Linux Journal Issue #76/August 2000

The Internet has shifted the power of presence, acquisition and is in the beginnings of shifting the power of commerce. Many Value Added Resellers (VARs) have recognized this shift and have been able to change with the new economy.

Reviews

Columns

Departments

Archive Index

Advanced search

Focus: Home Applications and Linux on the Mac

Editorial Staff

Issue #76, August 2000

In this issue, we have a double feature: we look at using Linux at home, and on the Macintosh.

Use of Linux continues to grow in all computing markets. Over The past few years has seen Linux grow fastest in the server market, where it currently owns 31% of the market (compared to 24% for Windows NT). Today, it appears that Linux is growing fastest in the Embedded Systems market. In fact, next month's issue of *LJ* will focus on that market. Look forward to an article on the Helius router. Learn about “plain Linux”, Yopy and much more.

In this issue, we have a double feature: we look at using Linux at home, and on the Macintosh. Certainly, a home Linux system could be running on a Mac, so the two subjects fit together well. Using Linux in the home is vital to its growth. Having a version of Quicken for Linux is a necessary step forward. In this issue, Ralph Krause details various options for personal finance management.

Setting up Linux to run on a Mac opens up the wonderful world of Linux to a whole new set of users. You don't necessarily need to throw that Mac away—learn how to install Linux, instead! Stew Benedict and Richard Kinne explain procedures for installing and configuring a Mac to run Linux.

While our features are here to address the issue focus, you might want to remember that Linux is Linux. That is, Linux at work, Linux at home and even Linux on a server is all the same basic Linux. Even if your only interest is in running Linux at home, much of what you will see in *Linux Journal* applies as well. Our Linux Apprentice column, for example, contains useful information and tips—no matter where you run Linux. This month we learn how to configure a heterogeneous Linux/Windows home network. Book and product reviews always contain much information. Root through our reviews of routers, Wordperfect Office 2000 Deluxe, OpenLinux eDesktop 2.4, Imagestream routers and much more. And don't forget this month's Take Command, which

explains the klogd kernel logging dæmon. All this in just one issue of *LJ*! Read on.

Our first focus article on the Mac side is by Stew Benedict: Yellow Dog Linux on the iMac. Yellow Dog is a port specific for the Mac, and Stew takes you through his experiences of getting Yellow Dog up and running.

Our other Mac feature is Richard Kinne's piece on Linux on the PowerPC. Richard discusses the various distribution choices for the Mac, and then goes on the concentrate on the LinuxPPC package. With commercial applications for LinuxPPC lagging behind their counterpart for the x86 platform, Richard explains why a Mac user of Linux isn't out in the cold.

Our third feature looks at finance programs available for Linux. Reviewed packages include cbb, Moneydance, QHacc, gAcc and GnuCash. The article is a great guide to your options and includes information on file compatibility with the de facto QIF format.

Archive Index Issue Table of Contents

Advanced search

LinuxPPC on the Macintosh PowerBook

Richard Kinne

Issue #76, August 2000

Graphical installation environments help Macintoshes play better than ever with Linux.

To say Linux has undergone a growth in popularity over the last few years is, of course, an understatement. One cannot read any computer-related medium today without being bombarded with news and views on the Linux operating system. Some would even have you believe that Linux has the chance to unseat Windows as the operating system of popular choice. In the Intel-based world, we can choose from several Linux packages that can be installed on our machines. Intel-based Linux is even being courted by the game publishers, with such popular titles as Quake III now available.

What some people may not realize is that Linux for the PowerPC chip, usually on a Macintosh platform, is not far behind and has experienced similar growth in the last few years since its first inception as MkLinux back in 1996. Today, PowerPC users can choose among the original MkLinux package, which still offers its Mach kernel version of Linux, or the PowerPC and Yellow Dog packages with their monolithic RPM-based distributions. New distributions on the PowerPC chip horizon include SuSE, a popular German-based package, TurboLinux, popular in Asia, and RockLinux, billed as a “power user” distribution.

This article concentrates on the LinuxPPC distribution package, its installation and use on the Macintosh PowerBook, specifically the G3 Wallstreet PowerBook. Longtime readers of *Linux Journal* might find that odd, since I wrote the breakthrough article on MkLinux back in 1996. As with everything connected with Apple Computer, MkLinux went through some difficulties in the late '90s. In the summer of 1998, Apple stopped its support of MkLinux and development was turned over to the MkLinux community, which caused a slowdown in support of Apple's newer machines, as MkLinux was largely still working off of the DR3 kernel release. At the same time, I became a certified

“road warrior” by getting rid of my desktop machine and computing exclusively on my Wallstreet G3 PowerBook. MkLinux's support of that model was far from ready for prime time. I am happy to report, however, that the MkLinux community has been busy generating what they have called “generic kernels” that do support most of Apple's latest machines, including the PowerBooks. However, since that wasn't the case a couple of years ago, I migrated to the LinuxPPC package.

Installing LinuxPPC

LinuxPPC recently released their LinuxPPC 2000 package. This package contained, of course, various kernels in the 2.2.*x* range, including some specifically compiled for certain models of Macintosh PowerBooks. Also included are both the KDE and GNOME desktop environments, the Netscape browser, and a Mac-on-Linux emulator that allows you to run the Mac OS in a window from within Linux. The CD also contains the full range of Linux development tools including GNU C, C++ and Java. Nice extras include PalmPilot synchronization software for both the KDE and GNOME environments. A second CD provides all relevant source code.

If you're not willing to wait for the CDs or aren't into development or the KDE desktop environment, you can download a LinuxPPC 2000 “lite” version (see Resources). The LinuxPPC lite version includes the basic system, libraries and the GNOME desktop environment. The file is about 170MB, so even though it's not a quick download, LinuxPPC did give some thought to those of us using modem connections to the Internet.

One of the highly touted features of the LinuxPPC 2000 release is its user-friendly graphical installation environment. The installation CD is meant to boot your system and load whatever it needs onto the Macintosh in order to start installing Linux. These components will vary from system to system, depending on whether or not the “Open Firmware” of the computer involved is well-supported, or whether the system boots better using the included BootX utility that operates somewhat like LILO on Intel-based machines. Due to oddities in the Open Firmware implementation on the Wallstreet PowerBook G3, it commonly uses BootX in order to load Linux, and the installer realizes this. Once any Macintosh portion of the installation is done, the computer reboots and continues with the Linux installation process.

The graphical installation environment now comes into play. The computer will boot into Linux enough to give you a simple, restricted X Window System environment. These simple windows will guide you through disk partitioning, formatting, mounting, choosing packages for installation and allowing you to choose your root password. Upon reboot, your Linux installation is complete. If

for some reason the X environment will not load, the DOS-like Red Hat installer (common with LinuxPPC in the past and familiar to anyone who has loaded Red Hat Linux on anything in the last few years) will automatically be invoked, allowing you to continue with your installation.

Simple, right? Well, anyone who has ever installed Linux onto any system knows that, try as the package companies may, installation is never quite that simple. It's worth sharing a couple of “gotchas” I experienced during my own installation. If you'll be using BootX to do your LILO-like choosing between MacOS and Linux, you must remember to reset the control panel from the RAM disk used by the system to do the initial installation to the root partition. Remember to write down which partition you use for root, since you'll need it at this point.

If you're not an old hand at Linux installation, that is, if the concept of the user-friendly installation interface is a selling point for you, accept the fact that the initial default desktop environment for the LinuxPPC installation will be GNOME. This was a little disheartening to me, since I'm a KDE fan and I wanted to set up KDE as my desktop environment. You are able to do that, but let the installation install GNOME as the default at first. Problems have been reported and experienced in getting KDE to run right after installation, because the installer apparently doesn't install the QT1x libraries that KDE needs. It's a simple matter, upon reboot, to go into GNOME, install the libraries via the RPM program if needed, then go into KDE. Once there, you can modify and customize things easily.

The good news is the installation procedure, if allowed to install the defaults, solved all the previous installation problems I'd had with Xconfig in the past.

Congratulations! You now have a portable UNIX system that would have run several colleges in the late '80s. So now, what can you do with this newfound power on a PowerPC-based Macintosh laptop?

Things to Do

The great thing about laptops today is that, for most intents and purposes, they are the equal of desktop machines. You may not want to do heavy-duty video production on a PowerBook using any operating system, but for writing, database work, telecommunications, networking (*especially* networking) and game playing, the PowerBook running Linux is the equal of any machine in its class.

Programming can be a major use of a home-based Linux machine. There is no better platform than Linux for learning to program. UNIXl recently, I wouldn't have said that, because my concept of programming in a UNIX environment

was always command-line-based, and professional programming has long since left the command-line world. With the proliferation of X-based GUI environments like GNOME and KDE, not only can a home-based Linux machine enable you to get a good introduction in modern, object-oriented languages like C++, Java and Tcl/Tk, but you have the opportunity to significantly contribute to the Linux community as your skills develop. Now you can develop those valuable skills on the bus ride home with all the included development packages Linux can put on your PowerBook.

A prime use for a home-based Linux machine is in networking. The Internet was built, and continues to be dominated by, UNIX-based machines. High-speed broadband Internet access has entered the home, and it's here that Linux can play one of its most useful roles. Included in any Linux computer is the ability to act as a masquerading gateway machine, allowing many machines on one side of the gateway machine to effectively share one Internet IP address at the same time. This ability allows you to put all your home machines onto an Internet connection simultaneously.

This use mandates grabbing the source code for the kernel you are running (it's on the CD if you bought the package, and on the Net if you downloaded the package) and recompile the kernel to include IP masquerading. Once that's done, grab the latest version of IP chains from ftp://ftp.linuxppc.org/. This tool will allow you to set up your IP masquerading gateway very simply via scripts. More information on how this might be set up for your machine can be found in the Linux on PowerPC FAQ-O-Matic (see Resources).

Feel a little queasy right now at the thought of compiling your own kernel? Well, especially for something as popular as IP masquerading, someone has already done the work for you. Investigate, via the Linux on PowerPC FAQ-O-Matic, the possibility of using a pre-compiled kernel which has already been set up for the features you need. With IP masquerading set up on an older PowerPC Macintosh (even a PowerBook), your home can enjoy the same networking possibilities, for a lower cost and less with complexity, that large corporations did just a couple of years ago.

Linux machines are better at multitasking their resources than any other operating system on the market today. You have unparalleled control over how much CPU time each application will need on average. I use this ability to run such applications as SETI@home and the RC5 code-cracking client in the background while using my machine to do other things. Both projects have PPC Linux versions available as binaries, or you can build your own client from sources in the RC5 project's case. Here, the creativity of the Linux programming community has shown through most strongly. For both of these projects, Linux users have far more choice and control over the information these clients are

generating than with either Windows or Mac OS. Use your portable Linux power to contribute to the search for ET or to convince the government that we need larger cryptography keys (see Resources).

How about everyday, real-world applications like word processing, databases and spreadsheets? Even though Linux started life as a “geek project”, significant progress is being made in getting office productivity packages onto PPC Linux. Having said that, I also have to report that PPC Linux is behind Intel-based Linux in this regard. The “version” of Linux you're using isn't a problem if you have the program's source code and can build the binaries yourself, but commercial applications such as Corel WordPerfect don't give their source code away for us to build PPC versions of their programs. We have to persuade these companies that there are enough of us out here in the PPC world to care about!

Does this mean we're out in the cold? Of course not! More and more progress is being made in educating companies to jump onto the Linux bandwagon to offer PPC versions of their binaries. And, as is usual with the Linux community, we've taken the bull by the horns ourselves. One of the most exciting “office productivity” projects currently taking place is KOffice, a Microsoft Office-like program suite made to operate under KDE and licensed under the GPL.

KOffice, which is currently designated “alpha software”, consists of several parts. KWord is the suite's word processor. It supports frames, multiple columns, headers, footers, numbering of chapters, auto-correction, spell checking and templates. Import filters that include Word97 are being written. The suite's spreadsheet component is called Kspread. Other components include KPresenter, KIllustrator, KImageShop, Katabase, KChart and KImage. These components each do exactly what you might guess they would. KIllustrator is a vector drawing program, whereas KImageShop is an image processor and KImage is an image viewer.

What makes me excited about KOffice as opposed to other possible productivity software offerings such as Corel WordPerfect or Sun's StarOffice? Quite simply, KOffice is distributed under the GPL. Leaving aside the legal, financial and possible moral issues with regard to the GPL, the fact that KOffice is distributed under that license means more than just being able to read and study the source code. Even though KOffice might be directly marketed to the x86 crowd, a recompile of the sources will make it a wonderful tool for the portable PPC-chip folks. That's an advantage you don't get with Corel or even Sun, as friendly as they have been to the Linux and Macintosh communities. KDE is currently at version 1.1.2, with the much-awaited version 2.0 due out by the time you read this. KOffice is designed to be integrated into KDE as a collection of components. Indeed, components and how KDE uses them are the main difference between KDE V1.1.*x* and the upcoming KDE V2.0.

Linux has continued its great advance into the PowerPC/Macintosh world. Slowly—I think more slowly than most Linux fans would want—the operating system is moving into an arena where it might be generally useful. Still, right now, Linux is a specialized operating system catering to developers, networkers and IT professionals. With the ever-increasing hardware support in both the Intel and PowerPC chip communities, and with GUI desktop support given by such packages as KDE and GNOME, Linux is positioning itself for mainstream big time. Already, this is percolating down to us in the form of equal installations on laptop computers, and attention by big-name publishers and developers in both the Intel and PPC-based worlds. The future continues to look bright.

Resources

The Good, The Bad

Richard Kinne (n2ikr@arrl.net) uses LinuxPPC on his PowerBook to improve the versatility of the machine. He holds the position of Technical Support Representative for The CBORD Group as well as Lecturer in the computer science department of Ithaca College. When not writing or hacking with his significant other, he enjoys Star Trek, *Babylon 5* and playing with his cats.

Archive Index Issue Table of Contents

Advanced search

Yellow Dog Linux on the iMac

Stew Benedict

Issue #76, August 2000

A guide to installing and running YDL on a power PC.

First, a disclaimer: I'm not a fanatic Macintosh guy. I got a Macintosh late in my computing career, just last year in fact—an iMac. I was working on a cross-platform Tcl/Tk project, and I was getting pretty involved in fine-tuning the GUI for Mac OS, Windows and Linux. I decided it was too much to rely on feedback from the users, so I opted to buy a cheap Mac on which to test the application. I ended up with one of the Revision B iMacs, which is a decent machine, and my wife likes it too.

Well, wouldn't you know it: I soon discovered that folks were running Linux on these things, so I just *had* to try that out. Linux on a Mac has now come of age, with about half-dozen vendors offering distributions for PowerPC processors.

Yellow Dog Linux is one of these vendors, and they offer a number of packages targeted at different levels of usage. Champion Server is just like the name sounds: a package targeted for server applications, but it also works well as an individual workstation. Yellow Dog also has a Gone Home edition in the works, originally targeted for home users, but now their vision has changed and Yellow Dog says Gone Home will be “a revolution in the way Linux is installed and experienced”.

The Package

Yellow Dog Linux put much thought into the packaging of their product, offering anything from plain CDs to binders to a full nylon zippered notebook sporting the Yellow Dog “Labrador” logo. The package *Linux Journal* gave me to evaluate was the zippered notebook (see Figure 1), which also included some Yellow Dog bumper stickers to help you show the world you're an avid Yellow Dog fan. Both bound versions have the CDs in plastic jackets in the notebook and a nicely done manual. If you opt to go the download route, YDL offers

image files of the install CDs on their web site and mirrors. (Do yourself a favor, though, and buy a CD, or go in with a group to buy a set and burn your own copies. It's not that much money ($24.95 US), and you won't be consuming the bandwidth for that 700MB download.) There are three CDs total: the “Install”, “Source” and a “Tasty Morsels” CD. Tasty Morsels contains additional games and applications, including some that do not fall under the GNU license. It also contains KDE2 and the Linux 2.3.49 source tree. Interesting to me was **pcb**, a basic printed circuit-board layout program, because I am a printed circuit-board designer.

Figure 1. Yellow Dog Package

Installing Yellow Dog Linux

Champion Server, like many distributions these days, is based on Red Hat Linux, using RPMs and the text-based Red Hat installer. To begin the install, insert the install CD in your CD drive with the power off, then power up the system holding down the “C” key, and you will be presented with a **yaboot** prompt. **yaboot** is a boot loader along the lines of the familiar LILO on x86 machines. At the prompt, type **install** and soon you will see the ever-present Tux and the Red Hat text-mode installer. There is no mouse at this point, so use the arrow keys, **TAB** and space bar to select and acknowledge the prompts. You are asked to select a language and keyboard type, which defaulted to English and mac-us-std, respectively, for me. You are then prompted for the installation media (CD-ROM for me).

The iMac uses USB (universal serial bus) for all its peripherals, including the keyboard, so the first thing you need to do is identify the keyboard and language so you'll be able to use the keyboard during the install. I had a problem with this—my keyboard worked at the yaboot level, but not in the Red Hat installer. After some trial and error, I found that putting my Belkin Hub on the USB chain was the root of the problem, and my keyboard worked after removing the hub. (USB is still a little shaky, in my opinion. Half the time, my mouse does not work in Mac OS until I unplug/replug it in.)

The first thing you must do is set up your partition table. The installer takes you into this step, offering to use either **disk-druid** or **pdisk**. I believe pdisk is the only true option for a Mac OS-partitioned drive, so this is your best bet. **pdisk** is

much like **fdisk** for Intel Linux, a text-based program where you use letter commands to create and name your partitions of various types. I opted to create a single Linux partition of about 1.4GB and a 20MB swap partition, because I had another working Champion Server setup that I wasn't ready to sacrifice just yet. In many cases, it's a good idea to create separate partitions for /, /var, /usr and /home. If you want to upgrade later, you can then opt to leave /home alone and reformat the other partitions and still retain your personal files and settings. It's important to remember to reboot after setting up and writing the partition table to ensure the changes take effect. I've seen a number of people encounter problems (not just with Champion Server) when they did not reboot. The install appears to go fine, although very quickly; yet upon booting, they find many things missing or not functioning properly. (It would be nice if the installer gave you a graceful way to reboot, rather than having to press **CRTL-OPT-POWER**. It would also be nice if, after the reboot, the installer could resume where it left off.) One thing I should mention is that the prompts show fdisk, but you are actually using pdisk. If you have already set up Linux partitions, as I did, you can skip the reboot.

I should also mention most Linux vendors recommend you do a clean install, not an upgrade to your existing install. If you have a separate /home partition where you keep your personal files, they will be kept during the install, provided you don't format that partition. Many people have a lot of problems trying to upgrade a Linux system. The safe bet is to make a full backup and start clean, then pull the things you need from your backup. You may find you don't miss much from your old installation.

The installer gives you a default set of packages to install that is fairly complete, and you can fine-tune this by category and individual package. If you attempt to install a package that depends on another, the installer will warn you and offer to install the necessary packages. I hand-picked what I installed, giving myself enough to get a feel for what the distribution had to offer, but omitting some pieces due to my limited partition size.

The installer takes a few minutes to install the packages, then detects your mouse, letting you fine-tune the selection, and then offers to set up networking. I have a network setup at home with my primary box running Linux serving as a server/Internet gateway for the rest of the network. I assigned a fixed IP address to the iMac, the same as used in Mac OS, and set up the main box as the gateway, with DNS addresses for my ISP. The installer then asks you to choose your time zone, which it also correctly detected as US/Eastern for me. You are also given a list of dæmons/processes to start automatically at boot, and you can enable or disable them as you see fit. (These are programs such as Apache, Sendmail and NFS.)

Next, you are asked if you would like to configure printing. Again, my printers are all networked and power-controlled by X10 controllers from the server. I opted to pass print jobs through this queue.

You have the opportunity at this point of setting the root password and choosing whether to use shadow passwords. You are then instructed on which partition you should enter in BootX, and the installer goes on to identify your video card.

The proper X server is configured. A message tells you to start X on your new system using **startx**, and if you have problems, to run **Xconfigurator**.

You are then congratulated on your install, and the system reboots.

Running Yellow Dog Linux

Running Linux on a Macintosh is a little different than with Intel. On the Intel side, most folks use LILO. On the Mac, the boot loader of choice used to be BootX, but now yaboot is also available.

BootX runs under MacOS and can be set up as an extension, so like LILO, you are offered the choice of booting MacOS or Linux. Unlike LILO, you are already well into the MacOS boot-up GUI before you get the opportunity to choose Linux, although you do have a nifty graphical screen, allowing you to enter some options that control the boot process (see Figure 2).

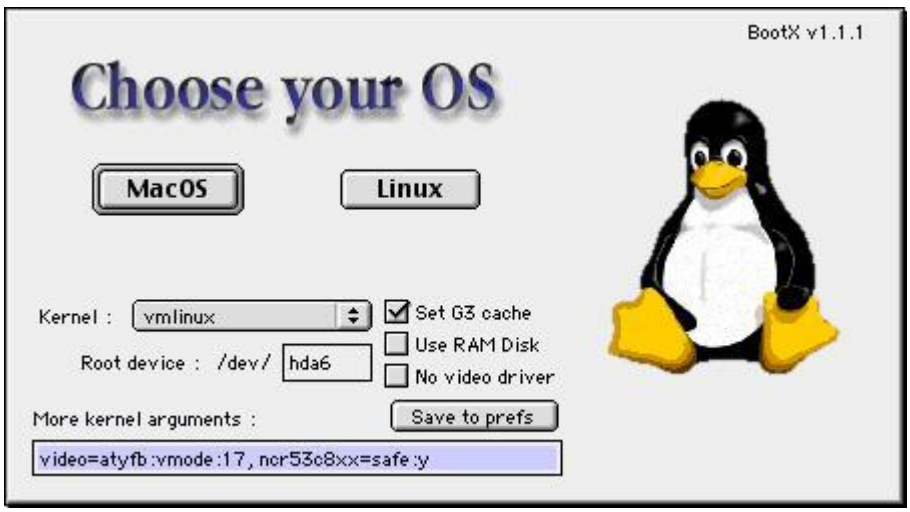


Figure 2. BootX Screen

yaboot works at a lower level, more like LILO, giving you a boot prompt at power-up. The Champion Server documentation steps you through the setup for both of these programs, with special notes as to options for the various Mac hardware platforms.

The Documentation

While I'm on the subject of documentation, let me take a moment to comment on the Yellow Dog manual. Although only a loose-leaf three-hole package, it is very well done with clear instructions covering the full install. As mentioned above, it also lists special considerations for the various Mac/PPC hardware platforms. They also have a FAQ area on their web site (see Figure 4) and the usual mailing lists, monitored by YDL staff. Questions are answered fairly quickly.

Figure 4. YDL Web Site

Booting YDL for the First Time

I started using Linux on the PPC with another distribution, and had considerable trouble getting basic things like the keyboard, video and mouse, which I had taken for granted on Intel, to work. Setting up X can sometimes be a true challenge on the PPC, and since the Mac has no text mode, you can end up in a situation where you don't even have a visible command prompt. YDL wisely opted to boot the system in run level 3 with only the command prompt, so you can set up and test your X configuration before committing to booting straight into X with run level 5.

I already had a BootX setup from my old install and chose to keep it. Older iMacs have an ATI Mach64 video card, and Yellow Dog recommends adding the

stanza **video=atyfb:vmode:17** to your kernel arguments and unchecking the “No Video Driver” option. I also have a non-standard iProRaidTV SCSI card/TV tuner installed in the mezzanine slot of my Revision B iMac, so I have an additional **ncr53c8xx=safe:y** option to force the card to wait for the SCSI bus to settle during the boot process. This also disables some of the advanced features of the card, but it does work—usually.

For BootX, take a copy of the Linux kernel (vmlinux—PPC does not use compressed kernel images) and put it in your system folder or the kernel folder under your BootX install. I generally use the system folder along with the BootX extension, so I can go right into Linux early in the Mac OS boot process.

In the Dog House

I ran the BootX program, checked my kernel parameters and pressed the “Linux” button. The Mac OS screen changed to a black screen with the ever-present Tux in the upper-right-hand corner, and I observed the normal stream of boot messages identifying and configuring my hardware—until it hit the iProRaid card. At this point, I ended up in a somewhat familiar endless loop of SCSI resets and was forced to power down the machine to get out. YDL apparently uses the sym53c8xx SCSI driver in their stock kernel, and it did not recognize my “safe” kernel parameter. (Both of these drivers are very similar, as I believe Symbios bought the old NCR chipset technology. The NCR/Symbios cards are very popular and inexpensive; I've got three of them in various Intel machines.) I tried replacing **ncr** with **sym** in my kernel arguments, as well as **sym53c875=0**, but nothing seemed to stop that endless SCSI bus reset.

But all was not lost. I've got my own home-compiled 2.2.14 kernel with the NCR driver, as well as modules for the video portion of the card, so I used it instead, and soon I was up and running, looking at the root login prompt. Although an annoyance for me, I do realize my machine is non-standard, and if it weren't for the iProRaid card, I would have been up and running much sooner. What would be nice is being offered both a monolithic kernel and a modular one, with a choice of modules to install.

X Marks the Spot

First, I tried startx with the default settings, but all I got was my monitor in standby mode and had to **CTRL-COMMAND-BACKSPACE** to get out.

In the old days of Linux, you had to either edit your XF86Config file by hand or find someone on the Net with the same video card as you who had already done so. Today, some nice tools can get you set up in X fairly quickly. **Xautoconfig** runs from the command line, and oddly enough, silently creates an appropriate XF86Config file based on your hardware. (If it seems like nothing

happens, it worked. If you get a message of any sort, it probably didn't.) **Xconfigurator** is a text menu-based configuration tool, and the choices included the integrated video card and monitor of the iMac, as well as an appropriate choice for my three-button USB wheel mouse. I chose a 1024x768x16bpp setup, as I knew from previous experience that 32bpp has some issues in the latest **fbdev** X driver.

Now for the defining moment. I type **startx -- -bpp 16** and I get the blue desktop and the X cursor, then back to the command prompt. I try again with **bpp 15**, and I go straight into X with a default KDE desktop. There are some snappy YDL links on the desktop leading to the YDL web site for documentation and updates. Another minor annoyance is the “Home” link generates an error that says “config file has no `Type=...' line”. The “Support” link does launch Netscape and triggers my demand-dialing connection on the server and takes me to Yellow Dog's support page—yay, networking works! (See Figure 3.)

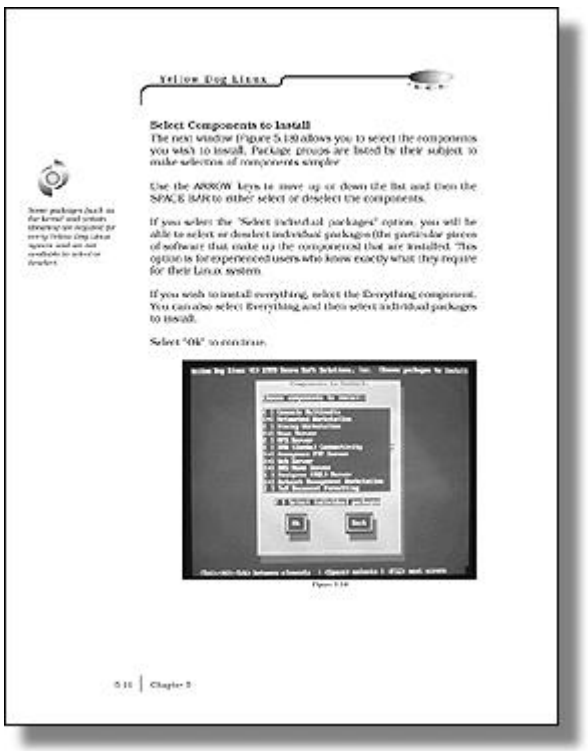


Figure 3. YDL Support Page

Window motion is smooth, and no artifacts are generated while dragging windows on the desktop. It appears I have accelerated video. I pulled up the KDE control panel and played around a bit with the various settings. Things look typical here, with the requisite lists of themes, colors and window manager options. The included X server has support for DPMS, which has been wanting on the iMac. It seemed to work fine in testing. I tried enabling the system sounds in the control panel, but all I got is one “eep”, then nothing until I restarted X. (I also knew in advance that sound has been an ongoing problem

for Mac users. My previous install has working sound only when I go right into Linux at boot. If I do something in Mac OS first, no sound.)

My reading on the mailing lists indicates that **esd** is the best working sound server for PPC, so I launch “switchdesk”, change my desktop environment to GNOME and restart X. GNOME now comes configured to use sawmill, rather than Enlightenment, but there is an option in the GNOME Toolbox to change to Enlightenment. (Watch it here, because it says Enlightenment is the current window manager. I actually had to select sawmill, save the changes, then select Enlightenment to make it take. See Figure 5.) I do this, enable system sounds and set up my preferred desktop theme, colors, etc., then shut down and restart X. I've got system sounds and am greeted by a female voice welcoming me to Enlightenment (see Figure 6). I pop in my favorite David Torn CD and run the GNOME CD player applet, and I'm one happy camper. One annoyance is that restarting X seems to place the blue desktop background over my chosen pattern until I reset it. I found I had to go into GNOME's settings rather than Enlightenment's to turn the blue background off.

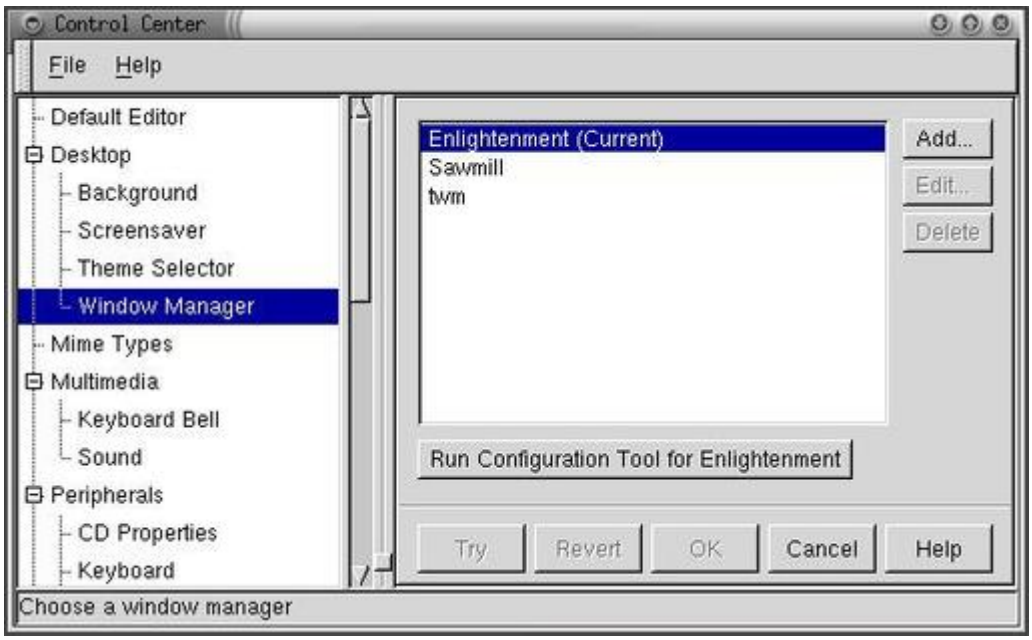


Figure 5. GNOME Control Center

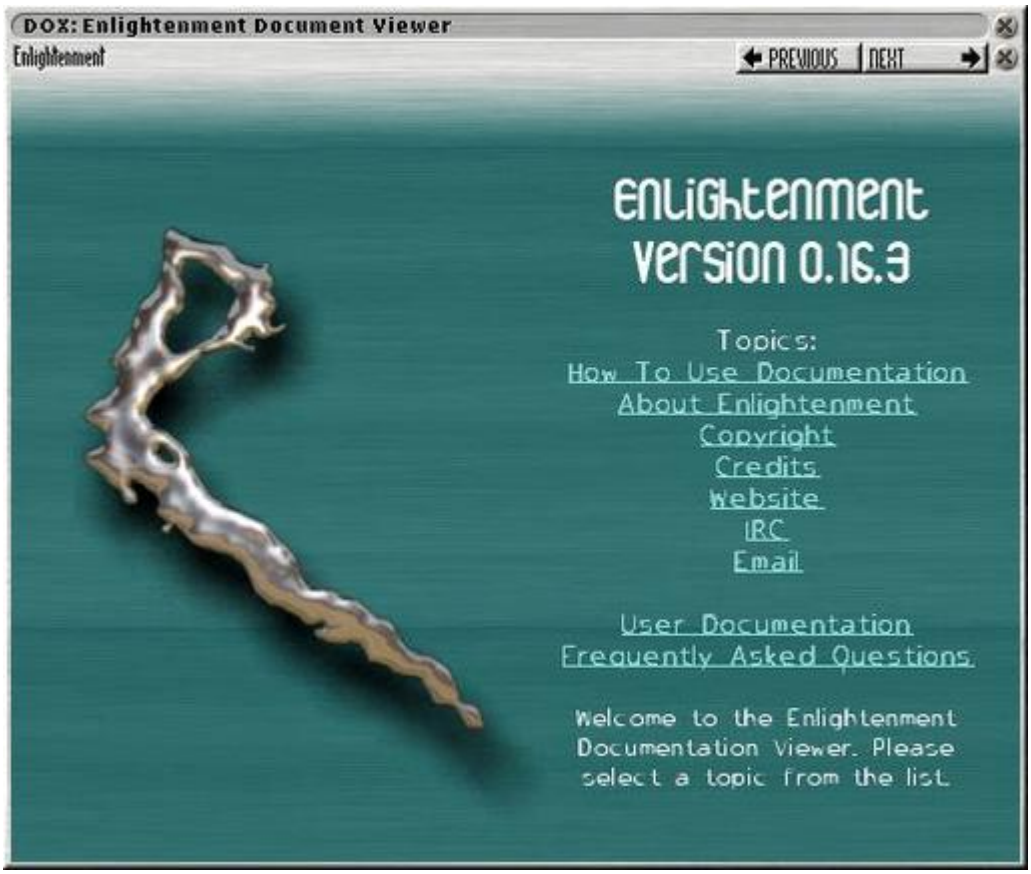


Figure 6. Enlightenment Screen

Network/Internet

The install process asked me about my local Ethernet configuration, so I assigned the domain name, machine name and fixed IP address I use for this machine. After rebooting, my Ethernet connection was fine, and I edited the /etc/hosts file to identify the other machines on my network by name. I could use TELNET, rsh, FTP, etc. to get to my main server. By default, incoming TELNET and FTP are disabled, which is a good thing, especially if you are going to be connected to the Internet.

