser.xul, contains the UI for the main Firefox window.

Additions and modifications to the user interface are created by overlaying additional XUL elements on to the existing Chrome. An extension's overlays are specified in chrome.manifest, with a line similar to the following:

```

overlay
chrome://browser/content/browser.xul
➔chrome://linuxjournal/content/firefoxOverlay.xul

```

This line specifies that firefoxOverlay.xul should be overlaid on top of browser.xul. If you add additional overlay files, or if you want to

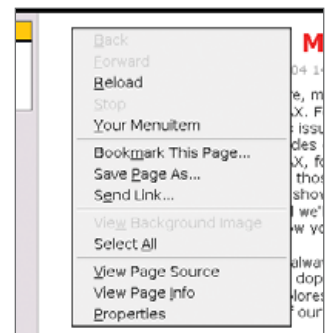


Figure 5. You also can create a custom right-click menu item.

LINUX JOURNAL



1994-2006
ARCHIVE

ISSUES 1-152 of *Linux Journal*

www.LinuxJournal.com/ArchiveCD

The 1994–2006 Archive CD,
back issues, and more!

modify other parts of the application, you need to add more lines to `chrome.manifest`.

Let's take a look at how the extension adds a new item to the context menu. First, open `chrome://browser/content/browser.xul` in the Chrome Browser, and search for `contentAreaContextMenu`. The second hit should look similar to this:

```
<popup id="contentAreaContextMenu" ... >
...
<menuitem id="context-stop"
  label="&stopCmd.label;"
  accesskey="&stopCmd.accesskey;"
  command="Browser:Stop"/>

<menuseparator id="context-sep-stop"/>
...
</popup>
```

Now, open up `firefoxOverlay.xul` from your extension. You should see a block that looks like this:

```
<popup id="contentAreaContextMenu">
  <menuitem id="context-linuxjournal"
    label="&linuxjournalContext.label;"
    accesskey="&linuxjournalContext.accesskey;"
    insertafter="context-stop"
    oncommand="linuxjournal.onMenuItemCommand(event)"/>
</popup>
```

When the overlay is loaded, the browser searches its existing Chrome for an element with the ID of `contentAreaContextMenu` and merges in the XUL from the overlay. It will end up with something like this:

```
<popup id="contentAreaContextMenu" ... >
...
<menuitem id="context-stop"
  label="&stopCmd.label;"
  accesskey="&stopCmd.accesskey;"
  command="Browser:Stop"/>

<menuitem id="context-linuxjournal"
  label="&linuxjournalContext.label;"
  accesskey="&linuxjournalContext.accesskey;"
  insertafter="context-stop"
  oncommand="linuxjournal.onMenuItemCommand(event)"/>

<menuseparator id="context-sep-stop"/>
...
</popup>
```

When this menu is rendered, our menu item appears along with the normal context menu items. In addition, we specified the `insertafter` attribute to tell the browser that we want our menu item to appear after the `context-stop` menu item.

Localization

The Chrome system builds on existing technologies to support localizing

Listing 1. `picnik.xul`

```
<?xml version="1.0"?>

<!DOCTYPE picnik SYSTEM "chrome://picnik/locale/picnik.dtd">

<overlay id="picnik-overlay"
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
  xmlns:html="http://www.w3.org/1999/xhtml">

  <script type="application/x-javascript"
    src="chrome://picnik/content/common.js"/>
  <script type="application/x-javascript"
    src="chrome://picnik/content/contextmenu.js"/>

  <popup id="contentAreaContextMenu">
    <menuitem class="menuitem-icomic"
      id="picnik-ctx-edit"
      insertafter='context-viewimage'
      label="&picnik.edit_picture;"
      oncommand="picnikContextMenu.editImage();"
      image="chrome://picnik/content/picnik_16x16.png"/>
  </popup>
</overlay>
```

the UI easily. One major part of this is the ability to store strings separately from the UI itself in a DTD (Document Type Definition) file. In our code, `firefoxOverlay.xul` references that DTD with the line:

```
<!DOCTYPE overlay SYSTEM "chrome://linuxjournal/locale/linuxjournal.dtd">
```

Chrome URLs for locale are special, because the browser automatically expands them to reference the proper location in the extension. For example, Firefox automatically expands `chrome://linuxjournal/locale/linuxjournal.dtd` to `chrome://linuxjournal/locale/en-US/linuxjournal.dtd` for US English speakers.

The DTD is used to define new XML entities, which can be thought of as macros. Our DTD contains:

```
<!ENTITY linuxjournal.label "Your localized menuitem">
<!ENTITY linuxjournalContext.label "Your Menuitem">
<!ENTITY linuxjournalContext.accesskey "Y">
```

These are referenced in the XUL by prefixing them with an `&`, as in:

```
<menuitem id="context-linuxjournal"
  label="&linuxjournalContext.label;"
  accesskey="&linuxjournalContext.accesskey;"
```

This separation of strings from the UI can be awkward at first, but it has other advantages beyond localization. For example, if you make a spelling error in something that appears in multiple places, you have to fix it only once.

Code

The actual code behind extensions is written in JavaScript, which makes writing them within reach of many seasoned Web developers. JavaScript

Listing 2. contextmenu.js

```
var picnicContextMenu = {
  onLoad:function()
  {
    // Attach the showContextMenu function
    // to the context menu
    var e = document.getElementById("contentAreaContextMenu")
    if( e )
      e.addEventListener("popupshowing", function(ev){
picnicContextMenu.showContextMenu(ev); }, false);
  },

  // Called right before the context menu
  // popup is shown
  showContextMenu: function(event)
  {
    if( gContextMenu )
    {
      var edit_picture = document.getElementById("picnik-ctx-edit");
      if( edit_picture )
        edit_picture.hidden = !(gContextMenu.onImage ||
gContextMenu.hasBGImage);

      if( gContextMenu.onImage )
        this.imageURL = gContextMenu.imageURL;
      else if( gContextMenu.hasBGImage )
        this.imageURL = gContextMenu.bgImageURL;
      else
        this.imageURL = '';
    }
  },

  // Called if the user clicks the 'edit'
  // menu item
  editImage: function()
  {
    var url = picnicCommon.baseUrl + "?import=" + escape(this.imageURL);
    gBrowser.selectedTab = gBrowser.addTab(url);
  },

};
window.addEventListener("load", picnicContextMenu.onLoad, false);
```

also makes it easy for extensions to be cross-platform with very little work. In order for the code actually to be loaded, it must be referenced in an overlay XUL file. In our case, the following does the trick:

```
<script src="overlay.js"/>
```

One of the most important points to remember when writing extensions is that JavaScript's global namespace is shared between all the extensions as well as the core browser code. This means that developers need to use techniques to prevent name clashes. One simple method is to add a unique prefix to all of your variables and functions. The preferred method is to create an unnamed object that contains all of your variables and functions. Taking a look at the

ASA COMPUTERS
The Expert on Customized Servers!
www.asacomputers.com
1-800-REAL-PCS

Hardware Systems for the open source community - Since 1989.

(Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MS etc.)

The AMD Opteron™ processors deliver high-performance, scalable server solutions for the most advanced applications. "Runs both 32-and 64-bit applications simultaneously".

Your Custom Appliance Solution!!

"Let us know your Needs..."



"We will build you a Solution...."

AMD Opteron(TM) Value Server Starts at \$847



- 1U 14" Deep 280W.
- AMD Opteron 140 CPU.
- 512MB PC 3200 DDR ECC Unbuffered.
- Support upto 8GB DDR RAM.
- 40GB SATA Hard Disk.
- 2x 10/100 Mbps Lan.

Quad AMD Opteron(TM) Server Starts at \$2,812

- 1U AMD Opteron Model 840.
- 2GB Memory. Max 128 GB
- Supports upto 64GB FBDIMM.
- 80 GB SATA II Hotswap Hard Drive.
- 2x Integrated Dual 10/1000 LAN.



Dual AMD Opteron(TM) Storage Starts at \$4,120



- 5U Dual AMD Opteron Model 246.
- iSCSI or NAS Software Options.
- Support upto 18TB of Storage.
- Fail Hard Drive LED Indicator.

Dual AMD Opteron(TM) Storage Starts at \$8,445

- 8U AMD Opteron Model 246.
- 4TB of Storage (36TB Max).
- 1GB RAM
- 2 x 10/100/1000 Gigabit LAN.
- NAS or iSCSI Software Options.



Why Do Business With ASA?

"We Provide Approved EVAL Server..."

Since 1989, ASA has served customers like Cisco, Juniper, Caltech, Fermilab and most Universities. We provide a total custom solution with OS of your choice.

Excellent pre and post-sales support.

"Reliable hardware at the most competitive prices".

Please call or contact us for your next hardware purchase.



2354 Calle Del Mundo, Santa Clara, CA - 95054

www.asacomputers.com

Email: sales@asacomputers.com

Tel: 1-800-REAL-PCS, Fax: 408-654-2910.



