AN SSC PUBLICATION

LinuxBIOS: CUSTOM FAST BOOT FOR YOUR ROCKET CONTROLLER (OR PC)

LINUX JOURNAL

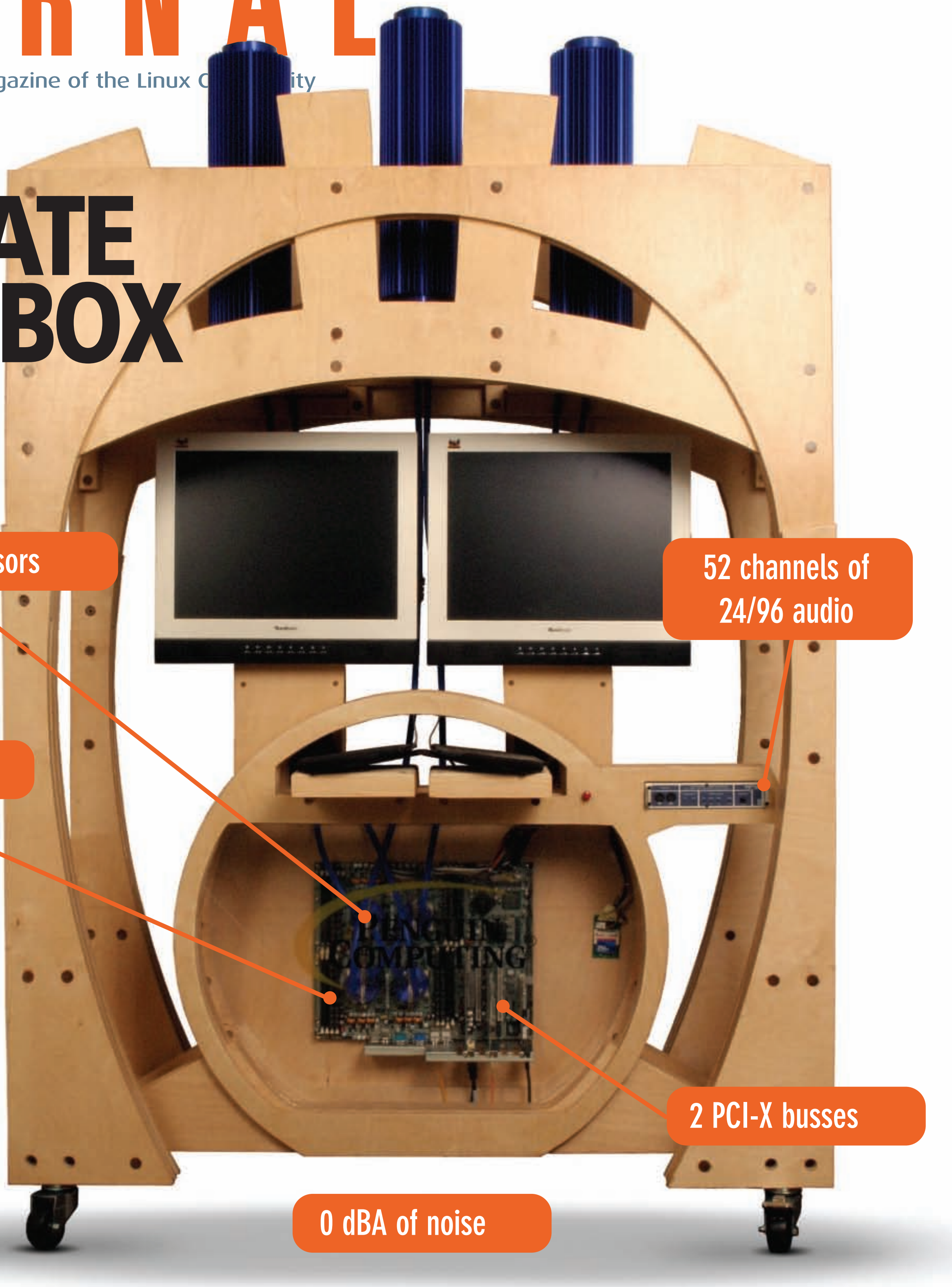
Since 1994: The Original Magazine of the Linux Community
AUGUST 2005

THE VOTES ARE IN!

LINUX JOURNAL EDITORS' CHOICE 2005

2005 Editors' Choice Awards

ULTIMATE LINUX BOX 2005



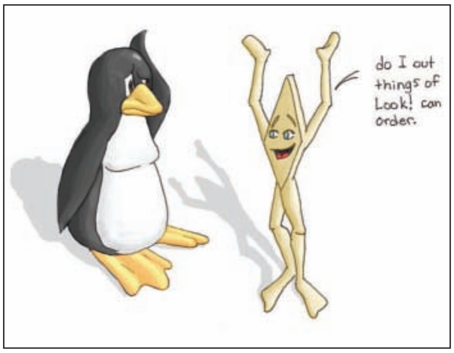
USA $5.00 CAN $6.50
www.linuxjournal.com
08
0 71486 03102 4

LINUX JOURNAL
ULTIMATE LINUX BOX · Editors' Choice Awards · ALSA · LinuxBIOS · Ubuntu
AUGUST 2005 ISSUE 136

LINUX JOURNAL

AUGUST 2005 ISSUE 136

COVER PHOTO: DON CAMERON



While it's waiting for data from main memory, your CPU rushes ahead and does other things—which could make the OS very, very confused. Paul McKenney explains how Linux keeps up on page 52.

NEXT MONTH

WIRELESS

Two-thirds of 802.11b networks don't use any encryption at all. In 2001, the commonly-used Wired Equivalent Privacy (WEP) security was shown to have a critical flaw. Today, we have Wi-Fi Protected Access (WPA)—is its security any better? John L. Macmichael does a survey of wireless security protocols and recommendations for your network.

We couldn't decide whether to do a Beowulf cluster article or a Linux in space article, so Ian McLoughlin, Timo Bretschneider and Bharath Ramesh helped us out with a detailed report on the first Beowulf cluster in space. Maximize your scarce downlink bandwidth by doing as much of the image processing as you can in space.

When you compress data, you're trading CPU time for bandwidth savings. Which compression tools, and which settings, give you the best trade? Kingsley G. Morse Jr. put a host of tools through a thorough test and has some advice for every connection capacity.

Did You Hear That?

If you put high-end digital audio on a Linux box, it had better run quiet. This issue looks at hardware, audio manipulation, security and more.

BY DON MARTI

I know you're more interested in the parts list for the Ultimate Linux Box (page 44) than this column, so go ahead and check it out. I'll wait. Like it? And now that we can run a four-way SMP, 64-bit monster in silence, shouldn't noisy fans be a thing of the past for ordinary systems too? My Commodore 64 didn't need a fan. The original Apple Macintosh didn't need a fan. But almost all of today's supposedly more-advanced systems come with them. And who let the power supply fan and the CPU fan mate and spawn the north bridge fan and the video card fan? Enough.

Hard drives do make noise, but we have ways to deal with the impact. Laptop mode, which Bart Samwel covered last year, lets you keep your hard drive spun down most of the time. And NFS, ATA over Ethernet and other technologies mean that you can move noisy disks to the other end of a long wire from your long-suffering ears.

Pretty much all desktop systems have more processing power than we can use, so now it's time to think about our quality of life. We have to thank the lm_sensors development team for making it possible to measure temperature safely—a must for fan-free cooling experiments.

Now that I think about it, my Commodore 64 booted pretty quickly too. Ron Minnich has part of the answer: replace the legacy proprietary BIOS with the fast-booting LinuxBIOS. Every motherboard is a little different, though, so getting LinuxBIOS working on a new one is a challenge. Get started on page 32, and check in with our Web site for details and further steps.

Since the theme of this issue is faster hardware, we made a point of bringing back kernel developer Paul McKenney, of RCU fame, to fill you in on what the CPU is doing behind your back. Your instructions in a different order putting. Make the CPU do a sanity check on page 52.

Now that you have your Linux system running as quietly as possible, Dave Phillips has some gnarly details on ALSA sound. Learn how you can do mixing, MIDI and more, whether you have the pro sound hardware from this year's Ultimate Linux Box or an ordinary PC sound card (page 58).

Toshiyuki Maeda is back with more on running ordinary applications as part of the kernel. This time he's using the x86-64 architecture and running a real application, MySQL, in kernel space to look for possible performance improvements (page 20).

We got our columnists and contributing editors together for Editors' Choice Awards 2005 (page 82). Disagree? Check out the Readers' Choice voting now happening on the Web site (www.linuxjournal.com/article/8266).

Our Paranoid Penguin column is going through a change. Mick Bauer, star of DEFCON, Linux Lunacy cruise talks and two editions of an O'Reilly book, is taking some time for other projects, and Paranoid Penguin will be written by various contributors, as Kernel Korner is now. Mick, thanks for all the good advice over the years and for the look to the future on page 28.

Don Marti is editor in chief of *Linux Journal*.

LINUX JOURNAL

AUGUST 2005
ISSUE 136

EDITOR IN CHIEF Don Marti, ljeditor@ssc.com
EXECUTIVE EDITOR Jill Franklin, jill@ssc.com
SENIOR EDITOR Doc Searls, doc@ssc.com
SENIOR EDITOR Heather Mead, heather@ssc.com
ART DIRECTOR Garrick Antikajian, garrick@ssc.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
PROOFREADER Geri Gale

VP OF SALES AND MARKETING Carlie Fairchild, carlie@ssc.com
MARKETING MANAGER Rebecca Cassity, rebecca@ssc.com
INTERNATIONAL MARKET ANALYST James Gray, jgray@ssc.com
REGIONAL ADVERTISING SALES
NORTHERN USA: Joseph Krack, +1 866-423-7722 (toll-free)
EASTERN USA: Martin Seto, +1 905-947-8846
SOUTHERN USA: Laura Whiteman, +1 206-782-7733 x119
ADVERTISING INQUIRIES ads@ssc.com

PUBLISHER Phil Hughes, phil@ssc.com
ACCOUNTANT Candy Beauchamp, acct@ssc.com

***LINUX JOURNAL* IS PUBLISHED BY, AND IS A REGISTERED TRADE NAME OF, SSC PUBLISHING, LTD.**
PO Box 55549, Seattle, WA 98155-0549 USA • linux@ssc.com

SUBSCRIPTIONS
E-MAIL: subs@ssc.com • URL: www.linuxjournal.com
PHONE: +1 206-297-7514 • FAX: +1 206-297-7515
TOLL-FREE: 1-888-66-LINUX • MAIL: PO Box 55549, Seattle, WA 98155-0549 USA • Please allow 4–6 weeks for processing address changes and orders • PRINTED IN USA

USPS *LINUX JOURNAL* (ISSN 1075-3583) is published monthly by SSC Publishing, Ltd., 2825 NW Market Street #208, Seattle, WA 98107. Periodicals postage paid at Seattle, Washington and at additional mailing offices. Cover price is $5 US. Subscription rate is $25/year in the United States, $32 in Canada and Mexico, $62 elsewhere. POSTMASTER: Please send address changes to *Linux Journal*, PO Box 55549, Seattle, WA 98155-0549. Subscriptions start with the next issue. Back issues, if available, may be ordered from the Linux Journal Store: store.linuxjournal.com.

Use the Archives, Luke

Just wanted to say thanks for an article by Michael Johnson from July of 1996 (www.linuxjournal.com/article/1237). This was an article about diff and patch. Although I've been a UNIX/Linux admin for ten years, I have never patched source and needed a quick lesson. This did the trick. Thanks again!

--
Scott Martin

Or We Could Have Put an InfiniBand Card on the Cover

My wife Regine is a belly dancer with the Shimmering Sands here in Alice Springs, Australia, so when I pulled my latest copy of *Linux Journal* out of the mailbox, my first thought was "Our worlds collide!" As the semi-official photographer for the Shimmering Sands and a longtime Linux enthusiast, I had already experimented with using The GIMP, gthumb, Nvu and other open-source tools to manipulate some of the hundreds of belly dance photos that I've taken.

But, I never really thought that much about the many ways in which my Linux hobby and my wife's belly dance hobby might potentially overlap. Thank you for opening my eyes. I'm now looking into open-source music packages such as the Hydrogen virtual drum machine, which seems like it may be quite useful for producing some wicked belly dance practice beats. Now if I can just find out what happened to the apparently defunct TablaBeat Project....

--
Brian Haynes

You might want to see if the tools in "Music Education with Linux Sound Tools" (www.linuxjournal.com/article/7606) can help for dance practice too.—Ed.

Mmmm, Cake

Planning a big wedding is HARD. Two things I planned were having Larry Ewing's Tux on my cake (Linux is part of my life and my job) and a helicopter to take my bride and me from the ceremony to the reception. Kelly, my bride, was all for having Tux—she made the switch to Linux before I did.

--
Chris Turner

WLAN Configuration Question

I have followed the last three issues of *Linux Journal* and the article "Securing Your WLAN with WPA and FreeRADIUS". I think that the article was very good and helpful, and now I am trying to implement that same solution.

I have one question: in Part III where you are configuring eap.conf you have:

```
private_key_file = \
${raddbdir}/certs/bt_keycert.pem
certificate_file = \
${raddbdir}/certs/bt_keycert.pem
```

The names don't match any of the previously created certificates. Which certificates/private key are those? The ca's, the server's or the client's?

--
Tulio

***Mick Bauer replies:** Oops, Listing 3 is incorrect! Those lines should instead read:*

```
private_key_file = \
${raddbdir}/certs/server_keycert.pem
certificate_file = \
${raddbdir}/certs/server_keycert.pem
```

That is, these lines both specify the path to the server's key/certificate file. Sorry for the confusion!

Why Call It DRM?

Digital Rights Management? Isn't that a five-dollar euphemism for "Copy-Protection", which is a perfectly good, accurate description of the practice? Indeed the DRM term arose because "copy-protection" had become such a negative brand that no one would *dare*

Photo of the Month: Ride in the Himalayas

While I was working in Bangalore, India, I started a Royal Enfield Bullet Owner's group (bullet-bangalore.org) and a few of our guys rode to the Himalayas on their bikes. They saw this interesting banner on the only tea stall at Himank, the world's highest motorable road, put up by another group of bikers before them. Take a look. The picture was taken by Sandeep Menon.

--
VaibhaV Sharma

Photo of the Month gets you a one-year extension to your subscription. Photos to ljeditor@ssc.com.—Ed.

offer a product afflicted with it. So why are we cooperating with these people and using their phony sanitized term?

--
j.g. owen

DRM went beyond just controlling copying, and sometimes even keeps you from using functions such as fast-forward. And having a vendor "manage" our digital rights just sounds wrong.—Ed.

Another Satisfied Reader

My daughter, Sofia Buentello, is pretty happy learning all about Linux running on high-end hardware.

--
Gilberto Buentello Ontiveros

/var/spool/fanmail Postal Department

Linux is my one friend here in jail, and this one friend communicates with me through *Linux Journal*. Thank you for this superb magazine.

--
Steve Zimmerman

Good Intro Book?

I am a longtime Microsoft user and am interested in opening some Web pages and Web-based businesses. I would appreciate any recommendations for books or other literature that you could give me for someone just starting out in Linux with the idea of running a server.

--
Richard Tewell

Readers, let's help Richard out here. To vote for your favorite intro to Linux book, visit our Readers' Choice Awards page at www.linuxjournal.com/article/8272. *We'll cover the winning titles in our November 2005 issue. For getting started running Linux servers, we like* Linux Network Administrator's Guide, *Second Edition, by Olaf Kirch and Terry Dawson, and our own Mick Bauer's* Linux Server Security, *Second Edition.—Ed.*

POSTAL MAILING LIST OFFER

Want to get in touch with other *Linux Journal* readers by postal mail? As an experiment, we're putting together a postal mailing list this fall.

Please send your name, mailing address and a brief description of your Linux interests (20 words or less) to:

Linux Journal
Attn: Postal Mailing List
PO Box 55549
Seattle, WA, 98155

We'll include all addresses received before Oct. 1, 2005. Include a self-addressed stamped envelope to receive a copy of the list.

Samba Stays Up, Training Program Doesn't

To the faithful subscriber with no computer access—you're not alone! I too am incarcerated—Florida D.O.C.

Your magazine is my only source of computer information. We used to have a computer program, but it was removed along with several other vocational programs in the name of cost savings. There is now no more skill training in most of the state's prisons.

In the two years prior to the removal of our program, we were able to replace three servers with an all-Linux back end using Samba, Postfix and NFS/NIS. With only a few minor adjustments to the clients, the students experienced only about 40 minutes of downtime. Not bad for a class of 65. In fact, the conversion of Windows servers to Linux was unnoticed by any of the students.

Our project allowed us to purchase new hardware for the students that otherwise would have been used to upgrade our servers. Using Linux saved us from the first round of budget cuts as we saved over $5,000. But sadly the great state of Florida feels it's better to offer no eduction to people. Still, for a while we were able to enjoy "freedom".

Keep your informative articles coming. And to all the pending new Linux users—our time will come—mine will be June 2006, and I have the goal of starting a local computer service that specializes in using open-source software.

--
Benjamin Davis

Labels over Covertext

Every month the subscription label covers some text describing the contents. And, the label is sometimes horizontal and sometimes vertical.

If you move the barcode and price to the lower left corner of the cover and leave mailing label (plus safety factor) room around it, the problem will be resolved. Subscribers don't need the barcode/price, and retail sellers don't care on which corner the bar code and price are located.

--
Rich

Garrick Antikajian responds: *we try to place text and the other cover elements in a way that achieves the best possible cover design. The mailing label placement is decided by our printer. These factors sometimes result in text being partially obscured. We appreciate your input on this matter and apologize for any inconvenience this may cause.*

New Installer for iCalendar

In the article "Dynamically Generated Calendars" in the June 2005 issue, Mr Lerner states that the package does not install automatically. I've been using this package myself for a commercial product I'm developing and found that the Web site he referenced was an

LETTERS CONTINUED ON PAGE 94

On the WEB

Cast Your Final Votes in the 2005 Readers' Choice Awards

As you all know by now, we've made some changes to how we're running the Readers' Choice Awards this year so that our readers are more involved every step of the way. By the time you read this, the final ballot—determined by your write-in nominations and first-round voting results— will be available on the *LJ* Web site (www.linuxjournal.com/article/8272). Final votes will be accepted during the entire month of July, and the winners will be announced in the November 2005 issue of *Linux Journal*. As the name says, these are the Readers' Choice Awards, so get on over to the Web site, check out the final ballot and send us your votes!

For complete information, details and dates regarding the 2005 Readers' Choice Awards, read "New Procedures for 2005 Readers' Choice Awards" (www.linuxjournal.com/article/8192).

As you'll see in this month's article "Porting LinuxBIOS to the AMD SC520", the port itself is an ongoing job. Author Ronald G. Minnich says, "The Porting article in issue 136 doesn't tell the whole story! Join us on the Web as we finish the port to the AMD SC520, dodging hardware glitches and software bugs as we go." To follow along as the Cluster Research Team at Los Alamos National Laboratory continues its port project, head on over to the *LJ* Web site and read "Porting LinuxBIOS to the AMD SC520: A Follow-up Report" (www.linuxjournal.com/article/8310).

Whether you've built your Ultimate Linux Box, or if you're trying to get maximum speed out of a lesser machine, it's time to set it up for optimum speed for desktop applications. In a series of articles on "Optimizing Desktop Performance", Tom Adelstein covers desktop performance tweaks from simple tools such as hdparm all the way up to a script that will make OpenOffice.org stay in memory and start more quickly when you open a document. Follow along with the comments and get some ideas there too (www.linuxjournal.com/article/8308 and www.linuxjournal.com/article/8317).

diff -u

What's New in Kernel Development

Linux kernel development was thrown into chaos recently, when **Larry McVoy** finally decided to pull the free-of-charge **BitKeeper** license as he has threatened many times to do. But within days of the event, **Linus Torvalds** and a horde of contributors had written an acceptable alternative, entirely from scratch. The git filesystem is Linus' brainchild, a low-level, extremely fast content tracker that appears to be almost completely alien to existing version-control ideas. Virtually opaque, it is intended to exist beneath a layer of scripts that make use of its various services. Anyone can script a new git user layer on top of the basic system. In fact, **Petr Baudis** has been working with tons of folks on **Cogito**, a git front end that looks to be Linus' choice for ongoing kernel development. Many Web interfaces and other auxiliary tools also are springing into existence at a rapid rate.

H. Peter Anvin has been keeping **kernel.org** up to date with all the latest git repositories, arranging for hosting services and generally tending house. Recently, however, kernel.org began yet again to bog down with the tremendous bandwidth demands from all over the world. This time, **Hewlett-Packard** was the one to charge to the rescue, donating two powerful computers. kernel.org will now operate as a DNS round-robin between both. This has reduced site latency significantly, and it drastically sped up upload time for site contributors as well. At last report, nary a glitch remained, although the round-robin does make it more difficult to derive network traffic statistics.

Joel Becker has created **ConfigFS**, another interface into kernel internals. The goal this time is to create something scriptable and completely readable. But with SysFS already in existence and performing a similar function, it's unclear whether ConfigFS will represent a true advance or just another addition to the mess. All of these filesystem-based interfaces have been born out of a desire to recover somehow from the ProcFS, /dev, ioctl nightmares Linux inherited from its great progenitors. But if these new alternatives themselves are not sufficient, SysFS, udev and now ConfigFS become only more legacy cruft to be hated by kernel developers for years to come.

The **FUSE** (Filesystem in USErspace) developers either are cleaning house or making a new mess. **Miklos Szeredi** posted some patches to make the user interface compatible between 32-bit and 64-bit modes of operation, on systems that supported both modes. Among its various benefits, the patch breaks the backward compatibility of the user interface. With FUSE already in **Andrew Morton**'s -mm kernel tree, this patch may be one painful yet required step along the bridge into the official kernel; or it may be a descent into breakage, on the way out of Andrew's tree altogether. Time will tell. At the last previous report, the FUSE developers were making good progress toward answering some of Linus' harsher objections, and he was no longer so completely dead set against even the idea of a user-space filesystem.

After a public discussion between the two groups of developers on the **open-iscsi** and **linux-iscsi** projects, they have decided to merge into a single project. For technical reasons, both groups have agreed to start from the open-iscsi codebase, because of that project's optimized input/output paths and the well tested iscsi-sfnet components for the control plane and user-space components. The open-iscsi Subversion repository will continue to be used, at least for now. This unification of two projects working toward the same goal is excellent. Hopefully most of the linux-iscsi group will remain and continue to contribute; and their previous accomplishments will continue to be recognized, in spite of the migration to a new codebase.

Randy Dunlap is leading the charge to reorganize the kernel's networking configuration options. This has been an ugly task to consider, because in many cases it is not at all clear how best to organize the hierarchy. Is something a driver, or is it a protocol? Should all drivers be grouped together, or is it all right to group some drivers with a specific related subsystem? Randy bit the bullet and made a first pass at answering some of these questions and was quickly joined by several other folks. With much wrangling, soul-searching and a little guesswork, an entirely new landscape of network configuration seems to be forming gradually. We should expect to see portions of this landscape with periodic minor earthquakes in upcoming 2.6 releases.

—ZACK BROWN

LinuxFest 2005

LinuxFest Northwest, the largest users group conference in the Pacific Northwest, was held again in Bellingham, Washington, 20 miles south of the Canadian border. Among the presenters was Google's Chris DiBona, shown here updating the audience on code.google.com, the portal site for the company's open-source projects.

LinuxFest Northwest was put on by the Bellingham Linux Users Group and six other groups, and hosted by Bellingham Technical College (BTC). As in previous years, LinuxFest was free of charge and open to all; an estimated 1,000 people attended.

More than 40 presentations covered topics from general interest to advanced systems administration. Presenters included people from IBM, Novell, RealNetworks, the Linux Professional Institute, the X.org Foundation and the Ubuntu Project.

The BTC Chefs Club served a grilled salmon lunch and espresso drinks.

The exhibits room included Google recruiters, some users groups, some free software projects, such as Ubuntu Linux and MySQL, and even the Seattle BSD users group.

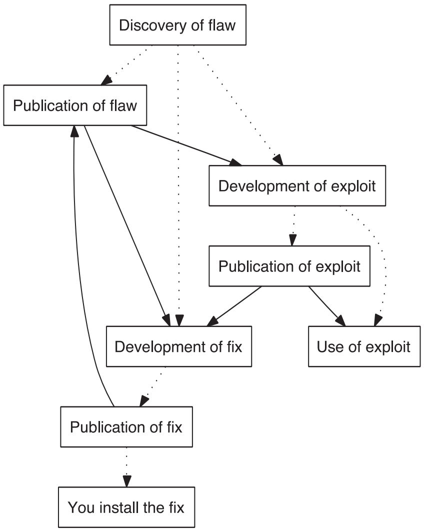
For the second year, Chuck Wolber hosted the "Alpha Geek" trivia contest, which was a great deal of fun. The day finished with the annual fund-raising raffle. Several thousand dollars' worth of donated prizes included

Graphviz

www.graphviz.org

I don't know about the rest of you, but I think best when I'm in front of a whiteboard drawing boxes and arrows. However, when it's time to put the idea up on a Web site, it's either upload huge photos of the whiteboard or spend hours dragging little boxes and arrows around in some GUI application. Graphviz to the rescue. I made this diagram in 15 lines of easy markup (yes, -> makes a line with an arrow) and converted it with one command. Multicolored boxes and lines take just a little more time with the on-line docs.

Graphviz really shines when it's time to generate big graphs from your own software. No matter how complicated a structure your program spits out, Graphviz turns it into a readable layout. See the Web site for examples.



INSIDE TALK

The best cases for Linux and open source in business often come from the resourceful people who put it to use there. What they provide are patches of wisdom that add to everyone's understanding. Eventually, resistance becomes futile because the advantages are understood too well. Here are three such "patches" from comments to just one *Linux Journal* on-line article:

> With GNU/BSD licensed software at least, the receiver of the codebase is left in complete control. Even Microsoft could grab a copy of the code and configure/support it the same as with the Windows codebase. There's no corporate competition from OSS/Free software, just service companies that sell packages including it.—Chris Bergeron of pcburn.com.

> In my experience, the main reason for buying into open-source projects goes far beyond "wanting something the market doesn't offer". It's more about servicing business needs quickly. Who can wait for a vendor to respond to a request, when it's so much easier to take pre-existing OSS systems or code and improve them slightly to solve your particular problem? With ever-decreasing time frames, OSS makes the impossible possible.—Dave Moskovitz of www.thinktank.co.nz/dave.

> For an IT professional, it often takes less time to install and configure an open-source package than to get approval to "buy" (actually, enter into a license for) a proprietary one. Transaction costs aren't just between vendor organization and customer organization—they're within organizations.—Anonymous

Source: "Getting Flat, Part I": www.linuxjournal.com/article/8251.

—DOC SEARLS


Opteron & Xeon EM64T
Blade Servers
- Lower Your Data Center Costs -
www servers
PRODUCT EXCELLENCE AWARDS
LINUXWORLD CONFERENCE & EXPO
2003 Best System Integration Solution
3U Blade Server
3U- 14 Xeon Blade Server - Highest Density Available
Low-Power Consumption
Choose from a broad line of Tatung rackmount servers.
3U and 4U Blade Servers
Superior Tatung architecture integrates up to 14 Xeon® processor server blades in each 3U chassis. The 3U blades can be configured in up to 196 servers per 42U rack. The 4U can hold up to 10 dual processor blades per chassis and 100 PCI-X servers can be configured with your choice of up to 200- 64 bit Opteron™ processors, 32bit Xeon processors or Intel's new Xeon EM64T, 64bit processors per 42U rack. Tatung blades use significantly less power than comparable systems while eliminating more than 80% of the wires.
Each hot-swappable blade is powered by the latest low-power consumption processor and has built-in dual gigabit ethernet switches and can be controlled with Tatung's advanced management software.
1U and 2U Xeon and Opteron Servers
High Performance single or dual Intel Xeon EM64T or AMD 64-bit Opteron processor rackmount servers are now shipping. Configure the 1U with up to three hard drives and a 2U with up to nine hard drives.
Tatung is building rackmount servers now. Call today and have us build a system for you.
4U-10 Dual Opteron or
Xeon EM64T PCI-X Blade Servers
Point to
tsti.com
1U Dual Opteron/Xeon EM64T
Rackmount, IPMI 1.5 Ready
The Intelligent Choice in Server Solutions
TATUNG®
SCIENCE & TECHNOLOGY, INC.
USA: 1-800-659-5902 • www.tsti.com
Europe: www.tatung.nl
Asia: www.tatungserver.com
Opteron is a trademark of Advanced Micro devices
Xeon is a registered trademark of Intel Corp.
Tatung is a registered trademark of Tatung Company.

Sharing Calendars

The last piece in the shared calendar project is letting users push their calendars up to the server. Here are two ways to do it. BY REUVEN M. LERNER

Over the last few months, we have explored the iCalendar standard and the ways in which it allows us to create our own calendars, as well as work with remote ones.

But if you think about it for a moment, you'll realize we are missing a key piece of functionality. We have seen how easy it is to create our own local calendars. We have seen how we can retrieve remote calendars. We have even seen how we can create and distribute remote calendars, generating events dynamically from a Web/database application. But we have never considered how an individual Sunbird user might be able to share his or her calendar with other people.

Anyone who has worked in even a medium-sized organization knows that scheduling appointments can be difficult. Having access to everyone's calendar, and being able to schedule meetings for them, is an increasingly useful feature for our software to have. If every change I make to my calendar is available for everyone to see, it will be easier for them to schedule meetings when I will be around. (Or when I won't be around, if they want to keep something secret from me.) I used to ask clients why they use Microsoft Exchange as a mail server, given the availability of excellent open-source alternatives; inevitably, the answer would have more to do with the calendar support in Outlook and Exchange, rather than the e-mail functionality.

This month, we close our exploration of Sunbird and iCalendar with a look at how we can publish calendars to a central repository for others to share. The results might not be as slick or smooth as some of the commercial alternatives, but as with many other types of software in the open-source world, I believe that this is rapidly changing, and that we soon will see open-source calendar servers that are equal or superior to their proprietary counterparts.

Sharing

Before we try to share a calendar, we should define exactly what we mean by sharing. You might think that shared calendars are stored in a single place and accessed by multiple calendar programs simultaneously. Although it is theoretically possible to configure Sunbird, or any other iCalendar-compatible program, such as Evolution, in this way, this is not what we would typically expect.

Basically, a shared calendar in the iCalendar world is an iCalendar file that is available for retrieval from a publicly accessible server. That iCalendar file might be updated once per hour or once per year; much like an RSS feed or a Weblog, there is no way to know how often a particular calendar file might be updated. For this reason, we need to make several assumptions: 1) everyone who is interested in this particular calendar is subscribed to it; 2) every subscriber downloads an updated version of the calendar on a regular basis, at least once per day; and 3) the calendar's manager publishes all changes and updates to the public server as soon as they are made.

In other words, the sharing does not take place in real time at all, but rather depends on all of the participating users to publish and retrieve updates on a regular basis. Between updates, a calendar user sees only the most recently downloaded iCalendar file, which is stored on his or her local computer. If a calendar subscriber is scheduled to retrieve updates only once per day, it is quite possible that he or she will miss last-minute updates to the calendar. Just how often someone should subscribe to calendar updates depends on the nature of the organization, how important it is to get updates and the load that might be placed on the server. After all, a server that can provide daily updates to 100 people might have trouble providing hourly updates to 10,000 people.

Storing with FTP

The easiest way to publish files on the Web is to use the old standby for file transfer, FTP. FTP has gone almost unused on my server for some time now, in no small part because of security concerns, but if you are working on a system that is properly secured, or if you would rather not use WebDAV (described below), FTP is a workable and simple way to share Web calendars.

On my server, running ProFTPd, I decided to create a new user (calendar) with a password (cal4atf). To ensure that this user cannot be used for remote logins or other mischief, I would like to give it a shell of /sbin/nologin, or perhaps /bin/false—both of which are programs that simply exit, without giving a malicious user any chance to log in and take advantage of my system. The problem with this approach is that FTP servers allow only users whose shell is in /etc/shells to log in. This presents us with something of a dilemma. We want to give the calendar user a non-interactive shell, but we also want the user to be able to use FTP. But, adding /sbin/nlogin to /etc/shells might open a security hole on our system. A simple solution is to copy /sbin/nologin to /sbin/nologin-but-yesftp and to add a line in /etc/shells with the latter shell's name.

