Linux Journal Issue #96/April 2002

Features

UNIX under the Desktop *by Doc Searls and Brent Simmons*
Doc and Brent speculate on the possiblities presented by the UNIX that is the latest Mac OS.

Build a Virtual CD-ROM Jukebox *by Jeremy Impson*
Jeremy shows how to set up a Linux server providing access to ISO 9660 images.

Connect to Microsoft SQL 2000 with the Perl Sybase Module *by Andrew Trice*
Thought you couldn't use Perl to interface with an MS SQL server? Think again.

Indepth

An Interview with Andreas Leimer *by Phil Hughes*
We talk to Inalambrica.net's CTO about how they use Linux to bring internet connectivity to Costa Rica.

Linux IPv6: Which One to Deploy? *by Ibrahim Haddad*
Ibrahim gives the dope on the various open-source IPv6 projects.

Toolbox

Take Command The m4 Macro Package *by Robert Adams*
Kernel Korner Hot Plug *by Greg Kroah-Hartman*
At the Forge Writing Zope Products *by Reuven M. Lerner*
Cooking with Linux Interoperate with Me *by Marcel Gagné*

UNIX under the Desktop

Doc Searls

Brent Simmons

Issue #96, April 2002

A penguin's-eye look at Apple's OS X.

When Steve Jobs introduced Apple's new iMac in January 2002, the spotlight was focused entirely on the physical architecture of the first mainstream computer that fully defied the term “box”. The new iMac is a white dome with a flat screen that floats on the end of a chrome arm. It looks like a cross between a Luxo lamp and a makeup mirror. Jobs called it “the best thing we've ever done”.

Coverage—including a *TIME* magazine cover story—was all about hardware. Nobody paid attention to Steve Jobs' slickest move of all, which is leveraging UNIX where it counts. Starting in January 2002, every new Mac will ship with OS X as its default operating system. OS X is built on Darwin, an open-source implementation of BSD on a Mach kernel. So now every new Mac is a Trojan horse that arrives with an invisible army of UNIX experts.

Regardless of the technical and religious differences that separate the many breeds of UNIX, expertise at one ports well to another: from Solaris to HP-UX to AIX to Linux to BSD to Darwin and OS X. If you want to hack, the environment is there—so are the tools and the community.

Put another way, OS X gives us the first popular desktop OS that fits into a prevailing Linux environment and also into the prevailing marketplace. On the bottom, it's UNIX. On the top, it runs Microsoft Office and the whole Adobe suite. This has its appeals.

In iDevGames.com, Aaron Hillegass writes:

> Tomorrow I will get on a plane. I'll have my PowerBook with me. On that flight, I can write Cocoa apps, PHP-based web sites, Tomcat web applications, AppleScripts or Perl scripts. I can use Project Builder, Emacs or vi. I'll have my choice of MySQL or PostgreSQL to use as a back-end database. I'll use Apache as my web server. And it is all free! If I'm willing to spend a little cash, I can also run Word or Photoshop. I may even watch a DVD on the flight.

The social effects of OS X on the Open Source community were already apparent at the O'Reilly Open Source Convention in July 2001, when slab-like Macintosh G4 Titanium laptops seemed to be everywhere. At one Jabber meeting, four out of the seven attendees tapped away on TiBooks, including Jabber's creator, Jeremie Miller. Terminal windows were scattered across his screen. When we asked what he was doing, he replied, “compiling code while I catch up on some e-mail”.

The growing abundance of OS X fruit on the UNIX tree creates new and interesting market conditions for Linux, along with every other UNIX branch. There are sales projections for six million iMacs alone. Many of these machines will be penetrating markets where Linux has strong incumbent server positions, such as science and education. Lawrence Livermore National Laboratory was once Apple's biggest customer and might easily reclaim the title. In January 2002, the state of Maine announced its intent to give a new iBook to every teacher and student in the seventh and eighth grades. All those kids will have their own UNIX machines. Consider the implications.

Life among Penguins

Is there a server market for OS X? It's worth noting that OS X Server has existed as a product for more than two years and has never attracted much attention. Also, while every new OS X Mac is ready to perform a variety of server functions, that's not why it sells. IT Manager and Mac columnist John C. Welch calls OS X an “okay server, mostly due to hardware limitations and immaturity”. Meanwhile he says, “Linux is an excellent server. It runs on more and better hardware than Windows can ever dream of, thanks to IBM and Sun.” So OS X is no threat to Linux in the server space. And it's utterly absent from Linux's other home turf, embedded computing.

Where OS X will succeed is in the one category where Linux has struggled for popularity (if not functionality) from the start: on the desktop.

Is this a problem? That was the question at the top of our minds when we visited Macworld in January 2002. To our surprise, the answer was quite the opposite. Not only were plenty of familiar Linux figures walking around kicking tires (approvingly, it appeared), but there were UNIX geeks wearing Sun and SGI

schwag as well. One Linux hacker told us OS X was “subversive” because it “seeds” the world with millions of open-source UNIX machines. Another said, “I can go to my Mom's, fire up her iMac, open a shell, ssh to my own server and get some real work done.” So the market logic of Linux and OS X appears to be AND, not OR.

Apple also has attracted some top talent from the open-source ranks. Brian Croll, who runs OS X engineering for Apple, was recruited from Eazel. Jordan Hubbard, the world's foremost BSD hacker (and a founder of FreeBSD), actually pitched his way into a job working on Darwin at Apple. After seeing OS X in preview form, he said “Hallelujah” and “This is what I've been waiting for the past 20 years....I never thought about working for Apple before, and now I was saying, How do I join?”

Working with the Open Source community is still new for Apple, and the relationship has been a challenge to the company's highly proprietary approach to intellectual property. But Apple has compromised on some issues. After hackers barfed on Apple's original public source license, the company issued a new one that the Open Source Initiative soon approved. Shortly after the new license was issued in January 2001, OS X product manager Chris Bourdon summarized it this way: “You can take Darwin and do anything you like. It's there for everybody.”

The Longest Review

We first tried OS X when it entered public beta in November 2000. At that time nothing in the last paragraph was anywhere in sight, and the new public source license was more than a month away. It was vastly different from the “Classic” Mac OS (which it ran under emulation) and from other desktops on both Linux and Windows. It was pretty, but also slow and notably devoid of drivers for CD-ROM burners, DVD players and other peripherals essential to the modern desktop computing experience. If we had reviewed it then, as we had planned, our thumbs would have pointed toward the floor. And we wouldn't have been alone. “Frankly, I think it's a piece of crap”, Linus Torvalds wrote in his book, *Just for Fun*, which was also in the works around that time.

Things got better the following spring, when Apple released the 1.0 version of the OS. But it was still slow and some key drivers were still lacking. In September 2001, version 10.1 was released with kernel version 1.4.1, which by all accounts was vastly improved. Soon plenty of drivers showed up, reports of kernel panics fell below noise level, and the OS was ready for prime time.

Since then a good thing has become steadily better, thanks in large part to what the open-source geeks have brought to the table. XFree86 and XonX

brought the X Window System to Darwin. Fink used Debian tools like dpkg and apt-get to build a package manager. Their site lists hundreds of available packages. In fact, you don't even need to run Darwin on Apple hardware, since an x86 version is available as well (from Apple, even). GNU-Darwin is already the leading non-Apple Darwin distribution. Watching all this happen, our thumbs began rotating toward the ceiling.

Then last week we put OS X on a Titanium laptop. It blew our minds.

When we plug in a second monitor (or a projector), the OS autodetects it, then lets us decide whether to make it mirror the laptop screen or operate as a contiguous extension anywhere on all four sides. When we plug in a FireWire drive or a USB storage device (such as a camcorder, an MP3 player or a digital camera), the device shows up on the desktop, in the appropriate applications and in the directory. When we soft-eject the FireWire drive, it goes away, unless something requiring it is in use—in which case the OS gives me a warning or a rebuke, never a crash.

Installing OS X (now at 10.1.2) was a snap. It immediately ran every one of several dozen Classic Mac applications that were already on the machine, including Adobe GoLive and Photoshop, both of which are big, resource-intensive programs. Running Classic and OS X native apps concurrently isn't a concern. Everything looks and acts just fine.

But OS X really performs when it comes time to move the machine. In the past this was an exercise in prophylaxis that required running down a checklist of things that might go wrong. Not anymore.

On its desk the Titanium is connected to the network through Ethernet and DHCP. The rest of the building is covered by an 802.11b wireless network. We can close the TiBook's lid while it's hooked up through Ethernet, watch it go to sleep instantly, unplug everything (seven cables, including USB, FireWire, Ethernet, speakers, power and monitor) and watch it wake up instantly when I open it up on the kitchen table, where it's now connected to the network, wirelessly. Later we can close the lid, return it to the desk, plug everything back in, open it up and watch it wake instantly and carry on as before. We've been doing this several times a day with no ill effects.

Not that life is perfect. Every so often the laptop starts to slow down. Then we usually look for responsible processes. (Memory, incredibly, feels like a non-issue, even though we have “only” 384MB.) Apple provides a helpful GUI uitility called Process Viewer, but we'd rather do it the right way: through a terminal session. We just run **top** or **top -u**, isolate CPU hogs and kill them off. In UNIX fashion, life goes on.

The usual culprit is TruBluEnv, which is part of the Classic emulation environment. Apple obviously still has some work to do there. But again, failures are never catastrophic and are far less common than they were with the Classic Mac OS, which would typically crash several times a day.

Want to know the killer app for OS X? It's the uptime command. Right now it says this about the Titanium: “11:21AM up 6 days, 16:54, 3 users, load averages: 1.81, 1.55, 1.36”. On another desk is a G4/500 dual-processor desktop, also running OS X. There uptime says, “11:22AM up 22 days, 1:50, 2 users, load averages: 0.16, 0.04, 0.00”. On that machine I'm also running a little program called CPU Monitor, which shows activities on both CPUs. Pretty cool.

The New Beginning

In his book *In the Beginning Was the Command Line*, Neal P. Stephenson compared Linux, Microsoft and Apple to car dealerships. Linux consisted of volunteers making tanks that were free for the taking. Microsoft made ironically popular but failure-prone station wagons and SUVs. Apple made “expensive but attractively styled cars with their innards hermetically sealed, so that how they worked was something of a mystery”.

There is still plenty of hermetically sealed stuff inside OS X, but you can open the hood and work on the thing. At last, it has a command line.

The terminal lives in /Applications/Utilities. Open it up, do an **ls /** and you'll find familiar directories (bin, etc, sbin, tmp, Users, usr, var) alongside others such as Applications, Library, Trash and so on.

Look in /usr/bin and many of the tools you expect to find are there: autoconf, bison, cvs, fetchmail, nslookup, perl, ssh, whoami and so on. The unexpected tools come from the Mac side of OS X's heritage. One example is osascript, which lets you run an AppleScript script from the command line—a feat one imagines the original AppleScript designers never envisioned.

Some usual suspects are missing: no Python or Pine, for instance. But these are itches open-source developers are free to scratch, which is exactly what many are doing. The OSXGNU archive already includes Python, bash (the OS X default is tcsh), Pine and Lynx. So their absence is corrected easily.

The Fink Project distributes ports of ant, MySQL, Nmap, Ruby and so on, with the stated intent to make OS X “a coherent, comfortable distribution that matches what Linux users are used to”.

It's interesting to discover that much of the software included with OS X is BSD- or MIT-licensed, while much of the software one downloads is GPLed. Though

that's just a rule of thumb, it's a definite contrast between OS X and Linux. For example, OS X comes with Curl, available under the MIT license, rather than the more familiar GPL-licensed GNU Wget. This may be attributed to an allergy to GPL-licensed software in Apple's legal department.

Ready to Serve, Sort Of

Though there is a separate OS X Server, OS X itself includes server software such as Apache, sendmail, named and sshd. AppleTalk and Samba support are built-in.

Because OS X is so new, kinks are still being discovered. For example, a security issue was uncovered in Apache, since HFS+, the Mac's native filesystem, is not case-sensitive. This meant a URL that would otherwise be denied would be made readable just by changing the case of the request. Apple responded with a new Apache module that enforces pseudo-case-sensitivity, and which any build instructions for Apache on OS X must take into account.

A Gooey GUI

There are, of course, GUI ways of doing command-line work. Some of them aren't bad. Turning on Apache is easy in the GUI. You press the Start button in Web Sharing utility in System Preferences. You can turn on FTP and SSH access the same way. The command line is still there, of course. Just edit /etc/httpd/httpd.conf with your favorite editor.

This System Preferences panel allows one to turn on and off Apache and FTP access

The GUI is called Aqua, named for its watery, gum-droppy look, which includes variably translucent windows and other stuff Microsoft can copy later. It includes a file browser (still called Finder) that resembles both the Classic Mac OS and the NeXT multicolumn file browser, depending on how you use it. Most GUI applications are stored in /Applications. It's worth noting that one founding UNIX sensibility OS X brings to the desktop is a relatively rigid hierarchical directory schema. Users in the past could scatter apps and documents wherever they pleased. With OS X, applications go in the Applications directory and documents go in the user's own Documents subdirectory. Though you're still free to place them elsewhere (this is UNIX), it's clearly in bad form (also UNIX).

In the OS X Applications directory there's a small pile of free-as-in-beer closed-source applications, mostly from Apple. Some, such as iTunes, iPhoto and iMovie, do what Apple always has done best, which is raise the bar in both GUI art and ease of use. Others, such as Apple's Mail app, calculator and clock, are handy placeholders. There are free-beer third-party apps too, including one that many Linux geeks would rather avoid: Microsoft's Internet Explorer. Fortunately, native OS X versions of Mozilla, Netscape and Opera are a

download away, and all appear to work quite well. OmniWeb 4 is another nice browser that works only on OS X.

The portfolio of available applications for OS X is still small, relative to Windows and even to Mac OS Classic (which OS X runs in emulation). But it's growing fast. As of January 2002 the number was about 5,000, including more than a thousand ported-over UNIX apps. BBEdit (which Mac geeks regard as the greatest text editor ever) is out. More members of the Adobe suite are showing up every month. And now there's an all-new Microsoft Office that some believe leapfrogs the latest Windows version of the suite. Steve Jobs has vowed to make OS X the best environment for Java apps, and we see nothing to defy the claim (while Java was always a pain on the old Mac OS). All these developments are sure to make OS X far more popular (and, we sarcastically suspect, its command-line interface even more necessary).

Little of OS X above its UNIX foundation is open source, although Apple has gone out of its way to invite and incorporate changes and improvements recommended by anybody who's interested—something that Apple never would have done before Steve Jobs returned (with his UNIX-seasoned NeXT team). One report said upward of 70,000 recommendations were sent to Apple during the public beta period alone, and many were incorporated throughout OS X.

Darwin has mostly attracted system-level outfitters. GNU/Darwin appears to have strong interest and developer involvement. XFree86 and XDarwin were developed in the XonX Project. You can run XDarwin rootless, meaning side by side with Aqua, or in its own window. The Fink folks even have GNOME running on OS X.

OroborOSX , a port of the Oroboros window manager, “attempts to make X11 act in a more 'Mac-like' way (whether you like it or not!)”. With Photoshop still not shipping for OS X, the GIMP is another popular download.

On the other hand, the going has been slow in bringing OpenOffice to OS X; Bill Roth of Sun told CNET, “I'm going to make a blatant plug here. We need people on this project.”

Table 1 shows how OS X stacks up against Debian GNU/Linux at the time of this writing.

OS X vs. Debian GNU/Linux

Developer Nirvana?

With every retail copy of OS X Apple ships a Developer Tools CD. Included are the familiar cc and gdb along with Project Builder and Interface Builder: an IDE and user interface layout tool. This is a huge change from earlier development environments. To develop for Macs prior to OS X, one had to purchase MPW (Macintosh Programmers Workshop) tools from Apple or CodeWarrior from Metrowerks.

As of this writing Metrowerks is on the OS X case. So is Borland, with JBuilder. More significantly from a Linux perspective, Trolltech is out with Qt/Mac.

Project Builder and Interface Builder have generated a lot of enthusiasm. Says Graeme Hiebert, “Project Builder is a very nice IDE, and Interface Builder is an amazing piece of work. Prior to Mac OS X, I've been pretty much a makefile guy, but I rarely use them anymore.”

Gabriel Ricard says,

> What I love the most about PB and IB is that I don't have to write a single line of code to create the interface. I just use IB to create the interface, write the classes and their selectors that the interface is attached to in IB and build the app.

For writing GUI apps, one has a choice of writing to the Carbon APIs, which are a revision of the Classic Mac APIs, or using Cocoa, which is an updated version of the NeXT frameworks. Carbon is the best bet for Mac developers porting Classic apps to OS X because it requires the fewest code changes. Cocoa is recommended for new applications. Cocoa applications can be written in Objective-C or in Java. Objective-C is, like C++, a superset of C.

Aside from Carbon and Cocoa apps, one also can create command-line tools, kernel extensions, plugins, pure Java apps and frameworks with Project Builder.

And there is outside participation on the kernel. The list of Darwin Committers is short, but it appears that these people are up to serious work. The Darwin Developers list (to which we subscribe) is quite active and helpful. It makes clear how much work is going on—and how much remains to be done.

It's still early. Almost everything we're talking about here is very new, from Darwin and OS X to the platform-spreading tools and applications developed by the Free Software and Open Source communities.

We'll continue to watch OS X developments closely and invite our readers to do the same and share what they learn. Interoperability isn't something you do alone.

Resources

Doc Searls is senior editor of *Linux Journal*. This feature serves as this month's Linux for Suits column, which will return next month.

Brent Simmons (brent@userland.com) is an experienced Linux, Mac OS and Mac OS X programmer who lives near the *Linux Journal* offices in the beautiful Ballard neighborhood of Seattle.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Build a Virtual CD-ROM Jukebox

Jeremy Impson

Issue #96, April 2002

Use a low-cost, scalable Linux server to share CD-ROM image files to Windows clients.

This article describes how to set up a virtual CD-ROM jukebox (VCDJ) using Samba and Linux. A VCDJ is a network server that provides access to the contents of a large number of CD-ROM disks, without the need for more than one CD-ROM drive. In addition, it simultaneously provides access to the ISO 9660 CD-ROM images in a format suitable for burning copies of the CD-ROMs using a CD-RW drive.

Why Would You Want This?

A CD-ROM jukebox is usually a file server (or file server appliance) connected to a CD-ROM drive tower. It is able to share (often via SMB/Windows Networking) the contents of a number of CD-ROMs to clients on the network. It's valuable because users of the network do not have to locate a particular CD-ROM physically when they wish to install software or access data.

However, this approach suffers from some drawbacks. The number of CD-ROMs it can serve is limited to the number of CD-ROM drives in its tower(s). To add more CD-ROMs, more drives must be obtained and installed. The CD-ROMs must be in the drives at all times, making them unavailable for other purposes. Also, there is no easy way to make copies of the CD-ROMs (especially bootable copies) without removing them from the server, which makes them unavailable for network users.

A VCDJ surmounts all of these limitations. It is different from a regular file server because, while a regular file server might contain the contents of a number of CD-ROMs, the VCDJ contains an ISO 9660 image of the CD-ROMs. When we're done, it will serve both the images and the contents of the images

efficiently (on a file-by-file basis) at the same time. Additionally, the original CD-ROM disks can be stored away where they won't get lost.

Whereas traditional CD-ROM servers are limited by the amount of CD-ROM drives they contain, the VCDJ is limited by the amount of disk space it contains. Hard drives are an order of magnitude cheaper than CD-ROM towers, and they scale better. One 40GB hard drive occupies one spot on an IDE or SCSI controller. It can contain the equivalent of 57 full-sized CD-ROMs (at 700MB each). We would need 57 CD-ROM drives attached to the server to get the equivalent functionality, which is a practical impossibility.

At my place of work, we have found the VCDJ invaluable for publishing the contents of regularly updated software subscriptions. We used to lose track of CD-ROM disks as we loaned them out to others; now we just give them access via their Windows Domain credentials. And, we easily can burn new copies of any bootable CD-ROMs we need. The original disks remain locked away.

The Pieces

To create our VCDJ, we'll need the following pieces:

- One CD-ROM drive in a computer running a recent version of Linux. The drive will be used to create ISO 9660 CD-ROM images. (ISO 9660 is the format of the filesystem usually used on CD-ROM disks. So we refer to a soft copy of a CD-ROM disk as an ISO 9660 image.)
- Enough hard drive space to hold all of the CD-ROM images we want to serve.
- A loopback device, to allow access to the files contained within the ISO 9660 images.
- The automounter, to mount the ISO 9660 CD-ROM images automatically.
- Samba set up to serve network shares.

Make ISO 9660 Images of the CD-ROMs

The first task is to obtain the ISO 9660 images of the CD-ROM disks. Any tool you can use to, say, make duplicates of CD-ROMs can generate proper images. You also can download ISO images of your favorite Linux distribution.

On Linux, the simplest way to make an image is with **cat**. Put the desired CD-ROM disk into the CD-ROM drive. Make sure the directory /mnt/images/ exists. If your CD-ROM disk block device is hdc, the image is created like this:

```
cat /dev/hdc > /mnt/images/image1.iso
```

You'll want to give the image file a more descriptive name. Reading the image may take awhile. Repeat this process for each CD-ROM disk of which you want an image.

Now that we have the CD-ROM images, we'd like to access the contents of the images. The normal method for accessing the contents of the image is to use a loopback device, like this:

```
mount -t iso9660 -o loop,ro /mnt/images/image1.iso
        /mnt/isosrv/image1/
```

This mount command says that we are going to mount some data that uses the ISO 9660 filesystem format. It also says to use the loopback device. The loopback device is a nifty kernel feature that allows you to designate a file, in this case /mnt/images/image1.iso, to be used as if it were a character device, like a hard disk or CD-ROM drive. This command mounts the image file in a read-only format. The contents of the CD-ROM image can be seen in /mnt/isosrv/image1/.

How Do We Know That the Image Is Correct?

Configure the Automounter

If our VCDJ has a lot of ISO 9660 images, it won't be possible to mount all of them statically. The next step is to configure the automounter. The automounter will mount an ISO 9660 image only when it is accessed. It will unmount it after a time of inactivity. We need this because there is a limit to how many filesystems can be mounted via the loopback device at one time. It's unlikely that all of the CD-ROMs will be in use simultaneously, so it's the automounter to our rescue. (See “Tux Knows It's Nice to Share, Part 4” by Marcel Gagné at /article/5298 for instructions to install and initially set up the automounter.)

First, edit /etc/auto.master, and append to it the following line:

```
/mnt/isosrv_auto   /etc/auto.isosrv    --timeout=60
```

Make sure the directory /mnt/isosrv_auto exists. Restart the automounter for this change to take effect.

Create the file /etc/auto.isosrv, and append to it the following line:

```
image1  -fstype=iso9660,ro,loop :/mnt/images/image1.iso
```

Create a similar line for every ISO 9660 CD-ROM image that is to be automounted.

If you mount on your VCDJ, you should see a line like this:

```
automount(pid782) on /mnt/isosrv_auto type autofs
(rw,fd=5,pgrp=782,minproto=2,maxproto=3)
```

(The various numeric values will likely be different on your system.)

The automounter does *not* have to be restarted when changes are made to /etc/auto.isosrv. So far we've told the automounter that when some process tries to access a file or directory somewhere in /mnt/isosrv_auto/image1/, it will mount image1.iso. After a period of time of no access to the directory, the image will be unmounted.

Laying Out the Filesystem

There's one last problem. List the contents of /mnt/isosrv_auto/

```
ls /mnt/isosrv_auto/
```

If nothing has accessed the contents of the CD-ROM image recently, this directory will appear to be empty.

If you explicitly list the contents of the CD-ROM,

```
ls /mnt/isosrv_auto/image1/
```

you will see the contents. Now go back and list the contents of /mnt/isosrv_auto/ again, and you will see image1. Eventually, the automounter will unmount the image, and once again the directory will be empty.

This is a problem because it means the users will have to know the names of all the CD-ROMs that they want to access. Directory browsing won't work, which is clearly not acceptable.

The solution is to create another directory called /mnt/isosrv/. Enter that directory and perform the following commands:

```
mkdir image1
cd image1
ln -s ../../isosrv_auto/image1 disc
```

Repeat this for every ISO 9660 CD-ROM image.

Listing the contents of /mnt/isosrv/ will show all the available images, regardless of whether the automounter has mounted them.

Further Explanation of the Filesystem Layout

Configure Samba

See “Tux Knows It's Nice to Share, Part 5” by Marcel Gagné (/article/5297) for instructions on installation and initial setup of Samba. Be sure that you have authentication set up to protect the CD-ROM contents properly, according to their licensing agreements. (Not a problem if everything is open source!)

Recall that one goal is to provide access to the ISO 9660 images and their contents simultaneously via Samba. To do this, edit /etc/smb.conf (or maybe /etc/samba/smb.conf), and append the following lines:

```
[isoimages]
        comment = ISO9660 CD ROM images
        path = /mnt/images/
[cdroms]
        comment = Contents of CD ROMs
        path = /mnt/isosrv/
```

Restart Samba. Now go to a Windows client (or any other computer that can be an SMB client) and browse your VCDJ. You should see two new shares: isoimages and cdroms. In the isoimages share are all the ISO 9660 images, and in the cdroms share are the contents of the images. Browse the contents of the directory image1 in the cdroms share. If you **mount** on your VCDJ, you should see a line like this:

```
/mnt/images/image1.iso on /mnt/isosrv_auto/image1
        type iso9660 (ro,loop=/dev/loop0)
```

Conclusion

Figure 1 shows a diagram of the order of things when a network client accesses data in the cdroms share. Note that if the client accessed an ISO 9660 image directly, the Samba process would directly read from /mnt/images/, bypassing all symlinks and the automounter.

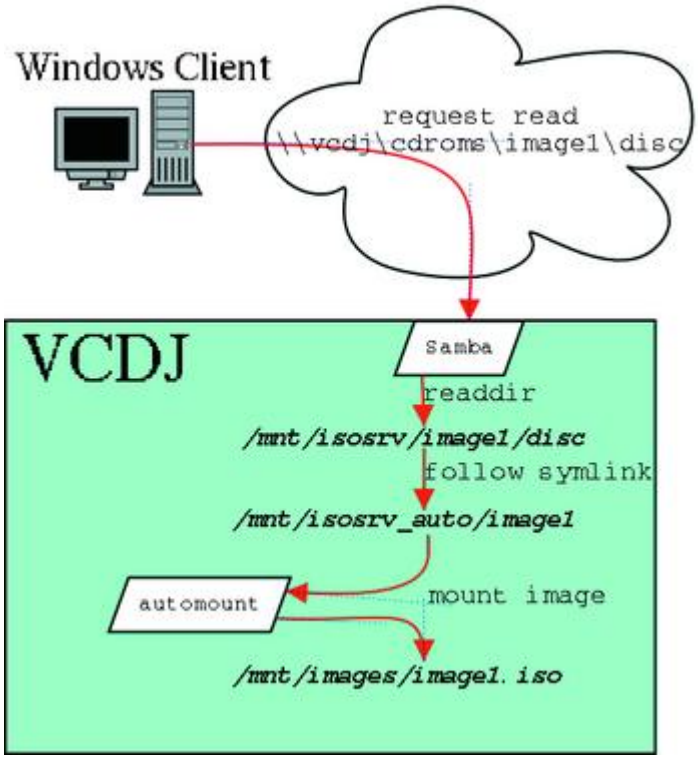


Figure 1. Network Client Accesses Data in the cdroms Share

As you add more ISO 9660 images, you'll come to appreciate the value of a VCDJ. You're only limited by available disk space, and the cost and degree of effort to add more capacity is much less than trying to add more physical CD-ROM drives. Now you can make all your CD-ROMs available across your enterprise, whether in your office or in your home, and you'll never have to worry about people not returning borrowed CD-ROM disks.

email: jdimpson@acm.org

Jeremy Impson (jeremy.impson@lmco.com) is a senior associate research scientist at Lockheed Martin Systems Integration in Owego, New York. There he's a member of The Center for Mobile Communications and Nomadic Computing, where he uses open-source software to develop mobile computing systems.

Archive Index Issue Table of Contents

Advanced search

Connect to Microsoft SQL 2000 with the Perl Sybase Module

Andrew Trice

Issue #96, April 2002

Andrew shows how to build the DBD::Sybase module with the TDS libraries.

For many system administrators, myself included, Perl is the scripting language of choice. Perl is nearly ubiquitous, available for most operating systems in popular use today. Thanks to the free availability of its source code, hundreds of modules extending its capabilities have been developed. Quite noteworthy among Perl modules is Tim Bunce's DBI (database independent). The Perl DBI provides an API for interfacing with any database, at least any database for which a corresponding DBD (yes, database dependent) module is available.