Listing 3. Expanded Version of the picnic.xul File

```

<?xml version="1.0"?>

<!DOCTYPE picnic SYSTEM "chrome://picnik/locale/picnik.dtd">

<?xml-stylesheet href="chrome://picnik/content/toolbar.css"
type="text/css"?>

<overlay id="picnik-overlay"

xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
xmlns:html="http://www.w3.org/1999/xhtml">

<script type="application/x-javascript"
src="chrome://picnik/content/common.js"/>
<script type="application/x-javascript"
src="chrome://picnik/content/contextmenu.js"/>
<script type="application/x-javascript"
src="chrome://picnik/content/screengrab.js"/>

<popup id="contentAreaContextMenu">
  <menuitem class="menuitem-icomic"
id="picnik-ctx-edit"
insertafter='context-viewimage'
label="&picnik.edit_picture;"
oncommand="picnikContextMenu.editImage();"
image="chrome://picnik/content/picnik_16x16.png"/>

  <menu class="menu-icomic" id="picnik-ctx-grab"
insertafter='context-sendpage'
label="&picnik.menu;"
image="chrome://picnik/content/picnik_16x16.png">
    <menupopup>
      <menuitem label="&picnik.grab_visible;"
oncommand="picnikScreenGrab.grabVisible();"/>
      <menuitem label="&picnik.grab_full;"
oncommand="picnikScreenGrab.grabFull();"/>
    </menupopup>
  </menu>
</popup>

  </menu>
</menupopup>

<toolbarpalette id="BrowserToolbarPalette">
  <toolbarbutton id="picnik-button"
type="menu-button"
class="toolbarbutton-1"
label="&picnik.label;"
tooltiptext="&picnik.menu;"
oncommand="picnikScreenGrab.grabVisible();">
    <menupopup>
      <menuitem label="&picnik.grab_visible;"
oncommand="picnikScreenGrab.grabVisible();"
event.stopPropagation();"/>
      <menuitem label="&picnik.grab_full;"
oncommand="picnikScreenGrab.grabFull();"
event.stopPropagation();"/>
    </menupopup>
  </toolbarbutton>
</toolbarpalette>
</overlay>

```

auto-generated extension we see:

```

var linuxjournal = {
  onLoad: function() {
    ...
  },
  showContextMenu: function(event) {
    ...
  },
  onMenuItemCommand: function(e) {
    ...
  },
};
window.addEventListener("load", function(e)
  ➤{ linuxjournal.onLoad(e); }, false);

```

Using this technique, our extension has exactly one entry in the global

namespace (linuxjournal). This makes name clashes easier to avoid.

Notice the call to `window.addEventListener`. This ensures that our `onLoad` function is called when the overlay is loaded. In the case of the generated code, it creates a variable that we can use to access the string bundle.

Picnik

Now that you have a basic understanding of how extensions are created, let's move on to a real-world example. My employer, Bitnik, recently published an API for its Flash-based photo editor, Picnik (www.picnik.com). Bitnik's initial plan was to make our service easy to integrate into third-party Web sites. However, the API also opened the doors to using extensions to achieve a level of integration that we couldn't get with Flash alone.

My first goal was to add a simple context menu item to allow users to edit existing photos easily. Luckily, most of that code is already in the code generated by the extension wizard. See Listings 1 and 2 for the full text of the two most important files (picnik.xul and

Advertiser Index

For advertising information, please contact our sales department at 1-713-344-1956 ext. 2 or ads@linuxjournal.com.
www.linuxjournal.com/advertising

contextmenu.js). I also created common.js to store some variables that will be shared between files as we add features.

Most of the modifications occurred in the function showContextMenu. This function is called immediately before the context menu is actually shown to the user. This gives our extension a chance to modify the menu items on the fly. In our case, I wanted to show only the Edit in Picnik option when the user was actually right-clicking on an image.

Firefox provides our function with a global variable (gContextMenu), which contains a wealth of information about what the user clicked on. For example, gContextMenu.onImage is true when the user activated the menu on an image. First, showContextMenu gets a reference to the actual menu item via its ID of picnik-ctx-edit. Then, it hides the item if the user didn't click on an image. Finally, the function saves the image URL so that the extension knows what image to load into Picnik if the user actually selects the menu item.

When the user selects the menu item, the browser calls picnik.editImage. This function constructs a URL to pass to Picnik and then creates a new tab with that URL. The server at picnik.com first downloads the image and then responds with a page containing the actual Flash application and the image, ready for editing.

Screenshots

While browsing on the Mozilla Developer Center's Web site, I encountered something that I saw as a natural improvement for Picnik's Firefox extension—the ability to take screenshots of complete Web pages. The first part of the puzzle is the new Canvas HTML element that provides a flexible 2-D drawing canvas for JavaScript. Canvas originally was envisioned as a way to produce dynamic graphics client side.

Two additional Canvas functions make screenshots possible. The first is the drawWindow function. drawWindow, as its name implies, renders an XUL window to the canvas. In our case, we'll use it to render the Web page. The second important function is toDataURL, which allows a script to get an image of what's on the canvas.

Typically, URLs reference an object on a remote server or in the local filesystem. Data URLs store the actual object as part of the URL. This can be handy for embedding small graphics directly in CSS or HTML. This technique allows the browser to avoid another request to the Web server. In our case, we'll use it to get a PNG file of our canvas.

As you can see in Listing 3, picnik.xul has been modified to add a pop-up menu to both the context menu and the Tools menu. An additional file, screengrab.js (Listing 4, available on the *Linux Journal* FTP site—see Resources), contains the code for actually grabbing the screenshot. As with the context menu code, this file also has an onLoad. In this case, however, the function's job is to detect whether either Canvas or toDataURL is missing. If so, it disables the screen grab functionality. This allows the extension to run on Firefox 1.5 without confusing error messages.

The two functions grabFull and grabVisible set up the parameters to grab either the full page or the visible area, respectively. They leave the bulk of the work to the aptly named grab function. In order to limit the amount of data that needs to be uploaded, grab scales the canvas so that it will be smaller than 2800x2800 before actually rendering the window. Next, grab creates a canvas and renders the window before retrieving a data URL via toDataURL.

saveDataURL has the job of actually sending the image to Picnik. It constructs a multipart/form-data request containing several API parameters as well as the image data. Picnik's servers respond with a URL for the browser to load. When that happens, requestState is called with a

Advertiser	Page #	Advertiser	Page #
ABERDEEN, LLC www.aberdeeeninc.com	33	LPI www.lpi.org	75
AML www.amltd.com	71	MICROWAY, INC. www.microway.com	C4, 73
APPRO HPC SOLUTIONS appro.com	C2	O'REILLY RAILS CONFERENCE www.railsconfeurope.com	83
ARCOM CONTROL SYSTEMS www.arcom.com	63	POGO LINUX www.pogolinux.com	7
ASA COMPUTERS www.asacomputers.com	29, 79	POLYWELL COMPUTERS, INC. www.polywell.com	35
AVOCENT CORPORATION www.avocent.com/remotecontrol	1	THE PORTLAND GROUP www.pggroup.com	23
CARLNET www.carl.net	52	QSOL.COM www.qsol.com	5
CORAID, INC. www.coraid.com	13	RACKSPACE MANAGED HOSTING www.rackspace.com	C3
COYOTE POINT www.coyotepoint.com	3	R CUBED TECHNOLOGIES www.rcubedtech.com	41
EMAC, INC. www.emacinc.com	69	SD BEST PRACTICES www.sdexpo.com	65
EMPERORLINUX www.emperorlinux.com	19	SERVERS DIRECT www.serversdirect.com	53, 57
FAIRCOM www.faircom.com	59	SILICON MECHANICS www.siliconmechanics.com	31, 91
GENSTOR SYSTEMS, INC. www.genstor.com	67	SUPERMICRO www.supermicro.com	15
HURRICANE ELECTRIC www.he.net	27	TECHNOLOGIC SYSTEMS www.embeddeddx86.com	61
IRON SYSTEMS www.ironsystems.com	89	TOTALVIEW TECHNOLOGIES www.totalviewtech.com	9
LINUX JOURNAL www.linuxjournal.com	6, 77, 82	UNIWIDE TECHNOLOGIES www.uniwid.com	11
LOGIC SUPPLY, INC. www.logicsupply.com	85	VERIO www.verio.com	43

Do you take

"the computer doesn't do that"

as a personal challenge?

So do we.

LINUX
JOURNAL™

Since 1994: The Original Monthly Magazine of the Linux Community

Subscribe today at www.linuxjournal.com

req.readyState of 4. Finally, requestState creates a new tab with that URL.

Distribution

Most Firefox extensions are distributed at Mozilla's Add-ons site as Cross-Platform Installs (.xpi). This site provides users with a central trusted source for extensions. Because the site is pre-trusted by the browser, it makes installation easier for users. The Add-ons site also makes it easy to provide auto-updates to users. You simply can upload a new version, and users automatically will be prompted to upgrade.

The extension wizard created a shell script (build.sh) to automate the process of creating the XPI. Once it's created, you can send it to friends, post it on your blog, upload it to addons.mozilla.org or distribute it any way you see fit.

Conclusion

I hope this article has given you an idea of how easy it is to write Firefox extensions and how powerful they can be. Don't forget that you can learn by using the Chrome List extension to explore the inner workings of both the browser and other extensions. Also, the Mozilla Developer Center has a wealth of how-tos and reference resources waiting to be tapped. Happy coding! ■

Justin Huff is a Software Engineer with an embedded systems background, working for a little Web 2.x company called Bitnik. He resides in Seattle, Washington.