Normally, non-anonymous users logging in via FTP are shown their own home directories. By default, ProFTPd goes one step further than this, forbidding users from going outside of their own home directories. Thus, we can rest assured that even if a malicious user gets a hold of our calendar user name and password, the worst that he or she can do is destroy or modify our calendar files. This is obviously not something we want to encourage, and in a production environment, you undoubtedly would want better security—giving everyone a unique user name and password, for example. But for this simple demonstration, we will forge ahead with our single calendar user, knowing that a security breach might well take our shared calendar files with it.

Assuming that we have configured FTP appropriately, how can we publish our calendar? From within Sunbird, we select

the calendar we want to publish, which is called My Calendar by default. A menu pops up, the last option of which says, Publish entire calendar. If you select this option, a small dialog box opens, asking for the URL to which you intend to publish the calendar.

It goes without saying that the URL will begin with ftp://, but what comes after that? Assuming that the user name and password are as we indicated above, and that the server is calendar.lerner.co.il, we can access it as ftp://calendar:cal4atf@calendar.lerner.co.il/calendar.ics.

As you can see, we separate the user name and password with colons, and then put an @ sign between the password and the server name. Following the server is the name of the file we want to save. Although theoretically it can have any name or suffix, the .ics suffix is considered quite standard and ensures that all of the programs involved will understand the MIME types.

Now, let's say I make a change to my calendar. Must I now manually upload it to the server, going through this same procedure again? No, there is a way around this. Click on the calendar's name to get the same menu that you have already seen. Instead of selecting Publish entire calendar, select Edit calendar. This opens a dialog box that includes, among other things, a text field into which you can enter a URL, as well as a check box indicating that the calendar should be published whenever a change is made. I had mixed results using this functionality, although it worked more often than not and did a good job of keeping my appointments synchronized on different systems.

Subscribing to the shared calendar is similar to publishing it. Enter the full URL, including user name and password, and any iCalendar-compatible program should retrieve and display it. Of course, the configuration that we have put in place requires that the program can handle HTTP authentication.

mod_dav

FTP is fine for some tasks, but it has a number of drawbacks. To begin with, you might not want to run an FTP server on your computer, given its history of security problems. You also might prefer to have everything run over HTTP for performance reasons or because you can encrypt the transmission over SSL. For a variety of reasons, then, you might want to consider another alternative: mod_dav.

DAV, or Distributed Authoring and Versioning, makes it possible to create and modify files on a server, rather than just retrieve and read them. That is, DAV turns HTTP into a read-write protocol. DAV has been around for a number of years already, and mod_dav modules for Apache 1.x and 2.x have existed for some time. I am still using Apache 1.x on my main server, but it should be equally easy to install and use mod_dav for Apache 2.x.

To begin with, you need to download mod_dav (see the on-line Resources). Because I had compiled Apache with DSO (shared object) capabilities, I didn't have to recompile it from scratch in order to incorporate mod_dav. I merely had to tell it where to find apxs, the automatically generated Perl program that gives Apache modules all of the information they need in order to compile without the Apache source code. After unpacking the mod_dav source code, I typed:

```
./configure --with-apxs=/usr/local/apache/bin/apxs
```

Once done, I compiled and installed mod_dav:

```
make
make install
```

I double-checked to make sure that my Apache configuration file, httpd.conf, was still intact after the modifications provided by `make install`. Following that, I configured Apache to include a new named virtual server, which I called davcal.lerner.co.il:

```
<VirtualHost 69.55.225.93>
ServerName davcal.lerner.co.il
ServerAdmin calendar@lerner.co.il

# Directory and file names not beginning with /
# are relative to ServerRoot
ServerRoot /usr/local/apache/v-sites/davcal.lerner.co.il

DocumentRoot www
ErrorLog logs/error-log
CustomLog logs/access-log combined
CustomLog logs/referer-log referer

DAVLockDB  DAVLock
  <Directory
  /usr/local/apache/v-sites/davcal.lerner.co.il/www/>
    DAV On
    <Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL
           COPY MOVE LOCK UNLOCK>
      AuthName "Calendar DAV access"
      AuthType basic
      AuthUserFile passwd
      Require user calendar
    </Limit>
  </Directory>
</VirtualHost>
```

Notice the DAV-specific directives in the above configuration section. I have set up where the DAV locking will reside with DAVLockDB, obviously outside of the HTTP-accessible DocumentRoot directory. I then turn DAV on for a particular directory and limit DAV access to the calendar user, with a password specified in an external file. That password file, which is also outside of the Web site's root directory, is created and updated with the standard htpasswd program, located by default in /usr/local/apache/bin.

Finally, notice that our <Limit> section specifies limits only for potentially dangerous requests. The standard HTTP GET request, by contrast, requires no user name or password. This is a good configuration if you want to let anyone subscribe to your calendar but give limited access for publishing and modifying the calendar file. If this calendar were going to be used in a business, you probably would want to limit access to it as well, perhaps by giving each user his or her own password.

We can publish this calendar by bringing up (once again) the Publish entire calendar dialog for a particular calendar. This time, we use an HTTP URL, without specifying a user name or password: http://davcal.lerner.co.il/calendar.ics.

This publishes the calendar to the site, as you can tell by

looking at the appropriate directory on the server. You similarly can publish the calendar using WebDAV each time the calendar is updated, just as we saw before.

Finally, we can subscribe to this calendar using the same techniques that we have seen in previous months. Choose Subscribe to remote calendar from the File menu and enter the URL for this calendar file. Thanks to the magic of WebDAV, we even can use the same URL for writing and reading the file.

Conclusion

Although the open-source world might not have a fancy back-end calendar system like Microsoft Exchange, solutions exist that are more flexible and more than good enough for most groups.

I should note that Sunbird does appear to have some problems with publishing and subscribing; if nothing else, meetings that were listed as private on my Sunbird application continued to be marked in that way when the file was uploaded—and were then displayed as private when I subscribed to the calendar with a different program. Moreover, Sunbird continues to be slow when working with large calendars; however, that problem has been noted by the Sunbird developers and presumably will be fixed in the coming months.

There is also the promise of a new server for handling iCalendar files in Novell's Hula Project. Since Novell acquired both Ximian and SUSE, Hula is one of the most-hyped new projects to emerge from that combination. If Hula does indeed include iCalendar support, I will be curious to see how it improves on the FTP and WebDAV solutions I have outlined above. Until then, there are workable solutions that satisfy my own needs, as well as those of many other small organizations looking to collaborate with each other.

Resources for this article: www.linuxjournal.com/article/8323.

Reuven M. Lerner, a longtime Web/database consultant and developer, now is a graduate student in the Learning Sciences program at Northwestern University. His Weblog is at altneuland.lerner.co.il, and you can reach him at reuven@lerner.co.il.

DIY Swapfest On-line

DIYparts.org is what Christian Einfeldt calls "a bazaar where people can develop connoisseurship around putting open-source software on old but useful boxes, thereby keeping them out of landfill". Christian is the filmmaker behind the *Digital Tipping Point* documentary, and he co-created DIYparts.org with Adam Doxtater of Mad Penguin. Think of DIYparts.org as a swapfest for used gear of all kinds—from cases and motherboards to drives, monitors, racks, controllers, video cards, interface cards, PDAs and much more. See diyparts.org, www.digitaltippingpoint.com and

Kernel Mode Linux for AMD64

When user code runs inside the kernel, system calls become function calls, 50 times faster. How does that affect the performance of a real application, MySQL? BY TOSHIYUKI MAEDA

Kernel Mode Linux (KML) is a technology that enables the execution of user processes in kernel mode. I described the basic concept and the implementation techniques of KML on IA-32 architecture in my previous article, “Kernel Mode Linux”, which appeared in the May 2003 issue of *Linux Journal* (see the on-line Resources). Since then, I have extended KML to support AMD64, or x86-64, architecture, which is a viable 64-bit extension of the IA-32 architecture. In this article, I briefly describe the background of KML and then show the implementation techniques of KML for the AMD64 architecture. In addition, the results of a performance experiment using MySQL are presented.

The Problem of Protection by Hardware

Traditional OS kernels protect themselves by using the hardware facilities of CPUs. For example, the Linux kernel protects itself using a privilege level mechanism and a memory protection mechanism built in to CPUs. As a result, to use the services of the kernel, such as the filesystem or network, user programs must perform costly and complex hardware operations.

In Linux for AMD64, for example, user programs must use special CPU instructions (SYSCALL/SYSRET) to use kernel services. SYSCALL can be regarded as a special jump instruction whose target address is restricted by the kernel. To utilize system services or, in other words, to invoke system calls, a user program executes the SYSCALL instruction. The CPU then raises its privilege level from user mode to kernel mode and jumps to the target address of SYSCALL, which is specified by the kernel in advance. Then, the code located at the target address switches the context of the CPU from the user context to the kernel context by using the SWAPGS instruction. Finally, it executes the requested system service. To return to the user program, the SYSRET instruction reverses these steps.

Some problems exist, however, in this protection-by-hardware approach. One problem is system calls become slow. For example, on my Opteron system, SYSCALL/SYSRET is about 50 times slower than a mere function call/return.

One obvious solution to speed up system calls is to execute user processes in kernel mode. Then, system calls can be only the usual function calls, because user processes can access the kernel directly. Of course, it is dangerous to let user processes run in kernel mode, because they can access arbitrary portions of the kernel.

One simplistic solution to ensure safety is to use virtual machine (VM) techniques such as VMware and Xen. If user programs and a kernel are executed in virtual kernel mode, user programs can access the kernel directly. However, this protection-by-VM approach does not quite work, because the overhead of virtualization is considerable. In addition, although VM can prevent user programs from destroying the host system outside of the VM, it cannot prevent them from destroying the kernel inside the VM. It is unlikely that these difficulties could be solved even if CPUs, such as Intel’s Vanderpool and AMD’s Pacifica, provide better support for virtualization.

A recommended way to execute user processes in kernel mode safely is to use safe languages, also known as strongly typed languages. The recent advances in static program analysis, or type theory, can be used to protect the kernel from user processes. For example, many technologies already enable this protection-by-software approach, such as Java bytecode, .NET CLI, Objective Caml, Typed Assembly Language (TAL) and Proof-Carrying Code (PCC). I currently am implementing a TAL variant that is powerful enough to write an operating system kernel.

Based on this idea, I implemented Kernel Mode Linux (KML) for IA-32, a modified Linux kernel that can execute user processes in kernel mode, called kernel-mode user processes. My previous article described KML for IA-32. Since then, I have implemented KML for AMD64, because AMD64 has come into widespread use as a possible successor to IA-32. Interestingly, in spite of the similarities between IA-32 and AMD64, the implementation techniques of KML for these two architectures differ considerably. Therefore, I describe the basic concept, usage and implementation techniques of KML for AMD64 in the rest of this article.

How to Use KML for AMD64

KML is provided as a patch to the source of the original Linux kernel. To use KML, all you have to do is patch the original source of the Linux kernel with the KML patch and enable the Kernel Mode Linux option at the configuration phase, as you might do with other kernel patches. The KML patch is available from the KML site (see Resources).

In current KML, programs under the directory /trusted are executed as kernel-mode user processes. Therefore, if you want to execute bash in kernel mode, all you have to do is execute the following commands:

```
% cp /bin/bash /trusted/bin
% /trusted/bin/bash
```

How to Speed Up System Call Invocations

In KML for IA-32, system call invocations are translated automatically into fast, direct function calls without modifying user programs. This is possible because the recent GNU C Library for IA-32 has a mechanism to choose one of several methods that the kernel provides for system call invocation, and KML provides direct function calls as one way of invoking system calls.

However, the GNU C Library for AMD64 doesn’t have such a mechanism for choosing among methods of system call invocations. Therefore, I created a patch for the GNU C Library. With the patch, kernel-mode user processes can invoke system calls rapidly, because the invocations automatically are translated to function calls. The patch is available from the KML site (see Resources).

What Kernel-Mode User Processes Can Do

One of the advantages of KML is the kernel-mode user processes are almost the same as usual user processes except for their privilege level. That is, kernel-mode user processes can do almost anything that ordinary user processes can do. For example, kernel-mode user processes can invoke all system calls. This means they can use filesystems. They also can call open, read, write and other functions, including network systems, with socket, connect and bind. They even can create processes and threads with fork, clone and execve. In addition, they have their own memory address space that they can access freely. Even if a kernel-mode user process uses tons of memory, the kernel pages out the memory.

Moreover, the scheduling mechanism and the signal mechanism of the original Linux kernel work for the kernel-mode user processes. You can check this by executing the following commands:

```
% cp /usr/bin/yes /trusted/bin
% /trusted/bin/yes
```

You should notice that your system does not hang. This is true, because the kernel's scheduler preempts the kernel-mode yes and gives CPU time to other processes. You can stop the kernel-mode yes by sending Ctrl-C. This means the kernel can interrupt the kernel-mode yes and send a signal to kill it.

What Kernel-Mode User Processes Cannot Do

As described in the previous section, kernel-mode user processes are ordinary user processes and can perform almost every task that user processes can perform. However, there are a few exceptions:

1. Kernel-mode user processes cannot modify their GS segment register, because KML uses the GS segment register internally to eliminate the overhead of SWAPGS instruction.

2. 32-bit binaries cannot be executed in kernel mode on AMD64. KML for AMD64, like other typical OS kernels for AMD64, runs in 64-bit mode and there is no efficient way to let 32-bit programs directly call 64-bit functions.

Please notice that, as in the case of KML for IA-32, these limitations are present only in kernel-mode user processes. Ordinary user processes can alter their GS selector, and IA-32 binaries can be executed if an IA-32 emulation environment is set up.

How KML Executes User Processes in Kernel Mode

The way to execute user processes in kernel mode in AMD64 is almost the same as it is in IA-32. To execute user processes in kernel mode, the only thing KML does is launch user processes with the CS segment register, which points to the kernel code segment instead of the user code segment.

In AMD64 CPUs, the privilege level of running programs is determined by the privilege level of their code segment. This is almost the same as in IA-32 CPUs; the only difference is the segmentation memory system is degenerated in AMD64. Although segment registers still are used in 64-bit mode of AMD64, the only segment that the segment registers can use is the 16 EB flat segment. Thus, the role of the segment descriptors is simply to specify privilege levels. Therefore, only four segments—kernel code segment, kernel data segment, user code segment, user data segment—exist in 64-bit mode.

The Stack Starvation Problem and Its Solution

Although it is fairly easy to execute user processes in kernel mode, as shown in the previous section, there is a big problem—the stack starvation problem. The problem itself is almost the same as that of KML for IA-32, so I describe it briefly here. Further details are available in my previous article.

The original Linux kernel for AMD64 handles interrupts and exceptions by using the legacy interrupt gates mechanism. For each interrupt/exception, the kernel specifies an interrupt handler by using the interrupt gates in advance, typically at boot time. If an interrupt occurs, the AMD64 CPU suspends the running program, saves the execution context of the program and executes the interrupt handler specified in the corresponding interrupt gate.

The important point is the AMD64

CPU may or may not switch stacks before saving the execution context, depending on the privilege level of the suspended program. If the program is running in user mode, the CPU automatically switches from the stack of the running program to the kernel stack, whereas the CPU does not switch stacks if the program is running in kernel mode. The CPU then saves the execution context—RIP, CS, RFLAGS, RSP and SS register—to the stack.

Now, let us assume that a kernel-mode user process accesses its memory stack, which is not mapped by the page tables of the CPU. First, the CPU raises a page fault exception, suspends the process and tries to save the execution context. This cannot be done, however, because the CPU does not switch stacks, and the stack where the CPU is ready to save the context is nonexistent. To signal this serious situation, the CPU tries to raise a special exception, a double fault exception. Again, the CPU tries to access the nonexistent stack to save the context. Finally, the CPU gives up and resets itself. This process is known as the stack starvation problem.

To solve the stack starvation problem, KML for IA-32 uses the task management mechanism of IA-32 CPUs. The mechanism can be used to switch CPU contexts including all registers and all segment registers, when interrupts or exceptions are raised. KML for IA-32 switches stacks using the mechanism when double faults are raised. However, in 64-bit mode on AMD64, the task management mechanism cannot be used because it simply does not exist.

Instead, KML for AMD64 uses the Interrupt Stack Table (IST) mechanism, which is a newly introduced mechanism of the AMD64 architecture. In AMD64, the task state segment (TSS) has fields for seven pointers to interrupt stacks. In addition, each interrupt gate descriptor has a field for specifying whether the CPU should use the IST mechanism instead of the legacy stack switching, and if so, which interrupt stack should be used. If an interrupt occurs that is specified to use the IST mechanism, the CPU unconditionally switches from a user stack to the interrupt stack specified in the interrupt gate descriptor.

In KML for AMD64, all interruptions and exceptions are handled with the IST mechanism. Therefore, even if an interrupt or exception occurs while a kernel-mode user process is running with its %rsp pointing to an invalid memory, the kernel can keep running without any problem, because the CPU switches stacks automatically.

There are two reasons why KML for AMD64 handles not only double faults but also other interrupts and exceptions with the IST mechanism. One reason is that the overhead incurred by the IST mechanism is negligibly small. Therefore, I think it is better to keep it simple. Handling only double faults with the IST mechanism requires complex modifications to the original kernel, as in KML for IA-32. Second, the red zone of the stack is required by System V Application Binary Interface for AMD64 architecture. The red zone is a 128-byte memory range located just below the stack, that is, from %rsp - 8 to %rsp - 128. System V ABI for AMD64 specifies that user programs can use the red zone for temporary data storage and signal handlers, and interrupt handlers should never touch the zone. If KML handles an interrupt with the usual interrupt handling mechanism, this red zone is corrupted, because a stack is not switched. In this case, some CPU contexts are overwritten to the red zone if a kernel-mode user process is running. Therefore, KML for AMD64 handles all interrupts/exceptions with the IST mechanism in order to provide System V ABI to user programs correctly.

There also is a limitation in KML for IA-32: kernel-mode user processes cannot change their CS segment registers. This is

Table 1. Experimental Environment

CPU	Opteron 850 (2.4GHz, L2 cache 1MB) x 4
Memory	8GB (Registered DDR1-333 SDRAM)
Hard disk	146GB (Ultra320 SCSI 73GB x 2, RAID-0, XFS)
OS	Linux kernel 2.6.11 (KML_2.6.11_002)
Libc	GNU libc 2.3.5 + patch for KML
MySQL	MySQL 4.1.11

Table 2. Result of Wisconsin Benchmark (in seconds)

	CPU	User	System
Original Linux	753.86	611.78	142.08
KML	728.61	605.95	122.66

not possible because KML for IA-32 requires at least one scratch register to switch from a user stack to a kernel stack manually when exceptions or interrupts are raised. It prepares the register by using the memory where the CS register is saved. This limitation is not applicable to KML for AMD64, because stacks are switched by the IST mechanism. It is not so important, however, to change the CS segment register in 64-bit mode of AMD64 because there can be only two code segments.

Performance Measurement

To see how much performance improvement is possible, I ran the Wisconsin benchmark for MySQL both on the original Linux kernel and on KML, using sql-bench, which comes with MySQL. The experimental environment is shown in Table 1. In the test on KML, both the MySQL server and the benchmark client were executed as kernel-mode user processes, and the patched GNU C Library was used to eliminate the overhead of system call invocations. In addition, the loop count of the test was increased to 10,000, as the default loop count of 10 was too small to produce meaningful results.

The result is shown in Table 2. The second column shows the total CPU time consumed by the benchmark. The third and forth columns show the breakdown of the total CPU time. The third column shows the CPU time consumed by the user process, and the forth column shows the CPU time consumed by the kernel.

The results show that the total CPU time was improved by about 3%. The user CPU time was improved by about 1%, and the system CPU time was improved by about 14%. The result indicates that KML could improve the performance of database applications slightly by eliminating the overhead of system call invocations.

Conclusion and Future Work

KML is a modified Linux kernel that can execute user processes in kernel mode. By executing in kernel mode, the performance of user programs can be improved by, for example, eliminating the overhead of system call invocations. Besides the performance improvement, KML also can be used to ease inspection and debugging of the kernel and development of kernel modules, because kernel-mode user processes can access the kernel and use a large amount of memory and CPU time. I now am considering implementing a helper library to provide kernel-mode user processes with an easy way to access kernel functions and data by exporting them as some kind of shared object.

Resources for this article: www.linuxjournal.com/article/8327.

Toshiyuki Maeda is a PhD candidate in Computer Science at the University of Tokyo. His favorite comic artists are Osamu Tezuka, Fujio F. Fujiko and Amin Oka da.

The Ultimate in Small Linux

Turn a borrowed machine into your personal Linux box, with a distribution you can carry on a business-card CD or USB key. **BY MARCEL GAGNÉ**

Honestly, François, why does *ultimate* always have to mean bigger, faster and more resource-intensive? *Mon Dieu*, sometimes all this speeding up just seems to make things work more slowly. Although I think your idea of building a supercomputer cluster in the restaurant would be a wonderful idea for this month's Ultimate Linux Box issue, there simply isn't room. The wine cellar? *Non*, François, the wine cellar is for wine and I would like to keep it that way, and I'm sure our guests would agree. Speaking of which, they will be here any moment.

Ah, François, they are already here. Welcome, everyone, to *Chez Marcel*, home of the world's greatest wine cellar and of course, the best in fine Linux fare. Your tables are ready. Please sit and make yourselves comfortable. François, to the wine cellar! Please bring back the 2003 Auslese Riesling from Germany. *Vite!*

While my faithful waiter fetches the wine, let's take a look at another definition of what constitutes the ultimate Linux box. François suggested a supercomputer. I was thinking of something much smaller, but nevertheless extremely useful—something small enough to fit in my pocket. On more than one occasion, I've been saved by having a copy of Linux with me. Actually, the person saved was usually a user of another OS who had the kind of trouble that only a Linux system could help them out of. The mini-distributions I carried with me tended to be single-diskette (sometimes two or three) distributions with basic text-based tools. Today, I want to introduce you to a couple of excellent ways to take Linux with you wherever you go. These mini-distributions are no longer stripped-down sets of text-based tools, but fully graphical, fully networked distributions that still can fit in your pocket or wallet. Best of all, they can run entirely from a live mini-CD or USB key.

The first item on the menu is one of my personal favorites, the cleverly named Damn Small Linux (DSL). DSL is a Debian-based distribution built using live CD technology. The whole thing is less than 50MB and can fit on a business-card-sized CD, which you can get at your local computer or office store. Download your ISO image (see the on-line Resources), burn it to a CD (it easily can be a standard CD as well as the business-card size), and reboot your PC.

DSL is extremely light and fast. It uses Fluxbox for a window manager. You can run it on modest hardware with very little memory—as little as 16MB. DSL comes with a number of desktop applications, all of which are designed to be equally light and fast. There are Dillo and Firefox Web browsers, Sylpheed for e-mail, an instant messaging and IRC client called Naim, XMMS for music, Xpaint for graphical editing and screenshots, FLwriter for word processing, Siag for spreadsheets and a host of others. Check out Figure 1 to see DSL in action.

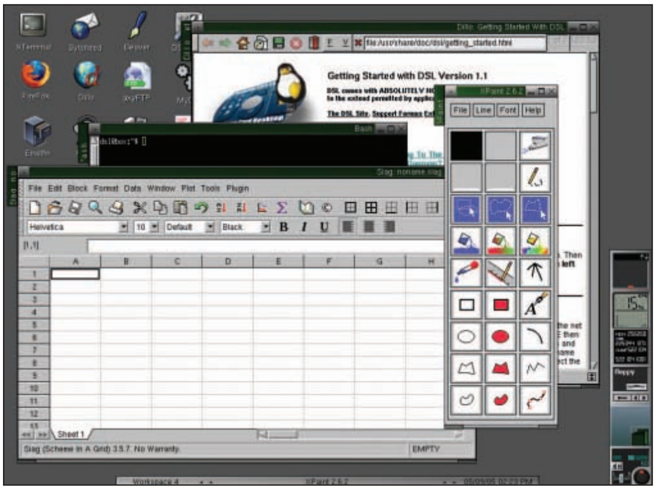


Figure 1. DSL provides a rich but resource-lean desktop experience.

There is no program starter button in the lower left-hand corner with this distribution. To bring up the menu, right-click anywhere on the desktop and the top-level application menu appears offering you several submenus covering everything DSL has to offer. To banish the menu, left-click on a blank portion of the desktop.

One of the first things you probably want to do is set up networking. Right-click to bring up the menu, select System, then Net Setup. The options there include dial-up, network card configuration, DSL (the other DSL) and some wireless support as well. ndiswrapper is included for those cards that support only Microsoft drivers. All of these network choices are menu-driven; simply fill in the blanks.

Speaking of the System menu, look under Daemons and you'll discover another rather amazing aspect of DSL. An SSH server, NFS, a Web server and an FTP server are there as well. Printer dæmon support also is available using classic LPD.

In all of this, DSL still manages to include some desktop eye-candy. From the Desktop menu, navigate over to Styles and you can choose from a small handful of alternate looks.

Before I move on to the next item on today's menu, let me direct you to the Tools menu under Apps. Look near the bottom and you'll see an option to install DSL to a hard drive, which can be pretty tiny, as well as one to install it to a USB pen drive so you can carry it with you. There are also menu items to enable apt and Synaptic so you easily can install other packages. The usefulness of this is obvious if you install to disk, but look back to the top of the Tools menu for another reason.

The option is labeled Make myDSL CD remaster, and with it you can create your own custom-DSL distribution. When you click on this option, another window appears with instructions on how to change to runlevel 2 to remaster. In effect, you need to reboot and type `dsl toram 2` at the boot prompt. Then, when the shell prompt appears, type `mkmydsl`. This process is somewhat beyond the space I have allotted, but I direct you to www.damnsmalllinux.org/talk/node/113 if you want to roll your own DSL.

Another tiny graphical Linux you might want to look at is Puppy Linux. This fully networked distribution also comes with a bevy of applications. In terms of networking, Puppy comes with Mozilla for Web browsing as well as sending and receiving e-mail with Sylpheed, SSH for remote administration, Gphone for VoIP calls, VNC and rdesktop clients to control remote desktops and much more. AbiWord is included for word processing, as is the Scribus desktop publishing application. There are file managers, graphic editors, HTML editors, a spreadsheet program, a personal finance application and more.

There's also a small handful of games. *Bubbles*, somewhat reminiscent of *Frozen Bubble*, is a lot of fun, as is *gtkfish*. That last one is a strange little game where you go fishing with a tissue-paper net. If the fish move too fast when you catch them, they break the net. Click the left mouse button to drop the net below the water and go for the slow-moving fish. Release the mouse button to catch the fish. It's very strange and yet strangely addictive.

For a copy of Puppy Linux, go to the Web site and download the latest ISO image (see Resources). Use your favorite CD burning tool (I tend to like K3b) and create your CD. When you have your freshly burned CD in hand, pop it in to the drive and reboot your system.

When Puppy Linux starts up, the first thing you'll see is a keyboard selection screen. I scrolled down to us qwerty and pressed Enter. It then asks for your mouse type. In all likelihood, you simply can accept the default choice made for you, in my case, ps/2. The program then asks if you have a scroll wheel. Immediately after this, the graphical desktop starts, offering you a chance to select the video mode you want to use whether it be 648x480, 800x600 and so on. The resolution will change on the fly and you can

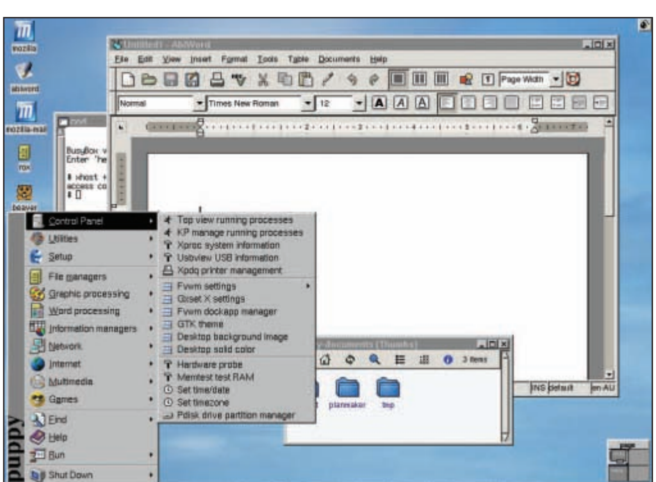


Figure 2. Haven't you always wanted a Puppy—Linux, that is?

lock it in by clicking OK at any time. That's it. Your Puppy Linux system is up and running (Figure 2). You even can remove the CD at this point.

On the Puppy Linux page, there's a statement that effectively says you can install Puppy to anything whether it be a hard drive, a Zip disk, on a network (to boot a thin client) or a USB key, much like DSL. That's the one that really got me excited. I kind of like the idea of carrying a fully graphical Linux system in my pocket. Besides, Puppy, in its default configuration, is too big to fit on a 50MB business card without some tweaking (more on that).

Click the Start button then head to the Setup menu. Under that heading and near the bottom, you'll find some rather interesting options, one of which is to install Puppy to USB card. Choosing this brings up a dialog that takes you through the various steps from plugging in your USB key to selecting a drive (if you have more than one plugged in), choosing a partition and finally copying the files. The copy itself can take place from local files on the hard disk or the live Puppy CD that you booted from.

The next step takes a few minutes while various files are copied (vmlinuz, image.gz and usr_cram.fs). After the copy is complete, you are asked to choose a default keyboard language. I chose us and pressed Enter. You have one more choice to make after this point and that's to decide how the Puppy filesystem is stored. The first choice is a vfat partition mounted as /root with no other changes. The second creates a small ext2 filesystem on the partition. This is the preferred choice and a more efficient one. The first option does have the advantage, however, that its files can be seen under Windows. I chose option 2 and pressed Enter.

Now that Puppy is installed to your USB key, you can edit the boot-up script to provide a password to an encrypted filesystem. This is an excellent idea if you want an additional level of protection in case your USB key is ever lost or stolen. Finally, your USB drive is made bootable and you are ready to take your Puppy for a walk (Figure 3).

A word of caution though—not every PC knows how to boot from a USB drive, although you may be able to change the boot device settings in your BIOS if it doesn't work immediately. If your PC still doesn't support a USB drive boot, there is still hope, assuming you have a diskette drive. On the Puppy site, there is a boot image (called boot2pup.img.gz) that you can copy to a diskette. Uncompress the image, then copy it:

```
gunzip boot2pup.img.gz
dd if=boot2pup.img of=/dev/fd0
```

Now, just make sure you carry this diskette with you as well.

Before I wrap up this exploration of Puppy Linux, I want to tell you about another great little feature. Under that Setup menu is an option labeled Remaster Puppy live-CD. This is a simple script that takes you through the various steps necessary to copy your existing CD into RAM (so you need at least 256MB for this), edit the filesystem, re-create the image and finally, burn it to a CD.

It takes a couple of tries to get the hang of it, but all in all, it's not a bad process. One strange step asks you to confirm your CD burner and reader. It is at this point that Puppy will reboot (yes, I know it sounds strange for a live CD) in order to turn on SCSI emulation. When the system is back up, return to the Setup menu and restart the remaster program. It should jump immediately to step three where you'll be asked to insert the CD into whichever device you identified as the reader. What follows is a question-and-answer session that lets you define exactly how you would like your next version of Puppy to appear.

As I mentioned, it can take a little time to get the hang of this, but treat it as a hobby project, and you'll be a pro in no time. When you have finished creating the new ISO image, Puppy launches the Gcombust CD burning program to let you finish the job.

Mon Dieu! Is it that time already? The clock seems to be telling us that closing time has arrived. No need to rush though. Relax a little longer as I am sure François would be more than happy to refill your glasses. Grab one of those business-card CD blanks and cook yourself up a little Linux to take home with you. Please raise your glasses, *mes amis*, and let us all drink to one another's health. *A votre santé! Bon appétit!*

Resources for this article: www.linuxjournal.com/article/8326.

Figure 3. The definition of take-anywhere Linux: Puppy Linux on a USB key.

Marcel Gagné is an award-winning writer living in Mississauga, Ontario. He is the author of *Moving to the Linux Business Desktop* (ISBN 0-131-42192-1), his third book from Addison Wesley. He also makes regular television appearances as Call for Help's Linux guy. Marcel also is a pilot, was a Top-40 disc jockey, writes science fiction and fantasy and folds a mean Origami T-Rex. He can be reached by e-mail at mggagne@salmar.com. You can discover a lot of other things (including great wine links) from his Web Site at www.marcelgagne.com.

The Future of Linux Security

Just because Linux can be more secure than other systems doesn't mean that your Linux system is. How can developers and distributors help the sysadmins of the future? BY MICK BAUER

Did you know that I've been writing this column for the better part of five years? And what an action-packed five years they've been! In that time, we've seen some of Linux's biggest former competitors embrace it, and Linux has made significant inroads as a desktop platform.