A quick scan of Perl modules listed on cpan.org, Perl HQ, shows DBD modules for Informix, Oracle, Sybase, IBM's DB2 and many more. Glaringly absent from this list is a DBD module for connecting to a Microsoft SQL server. A DBD::ODBC module has been developed, but using this requires a separate driver manager and suitable drivers. To the best of my knowledge, these are only available for Microsoft's SQL 2000 from third-party vendors, a solution that of course brings the benefits of professional development and support. As it turns out, there is another way.

Thanks to an arrangement with Sybase, Microsoft's SQL server products were developed with the same network communications protocol that Sybase used, namely Tabular Data Stream, or TDS. Through the release of SQL 7.0, Microsoft officially supported Sybase client software with a caveat that such support was at its end. Thus the Perl DBD::Sybase module, compiled with Sybase's freely downloadable client libraries, was able to connect to any Microsoft SQL server until the release of SQL 2000.

With the introduction of TDS 8.0 and legacy support for TDS 7.0 in SQL 2000, compatibility with the Sybase client, using TDS 4.2, is broken. However, the

DBD::Sybase module can be built with the TDS libraries from freetds.org, which does support TDS version 7.0, and is used to connect to SQL 2000. Here's how.

I have tested this on Red Hat Linux 7.1 and 7.2. Because this project requires working with source code, you'll need a compiler. GCC, the GNU C compiler, is very conveniently supplied with Red Hat's Linux distribution. Perl and the DBI module are likewise available in RPM form on Red Hat CDs.

First, download DBD-Sybase-0.94.tar.gz to a convenient location on your machine. This is available at cpan.org and also from the author's download page at www.mbay.net/~mpeppler. The FreeTDS source code is available on www.freetds.org. Get freetds-0.53.tgz, the latest revision as of this writing. I strongly recommend reading all the helpful documentation available on these respective sites.

Next, gunzip and untar these downloaded files, then **cd** to the newly created freetds-0.53 directory and run **./configure --with-tdsver=7.0**, which writes a Makefile suitable for compiling FreeTDS on your machine and specifies tds 7.0 as the default protocol.

Now run **make**:

```
make
Making all in include
make[1]: Entering directory
         `/home/atrice/freetds-0.53/include'
Making tds_configs.h
```

and so forth.

Then run **make install** to install your freshly compiled FreeTDS. By default, it installs to /usr/local/freetds, though you can change this when running configure with the -prefix=(PATH) switch. You must be root to install:

```
make install
Making install in include
make[1]: Entering directory
         `/home/atrice/freetds-0.53/include'
make[2]: Entering directory
         `/home/atrice/freetds-0.53/include'
```

and so on. The final comments from **make install** will be something like:

```
if [ -f /usr/local/freetds/etc/freetds.conf ]; \
then :; \
else \
/usr/bin/install -c -m 644 freetds.conf
/usr/local/freetds/etc/freetds.conf;
\
fi
make[2]: Leaving directory `/home/atrice/freetds-0.53'
make[1]: Leaving directory `/home/atrice/freetds-0.53'
```

You have now compiled and installed FreeTDS.

Next, we perform a very similar set of commands to configure and build the DBD:Sybase module. First, however (very important), we must set up an environment variable called SYBASE. This variable will be set to the path of the FreeTDS installation. I am using the bash shell:

```
export SYBASE=/usr/local/freetds
```

Confirm the proper setting of the variable:

```
echo $SYBASE
/usr/local/freetds
```

Then **cd** into the DBD-Sybase directory and run Makefile.PL:

```
perl Makefile.PL
Sybase OpenClient found.
The DBD::Sybase module needs access to a Sybase server
  to run the tests.
To clear an entry please enter 'undef'
Sybase server to use (default: undef):
User ID to log in to Sybase (default: sa):
Password (default: undef):
Note (probably harmless): No library found for -lcs
Note (probably harmless): No library found for -lsybtcl
Note (probably harmless): No library found for -lcomn
Note (probably harmless): No library found for -lintl
Using DBI 1.20 installed in
  /usr/lib/perl5/site_perl/5.6.0/i386-linux/auto/DBI
Writing Makefile for DBD::Sybase
```

Notice that the Makefile.PL script asks you for information about your Sybase server. It uses this information to write a file called PWD that is used by the test utilities provided with the module. These were designed to run against a Sybase server, not Microsoft, and I did not have much success with the tests. The messages regarding the libraries not found are due to compiling DBD::Sybase with the FreeTDS libraries instead of Sybase's.

Run **make**, then **make install**. The last few messages from make install should read as follows:

```
Installing /usr/share/man/man3/DBD::Sybase.3
Writing /usr/lib/perl5/site_perl/5.6.0/i386-linux/
auto/DBD/Sybase/.packlist
Appending installation info to /usr/lib/perl5/5.6.0/
i386-linux/perllocal.pod
```

You are done installing DBD::Sybase. Next we'll configure FreeTDS to talk to your SQL 2000 database.

FreeTDS has a configuration file quite logically called freetds.conf. It resides in the freetds installation directory under /etc, so in my case the full path is /usr/local/freetds/etc/freetds.conf. There is a sample Microsoft server configuration

already present in this file, and you only need modify it to reflect your server's information. Mine looks like this:

```
# A typical Microsoft SQL Server 7.0 configuration
[file1]
        host = file1
        port = 1433
        tds version = 7.0
```

My SQL server is called file1; it runs its database service on the default port of 1433, and I have specified the tds version 7.0. There is a global configuration section in the beginning of this file where you also can set the tds version. Make sure your client machine can resolve the hostname of your database server, or simply use its IP address.

Now we're ready to attempt a connection. Listing 1 is a Perl script written by my colleague Trevor Price that queries the sample database called Northwind. This database is included with a typical SQL 2000 installation. It makes use of a stored procedure called sp_help, which returns information about all the tables in that database. Copy this code into a file called something like testsql.pl, then edit $user and $password to reflect a database account with access to your server.

Listing 1. Trevor Price's Perl Script that Queries the Sample Database Called Northwind

Run the script and you should get an output looking like this:

```
perl testsql.pl
rows is -1
Alphabetical list of products   dbo     view
Category Sales for 1997 dbo     view
Current Product List    dbo     view
Customer and Suppliers by City  dbo     view
Invoices        dbo     view
Order Details Extended  dbo     view
```

If you've gotten this far, congratulations! I hope you find this as helpful as we have.

email: atrice@vitallink.com

Andrew Trice is a systems administrator with Vital Link Business Systems. He holds a BA in English Literature from Cornell University and is chief mastering engineer with Iron Robot Records, an independent label based in San Francisco.

An Interview with Andreas Leimer, CTO of Inalambrica.net

Phil Hughes

Issue #96, April 2002

Phil and Andreas discuss Inalambrica.net's solution for bringing inexpensive and reliable internet connectivity to Costa Rica.

Staff of Inalambrica.net.

Back row from left to right: Geraldy Castillo, Esteban Barrientos (Pato), Alfredo Delgado (Alf), Andreas Leimer (Andy), Alberto Brealey (Beto) and Paulo Campos. Front row: Randall Aguilar, Vera Sanchez, Alina Castro, Agustin Guevara and Federico Figueroa (Fede).

I recently interviewed Andreas Leimer, Chief Technical Officer of Inalambrica.net. Inalambrica, based in Costa Rica, is using Linux to offer inexpensive, reliable internet connectivity around the world. Here is their story.

Phil Start from the beginning. What got you interested in Costa Rica?

Andreas My wife is Costa Rican. She got pregnant, and she decided to come home to have the baby.

I came to use the Internet here, and it just didn't work. Dial-up was very bad and communication was very bad. So I met some Costa Ricans and told them we need to start a company to bring better communications, better solutions, here to Costa Rica. At the time of course, the OS of choice was NT.

Phil What year?

Andreas Five years ago—1997. We tried many different solutions. We tried compression software that worked on NT proxies, and that just didn't work. It sort of worked in the office, but when we went to demonstrate it, it just didn't work. And then we had to diagnose it for a couple of weeks to figure out how it worked or why it didn't work. At the time, all this was based on dial-up connectivity.

Then we learned about direct tier-1 type satellite connectivity with DVB. We were using NT, so it was expensive. Easy to configure but hard to diagnose—it just didn't work.

This led us to a Harmonic Data Enterprise BR-501.

It was a very badly configured Linux system, which almost kept us from switching to Linux. The drivers were especially bad. They had problems with their configuration. They had no simple way to use interfaces. It was a very complicated thing. That happened about two and a half years ago, about a year and half after we failed completely with any Windows applications that we had tried to use.

Phil So that was really your introduction to Linux. It was a pain to configure, but at least it had the potential of working.

Andreas Yes. We installed a couple of these systems, and we started testing. I learned of a technician at a local university that knew quite a bit about Linux. He said he wanted a job, and I hired him. That's how this mess started, and that was approximately two and a half to three years ago. We removed the drivers out of that box and built our own Linux box and tested it. We told Harmonic Data in San Diego what we'd done, and they told us we weren't allowed to use the drivers. We said okay, we'll respect that, we won't use them (they're not open-source drivers). Right around that time, Telemann came out with a card called the SkyMedia 200D, and they had open-source drivers. They actually gave us the source code.

Phil This is the card to connect to the dish?

Andreas Right, this is the card that plugs into the PC running a Linux distribution, with a satellite dish. We also use the PC as a proxy caching engine, and then we built some caching engines to give us multicaching over the one-way satellite. The rest is history. We started installing systems. We had customers with different requests to install the system. The customers said we can't use Linux; we can't do anything with this; we don't know how. So that's what gave us the idea to start with PHP.

We hired some PHP programmers and started building an interface, to change gateways, for example, and that was the beginning of the interface of INATS. Each customer had a different need—I'll buy this if you do this or if you'll do that. And that's how INATS was created and developed, through my experience with going to the customers.

That was two and a half years ago. This last year we've made our greatest developments. We've had a little more money to spend, and we've gotten to a stripped-down distribution. Down to a very small size, we've created our own base. It was based on Slackware, believe it or not. Our first base was a stripped-down Slackware distribution. [See the article “A Conversation with Alfredo Delgado of Inalambrica.net” in the January/February 2002 issue of our sister publication, *Embedded Linux Journal*.]

Phil Then you added your own packaging and administration system?

Andreas Yes, our own packaging and administration system called INATS (Internet Networking Administration Tool Software). INATS controls each of the modules. That's it in a nutshell.

Phil When did you start the development of this intelligent interface?

Andreas About three months after our first satellite installation. That was at the Ulatina University here and Earth University in Guapiles. By the way, in Earth it rains 320 days a year. We had some problems with the 76cm dish. That's when we learned about the 1.2m dishes and Squid; we really liked the way Squid worked. If we couldn't get the packets over the satellite during a fade, then Squid would revert back to its internet channel to pull the packets. So we really liked Squid, just for that simple feature.

Phil So all of this is one-way satellite?

Andreas One-way, everything simplexed—to bring broadband down. It's only for downloading. We've found other products like cellular CDPD, where we'd connect it through a serial port, and it would be able to reach speeds of 512Kb on the satellite down. CDPD is a 12Kb connection, which is very slow and bad to

use for internet surfing. But we found with the dish and the caching engines at multilevel, it was very fast. Also, we could do some video streaming on the satellites with a handshake on the cellular. Of course it's UDP multicast, but it works very well—high quality.

Phil Well, one other thing that you've got is compression. Is that inspired by CDPD?

Andreas No, compression was inspired by the fact that I still have phone dial-up at my house, and I don't have a wireless, high-bandwidth connection. I wanted to be able to get information from the Web faster, so we built a compression engine, using Squid, of course, to do the caching part of it, and then compressing the images and transmitting through phone lines at a high speed. I do have a 28.8K modem at the house, and with compression, it outperforms my neighbor's 56K modem.

Phil I know few people reading this magazine are going to understand how Costa Rica works as far as communications.

Andreas It's a total monopoly. I think it's one of the only telecommunication monopolies in the world—owned by the government. Costa Rica is completely controlled by the ICE and RACSA. RACSA is the primary internet provider. They consider themselves an internet access provider; there are no ISPs in Costa Rica. ICE is also an internet access provider. They are the same company with two different names. ICE owns the infrastructure, and RACSA sells on top of the infrastructure.

Phil And what they are selling mostly is dial-up connectivity?

Andreas Dial-up and dedicated lines. Currently in Costa Rica there are about 900 dedicated lines. ICE is working on a DSL project that could become very popular here. [This is covered in Phil's interview with Guy de Téramond in the January 2002 issue of *Linux Journal*.] The cable company here does have some cable modems installed, but they have many problems. One of the many problems in Costa Rica is mold, of course, and earthquakes. We had 3,000 earthquakes last year, 70 of which we felt. Power lines go down, cables get broken, fiber gets broken. They are talking about constructing some fiber around the central valley to connect all of the DSLs to all of the homes. We've done some low-earth orbit satellite technology here, but it really doesn't work. Iridium went out of business; Orbcomm is another company that has LEOs. We can use that frequency, but the satellites are at a lower altitude, have very low bandwidth, and the satellites hide behind the mountains. We get some activity for two hours, and in the end it just doesn't work.

Phil That's really the problem with LEOs—they aren't accessible.

Andreas So, then we went to the fixed satellite two and a half years ago, and it works absolutely perfectly. One of the things that we engineered and take pride in is having first-level backbone connectivity, putting a Squid caching engine at the teleport. Then, parenting with a satellite receive cache engine on a multilevel over a simplex system—we're actually pioneers of that. We've actually written and filed a patent, and it looks like it's going to be accepted. It's already been filed for two years, and they haven't found anybody with the same type of patent.

Phil It's essentially multilevel caching over...?

Andreas A simplex satellite. We don't own any patent on the caching; we own the patent on the idea of the simplex satellite multilevel caching, and we are using Squid.

Phil With all of these systems, with the broadband download, what's the other way?

Andreas Just a regular connection. It could be a dedicated line, a DSL, cable, cell phone or dial-up. Whatever it takes to make a connection to the backbone. When I say the backbone, I'm talking about the Costa Rican backbone that connects us to the Internet. It all travels through the local monopoly, all the packets on the outbound. The inbound over the satellite basically is to speed up your HTTP, HTTPS and FTP over HTTP. Basically that protocol is all that we handle over the satellite. Caching web content on level one in the US.

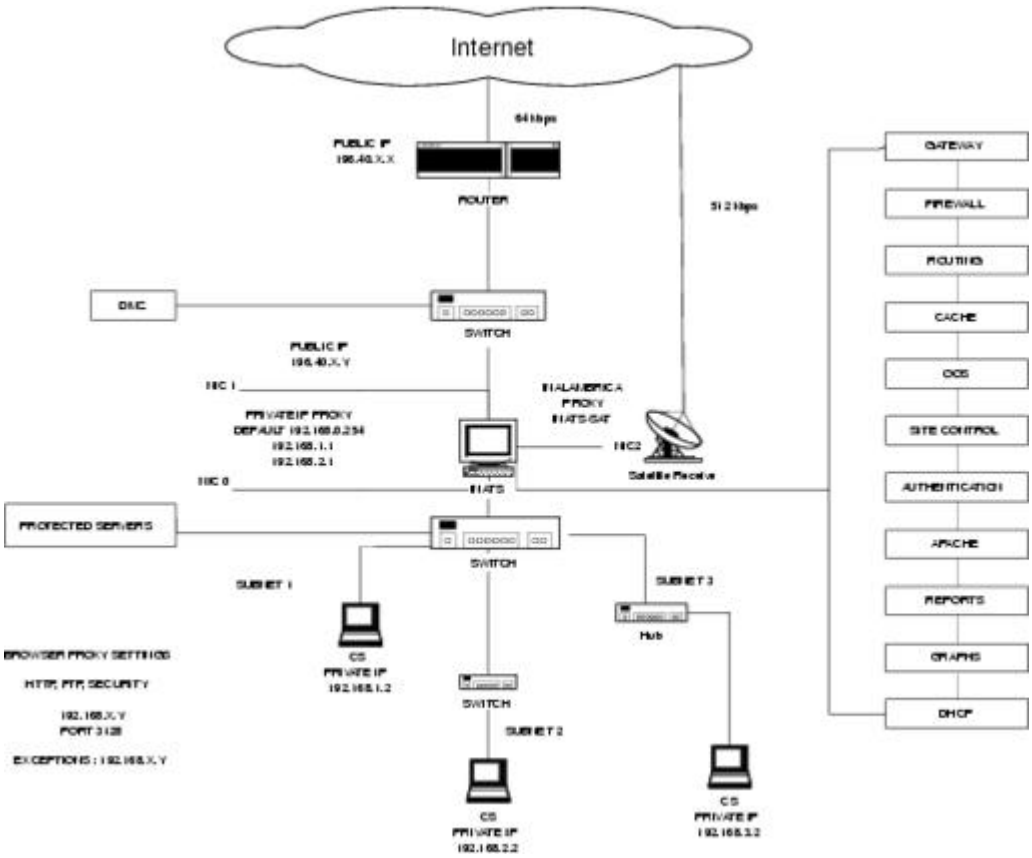


Figure 1. INATS Diagram

Phil How does INATS fit in to all this?

Andreas The idea in principle and the path of the packets are simple to understand in the diagram form [see Figure 1].

When it came to the server, nobody seemed to understand a Linux server, or very few people understood. This made our market very small. So, we determined it was time to start building a Linux-in-Windows or Windows-in-Linux-type machine—button pressing. That's what people like to do, press buttons, add IPs and basically not have to diagnose a server requiring stability and ease of use, like an appliance. That's how INATS started.

We started originally with the gateway program. All that we had was a customer that had two different gateways, two different internet connections; if one were to fail they would switch to the other one. Their main criterion was to have connectivity 24 hours.

So from the network managers' level, they didn't have control of the router or didn't know how to program it. They needed something simple so they wouldn't have to change 150 computers from one network to the other, which takes hours to do. We built the simple PHP interface and programmed everything. It was just a button that would switch from one network to another. Obviously, we had set all of the parameters inside of the Linux machine, and

through their web browser on their Windows machine—they just press a button.

From that we developed using it as a proxy, adding routing, subnets and control reports. The control aspect of it was nice because they were tired of the sales department going to the wrong sites. You know, XXX sites, whatever. And they wanted to account for it, saying “hey, stop doing this.” If they wouldn't stop, then they actually would deny those sites. So that was one of the programs that we wrote, what we called our control program, I guess. It has three different levels: unlimited use, denied sites and allowed sites.

“Denied” means that we deny these sites on the subnet or the whole net, or we only allow certain sites. “Allowing” is meant for schools, and we have some high schools that use the program. They wanted to give Internet to the classroom, but they were worried about students going to the wrong sites; you know how you can type in something random and a sex site shows up; you don't have to do it on purpose, it just happens.

Phil Whitehouse.com vs. Whitehouse.gov is a perfect example.

Andreas Exactly. So what the teacher can do is program into a database the sites that children are allowed to go to. They can't click through; there are no ways to go to any other pages—only those pages. Two things happen when this occurs. One is the pages cache, and then if you have 40 people in a classroom, they have a cached web page at 1.6 megabits per second. It's wonderful. Saves bandwidth for us and we can reduce the price for them for their connectivity to the Internet.

Phil What are your goals, who are potential customers and where are you going?

Andreas Costa Rica has a very small market. But per capita, it's a very technologically advanced market. We currently are working with large companies here, and we are doing some pilot projects with the PTT.

Worldwide, or at least Latin America-wide, we've signed a contract with a large company called Bismark International, which has subsidiaries in Latin American countries including Brazil, Peru, Chile, Ecuador, Colombia, Venezuela, Uruguay, Panama and Mexico.

Currently, in Mexico we have a test system we are doing with Telcel there. And, they signed a contract with us for us to provide them with the software platform that will integrate hardware items off the shelf. And that's basically what INATS does using Linux as its OS. With different open-source products,

plus some of our own that we created, we're operating different network appliance-type servers. The contract that they've asked for is approximately 30,000 different installations over a three-year period, and it could lead us to bigger and better things.

Phil Do you see Latin America as the primary market?

Andreas We'd like to take it into other markets and are looking for distributors in other markets. Currently, yes; in the future, no.

Phil This is sort of a Costa Rican question: Why is this the right answer, as opposed to, for example, decent fiber connectivity, which they're beginning to get?

Andreas Well correct, but even the fiber seems to saturate during the day, and the speed of the response directly off the one hop from the US backbone seems to compete very well with the fiber. It's amazing actually. We've been running tests these last few weeks. The fiber, when it's empty, is very fast. But when it's being used, it tends to slow down. The speed of the satellite, especially in remote areas where there is no fiber, is almost the same speed as the fiber. With the one-way dish, we're averaging approximately 370 milliseconds per round-trip. With clean fiber we're averaging about 140-180 milliseconds per round-trip. When fiber is saturated during the day here in Costa Rica (the country, by the way, has 90MB coming in) our fiber connection goes up to 500-600 milliseconds. And as we are like a highspeed motorcycle out to the US backbone, we are like a tractor-trailer down over the satellite. Our response times are very good over the satellite. And, it is used for areas where you can't get large fiber interconnectivity. There are areas in Costa Rica that have dedicated lines, and the copper can't go above a 384 or a 128 or even a 64Kb connection in some areas. We can compliment that dedicated line with a dish to receive more bandwidth in areas that can't have a larger than 64Kb connection.

Phil As far as rural, I know in the US, for example, that with DSL connections at least, Qwest specs them so that if you're more than three miles from the central office, you're screwed. And it seems that in this country there are 240 telephone offices in the whole country.

Andreas And everybody lives further than three kilometers from that office. So those areas have limitations. The 802.11 spectrum is very saturated. Even on the 802.11 that isn't saturated, we've complimented it with a dish to the point where we can use the 802.11 as our backbone connection and use the dish for the downloading directly off the US. When it's local connectivity, we don't use the dish but we do see about 70% of the traffic coming off the US backbone.

And that is for sites and information from companies that you want to be able to get quickly. So for a business, this is a great item to have. Of course the caching makes it a lot faster. The caching helps; it makes the speed incredible. Without the multilevel caching, we wouldn't have this kind of speed.

Phil I assume your customer base is primarily business?

Andreas Yes, small and large businesses. We are working on doing some home-user-type applications where a condominium actually would share one of these satellite dishes and wire it into each apartment. And, with the INATS software we can give good quality service to each apartment.

This brings me back to the OS that we're installing into INATS so we can give certain service-level agreements to each customer. If they're sharing a 512Kb when there is a 20 to 1 ratio, the minimum would be a 25Kb SLA. And we can guarantee that, using a 2.4 Linux kernel with an OS built into it.

Phil As opposed to inventing all this for...?

Andreas Exactly. And making it cost-effective for the customers. Total cost of ownership is probably the lowest in the world.

Phil That brings up one thing. Other than reliability, you mentioned quality of service. What else in Linux has made this an easier task than trying to do it with brand M or something?

Andreas MRTG, multirouter traffic grapher, is an excellent program giving us knowledge of how much bandwidth is being used and how much is needed. Or, if we can reduce connections in different businesses. Some businesses over-purchase. We actually can help save money by having that MRTG and using it in our administration tool. It's wonderful. It reads the traffic that's going through the NIC cards; or if it's a router card that's installed, it reads packets going through the router cards. It gives administrators management of their network; if they are saturated and need more bandwidth they are going to know.

The Squid Analysis Report Generator (SARG) is another great program written by Orso [orso@onda.com.br]. SARG is a tool that allows you to view where your users are going to on the Internet. It generates reports in HTML using the access_log file, with fields such as: users, IPAddresses, bytes, sites and times. It's very nice tool for control or accountability.

Using DHCP, of course, is a nice feature, as is using the routing, the proxying video streaming. The Darwin streaming server is wonderful with the IceCast radio streaming; you can plug in a radio or tape for viewing or learning and the Darwin streaming actually takes videos, compressing them into a digital format

so that they can multicast onto the network and people can use that for viewing or learning.

Phil So essentially...?

Andreas All open-source products.

Phil What you're doing is taking a whole bunch of open-source stuff and putting it together, and your added value is that you made it so Joe Moron can come in and take all of this open-source software and turn it into something that will solve this problem.

Andreas You don't have to learn a new operating system, but you get the benefit of using one.

Phil Okay, what did I miss?

Andreas I think that's pretty good. This same type of service is available using Cisco equipment—that is, routing, a caching engine and QoS. To do everything that we're doing would cost about $130,000 in Cisco hardware.

Phil And what does this cost from you?

Andreas Small network, 20 people, 20 bucks a month (if they furnish the hardware). If not, we can finance that (Costa Rica only at this time). It's off-the-shelf items. Try to buy Cisco products in different countries; you have to order it because there are limited supplies. Our idea is to use off-the-shelf items; go to the store, get a PIII or motherboard or an IDE hard drive and plug it in. And on Flash disks too. We are finishing that.

email: info@linuxjournal.com

Phil Hughes is the publisher of *Linux Journal* and *Embedded Linux Journal*.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Linux IPv6: Which One to Deploy?

Ibrahim Haddad

Issue #96, April 2002

If you are hesitant as to which IPv6 implementation to adopt for your Linux server, this article will help you decide.

IPv6, short for Internet Protocol Version 6, is the next-generation protocol designed by the IETF to replace the current version of Internet Protocol, Version 4 (IPv4). Most of today's Internet uses IPv4, which is now nearly 20 years old. IPv4 has been remarkably resilient in spite of its age, but it is beginning to have problems. Most importantly, there is a growing shortage of IPv4 addresses, which are needed by all new machines and devices connecting to the Internet.

IPv6 comes along to fix a number of problems in IPv4 and to add many improvements to cater to the future Internet. The improvements come in areas such as routing and network autoconfiguration, security and mobility. IPv6 represents a big package of capabilities, of which addressing is the most visible component. The addressing issue gets a lot of attention, but it is only one of many important issues that IPv6 designers have tackled. Other capabilities also have been developed in direct response to critical business requirements for scalable network architectures, improved security and data integrity, integrated quality of service, automatic configuration, mobile computing, data multicasting and more efficient network route aggregation at the global backbone level.

This article exposes part of the IPv6 work conducted within the ARIES Project (Advanced Research on Internet E-Servers) in the Open Systems Lab at Ericsson Research in Montréal, Canada. The ARIES Project started in January 2000 and aimed at finding and prototyping the necessary technology to prove the feasibility of a clustered internet server that demonstrates telecom-grade characteristics using Linux and open-source software as the base technology.

IPv6 is an essential technology to be supported on such telecom-grade systems and clusters. Ericsson's vision is that all mobile users in the near future will be “always connected, always on-line”. This new service paradigm, together with

the use of IP technology, opens up tremendous opportunities for service providers and network operators to create new and diverse services. However, the rapid increase in the number of internet users combined with the expected growth in the number of wireless internet devices requires a scalable and flexible IP technology to accommodate such fast growth. Therefore, it is essential to recognize that IPv6 is a key technology to realize the vision of a large number of users being always connected, always on-line. New services, such as IP multimedia, assume globally unique addressing for reachability. IPv6, with its very large address space, will guarantee a globally unique IP address for each device.

This article explores the different open-source projects working on IPv6. The goal is to experiment with the Linux IPv6 implementations currently available and present recommendations to favor an implementation for our Linux processors on our near telecom-grade Linux clusters. The recommendations must be based on several criteria such as the implementation development speed, its compliance to the standards and its performance vs. other implementations.

Open-Source IPv6 Projects

The first step was to survey the Open Source community and report on the IPv6 projects that aim at providing IPv6 implementations, enhancing an existing implementation or providing testing and verification for other implementations.

The WIDE IPv6 Working Group (IPv6 WG), part of the WIDE Project, was started in 1995 in Japan for the purpose of experimenting with and deploying IPv6. In late 1995, IPv6 WG had several independent implementations and held interoperability test events. As the specification was verified and interoperability became common, it appeared that it was ineffective for IPv6 WG to implement IPv6 stacks independently. For this reason, the WIDE Project started the KAME Project as a subproject for combining the power of implementations.

Although the members of IPv6 WG and KAME overlap, IPv6 WG mainly does technical and innovative research, while KAME is in charge of implementation.

The WIDE IPv6 WG goals can be summarized as providing IPv6 implementation and software, encouraging the deployment of IPv6 in production environments and developing a transition mechanism from IPv4 to IPv6. In addition, the project aims at establishing the technologies and expertise for the administration of IPv6 networks.

The expected output of the project is a free IPv6/IPSec code conforming to the RFCs.

The KAME Project is a joint effort of seven companies in Japan to create a free, solid software stack for BSD variants (FreeBSD, OpenBSD, NetBSD and BSD/OS), targeted especially at IPv6 and IPSec. The project was formed to avoid unnecessary duplicated development and to deliver a high-quality and advanced feature-full implementation.

The project started in April 1998 and is planned to run until March 2002. The core researchers are from the following companies: Fujitsu Limited, Hitachi, Ltd., IIJ Research Laboratory, NEC Corporation, Toshiba Corporation and Yokogawa Electric Corporation. They have committed to work on the IPv6 stack on a full-time basis, making the project one of their primary tasks. Their goal is to implement the best networking code possible under BSD copyright and present their results as free software.