Resources

Code for this article, including screengrab.js (Listing 4):
[ftp.linuxjournal.com/pub/lj/listings/issue160/9730.tgz](ftp://linuxjournal.com/pub/lj/listings/issue160/9730.tgz)

Extension Developers Extension: ted.mielczarek.org/code/mozilla/extensiondev

Console?: <https://addons.mozilla.org/en-US/firefox/addon/1815>

Chrome List: <https://addons.mozilla.org/en-US/firefox/addon/4453>

Venkman: <https://addons.mozilla.org/en-US/firefox/addon/216>

Firefox Extension Wizard: ted.mielczarek.org/code/mozilla/extensionwiz

Mozilla Developer Center: developer.mozilla.org

Mozilla Add-ons: addons.mozilla.org

Picnik API: www.picnik.com/info/api



17-19.09.2007
BERLIN GERMANY



Learn why Rails is taking Europe by Storm

Register Now and Save 10% ■ Use discount code `rce07ljr`

www.railsconfeurope.com

Streaming Audio with Ices and Icecast

Retransmit from a radio scanner to the Internet via Ices and Icecast. BRIAN MATHERLY

The Sioux Empire Amateur Radio Club operates a repeater on 146.895MHz with the call name W0ZWY. On Tuesday evenings, it provides a time for club announcements. I like to listen in when I can, but sometimes I am still at work when the announcements start and can't get to my radio. I found a way to solve this problem by using a UHF/VHF scanner, Linux, Ices and Icecast. By rebroadcasting the transmission over the Internet, I can listen to the local club repeater anytime, anywhere.

Tuning In

The first step was to find a VHF tuner that could tune to 146.895MHz and output analog audio to a sound card. I found that the least expensive way to go was to buy a consumer-grade UHF/VHF scanner. The scanner needed to have either a line-level output or a headphone jack. I found that line-level outputs are rare, but there are a number of scanners with headphone jacks. I also wanted one with a digital tuner so that it would not drift off frequency over time. I found a Radio Shack PRO-2050 on eBay for around \$75 US that met all my requirements.

Using an F-to-BNC adapter, I connected the scanner's antenna jack to the off-air TV antenna on the roof of my house. This worked well because 146.895MHz lies right between off-air TV channels 6 and 7. I spent some time setting the squelch as high as I could, so there would not be static when no one was transmitting. Figure 1 shows my scanner with the audio cable plugged in to the headphone jack.

Getting Connected

In order to put the audio on the Internet, it first must be encoded. Then, that encoded information can be made available for streaming to clients on the Internet. For these tasks, Icecast and Ices make a great team. Icecast is a media streaming program that supports Ogg



Figure 1. Radio Shack Scanner with Audio Cable Attached

Vorbis or MP3 streams. It receives encoded media from one or more sources and makes it available for streaming to multiple clients. Ices is an Ogg Vorbis audio encoder that works well with Icecast. I chose Ogg because it is a patent- and royalty-free format. Plenty of players can decode Ogg. XMMS on Linux and Winamp on Windows are the most common. If you are interested in using MP3 encoding, check out Livelce by following the link in the Resources for this article.

The encoding and streaming tasks can be done on the same machine, but I chose to do them on separate machines. This makes it easier to add more sources later. Additionally, it removes some of the processing burden from the streaming machine.

For the encoding computer, I chose a 233MHz Pentium I computer that had been unused for many years. I connected the headphone jack from the scanner to the line-in jack on the sound card using a 3.5mm-to-3.5mm audio cable that came from an old set of computer speakers. I connected the network interface to my LAN and set the IP address to 192.168.1.21.

For the streaming computer, I chose to use my existing router, which doubles as a Web server. The main reasons I chose it are that I

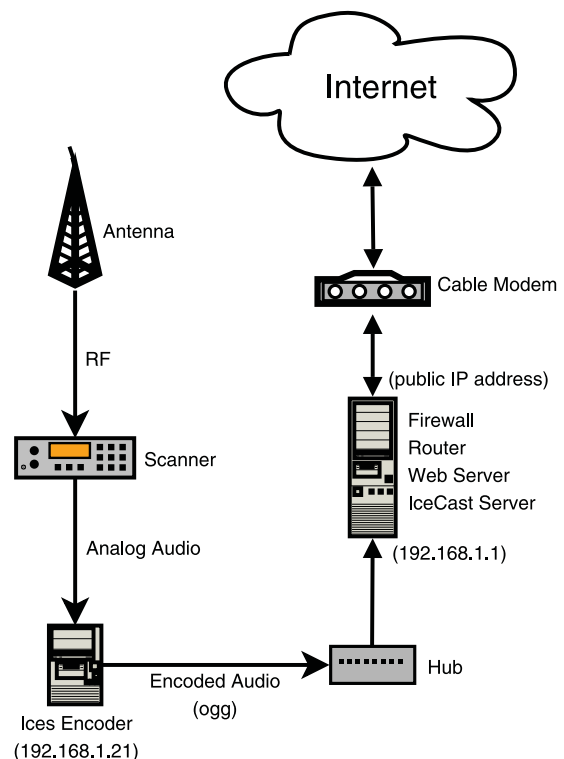


Figure 2. Network Diagram

already own it and it has my only public IP address. This computer has two network interfaces. One is connected to my cable modem and has a public IP address. The other interface is connected to my LAN and has the private IP address 192.168.1.1. Another way to accomplish this would be to use a consumer-grade cable/DSL router and forward port 8000 to the streaming computer's IP address.

Figure 2 shows a diagram of my network, including the scanner, encoder, streamer and cable modem.

Icecast Configuration

My streaming computer is running Mandriva. Icecast is available from Mandriva's contrib RPM repository. Once I added that repository, installing Icecast was as easy as typing `urpmi icecast` from the command prompt. If you want to install from source, you can download the latest release from the Icecast Web site (see Resources). Simply click the download link, extract the archive, and run the familiar commands `./configure; make; make install`.

The Icecast configuration file is an XML file and usually resides in `/etc/icecast.xml`. I was able to use most of the default settings, but I highlight some of the changes I did make here.

The limits section of the file allows you to set the maximum number of source streams (encoders) and the maximum number of clients that can connect to the streaming computer at once. The upload data rate on my cable modem is limited to 256Kbps by my ISP. As such, I need to limit the maximum number of clients to be sure that I don't use all of my upload bandwidth:

```
<limits>
  <clients>10</clients>
  <sources>2</sources>
</limits>
```

The authentication section is where you specify the user names and passwords. The source-password is the password used by the encoding machine when it connects. The relay-password is used by relays, which I am not using in my configuration. The admin-user and admin-password allow access to the administration Web page. It is important to change all passwords from the default for security:

```
<authentication>
  <source-password>hackme</source-password>
  <relay-password>hackme</relay-password>
  <admin-user>admin</admin-user>
  <admin-password>hackme</admin-password>
</authentication>
```

The hostname is used so that Icecast knows what address to append to the beginning of the links on the Web page:

```
<hostname>example.com</hostname>
```

The listen-socket allows you to set the port on which Icecast listens; 8000 is the default:

```
<listen-socket>
  <port>8000</port>
</listen-socket>
```

Once the configuration file is all ready, you can start Icecast by running the init script: `/etc/init.d/icecast start`. If there are any errors during startup, look in the log files to debug them.

Ices Configuration

My encoding computer is also running Mandriva. Because this machine is running only at 233MHz with 64MB of RAM, I decided to do a minimal install and leave off the window manager. Ices is also available from the contrib repository. So once again, typing `urpmi ices` is all it took to install Ices. However, you can download the latest releases from the Ices Web site (see Resources). Make sure you use the 2.0 series if you want to use Ogg. It also requires that libshout is installed. Once you have that, you can extract the archive and run `./configure; make; make install` to get everything installed.

The Ices configuration file is also an XML file. It is usually stored in `/etc/ices.conf`.

The beginning of the configuration file has some settings that dictate how Ices runs. It can be helpful to change these during the initial setup, so that Ices runs in the foreground and sends messages to the console. These messages can be very helpful in debugging problems on initial setup. Once everything is set up and running, make sure Ices runs in the background and logs



Mini-ITX Systems & Solutions for Embedded Applications, Industrial & Mobile Computing.

Mini-ITX Mainboards with VIA, Intel, or AMD processors.

- x86 platform
- small form factor design
- Linux and Windows XP compatible
- fully customizable systems



Fanless Mini-ITX Systems

Utilizing heat pipe technology, these completely fanless Mini-ITX PCs offer durability, security, and power-efficiency for a range of embedded applications in harsh, remote environments.



LOGICSUPPLY.COM
info@logicsupply.com
802 244 8302

Icecast is a media streaming program that supports Ogg Vorbis or MP3 streams.

messages to a file:

```
<background>1</background>
<logpath>/var/log/ices</logpath>
<logfile>ices.log</logfile>
<loglevel>3</loglevel>
<consolelog>0</consolelog>
```

The rest of the file is under the stream section. This is where you configure settings specific to this particular audio stream. Within the stream section, the metadata section is where you specify information about the stream. This information will be displayed on the Icecast Web page:

```
<metadata>
  <name>W0ZWY 146.895 MHz</name>
  <genre>Live</genre>
  <description>Live feed of the
    W0ZWY repeater</description>
</metadata>
```

The input section is the place to define where the audio actually comes from. There are many possibilities, including options for playlists

and scripts. Because I want to encode live audio, I used the oss module. Don't be alarmed if your system uses the ALSA sound system instead of OSS. ALSA has OSS compatibility, so this module works with both ALSA and OSS. Use the device parameter to specify the sound device from which to get data. On most systems it will be /dev/dsp. The rate parameter specifies the sample rate of the data in hertz. Most devices use 44100. Use the channels parameter to specify the number of channels available for capture. For most devices this will be 2 (stereo):

```
<input>
  <module>oss</module>
  <param name='device'>/dev/dsp</param>
  <param name='rate'>44100</param>
  <param name='channels'>2</param>
</input>
```

The instance section allows you to specify the number of instances of this stream. You might have more than one instance if you want to send the stream to more than one server, or if you want to have different versions of the same stream at different bitrates. For my system, I want only one instance.

The first part of the instance section is where you specify the streaming server information. The hostname tells Ices where to send the data. The port and password must match the values you specified in the Icecast configuration file. The mount option specifies what the name of the stream will be called.