For PPP, I used the Red Hat Internet Configuration tool, assigning the proper phone number, domain name and DNS information for my ISP. I was able to connect on the first try. This area of Linux has definitely improved for the first-time user. No more editing PPP configuration files and dial-up scripts by hand. The usual assortment of Internet applications are installed including Pine, Netscape, **slrn** (this failed, with a library error), Kmail, **xchat**, archie and ncftp (see Figures 7 and 8).

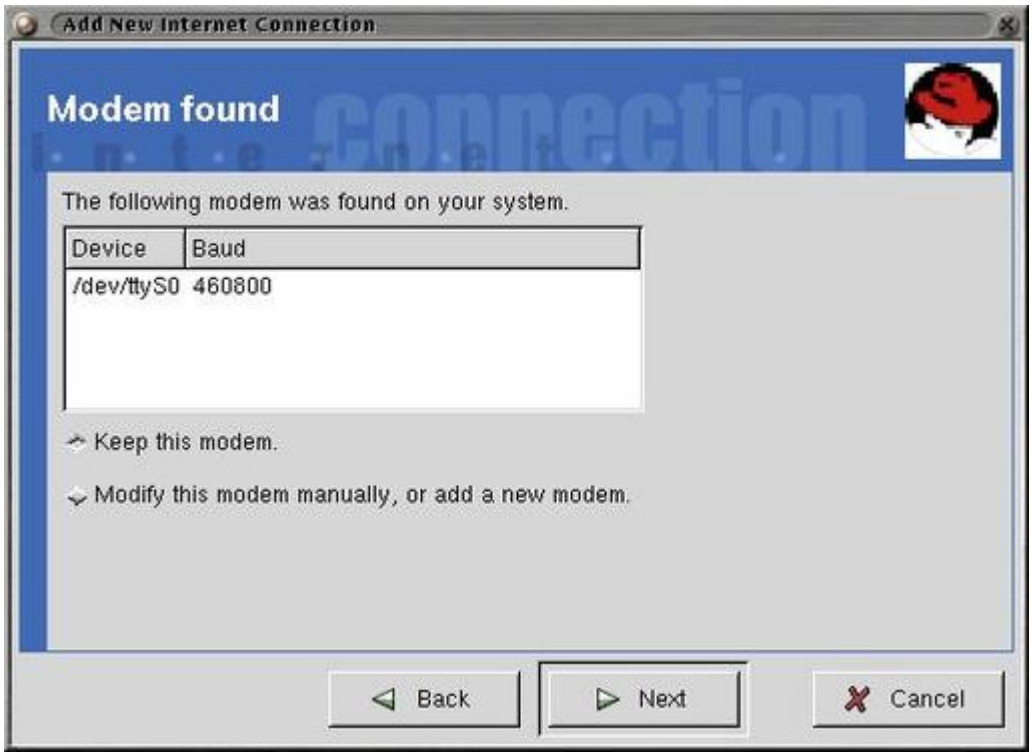


Figure 7

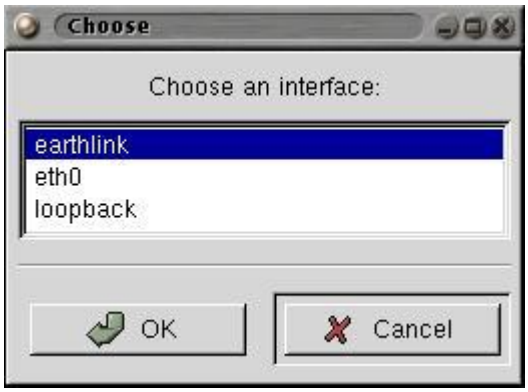


Figure 8

Printing

I don't have a printer hard-wired to the Macintosh. Since I have so many machines, I've opted to move my printers to a print server that hangs on the network. The **printtool** setup gives options for setting up the printers, and an option is included for a remote host. I have two printers, an NEC Silentwriter95 (PostScript) and an HP DeskJet 693C. PostScript and ACSII printing worked fine on both.

Applications

YDL comes with the standard set of Linux applications. For the office worker or student, you have Abiword and Gnumeric for your word processor/spreadsheet combination. Not as full-featured as some of the commercial applications, but certainly capable of creating and printing a basic document. I did get an error

related to **gnome-print** when running Gnumeric, but after re-installing the gnome-print RPM, the problem went away. Both applications printed fine. The GIMP is included for manipulating graphics files, as well as Electric Eyes. A number of games are also included. One I found particularly entertaining is Xboing, a spin on the old “Breakout” game with some slick effects and sound (unfortunately, the sound was a bit distorted on the iMac).

MOL

MOL stands for Mac-On-Linux and it is a wonderful contribution to PPC Linux, authored by Samuel Rydh. It can be run as a window in X or in console mode using fbdev (the frame buffer), and it allows you to use your existing Mac OS partition and applications from within Linux. Sound doesn't work, but you can set up a networking interface to access files and the Internet from MOL. I've used MOL before on my CS1.1 installation, and it worked quite well. For whatever reason, MOL on CS1.2 starts up but never finishes the boot process. I get a number of messages referring to an unsupported resource in the console from which it is launched. Thinking it may be an issue with the 2.2.14 kernel I'm using, I uninstalled the stock MOL and grabbed my copy from the other partition. It also failed, but with a different error. Looks like this one needs a little work. Again, it's most likely due to my being unable to use YDL's kernel (see Figure 9).

Figure 9. Mac-on-Linux

YUP

YUP stands for Yellow Dog Update Tool. It takes the RPM concept to the next level, querying an external network of databases to download and upgrade your software packages on demand, including itself, when necessary. From what I read on the mailing lists, there were some initial problems with YUP and it went through several rapid revisions, but in concept, it sounds like a neat tool. If you run **yup update**, it will go out and find the newest versions of your software on the Internet and update things for you. I prefer to control when and what I update, but for inexperienced users, this could be a nice option.

Summary

Linux, and in particular Linux on PPC, has come a long way. Yellow Dog Linux Champion Server 1.2 is a very good package. I did hit a couple of rough spots that may have stumped a first-time user, but fortunately, I have the experience to work around them. Were it not for my non-standard SCSI card in the iMac, things may have gone much smoother. In the end I had a very usable Linux system, not unlike what I use on x86 boxes (see Figure 10). I have used LinuxPPC and had even more problems with it before I went to Yellow Dog; however, that may have changed with LinuxPPC's newest release. Many resources are available on the Net, including mailing lists from both distributions as well as the mailing list/web site that I help maintain: http://www.imaclinux.net/. If you've got a Mac and would like to give Linux a try, now is the time. With MOL, you can now have the best of both worlds! If you do decide to give Linux on PPC a try and get stuck, I'd be happy to help where I can; drop me a line.

Figure 10. YDL GNOME Desktop

Resources

Stew Benedict (stewb@worldshare.net) is a Systems Administrator for a manufacturing plant in Cleveland, Ohio. When not sitting with his eyes glued to a CRT, he enjoys spending time with his wife, daughter and dogs, and dreams of early retirement to a lakeside home in the mountains.

Archive Index Issue Table of Contents

Advanced search

Linux Finance Programs Review

Ralph Krause

Issue #76, August 2000

We continually hear “I would only use Linux except I need to ...”. One of those missing pieces has been to run Quicken. Find out what programs are available for Linux that offer an alternative to Quicken.

As Linux finds its way onto more and more computers, the need for a broader range of applications increases. One popular request is a Quicken-like program for managing finances. There are several financial programs available for use on Linux, and I present some of them in this article. The programs covered are cbb, QHacc, GnuCash, Moneydance and gAcc.

All of these programs can keep track of basic financial information such as checking and savings accounts. Almost all of them also offer some type of reporting and graphing functions. Most of them can import files saved in the QIF format with varying degrees of success.

Most of these programs are not up to version 1.0 yet, and lack some features and polish. The authors are busy improving them and adding functionality, so, while there is not an exact replacement for Quicken or Money, you might find one of these programs to be suitable for you.

cbb

I installed cbb version 0.8.1 for this article. Written entirely in Tcl/Tk and Perl, installation was accomplished by simply typing **make install**. **cbb** contains no compiled code, so it should run on the majority of Linux systems.

cbb keeps account information in individual files, one **.cbb** file for each account. A list of accounts and their current balances is displayed at the bottom of the screen, and the account's entries are shown when you select it (see Figure 1).

Figure 1. cbb

Entering transactions in cbb is straightforward. The transaction fields are between the ledger and the account list. You enter the particulars of a transaction, such as date, amount and category, then press **ENTER** or click on the “Accept” button to add it to the ledger. **cbb** allows you to easily change the current check number via the “+” and “-” keys, and includes a tab-completion feature for the payee and category fields.

Data entry is made simpler by the use of keyboard shortcuts, the ability to tab through all the fields and the ability to accept a transaction simply by pressing the **ENTER** key. If you enter a lot of transactions, you will find these features very useful.

While you can use categories to group your transactions, cbb does not allow you to specify whether a category is an expense or an income, and you cannot create sub-categories. A workaround for sub-categories is to use a naming convention such as Utilities, Utilities-electric and Utilities-gas. **cbb** can display

the category list while editing transactions so that you can select the category you want.

cbb offers the following reports: transactions, transaction list by category, uncleared transactions, missing checks, average monthly expenses by category and transactions listed by payee. Graphs are available showing running balances, movements by categories, expenses by categories, expenses pie chart and categories by description pie chart. You can specify beginning and ending dates for reports and graphs and select the accounts to include.

Account reconciliation is accomplished by clicking on the “balance” button. Before doing this, however, you must enter any interest or service charges. When the balance window is displayed, enter the beginning and ending balances from your bank statement and start clearing transactions. If you notice a mistake, you can fix it in the ledger without closing the reconciliation window.

cbb allows you to import QIF files, so that you are able to use your existing data. Unfortunately, the import function did not work correctly on the QIF file I created in Quicken 99. While I did get transactions, there were no dates for them, and I could not edit them after I imported them. There were also some lines at the beginning of my ledger that contained data cbb apparently could not decode from the QIF file.

cbb is a good program for keeping track of checking and savings accounts. It allows you to enter transactions quickly and easily with the use of keyboard shortcuts. The reports and graphs provide useful information, and the ease of installation is a real bonus.

Moneydance

Moneydance is a shareware program written in Java. Of all the programs reviewed here, it is the most Quicken-like in appearance (see Figure 2), and it did the best job of importing my QIF files. I used Moneydance 2.0.4.

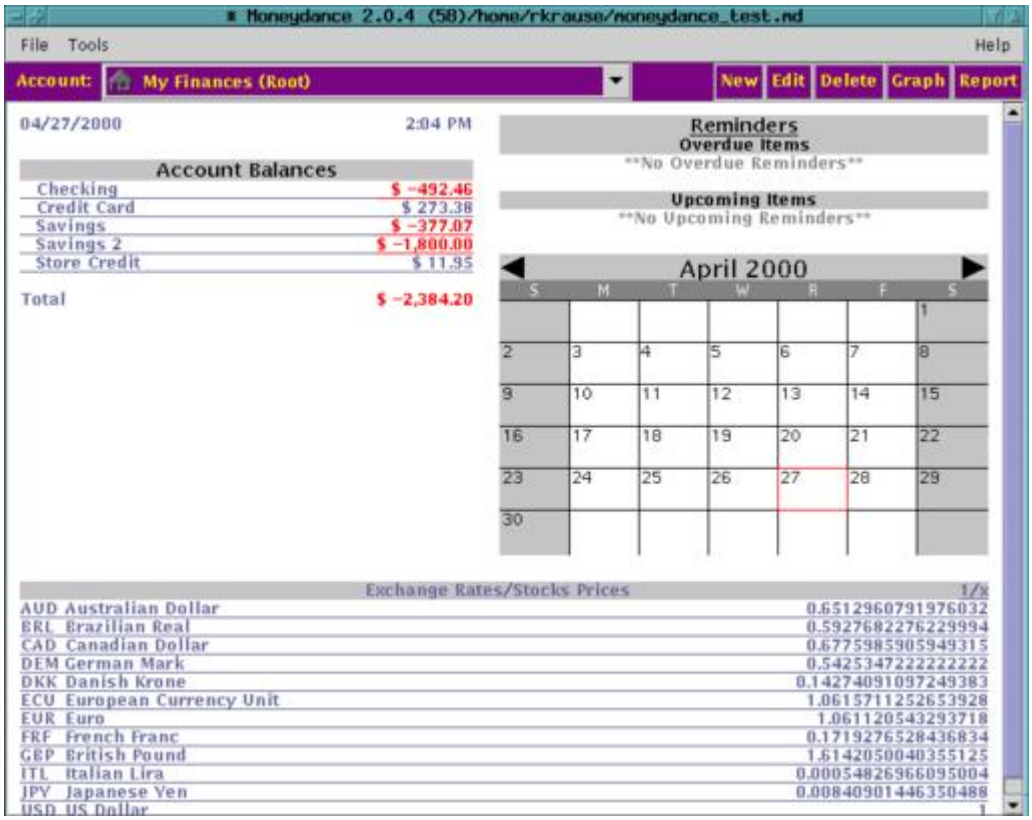


Figure 2. Moneydance

Installing Moneydance is very easy: simply unpack the tar package and modify the Moneydance file so that it points to the required .jar files. Then, place it where it can be found in your path (e.g., /usr/local/bin) and you are ready to go. If you are using an older Linux distribution, you might need to upgrade your Java environment.

Each transaction in Moneydance must contain a source account and a destination account. This is called double-entry accounting. Each type of income (e.g., salary) and expense (e.g., mortgage, fuel, credit card) appears on the chart of accounts, and all transactions are really transfers between accounts. For example, when you pay the electric bill, you transfer the money from your checking account to your Utilities:Electric account.

Moneydance provides a useful auto-completion feature which completes the payee name and displays the amount and accounts that you previously entered for that payee. Navigation through the ledger fields is easily done using the **TAB** key. When entering check numbers, you can either type the number or use the mouse to select actions such as “Next Check #” or “Print”. When you are done with the transaction, save it to the ledger by pressing **CTRL-R** or by clicking on the “Record” button. If you have to enter many transactions, you might find it a bit tedious to use the mouse for the check numbers plus two keys or the mouse to record each transaction.

Account reconciliation is done by selecting “Reconcile” from the Actions menu. You can add, edit and delete transactions from the reconciliation window.

Moneydance provides both reports and graphs. There are reports for transactions, account balances, income and expenses. You can customize a report by selecting beginning and ending dates, a specific account and the method of sub-totaling. Printed reports look sharp and contain titles and date ranges across the top of the page. You can graph the following information: Income, Expenses, Income and Expenses, and Account Balances. I could not get the graphs to print on my system for some reason.

You can print checks from Moneydance by selecting “Print” for the check number, then selecting “Print Checks” from the Actions menu. Moneydance supports standard checks and checks with stubs and allows you to print test checks to calibrate your system.

Another nice feature is the ability to search for transactions. Selecting “Find Transaction” from the Actions menu or pressing **CTRL-F** brings up the Find window. You can search for a specific Payee/Description, Memo or Check #. Selecting “Find Again” or pressing **CTRL-G** allows you to find the next matching transaction.

If you are used to Quicken, you will find Moneydance very easy to use. The only problems I found with it were minor ones. The help file does not provide much information, but it does cover basic account editing and double-entry accounting. On my machine, Moneydance ran somewhat sluggishly, but I think this is due to the fact that I am running an older version of Java on a slow Pentium.

QHacc

The next program is QHacc version 0.4.3. (Note: QHacc's authors have released version 0.5, but too late for me to include in this article.) This program requires the QT toolkit in order to work, which I found to be quite an easy installation process. I compiled the QT toolkit from sources and followed the instructions for installing it, then I compiled and installed QHacc without incident.

QHacc provides a simple two-paned layout. The left pane contains a list of accounts and balances, and the right pane contains the ledger for the selected account (see Figure 3).

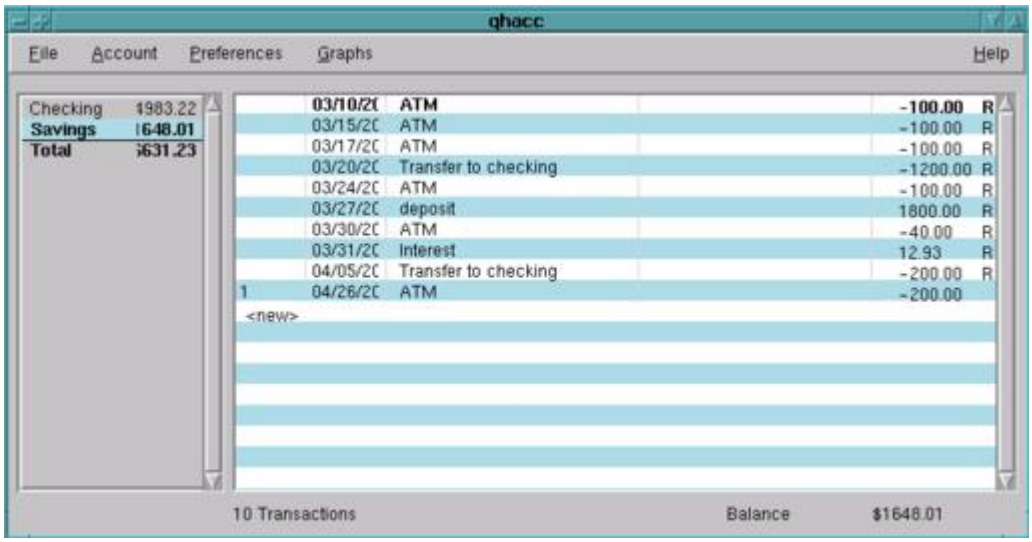


Figure 3. QHacc

Transaction entry is a little different from the other programs. Selecting “new” in the ledger brings up a transaction window where you enter the check number, date, payee, amount and memo. QHacc inserts the next available check number and provides an auto-complete feature for the payee. The transaction is entered into the ledger by pressing the **ENTER** key, while pressing the **ESC** key cancels it. Withdrawals must be preceded by a minus sign, because QHacc does not provide separate credit and debit text boxes.

QHacc also provides a mechanism for memorizing transactions. After entering the transaction that you want to memorize, right-click on it in the ledger and select “Memorize” from the pop-up menu. To insert a memorized transaction, right-click on an empty ledger line, go to the memorized item in the pop-up menu and select the transaction you wish to insert.

QHacc can be set up for single- or double-entry bookkeeping. If you want to use categories for keeping track of your transactions, you must use double-entry bookkeeping. You must also use double-entry accounting to automatically update account balances when transferring money between them, otherwise you have to enter the transfer in both accounts. If you elect to use double-entry bookkeeping, you can also split a transaction among several accounts.

QHacc provides a simple graphing function that shows the net total of transactions by the week. According to the companion TODO document, more graphs will be added in the future.

Account reconciliation is the same as in the other programs. Remember to enter any interest payments or service charges before using it. Enter the starting and ending balances from your bank statement, then select entries to clear.

At version 0.4.3, QHacc is the youngest of the programs at which I looked. I did find one problem. If I entered 00 for the year, it used 1900. Also, QHacc does not offer the ability to import QIF files.

GnuCash

GnuCash is the most ambitious financial program being developed at this time. It offers the greatest variety of account types, sub-accounts and stock price retrieval. This program was the most difficult to compile and get working because it depends on quite a few other programs, libraries and Perl modules. I looked at both the stable version (1.2.5) and the current development version (1.3.6).

Before attempting to use either of these versions, read the documentation closely to determine which additional programs, libraries and Perl modules you will require. Version 1.2.5 requires Motif or LessTif and version 1.3.6 uses GNOME and the GTK. I had better luck installing them on a Red Hat 6.1 system than I did on a SuSE 6.1 system.

GnuCash offers a slightly different interface than the other programs I tested. Its main window displays a list of accounts with balances, and a new ledger window is opened for each account. This allows you to view and edit more than one account at the same time (see Figure 4).

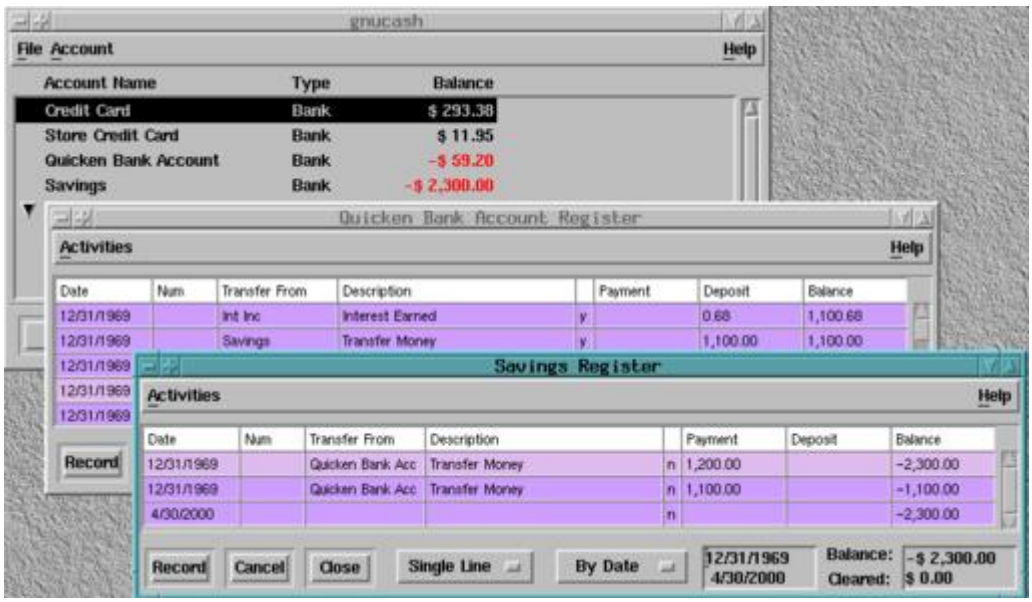


Figure 4. GnuCash: Multi-Account View and Edit

GnuCash offers more types of accounts than the other programs (see Figure 5). An account can be identified as a bank account, cash, asset, credit card, stock, liability, mutual fund, currency, income, expense or equity. Accounts can be children of other accounts, allowing you to create portfolios of funds. The

ledger windows change slightly depending upon the type of account you are working with.

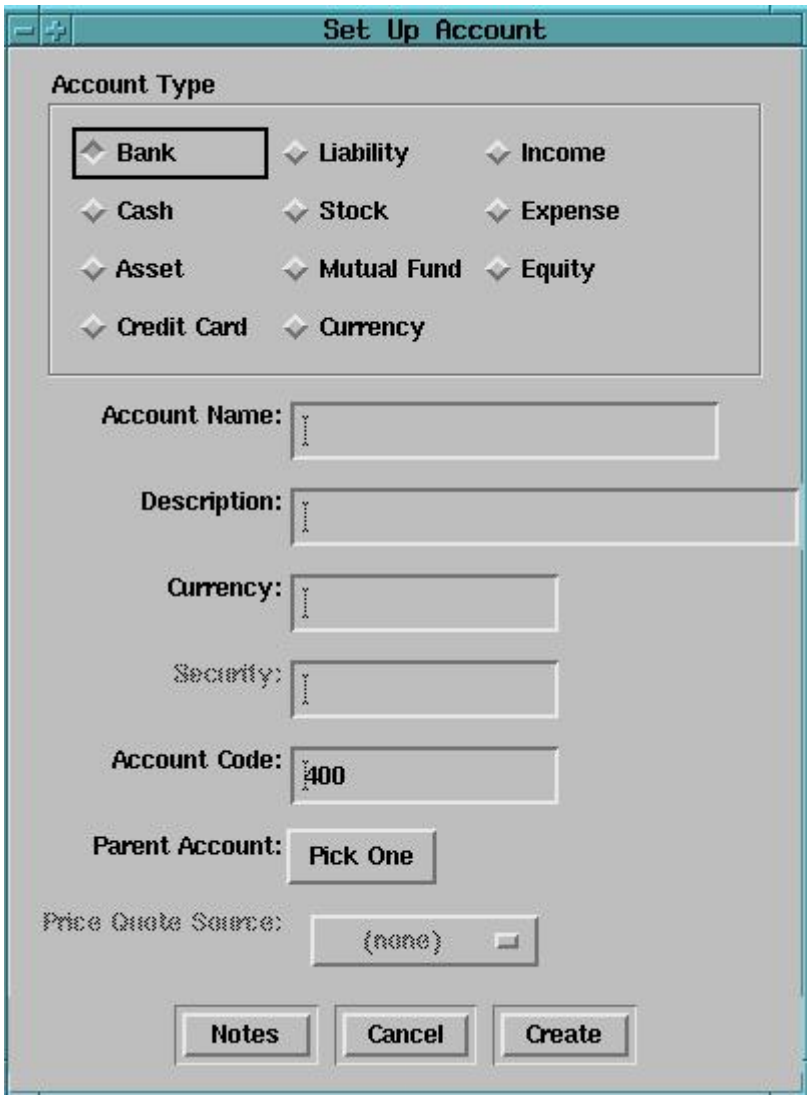


Figure 5. GnuCash: Account Types

The ledger windows offer you the choice of displaying single or multi-line entries and allowing you to sort transactions by date, check number, transaction amount, memo or description. Unfortunately, they do not remember your display selections after you close them.

Keyboard entry leaves a bit to be desired in version 1.2.5. While you can use the **TAB** key to move through the fields, in single-line mode you cannot tab over to the payment or deposit fields because the focus jumps from the account field to the “Record” button. Version 1.3.6 puts the command buttons above the ledger, fixes the tab movement function and accepts a transaction when you press the **ENTER** key. Neither version automatically increments check numbers in the ledger window.

I had some problems importing my QIF file from Quicken 99. GnuCash version 1.2.5 read my Quicken file and did a good job of creating my chart of accounts, but all the ledger entries from the QIF file had a date of 12/31/1969. Version 1.3.6 could not even read my QIF file, quitting with the message “wrong argument in position 1”.

GnuCash offers reports but no graphs at this time. The reports included in version 1.2.5 are Balance Sheet, Profit and Loss and Portfolio Valuation. Version 1.3.6 offers these reports plus additional ones, such as a budget report, but there is no way to create a budget from within GnuCash at this time.

While certainly the most ambitious program of the group, GnuCash was also the most difficult to install. The documentation does state which programs and libraries are required by GnuCash, and I had better luck installing GnuCash on a Red Hat 6.1 system than a SuSE 6.1 system. The dependence on so many external programs and the difficulty of importing QIF files are the main problems with GnuCash.

gAcc

The final program I looked at was gAcc version 0.7.1. gAcc requires the GTK+ toolkit, version 1.2.2 or later, to be installed. I found it relatively easy to compile the sources and install this program, but I did have to copy a few files manually before it would run.

gAcc's layout is similar to that of GnuCash, with a main account list (see Figure 6) and separate ledger windows, but you can work with only one account at a time. gAcc is from Europe and offers better support for European dates and currency than the other programs I tested. You can set these options from the “Preferences” dialogs.

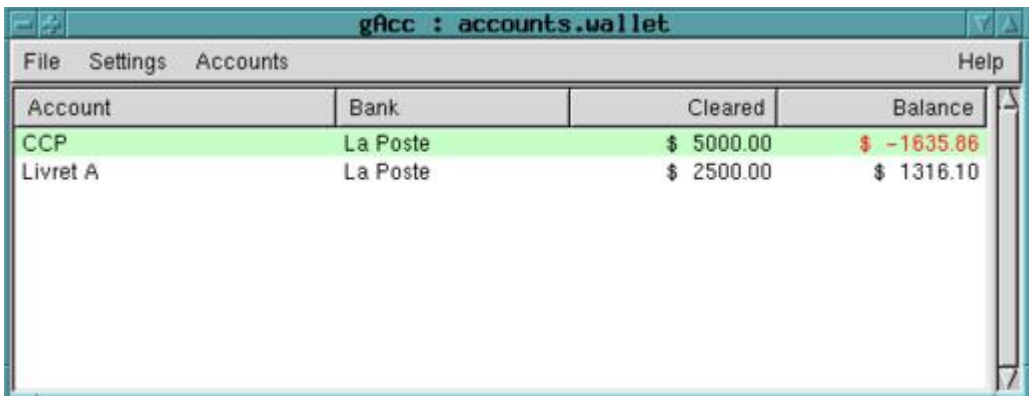


Figure 6. gAcc

Entering and editing transactions is not as user friendly in gAcc as it is in the other programs. There is no keyboard shortcut for adding new entries and no auto-complete feature for the payee. Categories, actions and accounts are

available from a drop-down list so you do not have to remember them. One quirk with the transaction entry is you have to delete the “$” in the amount text box or $0 shows in the ledger, no matter what you enter.

gAcc handles only checking and savings accounts. You can create categories to keep track of your transactions. At this time, it does not offer any reports or graphs, but these features are listed in the TODO file to be added in later versions. There is also no ability to import QIF files.

I originally tried version 0.7 and ran into a few problems with the source tarfiles and the preferences dialog. I sent a note to the authors and received a prompt reply from them. Shortly after that, version 0.7.1 was released and the problems were fixed. This was a very pleasant experience and a tribute to the authors.

Conclusion

I found cbb a very simple program to install and use, but it is limited to simple finances. Moneydance is the most Quicken-like in appearance and it also did the best job of reading my QIF files, but it ran slowly on my system. QHacc offers the ability to work with checking and savings accounts without much overhead. GnuCash is the most difficult to install, but supports the greatest variety of accounts. gAcc handles European dates and currency symbols natively, but can also be set up for American symbols.

There is certainly no lack of financial programs available for Linux at this time. If you have simple finances and do not mind entering data by hand, one of these packages should work for you. If your financial needs are complex or you want to move away from an existing Quicken system, you have a more difficult choice. At this time, there is no exact Linux replacement for Quicken, but I am sure this is only temporary as these programs continue to mature and improve.

Resources

email: rkrause@netperson.net

Ralph Krause (rkrause@netperson.net) lives in southeastern Michigan and divides his time among computers, reading and taking care of four dogs. He has been using Linux for over a year.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Writing a Java Class to Manage RPM Package Content

Jean-Yves Mengant

Issue #76, August 2000

A look inside RPM packages and how to use Java to extract information.