In the realm of Linux security, there also have been remarkable advances. Linux's firewall functionality now is so mature that it's the basis for a number of embedded firewall appliances, not to mention countless non-security-related devices as well. Linux supports a staggering variety of security tools, making it a favorite among security auditors and consultants. In addition, Linux has formed the basis for several ultra-secure role-based access control (RBAC)-based operating systems, most notably the NSA's SELinux.

But what about the future of Linux security? I've written a lot about present and past Linux security issues but never about the future, aside from my interview with the forward-looking Richard Thieme. This month, I'd like to indulge in a little speculating and editorializing and talk about where I think Linux security will go and where I think it *ought* to go.

What's Wrong with the Present?

The revelation a lot of people have been having about Linux security lately is typical Linux systems are not that much more secure than are typical Microsoft Windows systems. Before the e-mail flames begin, let me explain this statement. First, personally, I do happen to think that Linux is more securable than Windows, and I've said so repeatedly in this very column over the years. Users simply have more control over their Linux systems' behaviors than they do with an equivalent Windows system.

The problem is Linux users, like Windows users, tend to focus most of their energy on getting their systems to do what they need them to do, and they place too much trust in their system's built-in or default security settings. Then, when the inevitable software bugs surface, those bugs' effects tend to be more extensive than they would have been had greater precautions been taken.

For example, if I run BIND v9 for name services, it takes some work and some research to get things working. It takes still more work to get BIND running in a chroot jail, so that the named process can see and use only a subset of the server's filesystem. Therefore, many if not most BIND users tend *not* to run BIND in a chroot jail. When a BIND vulnerability surfaces in the wild, the majority of BIND users probably experience more pain than if they'd done the chroot thing. It's probably the same amount of pain they would experience if they had run a Microsoft name server with fewer security features than BIND has.

All of this is simply to say that many of Linux's security features and capabilities are not taken advantage of by its users. The end result is, at least according to friends of mine who regularly do professional penetration testing, your average Red Hat Enterprise system isn't significantly harder to break in to than your average Windows 2003 Server system.

This is unfortunate and perhaps surprising. Given the complete transparency of its codebase, Linux still seems to be prone to the same kinds of software bugs, in roughly the same quantity and frequency, as Windows. But if you think about it, why wouldn't this be so? As with Windows, Linux represents an amazingly complex mass of code produced by hundreds of different people. The more code there is, the more bugs there may be, right?

I recently was interviewed by SearchSecurity.com for an article about a Microsoft-funded study conducted by Security Innovation, Inc. The study concluded that Windows is more secure than Linux, a conclusion based mainly on frequency of security bugs and mean time to issue patches. I believe I correctly criticized the study for looking only at these easily quantifiable aspects of security and not taking into consideration Linux's other security advantages, such as customizability and greater choice of software packages. In other words, I felt the study had the most relevance when comparing default installation scenarios, irrespective of each OS' potential for being secured by its users.

But the more I think about it, the more I worry that perhaps a platform's security potential doesn't count unless most systems running that platform actually reach that potential. This isn't strictly a function of end-user behavior; I'm not trying to impugn system administrators. As I elaborate later, I think Linux's developers and distributors must continue to figure out ways to make security features more ubiquitous, transparent and easy to configure and use. By the way, because I'm comparing Linux with Windows, in fairness I should point out that Windows too has many security features that its users often do not take advantage of.

Okay, Linux and Windows both are much less secure by default than they could be, and both are subject to an unwinnable race between software bugs and security patches. What else are we up against?

Alas, both operating systems use a rather primitive discretionary access control model in which entire categories of security settings and behaviors are optional. In this model, one superuser account—root in Linux, Administrator in Windows—has god-like power over the entire system, including other users' files. In both OSes, group memberships can be used to create different levels of access, say, to delegate various root powers. In practice, however, on most systems you have to be logged on as the superuser or temporarily become that user in order to do anything important.

As a result, gaining complete control over any Linux or Windows system is a matter of compromising any process running with superuser privileges. But wait, you say, I've configured my important dæmons to run as unprivileged users; bugs in those dæmons can't lead to total compromise, can they? No, not directly, but bugs in other software may make it possible for a non-root process to escalate its privileges. For example, suppose you've got a Web server running Apache, and one day an attacker manages to exploit an unpatched Apache buffer-overflow vulnerability that results in the attacker getting a shell session on your server. At this point, the attacker is running as www, because that's the user Apache is running as. But suppose further that this system also has an unpatched kernel vulnerability that involves local privilege escalation.

You, the system administrator, may even know about this vulnerability but have opted not to patch it, because after all, it's strictly a local vulnerability, and nobody besides you has a shell account on this system, and who wants to have to reboot after patching the kernel? But now a remote attacker does have local shell access, and if she successfully exploits this kernel vulnerability, she's root! This all-too-common scenario illustrates that bugs are bad enough, but they're even worse when combined with a root-takes-all security model.

This, in a verbose nutshell, is the present state of Linux security. Securing Linux requires us to expend considerable effort to take full advantage of sometimes-complicated security features that usually are not enabled by default, to keep absolutely current on all security patches, and to do all of this within the limitations of Linux's simple security model. But we're in good company: most commonly used contemporary operating systems have exactly the same limitations and challenges.

Mandatory Access Controls

I've alluded to the fact that access controls or file permissions on Linux, UNIX in general and Windows are discretionary, and that this is a weak security model. Well, what about SELinux? Doesn't that use RBACs and type enforcement (TE), both of which are examples of mandatory access controls? Yes, indeed, it does. But I'm afraid that this probably isn't the future of Linux security, for the same reasons that SELinux isn't a huge part of present Linux security.

RBACs restrict users' behavior and access to system resources based on carefully defined roles that are analogous to but more far-reaching than the conventional UNIX groups mechanism. Similarly, type enforcement restricts processes' activities based on their predefined domains of operation. The net effect of RBAC and TE is to create segregated silos (my term) in which users and processes operate, with strictly limited interaction being permitted between silos.

This is an elegant and effective security model. However, for most people, RBAC, TE and other mandatory access controls are too complicated and involve too much administrative overhead. This, in many people's view, dooms SELinux and similar operating systems to the realm of niche solutions: OSes that are useful to people with specific needs and capabilities but not destined for widespread adoption. Despite admiring SELinux's security architecture and being a fan of the concept of RBAC in general, I do not think that mandatory access controls by themselves are likely to revolutionize Linux security.

Hypervisors and Virtual Machines

If RBAC and TE do in fact prove too unwieldy to compartmentalize security breaches at the OS level, hypervisors and virtual machines (VMs) may achieve this at a higher level. We're already familiar with virtual machines in two different contexts: runtime virtual environments, such as those used by Java programs, and virtual platforms, such as VMware, plex86 and VirtualPC, that allow you to run entire operating systems in a virtualized hardware environment.

The Java Virtual Machine was designed with particular security features, most notably the Java sandbox. In general, though, Java security comes from the fact that Java applets run isolated from raw or real system resources; everything is mediated by the Java Virtual Machine. Besides being a good security model, it's also relatively simple to use safely, both for programmers and end users. Java also is, for many reasons, already ubiquitous.

Virtual platforms take this concept a step further by mediating not only individual programs but the operating systems on which they run. Security architecture in this scenario, however, isn't as mature as with the Java Virtual Machine. For the most part, security is left to the guest operating systems running in the virtual environment. A SUSE Linux virtual machine running on VMware, therefore, is no more or less secure than a real SUSE system running on its own hardware.

Hypervisor technology addresses the need to isolate virtual machines running on the same hardware from one another, restrict their interactions and prevent a security breach on one virtual machine from affecting others. IBM has created a security architecture called sHype for hypervisors. An open-source hypervisor/virtual-machine project called Xen also is available.

Although the driving purpose of a hypervisor is to prevent any one virtual machine from interfering with other virtual machines running on the same hardware—for example, by monopolizing shared hardware resources—the idea of having some sort of intelligence managing systems at this level is powerful. It may even have the potential to overshadow or, at the very least, significantly augment traditional intrusion detection systems (IDSes) as a means of detecting and containing system compromises.

Mandatory access controls and hypervisors/virtual machines aren't mutually exclusive. On the one hand, I am of the opinion, strongly influenced by my friend and fellow security analyst Tony Stieber, that hypervisors have much greater potential to shape the future of Linux security than do MACs. But on the other hand, the two can be used together. Imagine a large, powerful server system running several virtual machines controlled by a hypervisor. One VM could be running a general-purpose OS, such as Linux, serving as a Web server. Another VM, serving as a database for sensitive information, could run a MAC-based OS such as SELinux. Both VMs would benefit from security controls enforced by the hypervisor, with SELinux providing extra levels of security of its own.

Anomaly-Based Intrusion Detection and Antivirus

One additional technology, like MACs and hypervisors, already exists today but potentially will have a much bigger impact on the future: the anomaly-based intrusion detection system. The idea of anomaly-based IDS is simple: it involves creating a baseline of normal network or system activity and sending an alert any time unexpected or anomalous behavior is detected.

If the idea is simple and the technology already exists, why isn't this approach commonly used? Because it isn't nearly as mature or easy to use as signature-matching. We're all familiar with signature-based IDSes; they maintain databases of attack signatures, against which observed network packets or series of packets are compared. If a given packet matches one in the attack database, the packet is judged to be part of an attack, and an alert is sent.

The strengths of this approach are that it's easy to use and typically involves few false positives or false alarms. The fatal weakness of signature-based systems is if an attack is too new or too complicated for there to be a corresponding signature in your IDS' signature database, it is not detected.

With anomaly-based IDS, in contrast, any new attack that sufficiently differs from normal behavior is detected. The trade-off is the IDS administrator must train and periodically re-train the IDS system in order to create the normal-behavior baseline. This results in a period of frequent false positives, until the baseline has been fine-tuned.

I attended a lecture by Marcus Ranum in 1999 or so in which he described anomaly-based systems as the future of IDS. Obviously, we're not there yet. Such products are available from vendors such as Lancope and Arbor Networks. But I remain hopeful that someone will figure out how to do this sort of thing in ways that are cheaper and easier to use than current systems. Potentially, this could lead to a sort of network hypervisor that lends the same sort of intelligence to networks, whether composed of virtual or real machines, that hypervisors lend to virtual platforms.

By the way, virus scanners need and can benefit from anomaly detection technology as much as IDSes do. This point is illustrated amply by the fact that the vast majority of organizations that use modern virus scanners, which rely almost exclusively on virus-signature matching, nonetheless suffer from major virus/trojan/worm outbreaks. Current signature-based antivirus tools clearly are not effective enough.

Conclusion—and Goodbye for Now

So those are my thoughts on the future of Linux security. In the meantime, keep on using the techniques this column has focused on over the years: firewalls, virus scanners, automatic-patch/update tools, VPNs and application-specific security controls such as chroot jails and audit trails.

With that, I bid you farewell, not only for this month but indefinitely. It's time for me to focus on other things for at least a little while and allow fresh voices to take over the Paranoid Penguin. I'm continuing in my role as Security Editor and in that capacity will keep on doing my bit to help *Linux Journal* bring you outstanding security content. I also will try to contribute an article now and then myself, on an ad hoc basis. But the article you are reading now is my last as exclusive author of this column.

Thanks to all of you for five years of support, encouragement and edification—I've never made a mistake in this column that wasn't noticed and corrected by someone out there and always to my benefit. It's been a great five years, and I'm grateful to this terrific magazine's staff and readers alike for all you've done for me!

Resources for this article: www.linuxjournal.com/article/8329.

Mick Bauer, CISSP, is *Linux Journal*'s security editor and an IS security consultant in Minneapolis, Minnesota. O'Reilly & Associates recently released the second edition of his book *Linux Server Security* (January 2005). Mick also composes industrial polka music but has the good taste seldom to perform it.

Porting LinuxBIOS to the AMD SC520

Building a Linux system that will boot in seconds, not minutes, requires a custom BIOS. But thanks to a new compiler and development process, we can build a BIOS for a new motherboard with only C code—no assembly. BY RON MINNICH

In this article, we describe the work done by the Cluster Research Team at Los Alamos National Laboratory to port LinuxBIOS to the AMD SC520 CPU. Although space does not permit a detailed description of all the work involved, we hope you can get some idea of what it takes to port to a new board.

The AMD SC520 is a small, low-power, integrated CPU. It is used in many embedded applications, one of the more interesting being the Portland Aerospace Society's open-source rocket. This rocket uses a standard board from Kontron to control all onboard computing functions. The board features a number of nice control buses, including the CAN bus for power control of rocket subsystems.

We were asked whether we could port LinuxBIOS to the board the rocket team uses. We purchased the board they use and found one main problem: the BIOS Flash is soldered on. If you burn a bad BIOS, the board is now a nice paperweight. It might be nice to have a fancy burned-out board as a paperweight, but we would rather have working boards.

After doing some research, we learned that Advanced Digital Logic (ADL) makes a nice SC520 board with a removable BIOS Flash part. We decided to use this board for development. We've used ADL boards for our miniclusters in the past, and they've worked well.

We would start our work by porting to the board with removable Flash. Once the port is solid, our plan was to take a deep breath and try it on the board with a non-removable Flash. If we fail, of course, we're the proud owners of a $400 brick!

Steps in Porting LinuxBIOS

The steps in any LinuxBIOS port process change little from board to board. First, enumerate the resources provided on the mainboard, such as the CPU, I/O parts and so on. Next, create the configuration files that describe the resources and populate the directory tree with those files. Then, fill in the blanks with code.

LinuxBIOS itself is about 98% C code. The small amount of assembly involved is common to almost all the boards for a given CPU. In this sense, LinuxBIOS is a far better piece of code than proprietary BIOSes, which we are told are almost completely assembly code. We have not seen this source code, of course.

How the LinuxBIOS Build Process Works

The LinuxBIOS build process bears little resemblance to the Linux kernel build process. Instead, the LinuxBIOS build process was inspired by the Plan 9 and BSD kernel build processes, although the LinuxBIOS process adds more formality and control. A lot of checking is needed for building a BIOS, as the price of error is high. Because our clusters may have 1,024 or 2,048 nodes, we want to make sure that the BIOS we flash to all the nodes at once is good. As we will see, however, we can afford to flash a bad BIOS if we use LinuxBIOS's fallback BIOS feature.

A target is a specific instance of LinuxBIOS for a motherboard. As built for a target, a LinuxBIOS image consists of glue code for resource management code and the resource code itself. A resource can be thought of as one or more .c files that control a hardware component, be it a motherboard, CPU or other chip. Resource code can invoke code for other resources as part of the configuration process. For example, the motherboard resource invokes code for CPU startup.

Each resource has a directory, so for the SC520, we need to have a directory called src/cpu/amd/sc520. The directory includes source code and two configuration files, one of which specifies options used for the resource and default option values. The other specifies what parts are built and how they are built. A given configuration file for a resource may specify other resources to be used, in which case the configuration files for those resources are read in and processed.

The LinuxBIOS configuration tool, starting from an initial configuration file called the target configuration file, creates a build directory. Once the configuration tool is run, the user changes to the build directory and types `make`. At that point, an image of the LinuxBIOS for that target is built and can be burned into Flash.

A given motherboard can have several target configuration files. Different options may be set for these different targets. One target might have a lot of debugging, another might use a different bootloader and so on. All of this control is set by options in the build process.

Options are defined in the LinuxBIOS source tree, and only defined options may be used. Options have default values and can be set only once in order to avoid confusion in how they are set and what values they may have.

The goal of this process is to make it easy to build all the targets on a single machine, quickly, while having only one copy of the source. A second goal is to avoid errors that cropped up in earlier versions of LinuxBIOS, when options were uncontrolled or set in too many places. The process supports cross-compilation, so we can build our PowerPC targets on an x86 machine.

LinuxBIOS Directory Tree Structure

A portion of the LinuxBIOS directory tree structure is shown in Figure 1. Starting at the top of the tree, there are three main directories: src, targets and util. The src directory contains all the source for all the BIOSes—all mainboards, all CPUs, all devices and so on. You build a specific BIOS in the target directory using a config file. For example, for our project, we built our BIOS in the targets/digitallogic/msm586seg directory, using the file Config.lb in that directory. Finally, the util directory contains many utilities used to create BIOS files or to burn the BIOS image into the motherboard Flash part.

Configuration Files

Configuration files in LinuxBIOS describe resources and how they are used in the construction of a target. Each resource can have a set of options defined for it. The set of all available options is defined in one file, src/config/Options.lb; only options defined in that file may be used or set in configuration files. Once a resource is named in a configuration file, resources defined within the scope of that resource inherit the options settings for that resource. The options have lexical scope; once the block for the resource ends, the options revert to values they had before the block was started. Options may have a default value set in the Options.lb file, or it may not be set; they may have a default value set in the mainboard configuration file; or they may be set in the target configuration file. To avoid the confusion we saw in earlier versions of the configuration tool, options may be set in only a few places: the target file, the mainboard file and CPU files. Options may be set only once. Thus, an option may have a default value, which can be changed once and only once in a configuration file. Forcing the set-once rule avoids problems we saw earlier with dueling configuration files.

A full writeup on the configuration language would consume this entire article. Therefore, this article touches on the important points, but we cannot cover all the aspects of the configuration language.

Static vs. Dynamic Information

In all mainboards, some resource hardware can be queried to determine what other resources it needs, for example, how much memory and I/O space it needs. There also is hardware that cannot be queried, such as the wires that wire a PCI slot to an interrupt controller. For the latter type of resource, the only way to tell the BIOS about it is to put the information directly into the BIOS. Unfortunately, this information is contained in many places in PC BIOSes. Interrupt routing may be found in the $PIR (uniprocessor), _MP_ (multiprocessor or IO-APIC) or ACPI tables. The configuration tool must generate these tables, but the user in turn must tell the tool what values go in the tables.

Super I/O chips cannot be queried dynamically, and the location in I/O space and type of Super I/O chip must be specified in the mainboard configuration file.

Newer PC mainboards are harder to figure out at runtime. For example, Opteron processors have three HyperTransport ports that can be wired in arbitrary configurations on different mainboards. The configuration file for a mainboard has to specify how these ports are wired.

Compiling C Code without Memory: romcc

On modern systems, with Synchronous DRAM chips, the memory is not accessible until a lot of setup has been done. The size and parameters of the DRAM are read in over a two-wire bus called the SMBUS. Thus, in order to establish working memory, the BIOS has to:

- Turn on the chipset to some extent.
- Enable the SMBUS, usually on a Super I/O or southbridge.
- Read in parameters of DRAM over SMBUS; more than 20 in some cases.
- Perform complex calculations to determine timing.
- Initialize DRAM control registers with proper values.
- Perform a complex sequence of reads not writes from DRAM to get it running.

All this has to be done without a stack, which means that function calls and variables are almost impossible to use. Without memory, programming is limited to the registers. Function calls can be made only one level deep. In the bad old days, a big, bad ball of assembly code was used to get this work done. Expert assembly code writers used every trick in the book to get this code working. Writing this code is the sin-

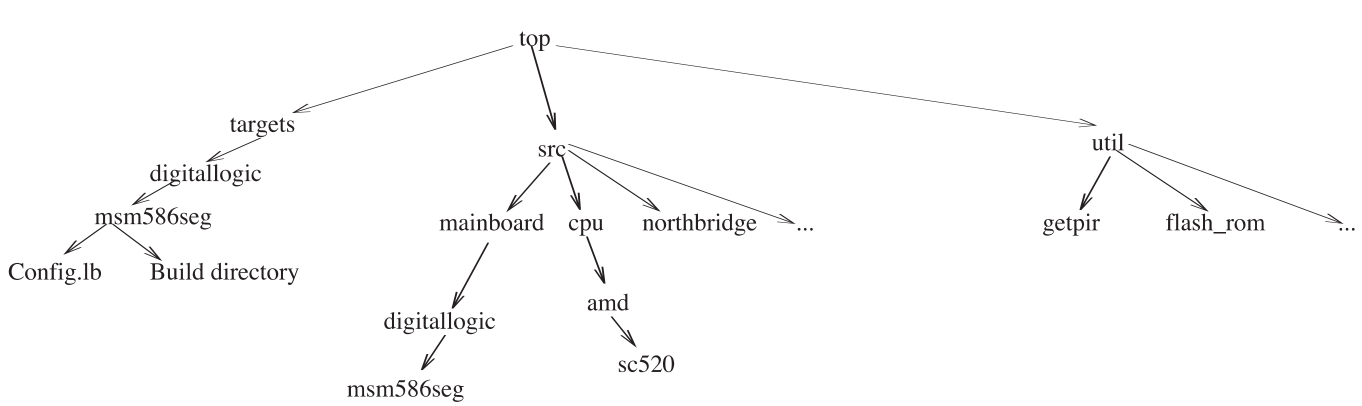


Figure 1. The LinuxBIOS directory tree includes three top-level directories for source, config files and utilities.

gle hardest part of any BIOS.

In 2002, Eric Biederman of Linux NetworX developed a compiler called romcc. romcc is a simple optimizing C compiler—one file, 25,043 lines of code—that uses only registers, not memory. The compiler can use extended register sets such as MMX, SSI or 3DNOW. romcc allowed us to junk almost all of the assembly code in LinuxBIOS, so that even the earliest code, run with no working DRAM, can be written in C.

romcc is used only for early, pre-memory code. For code that runs after memory comes up, we use GCC.

What the Build Process Builds

The build process builds a binary image that is loaded to a Flash part. LinuxBIOS provides a utility, flash_rom, for this purpose. Alternatively, you can use the MTD drivers in the Linux kernel.

The layout of a typical ROM image is shown in Figure 2. The top 16 bytes contain two jump vectors, a jump to the fallback and a jump to the normal. LinuxBIOS always jumps to the fallback first. If all is well, it jumps back to the jump to normal vector at the top of memory, and from there to the normal image. If the fallback code detects problems or if the CMOS settings indicate that fallback BIOS should be run, the fallback BIOS runs.

Building a Tree for the SC520 Board

Enough overview, let's get to work. To build support for a new board, we start with the mainboard first, and the easiest way to do this is to pick a similar mainboard. Because the Digital Logic ADL855 is much like the SC520, we start with that. We can clone much of the directory structure of the ADL855 for the SC520 board.

Mainboard Tree and Files

The basic naming process for directories in LinuxBIOS is to name the type of resource, in this case, mainboard; the vendor, here digitallogic; and the part name, in this case, msm586seg. Before we start the mainboard configuration file, we need to know what's on this mainboard. We don't have to get everything at first; in fact, we can leave a lot out simply to get something to work. Typically, the best approach is to make sure you know what drives the serial port and make sure you get that. To get DRAM up, you need to make sure you set up whatever device drives the SMBUS. None of these chips are in the right state when the board is turned on; you need to set a few bits to get things going.

For figuring this all out, you have a few choices. Almost always, the easiest thing to do is boot Linux and type `lspci`. For work with this type of board, it's easiest to have a CompactFlash part with a small Linux distribution installed so you can boot long enough to run the lspci command. You can use lspci to dump configuration space registers too, which sometimes is invaluable for discovering how to set control bits the vendor might have forgotten to tell you about. The setpci command also is handy for probing bits and learning the effects of setting and clearing them. On several boards, we've used setpci to probe the chipsets to find undocumented enable lines for onboard devices.

Devices

Although lspci shows discrete devices, on the SC520 they are integrated into the part. In the old days, we would create a new resource even if the part was integrated into the CPU. We have decided, based on previous experience, that if a part is integrated into the CPU, we do not consider it a separate resource. Therefore, there are no separate directories for the north and south bridge. The code for these devices is supported in the CPU device. The LinuxBIOS codebase is flexible in this way. A given BIOS can be implemented with different types of parts, but in fact none of them are required.

Our first step in getting the resources set up for the mainboard is to name the CPU and set up the directory for it. The code for a given CPU is contained in the src/cpu directory. Luckily, the CPU in this case is an x86 system, so there is no need to add an architecture directory.

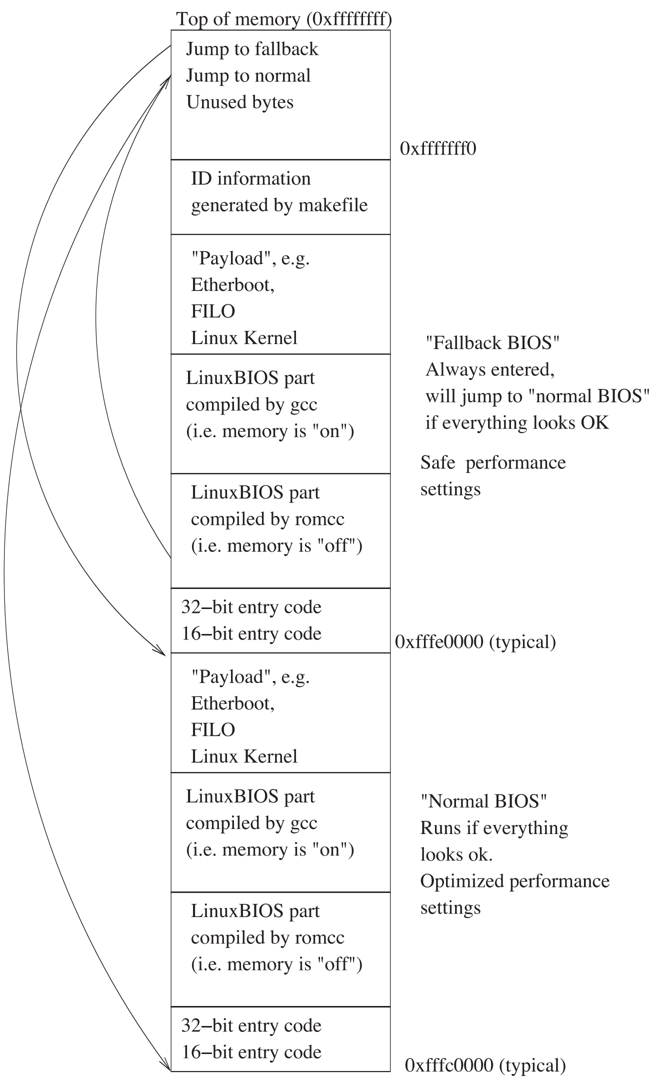


Figure 2. A typical ROM image includes a fallback BIOS to allow booting in case there is trouble with the main BIOS.

This article traces development from our point of view—a LinuxBIOS developer. If you want to develop a new tree, however, you can clone the LinuxBIOS arch repository, do development and submit patches to a developer. We will check your patches and help get them into the repository. In most cases with new developers, if their code is good, we allow them to become developers for our team.

CPU

We create a directory, src/cpu/amd/sc520, and populate it with files to support the CPU. We are not going to show all the commands for everything we do in this port, but for this first change, we show the commands to give you flavor of how it works. Even this simple part explains a lot of the important aspects of how LinuxBIOS is constructed:

```
cd src/cpu/amd
mkdir sc520
tla commit
```

This sets up the directory; now we need to populate it. The src/amd/socket_754 directory is a good candidate for providing model files, so we use them:

```
cd sc520
cp ../socket_754/* .
```

This gives us an initial set of files:

```
rminnich@q:~/src/freebios2/src/cpu/amd/sc520> ls
chip.h Config.lb socket_754.c
```

The chip.h file defines a simple data structure that is linked into the BIOS image by the Makefile, which is generated by the config tool. For this part, it's basically empty:

```
rminnich@q:~/src/freebios2/src/cpu/amd/sc520> catchip.h

extern struct chip_operations cpu_amd_socket_754_ops;

struct cpu_amd_socket_754_config {

};
```

What does this mean? First, we create an instance of a struct called chip_operations for this part, called cpu_amd_socket_754_ops. This is a generic structure, used by all chips. This generic structure looks like this:

```
/* Chip operations */

struct chip_operations {

        void (*enable_dev)(struct device *dev);

#if CONFIG_CHIP_NAME == 1

        char *name;

#endif

};
```

The chip_operations structure, in src/include/device/device.h, defines a generic method of accessing chips. It currently has two structure members: a function pointer to enable the device, enable_dev, and an optional name, used for debug prints, called name. Notice that in the style of the Linux kernel, C preprocessor-enabled code is controlled by testing the value of a preprocessor symbol, not by testing whether it is defined. As you can see, the enable_dev function takes a pointer to a device struct.

Why do we do this? Although there is one chip_operations structure for a type of chip, there is a device structure for each possible instance of a chip. We say possible because a device structure is defined for each chip that may exist in a system. Consider an SMP motherboard, which has from one to four or even eight CPUs; not all the CPUs may be there. Part of the job of the enable function is to determine whether the chip is even there.

The device struct looks like this:

```
struct device {
   struct bus * bus; /* bus this device is on, for
                      * bridge devices, it is the
                      * upstream bus */

   device_t sibling; /* next device on this bus */
   device_t next;    /* chain of all devices */
   struct device_path path;
   unsigned vendor;
   unsigned device;
   unsigned int class; /* 3 bytes:
                        * (base,sub,prog-if) */
   unsigned int hdr_type; /* PCI header type */
   unsigned int enabled : 1; /* set if we should
                              * enable the device */
   unsigned int initialized : 1;
    /* set if we have initialized the device */
   unsigned int have_resources : 1;
    /* Set if we have read the device's resources */
   unsigned int on_mainboard : 1;
   unsigned long rom_address;
   uint8_t command;

   /* Base registers for this device. I/O, MEM and
      Expansion ROM */

   struct resource resource[MAX_RESOURCES];
   unsigned int resources;

   /* links are (downstream) buses attached to the
    * device, usually a leaf device with no child
    * has 0 busses attached and a bridge has 1 bus */

   struct bus link[MAX_LINKS];

   /* number of buses attached to the device */
unsigned int links;

   struct device_operations *ops;
   struct chip_operations *chip_ops;
   void *chip_info;

};
```

This is a pretty complicated structure, and we don't go into all the issues here. During the configuration step, the LinuxBIOS configuration tool instantiates a struct device for each chip by writing C code to a file in the build directory. The C code that the config tool generates has initial values so that the array of device structures forms a tree, with sibling and child nodes. The LinuxBIOS hardwaremain() function walks this tree, starting at the root, and performs device probing and initialization.

The last structure member is a void *—that is, a pointer that can point to anything. The next-to-last element is a chip_operations pointer. As part of the creation of the initialized C structures, the config tool fills in the chip_info and chip_operations pointer with a pointer to the per-chip configuration structure and per-chip-type structure. Thus, each device in the tree has pointers to structures for the type of chip and the individual instance of the chip. The enable structure member, which is a function pointer, for the type of chip is called with a pointer to the structure for the device for each instance of the chip. The device structure has a lot of generic structure members, as you can see, and it has a pointer to a structure for non-generic chip components.

For each chip, we optionally can provide declarations of both structures, but it is not required. The chip_operations structure, or the type-of-chip structure, has a type fixed by LinuxBIOS itself; the chip_info structure has a structure fixed by the chip. The enable function in the chip_operations structure can be un-initialized, in which case there is no enable function to call for the chip—the chip is always enabled. That is the case for the SC520 CPU—there is only one, and it is always there.

Now we need to change these files to match the SC520. We show them before and after to give you an idea how it looks.

chip.h changes to look like this:

```
extern struct chip_operations cpu_amd_sc520_ops;

struct cpu_amd_sc520_config {

};
```

The enable_dev pointer is empty and is not called. We leave it empty for now but may fill it in later as needed. Similarly, there are no special structure members for the chip_info structure.

The C code looks like this:

```
#include <device/device.h>
#include "chip.h"

struct chip_operations cpu_amd_socket_754_ops =
  { CHIP_NAME("socket 754") };
```

The changes are simple; we rename the file to sc520.c and then change it to this:

```
#include <device/device.h>
#include "chip.h"

struct chip_operations cpu_amd_sc520_ops =
  { CHIP_NAME("AMD SC520") };
```

The final file is the Config.lb file. Here we get our first glance at what a configuration file looks like. The original file looks like this:

```
uses CONFIG_CHIP_NAME

if CONFIG_CHIP_NAME

        config chip.h

end

object socket_754.o

dir /cpu/amd/model_fxx
```

The first line declares that we are using the option CONFIG_CHIP_NAME. The language requires that we declare the variables we are going to use before we use them. In the case of this file that seems trivial, but in longer files this requirement is really useful. Second, if we are using the CONFIG_CHIP_NAME option, we use the chip.h file. Notice that nothing is set in chip.h unless we were using the CHIP_NAME macro, which is why this test is there. We declare any object files produced in this directory, in this case, socket_754. Finally, we include another directory using the dir keyword. The naming scheme in the config language for other directories is that the pathname is relative if it does not start with a /. Otherwise, it is rooted at the source of the LinuxBIOS source tree. In this case, the dir directive points to src/cpu/amd/model_fxx. As it happens, this is code for Opteron and is of no use to the SC520. After modifying this file for the SC520, it looks like this:

```
uses CONFIG_CHIP_NAME

if CONFIG_CHIP_NAME

        config chip.h

end

object sc520.o
```

That's about it. We've now set up support for the SC520.