The expected project outcome is an excellent quality code (provided as free software) for IPv6 and IPSec (originally based on WIDE Hydrangea IPv6/IPSec stack), which will be the basis for advanced internetworking in the 21st century.

The TAHI Project started in October 1998 in Japan as a project between the University of Tokyo, YDC Corp. and Yokogawa Electric Co, with the objective of developing and providing the verification technology for IPv6 through research and development of conformance and interoperability tests. The group works in collaboration with the KAME Project on the quality side by offering the verification technology developed in the TAHI Project and improving the development efficiency. It's also important to note that the TAHI Project also receives great support from the WIDE Project.

The TAHI Project resulted in creating:

- Conformance tests: the project releases its conformance test suites bimonthly. The release timing may be same as the KAME stable release.
- Interoperability tests with simple network: these tests verify if your target node can work in some certain typical networks. Since this test will be conducted in limited environments, this is set as the first step of the interoperability tests.
- Interoperability tests with a multi-implementation environment: these tests verify interoperability with the real world.
- Test scenarios and test tools.

The results of the project are open to the public for free.

The USAGI (UniverSAl playGround for IPv6) Project works to deliver the production quality IPv6 protocol stack for Linux in collaborating with the WIDE Project, KAME Project, TAHI Project and the Linux IPv6 User's group. We will discuss the USAGI Project and its IPv6 implementation in a later section.

The IPv6-DRET Project is a public Linux implementation of IPv6, funded by the DGA/DRET (French Military Research Agency) and codeveloped by INRIA Sophia-Antipolis and LIP6 Paris. IPv6-DRET was based on the Linux kernel 2.1, and the goal of this implementation was to test certain algorithms relative to quality of service (QoS).

The objective was not to provide a full-fledged implementation, rather to build enough functionality to test certain algorithms relative to QoS. Since routing issues are among the most important research interests concerning IPv6, their network stack is intended to accommodate both host and router IPv6 machines. Besides implementing the router specification, a great deal of effort also was put into the development of RIPng into GateD.

The project is no longer in existence. The development has stopped and the code is outdated.

The Linux IPv6 RPM Project prepares RPM packages for the RPM-based Linux distributions (i.e., Red Hat Linux, Turbolinux and Caldera OpenLinux) that consist of the software and tools needed to connect to IPv6 networks. The aim of the project is to provide RPMs that make it easy to get IPv6 connectivity.

The Debian IPv6 Project is part of the Debian Project. It aims at converting some of the Debian packages to be IPv6-compliant.

As you may have noticed, among all the projects listed above, only two provided serious IPv6 implementations targeted for Linux: the Linux kernel implementation and the USAGI Project. In the following sections, we look in detail at these two projects and implementations and see which one complies more with the IPv6 specifications.

The USAGI Project aims at delivering a production-quality IPv6 stack for Linux and the experimental operation of an IPv6-compliant network using the developed software in tight collaboration with the KAME, WIDE and TAHI Projects.

The group is comprised of several organizations from private and academic sectors to promote the implementation and deployment of IPv6 for Linux. The group members are the WIDE Project, CRL, GLUON PARTNERS Co., Ltd., INTEC,

Inc., Toshiba Corporation, Hitachi, Ltd., NTT Software Corporation, Yokogawa Electric Corporation, the University of Tokyo and Keio University.

The project started in October 2000 and is scheduled to end on March 31, 2002. The existing Linux kernel has its own IPv6 protocol stack; however, based on the evaluation results done by the TAHI Project, the quality of the current implemented IPv6 stack is not as good as the IPv6 stacks provided in other operating systems such as FreeBSD and Microsoft Windows 2000.

The project collaborates with other research and development projects and organizations regarding the IPv6 system development and deployment. The USAGI Project collaborates with the KAME Project (an IPv6 for the BSD UNIX system) and with the TAHI Project (an IPv6 test and evaluation specification and tools).

The USAGI implementation relies heavily on the KAME code, which is the IPv6 stack on various BSD systems, and most of their efforts are in the direction of enhancing the stack and porting it to Linux. Currently the USAGI Project has an IPv6 implementation for the Linux kernel.

The output of the project is a free IPv6 stack for the Linux kernel and much improved IPv6 APIs in the glibc library. The project released its second stable release on February 5, 2001, and snapshots of updated code are available every two weeks.

The Linux Kernel and IPv6

The Linux kernel has its own IPv6 implementation. However, as mentioned previously, based on the TAHI Project results, this implementation proved to be not as good as other implementations. However, this is not a big surprise given that no major development activity has happened for a while. Another factor is that the original writer of the Linux kernel IPv6 code, Pedro Roque, has left the community, and since then, Alexey Kuznetsov and others have made quite a few enhancements over the years; however, this is not a major development. The USAGI Project has submitted some small fixes, but the question of a complete integration with the kernel stack is still an open issue.

Conformance Tests

At this point, we already have two IPv6 test networks in our lab. One network had Linux nodes with the USAGI IPv6 stack and the other had Linux nodes with the kernel IPv6 stack. Much work has been put to perform the setup and solve routing and tunneling issues. However, the question was still which implementation to adopt.

To be able to answer this question objectively, Ericsson Research (Budapest) has performed conformance tests for the latest version of the official Linux kernel (at that time kernel 2.4.5) and the USAGI IPv6 implementation (based on 2.4.0). The tests were based on the University of New Hampshire InterOperability Lab IPv6 Test Description document (see Resources).

The result of each test case can be:

- Pass: the implementation passes the test.
- Fail: the implementation fails the test.
- Inc: the verdict is inconclusive if we cannot decide whether the implementation is capable of passing the test. For example, when the test consists of three request/reply sequences and we do not get an answer for the tester's second request, then the verdict is inconclusive.

  The Conformance Lab conducted four types of testing: basic specification, address autoconfiguration, redirect and neighbor discovery. Below, we explain these tests and present the results.
- Basic specification: this series of tests covers the base specification for IPv6. The base specification specifies the basic IPv6 header and the initially defined IPv6 extension headers and options. It also discusses packet-size issues, the semantics of flow labels and traffic classes and the effects of IPv6 on upper-layer protocols (see Figure 1).
-

	Linux2.4.5	USAGI
unrecognized_next_header()	PASS	PASS
payload_length_zero()	PASS	PASS
next_header_zero()	INC	PASS
header_option_processing_order_a()	FAIL	PASS
header_option_processing_order_b()	INC	PASS
header_option_processing_order_c()	PASS	PASS
header_option_processing_order_d()	FAIL	PASS
option_processing_single_option_a()	FAIL	PASS
option_processing_single_option_b()	PASS	PASS
option_processing_single_option_c()	FAIL	PASS
option_processing_single_option_d()	FAIL	PASS
option_processing_single_option_e()	FAIL	PASS
option_processing_single_option_f()	PASS	PASS
option_processing_many_options_a()	PASS	PASS
option_processing_many_options_b()	PASS	PASS
option_processing_many_options_c()	PASS	PASS
option_processing_many_options_d()	FAIL	PASS
option_processing_many_options_e()	FAIL	PASS
routing_header_a()	FAIL	FAIL
routing_header_b()	FAIL	FAIL
routing_header_e()	FAIL	FAIL
routing_header_f()	FAIL	FAIL
routing_header2_a()	FAIL	FAIL
routing_header2_b()	FAIL	FAIL
routing_header2_c()	FAIL	FAIL
reassembly_frag_entry_a()	FAIL	FAIL
reassembly_frag_entry_b()	FAIL	FAIL
reassembly_frag_entry_c()	FAIL	FAIL
reassembly_frag_entry_d()	FAIL	FAIL
reassembly_time_exceeded_a()	FAIL	FAIL
reassembly_time_exceeded_b()	FAIL	FAIL
reassembly_time_exceeded_c()	FAIL	FAIL
reassembly_time_exceeded_d()	FAIL	FAIL
fragment_header_m_bit()	FAIL	FAIL
max_reassembly_size_exceeded()	FAIL	FAIL
stub_frag_header()	FAIL	FAIL
frag_multiple_m_bit_zero_a()	FAIL	FAIL
frag_multiple_m_bit_zero_b()	FAIL	FAIL
reassembly_unfragmentable_header_a()	FAIL	FAIL
reassembly_unfragmentable_header_b()	FAIL	FAIL
reassembly_unfragmentable_header_c()	FAIL	FAIL
no_next_header_a() // TR1, TR2	FAIL	FAIL
no_next_header_b()	FAIL	FAIL

Figure 1. Specification Test Results

- Address autoconfiguration: these tests cover address autoconfiguration for IPv6. They are designed to verify conformance with the IPv6 stateless address autoconfiguration specification (see Figure 2).
-

	Linux2.4.5	USAGI
duplicate_address_detection_a()	PASS	PASS
duplicate_address_detection_b()	PASS	PASS
duplicate_address_detection_c()	PASS	PASS
duplicate_address_detection_d()	PASS	PASS
preferred_greater_than_valid()	PASS	PASS
ra_from_global_source()	FAIL	FAIL
ra_oa_bits_reset()	FAIL	FAIL

Figure 2. Address Autoconfiguration Test Results

- Redirect: the redirect tests cover the redirect function of the neighbor discovery specification for IPv6. Redirect messages are sent by routers to redirect a host to a better first-hop router for a specific destination or to inform hosts that a destination is in fact a neighbor, i.e., on-link (see Figure 3).

	Linux2.4.5	USAGI
redirect_handling_a	PASS	INC
redirect_handling_b	INC	INC
redirect_handling_c	PASS	PASS
redirect_handling_d	INC	INC
redirect_handling_e	PASS	PASS
redirect_validation_1_a	PASS	PASS
redirect_validation_1_b	PASS	PASS
redirect_validation_1_c	PASS	PASS
redirect_validation_1_d	PASS	PASS
redirect_validation_1_e	PASS	PASS
redirect_validation_1_f	FAIL	FAIL
redirect_validation_2_a	PASS	PASS
redirect_validation_2_b	PASS	PASS
redirect_validation_2_c	INC	PASS
redirect_validation_2_d	PASS	PASS
unexpected_option_a	PASS	PASS
unexpected_option_b	PASS	PASS
unexpected_option_c	PASS	PASS
unexpected_option_d	PASS	PASS
no_dce_redirect_a	PASS	PASS
no_dce_redirect_b	PASS	PASS
neighbor_cache_redirect_a	PASS	PASS
neighbor_cache_redirect_b	PASS	PASS
neighbor_cache_redirect_c	PASS	PASS
redirected_next_hop_unreachable	PASS	PASS

- Figure 3. Redirect Test Results
- Neighbor discovery: these tests cover the neighbor discovery specification for IPv6. The neighbor discovery protocol is used by nodes (hosts and routers) to determine the link layer address for neighbors known to reside on attached links as well as to purge cached values that become invalid quickly. Hosts also use neighbor discovery to find neighboring routers that are willing to forward packets on their behalf. Finally, nodes use the protocol actively to keep track of neighbors that are reachable and those

that are not. When a router or the path to a router fails, a host actively searches for functioning alternates (see Figures 4 and 5).

	Linux2.4.5	USAGI
ND_neighbor_cache_stale_c	FAIL	FAIL
ND_neighbor_cache_stale_d	FAIL	FAIL
ND_neighbor_cache_stale_e	FAIL	FAIL
ND_neighbor_cache_stale_f	FAIL	FAIL
ND_neighbor_cache_stale_g	PASS	PASS
ND_neighbor_cache_stale_h	PASS	PASS
ND_neighbor_cache_delay_a	PASS	PASS
ND_neighbor_cache_delay_b	FAIL	FAIL
ND_neighbor_cache_delay_c	FAIL	FAIL
ND_neighbor_cache_delay_d	FAIL	FAIL
ND_neighbor_cache_delay_e	FAIL	FAIL
ND_neighbor_cache_delay_f	FAIL	FAIL
ND_neighbor_cache_delay_g	PASS	PASS
ND_neighbor_cache_delay_h	PASS	PASS
ND_neighbor_cache_probe_a	PASS	PASS
ND_neighbor_cache_probe_b	FAIL	FAIL
ND_neighbor_cache_probe_c	FAIL	INC
ND_neighbor_cache_probe_d	PASS	FAIL
ND_neighbor_cache_probe_e	INC	INC
ND_neighbor_cache_probe_f	INC	INC
ND_neighbor_cache_probe_g	PASS	PASS
ND_neighbor_cache_probe_h	FAIL	FAIL
ND_neighbor_cache_reachable_a	PASS	PASS
ND_neighbor_cache_reachable_b	FAIL	FAIL
ND_neighbor_cache_reachable_c	FAIL	FAIL
ND_neighbor_cache_reachable_d	FAIL	FAIL
ND_neighbor_cache_reachable_e	FAIL	FAIL
ND_neighbor_cache_reachable_f	FAIL	FAIL
ND_neighbor_cache_reachable_g	PASS	PASS
ND_neighbor_cache_reachable_h	PASS	PASS
ND_na_r_bit_change_a	INC	INC
ND_na_r_bit_change_b	INC	INC
ND_na_r_bit_change_c	INC	INC

- Figure 4. Neighbor Discovery Test Results

	Pass		Fail		Inconclusive	
	USAGI	Linux	USAGI	Linux	USAGI	Linux
Basic Specification	18	8	25	33	0	2
Address Autoconfiguration	5	5	2	2	0	0
Redirect	21	21	1	1	3	3
Neighbor Discovery	28	28	30	31	9	8
Total	72	62	58	67	12	13

Figure 5. Test Result Summary

Based on these results, we can see that the USAGI implementation had better results than the Linux kernel implementation; it passed more tests, failed fewer tests and had less inconclusive cases than the Linux kernel implementation.

Lessons Learned

Typically, working on such assignments, a lot of learning and competence building takes place. However, there are always some special lessons that you learn along the way, and they are worth mentioning.

It always pays to simplify problems. For instance, it was not obvious to know why a ping6 was able to crash the Linux kernel (v. 2.4.5) while we were running three web servers (Apache, Jigsaw and Tomcat) with patched code for IPv6 support in addition to an Alpha version of Java Merlin 1.4. Things got clearer when we disabled these servers and the Java Virtual Machine.

Also, adopting a complete implementation is much easier and more flexible than deploying patches. In some cases, we were faced with some bugs, and it was not easy to tell where they came from. Was it the application's IPv6 patches, the USAGI IPv6 kernel patches or the IPv6 patched binaries? Simplifying the problem helped a lot but proved that it may be easier to adopt the complete integrated implementation rather than patches that are downloaded and applied individually.

One important issue to mention is that our experience with the Open Source community has been a very positive one. People working with open-source software do their best to help you fix bugs and answers e-mails quickly. Since most of the software we use on our Linux clusters is based on open source, sometimes we go back to the community with questions, and we had no problems getting support from the developers—an important factor to acknowledge.

Conclusion

The Linux kernel IPv6 implementation has not undergone any major development since 2.4.0 was released. In many cases, we were able to crash the kernel with IPv6 traffic. On the other hand, USAGI provides an IPv6 implementation for Linux that is ported from the FreeBSD IPv6 stack, which is one of the best IPv6 implementations currently available. It also has proved to be more stable and reliable, and had better conformance test results than the Linux kernel implementation. The conclusion that can be drawn is that the USAGI stack is at least more mature than the kernel implementation and provides more functionalities and compliance with the specifications.

If you want to make a decision as to which IPv6 implementation to deploy on your Linux servers based on different factors, such as the development status, the number of people working on it, the support it's getting from industry and academia and its compliance with the latest RFCs, then it is most likely that you will choose the USAGI IPv6 implementation. Nevertheless, there is an important

concern related to both implementations. Unfortunately, a complete "implementation" document describing all the supported RFCs and motivations behind different design decisions does not exist.

I hope that the Linux kernel community and the USAGI Project can work together more tightly and organize their efforts to bring to pass a very stable and efficient IPv6 implementation for Linux. I believe the result would be a Linux kernel with a stable and well-written IPv6 implementation—a needed success formula for the future IPv6 servers contributing to the making of the mobile internet.

Acknowledgements

Open Systems Lab at Ericsson Research Canada for approving the publication of this article; Marc Chatel, Bruno Hivert and David Gordon at Ericsson Research Canada for their help and support in the lab; and Conformance Lab at Ericsson Research Hungary for conducting the conformance tests.

Resources

Ibrahim Haddad (Ibrahim.Haddad@Ericsson.com, ibrahim.haddad@lmc.ericsson.se) currently is a researcher at the Ericsson Corporate Research Unit in Montréal, Canada. He is primarily involved in researching carrier-class server nodes for real-time all-IP networks.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

The m4 Macro Package

Robert Adams

Issue #96, April 2002

Author will write an article discussing m4, and illustrating it's use in two domains: building dynamic web pages, and using it to build a simple “database”.

When you installed Linux, you installed a lot of cool programs, many of which you use every day. At the same time hundreds of little utilities also got installed, simply because they were required by the bigger, more elaborate applications. However, many of these little utilities are extremely useful in their own right. This article describes one of these second-class utilities: m4.

m4 is a macro processor, meaning that it copies its input to the output, expanding macros along the way. In this regard it's similar to another macro processor you're probably already familiar with: cpp (C Pre-Processor). Like cpp, m4 originally was written as the pre-processor for a programming language (Rational FORTRAN); however, m4 is much more powerful and feature-rich than cpp, which makes it much more useful than just defining constants in programs.

Being a macro processor, m4 certainly provides the ability to define simple macros, but it goes much further. You also can define parameterized macros (macros with arguments), conditionally include text into the output stream, do looping via recursion, run a shell command and insert its output into the output stream, include a file, perform simple string operations (length, substring search, regexp search, etc.), perform simple integer arithmetic and much more.

The examples that follow illustrate many of the features of m4, but this article can't do justice to everything m4 can do for you. Take a look at the info page (see Resources) for a complete description. Also, the examples below are abbreviated versions of m4 macros that I actually use. You can find the full source at my web page listed in Resources.

A word of warning from the m4 info page:

> Some people [find] m4 to be fairly addictive...learning how to write complex m4 sets of macros....Once really addicted, users pursue writing of sophisticated m4 applications even to solve simple problems....Beware that m4 may be dangerous for the health of compulsive programmers.

Defining and Using Macros

The basic tool of m4 is the macro. Macros are defined with the define command. For example, **define(`hello', `Hello, World')** defines a macro called hello. Notice the quote characters ` and '. These group words together to form phrases, and when they surround a single word they inhibit macro expansion. Usually, m4 will expand macros within both the macro name and the macro body unless you tell it not to by quoting.

Like cpp, you don't need any special commands or prefix characters to use macros. Everywhere the macro name appears in the input stream, m4 will substitute the macro body.

To run m4, simply give it the name of the file(s) to use as input. It reads through the input, expanding macros as it goes, and generates text on the standard output.

Assume we have the following file called sample.m4:

```
define(`hello', `Hello, World')
hello, welcome to m4!
```

If we run this file through m4 with the command

```
m4 sample.m4 > sample.txt
```

it produces the following output:

```
Hello, World, welcome to m4!
```

HTML Example

As I said above, m4 goes beyond simple macros. This example demonstrates some of m4's advanced features by defining some macros to ensure that HTML pages all have the same look and feel. HTML purists probably could do this with JavaScript and CSS, but it's so easy with m4 and doesn't require anything special from the browser.

Four macros are defined: one to produce the HTML header, a second to create a banner at the top of the HTML page, a third to create banners within an HTML page (like section headers) and a final one to do some housekeeping at the end of the page. We put these macros into a file called html.m4 and use it as a macro package later.

First, the macro to start an HTML page:

```
define(`START_HTML',
`<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;
   charset=iso-8859-1">
  <meta name="Author" content="D. Robert Adams">
  <title>$1</title>
</head>
<body text="#000000"
  ifdef(`BACKGROUND_IMAGE',
        `background="BACKGROUND_IMAGE"')
  bgcolor="#e5e5e5" link="#3333ff"
vlink="#000099"
  alink="#ffffff">
')
```

Line one defines a macro called START_HTML that expands to all the text that follows. Note the use of $1 on line seven. This expands to the first macro argument, which will become the page's title (see the next section for how these macros are used). Also note the use of m4's ifdef macro on line 11. **ifdef** checks if a macro has been defined. If it has, it includes the text given in the second argument. In this case, ifdef checks if BACKGROUND_IMAGE has been defined. If it has, we include the HTML code to use the image as the web page's background.

Next comes the page header macro:

```
define(`PAGE_HEADER',
`<table border="0" background="steel.jpg"
width="100%">
  <tr>
    <td align="left">$1</td>
    <td align="right">$2</td>
  </tr>
</table>
<div align=right>
  Last Modified: esyscmd(`date')
</div>
')
```

Again, note the use of $1 and $2 on lines four and five that get expanded with macro arguments. Further, note the esyscmd() macro on line nine. **esyscmd()** tells m4 to run the given shell command and insert its output at the given location. In this case, we run “date” to produce a timestamp in our document.

Next, we create the macro to make a section banner within the page. This uses nothing you haven't seen before:

```
define(`HTML_BANNER',
`<table border="0"
background="steel.jpg"
width="100%">
  <tr>
    <td>
      <img src="$2">
      <h2>$1</h2>
    </td>
  </tr>
</table>
')
```

The final piece necessary is the macro to close the HTML “body” and “html” tags:

```
define(`END_HTML', `</body></html>')
```

Given the above four macros, creating a web page is extraordinarily easy. Create a file (index.m4) that contains calls to our HTML macros. The only thing new is the include macro that inserts our HTML macros:

```
include(`html.m4')
define(`BACKGROUND_IMAGE', `bg.jpg')
START_HTML(`Sample Page')
PAGE_HEADER(`computer.jpg', `Sample HTML
Page')
HTML_BANNER(`img1.jpg', `News')
HTML_BANNER(`img2.jpg', `Downloads')
END_HTML
```

Once we've declared the macros and a file that uses them, the final step is to run m4 to create the HTML page. The command is:

```
m4 index.m4 > index.html
```

That's it! With just seven lines of code we create a fully functional HTML document. By using the macros to create other HTML pages, every page will have the same look and feel. Furthermore, we can change the look by simply changing the macro definitions and re-creating the HTML files.

A Simple Database Example

In this second example, we create a “database” of exam questions for a course. The goals are 1) to manage exam questions in a single repository, 2) to be able to create a LaTeX-format exam by simply choosing which questions to include and 3) to produce an answer key with equally little effort.

The question database consists of an m4 macro for each question. We store the macros in a file called questions.m4. For example:

```
define(`QUESTION_1',
  `Why did the chicken cross the road?
  ANSWER(1in, `To get to the other side')
')
```

Obviously, this macro will be expanded with the question itself, but note the use of the embedded macro ANSWER. It expands to one inch of vertical space if we're producing the exam, otherwise it expands to the answer if we're producing an answer key. ANSWER is defined as:

```
define(ANSWER, ifdef(`ANSWER_KEY', `Answer: $2',
`\vspace*{$1}'))
```

If ANSWER_KEY is defined, we include the answer ($2), otherwise we include some vertical space ($1) so the student can write in the answer.

Using the question macros is as easy as using the HTML macros. The complete exam code is stored in a file called exam.m4:

```
include(examMacros.m4)
EXAM_HEADER(`JOKE 101', `Fall 2001',
`First Exam')
include(questions.m4)
QUESTION_1
QUESTION_2
EXAM_TRAILER
```

The “include” on line one includes the code for EXAM_HEADER and EXAM_TRAILER that generate boilerplate LaTeX at the top and bottom of the exam. Line three includes the question macros we just created. Lines four and five include two questions from the database.

To create an exam, we run the command **m4 exam.m4 > exam.tex**. Because ANSWER_KEY was never defined, each question will include space for an answer. To create an answer key, we use m4's command-line options to temporarily define an ANSWER_KEY macro:

```
m4 -DANSWER_KEY exam.m4 > exam.key.tex
```

Conclusion

m4 is a tool that has applications in an endless number of domains. Anywhere you want to reduce duplication of effort, m4 can help. It is feature-rich enough that you can do almost anything and produce any kind of content you may wish. I hope I've given you enough to whet your appetite for m4 and to give you an appreciation for what m4 can do for you. Happy macro writing!

Resources

email: adams@csis.gvsu.edu

Robert Adams is a professor of computer science at GVSU. When he's not teaching he enjoys playing the fiddle, dancing and spending time with his daughter Turah.

Archive Index Issue Table of Contents

Advanced search

Hot Plug

Greg Kroah-Hartman

Issue #96, April 2002

Hot-pluggable devices have been created to solve a number of user needs. On laptop computers, PCMCIA devices were designed to allow the user to swap cards while the computer was still running. This allowed people to change network adaptors, memory cards and even disk drives without shutting down the machine.

The success of this led to the creation of the USB and IEEE1394 (FireWire) buses. These designs allow for peripherals to be attached and removed at any point. They also were created to try to move systems away from the ISA bus to a full Plug-and-Play-type system.

From the operating system's point of view, there are many problems with hot plugging devices. In the past, the operating system only had to search for the various devices connected to it on power-up, and once seen, the device would never go away. From the view of the device driver, it never expects to have the hardware that it is trying to control disappear. But with hot-pluggable devices, all of this changes.

Now the operating system has to have a mechanism that constantly detects if a new device appears. This usually is done by a bus-specific manager. This manager handles the scanning for new devices and recognizes this disappearance. It must be able to create system resources for the new device and pass control off to a specific driver. The device driver for a hot-pluggable device has to be able to recover gracefully when the hardware is removed and be able to bind itself to new hardware at any moment. Not only does the kernel need to know when devices are removed or added, but the user also should be notified when this happens. Other kinds of kernel events, such as the creation of network devices or the insertion of a laptop into a docking station, also would be useful for the user to know about.

This article describes the new framework in the Linux kernel for supporting USB and other hot-pluggable devices. It covers how the past implementation of PCMCIA loaded its drivers and the problems of that system. It presents the current method of loading USB and PCI drivers, and how this same framework can handle other kinds of user configuration issues easily.

The Past

Linux has had support for PCMCIA since 1995. In order for the PCMCIA core to be able to load drivers when a new device was inserted, it had a user-space program called cardmgr. The cardmgr program would receive notification from the kernel's PCMCIA core when a device had been inserted or removed and use that information to load or unload the proper driver for that card. It used a configuration file located at /etc/pcmcia/config to determine which driver should be used for which card. This configuration file needed to be kept up to date with which driver supported which card, or ranges of cards, and has grown to be over 1,500 lines long. Whenever a driver author added support for a new device, they had to modify two different files to enable the device to work properly.

As the USB core code became mature, the group realized that it also needed something like the PCMCIA system to be able to load and unload drivers dynamically when devices were inserted and removed. The group also noted that since USB and PCMCIA both needed this system, and that other kernel hot-plug subsystems also would use such a system, a generic hot-plug core would be useful. David Brownell posted an initial patch to the kernel (marc.theaimsgroup.com/?l=linux-usb-devel&m=96334011602320), enabling it to call out to a user-space program called /sbin/hotplug. This patch eventually was accepted, and other subsystems were modified to take advantage of it.

Let the Computer Do It Itself

All USB and PCI devices contain an identifier that describes either what kind of functions they support (like a USB audio or USB mass storage device), or if they do not support a class specification, they contain a unique vendor and product identifier. PCMCIA devices also contain these same kind of identifiers.

These identifiers are known by the PCI and USB kernel drivers, as they need to know which kind of devices they work properly for. The USB and PCI kernel drivers register with the kernel a list of the different types of devices that they support. This list is used to determine which driver will control which devices.

The kernel knows when and what kind of devices are inserted or removed from the system through the device bus core code (USB, FireWire, PCI, etc.). It can send this information to the user.

Taking these three pieces together (devices tell the computer what they are, drivers know what devices they support and the kernel knows what is going on) provides us with a solution to let the computer automatically load the proper driver whenever a new device is inserted.

/sbin/hotplug

The kernel hot-plug core provides a method for the kernel to notify user space that something has happened. The CONFIG_HOTPLUG configuration item needs to be selected for this code to be enabled. The notification happens when the kernel calls the executable listed in the global variable hotplug_path. When the kernel starts, hotplug_path is set to /sbin/hotplug, but the user can modify the value at /proc/sys/kernel/hotplug to change this. The kernel function call_usermodehelper() executes /sbin/hotplug.

As of kernel 2.4.14, the /sbin/hotplug method is being used by the PCI, USB, IEEE1394 and Network core subsystems. As time goes on, more subsystems will be converted to use it. Patches are available for the PnP-BIOS (notification when a laptop is inserted and removed from a docking station), Hot-Plug CPU, SCSI and IDE kernel subsystems. These are expected to be merged into the main kernel over time.

When /sbin/hotplug is called, different environment variables are set, depending on what action has just occurred.

PCI

PCI devices call /sbin/hotplug with the following arguments:

```
argv [0] = hotplug_path
argv [1] = "pci"
argv [2] = 0
```

and the system environment is set to the following:

```
HOME=/
PATH=/sbin:/bin:/usr/sbin:/usr/bin
PCI_CLASS=class_code
PCI_ID=vendor:device
PCI_SUBSYS_ID=subsystem_vendor:subsystem_device
PCI_SLOT_NAME=slot_name
ACTION=action
```

The action setting is “add” or “remove” depending on whether the device is being inserted or removed from the system. The class_code, vendor, subsystem_vendor, subsystem_device and slot_name environment settings represent the numerical values for the PCI device's information.