Installing a Linux system for the first time is quite straightforward. You can find good and cheap Linux packages on the market that will quickly install themselves. You need only choose some configuration options describing the type of hardware on which you want to install, and that's it.

As time goes on, you will add some new components to your Linux system, and that's where the nightmare may begin. A Linux system is composed of hundreds of components and dynamic libraries. In order to keep your system up and running, you should be careful, since installing a new version of a component may introduce incompatibilities in your system, making it more unstable as time goes on.

Replacing an existing component with a new version at first looks like a trivial task: you just need to pick up a new compiled version of the component, generally available on the Net in a .tgz file (tar + gzip compression), and install it on your system. Some dynamic libraries of the component you just upgraded may already be used by other installed components and might not be compatible with the new version you just installed.

It would be great to have a tool which could report the dependencies of each component installed on your system. Such a tool could tell you the version of Samba (for instance) installed on your system or that you can't install egcs-1.0.2-8 on your system prior to having binutils 2.9 up and running. This tool already exists on Linux—it is called RPM—and is on many existing Linux distributions, including Red Hat, Caldera, SuSE and Linux Mandrake.

RPM stands for Red Hat Package Manager and is described by its creator as “an open packaging system available for anyone to use and works on Red Hat Linux as well as other Linux and UNIX systems” (from the Red Hat installation guide).

A Quick RPM Tour

Before starting the programming discussion, I will introduce the RPM package manager and give a general overview of it. If you need more information, see Resources for more than 400 pages of interesting details on RPM's history, design, usage and programming.

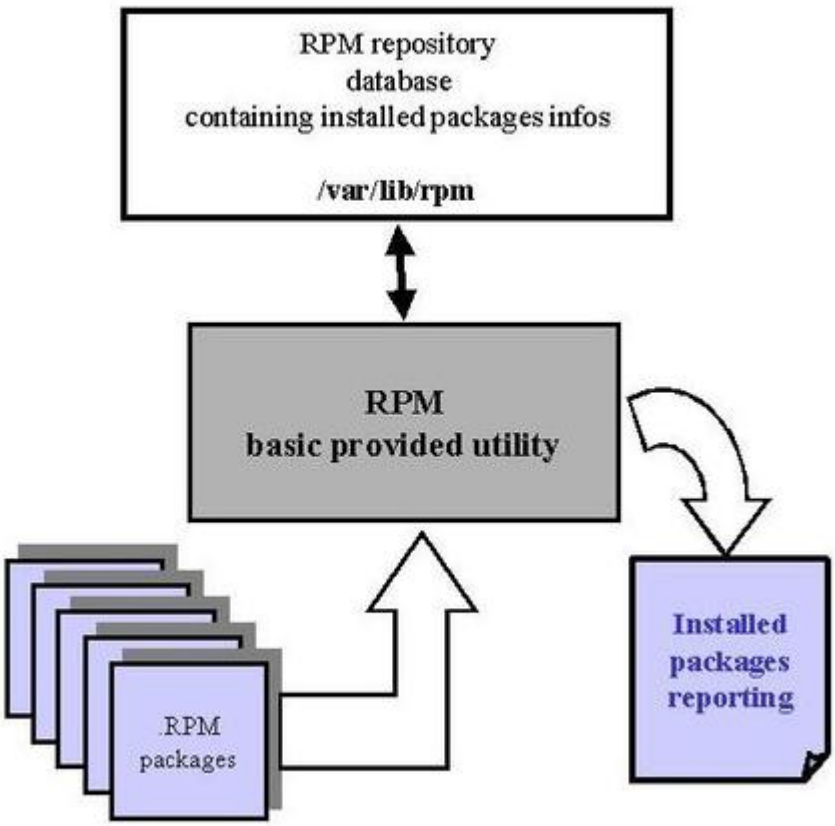


Figure 1. RPM Tool Components

The diagram in Figure 1 represents the different components involved in the RPM tool. RPM is composed of three main parts:

- a database
- candidate packages
- RPM utilities, which modify the database and packages

The RPM database, located in /var/lib/rpm on Red Hat distributions, is owned by root; it is a mirror of all the packages which are presently installed on your system. The RPM utility accepts various commands which query the database for installed packages, install or update the system with new RPM packages, remove unused packages from the system, and verify and check installed

package dependencies. Usually, when a new version of a given component becomes available on the Net, you have two choices:

1. Look for a tar gzip file containing the sources. Compile the sources on your system, then proceed with installation of the binaries. The given package generally provides a README, a make and make install procedure to help you.
2. Look for a tar, gzip file containing the binaries, meaning that someone else has already compiled the sources for you on the same kind of computer as yours. Then proceed with installation of the binaries.

The only way to check that everything went well is to try to execute the binaries. If something goes wrong, finding the reasons why may cause a lot of frustration and wasted time. With RPM, the process is quite different.

Let's say you are interested in installing version 1.9.18 of Samba. First, you should look on the Net for an RPM of the Samba package (instead of a tar, gzip package). Once you have it, type:

```
rpm -uvh samba-1.9.18p8-50.1.i386.rpm
```

This command will install (or upgrade) a copy of Samba on your system. It will also check that all dependencies needed by this version of Samba are present on your system. If the **rpm** command completes with no error messages, you're guaranteed the installed package will be ready to run without trouble at the end of the installation process.

This installation process will also update the RPM database which keeps track of all installed packages on your system and all their dependencies.

So if, six months later, you want to find out which version of Samba is installed on your system, typing the following command:

```
rpm -q samba
```

will tell you

```
samba-1.9.18p8-50.1
```

If you want to remove a package from your system, the RPM utilities will remove the files which were installed on your system during installation.

A Portable Tool to View RPM Packages

Now we know RPM packages are interesting. Many of them are available on the Net today and having a portable tool able to analyze an RPM package before installing it could be an interesting utility.

The Choice of Language

I think there are only two possibilities if you want to be portable to multiple UNIX and non-UNIX systems and easy to use in the Internet context: Perl or Java. From a technical point of view, there is no reason to prefer one over the other. The choice is a personal decision.

I have more experience programming Java than Perl. After a long and difficult thought process, I decided to start in Java, reasoning that if I later needed to add graphical presentation classes to the component, I could use the Java Swing package (which is available with JDK1.1 or JDK 1.2).

Where to Start

If you look at the /usr/lib directory of a Red Hat distribution, you will find a librpm.a static archive library. This library is provided with its corresponding C language prototypes: rpmlib.h, header.h and dbindex.h, located in /usr/include/rpm.

You can use those prototypes if you need to develop C utilities which deal with RPM resources. Chapter 21 of E. C. Bailey's book (see Resources) provides detailed information on how to do this. But, since we want to provide an independent Java package, these prototypes are of no interest to us.

The right place to start from (in the same resource) is Appendix A: Format of the RPM file, which gives us the RPM File format. The same appendix also provides us with the following sage advice: “RPM file format is subject to change.”

If an RPM file format is to be manipulated, you are strongly urged to use RPM routines to access the package file. Why? “RPM file format is subject to change”!

In our case, we will assume there is no immediate danger in querying an existing RPM package, since we commit to never modifying its structure inside our Java package.

very confusing. Please make sure a pair of *technical* eyes looks it over to make sure it sounds OK. Dave Wright's changes were incorporated. -Ellen

The RPM Class Design

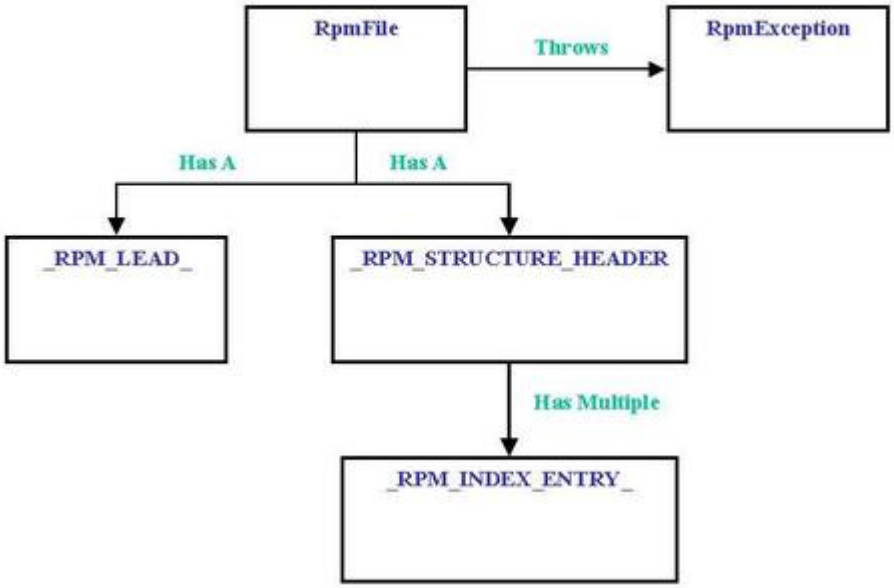


Figure 2. Structure of the Java RPM Classes Design in UML Format

Figure 2 represents the structure of the Java RPM class designs in UML format (Unified Modeling Language). Let's explain it in more detail. The UML class design provides a clean high-level representation of what an RPM package is.

Content is interesting information on the package and its installation rules. The content itself (not represented in the UML picture, for clarity) is only a compressed archive. When uncompressed, it is a cpio archive in SVR4 format with a CRC checksum (see Resources).

I cleanly separate the RPM object from its graphical representation. The classes in Figure 2 implement only operations on RPM files; they don't provide any graphical representation of them. Another class, called **RpmFilePanel**, will be added to provide a simple Swing display, which will graphically manipulate the basic RpmFile class, designed to implement the behavior of an RPM file.

The first interesting class is the **RpmException** class. This class inherits from the basic Java Exception class and implements a default constructor with no parameters and a constructor which takes a String message parameter. This class is the only exception rendered by the RpmFile Java Package. I am convinced that, when writing a new Java package, the first thing you should do is build an exception wrapper for the package. Later on, all the classes of the RpmFile package will throw an **RpmException** with an accurate message when something goes wrong. From an object-oriented design point of view, this technique improves your design's robustness, providing your package with full

isolation from the basic system layer. You can, of course, do the same thing in C++. The only problem is that support for exceptions by some C++ compiler implementations may not be available, and portability of your C++ code could be more difficult to implement.

The next public Java class is the **RpmFile** class itself. The public methods made available by the **RpmFile** class implement the following basic services (The constructor gives the ability to build a class instance. It does not take any parameters.):

- **set_rpmFileName (fileName)** method: this takes a URL fileName string as its parameter. This method binds the RpmFile instance with an URL representing a valid RPM package to view. If a problem occurs during the bind, a RpmFileException is thrown.
- Vector **get_rpmReport()** method: once the Rpm package has been bound to the RpmFile instance, this method can be called to get the package information. The information given back by this method uses a String vector which contains all the information found in the RPM package header structures.

The RpmFile logic is based on following two inner classes:

- **_RPM_LEAD_**: internally instantiated by the **RpmFile** class to validate the RPM lead structure of the loaded RPM URL file.
- Once the **_RPM_LEAD_** has been validated, the **RpmFile** class instantiates an **_RPM_STRUCTURE_HEADER_** class, which is used to check the RPM file header content. The header content consists of multiple **_RPM_INDEX_ENTRY_** stored in an internal array. Each element of this array represents a piece of header information which will be made available later via the **get_rpmReport** method. Since there is no reason to make those classes visible to mere mortals outside the RpmFile, they have been implemented as internal Java Classes inside the **RpmFile** class. A more precise UML graph is provided in Figure 3. I used the JVision 1.2 tool to automatically generate the UML class diagram from the Java source code. JVision is a very interesting, easy-to-use tool from Object Insight (http://www.object-insight.com/). This tool is able to automatically generate UML diagrams from Java source classes. Although not free, the license price is reasonable compared to other products. I have been using it for more than one year now, and it helps me in producing Java project documentation. A Linux beta version (free for non-commercial use) of the product is available on the Object Insight web site.

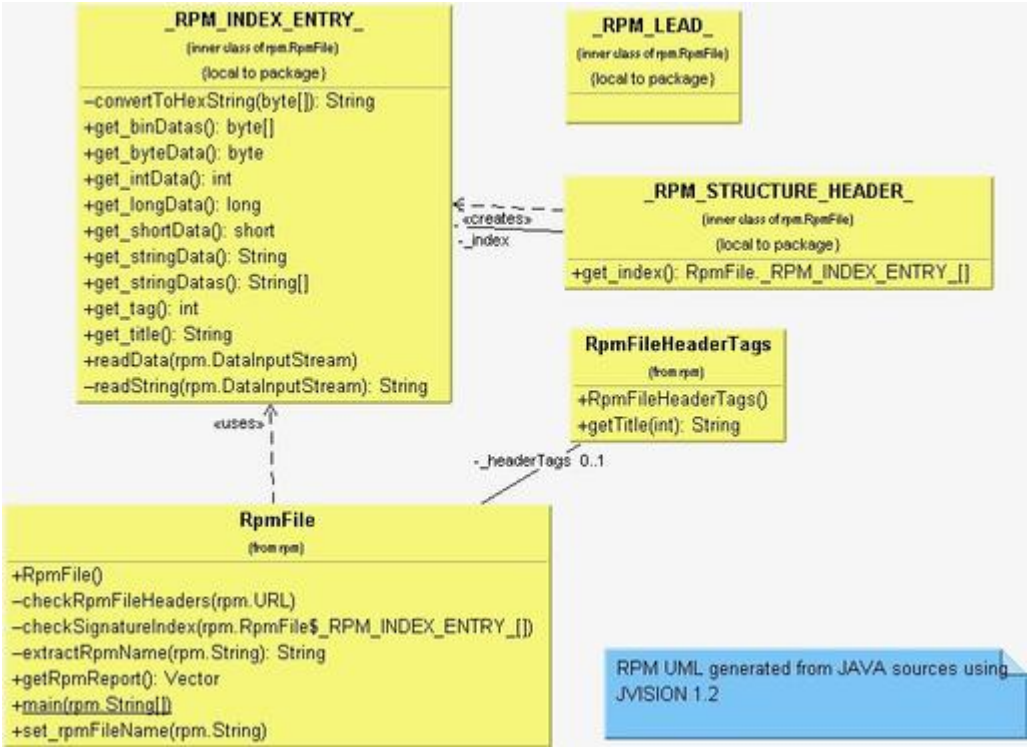


Figure 3. RPM

Adding a GUI Interface

Now our basic class structure, which is able to manipulate headers of RPM packages, is in place, and it would be a good idea to add a minimal GUI interface to it. We will build this very simple interface using the Java Swing package, available with either JDK 1.1 or JDK 1.2.

This very simple GUI interface is managed by the **RpmFilePanel.Java** class. This class inherits the basic JPanel graphical component and divides the screen into three parts using a Java BorderLayout:

- URL input field to input the RPM input file location in URL format
- scrollable ListBox to display the RPM file information
- button bar containing a “load” button, which proceeds with the URL RPM file loading Let's now explain how the GUI interface **RpmFilePanel** Java class interacts with the **RpmFile** class.

Starting the Java Application

The **RpmFilePanel.java** file contains a static **main** method to launch the Swing frame inside a Java application. You can use the JDK Java utility from the shell to start it:

```
java rpm.RpmFilePanel
```

Warning: you should have compiled the Java package and installed it in a location accessible from your **CLASSPATH**.

You should provide a valid RPM package file in URL form and type it into the URL field as shown below. Finally, pushing on the Load button will:

- instantiate a new **RpmFile** instance
- call the **set_rpmFileName** method with the RPM of URL field
- call the **get_rpmReport** method and display its String Vector result into the Swing List box as shown below

Figure 4 shows the screen layout after querying an RPM package.

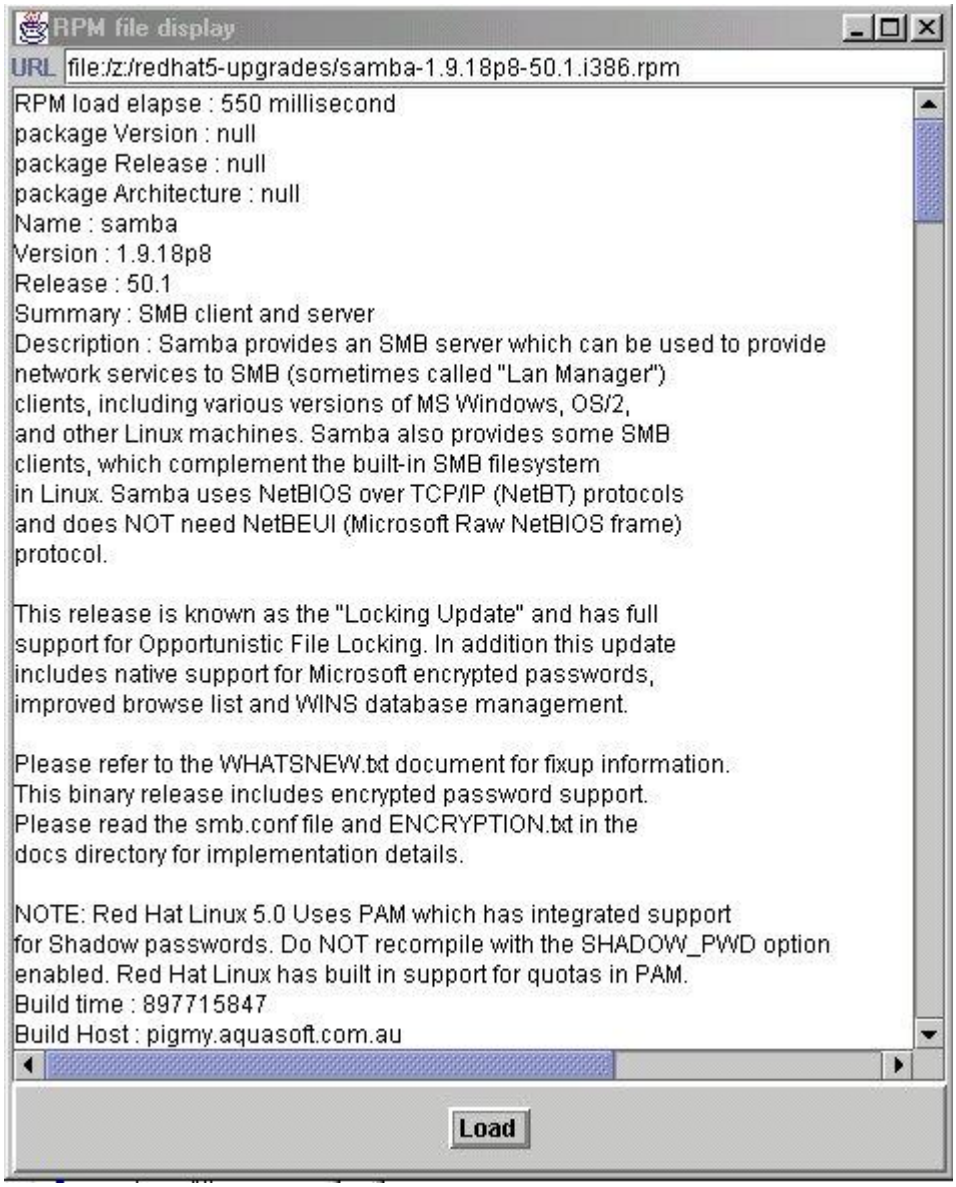


Figure 4. RPM Screenshot

When you need to build the provided source package, you may use either a JDK 1.1 version or a JDK 1.2.2 version. If you use JDK1.1, the prerequisite is to download the Swing package from Sun's site.

Conclusion

Using Java to analyse RPM file formats may be useful, and it gives you cross-platform portability, which may help you display RPM file contents on heterogeneous platforms. This basic tool may easily be enhanced, for instance to convert RPM contents into other formats or extract other parts of an RPM package.

Resources

email: jymengant@ifrance.com

Jean-Yves Mengant (jymengant@ifrance.com) is Chief Architect at SEFAS Technologies in Paris, France. He has been a professional programmer for more than 20 years on multiple OSes (MVS, Linux), languages (assemblers, C, C++, Java and Perl) and software topics (Network, Tp Monitors and GUI interfaces). He has used Linux systems professionally for programming tasks since 1995 and kernel 1.2, and the Java language since 1996 and JDK 1.0.

Archive Index Issue Table of Contents

Advanced search

The Artists' Guide to the Linux Desktop—Part IV

Michael J. Hammel

Issue #76, August 2000

Variety is the spice of life.

The two previous articles in this series were focused on particular window managers: Enlightenment and Window Maker. I looked very closely at those two window managers because they epitomize the personalization that is possible in the Linux desktop. Unlike in the Mac or PC world, the look and feel of the Linux desktop can vary to the extreme. One Linux user's windowing interface need not look—or act—anything like anyone else's. The windowing interface you choose is like a big empty apartment. Not until you add the furniture and hang some pictures does it become a livable space. The kitchen is still a kitchen, but in Linux, it's *your* kitchen.

In this, the fourth and final part of the series, I'm going to take a quick look at some of the other, less well-known window managers available for Linux. Most of them have unique benefits over any other window manager—smaller memory footprints, more efficient code, better extensibility and so forth. They all look and behave slightly differently, although aside from AfterStep, none are designed with the graphical variety intended in both Enlightenment and Window Maker. That doesn't mean you don't want to consider them. On the contrary, there are definite situations in which you'll want to consider these alternatives, particularly if you do much work on a laptop.

AfterStep

- URL: www.afterstep.org
- Requirements: XPM, JPEG and PNG libraries are recommended, but these should already be on any recent Linux distribution.
- Compliance: GNOME (partial), KDE (unknown)
- Extras: if it runs on Window Maker, chances are good it runs with AfterStep, but it's not guaranteed.

Figure 1. AfterStep's Default Theme

When I originally started this series, I had planned on looking at AfterStep in the same thorough manner I looked at E and Window Maker. After doing some research, I discovered Window Maker and AfterStep are pretty much the same beast. Internally and politically they may differ, but from an end-user perspective they're very similar, so much that if you read the previous article, you're up to date with AfterStep.

Still, there are a few things I need to mention about this window manager. AfterStep adds a bar across the top of the display, similar to the task bar of GNOME and KDE. The bar actually shows up only when the first application has been started; after that, each application gets added to the bar. It allows you to jump to applications very quickly, although personally I don't particularly like these types of features.

AfterStep features can be enabled or disabled prior to compiling the software. The configure script has far more options than most of the other window managers. This allows you to turn off some features, hopefully slimming down the window manager for use on resource-limited systems. However, if you use the **-prefix** option, be aware that the “make install” process doesn't create the /bin and /man directories under the directory you specified—you'll have to do that manually.

Unlike Window Maker, AfterStep has a pager similar to FVWM's pager, which I like quite a bit. It appears to look and work much like the one I'm used to and is available in the default configuration. Although I probably won't switch to

AfterStep (the icon features I disliked in Window Maker are also present in AfterStep), I am more curious about it now that I've found a reasonable pager.

Blackbox

- URL: blackbox.alug.org
- Requirements: none
- Compliance: GNOME (no), KDE (partial)
- Extras: BBPager, BBTime, various others; all available from the main Blackbox site. Supports Window Maker dock applications through the use of a built-in tool called the Slit.

Figure 2. Blackbox, Using an Accompanying Theme

Of all the window managers I tried, Blackbox is certainly the easiest to build and install. It needs no special libraries, has no special requirements and took only a minute or two to compile. Obviously, this will depend on your hardware configuration, but the point is, this is the slimmest, most streamlined window manager available. That also means it's the one with the fewest bells and whistles.

While most of the other window managers provide some form of pizzazz through icons or image handling, Blackbox provides nothing more than simple gradients. It does include a tool for placing background images on the root window, but that's about all the pizzazz you'll find here. Still, the implementation does include support for dockable applications from Window

Maker, through a built-in function called the Slit. I didn't try this feature, since the point of Blackbox is to *not* add snazzy appearance. It's truly a window manager designed to provide a decent look with as small a memory footprint as possible. This makes Blackbox a likely candidate for use on laptops.

Like with most window managers, you have to edit text files to make changes to the root menu. This isn't a serious problem, however. A fairly complete description of the menu file format is available on-line, linked from the Blackbox web site.

Blackbox has a panel across the bottom of the screen which includes a clock, two sets of arrows for cycling through windows and desktops, and an area displaying the title from the window which currently has the keyboard focus. An additional tool, BBPager, is available which provides many of the same pager features that FVWM's and AfterStep's pagers do. Although the pager accepts standard **-geometry** settings to set its position on the screen, the panel appears to be forced to the bottom of the display. I'd like a way of changing that, to move it to the top or side of the display, but didn't see whether that was possible.

One thing Blackbox doesn't do is enforce edge resistance by default. Edge resistance lets you slide a window up against an edge of the display, but not past it without a little applied force. I like window managers that do this; I seldom use windows that span workspaces, but do like to butt windows up against the edge of the display.

In general, I like Blackbox; I just need to learn how to configure it properly.

Icewm

- URL: icewm.sourceforge.net. A better site is at http://icewm.cjb.net/, where you can find user documentation.
- Requirements: XPM or Imlib, although these appear to be optional; sgml2html to build documentation.
- Compliance: GNOME (at least partial), KDE (unknown)
- Extras: a couple of preference editors are available as external utilities.

Figure 3. Icewm's Default Theme

The basic installation comes up looking a little like a Windows interface. At least, it's very usable right from the start, even if the theme makes me slightly ill. There doesn't appear to be a quick way of changing themes right from the menus or from the panel, as you can with Blackbox, but it does come with a number of different themes you can install manually.

One interesting feature is the extra blank bar that appears when you set the **TaskBarDoubleHeight** option in the configuration file. This provides an area in which you can type a command to launch programs without having to open an xterm. This is nifty for all those times I do **echo 123/54 | bc**, which I do quite a bit. If the command (such as my example) is not a program, it is run as a shell command.

Icewm, like Blackbox, is designed to have a small memory footprint. Unlike Blackbox, it also tries to provide the icons and fluff that many users expect from window managers these days. Configuring menus can be done using some graphical tools, although both tools are fairly crude. Chances are you'll probably end up configuring menus by hand.

Sawmill (or Sawfish)

- URL: sawmill.sourceforge.net
- Requirements: Imlib, librep Lisp interpreter (0.9+) from Sawmill author and the rep-gtk binding (0.7+)

- Compliance: GNOME (at least partial), KDE (unknown)
- Extras: RPMs are available, but I install from source.

Sawmill was, at one time, the default window manager for GNOME on Red Hat distributions—or at least I think it was. I'm not sure what they use these days. If you go to the main web site for Sawmill, you'll find it's presented more as a developer's platform than something an ordinary end user will be interested in. Use of Lisp is probably technically appropriate, but is likely to scare off the non-technical crowd.

Unfortunately, Sawmill would not build on my stock Red Hat 5.2 system because it was missing GNU MP, yet another of the massively obscure requirements so many packages seem to have these days. Switching over to my Red Hat 6.1 box, I had the same problem, so Sawmill never got built. Since there were multiple packages to download, I didn't try the RPMs either. Sorry—plug-n-play is essential here. One package, build and install. I'm not tolerant of much more than that these days.

FVWM

- URL: www.fvwm.org
- Requirements: none, essentially, with the latest Linux distributions (they'll have what you need).
- Compliance: GNOME (unknown), KDE (unknown)
- Extras: most of the external modules come with the distribution now.

Figure 4. My FVWM Theme

After all this, we're finally down to what I actually use myself—FVWM2. It's one of the grandfathers, two or three times removed in some cases, of most of the rest of the window managers I've talked about. Why do I use it? Because I have an extensive menu system designed around a minimalist display that I haven't been able to duplicate with any of the other window managers. I also adore the FVWM Pager. With it, I can move windows around between desktops without actually having to move to those desktops. I can also see at a glance what I have open. I tend to use three desktops with multiple pages, and use the same pages for the same things—xterms with logins to specific machines, Netscape running across the network on different boxes but displayed locally, my XNotesPlus package and so forth. With FVWM's pager, I can easily see what I have open. Then I just click once to jump between multiple projects—say, an article for *Linux Journal*, some analysis for Linsight and administrative work from my Graphics Muse site. It's a very handy tool. It also lets me work without dealing with a bunch of nasty icons.

I haven't been able to find anything quite like this in any of the other window managers in such a minimalist form. Enlightenment's pagers are close, but Enlightenment has a heavy memory requirement. With FVWM, I can run close to the same configuration on any system I have, and I have a bunch. AfterStep has a similar pager, but those big ugly icons annoy me.

Above all else, the window manager, to an old UNIX hack like myself, should help me do my work. It shouldn't sing and dance in front of me and *then* do the work. After everything I've said about personalizing the workspace, my way of doing so is to remove the visibility of the workspace, save for a clever *X Files*-style background I generated with the GIMP. I use a toolbar across the top of the display with a ton of menus for launching various applications. It is a bit of an annoyance that FVWM requires me to restart the window manager every time I make a change to one of the menus, but it's only a minor annoyance.

Series Summary

So there you have it. A few other window managers are probably out there, but we've covered all the ones with any serious following on Linux. Using a window manager of your own choice isn't too difficult, even if you plan on using GNOME or KDE, since most window managers either already do or will soon support the two primary Linux desktop environments. Your choice will be based more on what you want to look at and how much hardware you have on which to run it. In the end, the key is that you have a choice, and that leaves you in *control* of your desktop.

email: mjhammel@graphics-muse.org

Michael J. Hammel (mjhammel@graphics-muse.org) is a graphic artist wannabe, a writer and a software developer. He wanders the planet aimlessly in search of adventure, quiet beaches and an escape from the computers that dominate his life.

Archive Index Issue Table of Contents

Advanced search

Comparing Java Implementations for Linux

Michael Hirsch

Issue #76, August 2000

No hype here—find out what Java really is and what choices you have with Java for Linux.

Given the amount of hype that Linux and Java are currently receiving, the combination of the two rates a solid 12 on a hype scale ranging from 1 to 10. However, being an engineer—and thus a skeptic—at heart, I prefer hard facts over marketing slogans. So, I set out to create a simple benchmark program to compare different Java implementations for Linux.

No matter what the hype says, I believe the most important contribution of Java is that it is simply a very well-designed programming language that most professionals enjoy using. Some of the features of Java that make it attractive for software engineers are:

- Java is a “pure” object-oriented programming language (unlike C++, which is a hybrid), with a simple and elegant object model. Depending on your point of view, this may be an advantage or disadvantage. My personal experience is that programmers who are new to objects get up to speed a lot faster if they use “pure” OO languages such as Smalltalk, Eiffel or Java as opposed to hybrids such as C++.
- Java avoids many of the complexities of C++, thus making programs less error-prone and programmers more productive. For example, in Java there is only one way to make a new object: you call the new operator and get a reference (or pointer, if you like) to the object. Compare this to the many ways to create objects in C++.
- Java needs no preprocessor and therefore is immune to “macroitis” and endlessly nested include files.
- Java has a garbage collector to free memory consumed by objects which are no longer used. In C and C++, a fair amount of design and programming work must be spent on memory allocation and deallocation

schemes. Freeing objects in Java is automatic (as it is in Smalltalk and Eiffel).

- The sizes of basic data types such as int and float are defined by the Java language specification, so an int is always 32 bits, regardless of the platform your program runs on. How often have you implicitly assumed that a pointer has the same size as an integer in a C program, and suffered the consequences when you ported your program to an architecture where this is not true?
- Finally, Java has built-in support for multithreading and synchronization of multiple threads, and comes with a huge class library out of the box (although some people, including myself, feel that the library has gotten too huge lately).

Java is still not a standardized language, and it is doubtful if it will ever be. Sun has the final say over what's in the language and the libraries and what isn't, period. The “Java 2 Platform” or simply the Java Developer Kit (JDK) V1.2 has about 1200 classes in its libraries. Sun ships its JDK on Solaris/SPARC, Solaris/Intel, Windows-32 and recently on Linux, too. IBM has recently released a JDK 1.3 implementation for Linux. I ran the benchmark with this JDK on the same hardware and OS as all other benchmarks and got the following numbers:

Elapsed time: 1384 millisecondsObjects / millisecond: 361Output of java-version: Classic VM (J2RE 1.3.0 IBM buildcxdev-20000502 (JIT enabled: jitc))

Sun's JDK achieves platform independence of Java programs by relying on an architecture-neutral intermediate code called “bytecode”, which is interpreted on each target machine. The interpreter is called a “Java Virtual Machine” or JVM for short.