The encode section specifies how the audio will be encoded. The easy way to do it is to set the sample rate and channels to match the

Ices Settings and Bitrates

The quality, sample rate and channels settings all will affect the bitrate of the stream. Table 1 shows various combinations of these settings and their resulting bitrates.

	QUALITY	SAMPLE RATE	CHANNELS	BITRATE
Low-Quality Talk Radio	0	11127	1	20Kbps
Medium-Quality Talk Radio	2	11127	1	30Kbps
Low-Quality Music	0	44100	2	56Kbps
Medium-Quality Music	2	44100	2	84Kbps
High-Quality Music	4.5	44100	2	132Kbps

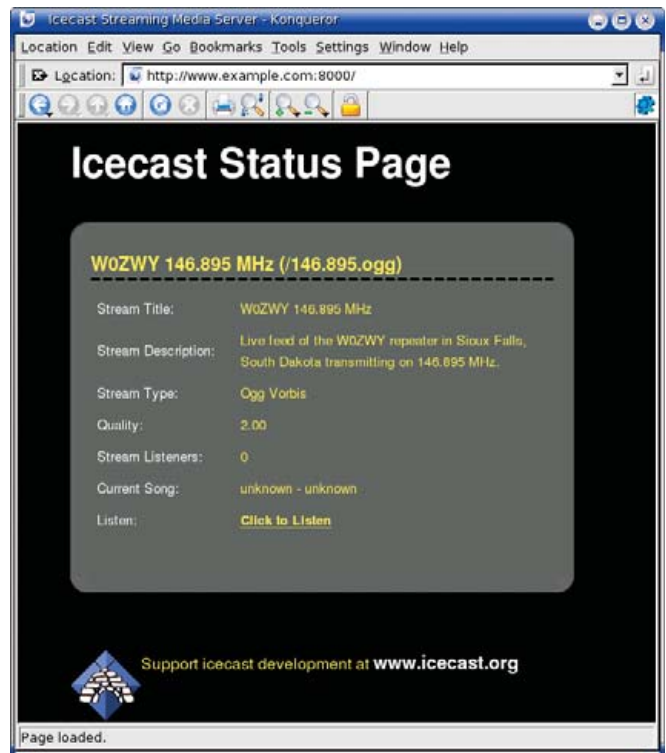


Figure 3. Icecast Status Page

input section above. But, I didn't need that much quality for my stream. So, I used the downmix and resample sections to tell Ices to resample the audio to 11127Hz and downmix it to 1 channel (mono). There are two options you can use to adjust the final bitrate of the stream: quality and nominal-bitrate. Notice that I commented out nominal-bitrate and set quality to 2:

```
<instance>
  <hostname>192.168.1.1</hostname>
  <port>8000</port>
  <password>hackme</password>
  <mount>/146.895.ogg</mount>

  <encode>
    <quality>2</quality>
    <!--nominal-bitrate>32000</nominal-bitrate-->
    <samplerate>11127</samplerate>
    <channels>1</channels>
  </encode>

  <downmix>1</downmix>

  <resample>
    <in-rate>44100</in-rate>
    <out-rate>11127</out-rate>
  </resample>
</instance>
```

Once the configuration file is all ready, make sure that Ices has permission to access the audio device through /dev/dsp. Mandriva creates an audio group, which owns the /dev/dsp file. It also creates an ices user and group when ices is installed. I simply added ices as a member of the audio group in the /etc/groups file by editing the audio group:

```
audio:x:81:ices
```

Finally, you can start Ices by running the init script: /etc/init.d/ices start. If there are any errors during startup, look in the log files to debug them. It also can be helpful to examine the Icecast log files on your streaming computer to debug problems.

Once everything is up and running, you can access the Icecast status page on the port you specified in the configuration file. Figure 3 shows an example of an Icecast status page. Clicking on Click to Listen launches your audio player.

Fine-Tuning

Once I had everything set up and running, I spent a fair amount of time fine-tuning the audio. The first thing I found was that the audio level was quite low. Normally, when I want to turn up the audio level, I use KMix. But, I hadn't installed a window manager on my encoding computer. My system uses ALSA as the sound system, and I found out that there is a great curses-based mixer for ALSA named AlsaMixer. It allows you to use the arrow keys to change settings. Figure 4 shows the settings I chose for my card. My final preference was to have the Master and PCM both at 100%. I also chose to mute everything else, because I knew they would not be in use.

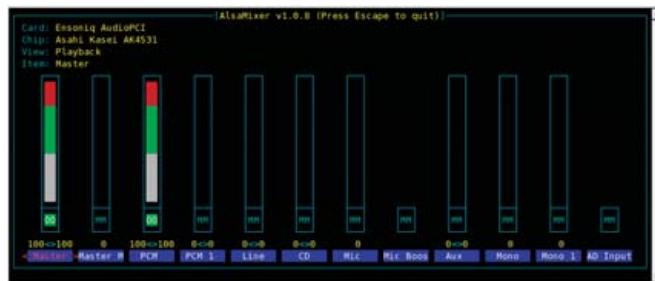


Figure 4. AlsaMixer Settings

Now that the audio was loud enough, I still had another problem. The audio sounded choppy. After looking around for a while, I realized that the processor usage was at 100%. I tried to reduce the bitrate by changing the value of the nominal-bitrate setting. With the bitrate set sufficiently low, the audio sounded good. I guess you can't encode very high-quality audio with a 233MHz processor.

When the audio sounded good, my goal was to lower the bitrate as much as possible while keeping an acceptable level of audio quality. In my case, an "acceptable level" is quite low. The traffic on the W0ZWY repeater is only voice, because it is illegal to transmit music in the amateur radio bands.

Three aspects of the encoded audio will affect its quality and bitrate: number of channels, sample rate and bitrate. I started by setting the downmix to 1, because the scanner is only mono. Then, I incrementally decreased the sample rate until I found the lowest setting that rendered acceptable audio; 11127Hz ended up being the minimum. Instead of setting the bitrate with the nominal-bitrate setting, I used the quality setting. The quality setting directly affects the final bitrate. With a few tries, I chose a quality level of 2.

With my final settings, the bitrate stays around 30Kbps, and the processor hovers around 40% usage.

Conclusion

There are probably many other applications where this type of system would be useful. Keep in mind, however, if you are interested in using this type of system for streaming music, make sure you have permission from the artists and recording studios.

This combination of open-source projects really worked well together. Setting up everything was quite simple. I spent most of my time tweaking the quality and sample rate settings. The traffic on my streamer is quite low. I usually have only one or two connections per week. But, it was fun to learn, and it is my little way of participating in the amateur radio community. ■

Brian Matherly is a Software Engineer at Sencore Electronics in Sioux Falls, South Dakota. He is also an adjunct professor at Colorado Technical University.

Resources

Icecast Web Site: www.icecast.org

Ices Web Site: www.icecast.org/ices.php

LiveIce Web Site: web.arm.ac.uk/~spm/software/liveice.html

Standard Operating Procedures for Embedded Linux Systems

Follow these procedures for the smoothest path to great embedded Linux.

CHI-HUNG CHOU, TSUNG-HSIEN YANG, SHIH-CHIANG TSAO AND YING-DAR LIN

Procedures for developing embedded systems are very complicated. New engineers typically take a long time to become familiar with these procedures. Therefore, we have developed a standard operating procedure (SOP) to save the costs of constructing an embedded system and reduce the complexity. The SOP includes five standard procedures for building a Linux-based embedded system, as shown in Figure 1. You can follow the procedures discussed in this article for building a prototypal system. Also, we introduce ten useful methods for downsizing your system. Finally, we show the effect of these methods on downsizing your embedded system—a content-aware network security gateway.

Five Standard Procedures

To build an embedded system, the first step is to select a target platform. The platform involves both hardware and software. The hardware platform includes the processor, bus and I/O; the software platform includes the bootloader, kernel and root filesystem. You must select each item in the target platform carefully to ensure that the hardware and software work together. For instance, bootloaders relate directly to the hardware. If the selected bootloader does not support your hardware platform, the whole embedded system cannot power on. Moreover, an operating system that requires MMU may fail to collaborate with MMU-supported processors.

Second, in addition to the target platform, a development platform also is necessary. You cannot compile embedded software programs on the target platform, because the target platform often has a small RAM and slow CPU to minimize cost and power consumption. Therefore, you need to prepare a development platform with a fast CPU and large RAM to compile these programs. Besides, because the two platforms

have different hardware architectures, a cross-compiler environment is necessary. Buildroot is such a package to offer this environment. It has a friendly user interface to assist in choosing the hardware platform and the required software package. By using Buildroot, you can generate a cross-compilation toolchain and a root filesystem easily with built-in application packages for your embedded system.

After setting up the environment, the next step is identifying the packages required by your system. You can accomplish this by selecting the built-in packages directly from the menuconfig of Buildroot, or you can download them from the Internet. In fact, Buildroot provides a list of useful packages, such as iproute2, freeswan and squid. Buildroot also ensures that these packages can link successfully with uClibc, a C library with a smaller size than Glibc. If you cannot find the suitable packages, you will have to modify existing packages or write new ones.