Mainboard

Now we set up the mainboard. We first `cd` to mainboard/digitallogic and issue:

```
mkdir msm586seg
```

We then populate it from the adjacent adl855pc directory.

There are a lot of files here. We do not have enough space here to go into the changes for each file, but we can summarize what we do to each one.

auto.c

This file is compiled by romcc, and in a proprietary BIOS it would be a large blob of assembly code. To start, we completely empty this file—all it should have is a print function. This is the easiest way to get a new port going—make sure you have the ability to get some output. There is not room to show the whole file, but you can see it in the repository or use `viewarch`. There are two key things to get right, however. First is picking include files. For romcc, additional C code is not linked in; it is included. The include files look like this:

```
#define ASSEMBLY 1

#define ASM_CONSOLE_LOGLEVEL 8

#include <stdint.h>
#include <device/pci_def.h>
#include <arch/io.h>
#include <device/pnp_def.h>
#include <arch/romcc_io.h>
#include <arch/hlt.h>
#include "pc80/mc146818rtc_early.c"
#include "pc80/serial.c"
#include "arch/i386/lib/console.c"
#include "ram/ramtest.c"
#include "cpu/x86/mtrr/earlymtrr.c"
#include "cpu/x86/bist.h"
#include "cpu/amd/sc520/raminit.c"
```

For romcc, we define the ASSEMBLY value to 1. We also set the console log level for assembly to a very high level—8 in this case. LinuxBIOS uses macros for printing so that when a production BIOS is built, the debug print macros can be compiled out to save space. A console log level of 8 ensures that every print call is compiled.

Here's the main function, which does nothing at all:

```
static void main(unsigned long bist)

{
    print_err("Hello\n");
}
```

With this simple main we can test a lot. We can build the BIOS, load it and see if we get a printout. Simply getting print to work is a huge step in getting your BIOS going.

chip.h

We saw chip.h for a CPU; is it different for the mainboard? In fact, it's not really different at all:

```
extern struct chip_operations
     mainboard_digitallogic_msm586seg_ops;

struct mainboard_digitallogic_msm586seg_config {
};
```

As before, there is a generic chip_operations structure and a specialized structure for the chip, which in this case is a mainboard. Every single device in LinuxBIOS is treated the same way. This uniform structure has proven to be powerful.

cmos.layout

cmos.layout defines the structure of the CMOS memory, which is a battery-backed memory on the motherboard. We leave this unchanged for now.

Config.lb

Config.lb is pretty standard across platforms, so for reasons of space we show only a subset here, the part that is mainboard-specific. We are going to touch on a few highlights, but for more detail you need to study the full file in the archive.

driver mainboard.o

This statement declares a driver file, mainboard.o, which is included in the set of binaries linked in to the final image:

```
##
## Build our 16 bit and 32 bit linuxBIOS entry code
##

mainboardinit cpu/x86/16bit/entry16.inc
mainboardinit cpu/x86/32bit/entry32.inc
ldscript /cpu/x86/16bit/entry16.lds
ldscript /cpu/x86/32bit/entry32.lds
```

These commands relate to early initialization. The config tool builds a loader script for the BIOS, an assembly code file as well as a C file and Makefiles. The mainboardinit command tells the config tool to add the entry16.inc and entry32.inc assembly code files to the assembly code file for the mainboard. The .lds files are used in the ld script to determine how the assembly code is linked.

A number of mainboardinit and ldscript directives are in this file. These are architecture-related, for example, for the x86 architecture; CPU-related, for example, specific to the SC520 CPU; and, in some cases, mainboard-related.

Now we come to the complicated part of the file, which we are going to simplify for reasons of space:

```
chip cpu/amd/sc520
      device pci_domain 0 on
            device pci 0.0 on end
            device pci 1.0 on end
    end
end
```

We are declaring the CPU and the nested devices under that CPU. The first device is the PCI domain, domain 0, which is the only domain this CPU has. We declare device 0:0.0 and 0:0.1. That's it for now—this does get more complex later, however.

Some of these files are complex, in some cases running to 100 or more lines, as some boards are complicated.

failover.c

failover.c is included in auto.c and is the code for managing failover of the fallback BIOS image if the normal BIOS image is corrupted in some way.

irq_tables.c

PC hardware does not have a defined way of mapping PCI slot interrupt lines to interrupt pins on the interrupt controller. There is a structure in the BIOS called the $PIR structure that the operating system reads to find out how to map interrupts.

The irq_tables.c file has an initialized C structure that defines the connection of the interrupt lines. This structure is compiled into LinuxBIOS and forms the $PIR table.

This file is generated automatically by a utility provided with linuxbios, called getpir. It is found in util/getpir. You run this utility under Linux, when booted under the factory BIOS. The utility prints out the $PIR table as C code. One caveat: we have found that the $PIR tables on many BIOSes have errors. On occasion, we have had to fix the tables to correspond to the actual hardware.

mainboard.c

This code is compiled by GCC, not romcc. There is not much to this file right now:

```
#include <console/console.h>
#include <device/device.h>
#include <device/pci.h>
#include <device/pci_ids.h>
#include <device/pci_ops.h>
#include "chip.h"

struct chip_operations
mainboard_digitallogic_msm586seg_ops = {
  CHIP_NAME("Digital Logic MSM586SEG mainboard ")

};
```

Options.lb

This file contains the names of options used for this mainboard. First, all the options to be used are listed, for example:

```
uses HAVE_FALLBACK_BOOT
```

If the option has some desired value, it may be set in this file:

```
## Build code for the fallback boot
```

Appro *Xtreme*Blade Solution

Clusters. High performance, high availability.

The Most Powerful InfiniBand™ Enabled Blade Solution

Fits up to 12 Blades in a 7U Height Subrack

Analyst Firm IDC examines the Appro *Xtreme*Blade based on AMD Opteron™ processors to address HPC and Enterprise needs. In addition, IDC provides good information on technologies that are experiencing growth in adoption such as blade servers. Get this IDC White Paper at http://www.appro.com.

- **Scalable** - 2-way and/or 4-way configuration
- **Flexible** - Ability to mix blade configurations in one rack cabinet
- **Connected** - Integrated Infiniband™ and Gigabit switches
- **Powerful** - AMD Opteron™ based server with leading 32-bit performance with 64-bit capabilities
- **Reliable** - Hot swappable blades, redundant switches, power supplies and cooling fans
- **Smart** - BladeDome II – Centralized remote blade server and system management
- **Balanced Architecture** - Memory, I/O and communication bandwidth match CPU bandwidth

AMD Opteron™ Processors
- Integrated AMD HyperTransport™ technology allows for concurrent multiple processors in a single system.
- Shorten run-time cycles and increase bandwidth for processing computing requests.
- 32 bit applications while you migrate to 64 bit computing for long-term investment protection.

APPRO
HPC Cluster Solutions

Appro has everything you need to create a network blade cluster-ready.
On-site maintenance and installations services are also available.

For more information, please visit www.appro.com
or call Appro Sales at 800.927.5464, 408.941.8100.

AMD 64 Opteron™

```
default HAVE_FALLBACK_BOOT=1
```

which sets the option to 1. This option may be overridden in the target file; that is, we can set the following in targets/digitallogic/msm586seg/Config.lb:

```
option HAVE_FALLBACK_BOOT=1
```

and the BIOS can be built without a fallback boot image. In general, the default values set in this file do not need to be changed.

We do need to change the default ROM size, as it is set to 1024*1024 for the other mainboard:

```
default ROM_SIZE  = 256*1024
```

Why make this a default? So that a target with a larger ROM size can override it. If you build a target for a 1MB of ROM, you would put the command:

```
option ROM_SIZE = 256*1024
```

in the target configuration file.

reset.c

This file contains code to perform a hard reset of the CPU.

Target Configuration File

Now we add the target directory for the mainboard:

```
cd targets/digitallogic
mkdir msm586seg
tla add msm586seg
cp adl855pc/Config.lb msm586seg/
tla add Config.lb
```

We then commit, and the code is in. Next, we fix up the Config.lb for the msm586seg:

```
target msm586seg
mainboard digitallogic/msm586seg
option DEFAULT_CONSOLE_LOGLEVEL=10
option MAXIMUM_CONSOLE_LOGLEVEL=10
romimage "normal"
        option USE_FALLBACK_IMAGE=0
        option ROM_IMAGE_SIZE=0x10000
        option LINUXBIOS_EXTRA_VERSION=".0Normal"
        payload /etc/hosts
end

romimage "fallback"
        option USE_FALLBACK_IMAGE=1
        option ROM_IMAGE_SIZE=0x10000
        option LINUXBIOS_EXTRA_VERSION=".0Fallback"
        payload /etc/hosts

end

buildrom ./linuxbios.rom ROM_SIZE "normal" "fallback"
```

The file defines seven basic things:

1. The target build directory is msm586seg; it could be anything.

2. The mainboard is the digitallogic/msm586seg.

3. The default console log level is 10; this controls which compiled-in messages are printed. It can be overridden by the CMOS setting in the normal BIOS image.

HOW TO SET UP A LINUXBIOS PORT SYSTEM

We do not use Flash part burners at LANL, and most other places also do not. To burn a new Flash part, we actually pop the Flash part out of a running machine, put in a new part and run the flash_rom program to erase and rewrite the part. By far, the easiest way to set up a LinuxBIOS port station is to have one machine on which to build, one machine on which to burn and one machine on which to test.

The worst case is to have the burn, build and test machine be one and the same. In other words, the user has to boot the machine, build the LinuxBIOS, pop the Flash BIOS part out and put in a test part, burn it, reboot the machine to test and, in the likely event of failure—this is a new port, after all—put the factory BIOS back in and boot. The edit/compile/test cycle time can be long, as long as 3–5 minutes. In some cases, the burn and build machine can be the same.

For the SC520, we had a build machine, our x24 laptop; a burn machine, which is an MSM586SEG board; and a test machine, another MSM586SEG board. To simplify the situation further, we ran the two MSM586SEG boards as two bproc slave nodes using the Clustermatic software suite. Clustermatic lets us set up the two slave nodes with no local disk of any kind. All the state and control is managed from the laptop. We have been doing ports this way for five years now, and it is the easiest possible way we have found.

We've made a 64MB CompactFlash image available at the LinuxBIOS Wiki, so you can make a slave machine with no effort. For more details, see the Clustermatic Web site for instructions on how to set up a laptop as a master node.

4. The maximum console log level is 10; this controls which print macros are compiled.

5. The normal romimage is not a fallback image; it is 0x10000 bytes (64KB), has a version tag of .0Normal and has a payload of /etc/hosts.

6. The fallback romimage is a fallback image; it is 0x10000 bytes (64KB), has a version tag of .0Fallback and has a payload of /etc/hosts.

7. The ROM target is linuxbios.rom; it has a size of ROM_SIZE, as defined in the mainboard Options.lb above, and has two images in it, normal and fallback.

Shoot the Dice and Wear a Blindfold

Well, let's see how it goes. We have a script for this part, to save some typing:

```
cd src/targets

./buildtarget digitallogic/msm586seg
```

This step works. It builds, but we get errors, which is expected. The version covered above, by the way, is:

```
linuxbios@linuxbios.org--devel/freebios--devel--2.0--patch-21
```

if you want to see what goes wrong. With a few modifications, we get a working version, which is stored at:

```
linuxbios@linuxbios.org--devel/freebios--devel--2.0--patch-22
```

It builds! The next step is to see if we can get any serial output. Make sure, of course, that you place the Flash part you want to burn into the Flash socket or you're going to be pretty unhappy. Better yet, *before* you start burning, make a backup of your factory BIOS to cover for mistakes:

```
flash_rom -r /tmp/backup
```

Put in a new Flash part:

```
flash_rom /tmp/backup
```

and store the Flash part somewhere safe.

We're building on a laptop and using an SC520 running Linux as the burner node. So use:

```
scp linuxbios.rom root@burnnode:
ssh root@burnnode flash_rom linuxbios.rom
```

Did It Work?

Let's find out if it worked. Be sure to follow our progress on the *Linux Journal* Web site.

Next Steps

You can track our progress on the Web page or the LinuxBIOS Wiki (see the on-line Resources)—we have set up a status page there so you can see how it is going.

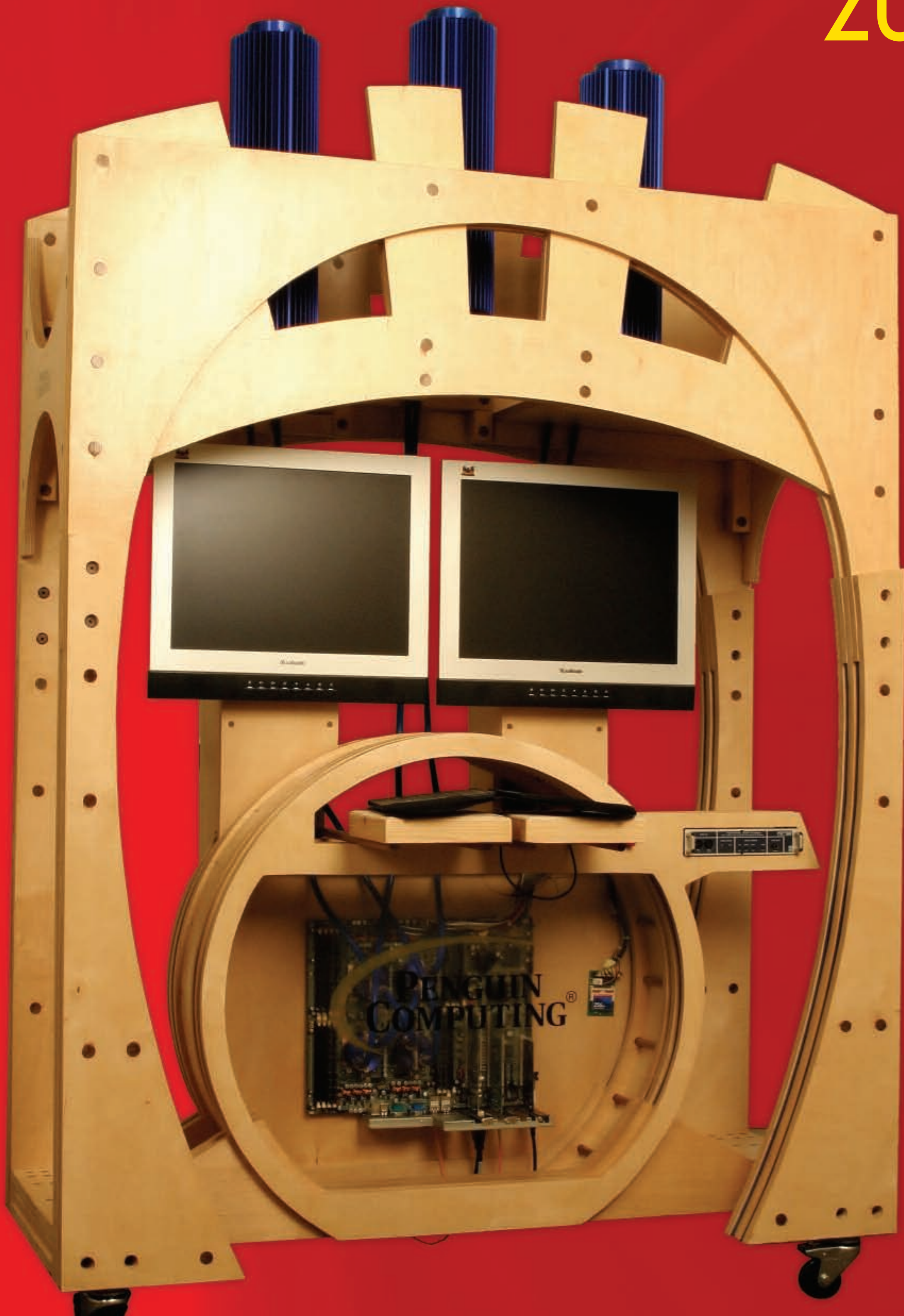
We have tried to show you a quick overview of how to do a LinuxBIOS port to a new system. If you really want to give it a go, join the mailing list and tell people what you are doing. There's a lot of expertise out there, and people are ready to help. For the record, it took one person totally unfamiliar with this system four hours to build a new BIOS port from scratch. That's not bad. Although it looks rather complex, once you see how to build a BIOS, you probably will find it to be pretty easy.

This research was funded in part by the Mathematical Information and Computer Sciences (MICS) Program of the DOE Office of Science and the Los Alamos Computer Science Institute (ASCI Institutes). Los Alamos National Laboratory is operated by the University of California for the National Nuclear Security Administration of the United States Department of Energy under contract W-7405-ENG-36. Los Alamos, NM 87545 LANL LA-UR-05-3336.

Resources for this article: www.linuxjournal.com/article/8327.

Ron Minnich is the team leader of the Cluster Research Team at Los Alamos National Laboratory. He has worked in cluster computing for longer than he would like to think about.

ULTIMATE LINUX BOX 2005



Some people wanted us to build a big powerful SMP system.

Let’s just call 2005 the year of power management. Processor vendors made a big deal out of whitepapers about saving watts, and we heard a lot about power management at LinuxWorld Conference and Expo in February.

Did the industry start caring about global warming? Do IT CEOs want to eat swordfish more often, so they have to reduce the mercury emissions of power plants? Not quite. Today’s server systems are packing more and hotter processors closer together, and customers’ air-conditioning systems aren’t ready for the strain. NASA had to install water cooling for its 10,240-processor Columbia cluster, as we showed in our January issue.

Every watt-hour you can save is heat that the customer doesn’t have to deal with—3.6kJ, or 3.4BTU to be precise. With data centers full of blade servers, and 1U systems sporting as many as four processors, all that heat really adds up.

The Linux desktop greedily devours the scraps from the multibillion-dollar Linux server market, and power consumption matters to us on the desktop too. Fans are loud. If you have better power management on your processors, they produce less heat, and you can run fewer fans or run the fans you do have more quietly. We took a different approach to fans, as you’ll see later on.

Finally, of course, power matters on the laptop and on portable devices because of battery life. We’ll leave the specifics of tweaking for maximum off-AC time to future articles.

Listing 1. Partition scheme as seen in /etc/fstab.

```
LABEL=/nstor-OS   /         ext3   defaults         1 1
LABEL=/cfboot     /boot     ext3   defaults         1 2
LABEL=/nstor-DATA /u1       ext2   defaults         1 2
none              /dev/pts  devpts gid=5,mode=620   0 0
none              /dev/shm  tmpfs  defaults         0 0
none              /proc     proc   defaults         0 0
none              /sys      sysfs  defaults         0 0
```

ULTIMATE LINUX BOX 2005 PARTS LIST

Motherboard: Tyan Thunder K8QS Pro (S4882)

CPUs: 4 x AMD 846HE Opterons

RAM: 8 x 4GB Registered ECC Samsung DDR PC2700 CL 2.5 DIMMs

Power supply: 510W Custom harness PC Power and Cooling Turbo-Cool 510 ATX (modified)

Case: Custom, designed by Matt Fulvio, constructed by Trevor Sherard

Fibre Channel: QLogic 2342 dual-port, 133MHz, PCI-X, 2Gb Fibre Channel adapter

Boot device: Sandisk 256MB CompactFlash card, DCFB-256-A10 with altec 30AL2051 CompactFlash-IDE adapter

Storage: nStor 4320F Fibre Channel RAID enclosure

Hard disks: 2 x 18GB Hitachi DK32DJ-18FC 10KRPM Fibre Channel drives in a RAID 1 array (OS install) and 6 x 73Gb Seagate ST373405FC Cheetah 73LP FC 10KRPM Fibre Channel drives in a RAID 10 array

Graphics card: PNY NVIDIA Quadro NVS 280 PCI

Displays: 2 x ViewSonic VX2000 20" 1600x1200 LCD displays

Audio card: RME HDSP9652 PCI audio card

Audio I/O: RME Multiface 36-channel 24-bit 96-kHz I/O box

Cooling system: 3 x Zalman Reserator 1s

CPU waterblocks: 4 x Zalman ZM-

Motherboard: the Heart of the System

We like Tyan motherboards, and companies that build custom Linux systems do too. The four-Opteron Tyan Thunder K8QS Pro came out just a little too late to make it into last year's Ultimate Linux Box. It's based on an AMD 8000 series chipset. When we say "chipset", we mean a slightly different combination of hardware from an Intel-based system, though. The AMD64 way is to have an onboard memory controller per processor, give each processor its own bank of memory and link them with HyperTransport. Your AMD64 "SMP" box is really a mini-NUMA, and the "chipset" doesn't include the memory controller.

Last year, we used a Celestica A8440 bare-bones rackmount system as the basis for the Ultimate Linux Box. Although starting with pre-integrated chassis and power supplies can be a great time saver, we realized that last year's box was on the loud side. This year, going back to our usual plan lets us pick everything else just the way we want it.

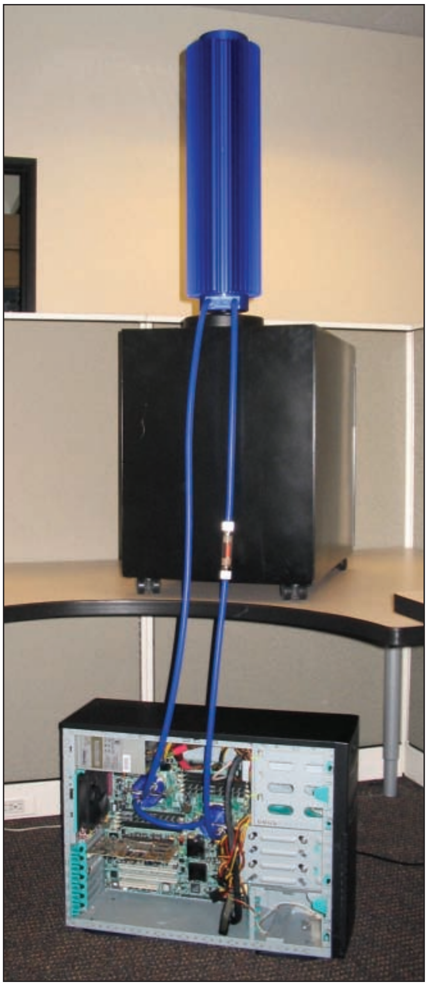
The K8QS Pro has two PCI-X busses, A and B. B is dedicated to two 133MHz-capable PCI-X slots, and A offers two PCI-X slots maxing out at 66MHz and one regular PCI slot. Onboard networking is two Broadcom BCM5704C Gigabit Ethernet interfaces, also on bus A.

There are all the regular PC ports, of which we're using the USB. SCSI and serial ATA are options, which you might want to keep in mind if you're planning to move this board into a more conventional server role when you're building your next Ultimate Linux Box.

Into this mighty board we plugged four of the best of the Opteron processors available at the time—the 846 HE, clocked at 2.0GHz and offering 1MB of L2 cache. See the sidebar for what became available while we were testing the system. We maxed out the system's main memory at 32GB.

Unfortunately for case shoppers, this board is SSI MEB size—13"×16" or 330.2×406.4mm. Not a problem for us because we're using a custom case this year, but the size does limit your case options.

When we're picking out a case for any custom-built system, Ultimate or otherwise, we usually get one that's quite a bit larger than what a big vendor would use for a comparable system. Smaller cases require less material and they're cheaper for vendors to ship, but since we like to tweak things, we get a case with more room to add devices and more room to work inside.


What's that in your cubicle, Justin? We tested convective cooling with a scratch system and lm_sensors.

Storage

In order to have a completely silent system, you need to move storage outside the box. Options for doing this have changed a lot since the days when you had a choice between NFS and external SCSI enclosures connected by a 3-meter cable.

Today, you can make your drives go away using USB, FireWire, SCSI of course, Fibre Channel or the new ATA over Ethernet, which we covered in the

June 2005 issue. A separate storage enclosure is no longer only an enterprise server-room thing.

Another option is simply to boot over the network and mount your storage via NFS. Since Penguin works with enterprise server-room hardware, and Fibre Channel does deliver impressive benchmark results, we went with it; an nStor 4320F Fibre Channel RAID enclosure, with Hitachi 18GB drives for the OS and larger Seagate drives for more storage.

Because we wanted the system to be self-contained and not depend on another server to boot, we installed a Sandisk 256MB CompactFlash card to boot from. This device looks exactly like another ATA drive to the system, so any PC motherboard will boot from it.

We considered using a USB thumbdrive, but that would have required some initrd drive juggling and GRUB wizardry. There are advantages to being able to pull your boot device out of the system and store it separately, but we didn't anticipate shipping the system through airports with drives loaded with encrypted confidential data.

If you plan to leave your silent Linux system on your network, you'll be a little more flexible in booting, and you can set up PXE booting. But if you want to take your Ultimate Linux Box over to a friend's house to play some music, you'll want to be able to boot independently. The Penguin crew plans to take this system to LinuxWorld Conference and Expo, and when you're wrangling hardware for a tradeshow one fewer thing to set up is good.

If you do build and install a silent Linux box, you'll probably end up doing a mix of both: NFS for user home directories, the company /usr/local/bin/ and other items that need to be in sync but aren't performance-critical. You can save your machine's own filesystems for big working files, like all the audio data you'll get from this system's high-end sound hardware.

Finally, to take even the keyboard clicking out of the silent system, Penguin founder Sam Ockman suggested a TouchStream LP keyboard, which works like a touchpad and requires no moving parts. It's also a pointing device and lets you map gestures to interface actions.

Audio

For the first time, we put professional audio hardware into the Ultimate Linux Box. What better place for a silent machine than the recording studio?

The RME Hammerfall HDSP9652 card we chose for this system is capable of up to 52 channels, and we matched it with an external box called the Multiface that brings out 8 1/4" jacks, as well as optical, coax and MIDI.

This card is as close as you can get to a "studio in a box", because it's built around an internal mixer and allows you to route signals around inside the card with low latencies and low load on the CPU. Other features include the ability to "punch in" and "punch out" like a conventional

HARDWARE OF THE FUTURE, LAWYERS STUCK IN THE PAST

It never fails. New products that we'd like to try in the Ultimate Linux Box come out right when we're in the middle of building this year's.

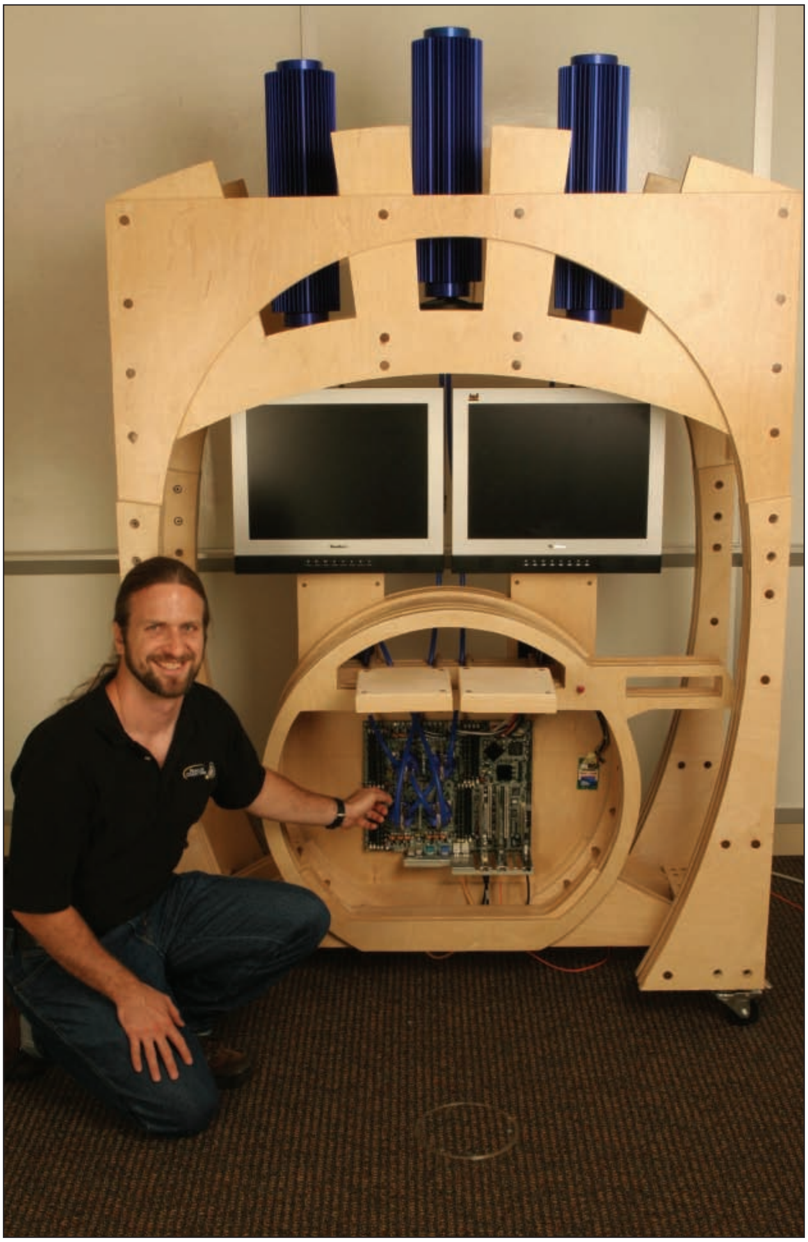
Too late to make it through our thermal testing, AMD introduced dual-core Opteron processors, which let you build an eight-way system on an existing four-socket motherboard with a BIOS upgrade. Today, that means spending $10,000 on processors, but (all together now) we expect prices to come down.

We're watching the progress of the LinuxBIOS Project (see page 32) and are planning to get a supported motherboard for next year. We know patience is a virtue, but booting in mere seconds is cool for its own sake.

This year's system sounded so nice that we'd like to do another quiet machine next year. That means we have to pick a storage technology, and added to next year's list of alternatives will be ATA over Ethernet, as covered in Ed Cashin's article in the June 2005 issue.

Video is still a weak spot, not because of hardware problems, but because of the vendors' lawyers. Everybody doing 3-D is infringing everyone else's patents, and burying the driver code behind a proprietary EULA with a no-reverse-engineering clause is only slowing the industry down. When the normal kernel development process frequently breaks the driver for commonly used hardware, that hardware needs to get with the program.

Graphics vendors, please get together, cross-license the patents for hardware, and come up with a license for software and documentation that lets developers release the new code that makes people want graphics hardware in the first place. It'll help everyone in the long run—NVIDIA maintains an entire parallel software distribution system just because of its licensing decision. Why not get that cost center out of the budget?



Look, everybody, no leaks! Justin sets up for the cover photo shoot (photo: Don Cameron).

tape deck.

Best of all, RME has been supporting the Advanced Linux Sound Architecture (ALSA) Project since 2000, so Linux users aren't second-class citizens. RME's site says, "ALSA support for the Hammerfall breaks the annoying chicken/egg principle—no professional hardware/driver, no professional software."

Peter Todd covered the necessary tools for working with the Hammerfall HDSP cards in our October 2003 issue.

For video, we used a relatively low-end card (see the on-line Resources). We'd really like to start putting interesting and innovative video on Ultimate Linux Boxes, but there are still some issues with the drivers (see sidebar).

Thermal Management

So how do we keep this thing cool? First of all, it's important not to start tweaking with hardware combinations unless you know how to measure the effects that your changes have on the system's temperature.

Don't change anything unless you know how to measure the effect of the change.

The good news is that the processor and motherboard vendors thoughtfully give us temperature sensors right on the key parts. And we can keep track of them using an all-important tool, lm_sensors.

We didn't have to measure drive temperature because we moved the drives to a separate enclosure, but smartmontools (see Resources) gives you an easy way to do that.

We ordered up some parts from Zalman, which offers a beautiful set of water-cooling hardware. The most visible part is the Reserator 1, a combined water reservoir and radiator that stands a half-meter tall and holds 2.5 liters of water. Besides the Reserator, we also ordered one CPU waterblock per processor and matching tubing.

Thermal estimates showed that we wouldn't need a full Reserator per processor, so we used one Reserator per two processors and one for the power supply.

The Reserator comes with a 5W pump, which would break our beautiful silence, so it was time to convert it to operate purely by convection. In its stock configuration, the Reserator's inlet and outlet are close to each other, so we installed a tube inside each Reserator, running from the hot inlet to near the top.

Did it work? The processor temperature climbed to about 50ºC, then the tubes leading up from the processors to the Reserators warmed enough to start the convection. Temperature fell to 47º or 48ºC in normal use, and running full-out, the system holds out below 50ºC.