Since interpretation is slow, most JVMs come with a Just in Time Compiler (JIT). A JIT translates bytecodes into machine code on the fly, i.e., while the interpreter is running. The resulting machine code is stored in memory and lost when the interpreter terminates. Generally speaking, pure interpreters show faster program startup times, while a JVM with a JIT takes longer to start (because it compiles bytecodes); but once a program is up and running, it is faster than an interpreted program. There are many optimizations that can be made to interpreters and JIT compilers. Sun's “Hotspot” JVM, which is not yet available under Linux, but should be eventually, is one attempt to get the best from both worlds.

Finally, there is nothing in the Java language specification which prevents the application of standard compiler technology, i.e., compiling Java source code directly into machine code. The Java front end to the GNU compiler system does just that.

Java Implementations for Linux

At the time of this writing, there were quite a few Java implementations available for Linux. These are the ones I am aware of and was able to get to work:

- The Blackdown port of the Sun JDK, version 1.2.2. I tested release candidate 4 of this port, which includes both an interpreter and a JIT. The JIT is a port of the JIT shipped by Sun with the JDK for Solaris. By default, the JIT is enabled, but it can be turned off with a command-line switch. The documentation that comes with this port warns that the JIT is not yet entirely reliable. I downloaded this port from one of the numerous Blackdown FTP mirrors, which are accessible from Blackdown's homepage at http://www.blackdown.org/.
- Sun's own version of the Blackdown port of Sun's JDK, version 1.2.2. Recently Sun began to publish a JDK for Linux on their own web site. This port is, as far as I can tell, the same as published by the Blackdown folks. There is at least one noticeable difference though: Sun's version comes without a JIT; it is an interpreter only. However, Sun recommends using the JIT developed by Borland. I downloaded this port from Sun's Javasoft web site at http://www.javasoft.com/.
- Borland's JIT for the Sun and Blackdown ports of the JDK 1.1.2. This is not a complete Java developer kit; it is only a JIT. It works with the Blackdown port of Sun's JDK 1.2.2, and hence with the Linux JDK 1.2.2 published by Sun. I downloaded this JIT, which is a simple shared library of about 170KB, from Borland's web site at http://www.borland.com/.
- The Blackdown port of the Sun JDK 1.1.8. I tested version 1 of this port, which only includes an interpreter (unlike the Windows version which comes with a JIT).
- IBM's JDK version 1.1.8 for Linux. This JDK has a reputation of being “very fast” and also very stable. It comes with a JIT, which is enabled by default, but can be turned off with a command-line switch. I downloaded this JDK from IBM's web site at http://www.ibm.com/.
- Kaffe version 1.0b4 from Transvirtual Technologies. Kaffe was developed by Tim Wilkonson and others from scratch, without any code from Sun. The version of Kaffe I used is compatible with Sun's JDK 1.1. I used the Kaffe package which is on the Red Hat 6.1 CD-ROM; however, there is a web site devoted to the open-source version of Kaffe at http://www.kaffe.org/. Kaffe is available for a wide range of UNIX versions and processor architectures, not just for Linux on x86 processors. The version of Kaffe I used includes a JIT, which is always on, or at least, I couldn't figure out how to turn it off.

- The native Java compiler from Cygnus Support shipped with their Codefusion-1.0 development environment. This compiler is an enhanced version of EGCS, the Experimental GNU Compiler System, although you can hardly call this high-quality compiler "experimental" any more. Unlike all other Java implementations I looked at, this compiler generates native code which is link-compatible with object files created from C and C++ source files. The compiler comes with a library with the necessary runtime support for Java programs, which includes, among other things, a garbage collector. The version of EGCS I used ("2.9-codefusion-990706") is not free; you have to buy it from Cygnus. Their web site is at www.cygnus.com/ or www.redhat.com.

Implementation	Output of "java -version"	JDK Compatibility	Compiler Technology	Website
Blackdown JDK 1.2.2 RC4	Classic VM (build Linux_JDK_1.2.2_RC4, native threads, sunwjit)	1.2	Interpreter and JIT	www.blackdown.org
Sun JDK 1.2.2	Classic VM (build 1.2.2-L, green threads, nojit)	1.2	Interpreter only	www.javasoft.com
Borland JIT for JDK 1.2.2	n/a, add on to Sun or Blackdown JDK 1.2.2	1.2	JIT	www.borland.com
Blackdown JDK 1.1.8 V1	Java version "1.1.8"	1.1	Interpreter only	www.blackdown.org
IBM JDK 1.1.8	Java version "1.1.8"	1.1	Interpreter and JIT	www.ibm.com
Kaffe Version 1.0b4	Kaffe Virtual Machine, Engine: Just-in-time, Version: 1.0b4, Java Version: 1.1	1.1	JIT only	www.kaffe.org
Cygnus Codefusion 1.0	2.9-codefusion-990706	1.1	Native code compiler	www.cygnus.com

Table 1. Java Implementations for Linux

The Benchmark

Let's start this section with a disclaimer: writing meaningful benchmark programs is very difficult, and no single benchmark can do justice to all aspects of a complex system such as a Java implementation. The benchmark I used is no exception. There are many issues it doesn't address; for example, it doesn't cover multi-threading issues, database access or graphics performance.

That said, let's look at the benchmark. The benchmark program does what most object-oriented programs do from a technical point of view: creating objects and calling methods on them. More specifically, the benchmark creates half a million very simple account objects, adds an amount to each object created, then adds up the amounts of all objects. The result of the benchmark is the elapsed time it takes to create and process all objects, and derived from that, the number of objects created and processed per millisecond.

I ran all benchmarks on a Dell Latitude CP Notebook with a 233MHz CPU and 128MB of RAM under Red Hat Linux 6.1. The code of the benchmark program is given in Listing 1.

Listing 1. Java Benchmark Source Code

As already said, this benchmark is not perfect, but I think it does give an indication of the relative performance of different Java implementations on the same platform.

Results

Now to the results. It comes as no big surprise that a JIT is generally faster than an interpreter, and native code is even faster than a JIT. Table 2 summarizes the results.

Java Implementation	Elapsed Time [ms]	Object / ms	Mode	Remarks
Sun JDK 1.2.2	4'815	103	Interpreter	To start: "java PerformanceTester"
Sun JDK 1.2.2	4'973	100	JIT	With Borland JIT V1.2.13 To start: "java -Djava.compiler=javacomp PerformanceTester"
Blackdown JDK 1.2.2 RC4	4'869	102	Interpreter	To start: "java -Djava.compiler=NONE PerformanceTester"
Blackdown JDK 1.2.2 RC4	4'123	121	JIT	With Sun JIT To start: "java PerformanceTester"
Blackdown JDK 1.1.8 V1	10'005	49	Interpreter	To start: "java PerformanceTester"
IBM JDK 1.1.8	7'356	68	Interpreter	To start: "java -nojit PerformanceTester"
IBM JDK 1.1.8	6'207	80	JIT	To start: "java PerformanceTester"
Kaffe 1.0b4	6'090	82	JIT	To start: "java PerformanceTester"
Cygnus Codefusion 1.0	2'854	175	Native Code	To compile: "gcj PerformanceTester.java --main=PerformanceTester -o perftest" To start: "./perftest"

Table 2. Java Benchmark Results

There are some observations I feel are worth mentioning. First, Java 1.2 implementations are generally faster than Java 1.1 implementations. This is a strong argument to choose Java 1.2 over Java 1.1, in addition to the much greater functionality offered by Java 1.2. Second, the fastest Java 1.2 implementation is currently the Blackdown port with the JIT enabled. The JIT provided by Borland does not speed things up, at least not in this benchmark. Third, if you have to stick with Java 1.1 (e.g., for compatibility reasons), your best options are currently Kaffe 1.0b4 and IBM's JDK 1.1.8 with the JIT enabled. Fourth, if you don't need Java bytecodes and Java 1.2, the fastest option of all is to use the Java front end of EGCS. I was not able to test the open-source version of EGCS with **gcj**, but I suspect the performance of it is within a few percentage points of the version sold commercially by Cygnus/Red Hat.

Finally, I ran the benchmark on the same hardware with Sun's JDK 1.2.2 on Windows NT 4.0 with service pack 5. The results are disappointing for Linux aficionados. The benchmark reports 198 objects per millisecond with the JIT enabled, and 133 objects per millisecond in interpreter mode. In other words, the Java interpreter on Windows is faster than the fastest JIT available for Linux, and the Windows JIT is faster than native code on Linux. As long as Java performance on Linux is not as good as on Windows, I don't think Linux will become the platform of choice for Java developers. Let's hope that in the near future companies such as Sun or IBM invest at least as much time in tuning Java on Linux as they do in tuning Java on Windows now.

What About C++?

Being a longtime C and C++ programmer, I could not resist the temptation to write a similar benchmark in C++. Listing 2 has the code of the C++ version of the benchmark.

Listing 2. C++ Benchmark Source Code

I compiled the C++ benchmark program with both the standard C++ compiler that comes with Red Hat 6.1 (EGCS 2.91.66) and the C++ compiler included in Cygnus's Codefusion 1.0 development environment. Table 3 shows the results of the C++ benchmark.

C++ Compiler	Elapsed Time [ms]	Object / ms	Remarks
EGCS 2.91.66	972	514	To compile: "g++ -o perftest PerfTest.cc"
EGCS 2.9-codefusion-990706	952	525	To compile: "g++ -o perftest PerfTest.cc"

Table 3. C++ Benchmark Results

The benchmark shows that a comparable C++ program is still faster, by a factor of 3, than the fastest Java implementation (the gcj native code compiler), and even 4.3 times faster than the fastest Java JIT.

Conclusion

If there is any conclusion to this simple benchmark at all, it is this: I'm very much impressed by Java as a programming language (i.e., its design), but I'm not too impressed by current Java implementations on Linux or any other platform, for that matter.

Java implementations are a good example for one of the oldest laws of computing, which says that software becomes slower faster than hardware becomes faster. No doubt, Java has its place, and very often—in fact, more often than not—its advantage in programmer productivity outweighs its disadvantage in performance compared to C++. Buying a CPU which is four times faster is usually less expensive than spending two times as long to develop an application. At least, this is true for custom-developed systems. Embedded systems or mass-marketed software with hundreds of thousands of copies are an entirely different story. Saving $5 on a CPU chip in a $100 device can make all the difference and pay for increased development time.

It comes as no surprise that Java is used today mostly for custom-developed enterprise applications, where development cost and time are of paramount importance. As long as Java implementations are not within 80% of the performance of comparable C++ applications, I don't think we will see off-the-shelf products such as word processors or spreadsheets written in Java.

Wouldn't it be nice to have a programming language with the elegance and simplicity of Java and the efficiency of C? Well, maybe this will be the next big open-source project.

All listings referred to in this article are available by anonymous download in the file ftp.linuxjournal.com/pub/lj/listings/issue76/4005.tgz.

email: Hirsch.Michael@acm.org

Michael Hirsch can be reached via e-mail at Hirsch.Michael@acm.org.

Archive Index Issue Table of Contents

Advanced search

Configuring Bash

David Blackman

Issue #76, August 2000

A quick introduction to the Bash shell.

Welcome to the world of Bash, most widely used shell in Linux. Bash is surprisingly configurable, and, by the time you finish reading this article, you'll have an environment more comfortable for you. Bash does not differentiate between internal shell variables and external environment variables. A shell variable is a variable (usually all caps), associated with a value, and carried around between shells. Many programs use their own variables, like **PILOTRATE**, which they check. Bash has its own variables, like **MAIL**, that are important to it. Environment variables are set using the syntax:

```
export VAR=VALUE
```

or in two lines:

```
VAR=VALUE
export VAR
```

To check the value of an environment variable, type **echo $VAR**, or to see all set variables, type **env**. bash executes your ~/.bash_profile file for the login shell (on the console), and ~/.bashrc for non-login shells (xterms and the like). Often you may just want to link one to the other. If you export a variable, or set an alias on the command line, it only stays active for that one bash session. You must put it in your login script for it to stick. If you start having a monolithic .bashrc file and want better organization, you can split it up. Often, people break up their .bashrc into aliases, variables and functions, and the .bashrc simply executes the others. To have your .bashrc execute other files put in a line like this:

```
source FILE
```

The Prompt

The first environment variable we'll discuss is PS1. PS1 stores a character string that is interpreted by bash for use as your prompt. Here is a sample PS1 and its generated prompt:

```
PS1='<\u@\h:\w>$'
<blackmad@moomintroll:/etc>
```

Backslashed characters are interpreted, while other characters are displayed verbatim. **\u** is translated to user name, **\h** is translated to host name up to the first period, and **\w** is the working directory. Some of the most important backslashed characters, which can also be found in the bash manpage, in the PS1 section, are shown in the table “Interpreted Characters.”

Table 1. Interpreted Characters

Cool Xterm Title

One of the cool things in all X terminal emulators (xterm, rxvt, Eterm, ...) is that if you print **"\033]0;STRING_HERE\007"**, the title of the term changes to **STRING_HERE**. Try it by typing:

```
echo -n "\033]0;Be Happy\007"
```

What I do with this is put a small function in my .bashrc function xtitle (see Listing 1), and I call this after I've set my PS1 variable, so at the end of my .bashrc file I have the line:

```
PS1=''\u@\h:\w>$''
xtitle
export PS1
```

This means that if I'm in a terminal emulator, it will set **TITLEBAR**, a string which will append user@hostname:directory to my prompt string (so it's printed each time I get a new prompt), and then export it. (Note that if your terminal emulator sets **$TERM** to something other then xterm* or rxvt*, you need to add another case, with **| *WEIRD_TERM_ENV*** on the line with **xterm* | rxvt***) before the close parenthesis.

Listing 1. Function xtitle

Aliases

One of the most useful things to use with Bash is aliases. Aliases simply direct Bash to interpret a text string as something else. For example, you can fix it so that when you type **happy**, Bash interprets it as:

```
echo I'm a shiny happy shell
```

All aliases take the same form:

```
alias ALIAS="COMMAND"
```

Often, you may want to change the default behavior of a command, such as **ls**. I alias ls in this way:

```
alias ls="ls -aF --color"
```

ls now prints all files, in color, with classification. **\ls** will execute the unaliased command. Other times you may decide to define a whole new command in order to shorten the amount of repetitive typing. Here are a few aliases I use:

```
alias mkall=\
        "./configure && make && sudo make install"
alias whizz="ssh whizziwig@www.whizziwig.com"
alias tgz="tar -xvzf" alias ll="ls -aFl"
alias ls-d="ls -Sc"
```

These all save time and keystrokes, and since anything you type after the alias is still passed to Bash, it will just translate the part that is aliased. In my case, executing **tgz linux-2.2.14.tar.gz** actually executes **tar -xvzf linux-2.2.14.tar.gz**.

Mail

The **$MAIL** variable specifies which mailbox you want Bash to poll for new mail. You generally want to set this to your inbox. I use procmail, so I have many mail folders. My inbox (where mail that's actually addressed to me gets sent) is /home/blackmad/Mail/inbox, so when I get new e-mail there, Bash tells me: "You have mail in /home/blackmad/Mail/inbox."

PATH

The **PATH** variable determines where, and in what order, Bash will look for executables. Each directory is separated by a colon (:). Bash interprets your path from right to left. Let's say your **PATH** is set to **/usr/bin:/bin/:/sbin/:/sbin/**. When you enter a command, Bash will look for it in its internal shell functions first, then /usr/bin, then /bin and so on, until it either finds the command or gets to the end of your **PATH**. Often you may simply want to append or prefix your current **PATH**; you can do this by specifying

```
PATH="$PATH:/next/path:/next_next/path"
```

or

```
PATH="/prev/path:$PATH:/next/path"
export PATH
```

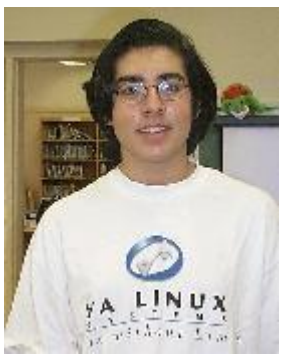
In the first example, Bash will look through **/next/path** and **/next_next/path** after it finishes with your current **PATH**. In the second example, Bash will first look in **/prev/path**. You may want to prefix your **PATH** with /usr/local/bin, since that is where hand-compiled programs are usually located, and these are generally more recent then those that came with your distribution. You may also want to prefix ~/bin and have a bin directory in your home directory where you can put customized versions of programs and scripts (useful if you don't have root on the box).

The Last Step

Since Bash just runs through your .bashrc file and executes everything in it, you can toss in programs you want to run each time you log in. At the end of my .bashrc file, I have the following:

```
fortune
mesg y
users
```

So whenever I log in, **fortune** greets me with a bit of wisdom, messages are turned on, and I find out who's logged in to the systems.



David Blackman is a sophomore and a system administrator at Stuyvesant High School. He hopes to write the killer application for Linux soon and get hired by VA Linux Systems. He loves Perl, even though he knows it's evil, and enjoys the pointer arithmetic of C.

Archive Index Issue Table of Contents

Advanced search

VARs: Increasing Margins through Free Software

Dean Taylor

Issue #76, August 2000

From resellers to solution providers, today's VARs are bringing Linux solutions to the world of application service. “It was the best of times. It was the worst of times.” --Charles Dickens

The above is a statement well-known throughout literature and throughout evolving societies. In this ever-changing, pervasive Internet society, such times are felt in all aspects of business. The Internet has shifted the power of presence and acquisition and is beginning to shift the power of commerce. Many value-added resellers (VARs) have recognized this shift and have been able to change with the new economy. They are deploying solutions that take advantage of this new market, a market that provides more margins in service than in products. Now and in the future, service margins will be “king”.

In fact, VARs are suggesting that their business is not even associated with the process of being a “reseller” of products, but a seller of services. These services include messaging, commerce, maintenance, support, software and hardware leasing to name a few. Quoting Scott McNealy of Sun Microsystems,

> Five years from now, if you're a CIO with a head for business, you won't be buying computers anymore. You won't buy software either. You'll rent all your resources from a service provider.

Many of the services can be made available because of the Internet and the low costs associated with its pervasive presence. Titles such as Internet Integrators, e-Solution Providers (eSP), e-Business Integrators and Application Service Providers (ASP) are now being declared throughout the channel. While the ASP market will continue to define itself, the one thing projected among all analysts is that this market is going to explode over the next two to five years. Analysts such as IDC project the ASP market to be in excess of eight billion dollars, while Dataquest put this market in excess of 20 billion dollars during the same time

period. However, the solutions these providers offer must allow customers to take advantage of ever-increasing Internet services that can be customized and based on a scalable and stable platform.

In addition, Linux is causing another market shift. Linux is being seen as a real business solution and is one of the fastest-growing server solutions in the industry. Examples of these solutions are being found in all sizes of corporations. The market conditions required for Linux to succeed are coming into focus. Some of these market conditions include industry leaders porting their software packages to Linux, hardware vendors' platforms being “certified” on Linux, Microsoft being declared a monopoly, users seeking viable software alternatives that do not crash and those who want to take advantage of the Internet's pervasive presence. Those solution providers who adapt their business models to provide such services and solutions will tap into a multi-billion-dollar market.

By combining customizable, extremely reliable and stable Linux-based solutions with the exploding ASP market, opportunities of service and product revenue abound for serving the largest to the smallest corporation. In a recent survey conducted among VARs, one-third stated they already support some type of Linux solution, while one-quarter will utilize Linux to deliver e-business solutions via the Internet. In other words, solution providers must recognize that they need to “deliver E or be Eaten”; Linux is a platform which can deliver the “E”.

A question that solution providers constantly ask about the Linux model is, “How do I make money?”. The answer is the same whether the operating system is proprietary or open—sell solutions, not simply a software package! However, Linux solutions are generally open for customization, are extremely stable and are low-cost by nature. Because of these qualities, service margins are increased. Linux was created on the Internet, by the Internet, for the Internet, and is optimized to work in this environment remotely. With such benefits, solution providers do not need to make on-site visits for service offerings such as configuration, deployment, diagnostics or administration.

As many solution providers begin to realize they can take advantage of Linux-based solutions, analysts and others are predicting revenue opportunities in the Linux traditional and specialized server markets.

A Dataquest study suggests that Linux servers will “represent approximately 3.4 percent of traditional server revenue or $1.9 billion ... by 2003.” Recently in Caldera Systems' “Tell All and Win Solutions” contest, some of Caldera's e-Solution Providers (eSPs) described how they deployed OpenLinux as a traditional server by completely replacing NT servers. Some eSPs leverage

OpenLinux by providing add-on solutions. For example, one client needed to upgrade to NT from NetWare 4.11 and ran low on disk space. Unfortunately, the client could not afford to add disks to the NT server, so they utilized their old NetWare server, OpenLinux 2.3 and Samba to create a software repository to solve the problem. Other providers utilize Linux as an Intel UNIX alternative and re-deploy their Intel UNIX applications for their clients.

Quoting from the same study regarding server appliances, “Dataquest believes that Linux servers will represent approximately 24 percent of server appliance revenue, or $3.8 billion... by 2003 ... The key trend is optimization and it is occurring on the software side and the hardware side.” As the Server Product Matrix in the study indicates, optimization will be critical for those servers in the future. Having open-source software allows for optimization.

These server appliances, or “specialized servers” as they may be called, will be deployed by solution providers and ASPs. Examples of these solutions from eSPs include Internet servers, file servers and backup servers, to name a few. The reason? These low-cost, specialized servers allow a client to continue using their existing hardware while increasing functionality. In addition, the cost to deploy such servers is minimal.

As for Linux client solutions, many eSPs are utilizing OpenLinux Desktop for Internet access devices. For example, one eSP utilizes OpenLinux Desktop in a library setting to provide kiosk security that was cost-prohibitive utilizing Microsoft Windows. Through OpenLinux Desktop, the eSP was able to provide a web browser on the desktop that offered Internet access and access to CD-ROMs available from the library. Another client solution involved leveraging OpenLinux and Sun products for creating “powerful solutions” for Java programs.

The solutions mentioned above solved real-life problems. The solutions were not cost prohibitive, but actually reduced short- and long-term costs to the customer while increasing margins to the solution provider. It is not a matter of “if” solution providers will be providing Linux-based solutions, but “when”.

While the Internet is creating the “new economy”, Linux is enabling customers to experience it. Some customers receive these solutions by having solution providers deploy them in-house, while others are leveraging the ASP model.

Indeed the times continue to change. For those prepared to accept the opportunities by integrating Linux into solutions, it will be the best of times. For those who continue to hold on to the old paradigm and refuse to embrace new opportunities, it will be the worst of times.

email: dean.taylor@calderasystems.com

Dean Taylor (dean.taylor@calderasystems.com) has been with Caldera Systems for over four years and is currently Vice President of Marketing. He holds a bachelor's degree in business from Brigham Young University and has been involved in the computer industry for over 15 years, with emphasis on building channels and launching new technologies in the computer field. Dean's main area of expertise involves educating the Value Added Reseller (VAR) Channel on new product benefits. His Channel experience includes working for companies such as Sanyo/Icon, SoftSolutions, WordPerfect Corp. and Novell Inc. He is co-author of the book Learning StarOffice for Linux in 24 Hours.

Archive Index Issue Table of Contents

Advanced search

EasyLinux v1.2

Pedro Bueno

Issue #76, August 2000

EasyLinux, developed by Easy Information Technology, has as its main goal an easy way to install Linux.

- Manufacturer: Easy Information Technology
- E-Mail: info@eIT.de
- URL: http://www.easylinux.com/
- Price: $21 US (cd only)
- Reviewer: Pedro Bueno

In my first contact with EasyLinux v1.2, which I received as the manager of LinuxGO, a Brazilian user group, I thought it was just another Linux distribution. In spite of some bugs, it was a very good-looking distribution. In the new EasyLinux v2000, it proves to be a really stable and competitive Linux desktop distribution.

Setup and Installation

EasyLinux, developed by Easy Information Technology, has as its main goal an easy way to install Linux. It doesn't specify any minimum requirements, but I tried it on two machines, a Pentium 133MHz with 16MB RAM and a 1GB hard drive; and a Pentium MMX 233 with 32MB RAM and a 1GB hard drive. On the first machine, the installation was very slow and the final results weren't good. The opposite occurred with the second machine; the speed and ease of installation were much better. The user can choose either a boot diskette made with rawrite or a bootable CD. With the use of Frame Buffer, Vesa2.0, which is supported by most video cards, can provide a very impressive graphical installation, superior to Red Hat's and even better than Caldera's. This also

makes it possible for any MS Windows user to use EasyLinux. It offers three different installation modes:

- Beginner: for new users who can choose between an Internet or a stand-alone PC
- Advanced: for users who know more about networking and may want a LAN client
- Professional: in this option, the user can also choose between a LAN client and a server.

All three modes, except the LAN server option, can also be used in another category called Laptop, designed specifically for those who want to install EasyLinux on laptops. Each mode has a basic packet installation and is supplied with the common applications. All new applications such as StarOffice, games, editors and compilers, are available to the user after the installation is complete just by clicking the CD-ROM icon on the desktop.

Configuring the System

With the intention of minimizing use of the terminal command line, EasyLinux built a set of applications to help the user administer the system: eLILO, eFdisk, ePrinter and eHelpAgent (see Figure 3). Two items are worth special note: eRegistry and eHelpAgent.

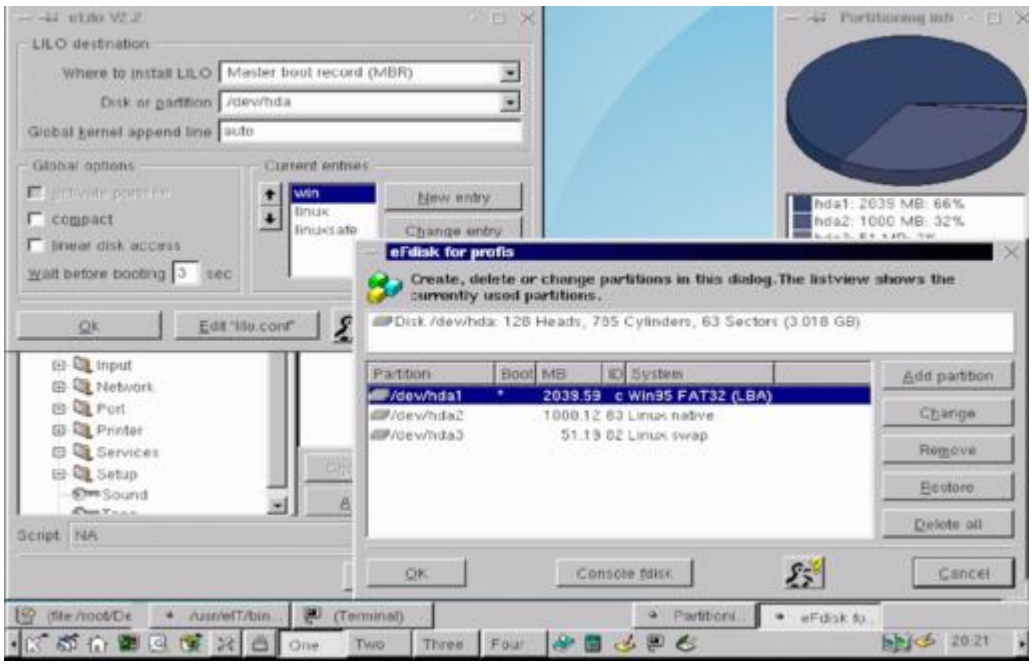


Figure 3. Administration Applications

The eRegistry application, or Regedit (see Figure 1), has nothing to do with the MS Regedit. On the contrary, it is just a system configuration application that is very useful in configuring some important system items.

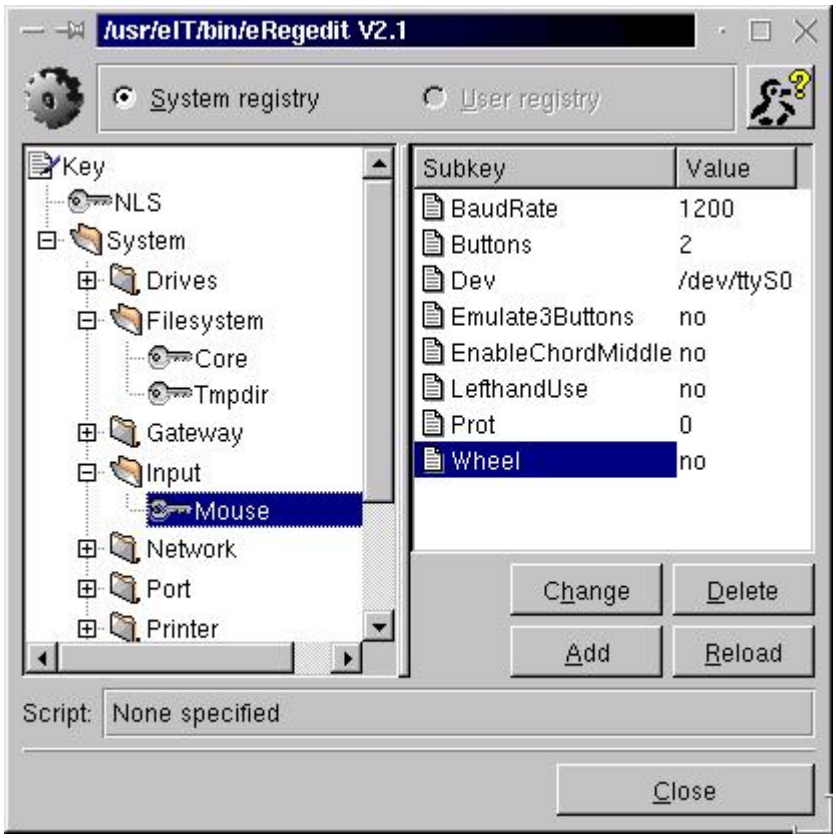


Figure 1. Regedit

eHelpAgent is another important application, because it can show information on the most common problems that may occur during the system's use. The System Window (see Figure 2) is another great aspect of EasyLinux, which acts like MS Control-Panel and gathers together all system configuration applications such as ePrinter, Hardware Setup and User Manager to simplify the config task.

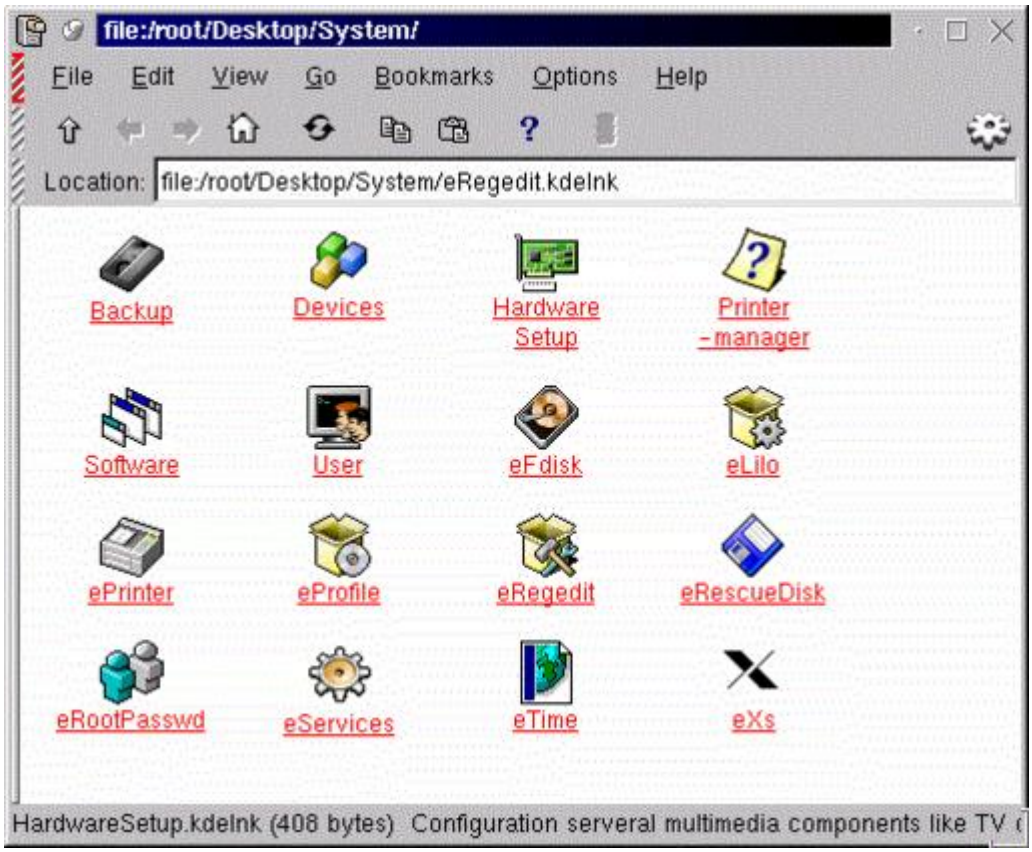


Figure 2. The System Window

Pros and Cons

As a self-proclaimed desktop Linux distribution, EasyLinux succeeds. It is really a robust operating system, very easy to install and configure, and designed for the user who likes GUI and doesn't know or doesn't want to know about console, terminal or any kind of command-line option. One benefit of EasyLinux is that it uses the .RPM (Red Hat Packet Manager) format for its packets, which provides an easier way to upgrade and install new packets. Another good point is that it is completely based on the KDE Window Manager. As I said before, there are three different installation modes, and although the results are surprisingly good in all modes, I didn't like the LAN server mode and imagine most current administrators wouldn't like it either. The reason is simple: as system administrators, we need the freedom to configure our systems in the way we need and like, but with EasyLinux, we are dependent on the GUI and its applications. As stated on the site, EasyLinux “concentrates on those users who want to apply Linux as a desktop workstation or as a PC (of course working with the shell is still possible).” In this aspect, it is very well done.