Having obtained the required packages, the next step is integrating them into the embedded system. Integrating here means using a cross-compiler to compile the source code into forms that can be executed in the target platform, and then adding them into the root filesystem. You can add packages into the root filesystem through Buildroot in any of three ways, as shown in Figure 2. Method 1 is to select them directly from the options in Buildroot. If the packages are not available in Buildroot, you may need to write a makefile for the package to indicate how to download, configure, compile and install the package. Also, you need to modify

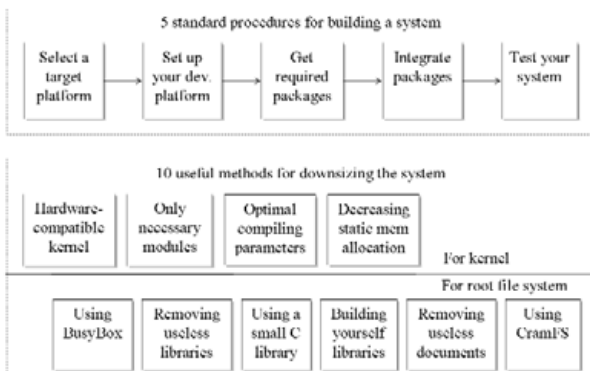


Figure 1. Procedures and Methods for Building and Downsizing Your Embedded Systems

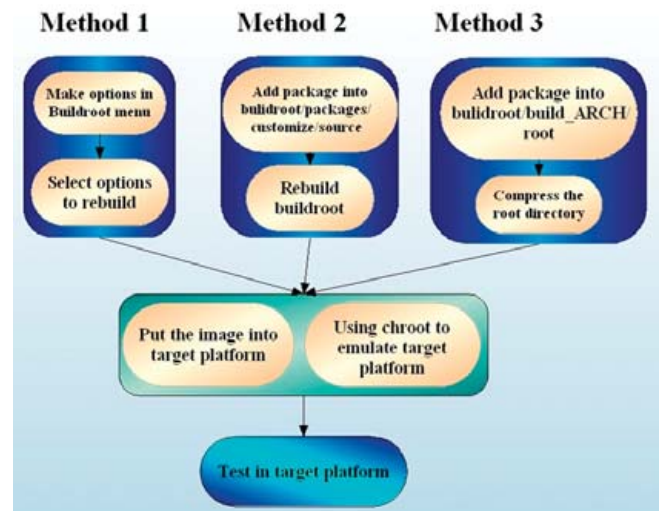


Figure 2. Three Methods for Adding Packages into Your Root Filesystem

the config.in file of Buildroot to display the option of the package in the configuration menu. However, if you would not like to write these configuration files, you can use Method 2. In Method 2, you simply place the compiled packages in the directory named customize, and then Buildroot copies these compiled packages into the root filesystem during the building procedures, according to the rules given in customize.mk. However, if you simply want to verify the functionality of a single package, you don't need to rebuild the whole image. The steps in Method 3 are to mount the root filesystem on any one directory and then copy the compiled packages into the directory. Finally, unmount the directory, and you will get an updated root image. However, Method 3 may fail if the free space in the mounted filesystem is not enough for the new packages. In that case, you can adjust the parameters given in ext2root.mk to reserve more free space in your root filesystem during the period of system development.

Finally, Figure 2 depicts two ways to test and verify whether the functions of a package are normal. The basic way is to download the root filesystem into the target platform and execute the package directly. However, doing this usually takes a long time. Another way is to boot the root filesystem in a virtual machine, such as QEMU and VMware. Using a virtual machine to examine a compiled image is fast and convenient, but a virtual machine may not simulate some characteristics, such as hardware interrupts. Hardware interrupts involve quick reaction behavior, so they cannot be implemented by virtual machines easily. Finally, if the target platform has the same CPU architecture as your development platform, you can use chroot to replace your local system with the target root filesystem.

By following the five procedures outlined, you can build the root filesystem for your embedded system. However, there is still one problem that may be troubling you—how to downsize your embedded systems or how to use less Flash RAM to store the kernel and root filesystem. Requiring less RAM means that you can cut the cost of your embedded system.

Ten Downsizing Methods

The organization of the downsizing issue is displayed in the bottom of Figure 1. We divide the methods into two parts, because the software platform of an embedded system typically consists of a kernel and root filesystem. The first part is how to get a small kernel, and the second part is how to downsize each component in the root filesystem, including libraries and shells. The second part also discusses how to compress the whole root filesystem. We describe all methods in detail below, along with experimental results. After explaining all methods, we show the effect of these methods on our laboratory embedded system, called the Wall system. Table 1 presents the specification for the Wall system. The system is a network security gateway that provides application-layer content filters, such as antispam and antivirus.

Methods for the Linux Kernel

Selecting an appropriate kernel is the first

step in downsizing the kernel. If you choose an inappropriate kernel, the system may be not only large but also unable to use processor power effectively. For example, a standard Linux kernel on a hardware platform without MMU cannot work normally. Such a hardware platform requires a specific MMU-less kernel, such as uClinux. Most people use the standard Linux kernel and attempt to trim its size.

The next step is to include only the necessary modules in the standard Linux kernel by a correct configuration. In fact, the default configuration of a Linux kernel includes many unused modules, which causes you to have a big kernel. Figure 3(a) shows the experimental results on the downsizing effect of the correct configuration. In this case, a system supporting TCP/IP has a kernel image that is only 59.84% of the size of a system supporting all network protocols.

To downsize the kernel, the third step is to use the optimization parameters when compiling the kernel. Using parameters -O1, -O2 or -O3 can improve performance, and using -Os can reduce size. However, optimizing for both performance and size simultaneously is not possible. Therefore, we generally select -O2 to achieve a balance between size and performance. As shown in Figure 3(b), the -Os parameter reduces the size of the kernel image by 22.82% as compared with -O3, but it causes worse performance.

Besides including only the necessary modules and compiling the kernel with the optimal parameters, to downsize the kernel further, you can

Ultra Dense, Powerful, Reliable... Datacenter Management Simplified! 15" Deep, 2-Xeon/Opteron or P4 (w/RAID) options



Customized Solutions for... Linux, BSD, W2K

High Performance Networking Solutions

- Data Center Management
- Application Clustering
- Network and Storage Engines

Rackmount Server Products

- **1U Starting at \$499:** C3-1GHz, LAN, 256MB, 20GB IDE
- 2U with 16 Blades, Fast Deployment & more...



iron
SYSTEMS™

Iron Systems, Inc.

540 Dado Street, San Jose, CA 95131

www.ironsystems.com

CALL: 1-800-921-IRON

Table 1. Specification of Wall

	FUNCTIONS	PACKAGES
Kernel	X86, MMU, QoS, Ethernet, Wireless	Linux 2.6.6; 1,302,362 Bytes
Connection	LAN, DMZ, WAN, DHCP, DNS relay, Dynamic DNS, Link load balance, Bridge mode	ppp-2.4.1, rp-pppoe-3.5
Security	IPSec, PPTP, L2TP, SSL-VPN	freeswan2.06, 12tpd-0.69
Firewall	NAT, firewall, UPNP, traffic profiling, APP firewall	iptables-1.2.9, hotplug, iproute2
Mail	Antispam, antivirus, POP3 proxy	p3scan
Web	Transparent proxy, URL, URL keyword, content keyword	p3scan
IM	MSN log	Development based on L7Filter
BW Control	TC	TC
Management	Web, SSL, FTP, log rotation	thttpd-2.21b, Openssl-0.9.7d, putre-ftp-1.0.17a, cron
Platform	i386, IXP (simple version)	

decrease the size of the static buffer and array allocated in the kernel, because the kernel typically declares a large buffer and array for standard PCs. To find out which buffer or array occupies large memory space, you can use the command nm. This command can list the allocated size of each variable in an object file. With that information, you can browse the corresponding source code of the object file and alter the initial size of the buffer or array. Another approach for shrinking the buffer size is to modify the options in the menuconfig of the kernel to decrease the maximum number of supported peripherals, as shown in Figure 3(c) and (d).

Methods for the Root Filesystem

As shown in Figure 1, we identify six methods for downsizing the root filesystem. First, you can adopt a tool called BusyBox, which provides a fairly complete environment for any small or embedded system. BusyBox combines tiny versions of many common UNIX utilities into a single small executable file, and it is highly modular, allowing commands to be included or excluded at compile time. The space used for BusyBox is 7.04% of that of the original tool, as demonstrated in Figure 4.

Next, we introduce three methods for removing unused libraries or downsizing required libraries. First, you can use the command ldd to identify the required shared libraries for each program, and then with this information, you can remove the unused libraries. Notably, if a shared library is not used by programs, you additionally should check whether it is used by other shared libraries. Figure 4 shows that removing redundant libraries reduces the root filesystem to 6.55% of its original size. Second, you can replace the standard C library with a small C library, such as uClibc, Newlib or diet libc. Such libraries remove the unused functions, so their size is smaller than Glibc, as shown in Table 2. This table presents the differences in functionality between the four libraries. Third, you can use a library optimizer tool named Libopt to rebuild the libraries that include the only necessary functions for the executable programs and shared libraries found in the root filesystem. This tool utilizes objdump and nm to gather information about library object files, shared libraries and executable programs.

The fifth method for downsizing the root filesystem is to remove

Table 2. Comparison between Different C Libraries

	GNU C Library	uClibc	diet libc	Newlib
Size	Largest	Small	Smallest	Small
Compatibility	Good	Good	Bad	Normal
Speed	Fastest	Fast	Fast	Fast
Portability	Yes	Yes	Yes	Yes
MMU-less supporting	No	Yes	Yes	Yes
Licensing	LGPL	LGPL	LGPL	BSD, GPL
Setting		menuconfig	only make	./configure
Note	Standard C library	Needs cross-compiler toolchain	Often linked as static library	Managed by Red Hat

unnecessary documents. You can eliminate some directories, such as /home, /mnt, /opt, /root, /boot and /proc, if unused. You also can remove the man, info, include and example directories to reduce the size when additionally integrating a package into the root filesystem. In general, an embedded system executes only specific programs, so users can operate it easily without the help documents or examples in these directories.