USB

USB devices call /sbin/hotplug with the following arguments:

```
argv [0] = hotplug_path
argv [1] = "usb"
argv [2] = 0
```

and the system environment is set to the following:

```
HOME=/
PATH=/sbin:/bin:/usr/sbin:/usr/bin
ACTION=action
PRODUCT=idVendor/idProduct/bcdDevice
TYPE=device_class/device_subclass/device_protocol
```

The action setting is “add” or “remove” depending on whether the device is being inserted or removed from the system, and idVendor, idProduct, bcdDevice, device_class, device_subclass and device_protocol are filled in with the information from the USB device's descriptors.

If the USB device's deviceClass is 0 then the environment variable INTERFACE is set to:

```
INTERFACE=class/subclass/protocol
```

This is because USB has a much more complex model for device configuration than PCI does.

If the USB subsystem is compiled with the usbdevfs filesystem enabled, the following environment variables also are set:

```
DEVFS=/proc/bus/usb
DEVICE=/proc/bus/usb/bus_number/device_number
```

where bus_number and device_number are set to the bus number and device number that this specific USB device is assigned.

Network

The network core code calls /sbin/hotplug whenever a network device is registered or unregistered with the network subsystem, and /sbin/hotplug is called with the following arguments when called from the network core:

```
argv [0] = hotplug_path
argv [1] = "net"
argv [2] = 0
```

and the system environment is set to the following:

```
HOME=/
PATH=/sbin:/bin:/usr/sbin:/usr/bin
```

```
INTERFACE=interface
ACTION=action
```

The action setting is “register” or “unregister” depending on what happened in the network core, and interface is the name of the interface that just had the action applied to itself.

CPU

The Hot-Plug CPU patch (available at sourceforge.net/projects/lhcs) calls /sbin/hotplug after a CPU is removed or added to the system, and /sbin/hotplug is called with the following arguments:

```
argv [0] = hotplug_path
argv [1] = "cpu"
argv [2] = 0
```

and the system environment is set to the following:

```
HOME=/
PATH=/sbin:/bin:/usr/sbin:/usr/bin
CPU=cpu_number
ACTION=action
```

The action setting is “add” or “remove” depending on what happened to the CPU, and cpu_number is the number of the CPU that just had the action applied to itself.

Examples

The /sbin/hotplug script can be a very simple script if you only want it to control a small number of devices. For example, if you have a USB mouse and wish to load and unload the kernel driver whenever the mouse is inserted or removed, the following script, located at /sbin/hotplug, would be sufficient:

```
#!/bin/sh
if [ "$1" = "usb" ]; then
    if [ "$INTERFACE" = "3/1/2" ]; then
        if [ "$ACTION" = "add" ]; then
            modprobe usbmouse
        else
            rmmod usbmouse
        fi
    fi
fi
```

Or if you want to run ColdSync (www.ooblick.com/software/coldsync) automatically when you connect your USB HandSpring Visor to the computer, the following script located at /sbin/hotplug would work well:

```
#!/bin/sh
USER=gregkh
if [ "$1" = "usb" ]; then
    if [ "$PRODUCT" = "82d/100/0" ]; then
        if [ "$ACTION" = "add" ]; then
            modprobe visor
```

```
            su $USER - -c "/usr/bin/coldsync"
        else
            rmmod visor
        fi
    fi
fi
```

If you want to make sure that your network devices always come up connected to the proper Ethernet card, the following /sbin/hotplug script, contributed by Sukadev Bhattiprolu, can do this:

```
#!/bin/sh
if [ "$1" = "network" ]; then
    if [ "$ACTION" = "register" ]; then
        nameif -r $INTERFACE -c /etc/mactab
    fi
fi
```

Listing 1 shows a more complex example that can handle automatically loading and unloading modules for three different USB devices.

Listing 1. Script to Load and Unload Modules Automatically for Three Different USB Devices

Need for Automation

The previous small example shows the limitations of being forced to enter in all of the different device IDs manually, product IDs and such in order to keep a /sbin/hotplug script up to date with all of the different devices that the kernel knows about. Instead, it would be better for the kernel itself to specify the different types of devices that it supports in such a way that any user-space tools could read them. Thus was born a macro called MODULE_DEVICE_TABLE() that is used by all USB and PCI drivers. This macro describes which devices each specific driver can support. At compilation time, the build process extracts this information out of the driver and builds a table. The table is called modules.pcimap and modules.usbmap for all PCI and USB devices, respectively, and exists in the directory /lib/modules/*kernel_version*/.

For example, the following code snippet from drivers/net/eepro100.c:

```
static struct pci_device_id eepro100_pci_tbl[]
__devinitdata = {
    { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL_82557,
      PCI_ANY_ID, PCI_ANY_ID, },
    { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL_82562ET,
      PCI_ANY_ID, PCI_ANY_ID, },
        { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL
          _82559ER, PCI_ANY_ID, PCI_ANY_ID,
},
        { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL
          _ID1029, PCI_ANY_ID, PCI_ANY_ID,
},
        { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL
          _ID1030, PCI_ANY_ID, PCI_ANY_ID,
},
        { PCI_VENDOR_ID_INTEL, PCI_DEVICE_ID_INTEL
          _82801BA_7, PCI_ANY_ID, PCI_ANY_ID,
},
        { 0,}
```

```
};
MODULE_DEVICE_TABLE(pci, eepro100_pci_tbl);
```

causes these lines to be added to the modules.pcimap file:

```
eepro100 0x00008086 0x00001229 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
eepro100 0x00008086 0x00001031 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
eepro100 0x00008086 0x00001209 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
eepro100 0x00008086 0x00001029 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
eepro100 0x00008086 0x00001030 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
eepro100 0x00008086 0x00002449 0xffffffff 0xffffffff
0x00000000 0x00000000 0x00000000
```

As the example shows, a PCI device can be specified by any of the same parameters that are passed to the /sbin/hotplug program.

A USB device can specify that it can accept only specific devices such as this example from drivers/usb/mdc800.c:

```
static struct usb_device_id
  mdc800_table [] = {
   { USB_DEVICE(MDC800_VENDOR_ID, MDC800_PRODUCT_ID) },
   { } /* Terminating entry */
};
MODULE_DEVICE_TABLE(usb, mdc800_table);
```

which causes the following line to be added to the modules.usbmap file:

```
mdc800 0x0003 0x055f 0xa800 0x0000 0x0000 0x00 0x00
0x00 0x00 0x00 0x00 0x00000000
```

Or it can specify that it accepts any device that matches a specific USB class code, as in this example from drivers/usb/printer.c:

```
static struct usb_device_id usblp_ids [] = {
  { USB_INTERFACE_INFO(USB_CLASS_PRINTER, 1, 1) },
  { USB_INTERFACE_INFO(USB_CLASS_PRINTER, 1, 2) },
  { USB_INTERFACE_INFO(USB_CLASS_PRINTER, 1, 3) },
  { }    /* Terminating entry */
};
MODULE_DEVICE_TABLE(usb, usblp_ids);
```

which causes the following lines to be added to the modules.usbmap file:

```
printer 0x0380 0x0000 0x0000 0x0000 0x0000 0x00 0x00
0x00 0x07 0x01 0x01 0x00000000
printer 0x0380 0x0000 0x0000 0x0000 0x0000 0x00 0x00
0x00 0x07 0x01 0x02 0x00000000
printer 0x0380 0x0000 0x0000 0x0000 0x0000 0x00 0x00
0x00 0x07 0x01 0x03 0x00000000
```

Again these USB examples show that the information in the modules.usbmap file matches the information provided to /sbin/hotplug by the kernel, enabling /sbin/hotplug to determine which driver to load without relying on a hand-generated table, as PCMCIA does.

Preprocessor Abuse

The macro MODULE_DEVICE_TABLE automatically creates two variables. For the example: MODULE_DEVICE_TABLE (usb, usblp_ids); the variables __module_usb_device_size and __module_usb_device_table are created and placed into the read-only data section and the initialized data section of the module, respectively. The variable __module_usb_device_size contains the value of the size of the struct usb_id structure, and __module_usb_device_table points to the usblp_ids structure. The usblp_ids variable is an array of usb_id structures with a terminating NULL structure at the end of the list.

When the depmod program is run, as part of the kernel installation process, it goes through every module looking for the symbol __module_usb_device_size to be present in the compiled module. If it finds it, it copies the data pointed to by the __module_usb_device_table symbol into a structure, extracts all of the information and writes it out to the modules.usbmap file, which is located in the module root directory. It does the same thing while looking for the __module_pci_device_size in creating the modules.pcimap file.

With the kernel module information exported to the files modules.usbmap and modules.pcimap, our version of /sbin/hotplug can look like Listing 2 [available at ftp.linuxjournal.com/pub/lj/listings/issue96/5604.tgz]. This example only tests for a match of the USB product ID and vendor IDs. The Linux-Hotplug Project has created a set of scripts that covers all of the different subsystems that can call /sbin/hotplug. This enables drivers to be loaded automatically when new devices are inserted into the systems. It also starts up network services when network devices are seen. These scripts are released under the GPL and are available at linux-hotplug.sourceforge.net. Almost all major Linux distributions are currently shipping this package, so it is probably already on your machine.

The Future

The current /sbin/hotplug subsystem needs to be incorporated into other kernel systems, as they develop hot-plug capability. SCSI, IDE and other systems all have hot-plug patches available for kernel support but need to have script support, kernel macro support and modutils depmod support added in order to provide the user with a consistent experience.

As the kernel boots, and discovers new devices, it tries to spawn /sbin/hotplug, but since user space has not been initialized yet, it cannot run. This means that any USB or PCI devices that are needed at boot time need to be compiled into the kernel or exist in an initrd RAM disk image as a module. Sometime during the 2.5 development process, the initrd RAM disk image will be converted to contain an entire small user-space tree. This will allow /sbin/hotplug to be run

during the boot process and load modules dynamically. Some links describing this disk image idea are: lwn.net/2001/0712/kernel.php3 -- marc.theaimsgroup.com/?l=acpi4linux&m=99705696732868 -- marc.theaimsgroup.com/?l=linux-kernel&m=99436439232254 and marc.theaimsgroup.com/?l=linux-kernel&m=99436253707952.

Because of the small space requirements of this RAM disk image, the dietHotplug program has been written. It is an implementation of the Linux-Hotplug bash scripts in C and does not require modules.*map files when the program runs. The executable size of the entire dietHotplug program is one-fifth of the size of the original modules.*map files themselves. The small size is due to the use of dietLibc (found at www.fefe.de/dietlibc) and other space-saving techniques. **dietHotplug** will undergo more development as the 2.5 kernel requirements are more fully known. **dietHotplug** can be downloaded from the Linux-Hotplug site.

Acknowledgements

I would like to thank David Brownell who wrote the original /sbin/hotplug kernel patch and most of the Linux Hotplug scripts. Without his persistence, Linux would not have this user-friendly feature. I also would like to acknowledge the entire Linux USB development team, who have provided a solid kernel subsystem in a relatively short amount of time.

Keith Owens wrote the supporting code in the depmod utility and has endured constant changes to the format of the MODULE_DEVICE_TABLE() USB structure.

The other developers on the linux-hotplug-devel mailing list who have helped with their patches and feedback on the hot-plug scripts also deserve recognition, along with the wonderful Linux distribution-specific support that Debian, Red Hat and Mandrake have provided.

This article was based upon a paper and presentation that I gave at the 2001 Ottawa Linux Symposium.

Greg Kroah-Hartman is currently the Linux USB and PCI Hotplug kernel maintainer. He works for IBM, doing various LInux kernel-related things and can be reached at greg@kroah.com.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Writing Zope Products

Reuven M. Lerner

Issue #96, April 2002

Last month, we continued our discussion of the Zope web development environment by installing and examining Zope products. As we saw, each product is actually a Python object module that can be instantiated one or more times on a Zope server. Hundreds of products are available for Zope, ranging from a small and lightweight fortune-teller to the large and impressive content management framework (CMF).

Many Zope administrators are content to install and use the products that they can download from the Web. And indeed, there are enough products to satisfy most basic sites; whatever you cannot do with an existing product is often simple enough to create in DTML, Zope's Dynamic Template Markup Language (described in the February 2002 issue of *LJ*).

But as easy and straightforward as it may be to do many tasks in DTML, it is neither as complete nor as flexible as Python. And while the addition of PythonScripts (and PerlScripts!) in Zope has certainly reduced the need to write products for many medium-sized tasks, most Zope hackers eventually find themselves writing a product of some sort.

This month, we look at how to build a simple Zope product that we can then integrate into our site. As you will see, it is quite easy to create a Zope product that integrates well into the rest of this environment.

A Trivially Simple Product

At its core, a Zope product is a Python module. The product (as we saw last month) is installed into the lib/python/Products directory under the main Zope directory. Zope looks for products when it is started (or restarted), which means that you must start or restart Zope after installing a new product.

We can instantiate an installed product as many times as we like, placing each instance somewhere in the web hierarchy. Each instance has a unique “id” attribute that both uniquely identifies it within a folder and allows us to invoke methods on the object.

If this already sounds confusing, remember that the URL /foo/bar normally means that a web server should retrieve the file “bar” within the directory “foo”. But in Zope, the URL foo/bar means that the system should invoke the method “bar” on the object “foo”. In other words, foo/bar gets turned into foo.bar. And when we say “the object foo”, we really mean to say, “the object whose ID is foo”. Setting the ID is essential if we are going to invoke methods on our object successfully.

We also will need to define a meta_type for our product. The meta_type name is the text string that will appear in the Add pull-down list of Zope products in the upper-right corner of the /manage screen. In general, you can name the meta_type the same as your class or perhaps something a bit easier to understand. Remember that items in the Add menu are displayed in ASCII order, which means that capital “Z” comes before lowercase “a”.

To create our own product, we need to do the following:

- Define a class in its own module and install it in lib/python/Products.
- Define an __init__ method in which we assign the “id” instance variable a value.
- Define one or more methods whose return value is a text string containing HTML.
- Define a meta_type class variable, which sets the meta_type for all instances of our product.

This turns out to be surprisingly simple to do, as you can see in Listing 1, which defines helloworld.py, a simple Zope product that you can install and almost instantiate in your site. (We'll soon see how to get around these limitations.)

Listing 1. helloworld.py, a Simple Zope Product

There are several important items to notice in our helloworld class. For starters, the class and its methods contain documentation strings. It's always a good idea to write docstrings, and Python's inclusion of such a feature is a rare and refreshing reminder that programmers can and should include documentation along with the source code. Zope enforces this restriction by mandating that classes and methods must include docstrings if you want them to be used in the system.

Our class also defines two methods, __init__ and index_html. The __init__ method is invoked automatically by Zope when it creates an instance of our object and typically is used to initialize instance variables and define other behaviors that will be needed later on. In this particular case, __init__ defines a single instance variable (self.id) that allows our object to keep track of its identity. As you might expect, __init__ is not meant to be invoked from the outside world, but rather from within Zope itself.

The index_html method, by contrast, is designed to be invoked via a URL. If we place an instance of “helloworld” in the root (/) directory of our Zope server, we can invoke the index_html method on it with a URL of /helloworld/index_html. But index_html is special; like the index.html file on many Apache servers, it is invoked by default if no other method is named explicitly.

Finally, notice that index_html returns HTML to its caller. It does not return a status code or anything other than the HTML.

What's Missing?

helloworld.py is a perfectly legal Zope product; we can install it in lib/python/Products, and Zope will not mind. Unfortunately, Zope also will fail to notice that helloworld.py is there at all, will not add it to the Add selection list and generally will ignore the work we did in writing our product. It's clear that we will need to beef up our skeletal product if we want it to interact with Zope. We will call this enhanced version the “smallhello” product.

For starters, we must change the structure of our product from a single standalone module file (helloworld.py) into a full-fledged Python package. A package is a directory (smallhello) within the Python search path (defined by the variable sys.path) containing one or more Python source files. In our case, the smallhello directory will contain two files: a smallhello.py file that is quite similar to helloworld.py (see Listing 2) and __init__.py (see Listing 3), which initializes and helps to register our object.

Listing 2. smallhello/smallhello.py

Listing 3. smallhello/__init__.py

The __init__.py file first imports smallhello.smallhello, defining the module's methods and attributes. But the crucial part of __init__.py, at least as far as Zope is concerned, is the initialize method. After Zope discovers and imports smallhello, it invokes smallhello.initialize, passing it a ProductContext object (called “context”). In other words, initializing an object results in that object registering itself with the server.

The initialize routine itself is pretty straightforward, although our version does some rudimentary error trapping (using try/except) to ensure that things work correctly. Our smallhello product only passes two arguments to context.registerClass: the finalhello.finalhello object that we want to add and then a tuple of constructors that should be invoked when we want to create a new instance of our product. Remember to include a trailing comma if you pass a single item to constructors; otherwise, Zope will fail to load the product.

The constructors parameter is just one of a number of named parameters that we can pass to context.registerClass to customize the way in which our object is registered with Zope. For example, we also can pass an icon parameter that tells Zope which graphic (a filename inside of our package directory) should be placed next to instances of our package within Zope.

Changes to Our Object

Turning helloworld.py into smallhello.py (see Listing 2) requires some small changes. We start by adding a method that allows Zope to create new instances of our product. By convention, such management-related methods begin with manage_, so our method is called manage_smallhello. This is the same method that was named in the constructors tuple passed to context.registerClass.

The most significant change to our smallhello class is also one of the least obvious: we have made it a subclass of OFS.SimpleItem.SimpleItem, one of the Zope product base classes provided as part of the Zope package (in the OFS.SimpleItem package). Without inheriting from SimpleItem, many things—from cutting and pasting of objects to acquisition—would fail to work as we expect, if at all. There are several possible base classes from which your products can inherit; SimpleItem, as its name implies, is designed to be the simplest and most straightforward of the bunch.

While modifying smallhello.py, I decided to add two other methods that produce content. One of them, other_html, produces output that is similar to index_html—except, of course, index_html is displayed by default when no other method is specified, while other_html only is called when explicitly named in the URL.

I also added a foo_file method that demonstrates how to return the contents of an HTML (or DTML) file from disk. It can be annoying and frustrating to put all of your HTML inside a Python module file; this way, you can keep the DTML files inside of the package directory but modify them independently of the program itself. Note that we had to import the HTMLFile method from the Globals package in order for this to work.

I modified the __init__ function in smallhello.py to take three arguments: self, id and title. (Previously, it only accepted self and id.) The __init__ function is invoked each time a new instance of smallhello.py is created, which is done through a call to manage_smallhello. Inside of manage_smallhello, our call to self._setObject sets the object ID to a generic smallhello_id, with a title of smallhello_title. Because we hard code the ID in our example, and because IDs must be unique within a folder, this means that we only can have one instance of our smallhello product in a given folder. There isn't enough space here to describe how to read and write parameters, but a quick look at the examples mentioned in the Resources section should make it obvious how to do this.

After the call to self._setObject, manage_smallhello then redirects the user's browser to the main (index_html) method. We could display some output instead, redirecting the user's browser to another method in our object, but I took the easy way out and decided to send users to /index.html on our site.

After you have installed the smallhello product, you can stop Zope and restart it again. You should see “smallhello” near the bottom of the Add menu; selecting it will send you to the index.html page on your Zope site. Because we haven't made our product as user-friendly as it could be, you will have to enter the URL manually (index_html, other_html or foo_file) in your browser. But there isn't any reason why these pages cannot contain links to each other or why other pages on the site cannot link to them.

What do you know? We've created a Zope product!

What's Missing Now?

If we were to release our simplehello project as it currently stands, no one would really want to use it. In addition to the problems mentioned above (e.g., the lack of unique ids for individual instances), our product lacks the management tabs that make Zope so user-friendly for administrators. It also fails to handle security permissions in a standard or easy way.

It is almost as easy to install these features as the others we have seen so far. For example, each tab is represented by a dictionary containing two name-value pairs, label and action. The value associated with label is what the user sees on the screen, while the value associated with action tells Zope which method should be invoked when someone clicks on the appropriate tab. To install your tabs in Zope, define a manage_options tuple in your object, the members of which are the dictionaries describing the tabs.

One of the most important items that we haven't addressed so far is user input. This is actually a pretty easy issue to address because Zope treats HTML form

inputs as if they were standard parameters to a method. For example, consider the following HTML form:

```
<form action="manage_edit" method="POST">
    <p>id: <input type="text"
name="id"></p>
    <p>Title: <input type="text"
name="title"></p>
    <p><input type="submit"></p>
</form>
```

Clicking on the “submit” button will submit the name-value pairs for id and title to our product's manage_edit method. We can define that method with a signature like the following:

```
def manage_edit(self, id, title):
```

Within this method, we can retrieve the values of the id and title HTML form elements using the variables of the same names.

Conclusion

Zope products are a more advanced and sophisticated way to build Zope applications than DTML files, giving greater flexibility but also requiring greater discipline and understanding of the underlying mechanisms. Knowing how to write Zope products is something like knowing how to write mod_perl modules for Apache; it means that the underlying system is completely at your disposal.

Unfortunately, while programmers can take advantage of a rich API for creating their own Zope products, the lack of good introductory documentation has scared many people from trying. Our simplehello product demonstrates that with just a little code, you can get impressive and useful applications working in a short period of time.

Resources

email: reuven@lerner.co.il

Reuven M. Lerner is a consultant specializing in web/database applications and open-source software. His book, Core Perl, was published in January 2002 by Prentice-Hall. Reuven lives in Modi'in, Israel, with his wife and daughter.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Interoperate with Me

Marcel Gagné

Issue #96, April 2002

Yes, François? It does look like something from the 1980s. That's because it is. You are right, *mon ami*. It is true that I am still running on my Linux system, but the effect is wonderfully nostalgic, *non*? *Oui*, there are several examples on tonight's menu, and I think our friends will be delighted. What did you say, François? Ah, they are already here!

Bonjour, mes amis. Welcome to *Chez Marcel*, home of the Linux world's greatest wine cellar and the finest Linux cooking. Your tastebuds are tingling in anticipation, is it not true? Then allow my faithful waiter to show you to your chairs, and we will have a look at the menu *immédiatement*. François, please, to the wine cellar. Bring up the 1989 Pessac-Léognan—it will be a perfect match for what is to come.

By definiton, interoperability implies the ability of some technology to work easily with some other technology or some piece of software with another piece of software. I tell you now, *mes amis*, that this concern for the machine side of things makes me want to drown my sorrows in a rich crème caramel. Come to think of it, I may do that regardless of how things turn out, *non*?

What was I saying, *mes amis*? *Ah, oui*. What about the software working with the wetware as they say—the person working with the machine. What makes Linux so wonderful to work with is that it gives us choices that other environments do not provide. Those choices are so exceptional that we even can choose to run our Linux systems with the friendly face and flavor of an earlier operating system we knew and felt comfortable with. After all, it is what is inside that counts, *non*?

You probably are running a flashy new GNOME or KDE desktop environment, and yet you long for a foray into the desktops of your youth, whether it be Amiga, an old Macintosh or even that particular ubiquitous desktop from circa

1995. Under no circumstances do you wish to give up the power of your Linux desktop, but the appearance is another matter.

When you start your GNOME or KDE desktop, you are running a window manager on top of the X Window System graphical manager (or simply X). On most Linux systems, this is the XFree86 server. If your system starts out at a text login and you start X with **startx**, then you are ready to experiment. If your desktop starts at a graphical login, you may want to switch back to a text login for a taste of these recipes. Log out from your desktop, switch to one of your virtual consoles (with Ctrl-Alt-*Fn*, where *Fn* is a function key from 2 to 6) and log in as root. Now type **init 3**, and the graphical login manager will exit. There are numerous ways to do this. Your graphical login screen even may have an option to switch to console mode.

The master program that starts all desktops is xinit. The startx program is just a script that calls xinit (with a few flags and parameters). Still, you could restart your KDE window manager like this:

```
xinit /usr/bin/startkde
```

If you do it this way, make sure you always specify the full path to the window manager. Another thing you can do is create a .xinitrc file in your home directory with a single line in it:

```
exec startkde
```

Doing it this way, you can start your window manager with the classic startx command. And indeed, classics is what we were talking about, *mes amis*--this 1989 red is most certainly a classic. François, please pour for our guests. While you enjoy your drinks, let me show you the first of our desktop classics, Takashi Hasegawa's MLVWM, which stands for Macintosh-Like Virtual Window Manager. Start by visiting his site, pick up the latest source and build it:

```
tar -xzvf mlvwm091.tar.gz
cd mlvwm091
xmkmf -a
make
make install
```

While you are there, you also may want to download the mini-icon set (which actually comes from the FVWM desktop) and extract those:

```
cd /home/natika/pixmaps
tar -xzvf mini-icons.tar.gz
```

Before you start up your new (or old) Macintosh look-alike desktop, you must first create a .mlvwmrc file. In the distribution directory (where you built mlvwm), you'll see a sample_rc directory. Simply copy all these files over to the user directory, then rename the mlvwmrc file. Let's continue with our friendly user, natika:

```
cd /home/natika
cp /pathto/mlvwm091/sample_rc/* .
mv Mlvwmrc .mlvwmrc
```

We are almost there. The default configuration file is a good place to start, but you will want to edit the path to those pixmaps. Look for the following line and change it to reflect the path of your system icons (or the mini-icons you just installed):

```
IconPath /usr/local/include/X11/pixmaps:/home2/tak
/bin/pixmap
```

Now, fire it up with **xinit /usr/bin/X11/mlvwm**. Figure 1 shows MLVWM in action. It is all coming back to you now, eh, *mes amis*?

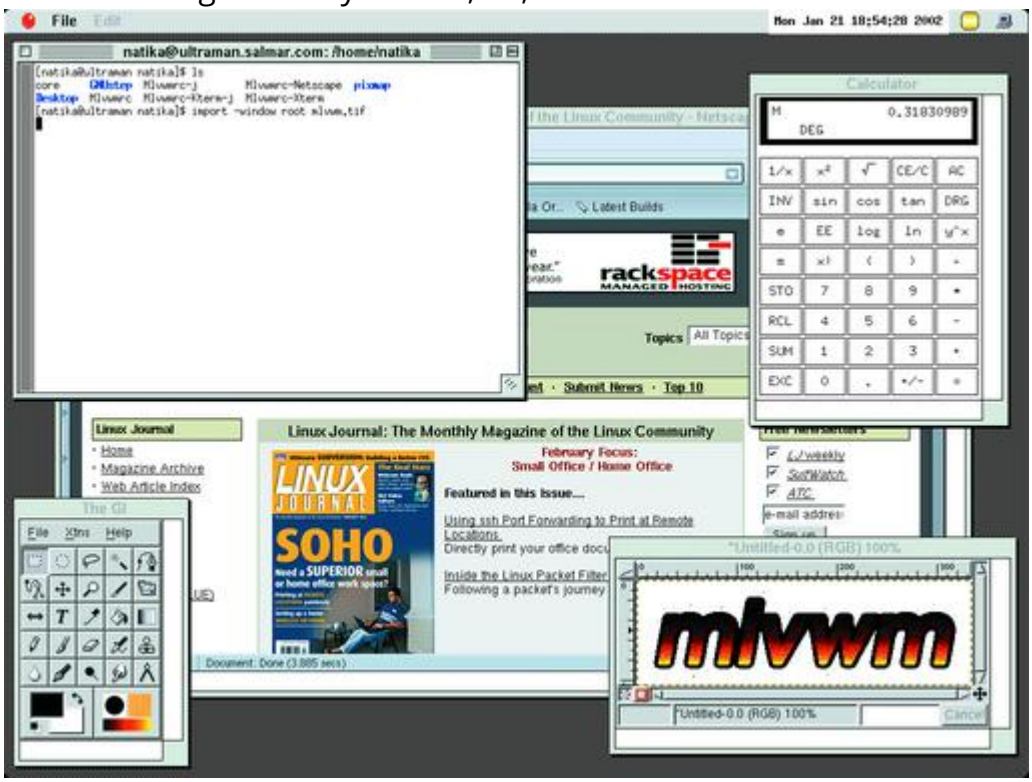


Figure 1. MLVWM in Action

For those who desperately need a dose of something friendlier, there was always the Amiga. Did I say “was”? *Mais non*, if not the Amiga itself, certainly the spirit lives in Marcus Comstedt's AMIWM. Start by visiting his web site and downloading the source. Then, as many times in the past, we build *comme ça*:

```
tar -xzvf amiwm0.20pl48.tar.gz
cd amiwm0.20pl48
./configure
make
make install
```

Check the included README for configuration file options. These runtime options come from either a local .amiwmrc file or the global system.amiwmrc. By default, you will find the global file in the /usr/local/lib/amiwm directory. Have a look at Figure 2 to see the AMIWM desktop.