Conclusion

Nowadays, with the discovery of Linux as a great and powerful operating system, a whole new market has been discovered: the desktop. All the existing Linux distributions are improving their systems in order to establish a friendly

interface with new users. New distributions are being created specifically to attract new users who want to experiment with Linux, and EasyLinux is a good example of this. It offers a fast and easy installation mode that allows both the novice and experienced user to begin to use Linux. If you want to try it, I suggest taking a look at a free download of EasyLinux v2000 from their site.

Pedro Paulo Ferreira Bueno (pedro.bueno@persogo.com.br), a computer science student from the Catholic University of Goias (UCG-Brazil), is the manager of LinuxGO, the Goias Linux User Group and the network card moderator at Linux Knowledge base. He has been a maniacal Linux user since kernel 2.0.7. When not in front of his Linux machine, he is probably playing soccer.

Archive Index Issue Table of Contents

Advanced search

PCI Symphony Network Cards

Denny Fox

Issue #76, August 2000

I have finally found an affordable wireless solution that works with Linux.

- Manufacturer: Proxim
- E-Mail: sales@proxim.com
- URL: http://www.proxim.com/
- Price: $120 for PCI-card, $130 US for PC-card
- Reviewer: Denny Fox

Recently, I integrated a wireless segment into my SOHO (small office, home office) network based on widely available and relatively inexpensive cards from Proxim, their new Symphony series. After watching the wireless networking developments for quite a while, I have finally found an affordable wireless solution that works with Linux. Until now, wireless network cards cost hundreds of dollars and hubs cost thousands.

My goal was to give my notebook computer running Win98 the ability to move about my SOHO. I was successful in accomplishing this at a cost of only $250 by utilizing my existing Linux server and two Symphony cards.

I use Linux as a multi-function server with Samba and dial-on-demand ISDN to provide masquerading for all the machines on my internal network. The Linux server is connected to the internal network on its eth0 interface. I added a Proxim PCI Symphony card to this box, which is now up and running as eth1. (See Figure 1.)

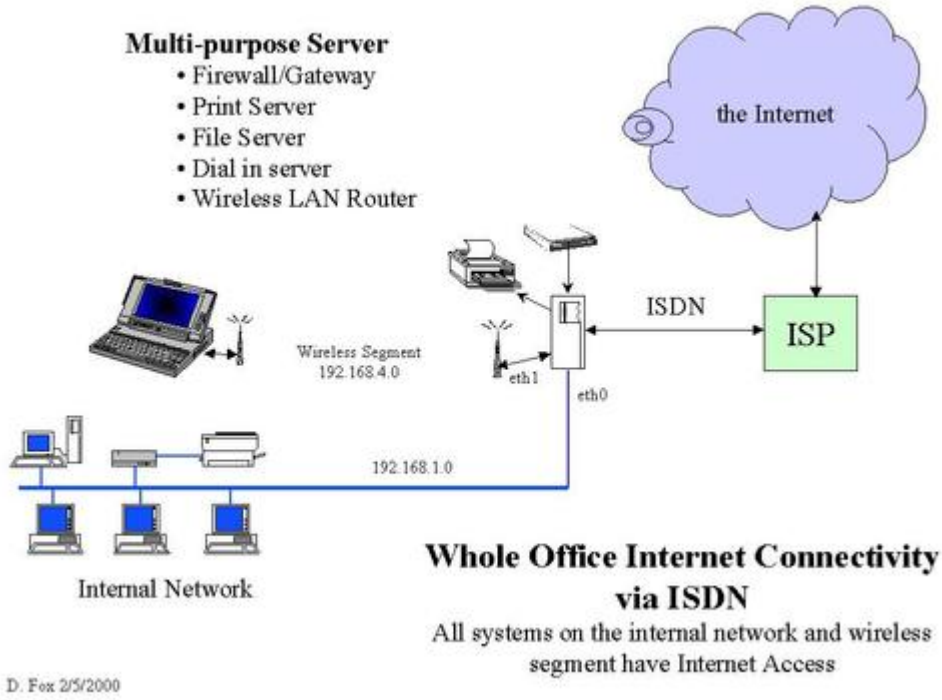


Figure 1. Network Diagram

Now that all the networking is set up, I have full connectivity to my internal network and to the Internet, and am able to work anywhere in the house. There is no apparent reduction in performance browsing the Web or e-mailing, since the bandwidth of the wireless segment is not the constraint. An FTP download from the server to the notebook transferred a 560KB file in about nine seconds, for a throughput of just under 64KB per second. It's not quite like a wired Ethernet, but it's also not too shabby.

The Challenge

My home has finished ceilings downstairs where my SOHO is located, making it very inconvenient to pull network or phone wires to other areas. I also have visions of being able to work outside on the deck when the weather permits. Clearly, some form of wireless networking would be ideal. I had been watching the wireless networking offerings off and on for the last couple of years, but prices were just too high.

The Answer

Recently, I came across the Proxim “Symphony” line of wireless networking products. They use the 2.4GHz technology found in cordless phones to achieve a raw throughput of 1.6Mbps. This gives performance that is comparable with the ARCnet I ran before Ethernet became ubiquitous. Even better, the product line is widely distributed and is reasonably priced.

Available Components and Sources

Proxim produces a full complement of wireless components in the Symphony line. There are cards for both the ISA and PCI bus, and a PC-Card for notebook computers. They also offer a Wireless Bridge and Wireless Modem for users who need them. I found the interface card products on the shelf at the local Best Buy, and the full line of products on-line at OnSale/Egghead and AtCost. There are many other sources such as CDW, HardwareStreet.com, Office Max, Office Depot and Staples. I ordered a PCI card for the Linux server and a PC Card (see Figure 2) for my notebook from OnSale's AtCost for about $120 US and $130 US respectively, with free shipping. As you will see below, those of us fortunate enough to employ Linux as our Internet connectivity server can avoid purchasing the Wireless Bridge for about $370 or the Wireless Modem for about $225.

Figure 2. PC Card with Plug-On Antenna

Technical Resources and Help

The Proxim web site at http://www.proxim.com/symphony/index.htm has a complete set of information including on-line manuals, software, FAQs and tech tips. Under operating system support, they point you to Linux drivers from www.komacke.com/distribution.html. Proxim does not directly support Linux, but I had great help from Dave Koberstein, the rl2 driver developer, and other folks on the rl2-library mail list. You can easily subscribe at www.komacke.com/maillists.html. The rl2 part of the name comes from the original and much higher-cost Proxim product line RangeLAN2. The mail list was instrumental in answering several basic questions I had regarding the usefulness of these cards under Linux and getting the driver working.

What the Data Sheets Don't Explain

The Proxim data sheets did a fine job of describing the products and functionality for use in a Windows NT environment, but did not provide some of the gory technical details I needed. Proxim delivers a layer of peer-to-peer networking setup software to configure and operate their products that insulates the user from the Microsoft networking properties dialogs. My questions were:

- Did these cards act like standard Ethernet interface cards to the operating system? Yes, the cards present a standard Ethernet interface to Windows and Linux. Under Win9*x* and NT, you use the standard Microsoft network property dialogs to set up and control the wireless network card.
- Could standard network routing be used on a Linux server to eliminate the need for a costly Wireless Bridge to provide the connectivity to the rest of an installed Ethernet? Yes, once a Proxim wireless card is installed and set up on the Linux box, standard TCP/IP routing provides the connectivity to the other Ethernet hosts and the Internet.

Thanks to the Linux driver mail list, I had enough information to proceed.

Network Architecture

The existing network operates on the private class C network 192.168.1.0. The Linux server provides IP masquerading for all systems on the internal network through an **ipfwadm** firewall and **diald** support for an ISDN connection to the Internet. I am currently running Debian 2.0 with kernel 2.0.34 on the server. I chose to use 192.168.4.0 as the network number for the new wireless segment.

PC Card Installation on Windows

Installation of the Proxim drivers on the notebook is quite straightforward. Make a note of the Secid you use, since this will need to be used for the Linux driver and any other nodes that will be on the same wireless segment. After the Proxim installation is done, go to the “Microsoft Network Neighborhood | Properties” to bring up the Network dialog box. On the Configuration tab, select TCP/IP Symphony Cordless PC Card Adapter—Properties to set the network parameters, using the appropriate addresses for your system, as follows:

```
IP Address:
     IP address:         192.168.4.2
     Subnet Mask:        255.255.255.0
WINS Configuration:
     Enable WINS Resolution Add 192.168.1.254 as a WINS server
Gateway:
     Add 192.168.4.1 as a new gateway
DNS Configuration:
     Enable DNS
     Enter a Host name and your Domain name
```

```
        Enter one or more IP addresses of your DNS server(s)
        Add Domain Suffix Search Orders as desired
```

The “new gateway” entry will be the IP address of the eth1 interface on the Linux box.

I use the domain name of my ISP and their DNS servers. You will need to use entries that correspond to your network and ISP. With diald, the first DNS query for a new connection brings the line up to the ISP. I chose to use a different host name for the notebook, depending on whether the Symphony card is active or if the regular wired 10/100 PC Card is active. When on the wireless network, the notebook is on the 192.168.4 segment; when it is on the wired Ethernet, it's on the 192.168.1 segment. I use different host names so I can state them explicitly in the /etc/hosts table on the Linux server.

On the Identification tab, set the computer name, workgroup and an optional computer description. Use the same name you used in the DNS setting above, and the name of the workgroup for your Microsoft networking. This will allow you to browse the shares on your Linux Samba server if the workgroups match. Once again, I made sure that the notebook, when on the wireless interface, had a different name than when it was on the wired network, since its IP address will be different.

Linux Installation

First, you need to fetch the driver from the ftp.komacke.com/pub/rl2isa-drive site. If you are running a 2.0.3*x* kernel, I would recommend trying the rl2-1.5.1.tgz version. As of this writing, the latest version, rl2-1.5.3.tgz, had changes to support 2.2.*x* kernels, but broke for the older kernels. A kind soul from the driver mail list helped identify the version that worked for me. The Symphony driver is used as an installable module, so you must have module support compiled into your kernel.

Place the .tgz file in a suitable directory, and unpack it with:

```
  tar -xvzf rl2-1.5.1.tgz
```

This will create a tree of directories under the current directory. I like to use /usr/src for source files, so I ended up with the driver source tree in /usr/src/rl2-1.5.1.

The rl2-version directory contains very good README files explaining the installation process. The examples and defaults are for Red Hat Linux, so I needed to answer several of the prompts differently for my Debian Linux host during the **make config** step.

For Debian, I answered as follows:

```
Module location?     /lib/modules/2.0.34/net
Executables?             /usr/local/bin
Man pages?               /usr/man
System header files?     /usr/src/kernel-source-2.0.34/include
CardType default []:     5
PC Card support? []:     n
```

Be especially careful with module location and system header files. Until I got them right, the driver module produced would not load. The README file gives the CardType values; 5 is PCI.

Next, do **make modules** followed by **make modules_install**. If the driver module compiles without error, you should be able to do an **insmod rlmod** and see the initialization message on the console and in /var/log.messages. You can confirm that the module has been loaded by doing **lsmod**. Since I used a PCI card, the I/O address and interrupt were set automatically when the driver module loaded.

Setting Up the eth1 Interface

Establish the IP address for the eth1 interface by typing

```
ifconfig eth1 192.168.4.1 broadcast 192.168.4.255\ netmask 255.255.255.0
```

Set the Secid for the Symphony wireless segment with the following command:

```
rl2cfg dev eth1 secid
```

Use the same Secid you used when setting up the notebook above, making sure you type it the same, including upper/lowercase. The Secid needs to be set only once; it is stored on the card in **nvram**. The rlmod driver versions after 1.5.1 use **proxcfg** instead of **rl2cfg** to set parameters for the Symphony card. Check the man page and READMEs.

Now, establish the route for the eth1 interface by typing

```
route add -net 192.168.4.0 dev eth1
```

You can check your results by typing **ifconfig eth1** and **netstat -r**. You should be able to see that the eth1 interface is fully configured with its IP address and that there is a route using eth1 to the 192.168.4.0 network.

There is one more configuration command that sets the Symphony card mode to “master” and sets its name. Type the following, using an appropriate name for your server:

```
rl2cfg dev eth1 msta
```

If you don't have a firewall running on your Linux system, you should be able to ping the notebook from Linux and ping Linux from the notebook using the IP addresses. You should also be able to telnet to the Linux system from the notebook, again using the Linux server's IP address. If this works, you're just about there; if not, check your log files for errors.

Finishing Up the Setup

Add a line to the /etc/hosts table on the Linux server, giving the IP address of the notebook when on the wireless subnet, using your domain as appropriate.

```
192.168.4.2    wnb-dfoxwnb-dfox.
```

If you don't already have a c:\windows\hosts file with an entry for the Linux server on your notebook from a wired Ethernet configuration, create one now. The pertinent line entry in my c:\windows\hosts is:

```
192.168.1.254   server1       server1.
```

If you have a firewall running on the Linux server, you will need to add input and output rules for the eth1 interface to allow the packets to be passed. I added lines to these two sections of /etc/rc.boot/masq as shown:

```
# local interface, local machines, going anywhere
# is valid
ipfwadm -I -a accept -W eth1 -S 192.168.4.0/24\
  -D 0.0.0.0/0
# local interface, any source going to local net
# is valid
ipfwadm -O -a accept -W eth1 -S 0.0.0.0/0 \
 -D 192.168.4.0/24
```

Depending on your Linux distribution and version and whether you are using ipfwadm or **ipchains**, the location and name of the rules file will vary. Refer to the documentation for your firewall package. At this point, you should be able to ping and use telnet by host name between the notebook and all systems on your original network.

To allow packets to be forwarded to the Internet through the firewall from the wireless segment, you will need to add rules similar to the following:

```
# Masquerade from local net on local interface to
# anywhere
ipfwadm -F -a masquerade -W eth0 -S 192.168.4.0/24\ -D 0.0.0.0/0
ipfwadm -F -a masquerade -W ppp0 -S 192.168.4.0/24\ -D 0.0.0.0/0
ipfwadm -F -a masquerade -W sl0 -S 192.168.4.0/24\ -D 0.0.0.0/0
```

Due to the way diald works, there are rules for both ppp0 and sl0. Remember to restart your firewall rule set after making changes.

Once I had the firewall rules in place, I had full connectivity from the notebook to all local hosts and the Internet. The last service to get working was the

Samba sharing from the Linux server. It took a little head scratching, but finally I realized I had a line in /etc/smb.conf that limited which networks had access to the Samba server. Since the wireless segment was new, it needed to be added to the config file, and then Samba needed to be restarted. The revised line in /etc/smb.conf is shown below.

```
hosts allow = localhost, 192.168.1., 192.168.4.
```

Integrating with Linux Startup Code

Once everything is working, you need to add the commands to initialize the wireless eth1 interface on system boot. Debian Linux uses SysV init, so I added the following lines to the /etc/init.d/network file after the eth0 interface section and before the line setting the default route:

```
insmod rlmod
ifconfig eth1 192.168.4.1 netmask 255.255.255.0\
  broadcast 192.168.4.255
route add -net 192.168.4.0 dev eth1 \
  /usr/local/bin/rl2cfg dev eth1 msta name server1
```

These lines load the module, configure the eth1 interface, add the route and set the mode to master. Make sure it all comes up on reboot.

Testing and Performance

When on the wireless segment, the notebook has full connectivity to the rest of the local network and to the Internet; it's just a little slower. The Proxim specs state a range of 150 feet. I have had no trouble with the connection anywhere within our house. I've measured FTP performance for a half-megabyte file at about 64Kbps download and about 56Kbps on upload between the notebook and Linux server. I am pleased with this performance, since I don't plan on transferring large amounts of data while on the wireless segment. When I need to do that, I simply take the notebook back down to the office and switch it back to the wired Ethernet interface. During normal Internet browsing and e-mailing, there is no noticeable performance degradation.

Switching from Wireless to Wired and Back

Proxim supplies a handy feature in the Maestro utility for switching back and forth between a wired and wireless Ethernet interface. The Maestro utility is installed as part of the Win9*x* setup, and normally resides at the bottom right side of the taskbar. If you double-click the Maestro icon, click Configure, then click the Network Profiles tab, you will be there.

Win9*x* can store only one set of many of the network parameters, even though there may be more than one network interface and environment. Maestro creates a couple of new registry sections, one for the “Original” network

settings (**profile0**) and one for the “Symphony” network settings (**profile1**). This makes it very easy to switch between a wired Ethernet interface and a wireless Symphony card on your notebook. The alternative is to change settings on several different screens manually when changing interfaces.

After fiddling with it for a bit, I realized it saves the necessary network parameters for profile0 when you change to profile1 and click “OK” and vice versa. This makes it relatively easy to get the settings the way you want them without actually changing the cards and rebooting each time. You can click back and forth as many times as you want. For instance, select the “Symphony” profile, click “OK”, then use the Windows network properties dialogs to make changes. When finished, click the “OK” on the Microsoft dialog, but don't reboot. Then go back to Maestro, click the “Original” profile, then “OK”, and your profile1 settings will be saved. If you don't let it shut down your system, you can go back and forth until you get the settings you want for each profile.

When you actually want to switch cards, run Maestro and select the correct profile for the card to which you want to switch. Click “OK”, then confirm the box that offers to shut down the system. After the system powers down, insert the correct card and power it up. It will have all the appropriate settings for that interface card and environment. If you are curious and want to see everything it saves, use **regedit** to look in HKEY_LOCAL_MACHINE\Software\ProximSwitch.

Expanding the Concept

Proxim states that you may have up to ten machines on a wireless segment. You may also have several wireless segments within the same physical area by using different Secids for each wireless segment. For some installations, this may be very useful. Dave Koberstein reports that the ten-unit limit is based on limitations of the Proxim Wireless Bridge and Wireless Modem. If you use Linux as the wireless router, you are limited only by what you consider acceptable performance and how many nodes you can fit into a 150-foot radius.

There are some exciting new developments for 2.2.*x* kernel versions and the later rl2 driver versions. Bridging is reported to work with kernel 2.2.14 and rl2-1.5.3. The configuration is more complex, but you can avoid the need for different names and IP addresses when on wireless. The next driver, rl2-1.6.0, will have improved reporting, improved troubleshooting capability and will show up in the Win9*x* Symphony tools network map. Subscribe to the mail list to check these out.

Summary

After using my notebook on a wireless segment for a couple of weeks now, I must say I am quite pleased. I have good performance at a reasonable price

and am able to configure and operate the wireless network with standard networking tools. I have great expansion capabilities and have avoided purchasing a costly Wireless Bridge. There is a great deal of help available, both from the Proxim documentation and web site and the Linux driver web site and mail list. This solution meets my needs quite nicely.

Acknowledgements

The Good/The Bad

Denny Fox (dennyf@mninter.net) has been active with designing hardware, software and auto test equipment since the late '60s. Linux caught his attention back at kernel 0.97. When not hacking on something, Denny enjoys hiking, sailing, reading and playing guitar. He is the president of Micro Time, Inc.

ImageStream IS Gateway and Rebel Routers

Jon Valesh

Issue #76, August 2000

Gateway and Rebel Routers from ImageStream Internet Solutions: heavy-hitting performance, lightweight price.

- Manufacturer: ImageStream Internet Solutions
- E-mail: info@imagestream-is.com
- URL: http://www.imagestream-is.com/
- Price: under $8,000 US
- Reviewer: Jon Valesh

Gateway and Rebel Routers from ImageStream Internet Solutions: heavy-hitting performance, lightweight price.

Or, if not lightweight, at least inexpensive, compared to their mainstream competition. ImageStream's backbone routers spring from a simple premise: take GNU/Linux, with all of its proven networking abilities, and package it in an industrial-strength wrapper. Simple premise, perhaps, but far from a simple accomplishment. Fortunately, ImageStream has had plenty of practice. Founded in 1993, they have been building Linux-based servers and routers long enough to know what is important.

Packing support for a wide range of physical interfaces, up to T3 and beyond, and enough expansion room to cram over twenty T1s, multiple Ethernet and a couple of big WAN connections in a fully loaded system, ImageStream has taken their simple premise and made it into a useful reality. Dynamic routing with RIP1, RIP2, OSPF and BGP4, bandwidth management, firewall support and seamless interconnectivity with Cisco and other major name routers make the accomplishment even more impressive. A price point well below Cisco's makes their reality compelling.

In order to build their routers, ImageStream first created a Linux distribution: one that maximizes network functionality while minimizing hard-disk space requirements so it fits nicely on a solid-state flash hard disk and has easy-to-use administration tools. You don't need to be a Linux guru to set up or maintain the system. Then, they developed and released, under the GNU GPL, an open-source Linux kernel extension called SAND, Standard Architecture for Network Drivers, which allows binary distribution of WAN hardware and protocol drivers. SAND is a compromise between the open-source ideal of always having open source, and the closed binary-only drivers that have caused so much trouble in the computer industry. It is a necessary compromise for routers, because many WAN drivers need to contain code or protocols licensed from companies that don't support open source.

The most significant feature of SAND is its ability to mix and match binary protocol, WAN card and even external CSU/DSU drivers from different manufacturers. This means a protocol developer can release a driver and have it work with any WAN card that uses SAND drivers, and a WAN card producer can release a driver to support its hardware knowing the various protocols will be supported. SAND also provides a standard method for configuring and monitoring varied WAN interfaces, simplifying performance and bandwidth monitoring across multiple interface types.

ImageStream's compact and network-heavy Linux distribution, when married to an industrial-quality rack-mount PC, becomes a high-power router fully capable of meeting the backbone routing requirements of ISPs, schools, businesses and anyone who's ever considered the benefits of a private T3 Internet connection to their home. Despite their PC heritage, these routers give a first impression far removed from the usual desktop-PC chintz.

I looked at two of ImageStream's routers, the Gateway and the Rebel. The main difference is in the physical size and level of expansion. The Gateway has twelve available PCI slots for WAN or LAN cards; the Rebel has four. Which router is right for a particular network application will depend more on current need and projected expansion than on differences in core functionality. Though it was not tested, parts of this review also apply to the ImageStream Enterprise router, the biggest of the line with 18 card slots.

The Hardware

Both of the routers tested are based on a single-board computer with integrated 100MB Ethernet. Additional network interfaces are added using PCI WAN and LAN cards, which plug into a passive backplane. Interfaces range from V.35 ports to HSSI and ATM, and can be pre-installed or user-added. The chassis are standard industrial 2U and 4U rack-mount PC cases and are available with dual hot-swappable power supplies for an additional $500 US.

Adding interface cards to the Rebel requires removing fifteen screws, which must be a record. A far more reasonable four screws grant access to the Gateway. In either case, once opened, access is excellent.

Documentation

Included with the ImageStream router is a slender manual made up of a quick-start guide, a configuration guide and an appendix (the largest part of the manual) filled with handy information, such as cable-wiring diagrams, subnet-mask tables and troubleshooting information. At first glance, the manual seems downright sparse, and a second glance will confirm that impression. However, the manual does one thing beautifully—it tells you what you need to know. It assumes you know how to mount the chassis in your rack and which end of the cord plugs in the wall. It tells you how to access most of the unique features of the router and how to fix the things that are likely to go wrong. It is hard to complain about that, but compulsive manual collectors will be frustrated by how unimpressive the ImageStream manual looks next to a stack of Cisco books. The most noticeable hole was in the instructions for configuring dynamic routing—there were none. If you intend to use BGP, RIP or OSPF, a book on configuring GateD or a visit to http://www.gated.org/ will be essential.

Setting Up

Configuration can be done in one of three ways. By connecting a standard VGA monitor and PC (AT or PS/2) keyboard, you can treat the router just like the Linux PC it actually is and watch the BIOS POST, see kernel messages during boot and even switch virtual consoles. If you have a monitor and keyboard lying around, this is definitely the way to go; if you don't, you can log in using TELNET, **ssh** or an RS-232 terminal. Without a monitor, you can't watch the system boot messages, so if something is wrong with the hardware or kernel, you'll have a harder time fixing it.

A simple text-mode menu system provides the primary user interface. From the menu, you can view the system status, edit the configuration files, back up and restore the configuration, change passwords, access a **bash** shell and do most routine maintenance tasks. The menu options represent only a subset of the abilities of the router, but a useful subset. If you want to do anything fancy, including most troubleshooting, you will need to leave the menu.

Both routers have bandwidth-limiting and firewall capabilities. Configuring bandwidth limiting will require a brief trip outside the configuration menu; the process is well-documented. You can edit the firewall configuration file directly from the menu, but there is no documentation explaining why or how you would do so.

Most routers use one of three basic configuration schemes: interactive command-line interfaces, form or template interfaces and configuration files. There is no best way—each has its own strengths and weaknesses, and most people prefer whatever they've worked with the most.

Conventional Linux network configuration is an example of an interactive command-line interface; commands like **ifconfig** and **route** change the state of the kernel networking subsystem while it runs. With Linux, the commands must be executed each time the system boots in order to set up the network. Router manufacturers often take things a step further by using commands to set up the active state of the router, then directly saving a snapshot of the active configuration into flash so that when the router reboots, the commands do not need to be re-run. Interactive command interfaces can be disconcerting if you don't plan properly or are a slow typist. Changes go into effect as you enter them, so complex configuration changes can leave the router non-functional while you enter your new configuration line by line. Mistakes and typos are usually easier to track down, though, since the error is usually in the last line you entered, and you will immediately see any error messages.

Template interfaces are most often used when only a limited number of possible configurations exist. They are usually the easiest to set up, but if the interface designer didn't anticipate your needs, you can be left hanging.

ImageStream departs from the Linux norm by using a configuration file for the LAN and WAN interfaces, which goes beyond the startup scripts common to all Linux distributions. A single configuration file contains information on all physical and logical interfaces, including both interface-specific parameters and IP configuration. The router, after a brief sanity check to make sure you didn't tell it to do anything too silly, puts the configuration into effect. The advantage is that you can spend as much time as you like modifying the configuration and activate it in just a couple of keystrokes—or not, if you chicken out. The ImageStream routers end up with a mix of interactive command-line and configuration-file interfaces, because the standard Linux networking tools and some of the ImageStream-specific features, such as bandwidth limiting, are still command-driven.

The file format ImageStream uses to set up the network interfaces is unique but very easy to follow and well-documented. Each interface has its own subsection with information such as IP address, netmask, static routing information, baud rate and port-type specific parameters, and the file provides a logical framework for maintaining the interface configurations. Each interface type has its own quirks and limitations, although ImageStream has standardized the configuration format and does its best to hide the differences, making configuration of WAN interfaces easier. A SAND reference guide, also

part of the manual, provides all information you need to set up each interface type.

Other than setting up the WAN interfaces, configuring the routers is just like configuring any other Linux box with similar features. Dynamic routing is handled by GateD, which has its own set of configuration files. Setting up GateD is fairly easy, and complete instructions and sample configuration files can be found at the GateD web site.

If, once everything is up and running, you expect to make only infrequent, major changes to the router configuration, files win hands down; they provide a complete view of the changes you are making, and they minimize down time. Unfortunately, each time a file is changed, there is a chance that a typo or minor error will take down *everything* until you find and correct the problem. Interactive-command interfaces shine when you are working with large, complex setups that need frequent minor changes, e.g., routers with numerous frame-relay permanent virtual circuits. The effect of configuration changes is usually more local, reconfiguring only one circuit or feature, and likewise, typos and bad configurations have a more local effect.

The solid-state hard disk imparts some quirks to the system and will seem a bit strange at first. The system boots from the “hard disk” (which is actually a 40MB SanDisk flash ROM device) but loads the active files into a 16MB RAM disk and runs from there. This imposes limitations similar to systems that boot from a CD-ROM drive: you can make changes to the system, create files, etc., but your changes are temporary. A menu option allows you to save all of your changes after you have verified they actually work. This means bad configuration changes can be undone by rebooting. It also guarantees that, unless you reboot while saving the configuration, the system will never corrupt its file system due to an improper shutdown. But, it can catch those unaware and cause frustrating, too-late realizations that “now was not the time” to reboot. The frustration of helplessly watching the router boot up with no memory of the elaborate configuration you just spent the last hour working on can be acute.

Operation, Maintenance and Troubleshooting

Testing the operation of a high-end router without having a high-end use for it can be a serious challenge. Fortunately, ImageStream was kind enough to provide a pair of routers so I could get a feel for setting up high-speed circuits. In all of my testing, I was never able to heavily load either router noticeably. In fact, both routers took everything I could throw at them and didn't blink once.

Regular configuration file backups will be easy for people with physical access to the routers. The menu-based backup option will use the floppy drive only to save or restore. An administrator could perform his own network backups, of

course, but an option to generate an archive file and store it using FTP would be nice. If you do not expect to do a lot of reconfiguring, you won't need to do a lot of backups, and the biggest maintenance hassle will be keeping the air filters clean—a hassle I could live with quite easily.

Tracking down problems when something does go wrong may be a bit of a headache. You will need to exit the menu in order to diagnose all but the simplest problems, but unlike mainstream routers, knowledge of Linux networking will actually help. The system and network interface status can be read and viewed using standard commands such as **ps**, **uptime** and **ifconfig** and exploring the /proc file system, but some of the most important information is unavailable. For example, you can read the current active route tables using the **route** command, but the router cannot tell you where the routes came from. Tracking down a BGP or OSPF configuration problem is going to be like dancing blindfolded—possible, but best done without an audience. ImageStream includes 24/7 tech support with their routers, so you'll always have a dance partner available and that should be a real help. The latest release of GateD includes new routing status monitoring tools, and when ImageStream updates their system later this year, the diagnostic capabilities will take a quantum leap forward. As it is, the limited diagnostic information available is the worst feature of these routers.

And, in the End...

The Gateway and Rebel routers, as tested, performed every task set before them flawlessly. While there is room for improvement, especially in the area of feature documentation and diagnostic information, their functionality and features cannot be denied. If you know Linux and are looking for a high-end backbone router, the ImageStream routers should be on your list—or in your rack.

The Good/The Bad

Born at the beginning of the microcomputer age, **Jon Valesh** (jon@valesh.com) has pushed and been pushed by computers his entire life. Having run the gamut from games programmer to ISP system/network administrator, he now occupies himself by providing technical assistance to ISPs and small businesses whenever his day job doesn't get in the way.

Archive Index Issue Table of Contents

Advanced search

FreeNetshop

Kevin Lyons

Issue #76, August 2000

FreeNetshop is an on-line e-commerce, sales and order-tracking software suite with a customer interface modeled on the popular “shopping cart” theme.

- Manufacturer: FreeNetshop
- E-mail: coretm@freenetshop.org
- URL: http://www.freenetshop.org/
- Price: As the name says, it's FREE!
- Reviewer: Kevin Lyons

FreeNetshop is an on-line e-commerce, sales and order-tracking software suite with a customer interface modeled on the popular “shopping cart” theme. It is designed to run on any UNIX system using any NCSA-compatible web server, including Apache. FreeNetshop is, of course, free and is licensed under the GPL.