The final method is to avoid uncompressing the whole root filesystem into SDRAM. The root filesystem is compressed to save the stored space, for example, Flash RAM. However, after the filesystem is uncompressing into SDRAM, the Flash memory allocated for the filesystem is no longer necessary. For instance, if the compressed size of the root

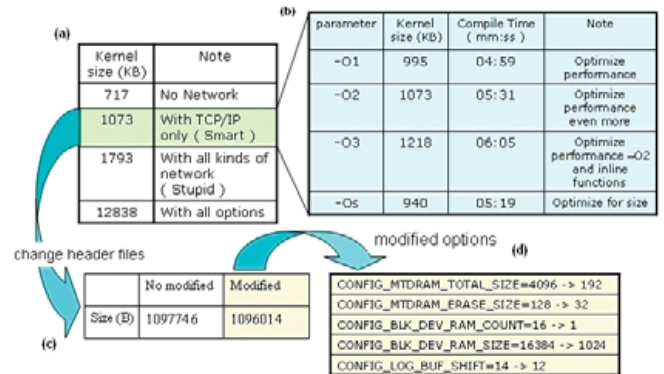


Figure 3. Effects of the Downsizing Methods on the Kernel

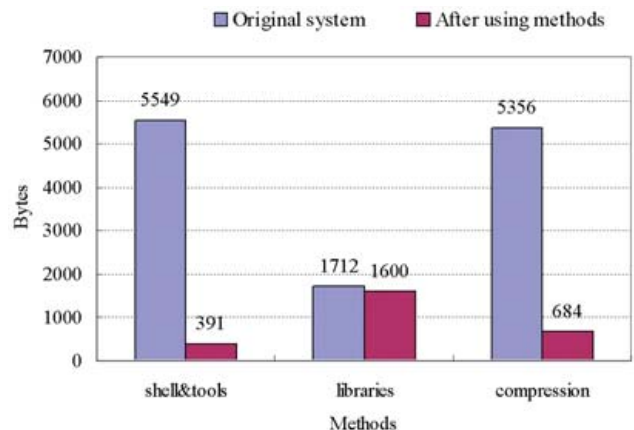


Figure 4. Effects of Downsizing Methods on the Root Filesystem

filesystem is 4MB and its compression rate is 50%, the system occupies 4MB of Flash memory and 8MB of SDRAM. Therefore, the system wastes much memory storage, because of the duplicate data. For this problem, you can use CRAMFS. CRAMFS is a read-only filesystem, designed for simplicity and space efficiency. You do not need to uncompress a CRAMFS image before mounting it. A CRAMFS image is zlib-compressed, one page at a time to enable random read access. The metadata is not compressed, but is expressed in a terse representation that is more space-efficient than in traditional filesystems, such as ext2 or FAT. However, due to the read-only property of compressed files, random write access is hard to implement for them. As shown in Figure 4, CRAMFS compresses the filesystem to 12.77% of its original size.

Now that we've covered the six methods, let's move on to the effect of these methods on the Wall Project, as shown in Figure 5. First, we used BusyBox to substitute for the multiple utility programs used in the original shell. Then, we compiled all the required packages with the parameters --strip-unneeded and -O2. Next, we used the commands strip and objcopy to remove the unnecessary contents of packages. Finally, we deleted unnecessary directories, such as man,

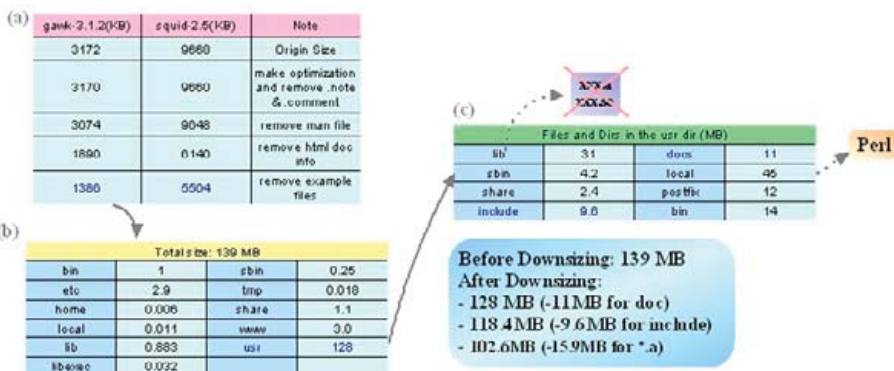


Figure 5. Downsizing Results on the Root Filesystem of the Wall Project

info and example. Figure 5(a) illustrates the result of these processes. However, the size of Wall was still 139MB. Hence, we had to view the contents of /usr indepth, as shown in Figure 5(b) and (c). In the Wall Project, removing unneeded documents and files saved 20.6MB of space. About 15.9MB of space then can be saved by eliminating unused libraries. However, as you can see, Perl occupied much space in our system. Other methods may exist to solve this problem, but it is sufficient to consider only what we have done above.

We found that the optimization of package size is also useful for downsizing when integrating a new package into the root filesystem. Actually, most programs and libraries are compiled at optimizing level 2 by default (gcc options -g and -O2) and are compiled for a specific CPU. On



Expert Included.

Falko provides expert, dedicated technical support for one of the most comprehensive server and storage product offerings in the industry.

He appreciates the Rackform nServ K501 because he knows AMD's Direct Connect Architecture ensures that all processor cores will work together with maximum efficiency and optimized memory performance now. And its 2 Dual-Core AMD Opteron™ 2000 Series processors are engineered for seamless upgradeability to Quad-Core later.

Falko knows that the evolution of technology is constant. He is impressed that the Rackform nServ K501's 24 hot-swap + 2 internal SATA drive system with redundant power supply delivers reliability without sacrificing capacity.

When you partner with Silicon Mechanics, you get more than a long-term investment in AMD technology—you get an expert like Falko.



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

Silicon Mechanics and the Silicon Mechanics logo are registered trademarks of Silicon Mechanics, Inc. AMD, the AMD Arrow logo, AMD Opteron, and combinations thereof, are trademarks of Advanced Micro Devices, Inc.



Intel platforms, software is compiled for i386 processors by default. To minimize the package size, you should not adopt the `-g` option, which adds the debug info in the execution files. Additionally, remember to use `-strip` and `--strip-all` to remove all symbols. In more-advanced methods, we used the command `readelf` to check for any redundant sections in the execution files, and we used `objcopy` to remove those redundant sections. However, this approach may be not efficient for small programs.

Conclusion

This article describes the five procedures for making a Linux-based embedded system and describes ten methods for downsizing the kernel and the root filesystem. After we used these methods, our Wall Project was downsized by 26.18%. The experiment's results reveal that the two most efficient methods are giving correct kernel compilation parameters and using simplified tools and libraries in the root filesystem. Hopefully, this article helps you understand the procedures and problems when building a Linux-based embedded system. ■

Chi-Hung Chou is currently working on his Masters' degree in Computer Science at National Chiao Tung University. His research interests include mesh network and embedded systems. He can be reached via e-mail at payton345.cs95g@nctu.edu.tw.

Tsung-Hsien Yang is currently working on his Masters' degree in Computer Science at National Chiao Tung University. His research interests include automatic block module tests and embedded systems. He can be reached via e-mail at thyang.cs95g@nctu.edu.tw.

Shih-Chiang Tsao is a PhD candidate in Computer Science at National Chiao Tung University and has been advised by Dr Ying-Dar Lin since 2003. His research interests include TCP-friendly congestion control algorithms, fair-queuing algorithms and Web QoS. He can be reached via e-mail at weafon@cs.nctu.edu.tw or through his Web site (www.cs.nctu.edu.tw/~weafon).

Ying-Dar Lin received a PhD in Computer Science from the University of California, Los Angeles (UCLA) in 1993. He has been a professor of Computer Science at National Chiao Tung University since 1999. He also is the founder and director of the Network Benchmarking Lab (NBL), which reviews the functionality, performance, conformance and interoperability of networking products, ranging from switch, router and WLAN to network and content security and VoIP. His research interests include design, analysis, implementation and benchmarking of network protocols and algorithms, wire-speed switching and routing, quality of services, network security, content network and embedded hardware software co-design. He can be reached via e-mail at ydlin@cs.nctu.edu.tw or through his Web site (www.cs.nctu.edu.tw/~ydlin).

Resources

John Lombardo, *Embedded Linux*, 1st ed., New Riders, July 5, 2001.

Todd Fischer, "Optimizing Embedded Linux", *Dr. Dobbs's*, May 2002: www.dj.com/184405050.

Lei Yang, Robert P. Dick, Haris Lekatsas and Srimat Chakradhar, "CRAMES: compressed RAM for embedded systems", International Conference on Hardware Software Codesign, Proceedings of the 3rd IEEE/ACM/FIP international conference on Hardware/software codesign and system synthesis, Jersey City, New Jersey, 2005, pp: 93-98.

"Buildroot—Usage and documentation v1.2", December 28, 2004: buildroot.uclibc.org/buildroot.html.

Karim Yaghmour, *Building Embedded Linux Systems*, 1st ed., O'Reilly, 2004.

AlphaMail Is Scalable and Accessible Web Mail

AlphaMail takes a unique approach to providing a Web-based IMAP client.

TONY KAY

AlphaMail is a high-performance, feature-rich, open-source Web mail system created at the University of Oregon. The interface includes message snippets in indexes, UTF8 composition and numerous viewers for attachments (such as image icon preview and file listings from tarballs). It also tries to strike a balance between desirable features and too much interface noise. It was created to address several problems that exist with other open-source and commercial Web mail systems.