Cooling the power supply was a little harder. Zalman's beefiest fanless power supply is only 400W, and a big four-way board needs more. We decided to use the PC Power and Cooling Turbo-Cool 510 ATX.

We decided not to design and build a power supply for the project, since it's important to apply power to components in the right order, and we know PC Power and Cooling solved that problem for us. The cooling problem remained.

Enter the magic of metalworking. Phil brought the problem to a machine shop called Global Precision, and we had them do three pieces of work. They machined down the original fins of the power supply's heatsinks to create flat areas for attaching waterblocks. They made the waterblocks themselves—using blue anodized aluminum to match the Zalman parts. And they made two custom Y-connectors to split the water flow between the two heatsinks.

We removed the fan control board from the power supply. We didn't need it anymore.

Case

Cases capable of accommodating and doing justice to Ultimate Linux Boxes are rare. This year, only one alternative would work: going full custom. This year's case has acrylic windows to show off the cooling system, integrated supports for the three Reserators and a mounting place for the RME Multiface.

Conclusion

Difficult as it might be for us to believe right now, many real-world systems don't need both 52-channel audio and Fibre Channel. But unusual combinations of hardware are what enable creative projects, and we're happy that Linux stays out of our way and lets us hook up what we want.

When you start with what's possible and take out what you don't need, you'll be confident that you can build a machine for your needs. We hope that whatever class of system you decide to build, you'll get some ideas out of this year's Ultimate Linux Box.

Resources for this article: www.linuxjournal.com/article/8330.

Justin Thiessen is a Linux Engineer at Penguin Computing. As head of this year's Ultimate Linux Box Project, he was responsible for system design, construction and testing, and was involved in component selection. When not busy with the Ultimate Linux Box, he works on new product development and improving Linux support for Penguin hardware by contributing to the lm_sensors Project.

Matt Fulvio is a freelance industrial and architectural designer in the Bay Area. He can be found teaching mathematics at the San

---

BENCHMARK RESULTS

dbench with 100 simulated clients:

```
%dbench 100
Throughput 1234.57 MB/sec (NB=1543.21 MB/sec 12345.7 MBit/sec)
```

Bonnie++ 1.03—a more accurate disk benchmark:

- Sequential output by character: 58,577Kb/s, 98% CPU
- Sequential output by block: 281,032Kb/s, 50% CPU
- Sequential output, rewrite: 52,603Kb/s, 18% CPU
- Sequential input by character: 34,717Kb/s, 58% CPU
- Sequential input by block: 90,097Kb/s, 11% CPU
- Random seeks: 257.5/s
- Sequential create: 5,924 files/s
- Random create: 6,056 files/s

Postmark benchmark:

Postmark simulates the operations of a busy mail server. For 20,000 base files and 100,000 transactions, we obtained the following results.

Time:

- 46 seconds total
- 40 seconds of transactions (2,500/s)

Files:

- 70,128 created (1,524/s); creation alone: 20,000 files (5,000/s); mixed with transactions: 50,128 files (1,253/s)
- 49,656 read (1,241/s)
- 50,199 appended (1,254/s)
- 70,128 deleted (1,524/s)
- Deletion alone: 20,256 files (10,128/s); mixed with transactions: 49,872 files (1,246/s)

Data:

- 303.46MB read (6.60MB/s)

Memory Ordering in Modern Microprocessors, Part I

One important difference among CPU families is how they allow memory accesses to be reordered. Linux has to support them all.

BY PAUL E. MCKENNEY

Since the 2.0 kernel release, Linux has supported a large number of SMP systems based on a variety of CPUs. Linux has done an excellent job of abstracting differences among these CPUs, even in kernel code. This article is an overview of one important difference: how CPUs allow memory accesses to be reordered in SMP systems.

Memory accesses are among the slowest of a CPU's operations, due to the fact that Moore's Law has increased CPU instruction performance at a much greater rate than it has increased memory performance. This difference in performance increase means that memory operations have been getting increasingly expensive compared to simple register-to-register instructions. Modern CPUs sport increasingly large caches in order to reduce the overhead of these expensive memory accesses.

These caches can be thought of as simple hardware hash tables with fixed-size buckets and no chaining, as shown in Figure 1. This cache has 16 lines and two ways for a total of 32 entries, each entry containing a single 256-byte cache line, which is a 256-byte-aligned block of memory. This cache line size is a little on the large size, but it makes the hexadecimal arithmetic much simpler. In hardware parlance, this is a two-way set-associative cache. It is analogous to a software hash table with 16 buckets, where each bucket's hash chain is limited to two elements at most. Because this cache is implemented in hardware, the hash function is extremely simple: extract four bits from the memory address.

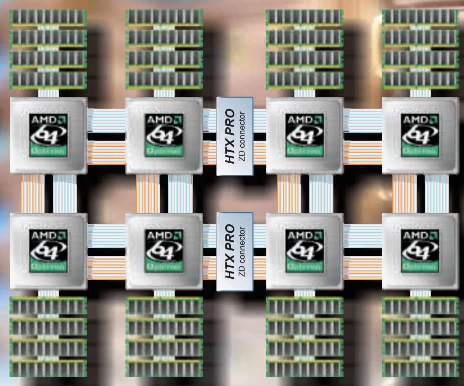
In Figure 1, each box corresponds to a cache entry that can contain a 256-byte cache line. However, a cache entry can be empty, as indicated by the empty boxes in the figure. The rest of the boxes are flagged with the memory address of the cache line they contain. Because the cache lines must be 256-byte aligned, the low eight bits of each address are zero. The choice of hardware hash function means the next-higher four bits match the line number.

The situation depicted in Figure 1 might arise if the program's code was located at address 0x43210E00 through 0x43210EFF, and this program accessed data sequentially from 0x12345000 through 0x12345EFF. Suppose that the program now was to access location 0x12345F00. This location hashes to line 0xF, and both ways of this line are empty, so the corre-

	Way 0	Way 1
0x0	0x12345000	
0x1	0x12345100	
0x2	0x12345200	
0x3	0x12345300	
0x4	0x12345400	
0x5	0x12345500	
0x6	0x12345600	
0x7	0x12345700	
0x8	0x12345800	
0x9	0x12345900	
0xA	0x12345A00	
0xB	0x12345B00	
0xC	0x12345C00	
0xD	0x12345D00	
0xE	0x12345E00	0x43210E00
0xF		

Figure 1. CPU Cache Structure for a Cache with 16 Lines and Two Entries Per Line

8 Opteron sockets
IWILL H8501 Barebone System



IWILL H8501 ▶

- 8x AMD Opteron Processor 940 sockets
- Supports 800 series Opteron CPUs with dual core tech.
- Up to 128GB DDR Registered ECC memory
- Support 4 Ranks memory module
- 1350W Redundant PSU 3+1
- Support IPMI server management
- Industry 19" rack-mountable 5U chassis
- 4 x Gigabit Ethernet ports, and 4 PCI-X slots
- Up to 10 hot-swap HDDs with option HDD canister
- Modularization design, I/O may vary

8-Way AMD Opteron Server Benchmark Rating
SPEC CPU2000: COMPILERS COMPARED

SPECint_rate_base2000

GCC: V3.3.4 Suse 9.2	97.4
GCC: V3.3.3 SLES9	93.7
GCC: V3.4.3 SLES9	95.5
PGI: V5.2.4, Suse 9.2	76.2
PGI: V5.2-4 SLES9	76.4
Pathscale: 2.0 Suse 9.2	101.0
Pathscale: 2.0 Suse 9	102.0
Intel: V8.1 Suse 9.2	101.0
Intel: V8.1 Suse 9	99.6

SPECfp_rate_base2000

GCC: V3.3.4 Suse 9.2	80.5
GCC: V3.3.3 SLES9	79.8
GCC: V3.4.3 SLES9	82.1
PGI: V5.2.4, Suse 9.2	81.6
PGI: V5.2-4 SLES9	84.1
Pathscale: 2.0 Suse 9.2	91.0
Pathscale: 2.0 Suse 9	94.2
Intel: V8.1 Suse 9.2	88.2
Intel: V8.1 Suse 9	84.9

IWILL Other Outstanding Barebone Systems:

▼ H4203

- 4 AMD Opteron Processor 940 sockets
- Supports 8xx Opteron CPUs with dual core tech.
- Up to 64GB DDR Registered ECC memory
- Support 4 Ranks memory module
- Support IPMI server management
- Industry 19" rack-mountable 2U chassis
- 4 x Gigabit Ethernet ports via PCI-X interface
- Modularization design, I/O may vary

▼ H2B Blade Server

- 2 AMD Opteron Processor 940 sockets
- Supports 2xx Opteron CPUs with dual core tech.
- Up to 16GB DDR Registered ECC memory
- Power distribution backplane in subrack
- 8U height, 10 blades subrack
- 2 x Gigabit Ethernet ports, one PCI-X slot
- Support IPMI server management (Option)

▼ H2103

- 2 AMD Opteron Processor 940 sockets
- Supports 2xx Opteron CPUs with dual core tech.
- Up to 16GB DDR Registered ECC memory
- Support 4 Ranks memory module
- Support IPMI server management
- Industry 19" rack-mountable 1U chassis
- 2 x Gigabit Ethernet ports via PCI-E interface

▼ ZMAX-DP

- 2 AMD Opteron Processor 940 sockets
- Dual processors Small Form Factor
- Supports 2xx Opteron CPUs with dual core tech.
- Up to 4GB DDR Registered ECC memory
- 1x AGP 8X , 1x GbE, 1x PCI and 1x mini PCI slot
- 3x 3.5" HDD bays, and 1x 5.25" CD-ROM bay
- 1x IEEE1394, 8x USB 2.0 ports
- 300W Power supply

IWILL USA Corp.
9004 Research Drive
Irvine, CA 92618
Tel: +1 949 753-5488
Fax: +1 949 753-5499

Visit www.iwill.net for more information.
Or contact us: sales@iwillusa.com, oem@iwillusa.com

Iwill®

sponding 256-byte line can be accommodated. If the program was to access location 0x1233000, which hashes to line 0x0, the corresponding 256-byte cache line can be accommodated in way 1. However, if the program were to access location 0x1233E00, which hashes to line 0xE, one of the existing lines must be ejected from the cache to make room for the new cache line. This background on hardware caching allows us to look at why CPUs reorder memory accesses.

Why Reorder Memory Accesses?

In a word, performance! CPUs have become so fast that the large multimegabyte caches cannot keep up with them. Therefore, caches often are partitioned into nearly independent banks, as shown in Figure 2. This allows each of the banks to run in parallel, thus keeping up better with the CPU. Memory normally is divided among the cache banks by address. For example, all the even-numbered cache lines might be processed by bank 0 and all of the odd-numbered cache lines by bank 1.

However, this hardware parallelism has a dark side: memory operations now can complete out of order, which can result in some confusion, as illustrated in Figure 3. CPU 0 might write first to location 0x12345000, an even-numbered cache line, and then to location 0x12345100, an odd-numbered cache line. If bank 0 is busy with earlier requests but bank 1 is idle, the first write is visible to CPU 1 after the second write. In other words, the writes are perceived out of order by CPU 1. Reads can be reordered in a similar manner. This reordering can cause many textbook parallel algorithms to fail.

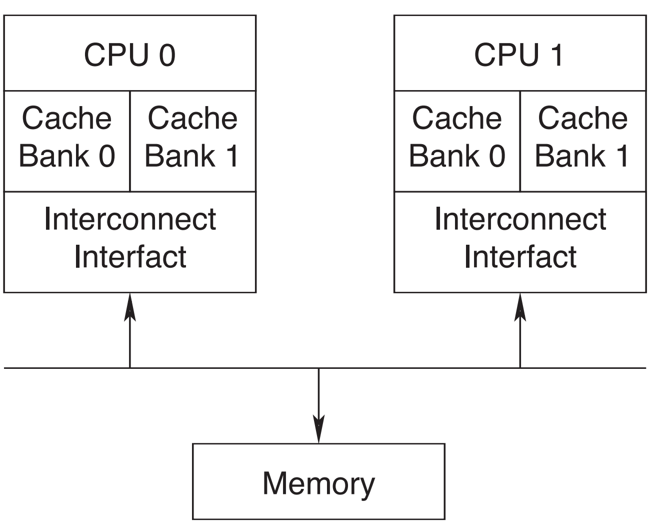


Figure 2. Hardware parallelism divides one large cache into multiple banks.

Memory Reordering and SMP Software

A few machines offer sequential consistency, in which all operations happen in the order specified by the code and where all CPUs' views of these operations are consistent with a global ordering of the combined operations. Sequentially consistent systems have some nice properties, but high performance does not tend to be one of them. The need for global ordering severely constrains the hardware's ability to exploit parallelism, and therefore, commodity CPUs and systems do not offer sequential consistency.

On these systems, three orderings must be accounted for:

1. Program order: the order in which the memory operations are specified in the code running on a given CPU.

2. Execution order: the order in which the individual memory-reference instructions are executed on a given CPU. The execution order can differ from program order due to both compiler and CPU-implementation optimizations.

3. Perceived order: the order in which a given CPU perceives its and other CPUs' memory operations. The perceived order can differ from the execution order due to caching, interconnect and memory-system optimizations. Different CPUs might well perceive the same memory operations as occurring in different orders.

Figure 3. CPUs can do things out of order.

Popular memory-consistency models include x86's process consistency, in which writes from a given CPU are seen in order by all CPUs, and weak consistency, which permits arbitrary reorderings limited only by explicit memory-barrier instructions. For more information on memory-consistency models, see Gharachorloo's exhaustive technical report, listed in the on-line Resources.

Summary of Memory Ordering

When it comes to how memory ordering works on different CPUs, there is good news and bad news. The bad news is each CPU's memory ordering is a bit different. The good news is you can count on a few things:

1. A given CPU always perceives its own memory operations as occurring in program order. That is, memory-reordering issues arise only when a CPU is observing other CPUs' memory operations.

2. An operation is reordered with a store only if the operation accesses a different location than does the store.

3. Aligned simple loads and stores are atomic.

4. Linux-kernel synchronization primitives contain any

needed memory barriers, which is a good reason to use these primitives.

The most important differences are called out in Table 1. More detailed descriptions of specific CPUs' features will be addressed in a later installment. Parenthesized CPU names indicate modes that are allowed architecturally but rarely used in practice. The cells marked with a Y indicate weak memory ordering; the more Ys, the more reordering is possible. In general, it is easier to port SMP code from a CPU with many Ys to a CPU with fewer Ys, though your mileage may vary. However, code that uses standard synchronization primitives—spinlocks, semaphores, RCU—should not need explicit memory barriers, because any required barriers already are present in these primitives. Only tricky code that bypasses these synchronization primitives needs barriers. It is important to note that most atomic operations, for example, atomic_inc() and atomic_add(), do not include any memory barriers.

	Loads Reordered After Loads?	Loads Reordered After Stores?	Stores Reordered After Stores?	Stores Reordered After Loads?	Atomic Instructions Reordered With Loads?	Atomic Instructions Reordered With Stores?	Dependent Loads Reordered?	Incoherent Instruction Cache/Pipeline?
Alpha	Y	Y	Y	Y	Y	Y	Y	Y
AMD64	Y			Y				
IA64	Y	Y	Y	Y	Y	Y		Y
(PA–RISC)	Y	Y	Y	Y				
PA–RISC CPUs								
POWER	Y	Y	Y	Y	Y	Y		Y
SPARC RMO	Y	Y	Y	Y	Y	Y		Y
(SPARC PSO)			Y	Y		Y		Y
SPARC TSO				Y				Y
x86	Y	Y		Y				Y
(x86 OOStore)	Y	Y	Y	Y				Y
zSeries				Y				Y

Table 1. Summary of Memory Ordering

In Table 1, the first four columns indicate whether a given CPU allows the four possible combinations of loads and stores to be reordered. The next two columns indicate whether a given CPU allows loads and stores to be reordered with atomic instructions. With only eight CPUs, we have five different combinations of load-store reorderings and three of the four possible atomic-instruction reorderings.

The second-to-last column, dependent reads reordered, requires some explanation, which will be undertaken in the second installment of this series. The short version is Alpha requires memory barriers for readers as well as for updaters of linked data structures. Yes, this does mean that Alpha in effect can fetch the data pointed to before it fetches the pointer itself—strange but true. Please see the "Ask the Wizard" column on the manufacturer's site, listed in Resources, if you think that I am making this up. The benefit of this extremely weak memory model is Alpha can use simpler cache hardware, which in turn permitted higher clock frequencies in Alpha's heyday.

The last column in Table 1 indicates whether a given CPU has an incoherent instruction cache and pipeline. Such CPUs require that special instructions be executed for self-modifying code. In absence of these instructions, the CPU might execute the old rather than the new version of the code. This might seem unimportant—after all, who writes self-modifying code these days? The answer is that every JIT out there does. Writers of JIT code generators for such CPUs must take special care to flush instruction caches and pipelines before attempting to execute any newly generated code. These CPUs also require that the exec() and page-fault code flush the instruction caches and pipelines before attempting to execute any binaries just read into memory, lest the CPU ends up executing the prior contents of the affected pages.

How Linux Copes

One of Linux's great advantages is it runs on a wide variety of different CPUs. Unfortunately, as we have seen, these CPUs sport a wide variety of memory-consistency models. So what is a portable kernel to do?

Linux provides a carefully chosen set of memory-barrier primitives, as follows:

- smp_mb(): "memory barrier" that orders both loads and stores. This means loads and stores preceding the memory barrier are committed to memory before any loads and stores following the memory barrier.
- smp_rmb(): "read memory barrier" that orders only loads.
- smp_wmb(): "write memory barrier" that orders only stores.
- smp_read_barrier_depends(): forces subsequent operations that depend on prior operations to be ordered. This primitive is a no-op on all platforms except Alpha.

The smp_mb(), smp_rmb() and smp_wmb() primitives also force the compiler to eschew any optimizations that would have the effect of reordering memory optimizations across the barriers. The smp_read_barrier_depends() primitive must do the same, but only on Alpha CPUs.

These primitives generate code only in SMP kernels; however, each also has a UP version—mb(), rmb(), wmb() and read_barrier_depends(), respectively—that generate a memory barrier even in UP kernels. The smp_ versions should be used in most cases. However, these latter primitives are useful when writing drivers, because memory-mapped I/O accesses must remain ordered even in UP kernels. In absence of memory-barrier instructions, both CPUs and compilers happily would rearrange these accesses. At best, this would make the device act strangely; at worst, it would crash your kernel or, in some cases, even damage your hardware.

So most kernel programmers need not worry about the memory-barrier peculiarities of each and every CPU, as long as they stick to these memory-barrier interfaces. If you are working deep in a given CPU's architecture-specific code, of course, all bets are off.

But it gets better. All of Linux's locking primitives, including spinlocks, reader-writer locks, semaphores and read-copy updates (RCUs), include any needed barrier primitives. So if you are working with code that uses these primitives, you don't even need to worry about Linux's memory-ordering primitives. That said, deep knowledge of each CPU's memory-consistency model can be helpful when debugging, to say nothing of writing architecture-specific code or synchronization primitives.

Besides, they say a little knowledge is a dangerous thing. Just imagine the damage you could do with a lot of knowledge! For those who want to understand more about individual CPUs' memory consistency models, the next installment will describe those of the most popular and prominent CPUs.

Conclusions

As noted earlier, the good news is Linux's memory-ordering primitives and synchronization primitives make it unnecessary for most Linux kernel hackers to worry about memory barriers. This is especially good news given the large number of CPUs and systems that Linux supports and the resulting wide variety of memory-consistency models. However, there are times when knowing about memory barriers can be helpful, and I hope that this article has served as a good introduction to them.

Acknowledgements

I owe thanks to many CPU architects for patiently explaining the instruction- and memory-reordering features of their CPUs, particularly Wayne Cardoza, Ed Silha, Anton Blanchard, Tim Slegel, Juergen Probst, Ingo Adlung and Ravi Arimilli. Wayne deserves special thanks for his patience in explaining Alpha's reordering of dependent loads, a lesson that I resisted learning quite strenuously!

Resources for this article: www.linuxjournal.com/article/8331.

Paul E. McKenney is a Distinguished Engineer with IBM's Linux Technology Center. He has worked on NUMA and SMP algorithms and, in particular, RCU for longer than he cares to admit. In his spare time, he jogs and supports the usual house-wife-and-kids habit.

A User's Guide to ALSA

Your Linux system's sound probably just came up and worked, which is great for games, chat or music listening. But with a little exploration, you can unlock the recording studio inside your hardware.

BY DAVE PHILLIPS

Since the public release of the 2.6 Linux stable kernel series, the Advanced Linux Sound Architecture (ALSA) has become the default kernel sound system. This change brings significant improvements to Linux audio and MIDI capabilities, including support for professional audio hardware, 3-D surround sound, advanced MIDI functions and software mixing or audio stream multiplexing. When combined with a kernel patched for low latency, ALSA provides resources for sound and MIDI that compare well with competing platforms and in some respects are superior to them. This is a bold claim, so let's look at ALSA to see what makes it tick.

Our Story Begins

The ALSA Project began when a young programmer named Jaroslav Kysela became frustrated with the kernel sound system's lack of full support for his Gravis Ultrasound audio/MIDI card. The Gravis card created its sounds by using sampled sounds stored in the card's memory in a file format known as PAT (patch). Banks of PAT sounds could be edited and stored by the user, as long as the user was running Microsoft Windows or Apple Mac OS. Linux, sad to say, did not provide such comprehensive resources, leaving Jaroslav with a sound card that was not fully operational.

At that time, the Linux kernel sound system was the OSS/Free system, a solid and serviceable audio/MIDI subsystem that had been with the kernel sources since the early days of Linux, thanks primarily to the pioneering work of Hannu Savolainen. Alas, OSS/Free had not kept pace with the rapidly evolving world of desktop audio production, and many sound cards either were unsupported or supported only partially, as was the case with the Gravis boards. To be fair, the OSS/Free maintainers were few; there was less organization in the general Linux audio world; and manufacturers then were, as some still are now, too secretive about their driver specifications.

It might have been possible to incorporate greater support for the Gravis cards into OSS/Free, but as Jaroslav Kysela researched the OSS/Free applications programming interface (API), he realized there was a need for a new API that could support more broadly the developments taking place with modern sound cards and digital audio interfaces. Professional and consumer-level expectations had risen, demanding support for features, such as high sample rates and bit depths for professional recording, 5.1 and other 3-D/surround sound audio output arrays; multichannel digital audio I/O; and multiple MIDI I/O ports. There simply were too many advances that required fundamental changes in OSS/Free, so Jaroslav did what any truly hard-core Linux coder does: he designed a new audio/MIDI API for Linux, calling it the Advanced Linux Sound Architecture.

Designing and implementing an API that would encompass the requirements of contemporary audio is a non-trivial task, and ALSA needed many years, many programmers and many releases to attain its current status as the kernel sound system. In its earlier stages, normal users had to install the system by hand, normally as a replacement for the OSS/Free system, in order to acquire support for a card or the extended features of a card. This process included uninstalling OSS/Free and recompiling the kernel for ALSA support, at that time a decidedly non-trivial task. Nevertheless, the ranks of dedicated ALSA users grew, development flourished and eventually ALSA was incorporated into the Linux 2.5 kernel development track. Finally, with the public release of the 2.6 kernel series, ALSA became the default kernel sound system.

What Is ALSA?

The ALSA home page gives us the following information:

> The Advanced Linux Sound Architecture provides audio and MIDI functionality to the Linux operating system. ALSA has the following significant features:
>
> 1. Efficient support for all types of audio interfaces, from consumer sound cards to professional multichannel audio interfaces.
> 2. Fully modularized sound drivers.
> 3. SMP and thread-safe design.
> 4. User-space library (alsa-lib) to simplify application programming and provide higher level functionality.
> 5. Support for the older OSS API, providing binary compatibility for most OSS programs.
>
> ALSA is released under the GPL (GNU General Public License)

and the LGPL (GNU Lesser General Public License).

Let's look at each one of these features, restating them in language more comprehensible to a normal user.

Efficient support means that you can manage the basic and advanced features of supported sound cards easily, using ALSA tools such as a sound card configuration utility and mixer programs. Such tools are integral components of the complete ALSA installation.

Modularized sound drivers means that ALSA sound card drivers are easy to install and update. They also provide the means by which the user can control a card's available options in more detail. Later in this article, we show how you can work with a driver module to access and control features of an ALSA-supported sound card.

ALSA supports multiprocessor, or SMP, machines. Thread-safe is a programming term that indicates the services provided by the software can run concurrently in different threads without bothering each other. In a modern audio/MIDI application, thread safety is a Very Good Thing.

ALSA's user-space library provides programmers, and hence their programs, with easy access to ALSA's services, and its significance to the normal user may seem a rather slight matter. However, the ALSA library provides the interface through which applications can reach those functions, helping form a more homogeneous environment at the user level. Your programs can run more harmoniously with one another, with enhanced possibilities for connection and communication between applications.

ALSA evolved during the first phase of Linux sound support when most applications were using the OSS/Free API, so an OSS/Free compatibility layer was an immediate necessity for normal users. A large number of Linux sound applications still need OSS/Free compatibility, so ALSA provides seamless support for the older API. However, programmers should note that the older API now officially is deprecated.

Installing and Configuring

Full details regarding installation are available on the ALSA home page (see the on-line Resources), so I mention here only a few details and clarifications. If you're using a distribution or customized Linux system based on a 2.6 kernel, ALSA already is installed. Distros and systems based on earlier kernels require a manual ALSA installation.

Installing ALSA is not especially difficult, and the way has been cleared at least partially by packages supplied by audio-centric Linux distributions/bundles, such as AGNULA/Demudi for Debian, Planet CCRMA for Red Hat and Fedora and AudioSlack for Slackware. Mandrake users can install one of Thac's packages (see Resources). Regardless of your base system, you must uninstall the OSS/Free modules before installing the ALSA package. Normally this task entails little more than moving the older modules into a temporary directory, in case you want or need to put them back, and making sure that the kernel's soundcore.o object file remains in its original place, usually /lib/modules/*your-kernel-number-here*/kernel/drivers/sound/. After removing OSS/Free you need to install the ALSA packages by way of your package manager of choice.

ALSA configuration has improved greatly over the years, but it still can be a tricky procedure, especially if your system has more than one sound device or if the device is connected to your computer on the USB or PCMCIA bus. Obviously, I can't go into the details regarding every possible configuration, but fortunately the ALSA Web site contains a large number of configuration pages for supported devices, often including tips and tricks from other users.

Basic Configuration

Basic configuration can be done with the alsaconf utility (Figure 1). alsaconf works well at recognizing single devices, but it might not do so well with systems containing multiple devices. Don't worry; it's still fairly simple to accommodate multiple audio and MIDI devices, and we return to that task in a few moments. For now, let's proceed as though you have only one audio device for your machine.

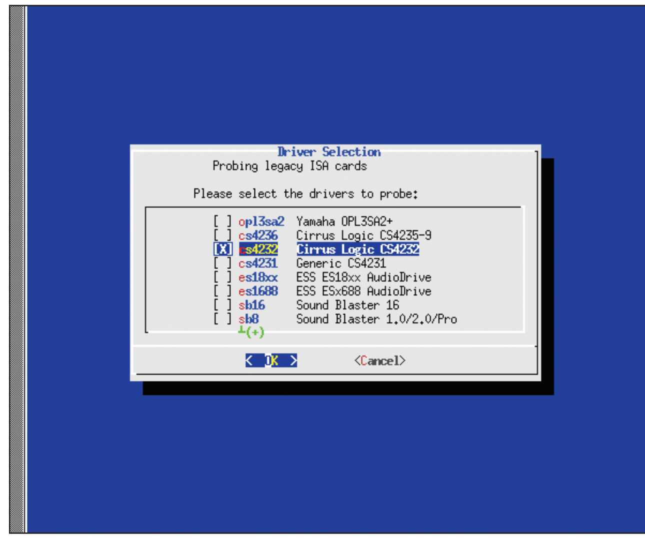


Figure 1. The alsaconf configuration tool is good for finding sound hardware on systems with one sound card installed.

After alsaconf has set up basic support for your sound device, you need to activate its playback and record channels. By default, ALSA started with all channels of your device muted. It may be an annoyance for some users, but it certainly reduces the likelihood of inadvertently blowing up your speakers when you first start your new system. You can set your sound device's channel capabilities with ALSA's alsamixer utility, a character-graphics mixer complete with sliders and

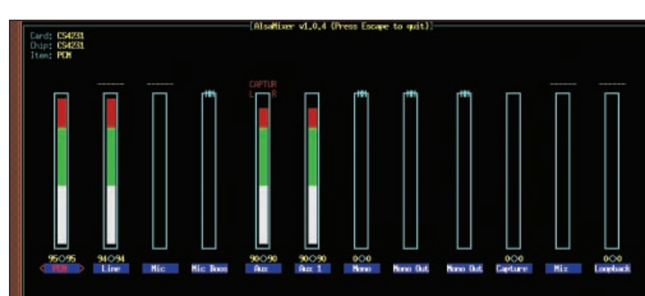


Figure 2. By default, ALSA starts with sound muted, so you need to set audio channel values with alsamixer.

DON'T BE SQUARE! GET CUBED!

L Series Laptop - LS1250-L
Light &Thin Performance
Starting at $1,562.54

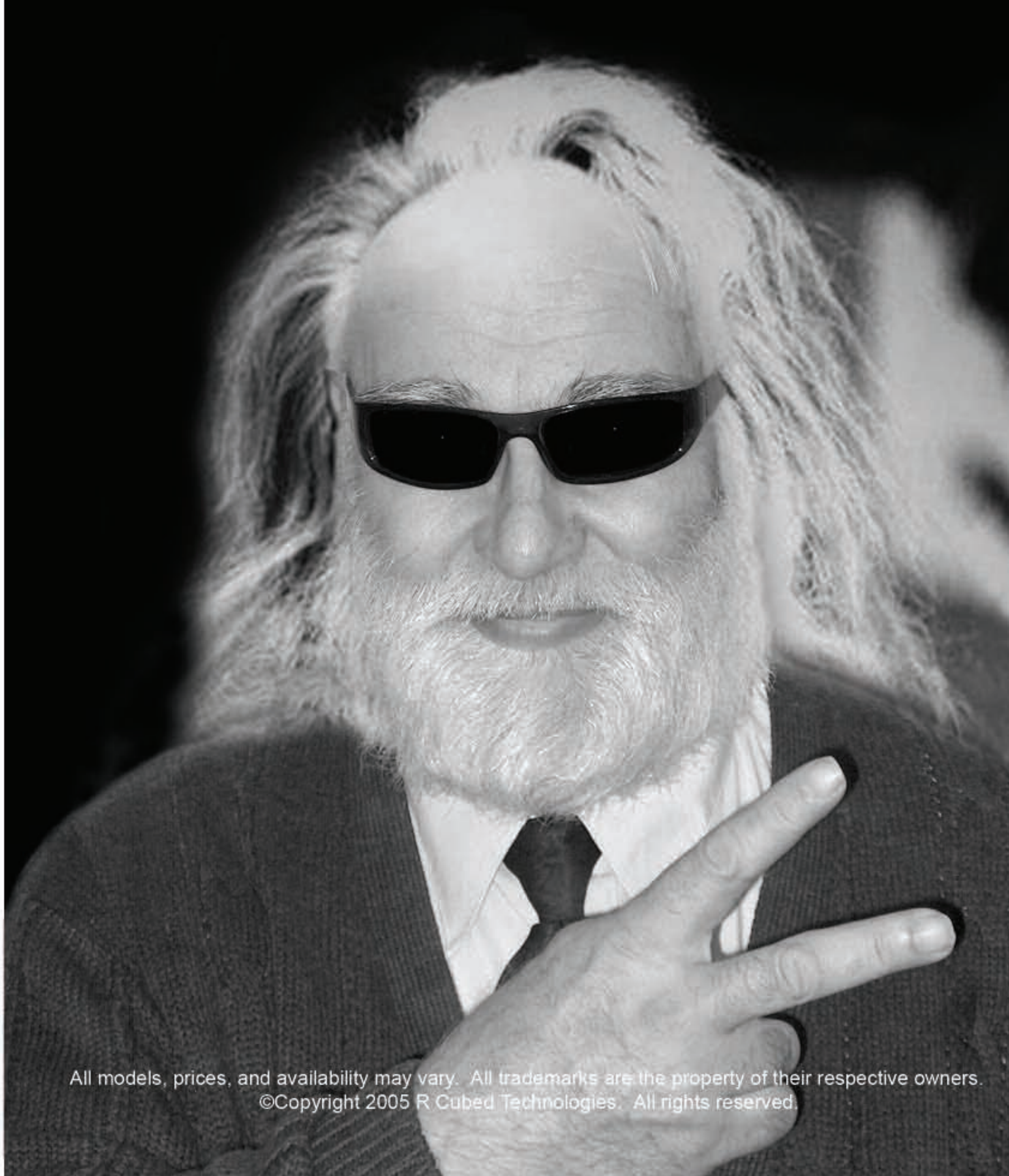
G Series Laptop - GW1550-L
Essential Technology on a Budget
Starting at $1,302.54

X Series Laptop - XW1550-L
Extreme Technology & Performance
Starting at $1,608.04

R Cubed Technologies has provided pre-installed Linux laptops without OS tax since 2003. We customize the Fedora Linux distribution for each laptops configuration providing support for: PCMCIA, USB, FireWire, X, CD/DVD/CDRW/DVDRW, Sound, Power Management, Ethernet, Modem, Wireless, and more. Our laptops are equipped with Intel Centrino Mobile Technology. We also offer Windows dual boot options. All of our laptops come with a one year parts and labor warranty. Visit us online at www.shoprcubed.com or call 309.34.CUBED for details.