FreeNetshop (FNS) works with the freely available mSQL relational database and is capable of running multiple independent copies on one server using a single instance of msqld. This makes it ideal for web presence providers and ISPs who serve multiple customers/domains from each system in their server pool. FNS has been tested on version 1.0.9 and 1.0.16 of mSQL. Porting to mSQL 2 and MySQL is underway.

FNS uses the mSQL C programming API, the excellent CGIC library from Thomas Boutell, and UNIX library calls to optimize speed and minimize runtime size. Independent configuration parameters are compiled at installation to reduce lookup overhead during execution. FNS is also customizable from site to site by editing a few HTML pages.

The FreeNetshop distribution is composed of Common Gateway Interface (CGI) programs and HTML forms. FNS was designed from the ground up to allow

multiple sites to be hosted on a single host machine. As such, FNS CGI programs are small and fast.

Performance of an FNS installation is primarily a function of the speed of the database (mSQL) and the web server. mSQL was chosen as the database since it has defined the performance mark for small to medium-sized relational databases and is easily obtainable. For example, an indexed lookup of one record in a table with 100,000 records took less than half a second, including web server and client response, on an AMD 586, 133MHz machine with 16MB RAM. A non-indexed search for a string in a 50-character field for 100,000 records took less than three seconds on the same machine.

Features

FreeNetshop version 1.3 features:

- Small, fast execution using compiled ANSI C
- Cookie-based user tracking
- Sales tax calculations
- Automatic HTML page generation with a customizable header and footer for all products in the database
- Shipping cost calculations for up to five classes of UPS shipping
- Password verification
- Support for public terminal account cleaning
- Both virtual and non-virtual host operation
- Support for SSL operation
- Small size: less than 424K in gzipped, tar file; installed to around 5MB; executables average less than 95K
- E-mail order notification
- Web browser-based administration
- Real-time sales reporting

Table Relations

A view of the FreeNetshop table schemas reveals that the database has not been fully normalized. This decision was made to minimize table lookups at specific points in the application CGI programs. Of course, the effect of this decision is to sacrifice some small amount of storage space to achieve faster response. Figure 1 shows the table relationships. It can be seen that customers may have one-to-many and one-to-one relations with Cart and Orders. Orders, in turn, can have one-to-many or one-to-one relations with Lineitems. Lineitems and Cart are one-to-one related to Products. Products (i.e., a product

record) will have a many-to-one relation with Lineitems and Cart, since one product number will be in multiple shopping carts and order line items.

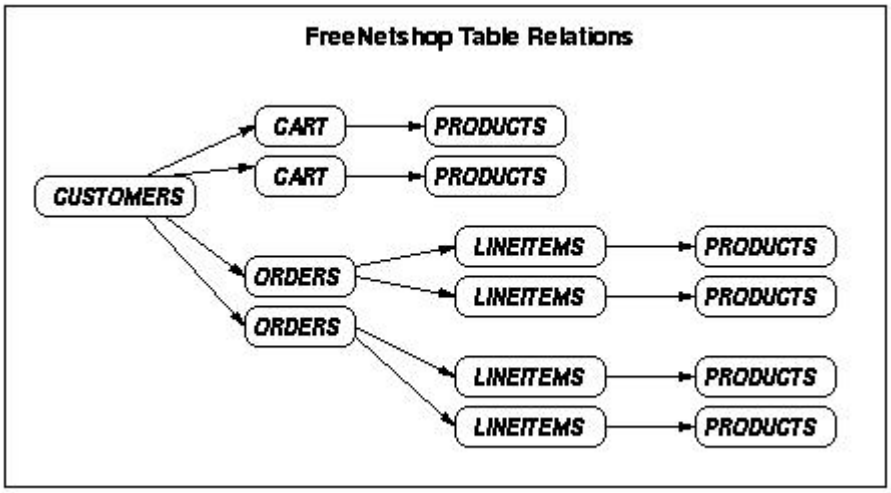


Figure 1. Table Relationships in FNS

Program Flow

The FNS distribution is divided into customer-level programs and administration-level programs. Customer programs are located in the site cgi-bin directory. Administration programs are located in admin/cgi-bin for each site. The admin directory is protected with the **htaccess** facility included in most NCSA-compatible web servers, including Apache.

Customer-level programs are linked into the HTML pages and each other so that the process of adding items to the shopping cart and placing an order is completely menu-driven. Nevertheless, it may be useful for the hosting ISP or administrator to understand the ordering process as it relates to the cgi programs. The customer-level programs are search.cgi, addcart.cgi, viewcart.cgi, upcart.cgi, preorder.cgi, ordercalc.cgi, order.cgi, clearcust.cgi and account.cgi. Figure 2 shows the customer ordering sequence.

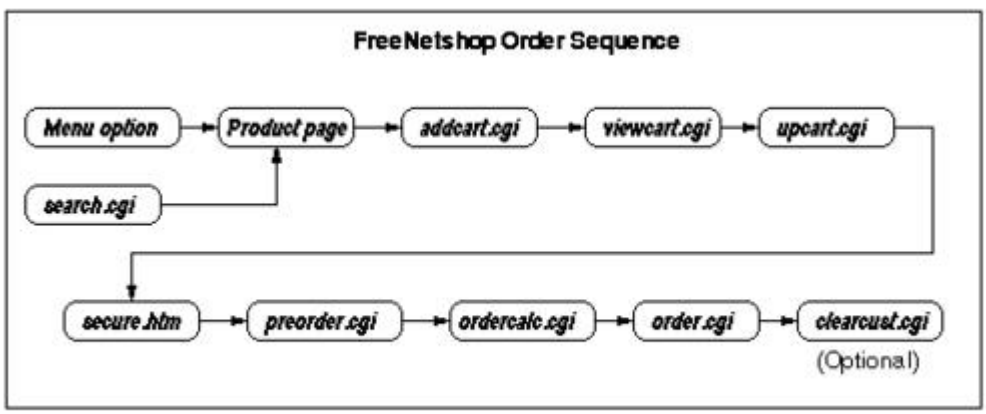


Figure 2. Customer Ordering Sequence

search.cgi provides the means for the customer to enter keywords to search the products database table. search.cgi prints out a listing of all matching records (products) with a short description and a link to product HTML pages. In addition, search.cgi pulls in header and footer HTML files so that the dynamically generated results page appears within a consistent-look site interface (refer to the CUSTOMIZE file included with the distribution for details).

addcart.cgi adds a product to the customer's shopping cart. Internally, addcart.cgi sends a unique cookie to the customer's computer and adds the appropriate product number to the Cart database table. In addition to the standard product number, price, etc., a sequence number is added to the Cart table. The sequence number is based on the date and time the item was added. This provides a means for the site administrator to remove old entries from the Cart table. There is no limit to the amount of items a customer may add to his shopping cart.

viewcart.cgi pulls up a listing that shows all product items in the customer's shopping cart. Additionally, quantities are shown in a form field so that the customer can change quantities or remove products from his shopping cart.

upcart.cgi performs the actual update of product quantity in the Cart table, then regenerates the shopping cart listing page with the updated quantity shown. Links are provided in viewcart.cgi and upcart.cgi to allow the customer to proceed with placing an order.

preorder.cgi was added to FNS 1.3 when shipping costs were added for five classes of UPS. preorder.cgi takes address information from the Customer table if the customer has ordered before, and provides blank form fields if the customer is new.

ordercalc.cgi takes the shipping information and address data entered from preorder.cgi and calculates order cost and shipping. order.cgi then provides form fields for the customer to enter credit card information. If the password the customer entered into preorder.cgi is correct, then the default customer credit card information is filled into the form fields automatically. There is also a field for the customer to enter any special comments regarding his order.

order.cgi commits the order information to the proper tables. In particular, order.cgi updates the Order tables, creating a unique order number, updates the Lineitem table and removes the selected items from the Cart table. Additionally, order.cgi creates an e-mail order notification message and sends it to the site administrator. order.cgi also provides output of the order to the customer for saving or printing. Once the administrator receives e-mail notification of the order, he can verify completeness of the order and forward a

copy of it to the customer via e-mail. The e-mail order notification can also be printed by the administrator and used as a packing list.

clearcust.cgi allows the customer to delete all entries from the Customer table that correspond to his cookie-based user ID number. This is intended as an additional privacy/security feature for customers who order from public terminals/kiosks. While a thief would still need to know the customer's password to gain access to credit card information, the shipping and contact information that was last entered on a particular terminal would appear the next time the FNS site was visited if this functionality was not provided. clearcust.cgi can be invoked by the customer from the bottom of the order completion/processed page which is generated by order.cgi.

account.cgi allows the customer to change default billing and shipping information. The customer must enter his e-mail address and correct password before being allowed to access any data.

Administrative Functions

FNS includes programs which allow complete web-based site administration. Products can be added, changed and deleted from the Web via links under the products section of the administration page. FNS also allows automatic web-page generation for products in the products database table. Access to customer data is provided, and the administrator can view, update, delete or select (by name) customer information over the Web. Likewise, the ability to view order data and update it is provided, plus sales reporting by product type and date ranges. Importing and exporting data in ASCII-delimited text is also offered.

Administrative CGI programs include executables for adding, updating and deleting data from Customer, Order and Product database tables. Most of the programs are self-explanatory, but several will be discussed below where there is a particular item of importance.

addprod.cgi adds product data to the Products tables. The data is entered into an HTML form page. The form includes data fields for a user-entered unique product identification number (prodno) plus description, name and price. Availability in days or weeks plus quantity on hand are available. A field to enter search keywords is also provided. Keywords should be entered with a space between each one; it is not recommended that special characters such as quotes or apostrophes be used. The keywords entered here are used by search.cgi to determine a match. A field for entering the URL of an image is given. The image referenced by this URL will be added to the product HTML page when generated by FNS's genppage.cgi program. The default location for images is in the /images directory. Entries should follow the format:

```
    ../images/prodimage.jpg
```

where prodimage.jpg is the image file name. The actual image can be transferred with FTP to the images directory before or after the product is added. Multiple products can also share the same image. Once the data is entered, addprod.cgi will insert the information into the Product table.

genppage.cgi generates product pages automatically in HTML format. genppage.cgi reads in the header.htm and footer.htm files and writes them to an HTML file after being combined with product data from the Products tables. genppage.cgi names the output files after the product number and creates the pages in the "products" directory. Thus, genpage.cgi creates the file products/1000.htm for product number 1000. When a product is updated and the header.htm or footer.htm files are changed, the site administrator can update the product HTML page by re-running genppage.cgi for that particular product or for all products by specifying the range (by product number). The search.cgi program provides a link to the product page for each product returned from a search.

upord.cgi can be used to update basic order information. If lineitem-level changes are required, the order can be zeroed out or voided and re-entered as a new order. When all product for an order ships, the actual shipping date can then be entered into the "Completion Ship Date" field for future reference.

export.cgi allows the administrator to export the contents of the FNS tables to text files. Running export.cgi will create the following files: dataprod.csv, datacust.csv, dataord.csv, dataline.csv, datacart.csv, datapack.csv and datahold.csv. The names of the files correspond to the contents; for instance, dataprod.csv contains all data from the Products tables. The files are created in the admin directory. Files are written in ASCII comma-separated value format with text delimited by quotation marks (sometimes referred to as ASCII-delimited text). Exported data files can be imported into spreadsheets or other databases that recognize this format.

Work on import.cgi is in progress. When completed, import.cgi will read data files exported by export.cgi into database tables. If one desires to import particular tables, for example Products, the other data files can be renamed, moved or deleted from the /admin directory. If import.cgi is used to insert data files exported from spreadsheets in comma-separated value format, it is important to ensure field lengths do not exceed the lengths allowed in FNS tables (see the createtables file for table definitions), or the data values will be truncated. For instance, the customer name field (name) shows a data type of **char(50)**. This means up to 50 characters are allowed in this field. However, the actual usable number of characters that can be entered is 48, since the delimiters must be accounted for in the total. It is also important to verify that

all text fields are enclosed by quotation marks, and the field sequence fits the field order as defined in FNS. Most spreadsheets will automatically enclose text cells with quotes when exported in CSV format.

In addition to these web-based utilities, mSQL comes with several command-line tools and a terminal monitor that allows maintenance of the database from a shell account. Detailed information on these can be found in the mSQL documentation.

All in all, FreeNetshop provides a fairly simple yet scalable and robust application on which to develop e-commerce sites.

Kevin Lyons (klyons@corserv.com) has been involved with UNIX since 1985. He has experience on SCO Xenix, NextStep, Solaris, Linux, FreeBSD and OpenBSD platforms.

Archive Index Issue Table of Contents

Advanced search

Linux and The Linksys EtherFast Instant GigaDrive

Billy Ball

Issue #76, August 2000

This product provides 20GB of instant storage to your Linux or “other operating” system LAN without the hassle of adding a keyboard, monitor or mouse.

- Manufacturer: The Linksys Group, Inc.
- E-mail: sales@linksys.com
- URL: http://www.linksys.com/
- Price: $695 US
- Reviewer: Bill Ball

The explosion of small-business and home-office networking has made on-line storage, print service and IP addressing critical issues for many installations. Enter Linksys with a Red Hat Linux-based solution, the EtherFast Instant GigaDrive. This product provides 20GB of instant storage to your Linux or “other operating” system LAN without the hassle of adding a keyboard, monitor or mouse.

Linksys has been a Linux-friendly hardware manufacturer for a long time. Most network-savvy Linux users know that Linksys was the first company to list Linux as a supported operating system on the outside of its networking products' boxes.

I recently spent the afternoon playing with this new network toy from the Irvine, California-based company. The GigaDrive is one of those new breed of small network storage boxes you plug in to a hub on your LAN, turn on, web administer, use—and fall in love with.

Hooking up this compact device to your network is a snap. In this review, I will cover some of the basics and features of this new, first-generation, Linux-based networking product.

Specifications

The box weighs five pounds, measures 8'' wide, 11'' long and 3'' high and includes a parallel port and an RJ45 jack. A small switch is used on the back when you connect the GigaDrive to a LAN hub or directly to a PC. Indicator LEDs on the front show remaining hard-drive capacity, power, status, hard-drive activity, NIC activity and whether DHCP service is being used.

The GigaDrive is basically a single-board, 45.26 BogoMIPS Pentium computer with (contrary to earlier reports on the Internet) only 16MB RAM, and a spacious 20GB drive running Red Hat Linux 2.0.36. The file system is installed on a 154MB root file system, and a 33MB swap partition is configured as a separate partition. The rest of the 20GB drive is devoted to a Linux EXT2 native file system, leaving nearly the entire drive available for storage of your LAN's or users' data.

When in use, several megabytes of memory remain free, as the majority of memory is used by normal Linux processes, such as the LDP printer dæmon, Apache, the AppleTalk **atalkd** dæmon and Samba's **smbd** dæmon. The default swap partition should be barely (if at all) used under load.

Configuration

A small, 30-page manual, Windows NT install disk and CD-ROM for Windows-based installation are included. The machine is initially configured by running, of all things, a Windows-based setup and configuration program. Why support for Linux-based configuration and installation is not included is beyond me. A convenient Win client provides initial setup, such as IP assignment and setting of the date and time.

Linksys has prepared this device as a hands-off Linux system. This means that (supposedly) your only interaction with the GigaDrive is meant to be through a Netscape web page. After connecting the unit to your LAN, pressing the power button and waiting a few minutes, you can set the GigaDrive's server name (I called mine “gigi”), system date and time, IP address, Windows workgroup name and AppleTalk zone.

You can also configure the GigaDrive as a DHCP server, in which case you'll need to specify a range of valid IP addresses. The GigaDrive can also be configured to obtain its IP addresses from another server.

After assigning an administrative password, I could manage the GigaDrive remotely, using a URL like this:

gigi/CGI_main/index.htm

Alternatively, you can specify the GigaDrive's default IP address and jump right in to remote configuration. Once you connect, a Javascript runs that changes your Netscape window and presents a setup dialog. Once the unit is configured, you'll need to use a login dialog, and then you'll see a connect window, as shown in Figure 1.

Figure 1. Connect Window

The Connect window lists the GigaDrive, presents information on your unit and offers an Administer button. Click this button to administer the GigaDrive, and you'll see a dialog with numerous buttons to configure the system and administer users or shares on the drive, as shown in Figure 2.

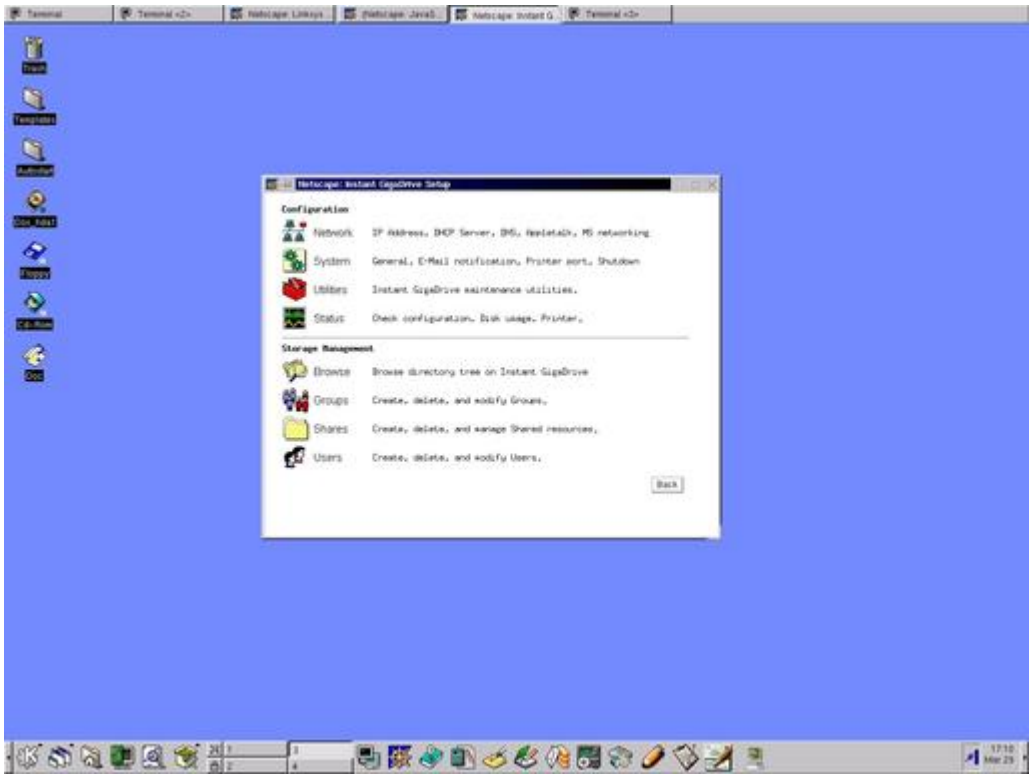


Figure 2.

The configuration buttons provide changes to networking, the system, utilities and status information. These are handy for changing the IP address, setting network features (such as turning DHCP on or off), adjusting date and time settings and troubleshooting. The GigaDrive can even e-mail you if there's a problem!

The storage management buttons allow you to browse the visible or “public” portions of the GigaDrive's hard drive and administer groups, shares and users. If you're new to Windows networking, you'll want to read the GigaDrive manual for a quick tip on how groups, shares and users are viewed by Samba, which provides networking storage and printer sharing. Plug a printer into the GigaDrive's parallel port, and you have an instant networked printer and print server. Windows users will find the GigaDrive and any attached printer under Network Neighborhood.

In Use

Although I was initially disappointed that only Samba was supported, I found that performing a Linux SMB mount was just as easy as using NFS, if not easier. With the ability to mount your remote share, there's really no reason to use FTP or another means of file transfers.

Although one normally avoids creating SUID root programs, a simple

```
# chmod 4711 /usr/bin/smbmnt
# chmod 4711 /usr/bin/smbumount
```

will allow you to quickly and easily mount public shares locally like this:

```
# smbmount //gigi/public backup
```

After entering your password, you'll find all your public shares on the GigaDrive files mounted locally (in this case, in the local directory named “backup” from the GigaDrive named “gigi”).

Hacking the GigaDrive

I knew there had to be more to the GigaDrive besides a simple web-based interface. Heck, I wanted to be able to use TELNET to log in to the machine and peek around its insides, behind the facade of CGI scripts. But I worried a little about what would happen if the system became corrupted or a problem developed with the GigaDrive.

I e-mailed Linksys tech support with some questions and received a speedy reply, which shows good customer support. According to Linksys technical support, if the GigaDrive “crashes, you would have to RMA it... at this time there are no restore disks for the gigadrive to restore it yet.”

This is quite interesting, because when I looked at the Windows CD-ROM included with the GigaDrive, I found out that the CD is actually a Red Hat Linux 2.0.36 install CD with a custom install program to restore the GigaDrive's file system on a new drive!

It took me about 20 minutes to figure out how to enable TELNET on the GigaDrive. One way to do this is to remotely run a simple, undocumented CGI script installed on the drive. After another 20 minutes, I was able to get the Red Hat system's root password, figure out a special TELNET user password and then TELNET in to the GigaDrive, gain root access and create my own user account “behind” the Samba and web-based interface. From there, it was a simple matter to copy favorite libc5-based clients, such as the **pico** text editor, onto the Red Hat file system. This will also be necessary if you want to install (unsupported) NFS or FTP service on your GigaDrive.

Now, before you throw your hands up in horror (“Oh my God! Anyone can get into our GigaDrive!”), you should know that cracking the root password won't do any good unless you can gain TELNET access (root TELNET access is denied through a TCP wrapper, and TELNET is disabled by default). The base Red Hat 2.0.36 system on the GigaDrive also uses shadow passwords, and you cannot get the special TELNET user password unless you know something about your GigaDrive—I'm not going to say what.

The GigaDrive is quite secure when properly installed and configured. There is no way a casual or even determined cracker can gain access, unless you leave your own password on your desktop or under your doormat.

I really like my “gigi” GigaDrive, and I really like Linksys for its hardware support for the Linux community. I heartily recommend the GigaDrive for any home, home-office or small-business user who needs quick, inexpensive, on-line and secure storage for a small network. You can also find the GigaDrive at a discounted price if you shop around, which makes this Linksys product a winner in a growing field of instant-storage devices.

Bill Ball (bball@staffnet.com) is the author of numerous books about Linux, but still doesn't know what to do with his multiple copies of shrink-wrapped Microsoft operating system software and CD-ROMs. He is a member of the Northern Virginia Linux Users Group.

Archive Index Issue Table of Contents

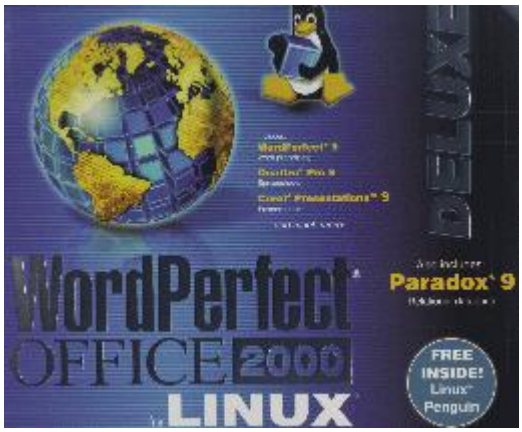
Advanced search

WordPerfect Office 2000 Deluxe

Jon Valesh

Issue #76, August 2000

One box contains everything needed to use Linux in an office environment, and if you don't have Linux yet, there is even a CD with Corel's LinuxOS distribution.



- Manufacturer: Corel Corporation
- E-mail: custserv2@corel.ca
- URL: http://www.corel.com/
- Price: $159 US
- Reviewer: Jon Valesh

Some ideas are so widely held, so reasonable, so attractive, so ... *right*, that they worm their way into your brain and are just about impossible to fight off. You hear someone talking and, without even needing to think about what they have said, you know they are talking truth. Linux is awesome. True. Microsoft is a monopoly. True. The patent office has gone insane. True. Linux will never be taken seriously by the computer industry as a whole until mainstream office software becomes available. True.

But are these truths really true? In the case of Linux, Microsoft and the patent office, of course they are! What about Linux needing mainstream office software to be accepted by mainstream users? Widely believed? Yes, but...truth,

in a warm wash of golden light and trumpeting trumpets? Linux needs applications, that is true. But we need applications that are unique to Linux, applications that will force users through Linux's doors, and not provide them with another excuse for walking past.

There have always been excuses, and there will always be excuses. From “not pretty enough” to “hard to use” to “no commercial support” to, well, I've never heard anyone cite their fear of penguins, but somewhere on this planet, someone is avoiding Linux because the penguin reminds them of Big Bird-inspired nightmares they had when they were four.

Excuses or not, there are some sizable reasons why full-featured office software may be a long time coming from the GNU world. First amongst them is the nature of office-software users. Unlike development tools and Internet servers, word processors and spreadsheets are generally used by non-technical people with non-technical problems. The users don't necessarily have the skills to write their own word processor the way a programmer can write their own compiler. They don't need those skills. The second big reason is the lack of standards. There are no ANSI-standard word processor files, no central body dictates spreadsheet interfaces, no RFC explains existing file formats. In fact, it is just the opposite: the interfaces and file formats have been used as an anti-competitive trump card so many times that even mainstream office applications can have difficulty reading older versions of their own files.

Would-be office software developers can't even go their own way and ignore existing office products, because users need file compatibility and interface familiarity.

So, the options can be reduced to one: commercial software. Here, the requirements shift. You still need file compatibility and an easy interface, but you also need a company that will stand behind their product. Ideally, they will also have a name that draws users. Unlike most GNU software, commercial applications must compete not only for users but also for user money. They must be able to stand feature-to-feature against the best products from the biggest companies. In office software, that means standing against Microsoft. [Although with the recent DoJ ruling, this may change. Someday. Perhaps. —Ed]

Fighting Microsoft in the commercial world means being a company with the money and staff to go one-on-one with the richest and most aggressive corporation in the industry—and not get bought up or buried. Unlike GNU/Linux, a temporary setback isn't just a matter of spending more time coding and waiting for the next opportunity to arrive. It means angry shareholders, lawsuits, layoffs, shattered dreams and all the nasty stuff that happens when people lose their shirts.

You need a company with not only the will but the strength to fight.

Corel has uniquely positioned themselves to fill that need. While most of us sat on the sidelines and grumbled about Microsoft's tactics, Corel decided to play the game, fighting Microsoft head-on using Microsoft's own tricks. Corel and Microsoft have always had similar business strategies. Both grew from humble beginnings, in part, by augmenting their product development resources through purchasing companies and products to combine with their own development efforts—shepherding the best they could build and buy toward commercial success. Corel's office suite, containing mostly bought goods, is no different. Each of the major applications started life as another company's brainchild, but Corel has added several of their own applications and given them all a consistent look and feel.

The key word, when defining either company's strategy, has always been “opportunistic”. However, a few years ago, Corel decided to take the gloves off and face Microsoft on Microsoft's home turf. The size difference is undeniable, the strategy bold. So far, the going has been rough for Corel, but the future looks very bright if Corel's latest moves are right.

And what is their goal? To capture the business desktop. Their strategy? Do it any way they can. Sell an office suite head-to-head with Microsoft Office. Sell a desktop operating system head-to-head with Windows. Make their software multi-platform. Remove barriers to leaving Microsoft, and provide alternatives with a good mix of features and price.

Oh, and sell Linux. Corel saw Linux and realized exactly what each of us did when we encountered the Penguin: here is something truly great and truly threatening to Microsoft. Corel quickly committed themselves to providing Linux versions of their application software, developing their own Linux distribution, even making a try at Linux-based thin-network clients and other Linux-centric products—and they have shown some remarkably good judgment in the process. The Corel LinuxOS distribution is based on Debian GNU/Linux, one of the oldest and most respected distributions around. Their approach to porting their applications was equally practical. Since supporting multiple source code bases or spending the necessary time on a massive inter-OS port would take too long, they spent their time making their applications run under WINE, the GNU free Windows emulator for Linux. The bulk of the Linux-specific software in WordPerfect Office 2000 is actually in the installation program.

And Finally, the Product Review

WordPerfect Office 2000 Deluxe includes everything most business users need to make a computer useful: a word processor, spreadsheet, database, calendar/PIM, presentation graphics software and even a game. One box

contains everything needed to use Linux in an office environment, and if you don't have Linux yet, there is even a CD with Corel's LinuxOS distribution.

At the heart of it all is WordPerfect. Hardly new or unknown in the word processing game, WordPerfect has been around since the beginning, and has steadily improved and expanded since its inception. WordPerfect 9 provides all the features expected in a professional-quality word processor, from page layout to automatic spelling correction to PDF generation; full-featured grammar checking, an “Expert” designed to walk you through the process of creating a document, and more. It may not have every feature imaginable, but it has more than most people will ever use.

Quattro Pro, Paradox and Corel Presentation Graphics round out the office suite. The spreadsheet, Quattro Pro, and database, Paradox, are longtime contenders in the office software world, but are less well-known than WordPerfect. Quattro Pro has all the modern spreadsheet features you could ask for and a consistent and easy-to-use interface. Paradox provides more database functionality than most end users will ever need. When it comes to graphics, few companies have the history and experience of Corel. With that in mind, the features of Corel Presentation Graphics look downright dowdy compared to CorelDraw, but it is designed to serve a different purpose and works well for office meetings and sales presentations.

Railroad Tycoon II from Loki is there to ease you through any frustrating moments you may experience, although some may question the rationale behind including a game in what is otherwise rather straight-laced business software.

Using It

Office software inspires user loyalty. After all, if you spend eight hours a day for years working with a program, you are going to understand it pretty well. Little changes can make a big difference to someone who “knows” how a program should work. For all that, the WordPerfect Office interfaces are standardized enough that, unless you are a power user of another suite, you won't be too bothered by the differences you encounter. If you are already using WordPerfect for Windows, you will feel right at home. WordPerfect for Linux looks exactly like, well, WordPerfect. The other applications in the suite are equally unsurprising in their appearance.

Whether you like the interface or not, some of the features are undeniably cute, and some are better than the competition. The WordPerfect grammar and spell checkers seem to ace Microsoft's offerings; the dictionary is larger and the grammar checker gives a little more detail. If you like thesauri, a drop-down on the property bar will keep you occupied, as it automatically finds alternate

words while you type. Upon further consideration, that's a feature rife with opportunity for abuse; but, abusable or not, WordPerfect has enough features to keep anyone with something to say busy for a long time.

The other applications in the suite are similar. The features may not be the ones you are used to, but you will rarely leave tasks unfinished for lack of features.

More important to most of us than feature selection is file compatibility. The programs in the WordPerfect suite all have import and export filters for a wide range of competing products, including the latest versions of Word. They even work, most of the time. There can be real, though arbitrary, incompatibilities between programs, which will cause problems from time to time. When importing an Excel spreadsheet to Quattro Pro, the import filter took exception to having a hyphen in the spreadsheet name. This is not a problem in Excel and shouldn't be a problem in Quattro Pro, but it is. The import filter offered to replace the hyphen with an underscore character, and other than the name change, the file read perfectly.

File compatibility is not a two-way street, however. WP can read most other file formats, but most programs are unable to read WordPerfect files. If you need to access your documents using Word, you must save them in Word format. Even then, you may not be able to access the file exactly as it was; differences in available fonts alone can cause otherwise perfect file conversions to look like a madman was in charge of your document layout.

All of the programs share some idiosyncrasies caused by the use of WINE. For example, when saving, some of the dialog boxes refer to your home directory as the D: drive, and the floppy (referred to as the A: drive) is always listed as a destination, but saves to the floppy only when you have a disk mounted. That could cause some confusion, especially for new Linux users.

The first time WordPerfect starts up for a new user, it must do some final setup, like create a directory for user-specific files, generate font information, ask the user if they intend to comply with the license and other little jobs of that nature. Unfortunately, WP does not tell you anything is taking place while it does that. You click, and nothing happens. Or nothing happens right away. Somewhere between 10 and 40 seconds after clicking on the WordPerfect icon, a copyright screen appears to let you know that your computer isn't being eaten by a runaway word processor.

The performance was acceptable on modern systems. I tested on an older P90 that was definitely struggling and would be marginal for word processing, unacceptable for more. Linux is less resource-intensive than most operating

systems, and WordPerfect and Quattro Pro running at the same time with moderately sized documents took between 29MB and 34MB of RAM (not counting buffers or cache), depending on the test system. A full installation requires over 450MB of drive space, not counting operating system, games, clip art or additional fonts. If you want to install everything in the box, you will need at least 1GB available.