Performance

The first concern AlphaMail addresses is performance. Almost all Web mail systems (such as Horde's IMP Web mail Client and SquirrelMail) use IMAP from within the Web server, which is incapable of persisting an IMAP session.

The IMAP protocol is designed to optimize access through persistent access, so this is an inherent and recognized problem. The problem is usually mitigated with an IMAP proxy that maintains a persistent connection. The problems with this solution are multifaceted.

One problem is that the code in the Web mail client itself cannot depend on the state of the IMAP connection and must repeat commands as if each mouse click were a new IMAP session. This is a problem, because the sequence of required events for a new session in the IMAP protocol include authenticating and selecting the desired folder. The benchmarks of several IMAP servers indicate that the repetition of the folder selection command, even if the folder is already authenticated and selected (that is, through a proxy), can cause significant extra server load.

These inefficiencies could be addressed through improvements in the IMAP server and proxy algorithms, but another problem is intractable: a proxy cannot improve the protocol. The fact that the Web mail client is using IMAP forces it to behave as a complete standalone client. If the developers want to add a complex feature, such as conversation views (à la Google mail), which requires complex message cross-referencing across several folders, the protocol itself becomes a major impediment.

AlphaMail solves these problems by including a middleware layer that uses a simplified and extensible protocol for the Web application and is responsible for optimizing access to the mail servers. The protocol supports highly specialized commands that allow the Web code to ask for the information needed for a page directly, without having to make any assumptions about the state of the IMAP connection.

This has the additional advantage that the IMAP protocol handling in the middleware layer can be written in a high-performance language (in this case C++), can cache results and can optimize the interaction. The middleware program is known as the `imap_webcache`, because it caches both data and network connections for the mail interaction.

Accessibility

The next concern was accessibility. Information systems in higher education, government and many other environments must support access for everyone. AlphaMail still is actively being tuned for access by the disabled, but it is already optimized for other access concerns. For example, site security policies might require or recommend that members of the community disable browser features that regularly appear in US-CERT advisories, such as JavaScript. The only Web e-mail I could find that did not require JavaScript for even basic functionality was SquirrelMail, and it was deemed too risky from a performance standpoint.

AlphaMail includes JavaScript enhancements, but those enhancements have traditional CGI alternatives that become active if JavaScript is disabled on the browser. All critical functions of the system will work with any browser, independent of capabilities and settings.

Installation

AlphaMail supports GNU autoconf and has been built cleanly on many Linux variants, FreeBSD, Solaris and Darwin (OS X). The easiest platform to install is Fedora Core 5 via `yum(1)`, as described on the AlphaMail home page.

I highly recommend using Fedora Core 5 on a test machine to avoid the headaches of dependency resolution for your initial trial run. The actual build is pretty much what you'd expect, but solving runtime dependencies can be challenging.

If you decide on a source build, first you need to install the Boost C++ libraries (see Resources).

Installing Boost

You need the Boost Jam utility (usually named `bjam`) as well as version 1.33 or better of the Boost C++ libraries. Jam is sort of a combination between GNU autoconf and `make`. Follow the instructions from the Boost Web site for details, but essentially, extract the files and run:

```
# bjam install
```

In rare circumstances, you may want to pass options (such as an installation prefix). See the Getting Started guide on the Boost Web site if you have special needs.

Building AlphaMail from Source

The build is very much what you would expect:

```
# ./configure
# make
# make install
```

The configure script checks your system for dependencies and tells you what is missing or out of date. You should be able to complete all three steps as long as you have Boost installed, even if Perl dependencies are not met.

Be aware that some versions of g++ have a bug that will cause the compiler to go into an infinite loop while building `imap_webcache`. Two of the files in the source (`IMAPFolder_rules.cc` and `RFC2822.cc`) can trip this bug, but even with a good compiler, these files take a large amount of time and space to build, so expect the compile to run for a few minutes.

On some systems, there can be problems with the location of files. For example, on Red Hat Enterprise Linux systems, Kerberos dependencies cause OpenSSL code to fail to compile, which can be corrected by passing a value for `CXXFLAGS`:

```
# CXXFLAGS="-I/usr/kerberos/include" ./configure
```

As with any other GNU autoconf system, check the `config.log` if the configuration script fails to complete.

Installing Runtime Dependencies

The biggest difficulty on most distributions is not building AlphaMail, but rather meeting all of the prerequisites. Many distributions come with an older version of `mod_perl`, `libapreq` and other Perl modules.

Make sure that you have `libapreq2` and `mod_perl >= 2.0` installed before messing with the other Perl dependencies, because some of them rely on one or both. Once this is done, you should be able to install the remaining Perl dependencies with your package manager or `cpan(1)`.

You need to use `cpan(1)` if the packaged versions of a module do not exist or are too old. See the sidebar on avoiding packaging conflicts with your distribution's package manager.

Configuration

The build instructs you to create template configuration files with the `alhamail_genconfig` utility. This script prompts you for all of the necessary configuration options for a basic installation and creates the necessary files in a location you provide.

You need a special user for a sandbox and knowledge of what user your Web server runs as before starting configuration. I recommend creating a user named `sandbox` for the former. Logins for this user should not be enabled.

The configuration will ask for an installation prefix, which is whatever you passed to `configure`. This is usually `/usr/local`, and the script will verify correctness before continuing.

You will be asked to provide an IMAP prefix and separator for each IMAP server you want to access with AlphaMail. Some IMAP servers use slash (/) for the separator; others use dot (.). The prefix is a subfolder where users put all their other mail folders. For example, if users have shell access and their mail folders are stored in their home directories, it might be policy to put all of them in a directory named `mail`, in which case the prefix is probably `mail`.

Avoiding Package Management Conflicts When Using `cpan(1)`

Many Linux administrators do not like to use `cpan(1)` due to competition between the files installed by the `cpan` utility and the distribution's package manager. In production systems, therefore, it is desirable to install Perl modules in a place where the package manager will not see them.

AlphaMail looks for its own libraries in `/usr/local/lib/alphamail` (if you chose the default prefix during the build), and you can install Perl dependencies there without having to change anything about the runtime configuration of the system.

First, set your environment to indicate you would like Perl to look in an additional location (this is needed only during the build, the runtime system already includes this path during startup):

```
# export PERL5LIB=/usr/local/lib/alphamail
```

If you use an alternate prefix during configure, alter the `/usr/local` part of this to match.

Next, run `cpan(1)` and (re)configure it:

```
# cpan
cpan> o conf makepl_arg
  => 'PREFIX=/tmp/unnneeded LIB=/usr/local/lib/alphamail'
```

The `PREFIX` argument tells the build process what the general installation prefix is; whereas, `LIB` tells it where to install the actual module code. I use `/tmp/unnneeded` for `PREFIX` and remove the files afterward, because AlphaMail needs only the library. Set `PREFIX` to something like `/usr/local` if you need the manual pages or other extras that come with these modules.

If you want to save the `cpan` settings for future sessions, do:

```
cpan> o conf commit
```

Now, install or upgrade the necessary Perl modules (which are listed when you run `configure`):

```
cpan> install Time::HiRes
...
```

If you want to use these modules in your own scripts (or need to change where an AlphaMail script looks for them), add the following line near the top of the file:

```
use lib qw(/usr/local/lib/alphamail);
```

It is important to note that some Web servers (such as Cyrus) use `INBOX` for the prefix and dot (`.`) for the separator. The following procedure can help you determine what to use. First, connect to the IMAP server from the command line, with:

```
# openssl s_client -connect imap.example.com:993 # For SSL
```

or:

```
# telnet imap.example.com 143 # for no SSL
```

These commands connect you to the IMAP server and allow you to enter protocol commands. Type the following (the numbers are part of the commands):

```
1 login username password
2 list "" "%"
3 logout
```

The username and password, of course, should be real user credentials for a typical IMAP account. The responses to the second command should look like this:

```
* LIST (\HasNoChildren) "." "INBOX.Spam"
* LIST (\HasNoChildren) "." "INBOX.Trash"
```

which indicates that `.` is the separator and makes it pretty obvious that `INBOX` is a common prefix (in this case all entries start with `INBOX.`).

The prefix parameter is primarily an interface optimization: the interface removes the prefix when displaying most folder names in order to make things more compact. You can hand-edit any of the parameters in the resulting `alphamail_config` file, which is a commented text file. The entry for defining a pair of typical IMAP servers that serve two mail exchanges looks like this:

```
imap_servers: example.com=imap.example.com:993[INBOX.],
  =>example.net=imap.example.net:143[/]
```

The above setting indicates that users should be able to select their mail domain on login (`example.com` or `example.net`), and associates these with a corresponding IMAP server, port, prefix and IMAP path separator.

The separator in the brackets is always required, but the prefix is not. The notation `[/]` means no prefix, with slash as the separator. The IMAP connections will be insecure if you use anything but the SSL alternate port 993.

Attachment viewers and other external programs run in a sandbox that uses a `chroot` jail, user ID protections and other filesystem restrictions to ensure that a bug in a viewer cannot compromise anything more than the file the user is trying to view, which by definition would be the file containing the exploit. This is where you will use the extra user you created earlier.

The sandbox utility is installed in `/usr/local/libexec/sandbox`, by default, and is a `setuid` program. It is important that the permissions of this executable allow execution by the Web server, but it is a security hazard to allow any other user access to the utility. I recommend that

AlphaMail be run on a standalone system that serves only Web mail and nothing else, with no shell access for users.