R3 Technologies™
Working Hard to Bring Technologies to Life.

309.34.CUBED
www.shoprcubed.com



switches for each channel of the detected device (Figure 2). Use the Arrow keys to select a channel, use the < > keys to unmute/mute channels, and use the spacebar to select a channel as a recording source (capture, in ALSA-speak). When you've set your mixer preferences, run the alsactl utility to save and recall your settings (`alsactl store|restore`).

As you already can see, ALSA thoughtfully provides a number of useful tools to help configure the system. If you want to know more details about using those tools, simply run the utility with the -h option or use the man command to see a more detailed description. The following examples display the manual pages for the utilities I've mentioned already:

- `man alsaconf`
- `man alsamixer`
- `man alsactl`

Advanced Configuration

Now that we've considered some of the basic installation and configuration details, let's look at how we might set up a more complicated system. For the following example, I've used the configuration details for my laptop system, a Pentium II 366MHz HP Omnobook 4150 with a combined audio/video chipset manufactured by NeoMagic.

Setting up laptop audio support under Linux can be a complicated task, and it just so happens that my hardware is slightly problematic. Thankfully, ALSA supplies the tools I needed to resolve my difficulties with finding the correct chip and driver identification.

The alsaconf utility tries to identify your system's audio and MIDI capabilities and then it writes a basic configuration file to /etc/modules.conf. However, in the weird world of laptop sound support, things may not always be what they seem. For example, alsaconf correctly identified my laptop audio chip as a NeoMagic NM256. However, the configuration failed, reporting that I should use either the basic SoundBlaster16 driver (sb16) or one of the Crystal Sound drivers (cs423x).

On the advice of ALSA guru Takashi Iwai, I tried using alsaconf to set up the driver for the CS4232 chipset features, selecting the cs4232 module from alsaconf's list of non-PnP ISA chipsets. When I chose to probe for all possible DMA and IRQ settings, my machine locked up, freezing the keyboard and forcing a power-down and cold boot. To be fair, I must mention that alsaconf warned me that might happen. Happily, when I rejected the more aggressive search, alsaconf completed its task gracefully and added the following section to my /etc/modules.conf file:

```
# --- BEGIN: Generated by ALSACONF, do not edit. ---
        # --- ALSACONF version 1.0.4 ---
alias char-major-116 snd
alias snd-card-0 snd-cs4232
alias char-major-14 soundcore
alias sound-slot-0 snd-card-0
alias sound-service-0-0 snd-mixer-oss
alias sound-service-0-1 snd-seq-oss
alias sound-service-0-3 snd-pcm-oss
alias sound-service-0-8 snd-seq-oss
alias sound-service-0-12 snd-pcm-oss
alias snd-card-1 snd-virmidi
alias sound-slot-1 snd-card-1
      # --- END: Generated by ALSACONF, do not edit. ---
```

alsaconf merely set up a series of aliases for the general and card-specific services ALSA can provide for my machine. For normal use this section should remain as alsaconf generates it. By the way, the entries for the virmidi modules are there because I'm running Red Hat 9 with the ALSA packages from Planet CCRMA, a suite of components for setting up a low-latency, high-performance Linux audio/MIDI workstation. Planet CCRMA installs the virtual MIDI modules by default.

Next, I edited the driver options in /etc/modules.conf. In this section, I can customize various features of my sound chip, setting I/O port and IRQ addresses, enabling or disabling onboard synth capability and defining the DMA channels.

I ran `alsaconf -P` to see a list of the legacy non-PnP modules:

```
# alsaconf -P
opl3sa2 cs4236 cs4232 cs4231 es18xx es1688 sb16 sb8
```

Next, I probed the CS4232 driver for its default options settings:

```
# alsaconf -p cs4232
port=0x534 cport=0x538 isapnp=0 dma1=1 dma2=0 irq=5
```

I could have accepted these values and had a working audio system, but thanks again to Takashi Iwai, I discovered that I also could enable an onboard synth chip, the Yamaha OPL3, an inexpensive 4-operator FM synthesizer notorious for its ubiquity in inexpensive sound cards and its general cheesiness of sound. Takashi also advised entering and disabling an option for a physical MIDI port, simply to indicate its presence as a chipset feature. Thus, my current options section in /etc/modules.conf now includes this more complete configuration for the CS4232:

```
options snd-cs4232 port=0x534 cport=0x538 mpu_port=-1
➥fm_port=0x388 irq=5 dma1=1 dma2=0 isapnp=0
```

With this configuration, I now have working audio I/O and a cheesy onboard FM synthesizer. However, the synthesizer needs a set of sound patches before it can make any sound, and of course ALSA supplies the needed utility (sbiload) to load the patch data into the synth—ALSA even supplies the patches. I use the loader as follows:

```
sbiload -p 65:0 --opl3 \
/home/dlphilp/soundfiles/sbi-patches/std.o3 \
/home/dlphilp/soundfiles/sbi-patches/drums.o3
```

The options include the required target port (determined with `aconnect -o`) and a switch for either OPL2 or OPL3 support; the OPL2 is only a 2-operator FM synth. The example's patches are included with the ALSA tools (see `locate *.o3` and `locate *.db` for locations). A few other patch sets for the

OPL3 are available on the Internet, and Patch editors also are available.

At this point, I opened alsamixer to set the channel status for the CS4232. Figure 2 shown above displays the results. I now could play OGG and other music files (PCM), listen to my music CDs (Aux1) and watch and listen to DVDs and other video formats (Aux). I could record analog audio through either a microphone input or line-in jack, and I even could listen to MIDI music files played by the soundchip synth (Aux1). By default, I can do only one of those activities at a time, but ALSA supplies a neat plugin for software mixing, which I describe later.

By the way, on Red Hat or Fedora the entire ALSA system can be started and stopped with these commands:

```
/etc/init.d/alsasound start
/etc/init.d/alsasound stop
/etc/init.d/alsasound restart
```

If you have installed the Debian packages, the file is /etc/init.d/alsa. This feature makes it easy to test new configurations. The exact location of the alsasound control can be determined with `locate alsasound`.

The ALSA Virtual MIDI Module

The observant reader might wonder how I can route MIDI data without the benefit of MIDI hardware. Thanks to ALSA's virmidi module, my system has four virtual devices usable as raw MIDI I/O ports for any other ALSA sequencer clients. The sequencer of what is known as the ALSA sequencer API is not a sequencing application such as MusE or Rosegarden. This sequencer manages the merging and timing of freely interconnected MIDI data streams, including multiple connections to single ports. Compliance with the ALSA sequencer API allows each client to interconnect freely to one or more other clients, and it has become an expected capability of modern Linux audio software.

The ALSA aconnect utility tells me what ports are available for connection via the ALSA sequencer:

```
aconnect -i
client 0: 'System' [type=kernel]
    0 'Timer           '
    1 'Announce        '
client 72: 'Virtual Raw MIDI 1-0' [type=kernel]
    0 'VirMIDI 1-0     '
client 73: 'Virtual Raw MIDI 1-1' [type=kernel]
    0 'VirMIDI 1-1     '
client 74: 'Virtual Raw MIDI 1-2' [type=kernel]
    0 'VirMIDI 1-2     '
client 75: 'Virtual Raw MIDI 1-3' [type=kernel]
    0 'VirMIDI 1-3     '
```

This report indicates that I have four virtual MIDI ports. Whatever software I assign to those ports then can be connected to any other ALSA sequencer clients:

```
aconnect -o
client 65: 'OPL3 FM synth' [type=kernel]
    0 'OPL3 FM Port    '
client 72: 'Virtual Raw MIDI 1-0' [type=kernel]
    0 'VirMIDI 1-0     '
client 73: 'Virtual Raw MIDI 1-1' [type=kernel]
    0 'VirMIDI 1-1     '
client 74: 'Virtual Raw MIDI 1-2' [type=kernel]
    0 'VirMIDI 1-2     '
client 75: 'Virtual Raw MIDI 1-3' [type=kernel]
    0 'VirMIDI 1-3     '
```

This report shows my available receiving ports. Thus, the following command connects the first virmidi port to my onboard FM synth:

```
aconnect 72:0 65:0
```

The kconnect, alsa-patch-bay and QJackCtl programs provide graphical interfaces for device identification and connection.

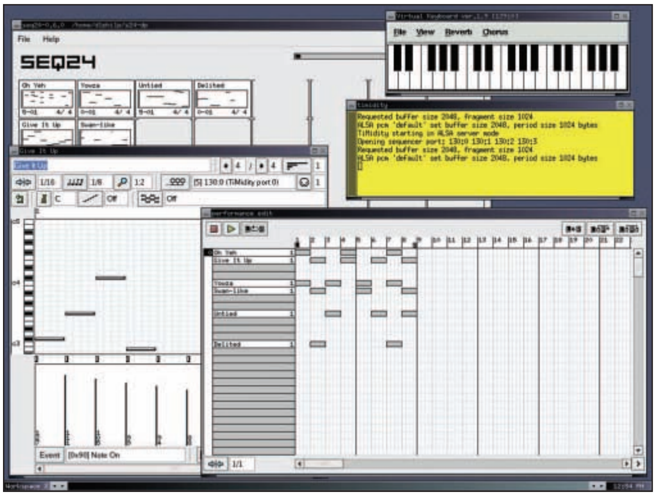


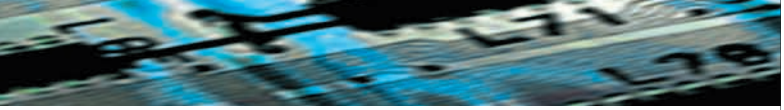
Figure 3. A Basic Linux Laptop MIDI System

Figure 3 shows off a small but powerful MIDI sequencing system. The main program is Rob Buse's seq24, a lightweight looping sequencer designed in the style of the hardware sequencers of the 1980s and 1990s. seq24 manages its connections internally, and the figure conceals the connections between the virtual keyboard and seq24 as well as the output targets for the individual sequences. Some of the sequences have been routed to the OPL3 synth; others have been sent to an instance of TiMidity running as a Soundfont server.

A USB MIDI Interface

Like many other laptop owners, I've hooked up a USB device to my machine, in this case a MIDIman 2x2 MidiSport. The MidiSport provides two independent I/O ports, and yes, ALSA supports multiport MIDI hardware. However, I don't always have my MidiSport with me when I use this machine, so I prefer to load the USB module after setting up my CS4232 and virmidi cards. To defeat the autoloading of my USB MIDI module, I added these lines to /etc/hotplug/blacklist:

```
# So I can keep my preferred order of sound cards:
snd-usb-audio
```



```
# uhci ... usb-uhci handles the same pci class:
usb-uhci
```

Next, I wrote the following script to configure and activate the MidiSport 2x2:

```
echo "Loading MidiSport firmware..."
modprobe snd-usb-audio
sfxload -I \
  /usr/share/usb/ezusbmidi/ezusbmidi2x2.ihx \
  -D /proc/bus/usb/001/003
echo "Done !"
```

The firmware and loader are included with the ALSA installation. You may need to query the /proc/bus/usb filesystem for your available USB identifiers, and you may need to try each identifier to find which one applies to your hardware. Use the `cat` command to list your identifier numbers:

```
$ cat /proc/bus/usb/001/
001 003
```

The system reports an error if you select the wrong identifier, so at least in my case the trial-and-error process didn't last long.

A PCMCIA Audio Card

As though I hadn't already stuffed my little system full of ALSA drivers, I also wanted to use the Core Sound PDAudioCF card, a high-quality digital audio input card made for handheld computers, such as the Zaurus, but quite usable with a CF-to-PCMCIA adapter. Again, I want to have my other devices configured before setting up the PDAudioCF, so I simply wait until I have everything else working as desired before inserting the card. The system autodetects the new hardware and loads the appropriate module (snd-pdaudiocf), a procedure totally transparent to the end user.

Using this card is easy. The following example illustrates the use of ALSA's arecord utility to record a 30-second stereo digital audio stream from the S/PDIF digital output of my desktop system's SBLive to the PDAudioCF card:

```
arecord -f dat -D hw:3,0 -d 30 foo.wav
```

The `-f dat` option sets the recording format to include a sample rate of 48kHz, which is the only output sample rate supported by the SBLive. I substituted `-f cd` for the DAT option and recorded again from the S/PDIF output of my Delta 66, this time with the standard redbook CD audio values, that is, 16-bit stereo audio with a sample rate of 44.1kHz. In both cases, the recording and playback were flawless and had beautiful audio quality, thanks to the PDAudioCD card. For more details regarding ALSA's playback and record utilities, see `man aplay` and `man arecord`.

Linux laptop sound support is a weird world, and I spent considerable time getting things working properly. However, my machine now has a sound system supporting stereo analog PCM I/O, a CD audio channel, a MIDI-accessible onboard synthesizer, four virtual MIDI I/O ports, an external 2x2 MIDI

Thanks to ALSA's virmidi module, my system has four virtual devices usable as raw MIDI I/O ports for any other ALSA sequencer clients.

interface and a high-quality digital audio input port. Not too shabby a set of capabilities for a PII 366, and, of course, the real thanks go to ALSA.

By the way, if I forget the ordered numbering of my cards, I always can query the proc filesystem for their numbers and status:

```
$ cat /proc/asound/cards
0 [CS4231         ]: CS4231 - CS4231
                     CS4231 at 0x534, irq 5, dma 1&0
1 [VirMIDI        ]: VirMIDI - VirMIDI
                     Virtual MIDI Card 1
2 [M2x2           ]: USB-Audio - Midisport 2x2
                     Midiman Midisport 2x2 at usb-00:07.2-1, full speed
3 [PDAudioCF      ]: PDAudio-CF - Core Sound PDAudio-CF
                     Core Sound PDAudio-CF at 0x100, irq 11
```

Thus, the specific hardware definitions would be hw:0, hw:1, hw:2 and hw:3. hw:1 and hw:2 are MIDI-only devices and cannot be used for audio purposes. And yes, proc is reporting a CS4231 where I've configured a CS4232, but Takashi Iwai assured me that this behavior is normal for the chipset. I know, it's weird.

Basic and Advanced Desktop Configuration

My desktop system has been much easier to configure. It still is a fairly complex system, supporting one sound card—a SoundBlaster Live Value, with external MIDI adapter—the virtual MIDI module and an M-Audio Delta 66 multichannel audio interface.

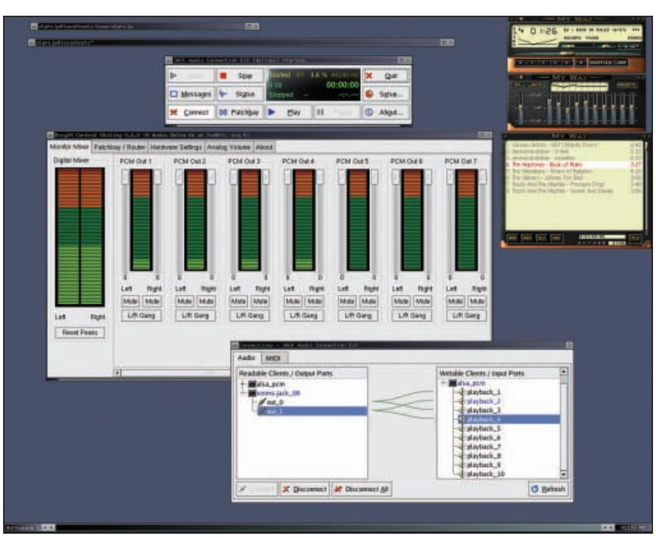


Figure 4. The envy24control Mixer

As with the OPL3 synthesizer on my laptop, I must load sound data into the SBLive's EMU10k1 hardware synthesizer, using the ALSA sfxload utility to load soundfonts into the synth. This command configures my SBLive synth with a General MIDI soundfont distributed with the sound card:

```
sfxload 8mbgmsfx
```

Recently, developer Lee Revell significantly improved the ALSA driver for the Creative Labs SBLive and Audigy sound cards, unlocking much greater potential than was available through the previous drivers. Lee followed the lead of the kX Project, an open-source Windows-based project intended to open all the capabilities of the SBLive/Audigy cards, including true multichannel I/O, access to the DSP registers and support for x.1 surround sound. Lee's work greatly expands the recording and playback possibilities for inexpensive hardware, bringing even more value to Linux as a desktop music and sound workstation.

Installation and operation of the virtual MIDI driver for my desktop is exactly the same as it was for my laptop. See the appropriate section above for details.

Channel settings for my SBLive can be made using alsamixer, but setting up my Delta 66 requires the use of the specialized envy24control mixer (Figure 4). This mixer provides access to and control of the advanced features of cards with the ice1712 chipsets, including the M-Audio Delta cards.

ALSA easily handles systems with multiple cards. The ALSA utilities usually include an option for specific card selection, as in these examples for my SBLive and Delta cards:

```
alsactl restore 0
alsactl restore 2
alsamixer -c 0
alsamixer -c 2
```

In my system, card 1 is the virtual MIDI card, which takes no channel settings and therefore has no associated mixer.

ALSA Plugins and the .asoundrc File

The ALSA plugins are utility services available through a file named .asoundrc, typically placed in your home directory. Plugin services include resampling, channel routing, sample format conversion and software volume control. Please see the ALSA Wiki notes on .asoundrc for detailed information regarding these and other ALSA plugins.

As I mentioned earlier, the default sound capability of my laptop is restricted to only one application at a time. Fortunately, ALSA provides a cool plugin called dmix, and its sole function is to provide a type of audio stream multiplexing called software mixing. Unfortunately, ALSA doesn't autodetect the need for the dmix plugin, so the user must prepare the necessary components.

Here is the .asoundrc for my laptop:

```
pcm.!default {
        type plug
        slave.pcm "dmixer"
}
```

```
pcm.dmixer  {
        type dmix
        ipc_key 1024
        slave {
            pcm "hw:0,0"
            period_time 0
            period_size 1024
            buffer_size 4096
            rate 32000
}
        bindings {
            0 0
            1 1
        }
}

pcm.dsp {
        type plug
        slave.pcm "dmixer"
}

ctl.dmixer {
        type hw
        card 0
}
```

This file defines a new PCM device named dmixer, of the plugin type dmix, which is slaved to the PCM capabilities of the soundchip. The plugin also lets me tailor the sample rate to the capabilities of my hardware, easing CPU demands.

With the dmix plugin I can run an audio player and a video player at the same time. In case you're wondering why I might want to do such a thing, consider that I often study t'ai chi videos available on DivX discs. The video is usually wonderful, but the background music isn't always to my liking, so it's nice to be able to listen to something more to my taste. The following commands launch Andy Lo A Fo's neat alsaplayer soundfile player and the MPlayer video player:

```
mplayer -ao alsa9:dmixer -aop list=volume:volume=0 \
  -framedrop foo.avi
alsaplayer -o alsa -d plug:dmixer cool-foo.mp3
```

The video player's audio output is negated, thanks to MPlayer's software volume control, while the alsaplayer plays my preferred music. Very cool stuff, courtesy of the dmix plugin.

I have no special needs on my desktop system, but I've configured my .asoundrc file for basic accommodations for the SBLive and the Delta 66:

```
pcm.emu10k1 {
        type hw
        card 0
}

ctl.emu10k1 {
        type hw
        card 0
}
```

```
pcm.Delta66 {
        type hw
        card 2
}

ctl.Delta66 {
        type hw
        card 2
}

pcm.DeltaPlug {
        type plug
        card 2
}

ctl.DeltaPlug {
        type plug
        card 2
}

pcm.DeltaPlugHW {
        type plughw
        card 2
}

ctl.DeltaPlugHW {
        type plughw
        card 2
}
```

The card numbering reflects the ordering list when I query /proc/asound:

```
$ cat /proc/asound/cards
0 [Live           ]: EMU10K1 - Sound Blaster Live!
                     Sound Blaster Live! (rev.8) at
0xd000, irq 3
1 [VirMIDI        ]: VirMIDI - VirMIDI
                     Virtual MIDI Card 1
2 [M66            ]: ICE1712 - M Audio Delta 66
                     M Audio Delta 66 at 0xd800, irq 5
```

ALSA does not provide a default .asoundrc file, nor is it an absolute necessity. However, many interesting ALSA features are accessible only through .asoundrc, and the reader is advised to study the example files found on the ALSA Web site.

For an advanced example, see Timo Sivula's El Cheapo HOWTO, a rather amazing hardware/software hack that allows sample-accurate multichannel recording using two or more consumer-grade sound cards (Timo used the Creative Labs PCI128). Under normal circumstances, such an approach would be doomed to fail from inherent instabilities between the clock crystals of the cards, but Timo's hardware modifications and the capabilities of .asoundrc make it possible. The El Cheapo HOWTO is not for the faint of heart, but it does succeed at providing an inexpensive path to high-quality multichannel recording on the Linux desktop.

A Brief Note Regarding JACK

Figure 4 shows off the envy24control mixer in a JACK environment. JACK is an audio connections manager designed to professional specifications for low-latency communication between the JACK server and its clients. JACK requires a native system audio driver, which for Linux can be a dummy driver, an OSS driver, PortAudio or, most typically, ALSA. I will present the JACK system in detail in a future article.

The ALSA Applications Software Base

It is no exaggeration to state that all contemporary major Linux audio software wants ALSA's special services. MIDI programs enjoy the connectivity of the ALSA sequencer, digital audio systems make use of ALSA's drivers for pro-audio hardware and thorough support is provided for common desktop audio/video activities. Figures 5 and 6 represent some screens commonly seen on my desktop, thanks to ALSA.

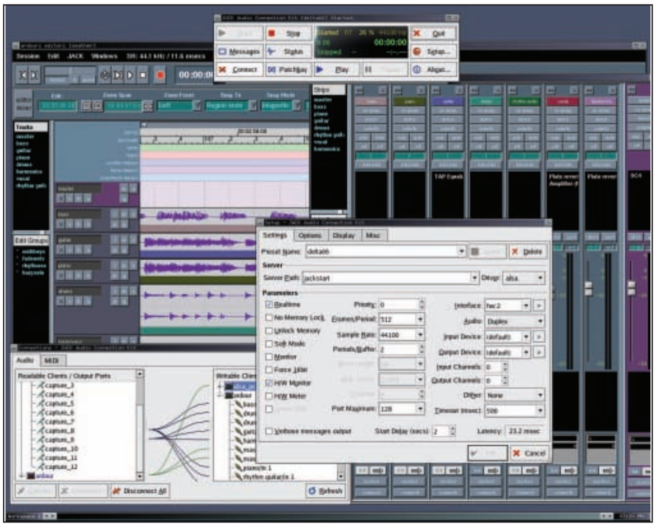


Figure 5. Recording with Ardour

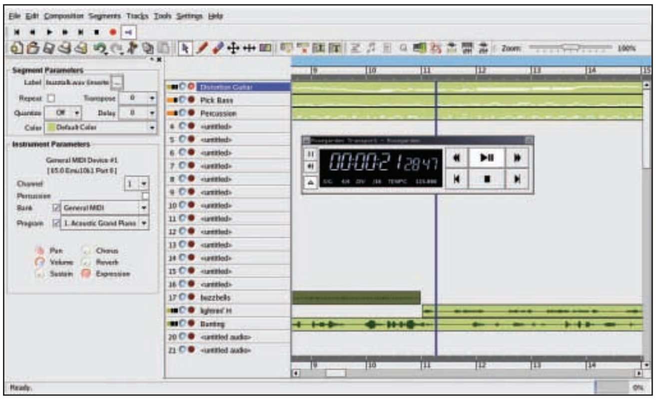


Figure 6. Audio/MIDI Sequencing in Rosegarden

Future Work

From the normal user's point of view, ALSA's most obvious weakness is in its lack of GUI front ends for the various tools and utilities that make up so much of the system's power: a configuration panel, complete with options for configuring and reordering your installed cards, loading the virtual MIDI driver, selecting plugins for .asoundrc and generating a new file, operating alsactl and so forth. ALSA is indeed feature-rich, but too many of its excellent features are available only to those of us willing to write scripts and resource files.

Fortunately, there is an abundance of ALSA documentation and information for users of all levels. I already mentioned the man pages for the ALSA utilities. The ALSA Web site includes many resources for basic and advanced use of the system. Also, the alsa-user and alsa-devel mail lists are founts of wisdom and assistance, as is the excellent ALSA Wiki.

The project always needs programmers, but it also needs graphics artists, technical writers and beta testers, so even if you can't code, your skills might still be valuable to the project. Donations of hardware and cash also are cheerfully accepted; please see the ALSA Web site for appropriate contact details.

The average user can expect to see more cards supported, with more features available to the user. Hopefully, card configuration will become easier: getting the most from ALSA can be a simple matter or it can be a difficult thing. It is true that what is difficult is not impossible, and the payoff certainly can be worth the effort. Hopefully, though, we also will see some more accessible tools for user-level configuration.

Acknowledgements

The author thanks Jaroslav Kysela and Takashi Iwai for their vast patience and excellent assistance over the years I've been using ALSA. Thanks also to all members of the ALSA development team, past, present and future, for this great gift to the world of Linux sound. Finally, thanks to Len Moskowitz for the extended loan of his outstanding Core Sound PDAudioCF card.

Resources for this article: www.linuxjournal.com/article/8324.

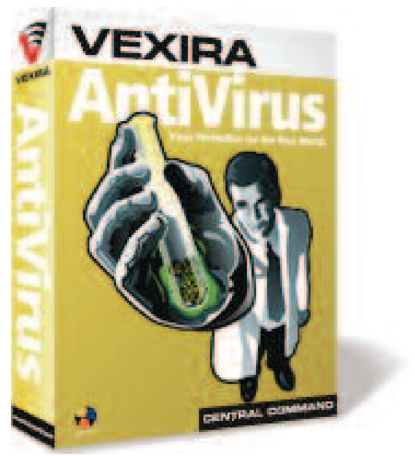
Dave Phillips is a musician, teacher and writer living in Findlay, Ohio. He has been an active member of the Linux Audio community since his first contact with Linux in 1995. He is the author of *The Book of Linux Music & Sound*, as well as numerous articles in *Linux Journal*.

Orion Multisystems DS-96

Orion Multisystems announced the availability of the DS-96, a 96-node deskside cluster workstation. Stackable up to four systems, the DS-96 boots 96 individual nodes as one system using a single on/off switch. It does not have special cooling requirements, and the maximum power draw is 1,500 watts from a standard power outlet. The entire system is based on eight Orion processor array boards, with each board composed of 12 individual nodes on a private network. Each node has its own x86 processor, chipset, memory, optional disk drive and networking capability. Other features include dual 10-GigE fiber cards and a 12-port GigE switch, a DVD/CD-RW and one 2.5" hard drive on the head node, and one optional 2.5" hard disk drive per node. The DS-96's software is based on Fedora Core 2.

CONTACT Orion Multisystems, 3090 Oakmead Village Drive, Santa Clara, California 95051, 800-344-1367, www.orionmulti.com.

Vexira Antivirus for Mail Server



Vexira Antivirus for Mail Server is a content security application that provides scalable protection from viruses, spam, spyware and other malicious applications. It can integrate directly with many e-mail systems, or it can act as its own SMTP relay server to shield the e-mail server itself from attack. Five different checks are performed on e-mail: file gate, file filter, field filter, virus filter and spam filter. Each user or domain owner can have specific configurations and rule sets. Vexira also offers scalable LDAP support. Vexira contains several embedded e-mail defense features, including DoS protection, blacklisting, zip-of-death protection and mail-bomb protection. Vexira supports header modification, subject modification and custom message marking, and it offers a real-time overview application of current operations.

CONTACT Central Command, Inc., PO Box 468, Medina, Ohio 44258, 330-723-2062, www.centralcommand.com.

SATA EtherDrive Storage

Coraid's EtherDrive Storage appliance now is available for Serial ATA disk drives. The refined chassis design includes 15 hot-swap drive bays that accommodate standard SATA disk drives. The new shelf offers a dual-GigE interface, redundant hot-swap power modules and fans. Fully populated drive bays using 400GB disk drives yield 6TB of storage, but using 500GB drives, the new shelf provides 7.5TB of storage. SATA EtherDrive Storage appliance uses the AoE (ATA over Ethernet) protocol. Using Ethernet connections, EtherDrive Storage Blades appear to servers on the network as locally attached disks. In addition, the EtherDrive Storage appliance can be assembled into large RAID sets and storage volumes.

CONTACT Coraid, Inc., 2730 Camino Capistrano, Suite 1, San Clemente, California 92672, 877-548-7200, www.coraid.com.

Silicon Graphics Prism Deskside System

SGI announced the newest member of the Silicon Graphics Prism product line, the Prism Deskside Visualization System. The Prism products offer visualization capabilities for tackling problems that generate massive data sets. Based on SGI's scalable, shared-memory visualization architecture and Altix high-performance servers, the Deskside Prism features dual Itanium 2 processors and up to 24GB of memory in a deskside form factor. The Deskside Prism can drive displays with up to 10 million combined pixels, as the system's dual ATI FireGL graphics processors simultaneously can serve four full bandwidth channels. With the Deskside Prism, users can transparently access and share data and resources from cross-platform clients connected across networks for efficient collaboration.

CONTACT Silicon Graphics, Inc., 1500 Crittenden Lane, Mountain View, California 94043, 650-960-1980, www.sgi.com.

Mobilinux 4.0

MontaVista's Mobilinux is an optimized Linux operating system and development environment suited for wireless handsets and mobile devices, with requirements for power management, hard real-time performance, fast startup and a small footprint. Based on the 2.6 kernel, Mobilinux features include enhanced core capabilities, footprint improvements, boot times of less than one second, advanced real-time support and support for requirements for mass-market, single-chip phone designs. Power management improvements include dynamic power management (DPM) for adjustments on the fly, MontaVista Power Manager and a cross-platform DPM Library. In addition, Mobilinux has ARM EABI support for compatibility with standard third-party tools, compiler support for thumb mode and an integrated graphical layer for user interfaces. Mobilinux is built on updated Eclipse 3.0.1 and CDT 2.1 technology as well as TinyX and GTK technologies.

CONTACT MontaVista Software, 1237 East Arques Avenue, Sunnyvale, California 94085, 408-328-9200, www.mvista.com.

Please send information about releases of Linux-related products to Heather Mead at newproducts@ssc.com or New Products c/o *Linux Journal*, PO Box 55549, Seattle, WA 98155-0549. Submissions are edited for length and content.

servers ▶ DIRECT™
More products, better service, guaranteed.

GO STRAIGHT TO THE SOURCE!

Introducing the Servers Direct Blade System with the power of Intel® Xeon™ Processor

Increased computing power in a smaller footprint and simplified maintenance help you expand your enterprise solution to meet the most intense application demands.

SDB-1100H Servers Direct Blade System

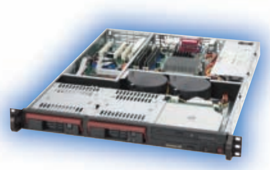
Featuring a Server Direct Server Compute Blade powered by dual ® Xeon™ 800FSB Processors.

Benefit of using Servers Direct Blade System:

- More power, bandwidth, and processing performance to meet the demanding requirements of departmental workloads
- Deliver world-class performance for peak server workloads
- Future 64-bit-enabled applications
- High performance small form factor SCSI hard drives (RAID 1 with 2 HDDs)
- Ethernet I/O for demanding, data-intensive applications
- Ability to easily add hotswappable SCSI hard drives, and additional Ethernet, or Fibre Channel I/O for increased application performance (RAID 1E requires use of the HDDs on the SBX82 as well as the HDDs on the SBESCSI)

1U Xeon Entry Level Server

SDR-1300T



Highest performing with Dual Xeon 800MHz. Excellent with general purpose applications and provide the most power.