Installation

Corel is aiming at several types of users with WordPerfect Office: people who already use Linux but want the power of a professional office suite; people who are putting together a new computer and don't want to tithe to Microsoft; and existing WordPerfect users who are ready to make the switch to Linux. To support the widest range of existing systems, they include an installation program that recognizes your distribution and installs itself accordingly. Corel also provides the suite in both Red Hat RPM and Debian DEB files, so if you have problems with the installation program, you can perform the installation manually. People buying WordPerfect Office as a complete desktop solution get a general-release version of LinuxOS, identical to what you can download from Corel's web site.

I installed WordPerfect Office on three computers running three different Linux distributions, just to see what how it would handle the differences.

The Old System

The first was a typical older system, a P90 with 32MB of RAM, a 4X CD-ROM and a 2GB hard drive. This is significantly below the recommended minimum system requirements, and you'll quickly see why when you try to run the office applications. It is the sort of system that Linux is often called upon to resurrect (or at least sustain) for a few more years after Windows bloat has rendered it “obsolete”. Don't expect WordPerfect to help, though—Corel recommends a minimum of a 166—200MHz CPU, and they mean it.

I started by popping the LinuxOS CD into the drive and rebooting, hoping the BIOS was modern enough to boot from the CD-ROM. It was, and after a few moments of earnest disk access, I was rewarded with a subdued low-res color graphic of Stonehenge, a sundial and other oddities, with messages politely telling me to be patient while Linux loaded.

Once the system was sure I remembered why nobody sells 4X CD drives any more, it got on with its job and started asking me questions. Not many questions, however. Agree to the license? Want the full installation? Repartition the hard drive? Yes. Yes. Yes. My name is Jon, and so on. Five questions later, the installation began in earnest, and I, remembering how slow a 90MHz CPU

feels in this age of 700MHz laptops, left it to its devices and went off to have dinner.

When I got back, the system was patiently waiting for me to eject any floppy disks that might have snuck into the drive so that it could reboot. A quick click of the “OK” button, and the system automatically ejected the installation CD and started toward its first LinuxOS boot.

Corel has done their best to hide the “Linux” in “LinuxOS”. A graphical boot loader hides as much detail as it can, telling you in the simplest possible terms that, yes, the system is booting, and you should just hang on while it does its job. The system probes your hardware and auto-configures itself based on what it recognizes, which may or may not be what is actually there. Once the auto-configuration process is complete, you are presented with a simple graphical login prompt. You can log in as root or the one regular user account created during the installation process, and, after setting your password, you are ready to go.

This is about as hands-off as you can get in a Linux installation. The system defaulted to 1024 by 768, 16-bit graphics, no network and no sound, which is surprisingly good considering how little I had to do to get it working. I would have preferred to answer some questions about my network card and preset the passwords, but their decision to keep the installation simple and hands-off is probably justified considering the target audience.

KDE is the default user interface, and Corel has kept that as simple as possible, too. No questions are asked about how you want the system to look. There is no choice of having a CD-ROM icon on your desktop. Instead, you are presented with a rather Windows-like desktop with a few icons on the left edge and a bar along the bottom having a few icons on the left and a clock on the right. The configuration isn't totally Windows-like, with four virtual desktop icons and no “Start” button, but any Windows user will be able to figure out that the icon where the “Start” button would be does the Start button's job.

For everything Corel intends LinuxOS to be, it is perhaps a little too simple. Corel seems to have done a good job of removing redundant menu options, which also means that unless you know where to look, you are going to have a harder time finding some features.

After a little bit of random searching, I had a directory listing of the WordPerfect CD on screen. A quick double-click of the setup icon launched the WordPerfect Installer.

The New System

I also installed WordPerfect on a new 500MHz Dell Inspiron 7500 Laptop running Mandrake Linux. Installation on this system was actually easier than using LinuxOS, or at least it required less thought. I just popped the CD in, and a few double-clicks later, the installation was on its way.

The Easiest System of All

The third computer to receive WordPerfect was an AMD K6-2 system running Caldera eDesktop 2.4. Unlike the Corel or Mandrake systems, Caldera immediately saw that something interesting was on the WordPerfect CD and started the installation process on its own.

And What Was the Installation Like?

In all three cases, the start of the installation process foretold the greatest problem—nothing happened. Okay, that's not totally fair, the hard disk light started flashing; but in all three cases, there was a significant delay between starting the installation program and any indication it was running. The lack of feedback is frustrating when the installation begins, but an equal lack of feedback when something goes wrong can cause more than frustration. If an error occurs, the installation program closes without explanation.

Installation must be done as root. It starts with the ever-present license agreement query and finishes by asking if you want a full or partial install. Everything else is automatic, although the installation program does give you a chance to panic and abort if you don't like the target directories. The installation program uses the system package tool for the actual installation, which is why the software installs in the normal system directories and not the “local” directories many commercial applications use. After the installation has started, a progress bar and slideshow presentation let you know something is happening.

Using the system package tool is a rather neat solution to providing cross-distribution Linux software, but Corel should work on fault recovery. In two out of the three installations, something happened (me) to cause the Corel installer to die without so much as a grunt of pain. Prematurely closing the installation program caused the worst problem, leaving the installer permanently unable to run on the system. If you get in too much trouble using the installation program, you can always use your system's native package tool to install manually. After I realized the full impact of my fateful installation-stopping mouse click, I was able to install WordPerfect using Debian's **apt-get**, which Corel uses for the LinuxOS package tool. Manual installation instructions are provided in an HTML README file on the CD.

WordPerfect automatically installs itself in KDE's Applications submenu for all users. After installing, you must either log out or restart KDE in order to see the updated menu. This isn't a problem, because if you do try to run the applications as root, you will be politely warned that it really isn't safe. You can, but it is not recommended.

Installation took from fifteen minutes to two hours depending on the computer, and, uneventful process or not, the software did install on all three systems.

Problems? What Problems?

I didn't have very many problems with WordPerfect Office or LinuxOS. The worst one occurred on the Mandrake-based laptop, which received its copy of Mandrake from an older CD. It would open, import and edit files flawlessly, but invariably crashed when called upon to save those files. Not simply crash, but wipe out the source file and any backups that may have been ferreted away. After a great deal of trial and error, I realized that virtually none of the WordPerfect software would run reliably on the laptop, even though most of it ran perfectly. Delving deeper, the problems occurred only for users; when the WordPerfect applications were run by root, they worked fine. This pointed to a problem with the operating system, and upgrading the Mandrake installation seems to have buried the problem.

When it did crash, WordPerfect was less than perfect about having a backup saved. It tries, but it doesn't always succeed.

I noticed only one problem with LinuxOS: the system was unable to establish PPP dialup connections as a user. As root, KPPP dials without difficulty, but normal users get an error message claiming PPP is not installed in the kernel. This is at odds with Corel's positioning of LinuxOS as a desktop operating system for non-technical users.

In the End

WordPerfect Office 2000 Deluxe offers a real solution for people who want the power of mainstream office software without contributing to the Bill Gates Retirement Fund. It is an even better solution for people who use Linux and need access to full-featured office software without rebooting. It is at its best in environments where compatibility with other software packages isn't the most important issue. The file import/export capabilities provide a link to other popular office software, but if compatibility is a matter of daily importance, you will find weaknesses. Overall, it is a great set of programs with the promise of becoming even better.

The Good/The Bad

Born at the beginning of the microcomputer age, **Jon Valesh** (jon@valesh.com) has pushed and been pushed by computers his entire life. Having run the gamut from games programmer to ISP system/network administrator, he now occupies himself by providing technical assistance to ISPs and small businesses whenever his day job doesn't get in the way.

Archive Index Issue Table of Contents

Advanced search

Caldera OpenLinux eDesktop 2.4

Jon Valesh

Issue #76, August 2000

The latest product in Caldera's business-friendly lineup is eDesktop 2.4, designed to bring the power and cost-effectiveness of Linux to the corporate and home desktop.

- Manufacturer: Caldera Systems, Inc.
- E-mail: linux@calderasystems.com
- URL: http://www.calderasystems.com/
- Price: $39.95 US
- Reviewer: Jon Valesh

Enthusiasts have a real advantage in the Linux world. For them, the value of Linux is self-evident: it is Linux—Linux is cool. Spending long hours and sleepless nights keeping up with the latest drivers, the greatest new applications and all the hot news is entertainment for a real enthusiast. Like a video game or a puzzle, it is a chance to think, learn and show off. It can even be financially rewarding for those with talent.

Businesses have a different perspective. Hours spent working with Linux (or any OS) aren't fun or rewarding—they are a wasted resource, also known as lost money. Businesses, especially businesses that do more than develop tech gear for tech people, have real-world problems, and the only reason they want computers is to help solve those problems. They don't even care how the problems are solved, as long as the solution works, and when they choose a computer system or OS they aren't interested in new drivers, hot applications or cool news. They are interested in enhancing their bottom line.

Caldera Systems, one of the first companies formed specifically to “productize” Linux, understands the distinction. They know that, unlike the techno-freaks and Linux fanatics that give Linux its technical drive, business users don't care what goes on under the hood and behind the scenes in an OS, as long as it works. They understand that the people who make corporate technology decisions must justify—in cold, practical terms—the technology they adopt. They must be able to point to clear benefits that will positively affect the corporate bottom line, not just say, “This is neat, let's use it!” They must also be able to allay the fears of upper management, fears ranging from reliability and product support to the simple fear of change. They also know that if a technology change turns into a fiasco, they will have a long time to think about what went wrong as they search for another job.

Caldera understands the business view of technology and has been working for years to turn Linux into a product that businesses, businesspeople or anyone new to Linux can be comfortable using. The latest product in Caldera's business-friendly lineup is eDesktop 2.4, designed to bring the power and cost-effectiveness of Linux to the corporate and home desktop. It is easy to install, 99% pre-configured, uses the elegant and easy-to-use K Desktop Environment and comes with a complete set of administration tools, so anyone with even basic computer skills can configure and maintain their system. Caldera has also bundled, developed or encouraged the development of Linux versions of many of the tools, protocols and features that businesses want and IT managers need. Interoperability with NovelT NetWare servers, web-based remote administration and a suite of Internet applications, including Netscape Communicator 4.7 which is pre-loaded and provides most of the plug-ins people need (news, mail and chat clients), Macromedia Flash Player, Real Networks RealPlayer, Adobe Acrobat and a host of other applications and servers.

Caldera eDesktop 2.4 has a little something for everyone, including techno-freaks and Linux fanatics, but nowhere does it shine brighter than as a Linux for non-Linux users—at home or in the office.

Documentation

For anyone not totally committed to a self-image of superhuman technical ability, a well-written manual is important. When delving into an unfamiliar technology, the manual must be especially good. A poorly written manual can mean the difference between a useful product and a frustrating waste of money. Caldera obviously knows most of their target audience won't know a lot about Linux when they begin using eDesktop, and the manual is both well-written and complete. When it comes to pertinent, easy-to-follow documentation, Caldera has done their job by describing the installation, configuration and administration of eDesktop in excellent detail. They also include many references and links to other information sources for users who wish to learn more.

While the manual is very good, “User's Guide” may be a bit of a misnomer. It is more an “Installation and Administration Guide” for people new to Linux than a user's guide. The manual tackles, fearlessly and with admirable clarity, the task of getting first-time Linux users comfortable making configuration changes, managing users, working with the command-line interface and doing everything else that longtime Linux users take for granted. It spends very little time on the minutiae of actually using the eDesktop, which is perfect for home or small business users, who usually manage their own systems. Corporate IS departments will probably not want to hand out the manual to office staff who need to get up to speed on *using* Linux/KDE, though—of the roughly 500 pages of manual, about 75 are specific to using, rather than administering, OpenLinux eDesktop.

Installation

OpenLinux eDesktop 2.4 can be installed a few different ways, but the easiest way is to insert the CD and reboot (a boot floppy is included for computers that cannot boot from CD). That allows you to install eDesktop in whatever free space is available on your hard disk or to overwrite any operating system you may have been using, and take up the whole disk. If you aren't ready to commit yourself to Linux completely, you will really appreciate the inclusion of PartitionMagic. PartitionMagic allows you to shrink your Windows partition without reformatting your hard disk or losing data. Just by inserting the eDesktop installation CD while running Windows, and walking through a few simple steps, you can make room for Linux without deleting your old operating system or data files. [The often-recommended advice of many computer professionals: make a backup of all important files or your entire hard drive, before doing any significant installation. —Editor]

While PartitionMagic is a great add-on for users wishing to try out Linux, the version shipped with eDesktop has one serious flaw. Because it is specifically

for use with eDesktop, there are only a few partition sizes it will create. Specifically, there is “too small for eDesktop” and “too small for Windows”. You can create either a really small Linux partition (350MB), a slightly small Linux partition (700MB) or take everything but 100MB of the hard disk for Linux. The first two choices aren't good for much more than trying out eDesktop, and anyone familiar with Windows will recognize the problem with the third option —Windows with only 100 megabytes of free space is even less usable than normal.

Whatever first step you take, the second step for first-time installers is Caldera's slick graphical installation tool, which they call the Lizard. Lizard guides you through all of the important decisions about your installation: setting system information, partitioning the hard disk (if you didn't use PartitionMagic), selecting your monitor and other hardware, choosing the software to install, setting up user accounts and, perhaps most uniquely, configuring dialup Internet access.

Caldera has made some of the hardest parts of Linux installation easy by including a couple of impressive databases. Getting the most out of your video card and monitor, a task that usually requires a fair amount of trial-and-error, is easy with eDesktop because of the huge monitor selection database, which seems to contain just about every monitor ever produced. Also handy is the database of ISPs that covers most of the world and makes setting up dialup Internet access easy for customers of major ISPs.

Lizard is smart enough to start copying files as soon as it knows where to put them, so on a fast computer the actual installation process is very quick, not much longer than answering the configuration questions. If you finish configuring the system before all the files are copied, Caldera provides a bit of entertainment in the form of a game to pass the time.

Lizard is very easy to use but acts as a gatekeeper of sorts for eDesktop. Its first task is to perform a hardware compatibility test, and if it doesn't think eDesktop will run on your computer, it refuses to proceed with the installation. This would be fine, except that it is a bit too strict in its testing. Lizard sometimes refuses to install eDesktop on systems with newer video cards, even though the display driver, or X Server, that eDesktop uses is compatible with those cards. For example, Lizard refused to install eDesktop on a laptop with an ATI Rage LT video chipset that is supported by eDesktop's Xfree86 version 3.3.6 X server. On systems Lizard refuses to interact with, you can install eDesktop using a backup, text-mode installation program called LISA.

There are several automated installation methods for system administrators and IT departments that need to get a lot of systems up and running with

identical configurations. If you have a LAN, you can set up an install server and quickly install eDesktop on any computer on your LAN. Just insert a floppy, reboot and the installation program takes care of the rest. You can also create a boot floppy with Lizard, configured to install the default configuration without prompting, or create a custom installation, specifying the exact programs you wish to install on each system.

Use

Perhaps the most significant aspect of using eDesktop 2.4 is there really isn't much to say about it. For the user interface, Caldera uses KDE, the K Desktop Environment, which has become the *de facto* standard for Linux desktops. With good reason, too: KDE is easy to use, looks great, is very configurable and uses an interface metaphor most Windows users are familiar with, the bottom-of-the-screen control bar, complete with a “start button” labelled “K”.

Most of the pre-installed and bundled applications work without much tinkering. The bundled commercial software must be installed after eDesktop is up and running, but the process is easy and fast. eDesktop is exactly what a business desktop OS should be: a gateway to the applications where you do your real work. It gets out of your way and lets you do your work, without forcing you to learn and re-learn its own features. That is not to say that it lacks features. KDE starts out with power-user features like multiple virtual desktops and is one of the most configurable user environments around, so you can tailor its look and behavior to your needs.

Administration

The three most common complaints new users have about Linux are it's too hard to install, has too few applications, and in the long term, probably the most important is that it's too difficult to “administer”. “Administer” in quotes, because that's probably the last term most new Linux users would think of; they just want to keep their computer running, their software up to date and their data safe. Whatever you call it, making a Linux system do what you want is often viewed as a dark art involving mysterious knowledge, and probably, animal sacrifice.

This is not true with eDesktop. Caldera has provided an overflowing banquet of administration tools in the form of two system administration interfaces which, if they don't actually guide you through every necessary administration task, at least provide a clear view of what you can and cannot change. COAS (Caldera Open Administration System) provides a graphical administration interface for use from the desktop, allowing you to configure most (but not all) system features without resorting to a shell or a text editor. For environments requiring more centralized management, there is Webmin, a new tool that

allows you to perform most routine system administration tasks remotely from any computer with a web browser. COAS and Webmin don't change the need to administer, but they do make it easy for people who don't know—and don't want to know—about “traditional” Linux configuration files to get started administering their systems, without learning a bunch of technical mumbo jumbo.

There are still quite a few tasks that will require a text editor and shell knowledge. From compiling new kernels to reconfiguring LILO, many advanced (but necessary) administration tasks haven't made it into the graphical or web administration tools yet. Fortunately, the manual describes these tasks in detail, providing step-by-step instructions for the most common administration tasks that cannot be performed with the administration tools.

The system's native package tool—the program you use to install and remove most software—is the Red Hat Package Manager (RPM). RPM is the most widely used package tool in the Linux community, and it is the most popular method for distributing third-party software. To easily manage installed packages, eDesktop uses **Kpackage**, which allows you to browse packages installed on your system and easily add or remove software.

Linux users familiar with package tools like Debian's **dselect** will suffer from Kpackage's inability to automatically download packages from Internet servers. To upgrade an installed package with OpenLinux, you must manually transfer the package to your system, and then install it. **dselect** can automatically retrieve a list of available packages from an Internet server, compare the package versions from that server to what is installed on your computer and (with your permission) download and install replacements for any older packages you might have. That sort of feature would go a long way toward enhancing Caldera's place in corporate IS department hearts, and it would help new Linux users by automating one of the most boring system administration tasks—upgrading older software and making sure that potentially security-compromised software is replaced promptly.

Neat Stuff

You will find a long list of neat add-on packages in the eDesktop 2.4 package. Depending on how you will use it, you may find some of the programs great—or totally useless. Corporate users in mixed computing environments may find the Citrix ICA and NetWare clients very useful. Web and Internet developers will like the Cameleo Lite graphics program, Omnis Studio Rapid Internet Application Development Environment, JDK 1.2.2, Apache and other Internet development tools. Small and home office users will find the StarOffice suite and the Moneydance personal finance application handy. There is something for everyone in the eDesktop box, but it is all rather straight-laced utilitarian

software and much of it is very specialized. If you don't have a Citrix server, you don't have a use for the Citrix ICA client.

If you are looking for entertainment from your OS, you are going to have some shopping or downloading to do.

Problems

The single biggest problem with OpenLinux is Caldera's “simplicity by hiding details” design philosophy. Hiding irrelevant or distracting details is a good thing. Hiding the truly useless babble that Linux often displays can go beyond a good thing. However, unless details and babble are replaced with concise descriptions of what the system is doing, you can be left wondering and frustrated when problems occur. If you take away the information an “expert” would use to diagnose a problem, you need to replace it with software smart enough to figure out the problem on its own.

An example is the all-or-nothing installation: some systems that should be supported aren't, because Caldera, in their wish to keep the installation process simple for new users, went too far in checking the system for compatibility and removing the details of those checks from user view. It is a tradeoff, and users of mainstream hardware and software will benefit; however, I would like to see the Lizard and all of Caldera's software a little more willing to divulge troubleshooting information. Even a note about what went wrong, a query asking how to proceed and a warning that if you proceed, things might not work as expected, would let more people use the eDesktop. If Caldera invested the same effort, energy and skill they have contributed to COAS and Webmin in developing software that monitored the system status and reported—in plain language—everything that is going right and wrong, they would have the perfect counter to the “Linux is only for geeks” argument. Not only a counter argument, they would truly be simplifying, instead of hiding, the most complex part of using Linux.

Summary

You won't find a simpler or easier-to-use full-featured Linux distribution anywhere, though you may find another Linux distribution that is better suited for some uses. Caldera has done a lot to make Linux easy for new users to adopt by making eDesktop the easiest and least risky, commercial Linux distribution for people who want to learn about Linux.

The Good/The Bad

Born at the beginning of the Microcomputer age, **Jon Valesh** (jon@valesh.com) has pushed and been pushed by computers his entire life. Having run the

gamut from game programmer to ISP system/network administrator, he now occupies himself by providing technical assistance to ISPs and small businesses whenever his day job doesn't get in the way.

Archive Index Issue Table of Contents

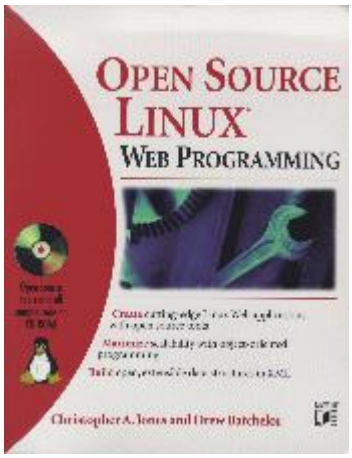
Advanced search

Open Source Linux: Web Programming

Daniel Lazenby

Issue #76, August 2000

In this book, the reader is guided through an extensible and scalable web application development methodology.



- Author: Christopher A. Jones and Drew Batchelor
- Publisher: M&T Books, an imprint of IDG Books Worldwide, Inc.
- URL: http://www.idgbooks.com/
- Price: $39.99 US
- ISBN: 0-7645-4619-8
- Reviewer: Daniel Lazenby

The ability to create easily maintained and easily updated web applications is crucial in today's fast-changing, browser-centric world. Rapidly changing business demands drive the need for rapid web site content updates/changes. Competing artistic factors supply a constant challenge to keep the web site's visual design interesting. Seldom does one individual possess all the talents required to meet each of these challenges. *Open Source Linux: Web Programming* addresses these types of challenges.

In this book, the reader is guided through an extensible and scalable web application development methodology. The tools used to carry out the methodology include Perl, CGI, HTML and XML. The recommended methodology is based on object-oriented and decoupled functionality concepts. By the time you reach the book's end, you will have developed and examined a series of web applications that apply each design concept or programming convention.

The authors assume the reader possesses at least an intermediate level of experience with computing and programming technologies. With this framework, the book states the concept to be applied or the task to be performed. The rationale for why each concept or task is important is also presented. What you will not find is a list of the steps required to perform the tasks. Extending what you learn to your own applications is where the intermediate to advanced experience comes in. What's presented will generally get you up and running. Other resources are required if you want more specific details.

The book is divided into two sections, “The Modern Internet Application” and “Going Beyond Today's Internet Model”. The first section (and majority of the book) is devoted to programming an Internet application. Each chapter opens with a few paragraphs on what you are going to build and why the way it will be built is important. Systems or functionality programmed in one chapter is often the foundation for systems or additional functionality presented in later chapters. For those who dislike typing, all program listings are available on the CD-ROM.

The first section opens with a quick review of distributed systems and an introduction to the Apache HTTP server. In the chapter titled “Using Perl to Implement CGI on Linux”, the authors present just enough Perl for the reader to understand the forthcoming application programs. A simple Perl-based CGI module that analyzes Apache logs is programmed in this section.

The “Separating Content and Logic with Objects” chapter further explores a modular and dynamic web site design. Decomposing application content, program and business logic and presentation into objects are discussed. Techniques presented facilitate coding an easily maintained site-wide look and feel. Here, “easily maintained” means a non-programmer can update content and change graphics. These designs and techniques result in coding a simple on-line catalog application.

The next three chapters focus on using XML. “Using HTML to Create Well-Structured Objects” walks one through basic XML syntax and structure. Writing XML-driven CGI scripts that translate XML to HTML is the focus of the “Handling

XML with a CGI Application Server" chapter. The discussion on XML continues in "Creating a Client-Based Web Content Administrator". This chapter concludes with the coding of a Perl, CGI, HTML and XML web content administrator application. This application enables a non-technical person to update web page content. All the concepts and coding techniques presented in these chapters may be put to use today.

In the book's second section, the authors pause to look a short distance into the future. They then close the book out with two very critical subjects. This section's opening chapter discusses limitations and issues related to the stateless transaction nature of HTTP and CGI. The authors offer a rationale for moving to a state-based transaction and an open-source XML application server.

Chapter 9, "Developing Java Applets on Linux", presents Java basics through the construction of simple Java applets. Chapter 10, "Creating a Java/XML Web Application", pulls together previously presented design principles and techniques to code a network chat and whiteboard application.

The last two chapters cover error handling and web deployment applications. Poor error handling can easily destroy the user's experience. Poor site deployment can ruin the user's experience and negatively impact the application's post-implementation performance and maintainability. An error-handling system is constructed in the "Bulletproofing Your Site with Error Handling" chapter. As in the previous projects, the authors begin by applying object-oriented concepts to the error-handling system. In this example, they decouple error messages from the error codes. Both objects are decoupled from the error handler and the actions to be taken upon an error situation. Nothing stresses an application, or its implementation, more than a real user. Next to the user, nothing can cripple a web site faster than coding techniques incompatible with the implementation strategy.

The chapter entitled "Deploying Your Linux Web Application" reviews how implementation decisions can reach back and negate application development and coding decisions. This chapter briefly discusses the need to consider various factors affecting throughput, scalability, system resource consumption and the need to avoid vendor-specific language extensions. General guidelines for identifying bottleneck and bandwidth problems are suggested.

By the time you reach the end of this book, you will have explored conceptual constructs, models and tools you can use to develop web sites. These sites should be easier to deploy, update and maintain. More importantly, web site content changes will not always require programming expertise to implement.

Daniel Lazenby (d.lazenby@worldnet.att.net) first encountered UNIX in 1983 and discovered Linux in 1994.

Archive Index Issue Table of Contents

Advanced search

Building Database Applications on the Web Using PHP3

Gaelyne R. Gasson

Issue #76, August 2000

A CD-ROM with PHP3, PostgreSQL, the latest GPL version of MySQL, various RFC text files and W3C's HTML 3.2, 4.0 and CSS2 specifications comes with the book.

- Authors: Craig Hilton and Jeff Willis
- Publisher: Addison-Wesley
- URL: http://www.php123.com/
- Price: $39.95 US
- ISBN: 0-201-65771-6
- Reviewer: Gaelyne R. Gasson

This book begins with an upbeat preface about PHP3's ease of use for web applications. We're told it “emphasizes applications using PostgreSQL and MySQL, two powerful database management systems”. The preface suggests taking a 30-minute “test drive” on the book's accompanying web site; unfortunately, the site wasn't available as of this writing. A note indicated they had a hard drive crash and were rebuilding the system. [The site has since been restored, and everything works except the login. —Editor]

The first chapter is an overview of the book. Chapters 2 through 11 are referred to as background information relating to web-based technologies. Sample applications comprise Chapter 12, followed by a few pages of technical notes in Chapter 13. The reader is only one-third of the way through the book at this point and may be startled to find that a sixteen-page index for the PHP3 Language and a Function Reference make up the bulk of the remaining pages.

Two appendices are included. The first deals with CGI variables; the second concerns timing comparisons between MySQL and two different configurations of PostgreSQL. The reading list doesn't specifically deal with PHP, PostgreSQL or MySQL. The references listed are excellent material for general programming and web development, but the list is broad rather than specific to

the topic of the book itself. There are two additional indexes in the back: one for alphabetically listed PHP3 functions and another for the book itself.

A CD-ROM with PHP3, PostgreSQL, the latest GPL version of MySQL, various RFC text files and W3C's HTML 3.2, 4.0 and CSS2 specifications comes with the book. According to a list in Chapter 1, the CD should also contain “all code built in this book”, but that wasn't the case. To be fair, the CD does include the full source for every version of PHP from v2.0 through v3.0.12 in tar.gz format and several Linux Red Hat RPMs for both PHP and PostgreSQL. It also has the PHP3 manual in several formats.

Generally, you would expect to read a discussion on the choice of operating system before reading about how to install software. However, UNIX (and more specifically, Linux) is covered briefly in chapter 7, “Why UNIX (Why not Windows?)”, while Chapter 1 describes how to install the supplied software from the CD.

Chapter 2 discusses PHP, and the next chapter offers a look at databases, SQL and PHP. This is followed by a brief chapter about UNIX (and Linux) and then a discussion of Internet and browser basics. Logically, one would have expected the flow to have been UNIX first (if at all), followed by Internet and browser basics, HTML, PHP and then databases.

The PostgreSQL chapter includes a chart comparing its features with other databases such as Oracle 8, Sybase 11 and MiniSQL. Strangely, the chart doesn't make any references to MySQL. The chapter also contains examples showing the syntax for connecting to a PostgreSQL database using PHP3 and how inserts and selections are made.

The chapter on MySQL seems to be missing some information. Instead of following the same pattern as the previous chapter, readers are given three short paragraphs about the history of MySQL and seven paragraphs discussing speed differences compared to PostgreSQL. If this were the beginning of the MySQL chapter and it carried on to cover connecting to MySQL databases and other MySQL information in a format similar to that of the PostgreSQL chapter, all would be well, but it's only a brief description of the software speed. Not even a look into the format of commands is offered. This wouldn't be an issue, except the preface of the book says it “emphasizes applications using PostgreSQL and MySQL.” Both database types should have been given the same type of coverage. As it is, MySQL could well have been “swept into” the short (three-page) Chapter 6, “Other SQL Databases”.

The PHP3 Language chapter is fairly detailed and uses several examples. Many of these contain humour to help teach functions and how they're used. The

example used for classes was the most memorable for its humour. I found this chapter most informative, but wish more time had been spent on how classes are used in real-world applications.

Chapter 11 comes back to a fuller discussion of databases and SQL, as well as database design. There seems to be only brief coverage of column (field) types such as string, numeric, date, etc. used in databases, although each type is mentioned within the scope of examples given. A beginner's book on building database applications should have more information on this topic in order to explain the pros and cons for some of the more common choices available. I expected to see the examples in the format of PostgreSQL first, followed by MySQL to show similarities and important differences between the two types of databases, but this wasn't the case.

The applications include Hello World, File-Based Counter, Data-Based Counter, Real-Time Graphics, Jeff's Super Form (a web-to-e-mail form), Specific Form Processor (a web-to-e-mail form that stores information in a database), Querying Application and Login Authentication (in which users allowed to access a page are stored in a database).

The application section had numerous typos that hopefully will be removed in the next edition of the book. Moderately experienced readers should be able to catch most errors, but inexperienced users might not. For example, the text discussing the second application uses a different variable than was used in the "Complete Code" listing—**$counter** mysteriously changes to **$tcounter** two pages after the complete listing.

The Real Time Graphics application shows how to build a runtime graphic "button" using text based on the user's input. However, it's not fully explained that in order for this code to work, there must be an existing graphic ("image/logo1.gif"). Perhaps the authors assumed the code (and graphic) would be included on the CD. Once a suitable image file was mustered, the example code worked as described. Beginners may find this frustrating and give up before going further.

Again, all of the database applications are listed using PostgreSQL exclusively. This could be forgiven if the front of the book had listed only the one type of database. Trying to use the examples as given require numerous modifications when used with MySQL.

After each application, in a segment labeled "Troubleshooting", readers are told "What can go wrong? About a million things" and "try your application directly on the Readers Only PHP3 Web Site". The same paragraph is repeated word-

for-word between each application. A short list of the most common types of errors would have been more appropriate.

The bulk of the book is the “Language and Function Reference for PHP3”, and as stated in the preface, this section isn't a repeat of the PHP3 manual. Items are listed by functional category, and each item includes a short example on how to use it. Where required, further information on the item is given in a clearly boxed area. It provides a convenient off-line reference for PHP3, and seems to have had a great deal of time and effort put into it.

Building Database Applications on the Web Using PHP3 doesn't live up to its title. It *is* a reasonable PHP3 reference, and it's nice to have this in a printed format. The writing style used is upbeat and pleasant, and the segments that cover detailed information are clear and understandable. The problem lies in the organization of the book, numerous typos and missing details.

Gaelyne Gasson (gaelyne@videocam.net.au) is a web administrator in South Australia and a fan of Commodore computers.