Figure 2. Who Says the Amiga Is Dead?

Those desktop alternatives provide a means to experience the tastes of the past. KDE and GNOME are the present. But what about the future-uture-uture? Do any of you hear an echo? Anyhow, Niklas Elmqvist, Robert Karlsson, Steve Houston, Antony Suter and others are working on providing us a glimpse of the future. Enter 3Dwm, a three-dimensional desktop that takes us out of window manager flatland and transports us into a world of depth. In fact, the 3Dwm team would like us to forget about the wm part of the name and think of 3Dwm as an environment for developing three-dimensional user interfaces rather than a window manager.

Have a look at Figure 3 for a sample of what is to come. Are you ready to step into the third dimension? Let's go then.

Figure 3. Desktops Enter the Third Dimension with 3Dwm

Since this is a 3-D application, you'll need your Mesa or OpenGL libraries. You'll need the expat libraries for the XML parts of the project, and the SDL libraries also are required. Pay attention to the release number of the SDL package (which you can get at www.libsdl.org). It must be version 1.2 or better for this to work properly. This seems like a lot just to get started, but luckily, most modern Linux distributions will have all these things already installed.

In order to build 3Dwm, you will need a few additional things. The biggest of these is the OmniORB package from AT&T's UK Labs, the same people who brought you VNC (an excellent remote control package). I will make a strange suggestion here. For the most part, I have little trouble recommending source packages for installs, but this is one place where it might make more sense to pick up one of the binary packages. Since dependencies are certainly a concern, those of you running an RPM-based system also might consider grabbing the source RPM and rebuilding it for your release and architecture. That is what I did. Then, I rebuilt a new binary RPM package in this way:

```
rpm --rebuild omniORB-3.0.4-0.src.rpm
```

This ticks along for quite some time. When it is complete, you should have both a complete omniORB and omniORB-devel package sitting in your build directory (which will vary from system to system). On my Red Hat test system, the finished RPMs were in /usr/src/redhat/RPMS/i386. Debian users will find prebuilt packages as well.

You could continue with the 3Dwm build now and do this process later, but let's start it now so that we don't forget. Modify your /etc/profile to include the following lines:

```
OMNIORB_CONFIG=/etc/onmiORB.cfg
OMNIORB_LOGDIR=/var/omninames
```

The omniORB.cfg file doesn't exist yet, so you must now create it. Here's what you need there:

```
ORBInitialHost localhost
ORBInitialPort 8088
```

Note that the “localhost” designation may not work depending on your hostname configuration. Because I start my desktop as a host within my domain, I had to use the fully qualified domain name here. You may have to do the same. Now, start the omniNames server like this:

```
omniNames --start 8088 &
```

On the dowload page for 3Dwm, you'll also find the source for meshio, a library for loading the 3-D model files. Yes, there are several pieces involved in this recipe, but the result will be worth it, I assure you. It is time now to compile meshio. You'll be happy to know that this is a quick process:

```
tar -xzvf meshio-0.2.0.tar.gz
cd meshio-0.2.0
make
make install
```

Now, it's time to build 3Dwm itself. At the time of this writing, the release level was 0.3.1, which technically still qualifies as alpha software, but we Linux-types, we love to live on the edge *non*? For the compile to go through properly, you will need to set the PYTHONPATH variable:

```
export PYTHONPATH=/usr/lib/python2-1/site-packages
```

And now, we build:

```
tar -xzvf 3dwm-0.3.1.tar.gz
cd 3dwm-0.3.1
./configure
make
```

When this is all complete, copy the configuration file tdwmrc (which you'll find under /etc in your build directory) to ~/.tdwmrc. Open it with your favorite text editor and make sure that the path to the default.zorn is correct. Remember that this will vary depending on the path to your install directory. Now, start the display server:

```
cd server
./tdwm-server
```

If everything went well, you should see a black display box appear on your screen. That is success. This may not sound interesting, but 3Dwm is a client-server application, and the 3-D client will run in this window. To jump into your 3-D world, you must now run a client. The 3Dwm site has a 3dwm-data.tar.gz file that you should download. It contains some sample 3-D models and textures. All you have to do is extract the file; there is no compiling to do here:

```
cd clients/geoclient
./geoclient office.3ds
```

A three-dimentional representation of an office (complete with desk, monitor, keyboard and mouse) will appear in the 3Dwm server window. If it looks a bit strange, try navigating through it like this.

Moving around includes some combination of the keyboard Ctrl key and mouse buttons. These combinations, by the way, are defined in that default.zorn file I mentioned earlier. If you hold the Ctrl key and click the left mouse button (while moving the mouse), you change the orbit around the focal point. Pressing Ctrl and the right mouse button zooms your focal point in or out. Ctrl and the center mouse button pans your focal point. Ctrl, Shift and the left button rotates left and right, while Ctrl, Shift and the right button changes your field of view.

There are other clients besides the geoclient, but I'm going to leave that exploration to you. Included with your 3Dwm package is a 3-D virtual clock, and as you can see from our 3Dwm virtual clock, the hour for closing time, she approaches once again, *non*? François, if you would be so kind as to refill our guests' glasses a final time. *Merci, mon ami*.

Whether you find your ideal work environment in the desktops of the past, the present or even the future, rest assured that when you cook with Linux, interoperability isn't a fancy technological catch phrase—it's the way we work each and every day.

A votre santé! *Bon appétit*!

Resources

Marcel Gagné (mggagne@salmar.com) is president of Salmar Consulting Inc., a systems integration and network consulting firm and the author of Linux System Administration: A User's Guide, published by Addison-Wesley.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Hardening Sendmail

Mick Bauer

Issue #96, April 2002

Mick examines sendmail's security controversies and builds an SMTP gateway for handling internet mail.

Ah, sendmail. You either love it for being so versatile and ubiquitous, or you hate it for being bloated, complicated and insecure. Or perhaps you're a complete newcomer to the e-mail server game and would like to give sendmail a try (after all, sendmail is arguably the most popular open-source software package on the Internet).

Well, contrary to popular belief, sendmail isn't a total loss where security is concerned, nor does it require learning the arcane syntax of sendmail.cf (although hardcore sendmail gurus do indeed master it). This month we examine these and other sendmail security controversies, using sendmail's handy m4 macros to rapidly build a secure but functional Simple Mail Transport Protocol (SMTP) gateway to handle internet mail.

Why (or Why Not) Use Sendmail?

Sendmail is one of the most venerable internet software packages still in widespread use. It first appeared in 4.1c BSD UNIX (April 1983), and to this day it has remained the most relied-upon application of its kind. Among message transfer agents (MTAs), sendmail is the great workhorse of the Internet, transferring e-mail between networks dependably and (to end users) transparently. But sendmail has both advantages and disadvantages.

On the good side, sendmail has a huge user community, with the result that it's easy to find both free and commercial support for it, not to mention a wealth of electronic and print publications. It's also stable and predictable, due to its maturity.

On the negative side, sendmail has acquired a certain amount of cruft (old code) over its long history, with the result that it has a reputation of being insecure and bloated. Both charges are open to debate, however. It's true that it has had a number of significant vulnerabilities over the years. However, these have been brought to light and fixed rapidly.

As for the “bloatware” charge, it's true that sendmail has a much larger code base than other MTAs such as qmail and Postfix, and a larger RAM footprint too. This probably has at least as much to do with the fact that sendmail is monolithic (one executable provides most of sendmail's functionality) as it does with cruft. Indeed, sendmail's code has been scrutinized so closely by so many programmers over the years that it's a little hard to believe that too much cruft (i.e., code that is strictly historical and obsolete) has survived intact over the past 20 years.

It's much more useful to observe that being monolithic, sendmail must run as root if *any portion* of its required functionality does, i.e., writing mail to multiple users' home directories. For this reason, sendmail can run only as an unprivileged user on systems on which it's to act solely as an e-mail relay or gateway.

Sendmail also is criticized for its complexity. The syntax of its configuration file, sendmail.cf, is non-instinctive to say the least—in my opinion, its difficulty ranks somewhere between C programming and regular expressions. As with these, this is because of how powerful sendmail is (though many of us do wish sendmail used C, regular expressions or some other standard configuration language in sendmail.cf rather than its own equally complex but much more proprietary syntax). Nowadays, though, this point is largely moot. Recent versions of sendmail can be configured via m4 macros, which provide a much less user-hostile experience than editing sendmail.cf directly.

Mick's Disclaimer

Regardless of one's opinions on sendmail, it's unquestionably a powerful and well-supported piece of software. If sendmail's benefits are more compelling to you than the negatives, you're in good company. But you'll be in even *better* company if you learn to run sendmail securely.

Sendmail Architecture

As mentioned earlier, sendmail is monolithic in that it does all its real work with one executable, sendmail. Sendmail has two modes of operation: it can be invoked as needed, in which case it will process any queued mail and then quit; or it can be run as a persistent background dæmon.

Dæmon mode is required only when sendmail's role is to receive mail from external hosts; if all you use sendmail for is sending mail, you shouldn't run sendmail as a dæmon, and in fact you can probably stop reading now, because sendmail really doesn't need any customization to do this unless you wish to run it chroot-ed.

The way sendmail works, then, depends on how it's being run. If it's running as a dæmon (i.e., with the -bd flag), it listens for incoming SMTP connections on TCP port 25 and periodically tries to send any outbound messages in its queue directory /var/spool/mqueue. If it's being invoked on the fly, it attempts to deliver the outbound message it's been invoked to send and/or checks /var/spool/mqueue for other pending outbound messages.

The Task at Hand

Before we go any further, I should make clear what we're about to build. I've chosen the SMTP gateway scenario because it's such a common role for sendmail and because it's so dependent on good security (in most organizations, publicly accessible mail servers face scarier if not more numerous threats than internal mail servers do).

An SMTP gateway typically needs meticulous attention to privilege levels, file permissions and in general, only as much enabled functionality as is needed to route mail. On such a host, sendmail should run as an unprivileged user where possible; it should run chroot-ed (in a subset of /) at least when writing files, and it should be configured to relay mail only for your organization, not for spammers.

It takes very little tweaking to harden sendmail on Red Hat 7 for SMTP gateway use and only a little more on SuSE and other distributions.

Obtaining and Installing Sendmail

I can state with absolute certainty that your Linux distribution of choice includes one or more packages for sendmail. Whether it's presently installed on your system and whether it's an appropriate version for you to use, however, is another matter.

If you use an RPM-based distribution (Red Hat, Mandrake, SuSE, etc.), you can see whether you've got sendmail installed and which version by issuing the command **rpm -qv sendmail**. Note that Red Hat and its derivatives split sendmail into three packages: sendmail, sendmail-cf and sendmail-doc. SuSE uses a single package, sendmail.

So, what version should you run? As of this writing, the latest version of sendmail is 8.12.2. Red Hat 7 and SuSE 7, however, still support variants of sendmail version 8.11. As far as I can tell, there's nothing wrong with sticking with your distribution's supported sendmail package if it's version 8.11.0 or higher. There were no major security problems in the 8.10 or 8.11 releases; 8.11, in fact, was a “features” release: rather than being released to patch security holes, it was released because the sendmail team had added support for TLS encryption and for the SMTP AUTH extension to SMTP.

If you've got the time and/or inclination, though, it's never a bad idea to compile and install the latest stable version. For sanity's sake, I'll assume for the remainder of this article that you're using sendmail 8.10.0 or higher (unless otherwise noted).

Note to Debian Users

Debian GNU/Linux v2.2 (Potato) still supports sendmail v.8.9.3. Although this is a stable and apparently secure release, it's now two major versions old (if one considers the second numeral to represent a major version, which I do because the first numeral has been eight for half a decade). In addition, 8.9.3 doesn't support TLS or SMTP AUTH.

If you want TLS or SMTP AUTH, or are uncomfortable running an older version, you always can uninstall the package, download the latest source code tarball from www.sendmail.org and compile and install sendmail from source. The source code tarball is well documented and compiles easily under Linux, assuming you've got a working gcc installation.

Once you've installed sendmail, either in the form of a binary package from your distribution or from a source code tarball you've compiled yourself, you've still got a couple of tasks left before you can configure and run the sendmail executable as a dæmon.

SuSE Sendmail Preparation

If you're a SuSE user, become root if you aren't already. Next, open /etc/rc.config with your text editor of choice and set the variable SMTP to “yes”. This is necessary for sendmail's startup script in /etc/init.d to run at boot time.

In addition, you need to edit the file /etc/rc.config.d/sendmail.rc.config so that the variable SENDMAIL_TYPE is set to “no”. Doing so essentially will disable SuSEconfig's use of /etc/rc.config.d/sendmail.rc.config, which in other circumstances can be set up to generate a simple sendmail configuration automatically. We're going to set up an SMTP gateway/relay, which is a bit beyond the scope of sendmail.rc.config. But if your host is to act only as a

simple SMTP server for its own local users, it will probably suffice to edit this file (having first set its SENDMAIL_TYPE variable to “yes”); if so, you'll find sendmail.rc.config's full documentation in /etc/mail/README.

After editing rc.config and sendmail.rc.config, run SuSEconfig. This will propagate the changes you just made to rc.config and sendmail.rc.config. To start the dæmon you can enter the command **/etc/init.d/sendmail start**, but I recommend you wait until sendmail is fully configured before you do so.

Red Hat Sendmail Preparation

If you're a Red Hat user, you have only one step prior to configuring sendmail: edit /etc/sysconfig/sendmail so that the variable DAEMON is set to “yes”. This will tell the startup script /etc/init.d/sendmail to start sendmail as a dæmon at boot time.

Configuring Sendmail

And now, at last, we configure sendmail to act as our domain's SMTP gateway. What follows applies to any installation of sendmail 8.9 or higher (you shouldn't in any circumstances run 8.8).

The simplified method of generating sendmail configurations (sendmail.cf and accompanying files) consists of these steps:

1. Enable needed features in sendmail.mc.
2. Set up domain-name masquerading, if needed, in sendmail.mc.
3. Run **m4** to generate sendmail.cf from sendmail.mc.
4. Configure delivery rules by editing mailertable.
5. Configure relaying rules by editing access.
6. Configure local user-aliases in aliases.
7. Convert mailertable, access and aliases to databases.
8. Define all local hostnames in local-host-names.
9. (Re)start sendmail.

Notable Settings in sendmail.mc

The first and probably longest task in setting up an SMTP gateway is generating /etc/sendmail.cf. This is done most easily by editing /etc/mail/sendmail.mc (or on SuSE systems, /etc/mail/linux.mc—it may be different on other distributions).

Depending on which Linux distribution you use, configuration information for sendmail.mc can be found in /usr/share/doc/sendmail/README.cf (Red Hat and

its derivatives), /usr/share/sendmail/README (SuSE) or some variant thereof. I don't have enough space to describe the syntax of the many settings in this file in detail. We will, however, look at some that are useful for security and for modularizing our configuration a bit.

In addition to sendmail.cf itself, we can configure sendmail to read several other files for configuration information. This is useful for two reasons. First, editing sendmail.cf directly is unpleasant and even regenerating it from sendmail.mc isn't always desirable. Second, if your SMTP gateway has multiple administrators with varying privileges, you may wish to keep sendmail.mc and sendmail.cf locked down but allow other administrators to edit user aliases or mail delivery rules (i.e., /etc/mail/access and /etc/mail/mailertable, respectively).

The most useful external configuration files to enable are mailertable, which defines local mail-delivery rules; virtusertable, which defines virtual domain mappings on a per-user and per-domain level; and access, which defines which hosts may use the server as an SMTP relay.

The sendmail.mc directives for enabling these files are shown below:

```
FEATURE(`mailertable',`hash -o
        /etc/mail/mailertable.db')dnl
FEATURE(`virtusertable',`hash -o
        /etc/mail/virtusertable.db')dnl
FEATURE(`access_db',`hash -o
        /etc/mail/access.db')dnl
```

(Note that the mailertable and access_db features are enabled by default under Red Hat, but that virtusertable must be added manually.)

Each of these lines tells sendmail to reference the specified file (although the access database is called access, not access_db), to use a hash database and the path of the respective file. We'll see how to use these files shortly, but first we've got a few more things to attend to in sendmail.mc.

If your users' e-mail addresses are generic to your domain rather than specific to the hosts they log on to, for example, mick@polkatistas.org rather than mick@myron.polkatistas.org, you probably want the From: fields of their outbound e-mail to reflect this. (Receiving e-mail addressed to those generic names requires user aliases—see below.)

Following are some sendmail.mc lines that tell our example SMTP gateway to rewrite the From: fields of polkatistas.org's users in this manner. All the lines in the example below must be added (or uncommented):

```
MASQUERADE_AS(`polkatistas.org')dnl
MASQUERADE_DOMAIN(`.polkatistas.org')dnl
EXPOSED_USER(`root')dnl
```

```
FEATURE(`masquerade_entire_domain')dnl
FEATURE(`masquerade_envelope')dnl
```

The MASQUERADE_AS directive specifies the fully qualified domain name you wish to appear in applicable From: addresses. The MASQUERADE_DOMAIN directive specifies the hosts to which MASQUERADE_AS is applicable. Note that the "." preceding polkatistas.org indicates that all hostnames with this domain name are to be masqueraded.

EXPOSED_USER specifies a user name for whom the From: address should *not* be masqueraded. **root** is a popular candidate for this because e-mail from root often contains alerts and warnings; if you receive one, you generally want to know which host sent it.

The feature masquerade_entire_domain causes MASQUERADE_DOMAIN to be interpreted as an entire domain rather than a hostname; masquerade_envelope applies the masquerading not only to the SMTP header but to the envelope as well.

Four other directives, one logistical and the other three security-related, are shown in Listing 1. The always_add_domain feature is enabled by default under Red Hat and SuSE; use_cw_file and smrsh are enabled in Red Hat but not SuSE; confSAFE_FILE_ENV always must be defined manually.

Listing 1. Some More Sendmail Features

The always_add_domain feature simply forces the local host's domain name to be appended to any e-mail originating from a host that identifies itself without a domain. For example, if the SMTP gateway receives mail from the user "bobo" on a host identified only as "whoopeejohn", sendmail will rewrite the From: field to read bobo@whoopeejohn.polkatistas.org rather than bobo@whoopeejohn (but of course masquerading directives still apply).

The use_cw_file feature tells sendmail to refer to the file /etc/mail/local-host-names for a list of hostnames sendmail should consider local. The file /etc/mail/local-host-names is a text file containing hostnames listed one per line. Suppose our example SMTP gateway receives e-mail not only for the domain polkatistas.org, but also tubascoundrels.net. If our gateway's hostname is "mail", then its local-host-names file will look like this:

```
localhost.localdomain
mail.polkatistas.org
mail.tubascoundrels.net
```

The third feature enabled in Listing 1 is smrsh, the sendmail restricted shell. This is an important security control that restricts the commands that may be executed from a user's .forward file.

The fourth line in Listing 1 tells sendmail to set sendmail.cf's SafeFileEnvironment variable to, you guessed it, some subdirectory of / that sendmail will chroot to (sort of). Actually, this only will happen when sendmail writes files. If you think about it, though, this 50% chroot makes sense: file-writes are what we're probably most worried about, and creating this sort of chroot environment is a lot simpler than your typical chroot jail (which must contain copies of every file hierarchy, file, executable and device your chroot-ed program needs).

Listing 2 shows a recursive listing (**ls -lR**) of my example SafeFileEnvironment / var/mailjail.

Listing 2. Contents of /var/mailjail

Sendmail created the files /var/mailjail/var/spool/mqueue/bobo and .../root. Beforehand, I had created the entire chroot jail with only four commands:

```
mkdir -p /var/mailjail/var/spool/mail
/var/mailjail/var/spool/mqueue
cd /var/mailjail
chown -R mail:mail *
chmod -R 700 *
```

If you're concerned about unsolicited commercial e-mail, there's some good news. By default, sendmail doesn't allow SMTP relaying, a common technique of spammers. This can be disabled in sendmail.mc, but you won't find out how from me. Leave this alone. In addition, you can direct sendmail to reject mail from known sources of spam, per the Realtime Blackhole List (RBL), by adding or uncommenting this line:

```
FEATURE(`dnsbl')
```

For this to work, however, you need to subscribe to the RBL. See Resources for a link to its home page, where you'll find subscription and use instructions and some important disclaimers. (Note that while RBL subscriptions are free for “Individual/Hobby Sites”, there is a fee-schedule associated with this service.) Using the RBL can block e-mail from some legitimate users as well as from spammers, so proceed with caution.

If you run Red Hat 7.1 or 7.2, there's one more sendmail.mc parameter to check, this time one that needs to be commented out. If your /etc/mail/ sendmail.mc contains a line like this:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')
```

Then you need to comment it out by appending the string dnl to the beginning of the line. If active, this line will cause sendmail to accept only connections on

the loopback interface and not from external hosts. Needless to say, for an SMTP gateway this is undesirable (though it unquestionably enhances security).

Those are the most important sendmail.mc settings for our purposes. There are others relevant to security, especially for nongateway roles (local delivery, etc.). For more information see the README.cf or README file I alluded to at the beginning of this section.

To compile our macro-configuration file into sendmail.cf, we use this command:

```
m4 /etc/mail/sendmail.mc > /etc/sendmail.cf
```

If your macro-configuration file's name isn't sendmail.mc, substitute it with linux.mc or whatever your macro-configuration file is called. Sendmail expects its configuration file to be named sendmail.cf; however, it looks for it in /etc, so that part of the command is the same regardless of your distribution or even your version of sendmail.

Configuring Delivery Rules

That was the hard part. Now we only need to tell sendmail what to do with incoming mail, what local hostnames are legitimate and which users, networks and domains may use the SMTP gateway to relay mail not destined locally.

The mailertable is used to define delivery rules. It has a simple syntax, which is described in /usr/share/doc/sendmail/README.cf or /usr/share/sendmail/README depending on your distribution. In a nutshell, each line contains two parts: a destination identifier and an action. The destination identifier matches destination addresses or parts thereof; the action tells sendmail what to do with messages whose destinations match the identifier.

If the identifier begins with a ".", then all e-mail source addresses ending in whatever follows the dot will match. If it doesn't, then everything following the "@" sign must be identical to the identifier. The e-mail address bobo@weird-al.polkatistas.org won't match the identifier polkatistas.org but will match .polkatistas.org.

The action takes the form *agent*:*action* where *agent* is either a mailer (defined in sendmail.mc/linux.mc in MAILER() statements) or the built-in agents "local" or "error". The "local" agent, of course, means the mail should be delivered to a local user, specified after the colon (if nothing follows the colon, the user specified in the message itself will be used).

Below is a mailertable with two different actions:

```
polkatistas.org       smtp:internalmail.polkatistas.org
mail.polkatistas.org  local:postmaster
```

In addition to delivery rules, sendmail needs to know which e-mail destinations should be considered synonyms of the local (SMTP gateway's) hostname. These are specified in /etc/mail/local-host-names, one per line:

```
mail.polkatistas.org
weird-al.polkatistas.org
1.23.234.2
```

Finally, we need to define allowed relayers by editing /etc/mail/access. The syntax is simple. Each line contains a source name or address, paired with an action (again, see README.cf or its equivalent on your distribution for details). The action can be RELAY, REJECT, DISCARD, OK or ERROR. In practice, the most useful of these actions is RELAY. Because by default relaying is rejected, REJECT and DISCARD are only useful when defining exceptions to explicit RELAY rules.

Here is a simple access file:

```
localhost.localdomain           RELAY
localhost                       RELAY
127.0.0.1                       RELAY
192.168                         RELAY
```

Do you notice the absence of real hostnames in the example above? In this example, the SMTP gateway performs only outbound relays; inbound mail must be addressed to a local e-mail address, and outbound relays must originate from hosts whose IP addresses begin with the octets 192.168 (obviously a noninternet-routable network). I like this technique (using IP addresses) because then I can prevent IP address spoofing with my firewall rules, but I can't prevent forged From: addresses in e-mail (however, your needs may be different of course):

```
access
local-host-names
mailertable
```

More-Advanced Sendmail Security

SMTP AUTH (in sendmail version 8.10 and later) adds authentication to SMTP transactions, e.g., to determine whether to permit relaying. This is especially useful when systems or users don't run their own MTA but need to send mail, i.e., need to relay outbound mail through a central gateway.

If you're running an SMTP server that relays mail from other domains, you probably want this feature, as it's an important defense against unsolicited commercial e-mail, the perpetrators of which rely heavily on SMTP relays.

There's just one more file that may need tweaking: aliases. This file contains a map of e-mail aliases to user names. Usually an SMTP gateway doesn't need a very granular alias database; to translate entire domains' (or virtual domains') e-mail addresses you're better off using the user database (which, sadly, I don't have space to cover). It's fairly self-explanatory, though, so edit it if you need to.

Now three of the four files we've just discussed, mailertable, access and aliases, actually can't be used by sendmail directly; they must first be converted to databases. The /etc/mail file contains a handy Makefile for this purpose. To use it simply change your working directory to /etc/mail and enter this command:

```
Make access.db mailertable.db
```

Note that this won't work for aliases, which has its own utility, newaliases. Run newaliases without any flags to convert your changed /etc/aliases file into a new /etc/aliases.db file automatically.

And that's it for now. There's much I haven't covered, notably the smrsh shell (applicable mainly to local mail delivery, not to gateways). But hopefully I've given you some useful hints and pointers to more complete sources of information. Good luck!

Resources

Mick Bauer (mick@visi.com) is a network security consultant in the Twin Cities area. He's been a Linux devotee since 1995 and an OpenBSD zealot since 1997, and enjoys getting these cutting-edge OSes to run on obsolete junk.

Archive Index Issue Table of Contents

Advanced search

Linux Graphics Drivers

Robin Rowe

Issue #96, April 2002

Linux graphics performance is affected by your choice of graphics hardware, video driver and graphics interface. Let's take a look at the open-source XFree86 server, the commercial Xi server, ATI Fire GL workstation video card, NVIDIA GeForce3 Titanium game video card and the Wacom tablet Linux drivers.

XFree86 is the best known X server because almost every Linux distribution includes it. This technology provides the X Window System graphics underpinnings for most Linux desktop users. “The January 2002 XFree86 4.2 release included anti-aliased fonts, a lot of bug fixes and many new drivers including the RADEON 8500 and Matrox G550”, says XFree86 President and Release Manager David Dawes. Dawes is a board member and senior software architect at Tungsten Graphics, a startup founded by some former VA Linux employees to work on drivers, OpenGL API extensions and DRI.

ATI Fire GL 8700

Luxi scalable fonts, both TrueType and Type 1, are part of the 4.2 release. Bigelow & Holmes, Inc. donated these new fonts created in Ikarus digital

format, and URW++ Design and Development GmbH did the TrueType and Type 1 conversions.

XFree86 4.2.0 provides a partial implementation of the new X Rendering Extension. Using simple compositing operators provided by most hardware, Render can draw anti-aliased text and geometric objects as well as perform translucent image overlays. Still to be implemented are geometric primitives and affine transformation of images. Just three applications have been modified to provide anti-aliased text in 4.2.0: xterm, xditview and x11perf.

Several 4.2.0 enhancements affect Darwin Mac OS X. On Mac OS X, a new rootless mode was added to allow X clients to display windows on the Aqua desktop. XDarwin now supports Xinerama, the capability to have windows span two monitors.

Asked what is at the top of his wish list for the next version, Dawes says it will be making XFree86 easier to install. “We keep making progress on that. For 4.3, I'm planning to be working on that myself.”

Commercial X servers such as Metro-X and Xi Graphics offer alternatives to XFree86. HP and IBM also produce proprietary X servers, bundled with their workstations. Metro-X maker Metro Link has shifted to the personal video recorder TV set-top box market. “Linux Metro-X is no longer in development, but is still available”, says CEO Morgan Von Esson. “The $25,000 PVR SDK market is our focus now.” Metro Link is working with the ATI as a technology partner on the ATI set-top HDTV reference platform. Metro Link made a significant contribution to XFree86 4.x by donating its runtime loader. That enables XFree86 to load drivers dynamically, even on operating systems that do not support dynamic link libraries.

PNY Verto NVIDIA GeForce3 Ti200

Xi Graphics first released version 2.0 of its Accelerated-X Summit X server in November of 2001 and has been adding support for even more graphics cards since—now over 30 cards and laptops. The XiG Accelerated-X server comes in models for desktop (DX), laptop (LX), multihead (MX) and workstation (WX). “The

big appeal to our product is lifetime support, performance and stability", says CEO Dave Methvin. "We support the concept of freeware, and we contributed our Solaris laptop PCMCIA fixes as open source. However, XFree86 doesn't offer the testing and support we can with a commercial product." While still a student in Germany, XiG CTO Thomas Roell ported the original X Consortium implementation to the Intel x86 platform by himself and donated that to MIT (and thereby later XFree86).

Accelerated-X features include the Color Magic color management system, DualView for dual-head displays, Video Window for high performance YUV display, Stereo 3-D viewing for shutter glasses and Power Throttle to conserve power on laptops. Accelerated-X can throttle the CPU using APCI and even turn off components of the graphics hardware when not needed. Methvin says that can add two hours of battery life to laptops over XFree86.

For accelerated OpenGL, XiG utilizes a leaner API called XDA, not DRI. "DRI is way too big and complicated", says Methvin. "We've had some conflict with XFree86 about their design." Installing Accelerated-X presents some challenges to XFree86 users because the two systems don't like to coexist. Accelerated-X wrote their own optimized implementation of OpenGL and recommends uninstalling Mesa first to avoid installation conflicts. OpenGL applications developers running Accelerated-X should download the XiG OpenGL development kit to replace the XFree86-compatible Mesa developer package.

The Wacom Cintiq ($1,900) combines a 15" LCD screen and a wireless pen with 512 levels of pressure sensitivity. Linux-based animators may draw directly on screen.

XiG offers a free download of a time-limited evaluation copy. In fact, you have to download the evaluation copy in order to buy it. Until unlocked with a purchase key, Accelerated-X will only operate for 25 minutes at a time. (You may restart it as many times as you like.) DX models cost from $39 to $99, LX

from $69 to $139, MX from $129 to $249 and WX from $129 to $379. We installed the $99 DX RADEON Platinum edition.

The documentation was a little confusing mainly because we had questions about what to do with XFree86 and Mesa, which we already had installed.

Accelerated-X installation follows these steps:

1. Disable graphical logins and exit the currently running X server.
2. **su root**.
3. Remove or disable Mesa.
4. Remove or disable agpgart kernel module.
5. Install the Linux kernel source.
6. Install the X services module (XSVC) RPM.
7. **make xsvc**.
8. **make xsvctest**.
9. If necessary, make the kernel with appropriate MTRR, AGPGART and SMP settings.
10. Install the Summit RPM.
11. Run **Xsetup** to configure the graphics board, monitor and mouse in text mode.
12. Run **startx**.
13. Run **Xsetup** again to configure advanced options in graphics mode.

We experienced a few extra hurdles because we were using Debian, not Red Hat. We converted the RPM files to deb format using **alien**. Our high-performance ASUS A7A266 motherboard was unsupported, as xsvctext revealed, meaning we didn't gain AGP acceleration. Probably because of the converted deb files, we had to add the xsvc modutils alias manually and create .xinitrc and .xserverrc files.