The configuration also asks you to configure the large file-sharing system. This option allows users to upload files to the AlphaMail system, so that others can download them later. Large file sharing is useful when someone needs to send a file that is larger than is allowed or recommended as part of an e-mail message. File sharing has several safeguards to prevent abuse, including terms-of-use agreements, size limits, password protection, encryption, download limits and time-based expirations. Choosing a zero size for the size limit in file sharing disables the feature.

The final step is to edit the Apache configuration. Make sure that `mod_perl2` and `libapreq2` are loaded with directives, such as:

```
LoadModule apreq_module modules/mod_apreq2.so
LoadModule perl_module modules/mod_perl.so
```

And, include the generated `alphamail.conf` Apache configuration file. For example:

```
Include /usr/local/etc/alphamail/apache/alphamail.conf
```

Running AlphaMail

Apache and `imap_webcache` must be running for AlphaMail to work. Startup order does not matter. A sample Red Hat init script for the Web cache is included and will be installed in `/usr/local/share/alphamail/util/init.d`.

A garbage collection script must be run periodically from cron. AlphaMail writes numerous files as the mail system operates, most of which are decoded MIME messages and attachments. These files cannot be cleaned reliably by the Web software, as there are no guarantees about user behavior. The script is called `garbage_sweeper` and is well documented in the Administration Guide.

AlphaMail is in production use at the University of Oregon. The performance and usability results have been very encouraging, and the former are available at the AlphaMail home page.

However, the system is still new, and there are some latent bugs that have yet to be solved. The `imap_webcache` itself is a rather complicated piece of software that may have occasional problems. As a result, I recommend running an included utility called the `hang_detector` (in `/usr/local/share/alphamail/util` by default). You must edit this script before using it, and it requires a valid IMAP user in order to work.

It runs a full query against the Web cache every 15 seconds and is capable of restarting the `imap_webcache` (via the included init script). It is also capable of sending mail to administrators if desired.

Common Problems

There are several places where you can run into difficulty on a new installation. The Administrator's Guide included with AlphaMail has many tips on how to solve these issues.

One of the most puzzling problems is produced by accessing the service via an incorrect domain name. The cookie that maintains the session is tied to the server domain that you specify during configuration and is stored in a parameter in the AlphaMail apache configuration:

```
PerlSetVar alphamailDomain server.example.com
```

If the user is able to access the login page with an unqualified hostname (such as `server`), the cookie will not be exchanged properly and login will fail without any error whatsoever. The login is actually succeeding, but the browser is not sending the cookie back because of the mismatched URL.

Fortunately, most users connect though a link or type an insecure HTTP URL. The former is never a problem, and a redirect does a nice job of correcting the latter:

```
<VirtualHost _default_:80>
  RedirectMatch ^/alphamail/?$ https://server.example.com/
  ↳ alphamail/index.html
  RedirectMatch ^/alphamail/mail/?$ https://server.example.com/
  ↳ alphamail/mail/index.html
</VirtualHost>
```

Other problems usually involve dependencies, configuration errors, incorrect permissions or missing auxiliary directories. The best indication of these are errors in `alphamail_ui.log`, which by default is created in `/var/log`.

SELinux also can be the source of problems. I have not run AlphaMail with SELinux, and unless you are willing to create your own security profile, you probably will need to turn it off or disable enforcing.

The Future of AlphaMail

AlphaMail development has been mostly concerned with performance and core functionality to this point, and I don't expect these two issues to become any less important in the future, as the product is directed at environments that have a large and diverse user community.

It is quite feature-rich, but certainly not all-inclusive. Limiting feature creep is an ongoing requirement, because the critical concern is easy and reliable access to essential functionality, not global coverage of mail client features.

Still, there are plans for significant improvements and additions, such as an internationalized interface, optional Ajax components that would improve client interaction and conversation views. ■

Tony Kay is the primary developer of AlphaMail and works for the University of Oregon in Eugene, Oregon. He can be reached at tkay@uoregon.edu or tony.kay@gmail.com.

Resources

AlphaMail Administration Guide: sourceforge.net/docman/display_doc.php?docid=38047&group_id=183361

AlphaMail Home Page: sourceforge.net/projects/alphamail

Boost C++ Libraries: www.boost.org

IMAP Performance Benchmarks: whizzo.uoregon.edu/public/src/mailperf/results.html

US CERT: www.us-cert.gov

The Benevolent Racketeer

We present a fair and balanced picture of the other side of the Microsoft covenant not to sue.



Nick Petreley, Editor in Chief

We at *Linux Journal* have taken a dim view of the Microsoft/Novell deal and Microsoft's attempts to monetize the work of the Open Source community by offering covenants not to sue. However, in an effort to remain fair and balanced in our coverage of these controversial issues, we are publishing the following testimonial from a happy Microsoft customer who wishes to remain anonymous because he doesn't exist.

Dear *Linux Journal* Editor,

I must take issue with the negative stance *Linux Journal* has adopted with respect to Microsoft software patents, licensing and covenants not to sue. Our company represents one of the customers to whom Microsoft Senior Vice President Bob Muglia refers when he says, "Customers have indicated to us that it's problematic to them that when they work with open-source software, they don't have similar intellectual property protection."

I find this to be so true. Our company simply could not adopt open-source software without intellectual property protection. Open source represented a grave liability to us because no company claimed control over it. Put simply, how could we have confidence

that someone would not sue us for using Linux unless we paid someone not to sue?

We were anxious to pay protection money to solve this problem, but to whom would we pay it? We would gladly have paid Red Hat, but Red Hat did not threaten to sue us, so what would be the point? That's the problem with Linux. It is not owned by a single entity with the power to license protection, and those who distribute it are remiss in threatening to sue in order to make intellectual property protection available to paying customers.

We feel Linux advocates and developers should be grateful that Microsoft stepped up to the plate in this regard. Now that Microsoft has graciously offered to solve this dilemma for us by threatening to sue us for using Linux, we have finally found a reason to adopt Linux and open source. We now know where to send our protection money.

Technically speaking, SCO was the first to offer us a covenant not to sue, and we were almost ready to adopt Linux in a big way when SCO provided us this option. We backed away when IBM and Novell tried to make it difficult for SCO to follow through. Why? IBM is much bigger than SCO. If IBM created enough trouble to put SCO out of business, it would set us back to square one, without anyone to pay protection money. This was a risk we were not willing to accept.

Then Microsoft approached us behind the scenes and offered us a sweet deal. We agreed to pay Microsoft \$20 for every copy of Linux we would install, in return for which Microsoft promised not to sue us for \$400 per copy throughout the terms of the agreement. This deal not only provided us with the intellectual property protection we customers demand, as noted by Bob Muglia, it also saves us \$380 per copy of Linux until the agreement expires! Think of it. Even if we deploy only 1,000 copies of Linux, that's a \$380,000 savings. With an economic incentive like that,

how can anyone question Microsoft's commitment to the open-source model?

This is not the first time Microsoft volunteered to assist us with intellectual property protection. Some years ago, Microsoft conducted an audit of our Windows and Office licenses, which brought to our attention the fact that we had not organized and stored our product licenses properly. Microsoft solved our problem easily by selling us all the replacement licenses we needed. It may seem unreasonable to some people to pay twice for the same software, but we beg to differ. We think of it as paying not for software, but for peace of mind, which is what intellectual property protection is all about.

Some like-minded customers advised us to adopt SUSE Linux, as it is covered under the Microsoft/Novell covenant not to sue. We consider this an acceptable option, but not the ideal one. It is much more effective to deal directly with the threatening party.

Think of it this way. Our regular fire insurance covers accidental fires. We've had two accidental warehouse fires (minor ones, thankfully) since we bought the policy. In what way did our insurance protect us against accidental fires? It didn't help at all. In contrast, we donate money in a manila envelope each month to another "insurance" company who has sworn to protect us against arson. Since we started making those payments, we've seen a 100% reduction in fires caused by arson. You can't argue with that kind of success, and it's all because we deal directly with the source of arson fires.

I hope this testimony has opened your eyes at *Linux Journal*. It's not too late for you to secure your own intellectual property protection. ■

Nicholas Petreley is Editor in Chief of *Linux Journal* and a former programmer, teacher, analyst and consultant who has been working with and writing about Linux for more than ten years.



Russ Barnard, President, FlapDaddy Productions

“Fanatical Support™ saved me from my own mistake.”

“Not long ago, I reformatted one of our servers. Not until I was driving home did I learn that I brought our entire site down in the process. I called my guy at Rackspace and he said, ‘We’re already on it.’ By the time I pulled in the driveway, my site was back up. Now that’s Fanatical Support.”

Keeping little mistakes from causing big problems is one definition of Fanatical Support. What will yours be?

Watch Russ’s story at www.rackspace.com/fanatical
1-888-571-8976



Hear Yourself Think Again!



WhisperStation™ **Cool... Fast... Silent!**

For 64-bit HPC, Gaming and Graphic Design Applications

Originally designed for a group of power hungry, demanding engineers in the automotive industry, WhisperStation™ incorporates two dual core AMD Opteron™ or Intel® EM64T™ processors, ultra-quiet fans and power supplies, plus internal sound-proofing that produce a powerful, but silent, computational platform. The WhisperStation™ comes standard with 2 GB high speed memory, an NVIDIA e-GeForce or Quadro PCI Express graphics adapter, and 20" LCD display. It can be configured to your exact hardware specification with any Linux distribution. RAID is also available. WhisperStation™ will also make a system administrator very happy, when used as a master node for a Microway cluster! Visit www.microway.com for more technical information.

Experience the "Sound of Silence".

Call our technical sales team at 508-746-7341 and design your personalized WhisperStation™ today.



Microway
Technology you can count on™