- Intel Xeon Processor 2.8Ghz with 800FSB 1MB Cache (Dual Processor Option)
- Intel Extended Memory 64 Technology
- 1U Chassis with 420W power supply
- Supermicro server board with Intel® E7320 (Lindenhurst VS) Chipset
- Kingston 512MB DDR333 ECC Reg. RAM (2x256MB)
- 1pc x Seagate 80GB SATA 7200RPM hard drive
- 2 x 1" Hot-swap SATA drive bays
- Integrated ATI Rage XL SVGA PCI video controller
- 2x Intel® 82541GI Gigabit Ethernet Controllers
- 2x SATA Ports via 6300ESB SATA Controller RAID 0, 1 Supported

$999

2U Xeon Processing Server

SDR-2103T

High-density 2U platform optimized for performance and flexibility; ideal for Web hosting, data center, terminal services and High Performance Computing (HPC)

- Intel Xeon Processor 3.0Ghz with 800FSB 1MB Cache (Dual Processor Option)
- Intel Extended Memory 64 Technology
- 1U Intel Chassis with 700W PFC power supply
- Intel® Server Board SE7520JR2
- Kingston 512MB DDR333 ECC Reg. RAM (2x256MB)
- 6pcs xSeagate 160GB SATA/150 W/ncq 7200rpm 8MB Cache
- Intel SRCS16 6Channel SATA RAID Controller Card
- Integrated ATI Rage XL SVGA PCI video controller
- 2x Intel® PRO/1000 MT Server Network Connections (Intel® 82546GB controller)

$3,499

5U File Server

SDR-5301S

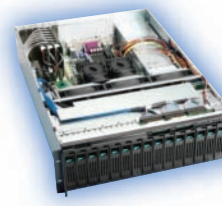
Outstanding performance, excellent data protection, and advanced management for departmental servers.

- Intel Xeon Processor 3.0Ghz with 800FSB 1MB Cache (Dual Processor Option)
- Intel Extended Memory 64 Technology
- Intel SC5300LX Chassis with Redundant 730W Power Supply
- Intel server board w/Intel® E7520 (Lindenhurst) Chipset
- Kingston 1024MB DDR400 ECC Reg. RAM (2x512MB)
- Adaptec 2200S SCSI RAID Controller Card
- Include 6-Drive SCSI Hot-Swap Cage Kit
- 6 x Seagate 36GB SCSI 10K RPM U320 SCA hard drive
- ATI Rage XL SVGA PCI video controller with 8MB of video memory
- Dual Intel® PRO/1000 Server Network Connections

$4,999

3U Clusterable SATA SAN Nodes

SDR-3303T



Provides a flexible, price and performance advantages to the storage needs of Small to Medium Business (SMB) market segment.

- Intel 3U SAN Storage Enclosure w/700W Redundant Power Supply
- Integrated Intel SE7501HG2 Server Board
- 2x Integrated SAN Intel Xeon 3.06Ghz/533FSB Processor
- Integrated SAN 2x256MB Compact Flash Memory Cards w/SAN Mgt Software
- 3 X Integrated Intel SRCS16 6-Channel SATA RAID Cards
- Kingston 1024MB DDR266 ECC Reg. RAM (2x512MB)
- 16pcs x Western Digital WD2500SD RAID SATA
- Integrated SAN Management Software & Storage System

$13,599

Big business power, small business price tag.

Your business requires solid server solutions. With Servers Direct server systems based on the Intel® Xeon™ Processor, you can count on high availability, maximum efficiency and proven performance to help you meet your business reliability requirements.

1.877.727.7127 | sales@serversdirect.com

intel inside XEON

Ubuntu Linux 5.04

REVIEWED BY STEVE R. HASTINGS

PRODUCT INFORMATION

Vendor:
Canonical

URL:
www.ubuntulinux.org

Price:
Free

THE GOOD

- Cutting-edge GNOME desktop.
- Outstanding package management system.
- Free, as in beer and as in freedom.

THE BAD

- Only available for x86, x86_64 and PowerPC architectures.
- Six-month release cycle sometimes leaves rough edges.

The Ubuntu Linux distribution is produced by a company called Canonical, working together with the Debian Project. Its goal is to make a free Linux distribution that simply works and is localized for as many different languages as possible. You can read the Ubuntu Manifesto on the ubuntulinux.org Web site. The name Ubuntu is an ancient African word that means "humanity to others".

This is the second release of Ubuntu, code-named Hoary Hedgehog. The previous release was Ubuntu 4.10. The version numbers are based on the year and month of the release; 5.04, therefore, was released in April 2005.

Ubuntu 5.04 provides cutting-edge Linux desktop features and easy administration with Debian's APT package management system. It also is available in a live CD version that runs without installing on the hard drive. Ubuntu is supported on x86, x86_64 and PowerPC architectures, and future plans call for releases to support additional architectures.

Getting Ubuntu

The usual way to get Ubuntu is to download a CD image either from the Ubuntu Web site or by using a BitTorrent client. Alternatively, you can order official Ubuntu CDs if you like; remarkably, they are free of charge. The hardware detection in the live CD is identical to the hardware detection in the Ubuntu installer, so if the live CD works, you can be confident that the installer will work as well.

A DVD image also is available for BitTorrent download. The DVD is suitable for installing Ubuntu on a computer without Internet access. It can be used as a live CD or as an install CD.

Installation

Installation is a straightforward process. Ubuntu 5.04 has a text-based installer, but it is easy to use and has excellent hardware detection. In the simplest case—installing to a blank hard disk—it handles partitioning and formatting automatically. Manual partitioning is possible as well, allowing you to delete and create partitions and format them as ext3, ext2, ReiserFS, JFS, XFS, FAT16 or FAT32 filesystems, all with LVM or RAID support. By pressing Alt-F2, you can access a second virtual terminal and use a root shell to set up your partitions by hand.

If the system has a connection to the Internet during the installation, the Ubuntu installer automatically finds and installs the latest package versions so your new Ubuntu system is fully up to date. And, thanks to the Kubuntu Project, an install CD that includes KDE also is available. Ubuntu 5.04 also offers support for network installs using Kickstart.

If you want to add additional desktop environments such as Xfce, after the initial install you can enable the universe component (see below) and install the necessary packages. In addition, you can choose the server install option to get a minimal Ubuntu system and then manually install exactly the packages you choose.

As is generally true of Debian-based systems, you need to run the installer only once. Even major releases can be updated using the standard package management tools. However, keep the install CD handy to use as a rescue disk.

If you have an NVIDIA or ATI graphics adapter and you want to use the vendor's proprietary binary drivers, with Ubuntu you can easily install the packages from the restricted package set. Furthermore, as updates to those drivers are released, your system can install them automatically.

Cutting-Edge GNOME Desktop

Ubuntu Linux 5.04 is based on the GNOME 2.10 desktop environment. It features the latest slick GNOME features from the GNOME developers as well as a few new features added by the Ubuntu developers. It uses the X.org X server.

The theme, desktop art and applets shown in Figure 1 are all out-of-the-box Ubuntu defaults. I had the mouse pointer hovering over the red update icon in order to read the tool tip saying that two new packages are available; the screenshot tool does not capture

the mouse pointer.

Ubuntu is developed on a six-month cycle, as is the GNOME desktop itself. Each Ubuntu release will include the latest GNOME release. Canonical has promised to provide security updates for each release for at least 18 months.

Ubuntu has a clean desktop philosophy, so your desktop initially is completely empty of icons and files. The Ubuntu developers wrote some GNOME applets, however, that allow all features of GNOME to be accessed from GNOME panels. For example, the Trash Can applet gives access to the Trash folder without needing to move any open windows to get to the desktop. Of course, you are free to put icons on your desktop if you prefer.

The GNOME menus are located on the top left of the default Ubuntu desktop, and as of GNOME 2.10, the menus are Applications, Places and System. The Applications menu includes icons to launch applications, filed into categories such as Games and Internet. The Places menu includes icons to open a file manager window for the user's home directory, the user's Desktop and a place called Computer, with all storage devices available on the computer. The Places menu also includes any locations the user has bookmarked from the file manager, as well as a few icons for accessing network servers, searching for files or viewing the most recently used documents list. The System menu is used for setting GNOME preferences, system administration, getting GNOME help and closing a GNOME session. Overall, these three menus are an excellent way to organize the system menus; it's easy to remember where to look for things.

The GNOME 2.10 desktop in Ubuntu is an excellent choice for beginning computer users. Thanks to the GNOME Volume Manager, GNOME does sensible things when a user works with storage devices. For example, when the user inserts a CD audio disk into a CD drive, the GNOME CD player automatically runs.

When the user plugs in a USB Flash drive, it is recognized, mounted and a file manager window opens that shows the mounted device. In addition, an icon appears on the desktop with a name such as 256M Removable Media, and an identical icon appears in the Places menu. Users coming from other OSes should learn to use the Unmount Volume command before unplugging the USB device, but as long as they don't unplug the device while it actually is writing data, nothing bad happens if they simply unplug it. The system simply removes the icon from the desktop and the Places menu.

Other removable devices are handled in similarly slick fashion. Plugging in a device with photos, such as a digital camera, results in a pop-up dialog offering to import the photos.

The GNOME file manager, by default, runs in a spatial mode where each place you can visit with the file manager opens in its own window, and the location and size of each of these windows are remembered. A browser window mode also is available, and a check box in the file manager preferences—Always open in browser windows—can be used to set the browser window mode as the default.

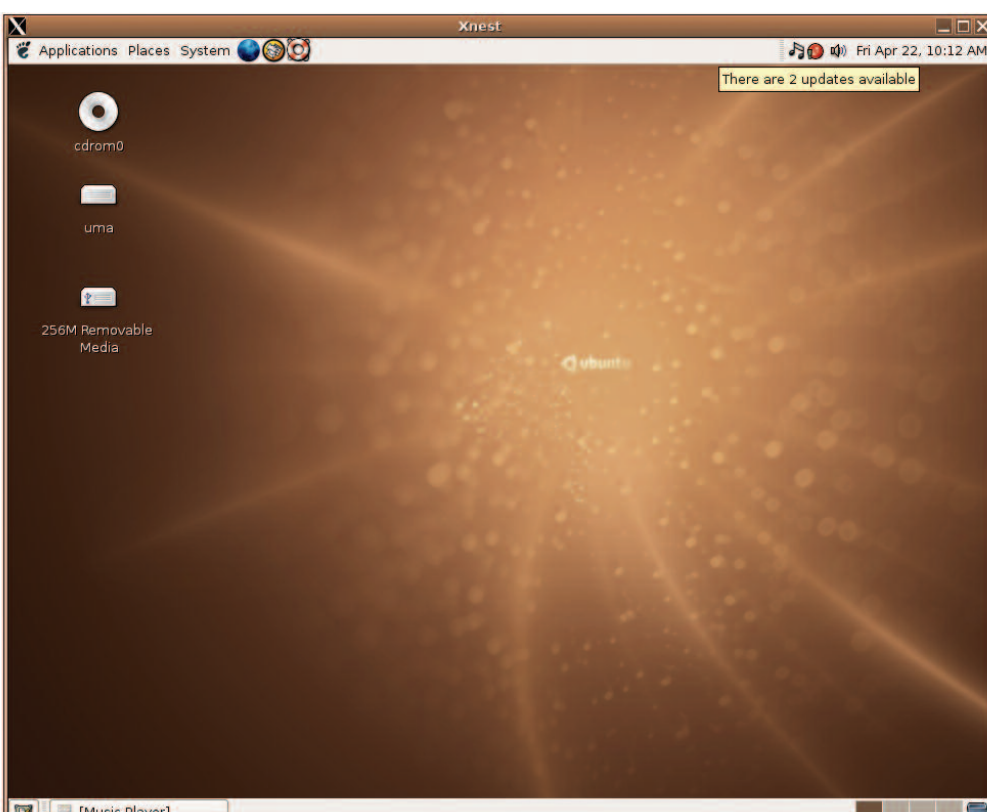


Figure 1. The GNOME desktop with a CD-ROM, a server called uma and a USB Flash drive all mounted. Music is playing. Updates are available (red icon, upper right).

Package Management

As noted before, Ubuntu is based on Debian GNU/Linux. Debian's package management system, APT (Advanced Packaging Tool) is famously easy to use. As long as your system has access to a server with the package you want, a single command installs the package and automatically brings in any other packages needed by the one you requested. There is no charge for downloading new packages or security updates.

Using the `apt-get` command-line tool, it also is possible to update your system, automatically retrieving any new versions of the packages you already have. There is also an ncurses-based character-mode tool called aptitude that makes it easy to browse packages, plus a GNOME graphical package browser called Synaptic Package Manager. All of these have been standard in Ubuntu since the

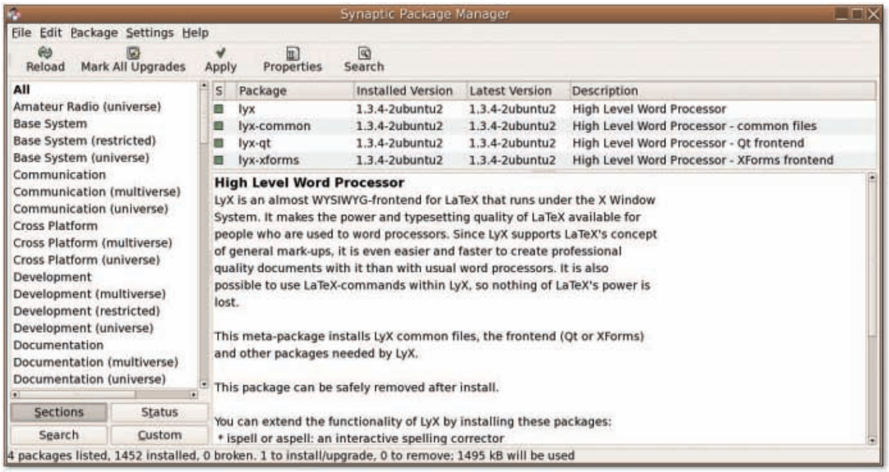


Figure 2. The Synaptic Package Manager showing the LyX packages.

Figure 3. The Ubuntu Update Manager showing that Adobe Acrobat Reader 7 is available for download.

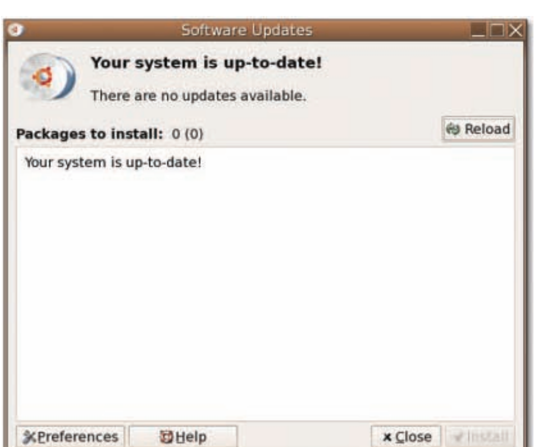


Figure 4. The Ubuntu Update Manager showing that the system is up to date.

first release.

With the 5.04 release, Ubuntu has made package management even easier, and the most common cases are now extremely simple and discoverable. When updated packages become available, a bright-red icon appears in the notification area. Clicking on the icon launches the Ubuntu Update Manager, which shows a list of packages with available updates; one click on the Install button updates the Ubuntu system to the latest packages. This handles both security updates and feature updates.

Under Applications/System Tools there is a launcher for the Add/Remove Programs dialog, another new feature to Ubuntu 5.04. The most common programs a user might want appear here, along with an icon, a friendly name and a terse explanation of what the program does. Simply marking a check box next to the program name selects that program for installation. Clicking on the Advanced button brings up the Synaptic Package Manager, which can perform any package management task. Expert users probably will go straight to Synaptic or aptitude, but beginners will appreciate this feature.

The Ubuntu packages are divided among four components: main, restricted, universe and multiverse. With all four package components enabled, an Ubuntu system has access to more than 16,000 different packages. Packages that are installed by default are listed in the main or restricted components. Main contains completely free software, plus some fonts and binary firmware files that are redistributable but not actually free software. Restricted contains non-free proprietary software distributed with restrictions, such as NVIDIA video drivers.

Ubuntu is free to distribute, install and use, and the restricted packages are essential to make a distribution that simply works, out of the box, on all common hardware. If you want to avoid proprietary software, you can remove the restricted component from your package sources.

The universe and multiverse components are disabled by default. Universe contains many thousands of packages from Debian GNU/Linux, compiled for Ubuntu but tested very little and not supported. Multiverse contains proprietary packages, such as Adobe Acrobat Reader 7.

Using Ubuntu

Ubuntu comes standard with a solid assortment of software—OpenOffice.org office suite, The GIMP image editor, Evolution e-mail client and Firefox Web browser—all the basics you would expect to find on a modern Linux desktop system by default.

Using the Synaptic Package Manager you easily can search through the thousands of packages and select new ones; a click downloads and installs them. It's really fun to browse through the package listings, shopping for new software. Any software that Ubuntu does not install by default can be added easily, which is a real strength of the APT package management system.

Before you use Ubuntu, I suggest you look over the tips collected on the Ubuntu Guide Web site. It's a treasure trove of useful information.

A major hole in GNOME 2.10, however, is the lack of a menu editor.

Figure 5. The Add/Remove Programs dialog showing office applications.

Figure 6. This is the dialog to set custom key shortcuts, with actions bound to the multimedia keys on my keyboard.

GNOME 2.10 adopted the new freedesktop.org menu standard, so older menu editors don't work, and there simply wasn't a new menu editor available to ship as part of GNOME 2.10. However, all of the packages in the Ubuntu base system are good about putting launchers in the menu, so the typical Ubuntu user does not need a menu editor. If you want to install a menu editor, you can install the KDE Menu Editor (provided by the kmenuedit package) or follow the step-by-step instructions from the Ubuntu Guide Web site to install a simple GNOME Menu Editor.

The six-month release cycle may cause this sort of rough edge to appear again in the future. But given how easy it is to update an Ubuntu system, any real problems that turn up can be fixed with updated packages. For example, once there is an official Ubuntu menu editor, all Ubuntu systems will get it when they update their packages.

If you want to use the universe packages, I suggest you set up the Debian menus. The universe packages may not add menu entries to the GNOME desktop menu, but they almost always add entries to the Debian menu. Install the menu and menu-xdg packages, and the Debian menu appears under Applications/Debian.

Ubuntu does not come standard with support for legally encumbered media technologies such as MP3 audio or MPEG-2 video. The Restricted Formats page on the Ubuntu Web site discusses the situation.

For system administration, Ubuntu encourages you to use `sudo`. By default, no root password is set. You can get a root shell by running `sudo -s`.

Support and Community

Paid support for Ubuntu is available directly from Canonical, and it is the only way Canonical profits from Ubuntu. Canonical offers full support for packages in main, limited support for packages in restricted and no support for software in universe or multiverse. In addition, companies other than Canonical are offering support for Ubuntu. A list of support companies is available on the ubuntulinux.org Web site.

Ubuntu has a large and growing community. Much of the documentation is community-written Wiki documentation. There are also Web forums, mailing lists and an IRC channel.

Conclusion

Ubuntu Linux is an excellent choice for anyone who wants to run Linux on a desktop system. It's easy to install and to administer. Everyone from beginners to experts can use and appreciate it. And it's free. If you are looking for a new Linux distribution, give Ubuntu a try.

Resources for this article: www.linuxjournal.com/article/8325.

Steve R. Hastings first used UNIX on actual paper teletypes. He enjoys bicycling with his wife, listening to music, petting his cat and making his Linux computers do new things.

Debian and Ubuntu have a close relationship. Ubuntu is built on top of Debian, using Debian tools and starting with Debian packages. However, the two projects cannot mesh perfectly.

Debian supports many different architectures, including ones considered obsolete, such as Motorola 68K; Ubuntu currently supports only three. Debian eschews hard deadlines for releases, while Ubuntu has committed to making a release every six months. Ubuntu already has transitioned to X.org's X11 system, while Debian still is using XFree86 release 4.3. Many packages from Ubuntu would install poorly on a Debian system, and vice versa.

Are Debian and Ubuntu fated to drift even farther apart? Debian cannot match Ubuntu's six-month release cycle without making major changes, and it probably shouldn't try. But once Debian finishes its next release, it likely will update Debian from Ubuntu, bringing the two projects somewhat closer together again.

The Debian Project and the Ubuntu Project have similar aims. Some of the Ubuntu developers are Debian developers too, and improvements and bug fixes done for Ubuntu are fed back into Debian as much as possible. Although other Debian-based distributions of Linux have branched off completely from Debian in the past, Ubuntu is making an effort to maintain closer ties.

Building the Perfect PC

by Robert Bruce Thompson & Barbara Fritchman Thompson

O'Reilly | ISBN: 0-596-00663-2 | $29.95 US; $43.95 CAN

As Linux users, we're used to cracking open our cases to modify our computers, but as *Building the Perfect PC* shows, this practice is no longer merely for techies. In fact, many ordinary people are building PCs from scratch. A grandmother that one of the authors met at a big-box store was in the process of building her third PC—this time for her granddaughter.

You may be comfortable hacking together a device driver without having the specs available. But, if you are like me, you might feel tentative about plugging an expensive CPU in to a motherboard. If so, then this book is for you.

Building the Perfect PC has a larger-than-usual format than other O'Reilly books. The larger size is due to the margins being filled with photos illustrating the proper method for putting together various components. So that's how the thermal compound is applied!

The book cites many reasons why you would want to build your own PC, including lower cost, broader options, better component quality and no bundled software. Most interesting to me, though, is the ability to build PCs for specific purposes. Not only does this book teach readers how to build mainstream PCs and SOHO servers, but there are chapters on building "Kick-Ass LAN Party PCs" and home theater PCs.

Each project is contained in a chapter that starts with a section called "Determining Functional Requirements and Hardware Design Criteria". When it comes to component considerations, the authors are not shy about recommending products by brand name. They don't claim that their recommendations are the only good choices, but they want you to benefit from their experience and research. After you're done designing your system, you're ready to build. The bulk of the chapter then guides you through building the system and offers many photographs and helpful explanations for doing so.

The Definitive Guide to Linux Network Programming

by Keir Davis, John W. Turner and Nathan Yocom

Apress, 2004 | ISBN: 1590593227 | $49.99 US

As the title claims, the scope of *The Definitive Guide to Linux Network Programming* is broad. The authors take a hands-on approach, and each chapter contains concrete programming examples of varying sizes and complexities. The three main sections cover fundamental networking concepts, alternative design architectures and security. The book also contains an appendix on IPv6. In addition, all of the code can be downloaded from the publisher's Web site.

Many of the concepts presented in the book are quite general and not limited necessarily to Linux. Hence, the book can be used as a concise introduction for developers new to networking and socket programming. Intermediate-level developers, on the other hand, could benefit from the explanation of architecture and performance. For instance, the book contrasts multiplexing, pre-forking and multi-threading server designs. Simple yet effective guidelines help developers make their design decisions.

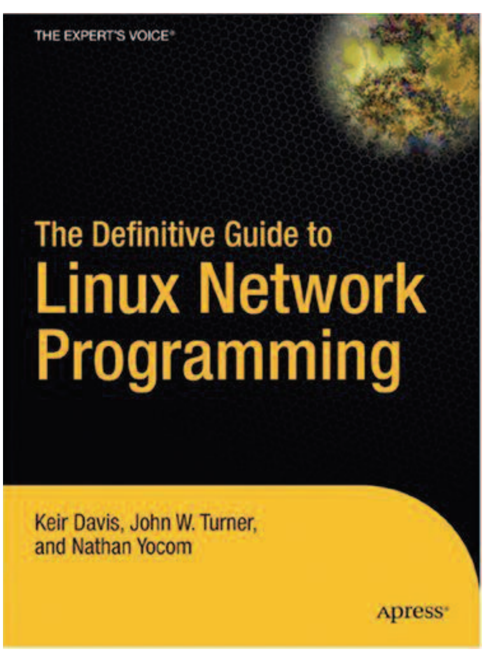
The material in the book typically is presented in a self-contained manner, but you do need to be familiar with C. Also, in explaining a few points, the authors rely on C++ and advanced libraries in order to provide more realistic coding examples. For instance, a GUI chat example uses the C++ Standard Template Library (STL) and the Qt graphical library.

Roughly a third of the book discusses how to secure code at different levels, from buffer overruns to authentication. Developers should consider security to be an essential activity, on the same level as debugging and performance tuning. The book also contains a section that briefly introduces tools for automated code analysis. These can be useful instruments to improve code quality and application stability.

The book does have a few shortcomings. Because of its introductory nature, the descriptions of several topics may be confusing. At a minimum, some topics, including non-blocking sockets and OpenSSL BIO, may require further reading if you are interested in a more in-depth understanding.

The book doesn't have too many technical details about configuring software, but that kind of information is available elsewhere. At times the chatty style of the authors seems a little more suited to a magazine article than a book. But if you're looking for a friendly guide to putting together hardware, I recommend this book. If you read it, you soon will be inspired to put together your own project, perhaps the home theater. The results will be better, more flexible and less expensive than any product you can buy ready made and off the shelf.

—JOHN KACUR

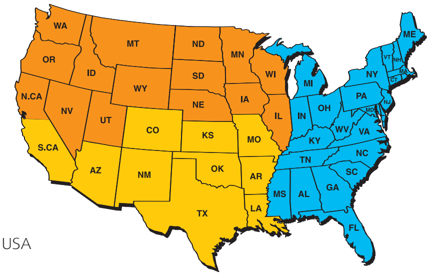


In addition, the book has no bibliography, and only limited pointers are offered to additional reference materials. Not-so-experienced programmers might benefit more from a more critical analysis of the code proposed in the book through exercises or extensions. Finally, the code examples contain some errors. The publisher's Web site has yet to make available the book's correction list.

—ANTONIO MAGNAGHI

LINUX JOURNAL

PO Box 55549
Seattle, WA 98155-0549 USA
www.linuxjournal.com



ADVERTISING SERVICES

VP OF SALES AND MARKETING
Carlie Fairchild, carlie@ssc.com
+1 206-782-7733 x110,
+1 206-782-7191 FAX

FOR GENERAL AD INQUIRIES
e-mail ads@ssc.com
or see www.linuxjournal.com/advertising

Please direct international advertising inquiries to VP of Sales and Marketing, Carlie Fairchild.

REGIONAL ADVERTISING SALES

NORTHERN USA
Joseph Krack, joseph@ssc.com
866-423-7722 (toll-free),
866-423-7722 FAX

SOUTHERN USA
Laura Whiteman, laura@ssc.com
206-782-7733 x 119

EASTERN USA AND CANADA
Martin Seto, mseto@ssc.com
+1 905-947-8846,
+1 905-947-8849 FAX

Advertiser	Page #	Advertiser	Page #
AML www.amltd.com	37	LinuxCertified, Inc. www.linuxcertified.com	91
APPRO HPC Solutions appro.com	41	Linux Networx www.linuxnetworx.com	39
Arkeia Corporation www.arkeia.com	31	LinuxWorld Conference & Expo linuxworldexpo.com	83
ASA Computers www.asacomputers.com	65, 67	LPI www.lpi.org	79
Avocent www.avocent.com	17	Mazda Technologies www.mazdatechnologies.com	21
Cari.net www.complexdrive.com	57	MBX www.mbx.com	27
Ciara Technology www.ciara-tech.com	8, 9	Microway, Inc. www.microway.com	C4, 89
Coraid, Inc. www.coraid.com	51	Novell www.novell.com	45, 47, 49
Coyote Point www.coyotepoint.com	25	Penguin Computing www.penguincomputing.com	59
Cyclades Corporation www.cyclades.com	C2, 1, 11	PFU Systems www.pfusystems.com	92
D.E. Shaw Research and Development www.deshaw.com	29	The Portland Group www.pgroup.com	18, 19
Emac, Inc. www.emacinc.com	94	R Cubed Technologies www.rcubedtech.com	61
EmperorLinux www.emperorlinux.com	15	Rackspace Managed Hosting www.rackspace.com	C3
Etnus www.etnus.com	2	Roaring Penguin Software, Inc. www.roaringpenguin.com	22
FairCom Corporation www.faircom.com	69	SAG Electronics www.sagelectronics.com	88
Free Software Foundation www.gnupress.org	75	SBE, Inc. www.sbei.com	5
Geek Cruises www.geekcruises.com	85	Servers Direct www.serversdirect.com	71
Google www.google.com/lj	50	Software Development Conference & Expo www.sdexpo.com	87
HP World 2005 Conference & Expo www.hpworld.com	73	Swell Software, Inc. www.swellsoftware.com	93
Hurricane Electric www.he.net	23	Tatung Science & Technology www.tatung.com	13
Iron Systems www.ironsystems.com	77	Technologic Systems www.embeddedx86.com	35
Iwill Usa Corp www.iwillusa.com	53	TUX magazine www.tuxmagazine.com	78
Layer 42 Networks www.layer42.net	39	Tyan Computer USA www.tyan.com	63
Levanta www.levanta.com	55	ZT Group International www.ztgroup.com	33

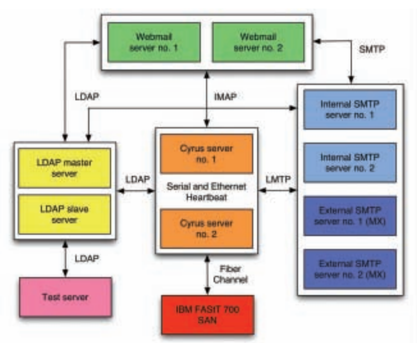
Editors' Choice Awards 2005

BY DON MARTI

SERVER HARDWARE

IBM eServer xSeries

When he's not writing for *LJ*, Ludovic Marcotte is architecting big enterprise IT projects, including last year's 35,500-user mail project at the Canadian business school HEC Montréal. He chose IBM eServer xSeries x305 and x335 servers for the project and recommends the server line for Editors' Choice. Systems are available in all sizes from blades up to a 32-way xSeries 445.



Each box in this enterprise mail project is an IBM eServer xSeries system.

This IBM eServer 336 is the new model in the eServer xSeries, replacing the discontinued x305 and x335 mentioned in last year's article.

PERSONAL COMPUTER OR WORKSTATION

Apple/Terra Soft PowerMac G5

Robert Love writes, "Fast, beautiful and it even runs Linux." Don't forget "quiet". With fans under software control, this box will run only as loud as it needs to in order to stay cool. The idea is as simple as a thermostat, and we're surprised more manufacturers don't do it. Terra Soft Solutions sells the G5 with Linux pre-installed, including the driver for the fans. Based on the POWER architecture and the PCI-X bus, this system's other features include Gigabit Ethernet, Serial ATA and two FireWire interfaces.

Terra Soft pre-loads this Apple G5 with Linux.

SECURITY TOOL

Max Moser and Contributors, Auditor Security Collection

Mick Bauer calls this Knoppix-based bootable distribution, "the best one for network scanning, particularly wireless and Bluetooth scanning." He adds, "If you need to validate the security of your networked systems periodically, or even if you perform security assessments for a living, Auditor provides most of what you need to do the job, especially if you don't want to dedicate hardware for the purpose." You don't need to set up a disk partition or, worse, transfer sensitive data over the network. Use a USB drive or some other removable media to take your security data out and take it with you.

Honorable mention goes to OpenSSH. Paul Barry writes, "It really comes into its own when I combine it with one of those bootable/live Linux CD distros (I use Morphix). When supervising student lab sessions, I can pop Morphix into any PC on campus, reboot into Morphix, open up a terminal, do an `ssh -C -X -l barryp` to my main office desktop and keep working. All my apps and my environment are right there with me. And, of course, my traffic is nicely encrypted, so any students running sniffers can't see what's going on."

WEB BROWSER OR CLIENT

The Mozilla Organization, Firefox

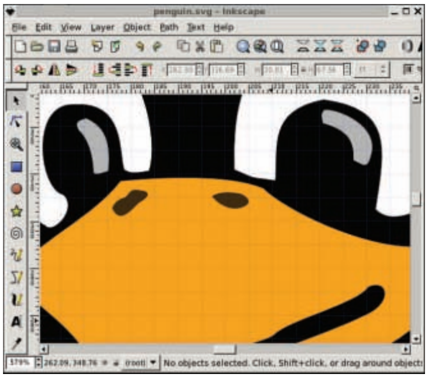
Robert says, "Firefox isn't just a great browser, it is a great example of doing a cross-platform project that everyone, on every platform, loves." You can tell when hackers love something by the volume of tweaks, add-ons and extensions. Nigel McFarlane covered configuration hints in the April 2005 issue, and watch for more on our favorite Firefox extensions coming up soon.