```
# vi /etc/modutils/aliases
.
.
alias char-major-10-175 agpgart
alias char-major-10-179 xsvc
.
.
tbird:/etc/modutils# update-modules
```

We had to kill gpm and select MS IntelliMouse in Xsetup to get our optical Logitech MouseMan Wheel mouse to work.

XiG plans to continue to grow Accelerated-X as its primary market. “We're moving into the high-end workstation cards”, says Methvin. “We're really impressed with the power of the new high-performance cards. We just released

a driver for 3DLabs Wildcat II 5110.” XiG OpenGL support is being upgraded from version 1.2.1 to 1.3.

XiG doesn't support NVIDIA cards. “We'd like to support NVIDIA cards”, says Methvin. “But, NVIDIA won't provide the specs. We have to content ourselves with writing faster drivers for competitor ATI.” Methvin's biggest complaint with the X Window System is the number of desktop configurations to be supported. “GNOME is tied to XFree86 in ways that tend to break other X servers, and there are just too many desktop window managers. I think not having one standard desktop is holding back application developers like Adobe from supporting Linux.”

Graphics card manufacturer ATI acquired Fire GL Graphics and their line of workstation graphics cards in April 2001. Workstation cards such as the Fire GL are intended for 3-D animators and engineers, not consumers. In other words you might play *Quake* on a high-performance game card such as the ATI RADEON, but a game designer would use an even higher-performance Fire GL when creating the 3-D world for a game or for special effects in motion pictures. What differentiates these cards most is the OpenGL acceleration performance. Other pro-Fire GL advantages include high frame rates in multiple windows, double buffer overlay, dual-monitor support, plus greater stability, reliability and technical support.

ATI kept intact the German Fire GL Linux driver team. The Fire GL driver is a closed-source driver and probably always will be. “There are intellectual property and competition issues in taking a closed driver to open source”, says Director of Workstations Ed Huang. “We believe we have the fastest OpenGL implementation for Linux and don't want to show competitors how.”

XFree86 and MacGimp with a Mac OS X Finder window open showing the two different window systems operating together.

The Fire GL 2 and Fire GL 4 use the same closed-source driver. ATI has taken a different strategy for the RADEON. It is an open-source driver. The top-end Fire GL 4 costs about $1,500 (at buy.com) and the less-expensive Fire GL costs about $725 (at cdw.com). Two new Fire GL models are replacing these. The Fire GL 8800 will be the first ATI-developed Fire GL card since the acquisition, and the first based on the RADEON chipset. This 128MB card, with about 50% better performance than the Fire GL 4 in most applications, is expected to cost less than $900. The Fire GL 8700, based on the RADEON 8800LE chip with 64MB, replaces the Fire GL 2. It has about 50% better performance than the GL 2 and should cost less than $400.

All the Fire GL cards use the same Linux driver. The Fire GL driver (currently for XFree86 4.1.0 with libc 6.2) is available on the ATI web site in .tgz or RPM format, about 5.5MB.

Graphics chip manufacturer NVIDIA's new high-performance game card, the GeForce3 Titanium, is offered in Compaq, Dell, HP and IBM PCs. Their graphics cards are also standard in all Apple desktops and in the Microsoft XBox. NVIDIA has had a good year, with its stock the best performer in the 2001 S&P 500. The GeForce3 models include the Ti 200 (about $160), and the even higher-performance Ti 500 (about $300). These feature fast high-resolution anti-aliasing on the GPU (HRAA Quincunx), DVI out and TV s-video out. For content

creators, NVIDIA offers the even higher-performance Quadro2-Pro card (about $615).

For Linux users perhaps the most interesting aspect of the NVIDIA card is its unified driver architecture. “We use the same unified driver code for Linux and Windows for all cards”, says software VP Dwight Diercks. “Our Linux driver is ready at the launch of a new board just like our Windows driver. We release both Windows and Linux drivers within 60 days of new graphics chips being manufactured.”

OpenGL 1.3 is fully implemented and shipping on all platforms. “Our unified driver architecture gives us a single driver binary across all our cards”, says director of OpenGL and Linux workstations Nick Triantos. “When we make a driver performance optimization it applies to all our products. Other companies supporting multiple drivers can't keep them current.”

Diercks says NVIDIA reuses as much driver code as they can. “The major internal difference is the OpenGL GLX layer for Linux and the Wiggle layer for Windows. All the optimizations, new features and extensions come to the Linux side without much work.” Nvdriver is a Linux kernel mode driver. The Windows driver is a VXD miniport driver.

The Linux Nvdriver is a closed-source driver. There is also an open-source NVIDIA driver offered as part of XFree86. The open-source driver has the important 2-D functionality (accelerated video, DVD playback and monitor detection), but OpenGL 3-D acceleration was lost when the XFree86 architecture changed in 4.x.

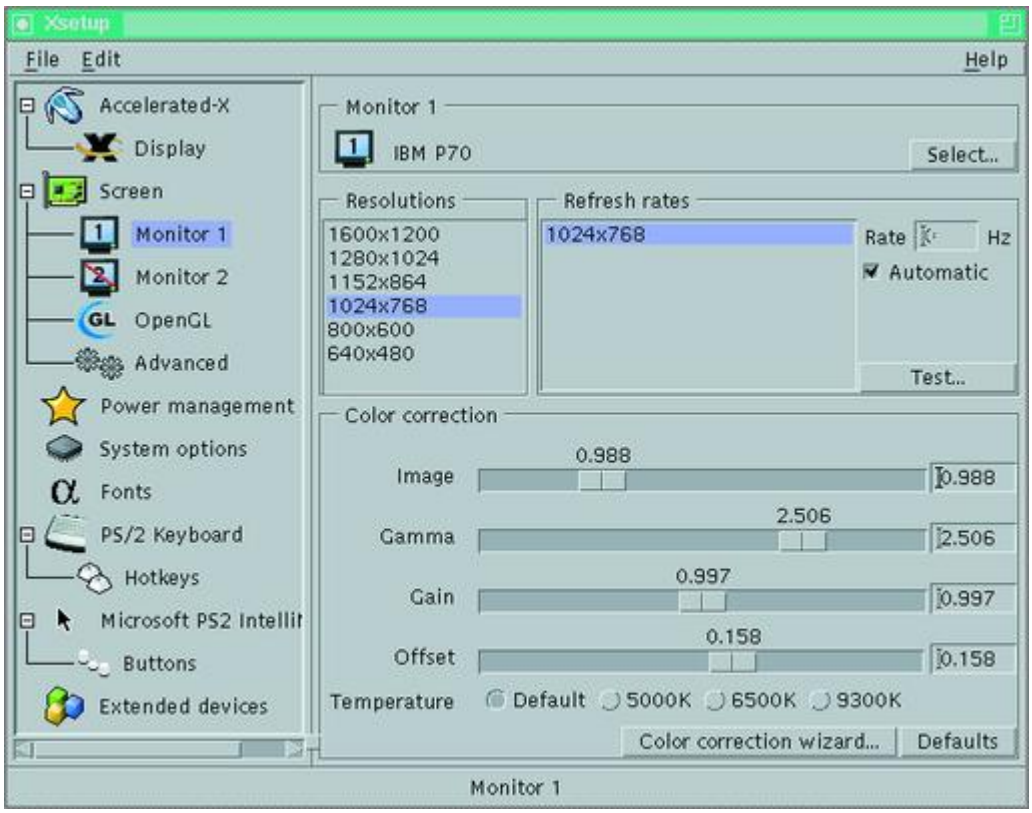
Led by architect Mark Voikavich, NVIDIA has a team of ten engineers supporting Linux. Although Microsoft's proprietary DirectX protocol obviously isn't part of the Linux driver, all the other driver features are in Linux. “Linux has full TwinView dual-head support”, says Diercks. “Linux laptops have multihead technology for monitor output for presentations. The user can drag windows between screens.” The Linux driver is actually a little leaner than the Windows driver because it doesn't have to duplicate functionality offered in OpenGL and DirectX. Depending upon configuration, the Linux driver can have faster performance than the Windows version.

NVIDIA works closely with the XFree86 Project. “In the future we're looking at more support for laptop features, such as hotkeys to move windows dynamically between displays”, says Triantos. “There's no interface in X to make a window appear on just one screen.” Another feature missing in X is a mechanism to do screen captures of accelerated windows. In Windows there is a callback hook to the driver, but to capture an accelerated screen in Linux

requires help from the application, such as pressing the F11 key in *Quake* to do an OpenGL glReadPixels screen capture. Because accelerated windows blow a hole in the desktop to write to, it takes extra effort to capture them instead of a solid black window.

NVIDIA recently released their Personal Cinema graphics card to compete with the ATI AIW RADEON. “Personal Cinema provides video editing and video capture, but those features aren't available in the Linux driver”, says Diercks. “We're interested, but not sure there is enough of a demand there for Linux.” The ATI AIW RADEON, on the other hand, has a new Video4Linux driver for video capture.

NVIDIA uses Linux extensively in their chip development. “We use Red Hat 6.2 Linux on a 1,500 server computing farm”, says Diercks. “Our engineers test the progress of our chip designs with that.”



The XiG Xsetup configuration tool enables quick adjustment of color temperature using the Color Correction Wizard.

Not all graphics-related Linux drivers are for video. Wacom tablets are popular with GIMP users and with motion picture animators working on Linux workstations. XFree86 developer Frederic Lepied is responsible for the Wacom Linux drivers. “In my previous job, we needed to access the Solaris X server through a tablet, and the protocol for the Wacom IV devices was available from Wacom's web site”, says Lepied. “I later ported my Solaris Wacom driver to

Linux XFree86. Making the XInput extension work in XFree86 is how I became an XFree86 developer.”

Linux Wacom driver development started in 1995. Lepied is the only developer, but he receives contributions from many people. The development is a continuous effort to support the new models. Lepied's work is supported by Wacom, but sponsored by his employer MandrakeSoft. Lepied is a team manager and developer of the Mandrake Linux distribution. The open-source Linux Wacom driver supports Wacom IV, Wacom V and USB protocols.

Resources

email: Robin.Rowe@MovieEditor.com

Robin Rowe (robin.rowe@movieeditor.com) is a partner in MovieEditor.com, a technology company that creates internet and broadcast video applications. He has written for *Dr. Dobb's Journal*, the *C++ Report*, the *C/C++ Users Journal* and *Data Based Advisor*.

Archive Index Issue Table of Contents

Advanced search

On-Line Privacy

Lawrence Rosen

Issue #96, April 2002

A privacy policy is a statement made by a web site owner that it will use your private information only for certain stated purposes. You are expected to review the privacy policies of the web sites you visit and to avoid those web sites that won't safeguard your private information according to your preferences.

Failure by web site owners to comply with their own published privacy policy may be actionable under the law, for example, as negligence or fraud. Gross recklessness or intentional misrepresentation regarding privacy promises also may result in a web site owner having to pay substantial punitive damages.

Most of us ignore on-line privacy, though. In our interactions on the Internet, for example, we no longer even bother to read the “Privacy Policy” statement that is a link on almost every page. We either assume that data about us is not being collected and disseminated by the web site owner without our express approval, or we no longer care that our private information is being shared.

As for me, I had concluded that the battle for my privacy was lost because I didn't have the energy any more to do what it takes to secure it. I stopped reading privacy policies. I even ignored the notices from my banks giving me the option to prevent the sharing of private financial data they held about me. (I bet the vast majority of readers of this article are just like me in this regard!) There is so much data gathering and sharing going on that protecting privacy seems to be impossible to worry about.

Then a friend of mine brought the P3P standard to my attention. Promulgated by the World Wide Web Consortium (W3C), the “Platform for Privacy Policy” standard empowers users to control their on-line privacy in a simple and effective way.

Danny Weitzner, the technology and society domain leader of W3C and the chairman of the P3P committee, described the new standard this way in his testimony before the United States Senate Committee on Commerce, Science and Transportation:

> W3C and its members became concerned about privacy on the Web because people won't use the Web to its full potential if they have to face such uncertainty. The majority of users are perfectly willing to share some information on the Web. At the same time, basic human dignity demands that we have meaningful control over which information we chose to expose to the public. Our goal is to include in the basic infrastructure of the Web the building blocks of tools that can provide each user this basic control.

All you have to do for P3P to work is to instruct your browser to check whether web sites you visit support the P3P standard. You can elect to avoid those that do not support the standard, or you simply can be more vigilant about sharing your personal information with such web sites.

Your browser automatically retrieves, from P3P-enabled web sites, machine-readable XML information that encapsulates the web site's privacy policy. Thus your browser can determine whether the web site owner promises to safeguard your private information or whether it shares your information with others.

You can set your browser to refuse to visit, or you can refuse to share data with, web sites that don't satisfy your privacy preferences.

You no longer will have to read lengthy (and boring) privacy policies on each web site you visit. Instead, software built into your web browser, plugins or other tools can enforce your privacy rights automatically and effectively by exchanging XML data with the web site before you even get there.

Many of the major proprietary software companies, including Microsoft of course, participated in the W3C P3P committee. The resulting standard also has been supported by consumer-focused organizations, including the Electronic Frontier Foundation.

Our privacy rights have become so fundamental to us that they usually are taken for granted. But privacy must be hard won through diligence. The software tools we create have the potential to help us secure our privacy rights —and the P3P standard is one kind of software tool that does just that.

Legal advice must be provided in the course of an attorney-client relationship specifically with reference to all the facts of a particular situation and the law of

your jurisdiction. Even though an attorney wrote this article, the information in this article must not be relied upon as a substitute for obtaining specific legal advice from a licensed attorney.

email: lrosen@rosenlaw.com

Lawrence Rosen is an attorney in private practice in Redwood City, California (www.rosenlaw.com). He is also executive director and general counsel for Open Source Initiative, which manages and promotes the Open Source Definition (www.opensource.org).

Archive Index Issue Table of Contents

Advanced search

Defining Interoperability

David A. Bandel

Issue #96, April 2002

Accounting applications, typing tutors for children, an FTP client and more.

For me, the term interoperability means how to bend 106-plus UNIX-type operating systems to work with Microsoft's proprietary, non-RFC-compliant, all-too-often frustrating systems. All the UNIX-type operating systems work together most seamlessly, thanks to their adherence to standards. But Microsoft's marketing strategy of “100% Microsoft” and their deliberate barriers to interoperability cause no end of anguish to system administrators, many of whom are of the opinion that “0% Microsoft” would be the best goal for sanity's sake. Meanwhile, Microsoft's hidden APIs are more hindrance than the moving target that is Linux. Thankfully, teams of programmers, like those working on Samba, have delivered us this interoperability successfully despite Microsoft's best efforts. But complete interoperability between Microsoft and Linux has a long way to go.

Freemoney www.freemoney.org

If your business is point of sale, with emphasis on sales of a large number of varied products, you might want to take a look at Freemoney. To preview this application, go to their home page and their preview page. While this accounting package is feature-rich, whether for web sales or in-store sales, setup needs a *lot* of work and better documentation. You won't be able to skip through this installation, but once installed, you will be pleased with its ease of use. Requires: interchange, web server, Perl, Perl modules: Bundle::Interchange, DBD::Pg, DBI & Schedule::At, PostgreSQL.

phpPgAdmin phppgadmin.sourceforge.net

If you know phpMyAdmin, then you know phpPgAdmin. This web-based PostgreSQL administration is a port of the very popular phpMyAdmin. It makes administration of Postgres a breeze. Installation is simple and straightforward,

and you can set up the system to administer all the databases or just one. Requires: web server with PHP and PgSQL support, web browser, PostgreSQL.

TuxTyping www.geekcomix.com/dm/tuxtype

This program for children is a typing tutor. It features falling fish with letters that Tux the Penguin must run beneath and eat as they fall. For more advanced practice, you can substitute words (generally three-letter words) to accomplish the same thing. The graphics and game play are excellent diversions for children while they practice touch typing. Requires: libSDL, libSDL_image, libSDL_mixer, libm, libdl, libartsc, libpthread, libX11, libjpeg, libpng, libz, libtiff, libvorbisfile, libvorbis, libogg, libsmpeg, glibc.

di www.gentoo.com/di

The di utility, standing for disk information, will provide you with quite a bit of information about your hard disk. Most of the information mirrors what you get from the df utility, but not all. Lately, with the addition of ReiserFS and ext3 to the kernel (not to mention LVM), often I need to know what type of filesystem I'm dealing with. This information isn't provided by df, but di shows you this by default. The output format is also cleaner than df, especially on a system with the horribly long devfs naming scheme. Requires: glibc.

GnuLedger webaccountant.sourceforge.net

One of the biggest complaints I hear from folks who'd like to convert 100% to Linux is the lack of a personal financial system. Several good applications exist for business accounting, but few for personal accounting. I can think of three, but they all use a flat file database and only can be run locally. GnuLedger runs from a web browser, so it could be run from anywhere in the world (ideally using https). And because the data is held in MySQL, it's fast and can be queried easily without using GnuLedger if you know basic SQL. My only note about GnuLedger is that it (at least the version I downloaded and tested) had no accounting tables at all; they had to be created from scratch—not something your average home owner wants to deal with. A comprehensive set of tables properly linked for the average person would go a long way in helping GnuLedger along. Anyone want to donate a schema they're using? Requires: MySQL, web server, Perl, Perl modules: Number::Format, DBI, DBD::mysql, Math::Currency, Math::FixedPrecision.

hardmon www.fcoutant.freesurf.fr/hardmon.html

If you have a motherboard with supported sensors (mostly ASUS motherboards, but others are being added slowly), you can monitor your

system's health (CPU temperature, etc.) graphically. Living in Panama, and particularly with systems in non-air-conditioned locations in the summer, this utility shows at a glance if the temperature is getting too high and a build or two needs to be aborted. Works on multiprocessor systems as well as single CPU systems. Requires: lm_sensors, libXpm, libX11, libm, glibc.

gFTP gftp.seul.org

This month's choice from three years ago was easy. This is the Burger King of FTP clients. It boasts both a text and graphical mode, and it can use FTP (passive or active), SSH, SFTP or HTTP for file transfers. It is threaded, so it allows for multiple simultaneous downloads, can queue downloads, transfer between two remote systems and much more. If you need to move files around a network, you need to take a look at this application. Requires: gftp-text: libglib, libnsl, glibc; gftp-gtk: libgtk, libgdk, libgmodule, libglib, libdl, libXext, libX11, libm, libpthread, glibc.

Until next month.

David A. Bandel (david@pananix.com) is a Linux/UNIX consultant currently living in the Republic of Panama. He is coauthor of Que Special Edition: Using Caldera OpenLinux.

Archive Index Issue Table of Contents

Advanced search

An Interview with Preemptible Kernel Patch Maintainer Rober

Rick Lehrbaum

Issue #96, April 2002

In this interview, *Linux Journal* chats with Robert Love, the principal maintainer of an increasingly popular kernel-preemption patch that improves the real-time responsiveness of the Linux kernel. Robert describes his role in the project, explains why the preemption enhancement is important to a broad range of Linux applications beyond just embedded/real-time (including end-users' desktops) and shares his vision of the future of Linux in the embedded and desktop markets.

Robert Love, Principal Maintainer of the Preemptible Kernel Patch

Rick You have been becoming well known as the maintainer of a preemptible kernel patch for the Linux kernel. How did that situation come about?

Robert I am interested in innovative ideas, and the preemptible kernel is a very neat concept. The idea was first talked about during 2.3 (we finally had a kernel that had fine-grained enough locking that it was doable)--Linus actually outlined

the first design. MontaVista released a patch in this time frame and worked on it through the early days of 2.4.

About this time I tried the patch, rediffed it to the newest kernel and starting making changes. I eventually posted them. People started trying it, and use of the patch took off. I've been working hard since.

Rick What is different between what you have done and what was (or is) being done by MontaVista Software with their preemptible kernel patch? Are you now collaborating with them, and if so, how closely?

Robert The patch I am maintaining now is MontaVista's. They still put time into the patch, and I work closely with a couple of their engineers. Obviously it has evolved a lot since I got involved, but it's still their work. I believe they are using the latest release in their product.

Rick As I understand it, your interest in the preemptible kernel is for reasons that would tend to be of interest to a broader range of Linux applications such as streaming of audio/video, etc., as opposed to the embedded/real-time system focus of MontaVista. Is this accurate? Can you give some examples of the sorts of applications that your patch can noticeably improve the performance of?

Robert This is correct. I see a preemptible kernel as a means to an overall better system. Besides the traditional markets for low latency (audio/video, specialized embedded/real-time, etc.), a preemptive kernel can benefit any interactive task. The result is hopefully a smoother, more responsive desktop.

Rick With the preemptible kernel patch added, does Linux outperform what the latest versions of MS Windows can offer to desktop, enterprise and embedded users?

Robert I've got no idea how we perform compared to Windows. I'd certainly say even earlier versions of Linux were superior to Windows 9x. How we compare to Windows NT (i.e., 2000 and XP) for desktop performance is hard to measure.

The latency of the kernel certainly can be measured—scheduling latency, jitter, etc., are quantitative—but the quality of the desktop only begins with the kernel. That means X and everything on top of it factors in, too.

Rick When Red Hat announced that they were going to use RTLinux to provide real-time capabilities to their customers, both Mike Tiemann (CTO of Red Hat) and Alan Cox went on record to say that real time doesn't belong in the kernel. Would you care to comment on that?

Robert Red Hat has an agenda here, and I tend to disregard any technical comments when there is bias. There are benefits to the hard real-time interrupt-driven approach, I don't deny, but that solution is not Linux. I think Linux can become a contender in the embedded/real-time market without giving up on itself, while still being a UNIX and having the standard Linux API. In fact, I think a lot of the technologies that achieve this could live right in the official kernel. Kernel preemption is one such innovation, and it's an innovation that does not benefit solely real-time applications.

Rick Have you seen any indication from Linus or other key kernel maintainers that your patch (or the MontaVista patch) is being seriously considered for the 2.5 kernel? What do you think the chances are that it will get incorporated into 2.5 or 2.6?

Robert Linus said at ALS [Annual Linux Showcase] this year [2001] that he was interested in the preemptible kernel patch. That doesn't mean anything to me until we are in, but it is a good sign. There is opposition. There are various issues that need to be dealt with. I believe it is a sane move for 2.5. The patch has seen a lot of testing, and we have a lot of users. I do not want to predict whether it will be merged for 2.5. Time will tell. [Subsequent to this interview, the announcement was made that Linus has merged the patch into the main Linux development-kernel tree, beginning with version 2.5.4-pre6.]

Rick Please summarize the advantages in general, not just for embedded real-time applications, of having the preemptible kernel enhancement included in the kernel. What about any disadvantages?

Robert I'll start with a quick explanation of how the patch works. Right now, the kernel is not preemptible. This means that code running in the kernel runs until completion, which is the source of our latency. Although kernel code is well written and regulated, the net result is that we effectively have an unbounded limit on how long we spend in the kernel. Time spent in kernel mode can grow to many hundreds of milliseconds. With some tasks demanding sub-5ms latencies, this non-preemptibility is a problem.

The preemptible kernel patch changes all this. It makes the kernel preemptible, just like user space. If a higher-priority task becomes runnable, the preempt patch will allow it to run. Wherever it is. We can preempt anywhere, subject to SMP- [symmetric multiprocessing] locking constraints. That is, we use spin locks as markers for regions of preemptibility. Of course, on UP [uni-processing] they aren't actually spin locks, just markers.

The improvement to response is clear: a high-priority task can run as soon as it needs to. This is a requisite of real-time computing, where you need your real-

time task to run the moment it becomes runnable. But the same effect applies to normal interactive tasks: as soon as an event occurs (such as the user clicking the mouse) that marks it runnable, it can run (subject to the non-preemptible regions, of course).

There are some counter arguments. The first is that the preemptible kernel lowers throughput because it introduces complexity. Testing has showed, however, that it improves throughput in nearly all situations. My hypothesis is that the same quicker response to events that helps interactivity helps throughput. When I/O data becomes available and a task can be removed from a wait queue and continue doing I/O, the preemptible kernel allows it to happen immediately—as soon as the interrupt that set need_resched returns, in fact. This means better multitasking.

There are other issues, too. We have to take care of per-CPU variables, now. In an SMP kernel, per-CPU variables are implicitly locked—they don't have explicit locks, but because they are unique to each CPU, a task on another CPU can't touch them. Preemption makes it an issue because a preempted task can trample on the variables without locks.

Overall, I think the issues can be addressed, and we can have a preemptible kernel as a proper solution to latency in the kernel.

Rick From your brief explanation of how the preemptible kernel does its business, I get the impression that nothing new needs be done on a user's system other than installing the modified kernel in order to gain benefits. Specifically, if I were to install the patched kernel on my desktop system, would I immediately begin noticing performance improvements during my work (and play) on the system, such as doing work during downloads or burning CDs while listening to streaming audio or watching streaming video? Or, do I need to have applications that can take advantage of the kernel's preemption capability? Would it be necessary to alter the priority of the apps I run, or make any other adjustments?

Robert You don't have to do anything. The preemptible kernel patch requires no change in user applications, environment or API. The change (from switching to a preemptible kernel) should be immediate. Your normal programs and your normal benchmarks will show latency improvements. Renicing your programs would help, just because they would be able to reap the benefits of preemption more readily (anything they need to do could cause a preemption).

This is why I see elegance in this solution; Linux can be made more RT without abandoning itself.

Rick Please describe yourself, what you do and what your interests are. Also, what is your background relative to Linux and your philosophy with respect to open-source software?

Robert I am a student at the University of Florida studying mathematics and computer science, originally from South Florida. I first used Linux six or seven years ago with an early Slackware release. I've run nothing but Linux for the past couple years.

Outside of computing I'm really into rock music, good food and the military-industrial complex.

My philosophy with respect to open source is that of a pragmatist. While I enjoy the freedom of the code and certainly appreciate the rights gained from free software, I see the biggest benefit in merely having open code. I can see it; I can fix it; I can play with it. With enough momentum, open source creates amazing software. I am very proud of our kernel.

Rick Approximately how much time per week do you spend working on your kernel patch for Linux?

Robert My girlfriend would probably say too much. Anywhere from a couple hours a week to many hours a day.

Rick What is your vision of the future of Linux in the embedded market and in the home and business desktop computer market?

Robert Linux is going far in the embedded market. I'm sure you as much as anyone can see the vast potential there. Linux offers key benefits—particularly available source and a highly configurable system—that should appeal to the embedded space. It is light, so it embeds well. It has a simple API. A lot of user-space solutions exist, and it is easy to create entirely new ones.

I certainly think Linux is an excellent desktop system (both GNOME and KDE are quite mature), but I don't see huge strides here in the coming year. Market share will certainly increase, perhaps significantly, but we will not be overtaking any evil monopolies this year.

Rick Is there anything else you'd like to add?

Robert The latest version of all this fun stuff is available for FTP download at ftp.kernel.org/pub/linux/kernel/people/rml or at your favorite mirror.

Rick Thanks very much!

Editor's note: Look for an article by Robert on his preemptive work in next month's issue.

Rick Lehrbaum (rick@linuxdevices.com) created the LinuxDevices.com and DesktopLinux.com web sites. Rick has worked in the field of embedded systems since 1979. He cofounded Ampro Computers, founded the PC/104 Consortium and was instrumental in creating and launching the Embedded Linux Consortium.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

The CodeWeavers CrossOver Plugin

Dave Phillips

Issue #96, April 2002

Surfing multimedia web sites under Linux has been a somewhat frustrating experience. We do have support for RealPlayer and Flash, and we have some other fine streaming-media clients such as XMMS, but for various reasons we have been barred from watching most QuickTime movies or any Shockwave Director animations. Barred until now, that is; thanks to the programmers at CodeWeavers now we can enjoy those same movies and animations using their native Windows players running on web browsers under Linux.

On August 27, 2001, CodeWeavers announced the availability of their CrossOver Plugin in its 1.0 release. This software provides a translation layer that lets you download and install native Windows plugins as though they were being installed to a real Windows partition, making it possible for Linux web browsers to run multimedia content created for Apple's QuickTime 5 and Macromedia Shockwave. The 1.0 package also supports Microsoft's WordView and XLView browser plugins for viewing content created in Word 97 and Excel 97, and CodeWeavers has announced their intention to support all Windows browser plugins via CrossOver eventually.

Getting It and Installing It

The CrossOver Plugin is available from the CodeWeavers web site. You can purchase and download the plugin by itself, or you can buy the CodeWeavers Wine CD. The disk includes the plugin and printable documentation along with CodeWeavers Wine 1.0. Note, however, that the CrossOver Plugin is a standalone program that requires neither Wine nor Windows.

Installation is simple and well documented, and a series of helpful wizards will guide you through the process. After displaying the CodeWeavers license agreement, the installer sets up CrossOver itself (Figure 1). This part of the installation requires little intervention from the user beyond specifying the target directory ($HOME/crossover by default). CrossOver first installs a

fake_windows and a set of DLLs (derived from CodeWeavers' work on the Wine Project) needed by the players and viewers. The package does not provide the supported players and viewers: once CrossOver has installed itself, you will see the Plugin Setup window (Figure 2) where you actually connect to the Apple, Macromedia and Microsoft download areas to retrieve the plugins. From this point you follow the course of downloading, installing and setting up the plugins exactly as though you were working in Windows itself. You simply can accept the default values throughout the installation and configuration for each plugin. Remember, CrossOver deceives the native Windows plugins into believing that they are installing themselves in a real Windows environment, so the default installation paths are the same as those expected in Windows.

Figure 1. CrossOver Begins Installing Itself

Figure 2. The Plugin Setup Window

When you have finished setting up the plugins you want CrossOver to manage, restart your browser and head over to Apple's QuickTime movie trailers site and catch the latest previews (Figure 3), or check out Director Web for some impressive Shockwave sites (Figure 4). I spent an evening wandering through sites from a Google search for QuickTime movies and Shockwave pages, testing the plugin's ability to play the various file versions and formats. I'm happy to report that the plugin had no trouble playing anything the Web threw at it.

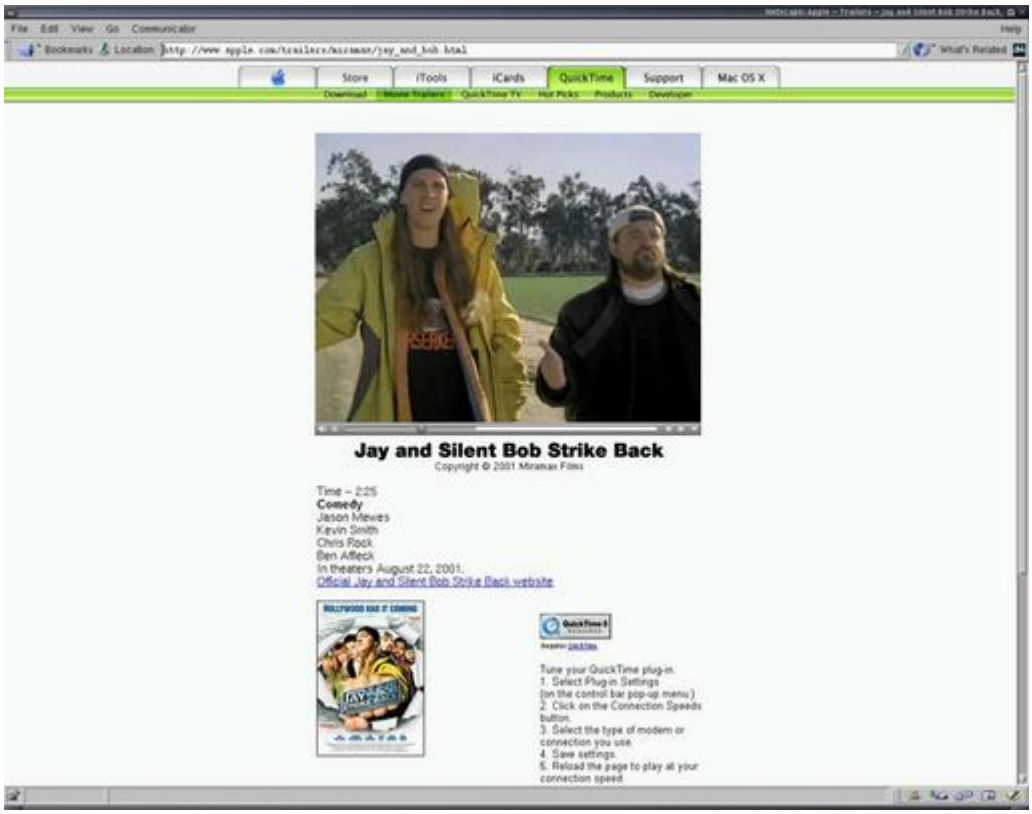


Figure 3. A QuickTime Movie Trailer in Linux Netscape

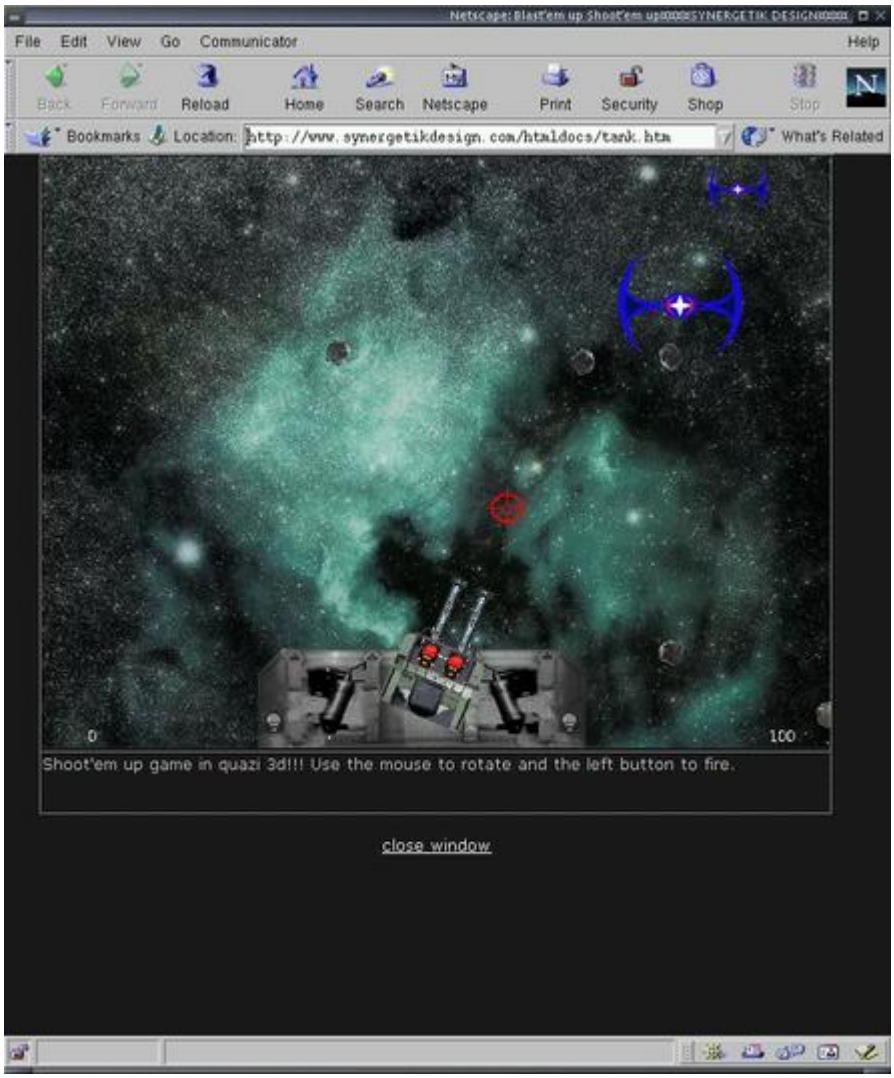


Figure 4. An Interactive Shockwave Game Running in Netscape

You also can choose to set up QuickTime as your default player and viewer for other media file types, including AVI and MPEG movies or JPEG and GIF images. The CrossOver documentation notes that you must disable your browser's existing file associations and manually reconfigure them for QuickTime.

The Plugin Setup window includes tabs for configuring your preferences in Netscape, Mozilla and Konqueror, the browsers currently supported by CrossOver. Netscape is the default browser, but the CrossOver package includes advice and instructions regarding setup for the other browsers. Incidentally, you don't have to install all your plugins at once. Running $HOME/crossover/bin/pluginsetup will bring up the Plugin Setup window for later addition or removal of CrossOver's supported plugins.

The only problem I had during the installation was resolved quickly with the help of the CodeWeavers team. My $HOME/.netscape/plugins directory was read-only, and the CrossOver Plugin could not install itself until I changed the directory's permissions to read/write for normal user. With that correction the installation proceeded smoothly.

Performance

The test system included a Sound Blaster Live! Value sound card driven by the ALSA 0.5.11 driver package and a Voodoo3 video card managed by XFree86 4.01, all running happily under a Linux 2.4.5 kernel patched for low latency. The internet hookup was a 768kbps DSL connection, and I used Netscape 4.76 for my browser. I was unable to compare performance with the same plugins employed in Windows itself, but I was quite satisfied with them running in Linux. Bear in mind that CrossOver is not an emulator; those are real native Windows plugins that believe they are working in Windows, so there should be little or no performance penalty usually associated with emulation environments.

Streaming QuickTime performance was generally excellent, though of course the reception speed depended on the transmission rate at the sending end. Video and audio synchronize well, and their quality is superb; my friends literally jumped back when they heard the sound from some of the movie trailers we watched, and they agreed that the streaming video in QuickTime was the best they've seen in a Linux browser.

Some Shockwave pages were problematic with fonts, and it seemed that any 3-D-enabled site failed to run. Fortunately, I can say that most Shockwave sites ran beautifully, including some very interesting interactive music and sound pages I found in the listings at Director Web. The CodeWeavers team is aware of the 3-D problem and may have a fix by the time you read this article.

I must use Word 97 occasionally, so I installed Microsoft's WordViewer plugin and clicked on some .doc files in my Windows Word directory (Figure 5). My files included not only standard text but also various notations and indicators added by the DOT template used by my editors. As you can see, apostrophes have been replaced by those boxes, but that was the only visual fault I found in the display.

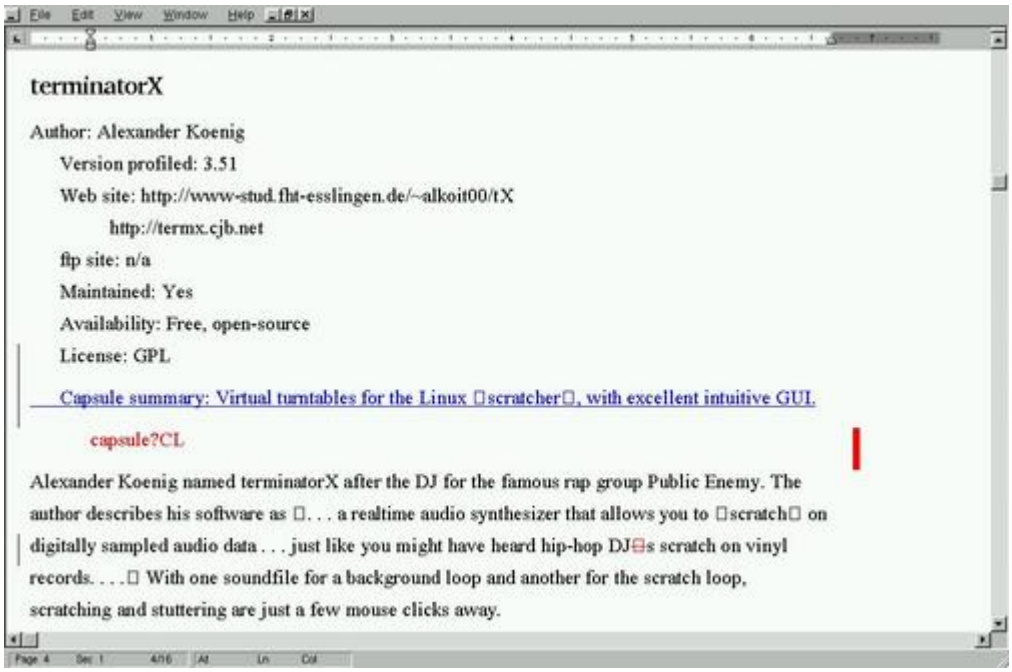
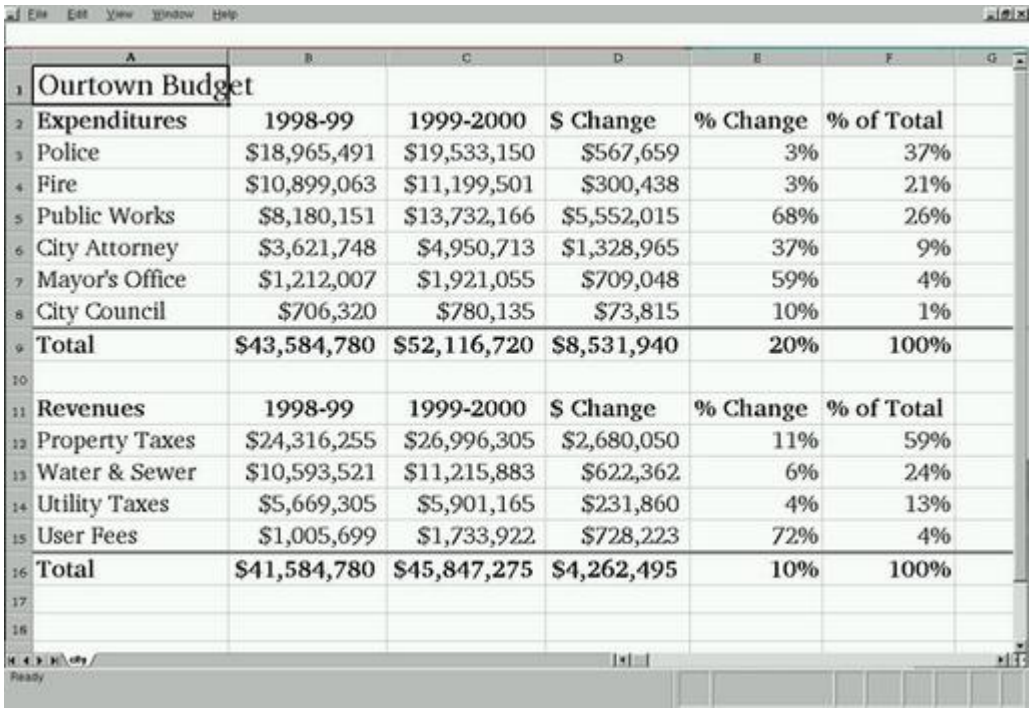


Figure 5. Viewing a Word 97 Document with WordView in Linux

The Excel Viewer installed and set up without a problem, but it would not view the Excel files found on some web pages (such as the Power Reporting site listed below). Thanks to CodeWeaver John Sturtz, I resolved the problem by adding this entry to my Netscape preferences:

Description: MS Excel Viewer MIMEType: application/xlsSuffixes: xlsApplication: /home/dlphilp/crossover/bin/wine.sh “C:/Program Files/XLView/xlview.exe” :switch:e “%s”

This entry is identical to the entry created by the Plugin Setup, except for the MIME type. After adding the entry to my preferences, I restarted Netscape, logged on to the Power Reporting site again, and *voilà*, I could view the sample spreadsheets while on-line (Figure 6).



	A	B	C	D	E	F
1	Ourtown Budget					
2	**Expenditures**	**1998-99**	**1999-2000**	**$ Change**	**% Change**	**% of Total**
3	Police	$18,965,491	$19,533,150	$567,659	3%	37%
4	Fire	$10,899,063	$11,199,501	$300,438	3%	21%
5	Public Works	$8,180,151	$13,732,166	$5,552,015	68%	26%
6	City Attorney	$3,621,748	$4,950,713	$1,328,965	37%	9%
7	Mayor's Office	$1,212,007	$1,921,055	$709,048	59%	4%
8	City Council	$706,320	$780,135	$73,815	10%	1%
9	**Total**	**$43,584,780**	**$52,116,720**	**$8,531,940**	**20%**	**100%**
10						
11	**Revenues**	**1998-99**	**1999-2000**	**$ Change**	**% Change**	**% of Total**
12	Property Taxes	$24,316,255	$26,996,305	$2,680,050	11%	59%
13	Water & Sewer	$10,593,521	$11,215,883	$622,362	6%	24%
14	Utility Taxes	$5,669,305	$5,901,165	$231,860	4%	13%
15	User Fees	$1,005,699	$1,733,922	$728,223	72%	4%
16	**Total**	**$41,584,780**	**$45,847,275**	**$4,262,495**	**10%**	**100%**
17						
18						

Figure 6. Viewing an Excel File with XLView

The various plugins also can be run as standalone applications. For instance, this command

```
$HOME/crossover/bin/wine.sh "c:/Program
Files/QuickTime/QuickTimePlayer.exe"
```

will start the QuickTime player from an xterm (Figure 7).

Figure 7. The QuickTime Player Running as a Standalone Application

Crossing Over

As of January 2002, the CrossOver Plugin has evolved to version 1.01. Notable additions and changes include support for Microsoft's PowerPoint viewer (completing the CodeWeavers' Microsoft viewers' collection), a simplified printing procedure (just select the viewer's Print menu item) and support for more browsers (including Galeon and Opera). This release also includes bug fixes for some 24bpp display problems and improved handling of QuickTime channels. Last but not least, a demo version of CrossOver is also now available (see the CodeWeavers web site for details).

Version 1.02 should be available by the time this review is published. Thanks to CodeWeaver François Gouget, I learned that we can expect the following improvements (and more): expanded browser support (the SkipStone browser will accommodate the plugin, and remaining problems with Opera and Konqueror should be fixed); enhanced multi-user support; simplified installation for the iPIX, MGI and Chime viewers; improved plugin retrieval, installation and setup; and the usual bug fixes.

The programmers at CodeWeavers have done the Linux community a great service with this product. The CrossOver Plugin is a well-designed package that installs easily and performs flawlessly. At long last I can enjoy QuickTime and Shockwave content from Netscape, and I don't have to boot into Windows just to look at Word or Excel files. I realize that some members of the community will object to paying for this software, but the price is reasonable, and the CodeWeavers truly deserve the support. They are major contributors to the Wine Project, and your purchase of the CrossOver Plugin helps fund that work. If you need browser support for QuickTime or Shockwave under Linux, or if you'd like to view your Excel and Word files without rebooting, then you need the CrossOver Plugin. There's just nothing else quite like it.

Acknowledgements

I would like to thank CodeWeavers Jeremy Newman, Jeremy White and John Sturtz for their assistance and for making something like the CrossOver Plugin in the first place. Thanks, guys!

Product Information/The Good/The Bad

Resources

email: dlphilp@bright.net

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 for *Linux Journal*. His favorite activities are still Linux, playing the blues and spending any time with his beloved Ivy Maria (not necessarily in that order).

Archive Index Issue Table of Contents

Advanced search

SnapGear Lite: an Inexpensive Home Office/Small Office Firewall and VPN Client

Alan Zeichick

Issue #96, April 2002

You need protection, and with due respect to my programmer friends, the best simple protection is a hardware firewall.

The Internet is great. But, connecting a computer—or a computer network—directly to a broadband access device, like a cable modem or DSL router, is asking for trouble. Crackers prey on small-office or home computer users. Indeed, a friend's computer was cracked less than a day after it was connected to his new cable modem last September. When I heard, I immediately advised my friend to purchase a hardware firewall. That's my advice for you, too. If you're not using some sort of firewall, it's only a matter of time before a cracker finds you. You need protection, and with due respect to my programmer friends, the best simple protection is a hardware firewall.

That's where SnapGear's Lite firewall appliance comes in. It plugs in to your broadband access device and your computer (or your home or office network), then plugs in to the Lite box. I didn't know about this particular device when my friend had those problems last fall, but had I known about the Lite, I would have told my friend about it.

The Lite appliance has two additional benefits beyond the anti-cracker firewall: network address translation (NAT) and a virtual private network (VPN) client.

Network address translation lets you put more than one computer onto the Internet. Normally, you need one public IP (Internet Protocol) address for each computer, like 123.45.67.89 or 65.11.11.11. That number, which is analogous to a telephone number, is assigned by your internet service provider. When you use NAT, one device (in this case the Lite box) is assigned the public IP address, and it acts as the IP gateway. Meanwhile, computers on your home or business LAN have private IP addresses that are for use only on the LAN. Those private

IP addresses are generally in the range of 192.168.x.x or 10.x.x.x, which are reserved for that purpose.

Let's say one of your computers, with a private IP address of 192.168.0.14, wants to check a web site. It asks the Lite, as the internet gateway, to get the information. The Lite translates the request to make it look like it came from 123.45.67.89 and forwards it to the web site. The web site responds to 123.45.67.89—that is, to the Lite. The Lite box then retranslates the request back to 192.168.0.14 and puts the packets onto your private LAN, so that your PC can receive the data and display the web page. With the Lite, the network address functionality works very well; you can put as many computers onto a single internet connection as you'd like, without an arbitrary limit.

Virtual private networking uses authentication and encryption to provide a secure link between two computer networks over the public internet. VPNs often are used to give telecommuters or branch offices direct access to a large corporate network, without exposing that network to crackers. Think of it as a secured tunnel between two buildings. To set up that tunnel, you need VPN functionality at both ends; one end acts as the host or server, and the other as a client that logs in to the VPN server.

VPN clients can be either in software, which enables a single computer to use the VPN, or in a hardware access device, so that it can bridge two networks and all the computers on those networks. The SnapGear Lite is a hardware-based VPN client, which can not only log in to most corporate networks using the IPSec protocol (which is the most common), but it also can emulate Microsoft's own PPTP (Point-to-Point Tunneling Protocol), which is the software-based VPN system built into Windows servers and Windows workstations. The Lite thus lets you use non-Windows PCs, such as Linux workstations or servers, to access a Microsoft-based VPN. That's an important benefit, if your employer uses Windows NT/2000 to host remote access.

Before we look at the Lite in detail, it's important to point out that none of these basic features—a hardware firewall, network address translation or a VPN client—is new. I've been using a similar hardware appliance from SonicWall for more than two years, and with the exception of the PPTP client, it's nearly identical. At least half a dozen other companies also make devices like this. What makes the Lite noteworthy is its driver support for Linux and its low price, which at $249 is a few hundred dollars less than other similar devices that I've used.

The Lite in Detail

About the size of an external Iomega Zip drive, the SnapGear Lite is a small box with a few LEDs, a jack for an AC adaptor and two RJ-45 connectors, one a

10Mbps Ethernet jack for hooking to your DSL or cable modem, the other a 10/100 Mbps Fast Ethernet jack for hooking up directly to a PC's network card or to your LAN switch or hub. The necessary cables are included.

The Lite also has an RS-232 serial port for configuring the Lite to work with a regular telephone modem. I didn't test the serial-port modem connection but rather used the device with my cable modem, in place of the SonicWall firewall appliance mentioned above, for about three weeks. On the LAN side, the Lite was hooked into a Linksys 16-port 10/100 Ethernet switch, which generally had between three and ten active computers at any time.

Also, as a matter of interest to *Linux Journal* readers, the Lite uses Linux internally, embedded into a 66MHz Motorola ColdFire XFC5272 microprocessor. A recent firmware upgrade, which came out during our review, gave the device the 2.4 kernel. Bear in mind that the Linux kernel is hidden in the device; you won't ever see or work with it directly.

Initial setup was simplicity itself; all you have to do is run a setup program on a local PC, which tracks down the Lite on the network and configures its private IP address (in my case, 192.168.0.14) and related information. That only has to be done once. Then, you browse to that IP address via Netscape or Opera, for example, and use that to configure the public IP address (so you can have internet connectivity) and then set up the firewall and VPN options.

The good news is that the Lite has a configuration program for Linux. The bad news is that it's not included on the CD-ROM that ships with the box—it only has the Windows version of the client. That's naughty and is a rather needless extra step for Linux users, considering that the compact disk only has 14.8MB of stuff on it. Be sure to download the Linux package from www.snapgear.com/downloads.html before you start tearing your network apart.

Setting up the Lite to provide network address translation and to use the proper public IP address was also straightforward using Netscape; my cable modem has a static IP address, issued by the ISP. In some cases those ISP IP addresses are issued automatically and dynamically, and according to the Lite's documentation, it can accommodate that type of network.

The Lite also can act as a DHCP (Dynamic Host Configuration Protocol), where Lite can assign private IP addresses automatically to the computers on your networks. I didn't use this setting, as my computers already had fixed IP addresses. Business networks wouldn't need that feature, as they likely would use fixed IP addresses or have a separate DHCP server, but this will be a useful feature for small-office or home networks.

One of the resources on my network is a web server. If you are expecting incoming traffic from the Internet, you have to configure the firewall to pass the appropriate type of data packets from the Internet to the appropriate private IP address, and thus to the right computer on your LAN. It was easy to redirect traffic on IP port 80 (HTTP) and port 23 (FTP) to my web server. The firewall appeared to do a good job of filtering bad packets; from outside my network, I launched Sub-Seven and Ping-of-Death attacks against the firewall and also attempted a port scan, and it blocked those attempts. Those are common crack attempts made against cable-modem users, and the firewall worked admirably.

My only reservation with the SnapGear Lite firewall is that it is not certified by a major firewall tester. Every other firewall I've tested, including SonicWall SOHO+, WatchGuard's Firebox and Check Point's best-selling (but expensive) Firewall-1 are certified by ICSA Labs (www.icsalabs.com), which puts firewalls through a pounding with a well-established and industry-standard test suite. The Lite isn't certified, and SnapGear doesn't even mention ICSA on its web site. However, according to a company spokesperson, SnapGear began working toward certification in January 2002 and hopes to have achieved it by the end of the second quarter.

The final tests involved making two virtual private network connections over the Internet. The first was to a Check Point VPN-1 device, which was configured for IPSec-based access. The second was a PPTP-based link to a Windows 2000 server. I had no trouble making either connection, though I would judge the process somewhat complex.

Someone with experience and good understanding of VPNs should have no trouble making it work. Someone without that experience will find the sketchy documentation (a 12-page booklet and a 73-page PDF electronic manual, with 20 pages devoted to VPNs) confusing and likely will need assistance from SnapGear or a knowledgeable system administrator. Once the VPN is set up, however, it appears to be reliable and sufficient for a typical small-office or home computer user. By the way, there is also an ICSA certification for IPSec compatibility that most major VPN product manufacturers use and promote; the Lite doesn't have that either.

So, here's the bottom line. If you're looking for hardware protection for your internet connection, the Lite is inexpensive and relatively easy to use. It also has the VPN functionality, and as far as I can tell, is unique in acting as a PPTP client, which is a real plus for Linux users. On the other hand, the firewall is not yet certified and the VPN functions aren't easy to set up. If you're okay with that, it's a good solution and certainly worth the price.

Product Information/The Good/The Bad

Alan Zeichick (zeichick@camdenassociates.com) is a technology analyst in the San Francisco Bay area who focuses on networking and software development.

Archive Index Issue Table of Contents

Advanced search

Letters

Various

Issue #96, April 2002

Letters

Zippy Linux?

I am very interested in your approach of installing Linux on an Iomega Zip 250MB drive for parallel ports ["Maragda: Running Linux from a CD" by Jordi Bataller, *LJ* March 2001]. I notice you were successful and wonder if you care to share your knowledge in this area. The PCs in our lab have internal 100MB IDE zip drives. Thank you for your great article and any assistance you can provide.

—David Cooper

Jordi replies: Running a Linux distribution installed on an Iomega Zip disk works and is usable, but it is slow compared to other solutions. The basics to make it work are the same as in the CD-ROM case, but you need a floppy (1.44) disk to boot it. Besides the slow speed, capacity is another drawback. I used a 250MB disk and still have to cut down the contents. There are other solutions available: running it from a CD-ROM (see www.iti.upv.es/~maragda) and using a parallel box enclosing a common IDE hard disk. This is a box connected to the parallel port, which interfaces a standard IDE hard disk. I used this solution for over a year with no problems (of course it is a bit slower). It is easy to create and the only problem is developing a bootable floppy disk. The installation is trivial; I connected the disk directly to the IDE bus. Once Linux is installed you remove the disk from the bus and put it into the box. It is cheap, since you can reuse old IDE disks and probably can find boxes for $60-$80 US. There are also other types of box adaptors. I'm waiting for an IDE-to-USB box; well I'm waiting for the developers of the Linux kernel USB support to patch some bugs.

One Cool Guy

I'd like to say that I enjoyed the January 2002 issue of *Linux Journal*, especially your interview with Guy F. de Téramond, Costa Rica's Minister of Science and Technology. This was the first time I have read your magazine (I picked it up initially due to the networking articles), and you can be sure that it won't be the last if this issue was any indication of its quality and breadth of coverage.

—Marc Rosenthal

Scalable Clairvoyance

Wow, have you got Cleo working there or what? I made an appointment with my new ISP for wireless 802.11 access in the morning, and the February *LJ* arrived later that day. Well done!

—Dave Moulton

Garage Comments

In “The Perspective from My Garage” [Linux for Suits, *LJ* February 2002], Doc Searls ends his review of misguided prognostications with a July 1976 ad for the Altair 8800 in *Popular Electronics*. True, this early microcomputer did not succeed. But it was at the center of a significant milestone in the development of the personal computer—a milestone that will impact the growth of Linux for years to come. The Altair 8800 was chosen by two young Harvard University students as the platform for the first microcomputer, Basic interpreter. The older of the two, Paul Allen, simulated the Altair instruction set on a DEC PDP-20 and was pleasantly surprised when his interpreter, stored on paper tape, actually worked when he brought it to MITS in Albuquerque. He and his younger colleague thereby were able to start a software company. Mr. Allen eventually left the company, but his partner, Bill Gates, has become the wealthiest man in the world from that company, Microsoft. (Mr. Allen is not far behind.)

—Jay Braun

Mozilla Corrective

There is a small mistake in the February 2002 issue of *Linux Journal*. The Tech Tip on page 88 states that you can specify a minimum font size for Mozilla with:

```
user_pref(font.minimum-size.x-western, 13);
```

On my system (running Mozilla 0.9.7) I need to place quotes around the first argument:

```
user_pref("font.minimum-size.x-western", 13);
```

Otherwise, I get the following error:

```
An error occurred reading the startup
configuration file. Please contact
your administrator. ReferenceError:
font is not defined.
```

By the way, thank you for an excellent issue of *LJ*. It's my favorite one yet!

—Mike Voytovich

Sophomoric?

The article “Using Debian Apt-get over Freenet” in the February 2002 issue of *LJ* was interesting and informative. Although there are reasons to claim dpkg/apt is more sophisticated than RPM, the statement, “Get a real package manager!” is sophomoric enough to have pleased the *Maximum Linux* bunch (whatever actual and useful information they may have provided—RIP).

—Ken

Archive Index Issue Table of Contents

Advanced search

UpFront

Various

Issue #96, April 2002

Cuzya Hafta, Sometimes

If you have to stay in a Windows environment, you still can save money with Linux. That's the thinking behind Thin Computing's WinConnect, a “Remote Desktop Protocol” that lets you use a generic Linux workstation—even cheap or recycled iron—for Windows 2000 terminal sessions. Thin Computing is at www.thincomputinginc.com.

—Doc Searls

LJ Index—April 2002

1. Percentage of its enterprise that Plastic Dress-Up (the leading supplier of trophy and award components) has moved to the Linux platform: 80-90
2. Linux-derived cost savings in dollars on each new enterprise server purchased by Plastic Dress-Up: 30,000
3. Millions of images in the 1mage document imaging system on one Red Hat Linux server at Plastic Dress-Up: 1
4. Recovery percentage of its image database after a power outage at Plastic Dress-Up, in spite of a damaged drive partition: 100
5. Years during which Streptococcus mitis bacteria survived on the Moon between visits from Earthlings: 2.5
6. Years during which bacteria lived in the brickwork of Peruvian pyramids: 4,800
7. Years during which bacteria lived in a mastodon corpse: 11,000
8. Range in millions of years during which bacterial spores survived in amber-trapped bees: 25-40
9. Age in years of bacteria revived after recovery from a New Mexico mine shaft: 250,000,000
10. Estimated number of computer viruses in 1990: 200-500

11. Estimated minimum number of computer viruses in 2000: 50,000
12. Total number of reported Linux viruses: 1
13. Cost in billions of dollars in virus damages by September 2001: 10.7
14. Number of observable domains hosted by 21VIANET.com in Beijing, China: 3
15. Latest uptime in days of the 21VIANET domain (www.encantata.com.cn) running Apache on Linux: 305
16. Latest uptime in days of the the two 21VIANET domains running IIS on Windows 2000: 10, 19

Sources

1-4: 1mage (www.1mage.com)

5-9: Panspermia.org

10-11: CKnow.com

12: Librenix.com

13: Computer Economics (www.computereconomics.com)

14-16: Netcraft (www.netcraft.com)

Raising the Red Flag

We all know Big Blue is spending a billion dollars on Linux because a visible hunk of that money is being spent on advertising and promotion.

Less obvious is how the Chinese government, representing more than 1.2 billion people, is expressing its own love of Linux by encouraging adoption of its own distribution: Red Flag Linux.

Red Flag was created in 1999 by the Academy of Science, which is headed by Jiang Mianheng, son of President Jiang Zemin, with financial assistance from the government-owned Shanghai NewMargin Venture Capital.

Red Flag's purpose is to prevent increased domination of the Chinese computer market by Microsoft's Windows operating systems. The best way to do that, as

the Chinese government sees it, is to promulgate an already popular alternative with “full transparency in terms of underlying code”, as one commentator described it. They are doing this by encouraging state institutions and state-owned companies to adopt Red Flag Linux.

And that's just one strategy. Another is to avoid copyright infringement and “software piracy” issues by promoting use of software that avoids the issue. Another is by purchasing software from domestic companies that build on Linux rather than Windows.

According to Gartner, the Beijing municipal government gave contracts to six local vendors and rejected the seventh: Microsoft. One of the six was Red Flag.

Not surprisingly, Linux is now showing up in quantity on desktops, at least in retail stores.

> On a recent trip to China, I noticed that many of the Intel-compatible PCs for sale in several major department stores had a slightly different appearance than usual, apart from the language differences. I looked closely and realized they were running GNU/ Linux

writes Dan Gillmor of the *San Jose Mercury News*.

—Doc Searls

Stop the Presses: A Tale of Two Bazaars: the Marketplace and the Kernel List

At Linux WorldExpo in August 1999, I met with some IBM folks who conveyed the company's cautious interest in the operating system. Linux had been arriving “over the transom”, they said, showing up significantly (though nonstrategically) in servers all over the company. So IBM did a survey of one division, testing Linux “awareness” on a scale that ran from “can spell Linux” at one end and “hacks kernel code” at the other. All 600 surveyed could spell Linux, and 120 were hacking kernel code. It was a revelation that no doubt informed the company's strategic commitment to the OS.

Now it's 2.5 years later, and Linux is a mainstream operating system. That was the summary news story from the latest LinuxWorld Expo (just completed as I write this).

IBM showed off marquee customers L.L. Bean, Boscov, Pixar and Solomon Smith Barney. Hewlett-Packard showcased DreamWorks SKG. Egenera reported a deal with Credit Suisse First Boston.

In her keynote at LWE, Carly Fiorina of HP said 2002 would be a “breakout” year for Linux, pointing to a Gartner projection of 15% growth, despite the economic downturn. HP also announced Linux products for enterprise and telecommunications customers, showcasing Amazon, BMW, Boeing, Speedera, ViaWest and Verizon.

Also at LWE, Holger Dyroff of SuSE said, “Up to now Linux was adopted by technicians. Now it's exactly the opposite.” CIOs are placing orders for SuSE support contracts on IBM mainframes at a rate of 2-3 purchase orders per week and $4,500-$11,500 per contract. About one-third of the customers for the mainframe distribution, which has been out for about a year and a half, are banks, Dyroff said.

IBM also introduced a special bargain price on a “Linux-only” version of its flagship zSeries (better known as S/390) mainframe.

In fact, IBM, which famously bragged about spending one billion dollars on Linux over the last year, now says it has recouped most of the investment. And it's hardly alone in its bottom-line enthusiansms for the operating system.

And the story goes way beyond what we saw at LWE.

Egenera is selling Linux-based servers ranging in price from $200,000 to more than one million dollars.

Amazon.com recently moved its services to Linux and credits the OS with enormous savings, no doubt contributing to that company's first profitable quarter.

Mary Anne De Young of 1mage (“One Image”) attributes the company's recent successes to a tremendous growth in demand for Linux as a UNIX product platform. One of 1mage's largest customers, Reynolds & Reynolds, is both making and saving money by shipping its (and 1mage's) products out to thousands of car dealers on Linux boxes.

John Gantz of International Data Corp. had enthusiastic words about Linux in his Top Ten Predictions for the New Year.

And if none of that makes a convincing case for the long-term commercial success of Linux, there's this from Microsoft CEO Steve Ballmer:

> I think you have to rate competitors that threaten your core higher than you rate competitors where you're trying to take from them....It puts the Linux phenomenon and the UNIX phenomenon at the top of

> the list. I'd put the Linux phenomenon really as threat No. 1.

—Doc Searls

They Said It

Consider your newspaper. The advertisements are the land. The stories are the Dead Sea, slowly evaporating. Beneath the surface, the advertisements form an unbroken continental shelf. Here and there a ridge of PR crops through the shallow surface. Soon salt and sand will blow across a desert.

—Wealth Bondage

Listen attentively, and above all, remember that true tales are meant to be transmitted. To keep them to oneself is to betray them.

—Elie Wiesel

The motivation for this counterrevolution is as old as revolutions themselves. As Niccolò Machiavelli described long before the Internet, “Innovation makes enemies of all those who prospered under the old regime, and only lukewarm support is forthcoming from those who would prosper under the new.” And so it is today with us. Those who prospered under the old regime are threatened by the Internet. Those who would prosper under the new regime have not risen to defend it against the old; whether they will is still a question. So far, it appears they will not.

—Lawrence Lessig

Our terrorists wear suits and have law degrees. Their involvement in software design, at a very intimate level, will result in orphaned software, bankruptcies and users without tools to use. Movement stops—who knows how a creative lawyer will be able to maneuver a broad patent to cover something truly new and innovative. Sometimes the innovation is in cooperation.

—Dave Winer

We were born naked, wet and hungry. Then things got worse.

—Simon Fraser

There isn't much value in free.

—Doug Miller, group product manager for competitive strategies at Microsoft

The reason that we have not seen a real Linux virus epidemic in the wild is simply that none of the existing Linux viruses can thrive in the hostile environment that Linux provides. The Linux viruses that exist today are nothing more than technical curiosities; the reality is that there is no viable Linux virus.

—Ray Yeargin

A time is marked not so much by ideas that are argued about as by ideas that are taken for granted. The character of an era hangs upon what needs no defense. Power runs with ideas that only the crazy would draw into doubt. The "taken for granted" is the test of sanity; "what everyone knows" is the line between us and them.

—Lawrence Lessig

Everywhere is walking distance if you have the time.

—Steven Wright

If I had eight hours to chop down a tree, I'd spend six sharpening my axe.

—Abraham Lincoln

Beware of methodologies. They are a great way to bring everyone up to a dismal, but passable, level of performance, but at the same time, they are aggravating to more talented people who chafe at the restrictions that are placed on them.

—Joel Spolksy

Never threaten a writer....We can immortalize you in ways you might not find pleasant.

—Garrison Keillor

Hey, Maybe There's a Patent for Badness Itself

What do 3-D pie charts, training manuals, gene profiling and advertising effectiveness measurement have in common? Gregory Aharonian of BustPatents.com lists them as his top four bad patents of all time, in response to a request for the list from *Scientific American*, which published the results.

All four apparently passed (or flunked, depending on your point of view) the Obvious Test at the Patent and Trademark Office (PTO).

According to Aharonian, patents recently invalidated for various reasons include:

- Device for perfusing an animal head
- Video processing for composite images
- Call message recording for telephone systems
- Negotiable instrument fraud detector and processor

Perhaps the last one failed to detect itself.

—Doc Searls

Archive Index Issue Table of Contents

Advanced search

LinuxWorld, New York

Richard Vernon

Issue #96, April 2002

LinuxWorld, New York

Tux eats big apple, but will he get indigestion?

New York is a business town—a city of professionals who carry out with efficiency whatever it takes to make their buck. This is evidenced not only in the office buildings, but in the numerous clubs, restaurants and stores that take only cash; the race-car taxi drivers (like racers not so much in speed as in skill and disregard for lanes and turn signals); and in the high number of suits in attendance at LinuxWorld Expo, New York.

That number was reflected by the many vendors exhibiting “enterprise-level” hardware and applications. The big players on the tradeshow floor were no longer so much trying to convince a community of hackers of their commitment to Linux as proving to a community of businesses the viability of Linux for all of their computing needs. Judging from some of my discussions with vendors, much of that community needs no convincing. Many vendors from 1mage to Egenera to CA report that the number of customers coming to them asking for Linux-only solutions is mushrooming.

While the fact that use of Linux is growing rapidly is certainly not news, it's heartening to see that it's not only in the server market. For example, the new HP X4000 workstation was designed for Linux with use in the Hollywood GFX/

animation market in mind, and Hancom's office suite seems the answer to certain MS products that StarOffice has never quite delivered.

Some old-timers at the show expressed concern that the heavy business migration of the open-source OS would change the face of Linux—and that even Tux might be replaced with some corporate logo-like logo thing.

Certainly the fear is understandable. We hate to see tradition sacrificed to business—as does any New Yorker old enough to remember the days when vast herds of strong roomy reliable Checker A11 taxis roamed the wild streets of the city before being decimated by rising oil prices. The turn to more economical cars by taxi companies caused Checker Motors to retire from the auto manufacturing business, but the final blow was delivered in 1999 by the New York Taxi and Limousine Commission when they failed to grant any allowance regarding the standard technical inspection to the last remaining Checkers, thus making the city's symbol a symbol of the past. (As a Checker owner, I'm glad to see their collector value soar as a result, but as a Checker lover I'd rather see them in their natural environs.)

But I don't think Tux will go the way of the Checker (but if he did, think of the fortune you'd make on eBay with all your Tux paraphernalia). Like Doc Searls, I'm convinced that the mix of business and Linux is a good thing—and for the same reasons. After all, as long as there is Linux, there will be Linux developers and Linux will remain a hackers' OS.

email: sophie@peoplepc.com

Richard Vernon is editor in chief of *Linux Journal*.

Archive Index Issue Table of Contents

Advanced search

Best of Technical Support

Various

Issue #96, April 2002

Our experts answer your technical questions.

RPM Won't Let Me Upgrade

On a Red Hat system with RPM 4.0.3-1.03, it seems the options --nodeps and --force don't prevent the system from checking dependencies. Therefore, I am not able to update certain packages. For example,

```
rpm --nodeps --force -Uvh db3-3.2.9-4.i386.rpm
```

gives the error:

```
failed dependencies:
libdb-3.1.so is needed by pam-0.72-26
libdb-3.1.so is needed by sendmail-8.11.0-8
# rpm --nodeps --force -Uvh pam-0.75-14.i386.rpm
error: failed dependencies:
        libdb-3.2.so   is needed by pam-0.75-14
```

My older versions of RPM used to work.

—Yi Zhao, yzhao2@yahoo.com

I usually get rid of this kind of message by upgrading everything in a single RPM call:

```
rpm -Uvh -force --nodeps pam-0.75-14.i386.rpm
sendmail-XXX db3-3.2.9-4.i386.rpm
```

—Mario M. Bittencourt Neto, mneto@buriti.com.br

Checking Both /etc/hosts and DNS

When I try to resolve a name that is not in DNS but is in my /etc/hosts file, nslookup refuses to look at /etc/hosts. I want to look at both /etc/hosts (first) and then dns. My /etc/nsswitch.conf file has this:

```
"hosts: files [NOTFOUND=continue] dns"
```

I have a mixed environment of UNIX platforms, and my other UNIX boxes (HP-UX and Sun) work with this configuration. Can you tell me what I am missing with my Linux setup?

—Jim Booker, jim.booker@verizon.com

nslookup will not look at /etc/hosts; this is normal behavior. The **host** command will look at both, however. The HP-UX and Sun nslookup command may have been modified to function differently.

—Marc Merlin, marc_bts@valinux.com

USB Keyboard Stopped Working

I have a Compaq Presario 5000 series PC, on which I've installed a number of different systems, including Red Hat 7.1, from which I'm writing this question. When I attempt to install Red Hat 7.2, I am having a problem getting the installation procedure to accept keyboard input. I have a USB Compaq Internet PC keyboard. Has there been a regression in the installation procedure that causes problems with USB keyboards? If so, is there a workaround?

—Brian W. Masinick, masinick@yahoo.com

Try configuring your BIOS to provide “Legacy Keyboard Support”.

—Christopher Wingert, cwingert@qualcomm.com

I see you are using Red Hat, but for those having the same trouble using Mandrake, I have seen reports of Mandrake 8.1 having a file /etc/sysinit/usb that requires the entry KEYBOARD_AT_START=NO in order for Linux to work and function correctly with a USB keyboard.

—Felipe E. Barousse Boué, fbarousse@piensa.com

SuSE Won't Boot, Red Hat Won't Display

I installed a copy of SuSE, which uses LILO, on my Dell Dimension L866r, and when I boot, I just get a bunch of 0s and 1s flowing over my screen. I installed Red Hat 7.2 and it installed GRUB, which worked. Unfortunately, 7.2 (like all Red Hats from 4.x onward) can't handle my Nokia Multigraph 447x monitor. I sent an e-mail to them, but they didn't respond.

—Jim Macdonald, jimm@mediaone.net

I think the simplest thing to do would be to use Red Hat 7.2 and set up the monitor modes manually. The mode information for your monitor is located at www.ibiblio.org/pub/linux/distributions/redmondlinux/redmond/build38/live/usr/share/hwdata/Monitors.

—Christopher Wingert, cwingert@qualcomm.com

Chances are that LILO is having a BIOS geometry problem. Do you have your BIOS set to some sort of geometry translation? Usually this is called large disk support, or LBA mode. This will bring the number of cylinders down to a lower number (hopefully below 1,024 cylinders) and allow LILO to see the whole disk. If this isn't possible, make sure that the partition on which your kernel gets installed (/ or /boot depending on the distribution) is all contained below 1,024 cylinders. This error also may mean that LILO interpreted your disk geometry incorrectly. You may have to tell LILO what your real geometry is, or you may need to give LILO the "linear" option. Look at section 2.2 at www.linuxdoc.org/HOWTO/mini/LILO-2.html for more information.

—David Brown, david@caldera.com

The page www.geocities.com/SiliconValley/Peaks/3233/linux.html has a link to an /etc/X11/XF86Config file running with a Nokia Multigraph 447X as yours.

—Felipe E. Barousse Boué, fbarousse@piensa.com

You Can't Log in Now; Get a Life

I have a couple of questions. First, how would you create a backup user for root with the same privileges as root? Second, is there a way to allow users to log in only within a specific time frame? An example of this would be to allow a user to log in between the hours of 6 **A.M.** and 6 **P.M.**, but not to allow logins outside of this window.

—Jerry Fulkerson, hrlinkin@aol.com

To add a second root account, edit /etc/passwd and /etc/shadow (using vipw and vipw -s). In both files, duplicate the line with root and change the name to backuproot.

—Marc Merlin, marc_bts@valinux.com

Regarding control of login time, there are several ways of doing it. Using the Pluggable Authentication Modules (PAM) system's pam_time module is one of them, for instance. The file /etc/security/time.conf could have the line:

```
login;*;joe;Al0600-1800
```

This means the user joe is allowed to use the service called login, from any terminal (*), all days (Al) only during 06:00 and 18:00 hours. Be aware that this requires the entry:

```
login   account required   pam_time.so
```

on the PAM configuration file, which is usually at /etc/pam.conf or alternatively, the file named login to be within /etc/pam.d/ and containing:

```
account required pam_time.so
```

A good PAM reference can be found at www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam.html#toc4.

—Felipe E. Barousse Boué, fbarousse@piensa.com

I Have No DHCP and I Must Network

Currently, I use Red Hat 7.0 on PCs in my computer programming lab. These computers use DHCP to connect to the building network and the Internet. Recently I attempted to upgrade these machines to a stock version of Red Hat 7.2. I have been unable to get computers to connect via DHCP. I have tried to configure the card and networking using the various GUI tools but have had no success.

—Bill Hummel, hummelb@readingsd.org

Looks to me that you are having trouble connecting because of two possible issues. First, you have not configured the DHCP client to connect to the network. I just did it on a Red Hat 7.2 machine by having the minimally configured file /etc/sysconfig/network-scripts/ifcfg-eth0 for network interface eth0 as:

```
DEVICE="eth0"
BOOTPROTO="dhcp"
ONBOOT="yes"
```

This will provide enough information for the system to boot up, and when starting networking facilities, to look for an IP address provided by a DCHP server. Second, you probably set a firewall or network filter that is not allowing DHCP to work properly. Did you request a “high” security level when installing? Just one time, get rid of the ipchains setup that Red Hat 7.2 sets up by commenting all lines in /etc/sysconfig/ipchains and then try step one.

—Felipe E. Barousse Boué, fbarousse@piensa.com

What Does ./ Mean?

When (and why) is it necessary to put ./ in the beginning of a command line?

—Murray Zangen, murray@nj.com

The ./ means current working directory. You need to put ./ in front only when you want to run a program in the directory you are in, and that directory is not in the PATH shell variable ($PATH).

—Usman Ansari, uansari@yahoo.com

The reason the current directory is not in the path is due to security concerns. If an adversary installed a Trojan ls command in the /tmp directory and you cd-ed into /tmp and typed **ls**, you would run the adversary's program. If you really don't like having to type **./**, place the current directory at the *end* of your PATH shell variable, this will make your file system surfing safer.

—Christopher Wingert, cwingert@qualcomm.com

Opening a File with filep_open

I have created a configuration file stored in the directory /etc. I have been trying to open this file from a C program without success:

```
struct file     *filp;
char *Filename = "/etc/pg.conf\0";
filp = filp_open(Filename,00,O_RDONLY);
if (IS_ERR(filp)||(filp==NULL))
        return;
```

Could someone advise me how to fix this?

—Senthil, senthil@singnet.com.sg

I believe that you do not have the parameters correct to filp_open(). Try

```
filp = filp_open( Filename, O_RDONLY, 0 );
```

—Christopher Wingert, cwingert@qualcomm.com

Setting up a Video Conference

Anyone done anything with web cameras under Linux? I have a need to do some video conferencing with some people in the US.

—Arnold Robbins, arnold@skeeve.com

Many Parallel/USB web cameras are supported by Linux. You should pick a camera that you are interested in and search the web to make sure that camera is supported under Linux. Check out www.openh323.org/h323_clients.html for video conferencing support.

—Christopher Wingert, cwingert@qualcomm.com

I guess the best for you would be to visit www.linux-usb.org (look for the Working devices list) and www.freesoft.org/software/NetMeeting. This HOWTO is about communicating between Microsoft Netmeeting and Linux. Lastly, video and conferencing in Linux are evolving rapidly and are subjects in constant improvement. These links may be good initial references for you to begin with, as they were for some people who just deployed a large Linux-based videoconferencing system.

—Felipe E. Barousse Boué, fbarousse@piensa.com

Philips has a site for Linux-compatible cameras using USB: www.smcc.demon.nl/webcam. For conferencing, check out www-nrg.ee.lbl.gov/vic/#overview and www.gnomemeeting.org.

—Paul Christensen, pchristensen@penguincomputing.com

Archive Index Issue Table of Contents

Advanced search

New Products

Heather Mead

Issue #96, April 2002

New Products

1000 Series ATM Cards

ImageStream Internet Solutions is shipping its new 1000 series ATM network adaptors, including DS3/E3 and OC3 cards in PCI and PMC formats. The new cards are designed to work as components in routers, servers and test equipment. The 1000 series ATM adaptors are 32-bit PCI 2.1-compliant cards that feature ATM segmentation and reassembly to optimize PCI bus performance with small packets. The PCI and PMC format cards include the 1001-DE, a dual-function DS3/E3 card and the 1001-O3M and 1001-O3S, which are OC3 cards for multimode and single-mode applications.

Contact ImageStream, Inc., 7900 East 8th Road, Plymouth, Indiana 46563, 800-813-5123 (toll-free), info@imagestream.com, www.imagestream.com.

eclipse.org C/C++ IDE

eclipse.org has made an open-source C and C++ integrated development environment (IDE) available for the eclipse platform on Linux workstations. The C/C++ IDE for the eclipse platform includes a GUI-based source code editor, an integrated debugger and the ability to access a command-line interface for forming advanced debugger requests. It joins existing Java support on the eclipse platform, an open-source environment for creating, integrating and deploying application development tools across a broad range of technologies. The C/C++ IDE is available for download from the web site.

Contact www.eclipse.org.

NIC Express 1.0

NIC Express 1.0 is a hardware and vendor-neutral trunking solution for load balancing network traffic across two or more network connections, thereby increasing throughput and fault tolerance. NIC Express also increases fault tolerance by searching the network in addition to the server. It locates both physical and logical faults and reroutes traffic, eliminating single points of failure between any two endpoints on the network. Version 2.4 and higher of the kernel are supported, and the NIC Express works with any 10/100/1000 Ethernet NIC and any layer 2/3/4 switch.

Contact IP Metrics Software, Inc., 416 North Main Street, Suite #231, Euless, Texas 76039, 877-358-1007 (toll-free), sales@ipmetrics.com, www.ipmetrics.com.

Geodesic Suite

The Geodesic Suite contains analyzers, debuggers and diagnostic tools for application development, testing and deployment. Included in the suite are Great Circle, a development debugging environment that operates via a web browser; Geodesic Runtime Solutions, which integrate into deployed applications to diagnose and resolve server-based problems at runtime without interruption; and Geodesic Analyzer, which installs at runtime and reports performance bottlenecks and reliability risks. Support is provided for 64-bit Itanium processors and 32-bit Pentium processors.

Contact Geodesic Systems, 414 North Orleans, Suite 410, Chicago, Illinois 60610, 800-360-8388 (toll-free), sales@geodesic.com, www.geodesic.com.

Ch 2.1

Ch 2.1, a C/C++ interpreter for scripting that is free for academic and nonprofit use, is now available from SoftIntegration. A superset of C with C++ classes, Ch 2.1 supports C90, major features of C99, POSIX, X11/Motif, OpenGL, ODBC, XML and GTK+. It also offers support for generic mathematical functions, computational arrays and advanced numerical functions for linear systems. Geared toward rapid application development, Ch can be used for cross-platform shell programming for regression testing, system administration, automating tasks, real-time interactive computing, rapid prototyping and 2-D/3-D plotting.

Contact SoftIntegration, Inc., 216 F Street, #68, Davis, California 95616, 530-297-7398, sales@softintegration.com, www.softintegration.com.

Red Hat Network Workgroup

Red Hat announced enterprise-level offerings for its Red Hat Network Services, under the Workgroup heading. The new Workgroup features and services include system grouping, allowing administrators to group workload-focused systems together for management purposes; multiple administrators, so that rights to specific systems can be spread across large organizations; and system set managers, allowing actions to be applied to sets of systems instead of single systems. In addition, the Workgroup Service subscription allows access to the network proxy server and the network satellite.

Contact Red Hat, Inc., PO Box 13588, RTP, North Carolina 27709, 888-733-4281 (toll-free), www.redhat.com.

Compaq ProLiant BL e-Class

The ProLiant BL e-Class, developed by Compaq, is an ultra-dense server blade architecture engineered for enterprise-level systems, enabling 280 servers to fit into a standard 42U rack. Designed for efficiency in corporate data centers and xSP environments, the BL10e consists of one Intel ultra-low voltage Pentium III processor, up to 1GB of ECC memory, 30GB of disk storage and two 10/100 Ethernet connections. The 3U chassis supports redundant hot-plug power and cooling, network connections and embedded technology that allows customers to manage Windows 2000 and Linux OSes within the same enclosure.

Contact Compaq Computer Corporation, PO Box 692000, Houston, Texas 77269, 800-282-6672 (toll-free), www.compaq.com.

Archive Index Issue Table of Contents

Advanced search