Thanks to Firefox, the Mozilla Organization dethroned Microsoft as the number-one browser source for linuxjournal.com readers too. Mozilla browsers, not counting old proprietary Netscape, rose from 28.1% to 44.4% since last year.

GRAPHICS SOFTWARE

inkscape.org, Inkscape

Ludovic writes, "I always missed a good tool like CorelDRAW on Linux, but I think Inkscape is one truly great scalable vector graphics editor." Vector graphics aren't only for print these days—with users' browsers ranging in size from mobile devices to multi-monitor desktops, you're going to need graphics that look good at a variety of sizes no matter what you use them for.



Inkscape lets us zoom way in on this SVG penguin, drawn by Nicu Buculei for OpenClipart based on Larry Ewing's original design. Look, Tux, no jagged pixels!

COMMUNICATION TOOL

Ryan Boren, Matthew Mullenweg and Contributors, WordPress 1.5

Reuven Lerner writes, "After trying different Weblog software (in my column, on my server and on my desktop machine), I chose to go with WordPress for my own work, as well as to recommend it to others. The release of WordPress 1.5 several months ago demonstrated that the project has reached maturity. Not only is the code solid, but it's easy to install, easy to use, has a plugin architecture that's simple to work with and can be extended in a number of different ways by programmers and non-programmers alike."

We're seeing more and more WordPress blogs—especially from smart people who aren't full-time Webmasters and just want to get a virtual host, drop in a blog package and go.

DESKTOP SOFTWARE

Novell Evolution

Call it a mail and calendar program or a "groupware client", this software plugs you in to collaboration with your coworkers, even if they're still running a legacy mail server. Evolution saved Paul from having to switch desktop environments. He writes, "I hadn't looked at it until work recently made the move to Microsoft Exchange and 'gently forced' everyone to get their e-mail through the truly awful 'Outlook Web Access'. I opened up Evolution, pointed it at the Exchange server and kept on using my preferred working environment: Linux." For keeping your Linux desktop afloat in a sea of proprietary jibber-jabber, we salute you.

Our runner-up in the desktop category is GnuCash. Reuven writes, "Accounting software doesn't have the flash or appeal of many other desktop applications. Moreover, it has an even greater responsibility to get everything perfectly right. And the ability to create your own reports, record regular transactions and synchronize your accounts with OCX files from your bank makes it even more useful." As a bonus, the documentation provides a non-accountant's friendly intro to how double-entry bookkeeping works.

LANGUAGE

C# Language Design Team and The Mono Project, C#

Robert writes, "Finally, a usable, fun, rapid-development-yet-powerful language for Linux, with excellent GNOME and Gtk bindings." You can tell a good language by one simple test: do people write great original software in it? For C#, the answer is yes, as you'll learn from a quick Beagle demo. Beagle, written in C#, is "a GNOME-based search infrastructure that ransacks your personal information space to index and find whatever you are looking for instantly", Robert writes. While you work, it watches you and comes up with relevant and potentially helpful information. And it provides a counterexample that will help you put the tired "open-source desktop software only copies proprietary apps" argument to rest.

SOFTWARE LIBRARY OR MODULE

Simon Cozens and Sebastian Riedel, Maypole

Don't give yourself a repetitive strain injury pounding out thousands of lines of scripting language, HTML and SQL to create a Web app. You'll only have to maintain it later.

Paul did it smarter for our March 2005 issue—in 18 lines, thanks to Maypole. And others are catching on too. "I've had a number of readers contact me via e-mail with queries about my '18 lines of code' article. They are all new to Perl but are still willing to give Maypole a go, which is a great sign", he writes, and adds, "I think Jerry Pournelle (from *BYTE* magazine) used to have a saying for stuff like this: infuriatingly excellent."

DATABASE

PostgreSQL Global Development Group, PostgreSQL 8.0

More and more organizations are working with high-end database systems but can't afford, or don't want, a full-time database administrator. PostgreSQL complies with SQL standards but needs less babysitting than complicated legacy databases. Ludovic calls it, "easy to install, configure and relatively easy to tune for performance." In our June 2005 issue, he covered Slony-I, which adds replication to PostgreSQL, giving you multisite redundancy, increased performance or both. Reuven points out that PostgreSQL has programmer-friendly features, which for 8.0, include server-side scripting in Perl.

MANAGEMENT OR ADMINISTRATION SOFTWARE

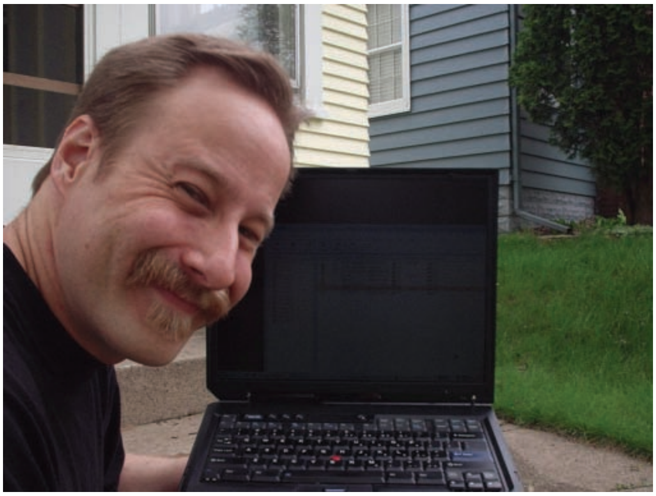
Alfredo K. Kojima, Michael Vogt, Gustavo Niemeyer and Contributors, Synaptic

Paul is happy with the Ubuntu distribution, and one reason is this "embarrassingly easy-to-use" tool for installing software and keeping it up to date. Click what you like, and Synaptic will install it with all dependencies—even browse the documentation so you know what you're getting. More info in our Ubuntu review on page 72.

MOBILE DEVICE

IBM and EmperorLinux, IBM ThinkPad T series/EmperorLinux Toucan

Ludovic praises this system for its "excellent level of compatibility with various Linux distributions" including Fedora, Red Hat Enterprise Linux and Ubuntu. Several *Linux Journal* editors are happily using these, and all the features



Mick Bauer won't put security tools on a critical server—he carries them to the job site on an IBM ThinkPad or a bootable CD, and removes them when he's done.

work under Linux. We're all about ThinkPad keyboards.

The ThinkPad line still lags the market leaders in one key area, though: availability with Linux pre-installed. After success with Linux on the nx5000 laptop, HP now offers Linux across the board—but not listed on the Web site. You have to call and order it via "Factory Express".

This will be the last year that IBM is eligible for this award, as it has sold off the ThinkPad business to Lenovo. Maybe the brand's new owner will be more accommodating with the Linux preloads.

GAME OR ENTERTAINMENT SOFTWARE

Jasmin F. Patry and Contributors, *Tux Racer*

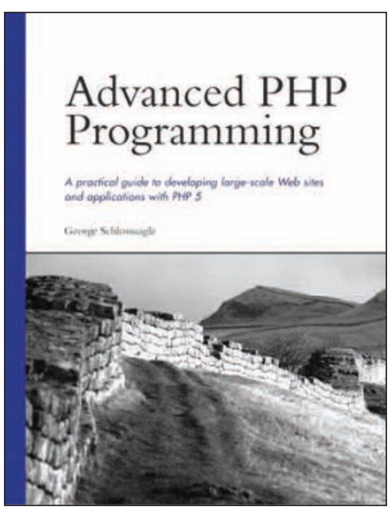
With more than a million downloads and a stack of awards on the home page, this game doesn't need yet another one. But we're going to give it anyway. Flop on the ice and race to grab all the fish you can in this easy-to-learn game that your little penguins can play too.

This is the first GPL game to be released in an arcade version. Innovative Concepts in Entertainment calls its 400-pound cabinet a "Dazzling children's racer with adorable penguin character."

DEVELOPMENT BOOK

George Schlossnagle *Advanced PHP Programming*

Reuven writes, "This is not a simple 'here is how to write a Web application' book, but rather a book that teaches you how to think about Web applications before you deploy them. He doesn't just tell you that you should tune your database for the Web—he shows you design patterns for talking to the database server, so as to structure your code more readably and efficiently. He doesn't just tell you that authentication is important—he gives strategies for checking that the user hasn't been switched out from under you. Even if you don't program in PHP, this book is worth reading."

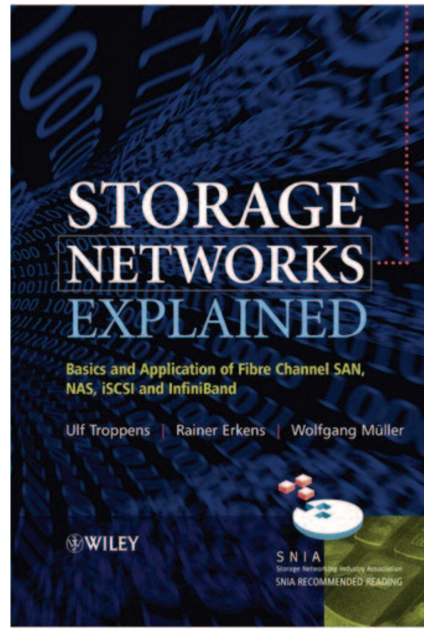


Think before you clobber your database server. Read this book to learn to develop efficient, maintainable Web applications.

SYSTEM ADMINISTRATION BOOK

Ulf Troppens, Rainer Erkens and Wolfgang Müller, *Storage Networks Explained*

Ludovic writes, "Finally a good book on SAN." This 432-page hardcover is full of storage network examples, including InfiniBand, and is well illustrated. The book is on the expensive side, but compared to SAN mistakes, it's a bargain.



Before you step up to a big-iron storage system, step up to this big hardcover storage book.

END-USER OR NONTECHNICAL BOOK

Paul Graham, *Hackers & Painters*

We started visiting paulgraham.com for the spam-fighting ideas, then came back for his other writing about hacking, business and culture. Now a collection of his essays is out in hardcover. Why do smart people tend to be "nerds" in high school? What business ideas did the dot-com bubble get right? And, perhaps most important, what should you look for in a programming language?

TECHNICAL WEB SITE

Eklektix, Inc., LWN

LWN wins again. At first glance, it looks like just another "meta-news" site with links to articles on the Web, Slashdot-style layout and comments. But look again. The clean layout is unpolluted by the annoying Macromedia Flash ads found on some Linux sites we could name, and comments come in from "subscriber gregkh" (kernel guru Greg Kroah-Hartman) and others who actually write the software we're all chattering about. LWN editor Jonathan Corbet helped plan the 2004 Kernel Summit, and LWN's coverage of the event was a must for anyone who needs to keep up with the kernel.

NONTECHNICAL OR COMMUNITY WEB SITE

Wikimedia Foundation, Wikipedia

Robert calls Wikipedia, "probably the single greatest thing on earth." It's hard to comprehend an encyclopedia with 1.5 million articles and editions in 195 languages, so just visit the site and click "random page". One visit yielded a history of Kincheloe, Michigan; an unfinished "stub" article about a political party in Suriname; a biographical entry on Admiral Walter F. Doran, Commander of the US

Pacific Fleet; and the ingredients and history of mortadella.

Why doesn't Wikipedia get cluttered up with flaming, drivel and spam like other on-line fora? Part of the answer has to be in the Wiki philosophy, where anyone can "edit this page" and put problems right, and part of the credit has to go to the MediaWiki software, which makes it easy for helpful people to find and fix vandalism.

PROJECT OF THE YEAR
freedesktop.org

On the Internet, any movement looks like a big argument. But forget all the arguing over K this and G that, and get plugged in to the grand unified master plan to clean up the ragged legacies of UNIX, advance the X Window System to keep up with leaps in hardware and put a secure, friendly GUI everywhere.

The list of hosted projects includes D-BUS, X.org and all the hard-to-get-right infrastructure such as vector graphics, fonts and internationalization.

Marco Fioretti wrote in our May 2005 issue, "If protocols and formats stop being tied to specific implementations or toolkits, they can be shared across multiple 'desktop environments'. Code stability and lightness would directly benefit from this, as would innovation. Completely new programs could interact immediately with existing ones."

PRODUCT OF THE YEAR
Ralink Technology Corp., RT2500 Chipset Solution

If binary-only 802.11g drivers are the rat dookie in your raisin bread, get a card based on the RT2400 or RT2500 chipset and be happy. Instead of giving other vendors grief over "take our word for it, it's a raisin" drivers, we're going to celebrate a company that gets it right. Ralink worked with Mark Wallis, Ivo van Doorn, Luis Correia, Robin Cornelius and others to get a supported driver out there under the GPL.

Paul writes, "On my aging laptop, I popped in the PCMCIA card, downloaded the source code and installed the device driver into Fedora Core 3 and—about two minutes later—joined the wireless revolution!" Special thanks to Minitar, the network gear vendor with the foresight to ask Ralink to make the driver GPL.

Resources for this article: www.linuxjournal.com/article/8332.

Don Marti is editor in chief of *Linux Journal*.

The Prime Internet Eisenstein Search

Plug in to an international project and discover new mathematical truths.

BY BOB BRUEN AND PHIL CARMODY

The Prime Internet Eisenstein Search, PIES, is a long-term effort to discover prime numbers. PIES is trying to exploit a property of a small class of numbers previously overlooked by other mathematicians, called Generalized Eisenstein Fermat Numbers. These Numbers have the newly discovered property that they are quicker and easier to prove prime than are typical numbers. Also in their favor is the fact that they are exceptionally dense in primes, more so than the candidates in any other prime-hunting project.

The PIES Project is orchestrated by Phil Carmody, a British mathematician living and working in Finland. Phil is the mathematician who discovered, back in 2001, the first "illegal" prime. This prime number can be unpacked into the original source code for DeCSS, the software that decodes the DVD encryption scheme. He also has discovered a second prime number that actually can execute the code.

Contributors to PIES come from the US, Canada, Finland, Germany, France and a couple of other places around the world, although it is a relatively small international project. In true Linux form, the project is based all on volunteer work, runs on a small budget, is international and produces real results. The goal is pure research and somewhat esoteric—the discovery of large prime numbers of a slightly unusual form.

Prime Numbers

Prime numbers are those numbers that can be divided by 1 and themselves only. The numbers 1 and 0 are not considered prime, and the number 2 is the only even prime number. Primes are a fundamental part of our numbering system, and the search for prime numbers has fascinated mathematicians for more than two millennia. Today, prime numbers are used for public-key encryption, and large prime number searches are computationally intensive. The world's largest primes all are archived at Professor Chris Caldwell's "Prime Pages", hosted at the University of Tennessee at Martin. Prime Pages not only archives the world's largest primes, but it also is the world's most complete resource for information on prime numbers.

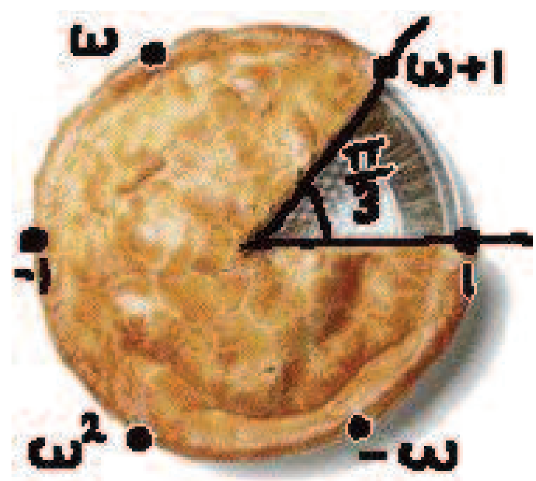


Figure 1. PIES Logo

The simplest method of determining whether a number is prime was understood by the ancient Greeks. Simply divide the number by primes smaller than the square root of the number being tested. Doing so finds all factors of the number; if none are found, the number is prime. This works reasonably well if your numbers are small, but when they get large, you need to be a bit smarter about how you search, calculate and prove that the number is indeed prime. Finding what you believe to be a prime number is not enough. Mathematicians are required to provide proof.

Bernard Riemann gave a lecture in 1859 in which he proposed a way to count prime numbers as a general rule. Proving what is known as the Riemann Hypothesis was one of the great mathematical challenges of the last century, and it continues to be so in this century as well. Trying to figure out how many primes are in a range and what the distribution looks like within that range is an active area of research that helps drive the search for prime numbers.

Prime numbers are a kind of backbone for our number system. The use of prime numbers is more than simple intellectual play for mathematicians. Once Ron Rivest and his colleagues figured out that prime numbers were the way to make Whitfield Diffie's idea of asymmetrical, or public-key, cryptography a reality, prime numbers became indispensable. The more security required, the larger the prime numbers have to be.

The PIES Project

The mathematics behind the PIES Project is somewhat esoteric and is explained partly on the project home page. It shares some properties with other large-prime-hunting projects, namely that it is a cyclotomic form, that is a factor of a^b–1. Other cyclotomic forms are Mersenne (2^p–1) and Generalized Fermat Numbers (b^{2^n}+1). The PIES primes are the first of the cyclotomic forms that can be found in large sizes, in large quantities and quickly but are not explicitly of the form a^b-1 or a^b+1. This particular PIES form, Generalized Eisenstein Fermat numbers, was first looked at in-depth by English amateur mathematician Mike Oakes several years before PIES started. But, it was because of Phil Carmody's advances in sieving—that is, quick removal of obvious non-primes because they have small, easily found factors—and fast primality testing algorithms that it became practical to look at the larger numbers with which PIES currently is working. Cyclotomic numbers are what you get from evaluating cyclotomic polynomials. The nth cyclotomic polynomial is denoted by Phi(n), and its value at b is denoted by Phi(n,b). Mersennes are Phi(p,2), and Generalized Fermat Numbers (GFNs) are Phi(2^n,b). The PIES Generalized Eisenstein Fermats are Phi(3*2^n,b).

Dr David Broadhurst of the Open University has been watching the development of the PIES Project with interest, although he has not devoted any cycles to it. When asked for his opinion, he said:

> This is good maths, good programming and good fun. Phil Carmody managed to enliven Professor Chris Caldwell's database of the top 5,000 proven primes. Previously it consisted almost entirely of strings ending with -1 or +1, since those forms were tuned to existing primality proving programs. Now, Phil and his friends have added several hundred entries beginning with Phi, which is math-speak for a cyclotomic polynomial, albeit a rather simple one in this case, based on the cube roots of unity. Phil was able to do this without losing processing speed. In fact, he even may have gained speed on rivals, thanks to specific properties of the two cube roots of unity that are complex numbers.

Although Phil is serious about mathematics and his various projects, he does it all for fun. His somewhat unusual sense of humor can be seen on the PIES Project home page. He believes that PIES is the only distributed computing project with a project song, for example. As one might guess from how the project name doesn't quite seem to parse correctly, it is indeed a complete contrivance, done simply so that the project name was fun and the search could be "themed". Each fixed value of n in Phi(3*2^n,b) defines a band in which primes can be hunted as b varies. Phil calls the small n=13 range "cherries", the n=14 range "peaches" and the recently started n=15 range "apples". Only he and his girlfriend, Anna, who assists with the project's image, words and song lyrics, know what the upcoming ranges will be called.

Distributed Computing

The work for such prime number finding projects falls into two main areas. First, the head-scratching is performed by the mathematicians to determine how to find prime numbers and prove they are prime. If necessary, this step involves writing custom programs that are optimal for the task at hand. The second part is the computational work involving network communications and systems management. It makes for a productive partnership, with little of the overhead that accompanies larger projects.

Most large primes are found by distributed computing projects, as can be seen from the top finders' tables on the Prime Pages. Therefore a real but friendly sense of competition exists among projects and also among individuals involved. Both get scores and are ranked by discovering large numbers, the most numbers and numbers with particular special forms. For most of 2004, PIES was the single largest project by count of prime numbers, as it was working on a hugely fruitful band of small prime numbers, of about 50,000 digits. Alas, all those primes have dropped off the Prime Pages' Top 5,000 list, and the project now is only the third largest producer by count of primes. Phil considers the ranking by count to be not particularly important—large quantities of small primes are not particularly challenging. They also are a bad investment, as they don't stay on the list long.

Probably the best known search project is the Great Internet Mersenne Prime Search (GIMPS). This project is seeking the largest Mersenne prime number, which is, at the moment, also the largest prime number of any form. In February 2005, the largest known prime number, with 7,816,230 digits, was discovered. The calculations took 50 days on one 2.4GHz machine, and independent verification required an additional five days. A second verification took 15 days. The discoverer, Dr Martin Nowak from Germany, joined GIMPS six years ago. In essence, he has been calculating for six years to find this one number, only the 42nd Mersenne prime found. The 41st was discovered in May 2004; the project has found only eight since 1996. GIMPS lists about 41,000 people involved in the calculations, many of whom allow their personal machines' idle CPU cycles to be used to crunch numbers. Other participants have large academic or commercial facilities at their disposal, helping the GIMPS global network sustain more than 17 teraflops.

According to Professor Caldwell, Phil has implemented an important advance by looking at numbers that often are quicker to test for primality than are the usual numbers. In a decade or so, such a project may be able to compete seriously with GIMPS for the primes of record size. This would happen not because they are somehow better projects but because the Mersenne numbers steadily thin out, and many other forms don't thin out so quickly. Even when the Mersennes once again lose their lock on the largest known prime, they may stay the most important primes because of their long history.

Professor Caldwell, who happens to run his Prime Pages on Linux, said, "PIES quickly has established itself as a major player by finding a large number of primes in the 100,000-digit range. I myself am a PIES member, and I really appreciate the thought and effort Phil has put into his system." There are a number of other projects, some even in the teraflop category. In contrast to these larger projects, PIES is working on a smaller scale and within smaller ranges of digit size. More numbers are being discovered more quickly, in a sense backfilling from the high Mersenne numbers. In the list of the largest known prime numbers, PIES has reached 191,362 digits as of April 4, 2005, but expects to find a new larger prime roughly once a week.

For simply attracting project members, the ideal distributed computing setup is client-architecture neutral. All of the largest distributed projects work equally well with Linux, *BSD and other OSes running on the clients. Phil decided to use Perl to write both his client and his server, as it provided all of the networking primitives that he needed and is secure by design. The task of actually exchanging assignments and results is such a small part of the whole project that a p-coded, or compiled into interme-

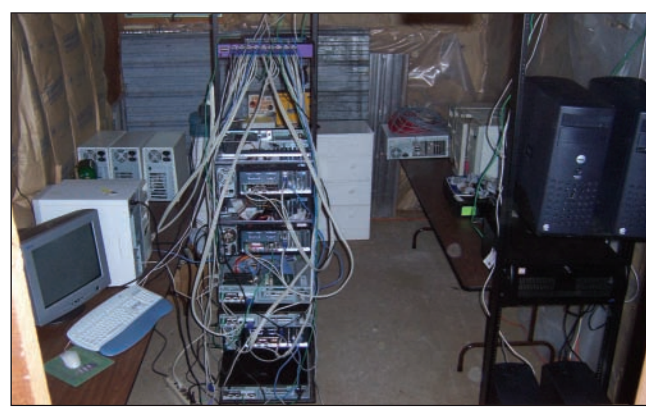


Figure 2. PIES Computational Facility

Figure 3. Bob's Barn

diate form, language such as Perl is perfectly fast enough.

Phil intends to adapt his server so that it can be used for arbitrary distributed prime-hunting projects. He then plans to release it as free software.

Linux Computational Facilities

PIES runs almost exclusively on Linux machines. Linux has enabled the installation of a reliable OS across many machines, and an individual license for each machine would be a significant cost. Linux administration is easy, making it possible for one part-time person to administer a cluster, and a lot of admin tools are available. Linux works well on almost any hardware, which means a 200MHz machine can be used as a subnet gateway or a DNS server, further saving money. Inexpensive hardware and a free OS give the hobbyist the ability to run advanced facilities that produce first-class results.

Phil runs the project's central server from an Alpha machine at his home. He first used Linux as his OS of choice in late 1993 and turned his back entirely on what one might call high-street operating systems about five years ago. He has several other Linux machines, which he uses as clients. He typically develops for Linux only, but he doesn't discriminate against architectures—for example, he has enrolled several Alpha testers. He happily builds for BSDs and UNIX-alikes and begrudgingly for whatever else may be out there.

One of the US computational support sites is located in Vermont, in Bob Bruen's barn. The barn was built for horses, so there are stables and two open areas on the first floor and a large open space on the second floor. One of the two first-floor spaces now is the PIES computational facility. The facility was under construction before PIES to support work in Linux, networks and security. Rather than let the facility waste CPU cycles, Bob offered PIES access to the machines while they aren't working on other tasks. For a while now, there have been no such other tasks.

The Vermont facility is a dedicated cluster of more than a dozen machines ranging from 450MHz to 3GHz, with several SMP machines on a separate subnet in Bob's barn. The facility is linked by a wireless bridge to the house, which in turn has a cable modem connection to the Internet. The majority run Red Hat 9.0 and Fedora Core 2, but SUSE, Mandrake and Debian have been installed as well.

The hardware was purchased at auction or on sale, and it is server-class hardware: rackmount, heavy-duty case, some SMP, SCSI and a lot of memory. The same auctions yielded racks, switches and other hardware for pennies on the dollar.

Even with Linux as a foundation, there still are some problems. Although the auction-bought servers did not mind, one Dell SMP server failed in the extreme cold in the unheated barn. There were several days of temperatures 20–25 degrees below zero, Fahrenheit. Occasionally, an individual server has required attention, but by and large, as one would expect from Linux machines, they keep on running. The wireless bridge required a reboot once during the same cold snap. The most serious problem, however, has been the primitive electrical power in that part of Vermont.

One additional, small US facility is located in Arizona, where Steven Harvey, a criminal defense lawyer, runs PIES on Mandrake 10.1. He uses several machines for other prime number searches. Phil, a permanent resident of Espoo (near Helsinki), which is also home to the server and his few client machines, is working several hundred kilometers away, in Turku. Within a few days of moving there, he already investigated buying a handful of refurbished PCs. In order to keep costs minimal, he intends to buy systems with no hard or floppy drives, simply booting off a CD instead. Although Debian is his favorite distribution for desktop and server use, he plans to boot diskless machines with Knoppix.

Results

The outcome of the project can be seen on Professor Caldwell's prime number Web site at the University of Tennessee at Martin (see the on-line Resources).

The Vermont facility has found more than 50 prime numbers, including the six largest, in about 18 months. PIES overall has found more than 900 large prime numbers.

Nearly 100 more primes from 150,000–200,000 digits are expected from the current “apple” band. The next band, which won’t be started until the current band approaches completion, probably will contain at least 40 primes of between 300,000 and 400,000 digits. Presently, only 50 known primes of that size exist in the world. The PIES Project’s impact on the record tables, despite its current relatively small size, therefore is expected to be quite significant.

Resources for this article: www.linuxjournal.com/article/8273.

Bob Bruen teaches computer security and Linux at Springfield Technical Community College in Springfield, Massachusetts. He has been working with Linux for over a decade and has been the book review editor for *Cipher* for almost as long.

Phil Carmody is a 34-year-old mathematician who earned his degree from England’s prestigious Oxford University in 1991. When he isn’t coding for work or pleasure, he enjoys wordplay, live music and drinking single-malt whiskey and English beer.

LETTERS CONTINUED FROM PAGE 7

old, outdated page that has been quietly replaced. The old site, unfortunately, makes no acknowledgement of the new site, which resides at codespeak.net/icalendar. It contains newer versions, which amongst other things, add an installer.

--

Eric Windisch

Loop, Loop, Loop

Mike Mattice suggested that I deserve a UUOF award (Useless Use of For) [Letters, June 2005] for the for loop I used to list all the entries in a directory using only bash built-ins [“My Favorite Bash Tips and Tricks”, April 2005].

`echo *` will print all the filenames with only a space between them, wrapping the output to the next line as necessary. This can be harder to read when there are many files in the directory. The use of the for loop prints each filename on a line by itself.

Using a for loop also allows the insertion of a read after each echo, as a crude form of scroll control:

```
for file in *;
  do echo -n $file;
  read;
done
```

After each filename is printed, it will wait for you to press Enter before printing the next filename.

--

Prentice Bisbal

Cold Enough for You?

On a recent trip to the northern Montana town of Cut Bank I spotted this rather large penguin.

--

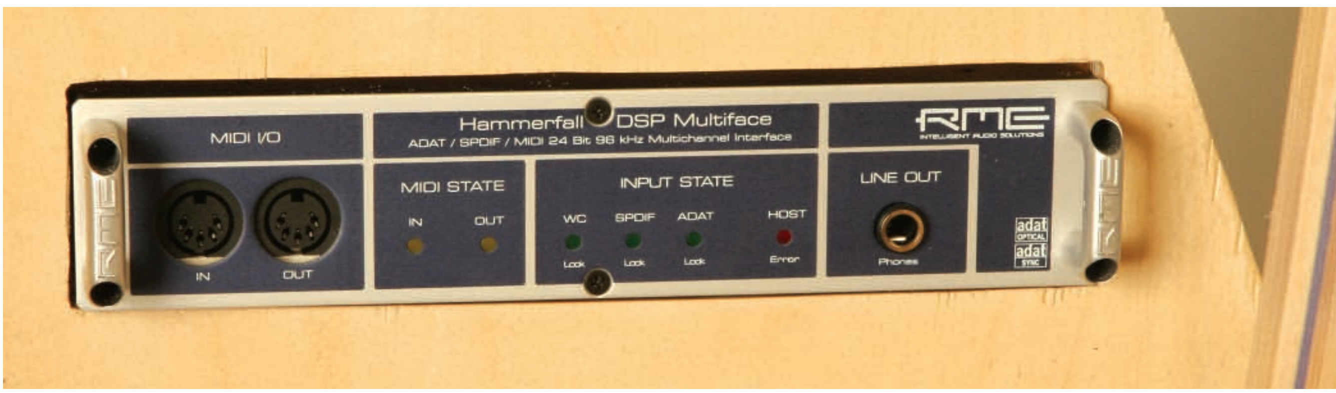
Herb

We welcome your letters. Please submit “Letters to the Editor” to ljeditor@ssc.com or SSC/Editorial, PO Box 55549, Seattle, WA 98155-0549 USA.

Inside the Ultimate Linux Box 2005

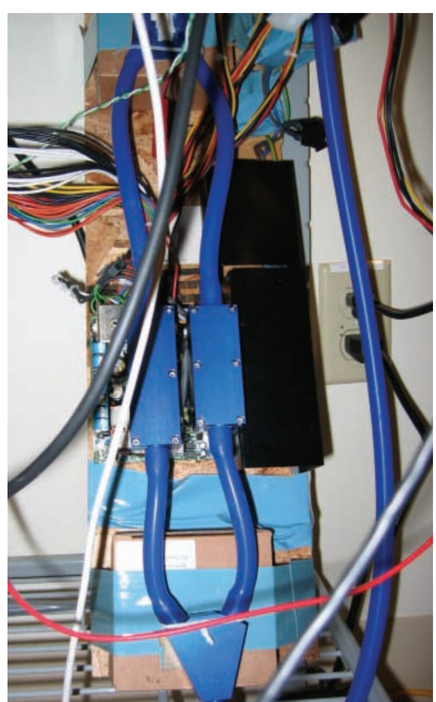
Turning the pages of this magazine makes more noise than this year’s Ultimate Linux Box does.

BY DON MARTI



The RME sound card uses this handy Multiface box to offer standard connections for digital and analog audio, along with MIDI. Because RME uses the same interface for its PCMCIA cards, you can take the same Multiface along to use with your laptop for

The Ultimate Linux Box has three separate cooling loops: one for the power supply and two that each handle two CPUs. We carefully monitored CPU temperatures with lm_sensors. CPU temperature rises a little before the water in the “up” tube warms up enough to start



With the heatsink fins milled flat, we were able to attach custom waterblocks for fanless cooling of the modified power supply, shown here mounted on a temporary rack for testing. The waterblocks and the custom Y-connectors are anodized blue to

The Ultimate Linux Box boots from a CompactFlash card with an ATA adapter. Pull the card out to make an easy backup. 256MB is plenty of space for /boot, and the rest of the storage is at the other end of a long fiber-optic cable. Going back to a noisy PC after using this machine was sure hard on the ears

Don Marti is editor in chief of *Linux Journal*.


Cyclades AlterPath™ System makes
AlterPath™ System
out-of-band manager
blade manager
console
kvm over IP
Intelligent power
IPMI
A+
For a FREE copy of Enterprise Management Associates' white paper Increased IT Operations Effectiveness With Cyclades AlterPath™ System for the Out-of-Band Infrastructure, please visit us at www.cyclades.com/ema