Archive Index Issue Table of Contents

Advanced search

A Heterogeneous Linux/Windows 95 Home Network

Chirakkal Easwaran

Issue #76, August 2000

Share resources such as printers, CD-ROM drives and data connections to the Internet.

With more than one computer common in many households, the need to network all machines and provide Internet access is important. Linux offers a stable network environment that can enable different operating systems (UNIX, Windows and Macintosh) to work together, and is quite easy to set up and administer. Such networks facilitate convenient file and print sharing and centralized backups. I will describe my home network configuration (see Figure 1) which is a mixture of Linux and Windows systems.

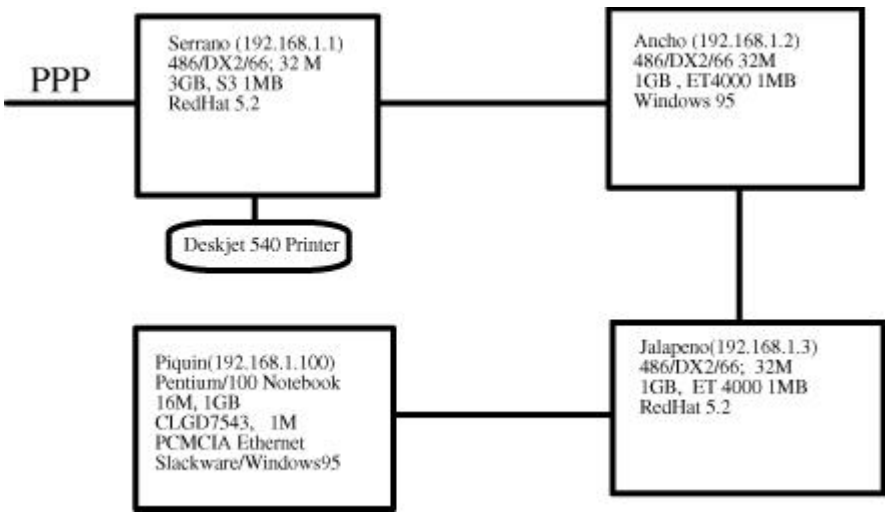


Figure 1. Network Configuration

I will assume you have basic knowledge of Linux installation and administration, and that PPP is set up on your Linux box. I will explain in detail how to network the Windows 95 machine named “ancho” with the Linux server named “serrano”. Adding other machines to the network is essentially a repetition of this process. I will also briefly discuss setting up the Linux client and NFS.

The Linux machine serrano is the file and print server for the Windows 95 and Linux clients, and connects to the outside world through a PPP dial-up connection. It also acts as a firewall; all other machines can reach the outside world through it. The CD-ROM drive on serrano is also available to the other machines on the network.

Hardware

My hardware consists of 486 DX2/66 machines with 32MB RAM, and a Pentium 100 notebook which is used both on and off the network. I use NE2000 network cards (Realtek PnP, about $15) and coaxial 10Base-2 cables. If you have PCI boards (and I certainly hope you do), setup is easier, but the networking particulars still apply.

The Linux kernel probes address 0x300 for NE2000 cards, so the jumpers on the card are set for this I/O address on the Linux machine. On the Windows 95 machines, the network card jumpers are set for Plug-and-Play mode. When you network the machines with coaxial cables, make sure you use 50-ohm terminators (about $3 at Radio Shack) at the ends.

Software Installation and Configuration

I installed Red Hat Linux 5.2 off a CD. Installing everything took about 500MB of disk space. I have used Linux for a while now, so this part was painless. I was able to get X working nicely, PPP configured and a printer set up.

If you have a machine with a CD-ROM drive, Windows 95 installation should also be easy. I had to install Windows 95 from a parallel port CD-ROM drive. That was tricky, and took several tries.

The first goal in the networking effort is to establish communication between the Linux machine serrano and the Win95 machine ancho.

All the network configurations on serrano can be done from the Red Hat control panel (Network Configurator). Read the Net-3.HOWTO for more information on Linux networking. Basically, you need to have the eth0 interface with IP 192.168.1.1 (or something like that) attached to it. To do this manually, execute the following command as root:

```
/sbin/ifconfig eth0 192.168.1.1 netmask\
255.255.255.0 up
```

The IP addresses chosen correspond to class C addresses. Typically, these are used for intranet networks. I had to explicitly put in a route:

```
route add -net 192.168.1.0 eth0
```

This routing command says all packets to the 192.168.1.0 network should be sent to the eth0 interface. This line is appended to the /etc/rc.local file, so that the route is set up at boot time.

On serrano, set up the host table /etc/hosts in a manner similar to this:

```
192.168.1.1     serrano    loghost
192.168.1.3     jalapeno   #Linux machine
192.168.1.100   piquin     #Linux/Windows
192.168.1.2     ancho      #Windows machine
127.0.0.1       localhost
```

To confirm that the eth0 interface is working, type **/sbin/ifconfig eth0**. The output from this command will look like Listing 1. If you then type **more /proc/net/dev**, the output will look like Listing 2. If PPP is up, you will see an additional line for the ppp0 interface. This should take care of serrano.

Listing 1

Listing 2

Now we proceed to the network configuration of ancho, the Windows 95 machine. You should have Windows 95 installed, the network card detected and drivers installed. At the time of network card installation, you will be asked for a machine name and a workgroup name. Give the machine name “ANCHO”, workgroup “WORKGROUP” and some optional comment like “My Compaq 486 machine”.

Go to the Control Panel and select “Networking”. Under “The following network components are installed”, you should see your network card (mine says “Realtek RTL8019 PnP LAN Adaptor or compatible”). You may see NetBEUI and IPX/SPX and TCP/IP in the same window. If you do not see TCP/IP, choose “Add”, then “Protocol”, click on “Add”, choose “Microsoft”, then choose “TCP/IP” and click “OK”. This should add TCP/IP to the installed components window. Once TCP/IP is added, if you highlight it, you will be able to choose “Properties”. When you click “Properties”, you will see a dialog containing several tabs. Pick these tabs, and assign the following:

```
IP Address

  IP Address: 192.168.1.2
  Subnet Mask: 255.255.255.0
Gateway (
  192.168.1.1
DNS Configuration
  Enable DNS
  Host name: ancho
  DNS Server search order: 192.168.1.1
WINS Configuration
  Leave it Disabled.
```

You may leave the DNS disabled, or add your ISP's DNS server IP address. The important items here are your gateway and IP address. After you specify these and click “OK”, the machine will reboot.

After reboot, open a DOS window and type **ping 192.168.1.1**. After a short pause, you should get a response like “Reply from 192.168.1.1: bytes=32 time=4ms TTL=64”, repeated four times. If this fails, your network setup was not successful.

Now go to serrano and type **ping -c2 ancho**. If it can find ancho, you should get these two messages:

```
64 bytes from 192.168.1.3: icmp_seq=0 ttl=32 time=3.9 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=32 time=2.3 ms
```

The **-c2** argument to ping sends two packets. Without it, ping will have to be terminated with **CTRL-C**.

If this works, congratulations—the Linux machine and the Windows 95 machine can find each other.

Samba Setup

To enable file and print sharing between Linux and Windows 95 machines, you need the Samba suite, which is installed as part of the Red Hat installation.

I wanted to use serrano as a file server so I can install all my Windows 95 software on serrano. Under my home directory on serrano, I created a directory called samba, which will be used to install all my Windows 95 software. I will make this directory (/home/easwaran/samba) visible from ancho as its D: drive. In addition I have a DeskJet 540 printer, attached to serrano, that I want to use to print from ancho. I also want to access the CD-ROM drive on serrano from ancho. My CD-ROM mount point on serrano is /mnt/cdrom, and I would like ancho to see this as the E: drive.

Samba is configured via the file /etc/smb.conf. Read the Samba documentation available at http://www.samba.org/ for more details. My smb.conf file is shown in Listing 3. Samba gives many other options: the original /etc/smb.conf file has many good examples. Also check the Samba documentation (/usr/doc/HOWTO/SMB-HOWTO or http://www.samba.org/).

Listing 3

Mounting Directories from the Linux Machine

Using the smb.conf file, I have made my serrano home directory (/home/easwaran) and the /tmp directory on serrano visible to the Windows 95 machine. The names in square brackets, [easdir] and [cdrom], are the names under which /home/easwaran/samba and /mnt/cdrom will be available to ancho and seen under “Network Neighborhood”. The line

```
log file = /var/log/samba-log.%m
```

toward the beginning of smb.conf file directs Samba to log in to /var/log/samba-log.serrano. In case of trouble, this will be quite useful.

Once you have modified /etc/smb.conf, stop and restart Samba:

```
/etc/rc.d/init.d/smb stop
/etc/rc.d/init.d/smb start
```

In the following instructions, I assume your Windows 95 logon password is the same as your login password for serrano. If they are not the same, you will get a password prompt when you try to mount a directory from serrano on ancho. I make the passwords the same to avoid the hassle, but this may not be a good idea in other situations.

At this point, if you click on “Network Neighborhood” on the Windows 95 machine ancho, you should see serrano listed. Double-click on serrano, and you should see the directories cdrom, easdir, tmp, easwaran, lp and others you may have shared. Open a DOS window and type

```
net use d: \\serrano\easdir
```

You should then see “command completed successfully”, which means that /home/easwaran/samba is available to you on ancho as your D: drive. You can install software and do whatever you want in this directory.

The command

```
net use e: \\serrano\cdrom
```

will mount the CD-ROM on serrano as your E: drive on ancho. Make sure your CD is mounted on serrano as /mnt/cdrom—you may have to do this manually. When I installed Red Hat 5.0, initially only root had mount permission for the CD-ROM drive, so that had to be changed.

To have these drives automatically mounted each time you boot your Windows 95 machine, you can put the **net use** commands in your Startup folder. To do this, look for Windows 95 help under Startup. Alternatively, you could open the

“Network Neighborhood”, select the machine and the directory and click the right mouse button. This will give you an option to “Map Network Drive”. Here, you can specify the drive letter for that directory to be mounted under, and whether this drive is to be mapped at logon time.

Printer Setup

Installing the printer correctly on the Linux machine is easy to do from the Red Hat control panel. We need to make this printer the default printer for ancho. Go to ancho's Control Panel, click “Printers”, then “Add Printer”. Choose “Remote Printer, Browse”. At this point, you should see your printer on serrano listed as lp. Select it, click “OK” and follow the instructions. You will have to load the driver for your printer. (Check the Windows 95 CD or the printer manufacturer's disks or find it on the Web.) Print a test page, and make this your default printer.

Sharing Windows 95 Directories

The next task is to make the C: drive on ancho visible to serrano and other machines on the network. Click on “Network” in the Control Panel and make sure you have “Client for Microsoft Networks” installed (if not, install it). Under “File and Print Sharing”, click on “want to be able to give others access to my files” and “OK”. This may install some more files from the Windows 95 CD. If you now go to “My Computer”, click on “C:” and pull down “Properties”, you should see “Sharing” in the menu; click on it. Then click on “Shared as” and give a share name, “C-Ancho” with Access Type: Full. Type in an access password; I use the same one as my Windows login password. Click “OK”. Now your C: drive on ancho is available to any machine on the network under the name C-Ancho. The icon for the C: drive under “My Computer” will have a blue hand holding the drive.

There is an FTP-like interface that allows you to mount the C: drive from ancho on serrano. If you type **smbclient -L ancho** on serrano, you should get a listing of shared resources on ancho similar to the one shown in Listing 4.

Listing 4

If you type **smbclient \\ancho\C-ANCHO** on serrano, you will be prompted for the password (the one you gave ancho to share its C: drive) and you will get an **smb>** prompt. If you type **ls**, you will see the files and directories on ancho's C: drive. Read the SMB-HOWTO documentation in /usr/doc/HOWTO for more information.

IP Masquerade Installation—Reaching the Outside World

I have a PPP account at my school that assigns an IP address each time I connect through the ppp0 device, which is my default gateway.

To enable machines behind the firewall to communicate with the outside world, you can install IP masquerade. This requires support for the firewall to be built into the kernel. Detailed instructions on how to recompile your kernel with IP firewall support is in the IP-masquerade HOWTO (www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html), available at the LDP (Linux Documentation Project) site. The Red Hat's 5.0 kernel has this support already compiled in. You need to enable IP forwarding on the firewall machine serrano. On Red Hat, this is done by setting **FORWARD_IPV4=yes** in the /etc/sysconfig/network file and restarting network services. Then execute

```
ipfwadm -F -p deny
ipfwadm -F -a m -S 192.168.1.0/24 -D 0.0.0.0/0
```

as root, or place these commands in your /etc/rc.d/rc.local file for automatic execution. For explanation of the commands, read the IP-Masquerade-HOWTO.

You can play with the **ipfwadm** command to selectively grant access to the machines on your network to the outside world. Once this is done, you can open a DOS window on your Windows 95 machine and ping a machine on the Internet. If this works, you can access any machine on the Internet from any machine on your network. To the outside machines, all of the TCP/IP packets will appear to come from your firewall machine.

Configuring the Linux Client

The Linux client (piquin, 192.168.1.100) should be set up with the server 192.168.1.1 as its default gateway. This is done with the route command

```
route add -net default gw 192.168.1.1
```

File sharing between two Linux machines is done easily with NFS. The NFS dæmon, **nfsd**, is usually started at boot time. We would like to make directories on serrano available to other Linux clients via NFS. To do this, specify the directories to be exported in the /etc/exports file. I have the following /etc/exports files on serrano:

```
/home           jalapeno(rw) piquin(rw)
/home/easwaran  jalapeno(rw) piquin(rw)
/mnt/cdrom      jalapeno(rw) piquin(rw)
```

This allows the /home, /home/easwaran and /mnt/cdrom to be exported to jalapeno and piquin, with read and write permission. Reboot the machine after you modify this file, or use these two commands:

```
kill -HUP rpc.nfsd
kill -HUP rpc.mounted
```

On the Linux client, piquin, one needs to edit the /etc/fstab file to enable mounting of remote directories. On piquin, the /etc/fstab file has this entry:

```
serrano:/home/easwaran /home/easwaran/serrano\
nfs defaults,rw,user,noauto 1 1
```

This command allows any user to mount /home/easwaran from serrano on piquin as the directory /home/easwaran/serrano. If **auto** is used instead of **noauto**, this remote directory would be automatically mounted at boot time. To mount this NFS directory manually, type **mount /home/easwaran/serrano**. In this case, because the directory mount point is easwaran's home, easwaran can use the mounted directory as can anyone with permissions to easwaran's subdirectory, serrano. easwaran has read and write permissions on /home/easwaran/serrano.

Linux Remote Printing

To enable the client Linux machine piquin to print on serrano, we need to configure the lp print dæmon **lpd**. See the Printing-HOWTO (/usr/doc/HOWTO/Printing-HOWTO.gz or visit LDP) for additional documentation and alternatives to lp. On serrano, create an /etc/hosts.lpd file listing the machines allowed to print on it. I added piquin to this file.

Printing with lpd is controlled via the /etc/printcap file. On serrano, the file looks like this:

```
# /etc/printcap file
lp|dj:\
        :sd=/var/spool/lpd/lp:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp1:\
        :if=/var/spool/lpd/lp/filter:
```

lp and **dj** are names for the printer; **sd** is the spool directory; the **mx** line means there is no maximum size for files printed; **sh** suppresses headers; **lp** is the actual line printer device; and **if** is a magic filter shell script that deals with staircase effects when printing text files and other special processing that files need. On Red Hat Linux, the control panel has a print configuration tool that allows easy configuration of printers.

On piquin, the /etc/printcap file should look like this:

```
lp|dj:\
      :sd=/var/spool/lpd/dj:\
```

```
        :rm=serrano:\
        :rp=lp:\
        :lp=/dev/null:\
        :sh:
```

rm defines a remote machine; **rp** defines the name of the remote printer; and **sd** is the local spool directory. Make sure the spool directory exists. After making changes to the /etc/printcap file, restart the lpd dæmon or reboot. Now, if you print on piquin, it should appear on the printer attached to serrano.

Backing Up Windows Files on a Linux Box

Listing 5

My Linux server serrano has a tape drive, so I back up all files on my network to tape. To back up the files on the Windows 95 machine ancho, I first use Samba and a Perl script in which I specify a file called MANIFEST-SMB that contains all directories and files on ancho to be backed up. Then I use the Perl script shown in Listing 5. In that script, ****** is my Samba password to connect to C-Ancho. The manifest file can contain blank lines and comments—the first line in the while loop deals with this. It backs up files on ancho specified in MANIFEST-SMB into my Linux machine's /tmp directory. Then I run another script to back up to tape.

This network has provided me with a stable and efficient home work environment for more than a year.

email: easwaran@mcs.newpaltz.edu

Dr. **Chirakkal Easwaran** (easwaran@mcs.newpaltz.edu) is a professor of computer science at the State University of New York at New Paltz. He has worked with Linux since 1994. His research interests are in TCP/IP networks and scientific computation. He thanks the National Science Foundation for a grant that made a Linux laboratory possible at New Paltz.

Archive Index Issue Table of Contents

Advanced search

klogd: The Kernel Logging Dæmon

Michael A. Schwarz

Issue #76, August 2000

klogd reads kernel log messages and helps process and send those messages to the appropriate files, sockets or users. This month we discuss memory address resolution and how to modify klogd's default behavior using command-line switches.

klogd reads kernel log messages and helps process and send those messages to the appropriate files, sockets or users. This month we discuss memory address resolution and how to modify klogd's default behavior using command-line switches.

The syslog dæmon takes messages sent through the syslog applications programming interface (the openlog, syslog and closelog library calls) and dispatches them to various files, sockets, users, pipes, or the bit-bucket (meaning a “discard”--any log message that doesn't match a selector in the syslog.conf file is discarded). Kernel messages, however, cannot use the syslog API.

The basic reason for this is that the syslog API is provided by the kernel itself. Code that can be called by itself is said to be “reentrant”. In order to be reentrant, a secondary call must have no effect on a prior call. If code makes any use of global or static variables, for example, it cannot be reentrant, since a change to a static variable in the secondary call will change it in the primary call.

Think about it this way: what would happen if a program were in the middle of a syslog call when the kernel needed to call the syslog function to report some kernel event? Unless the syslog call were totally reentrant, the kernel call would clobber the user-space call.

Many parts of the kernel are reentrant, either by being truly reentrant or through the use of selective locking. It would be very bad news indeed if a user-space program could block kernel execution merely by getting stuck in a

system call. For this reason (among others), a very clean separation is maintained between the functions that kernel-space code may use and the code that user-space code may use. No kernel API function is dependent on the state of any user-space call (or, to be a bit more precise, those dependencies that exist are predictable and well understood). Thus, the kernel doesn't have to worry about what user-space programs are calling what user-space API functions from moment to moment. The kernel code may simply get on with business.

The kernel does, however, need a way to send messages to report abnormal situations, and, when debugging kernel layer code, to “see how far we got”. Thus, the kernel has its own logging API.

(For the curious, the user-space syslog call is provided by a function in the kernel called **sys_syslog**. The kernel syslog call is called **printk**. You can find the source code for both in your own copy of the Linux kernel source, which is more than likely to be in the /usr/src/linux directory on your system. Go ahead and look. Remember, it's *your* source!)

The kernel is built for simplicity and speed. Thus, the kernel's conception of logging is a bit more basic than that of syslog. Kernel messages are simple text with the convention that a priority of 0 to 7 will be encoded in *<n>* characters (where *n* is the priority, from 0 to 7) prefixed to the rest of the message text. The kernel logging API doesn't have the concept of a “facility” as syslogd does. Level 7 is the lowest priority and level 0 is the highest.

Sometimes, as when a protection fault occurs, the kernel logs contain memory addresses. The protection fault report from a Linux kernel isn't much use to anyone in debugging your problem, because kernels are almost certainly locally compiled. Even if you have never recompiled your kernel, the sheer number of distributions and versions of distributions out there makes it impossible for someone to help you with a raw protection fault log.

Fortunately, Linux kernels since 1.3.43 have reported addresses in a standard format. The klogd program recognizes that format and attempts to resolve addresses to symbol names so that one can actually find the object or code an address refers to. Later on, we'll cover how klogd does this.

These aspects of kernel messages (separate API, non-syslog attributes and memory address resolution) are the reason for a separate dæmon for kernel messages. There were (and are) patched versions of syslog that pick up kernel messages, but this practice is no longer popular, and with good reason. A clean separation of user-space and kernel-space features makes sense. If syslogd did some of these things, it would tie syslogd fairly tightly to the kernel version. Of

course, klogd is bound fairly tightly, but this is considered much more acceptable, since it “encapsulates” this “dependent” code and then passes it on to a standard logging mechanism (syslogd).

So, what klogd does, basically, is read kernel log messages, transforms them slightly (by resolving kernel memory addresses to symbols) and then calls the user-space syslog API with the kernel facility and the priority as encoded in the kernel message. This is the dæmon's default behavior. Let's take a look at how the default behavior can be modified.

First off, klogd does not have a configuration file as syslogd does. Its behavior can be modified only through command-line switches and signals. We'll cover the switches first, then we'll discuss address resolution. Finally, we'll go over the signals to which klogd responds.

Command-Line Switches

-c Default console logging level. The kernel writes log messages not only to the kernel message buffer, but also to the system console (usually /dev/console). The default level for the kernel is 7, which means that messages of a value lower than 7 (higher priority) are written to the console. Often, you will want to change this once klogd is running, so the console isn't always scrolling through a lot of low-priority messages. The klogd/syslogd combination gives you quite a bit more control over your kernel messages than simply dumping them to a screen. You can specify a number *n* here (ex. -c 4) where messages of a value lower than but not equal to *n* will go to the console. Note that klogd doesn't route messages to the console itself. It merely provides this interface to change the kernel's setting of the console logging level. Keep in mind that lower values of *n* are higher priority messages.

-d Debugging mode. This generates lots of output on **stderr**. Give it a try if you're curious, although I do not recommend running this way for any length of time.

-f Log messages to file. This switch allows you to bypass the syslogd interface and log kernel messages directly to a file. You lose all of syslogd's ability to separate messages by facility and priority, to route a message to multiple destinations, and to route to pipes, sockets and users. It has obvious value, however, if for some reason you aren't running syslogd! (Ex. **-f /var/log/kernel.log**)

-i, **-l** Signal the currently running klogd. We'll go over these two switches (they are distinct!) in the section on memory address resolution.

-n Do not auto-background. There are three ways you might run a dæmon: by command at the console, by startup script, or directly with the System V init model (/etc/inittab). When you run with init, you don't want the process to “fork and die” (which is how a *nix process puts itself in the background; see chapter 2.6 of W. Richard Stevens' excellent book *UNIX Network Programming* if none of this makes any sense) as you would in the other two cases. Generally, this need not concern you if klogd is already running on your box.

-o One-shot mode. When started with this option, klogd will read all the messages presently in the kernel log buffer, and then it will exit.

-p Paranoia mode. This changes when klogd loads kernel symbol data. We'll cover this in more detail in the section on memory address resolution.

-s Force system call mode. Normally, klogd checks at startup for the existence of the /proc/kmsg file. If it is there, this is opened as the place to read kernel messages. If it is not there, klogd will poll the kernel through a system call for kernel messages. The /proc/kmsg is favored because it has lower overhead, especially when there are no kernel messages (which is a common case). You can override the preference for the /proc/kmsg interface and force klogd to use the system call instead with this switch.

-k Kernel symbol file. See the section on memory address resolution.

-v Print version and exit. This document is based on klogd 1.3-3.

-x Do not resolve addresses. See the section on memory address resolution for more information.

Memory Address Resolution

(The following discussion presumes Linux running on an x86 processor. I would imagine other processors are similar, but I have not examined the code for them, so I'm not prepared to state that the following holds true for those processors.)

Let's begin by noting that real protection exceptions resulting in kernel logs are very rare events. Most protection faults occur in user-space code. User-space protection faults result in a program termination and core file dump. You can use the core file and your favorite debugger to post-mortem the application. These events hardly bother the Linux kernel, which merrily goes on handling all the other applications in the system.

The faults we are talking about here are processor exceptions that happen in kernel code. These are so rare that I have seen only five since I started using

Linux in 1993. Three of them occurred when I was using the “TAMU” Linux release from Texas A&M University. We're talking pre-0.99 Linux. I think that was to be expected. The next occurred when I had a dying hard drive and my swap partition was the defective area. The fifth and last occurred when I had an overheating CPU in my laptop. Since 1994, I haven't seen one for any reason, excepting a hardware failure.

That said, they do happen. Some never-before-used combination of hardware leads to a combination of kernel code never previously run; or perhaps you are a daring soul and you are running a development kernel. Whatever the reason, sometimes good code goes bad. The good news is Linux is an open-source OS. You can fix the bug. Or if not, you can post a bug report that goes directly to the people who can fix the bug. Try *that* with Windows!

When a protection fault occurs, Linux dumps out a dump of the processor state, including all the registers and the last several entries of the system stack. The latter is critical for finding the source of the problem. Trouble is, the raw dump consists entirely of memory addresses. Since Linux is an open-source system and since many installations have been custom compiled, the likelihood that these addresses will help anyone at a support desk to figure out the problem is small indeed.

Luckily, if you built your kernel in a normal way, there is a file called System.map installed with your kernel (probably in /boot). This maps code and symbols to physical addresses. The klogd dæmon reads this file. This takes care of all the “compiled-in” kernel code, but since the 2.0.*x* kernel series, Linux has supported kernel modules, which are dynamically loaded kernel-code modules. These could be at any address, depending on which are loaded at a given moment and in what order.

At program start, or in response to a signal, klogd will query the kernel for a list of modules and their load addresses. Kernel modules may register individual function or identifier addresses with the kernel when they are loaded. The klogd dæmon will use this information to report addresses in a fault dump. It is important to note that module addresses from klogd can be out of date! If modules are loaded or unloaded after klogd is initialized, then these module/address resolutions will be incorrect. Your distribution may take care of this for you by providing scripted utilities to refresh klogd automatically. If it does not, then some of the switches we skipped over earlier come into play to help you keep the memory map up to date.

The **-i** switch tells klogd to reload the module symbols. The **-I** tells klogd to reload the System.map file. The **-p** switch enables “paranoia” mode. What this does is cause klogd to attempt to reload the module symbols whenever it sees

“Oops” in the kernel message stream. Protection faults have this string in them. I personally consider this kludgy, and I don't use it. Also, if there has been a protection fault, it is possible that the kernel is about to halt or the memory map may be in a corrupt state. It is available if you want it. The **-k** option allows you to specify the file that contains the kernel symbol information. See the section on multiple kernels below. The **-x** switch tells klogd to not read kernel and module symbols and simply to dump the protection fault messages untranslated.

Multiple Kernels

Linux beginners are unlikely to encounter this, but more seasoned users will often have more than one bootable kernel on their system at a time. If the box is a hobbyist or kernel hacker's box, it is likely to have a number of stable and a number of development series kernels on it. I myself always keep three generations of stable kernels on my systems, so that if a bug should show up, I can immediately reboot into the older kernel.

When klogd starts, it identifies the kernel version (all kernels since 1.3.43 put version information in the map files) and then looks at:

```
/boot/System.map
/System.map
/usr/src/linux/System.map
```

It will use the kernel version information to choose the correct one, if possible. When I have a development series kernel on a box, I leave the stable kernel map in /boot and I leave the development kernel map in /usr/src/linux. As for my two “old” stable kernels, I just live with the fact that klogd will not be able to resolve addresses in the event of a fault if I boot into them. Remember, you can use the **-k** switch on klogd to force it to use a particular map file if you build an archive of them.

Just bear in mind as you read this discussion that these events are so rare that in 15 machine years (three machines running Linux 24x7 for five years), I have seen this happen twice, and both times it was due to failing hardware.

Signals

In addition to the command-line switches, klogd will respond to certain signals. You send signals with the **kill** command.

The signals klogd responds to are:

- SIGTSTP/SIGCONT

SIGTSTP suspends and **SIGCONT** resumes kernel logging. The resuming includes re-initialization, so you can use this to, for example, unmount the /proc file system without killing klogd:

```
kill -TSTP <pid>
umount /proc
kill -CONT <pid>
```

- SIGUSR1/SIGUSR2

SIGUSR1 causes a reload of the kernel module symbols. **SIGUSR2** reloads both the static kernel symbols and the module symbols.

See the Memory Address Resolution section for more information.

- SIGINT/SIGHUP/SIGKILL/SIGTERM

These signals all gracefully shut down klogd.

Summary

The klogd works with syslogd to handle the dispatch of kernel messages. It exists solely because the kernel itself is unable to use the syslog API directly. Klogd provides for resolving raw memory addresses into kernel symbol names.

Michael A. Schwarz (mschwarz@sherbtel.net) is a consultant with Interim Technology Consulting in Minneapolis, Minnesota. He has 15 years of experience writing UNIX software and heads up the open-source SASi project. He has been using Linux since he downloaded the TAMU release in 1994, and keeps the SASi project at http://alienmystery.planetmercury.net/.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

kHTTPd, a Kernel-Based Web Server

Moshe Bar

Issue #76, August 2000

Linux kernel developers realized that a kernel-based web server was needed.

Web servers have become an important part of today's infrastructure for business, trading, entertainment and information. Some of the Web's sites take millions of hits every day, or even every hour. It is only natural that computer science researchers soon began wondering how to make web servers faster, more resource-efficient and fail-safe.

The search for more speed triggered a whole new area of operating system theory research, that of execution path analysis. The author of this article, while a Ph.D. student, researched this topic at length studying the Apache server. One of the most interesting discoveries was that for static pages, more than 80 percent of the instructions are actually executed in kernel space (protected mode). This has some serious implications.

First, as we learned in the previous three columns, the only way to enter kernel space for a user program such as Apache is to execute a system call. System calls are expensive because they involve complex checks and search many kernel tables.

Also, switching from user space and back often flushes the on-processor TLB (Translation Lookaside Buffer) cache as well as primary and secondary cache entries.

As a consequence, Linux kernel developers realized that a kernel-based web server was needed. Such a kernel-space web server would not incur the costs involved in switching back and forth to and from protected mode.

Just such a kernel-space web server, called kHTTPd, was implemented in Linux kernel versions 2.3.*x* and 2.4. kHTTPd is different from other kernel web servers in that it runs from within the Linux kernel as a module (device driver).

kHTTPd handles only static (file-based) web pages, and passes all requests for non-static information to a regular user-space web server such as Apache or Zeus. Static web pages, while not complex to serve, are nevertheless very important. This is because virtually all images are static, as are a large portion of HTML pages. A “regular” web server adds little value for static pages; it is simply a “copy file to network” operation. The Linux kernel is very good at this; the NFS (network file system) dæmon, for example, also runs in the kernel.

“Accelerating” the simple case of serving static pages within the kernel leaves user-space dæmons free to do what they are very good at: generating user-specific, dynamic content. A user-space web server such as Apache, typically loaded with many features and many execution paths, can't be as fast as kHTTPd. There are, however, a few web servers that are as simple as kHTTPd but implemented in user space, so they are not expensive consumers of processor cycles, even compared with kHTTPd.

kHTTPd is very simple; it can't handle dynamic content. So, it proxies all requests for those directories you configure via the sysctl called “dynamic” to a fully functional user-space web server such as Apache. It's a global win, though, since most of the transfers of a common web server are images, which are definitely static.

kHTTPd is actually not much different from a normal http dæmon in principle. The main difference is that it bypasses the syscall layer. Normally, an http server contains code like this:

```
socket(..)
bind(..)
listen(..)
accept(..)
```

and each call has to enter the kernel, look up kernel structures as function(s) of the parameter(s) passed, return information to user space, etc.

Being a kernel dæmon itself, kHTTPd interfaces directly with the internal kernel structures and system calls involved and so avoids the user-kernel interaction completely. Also, because it's a kernel dæmon, it avoids **switch_mm** and TLB flushes. Last but not least, it avoids all enter/exit kernel overhead.

There are not many data structures for kHTTPd. They are in net/kHTTPd/ structure.h.

The first is a per-connection structure. The second is a per-kHTTPd-thread structure by which many http_requests can be queued.

Listing 1. /proc/sys/net/khttpd/structure.h

kHTTPd can be compiled as a loadable module, or linked statically into the kernel. Linking statically into the kernel will provide better performance, because it will be allocated in a more efficient and TLB-persistent page-table mapping.

Configuring kHTTPd

Control of kHTTPD is performed via the /proc filesystem at /proc/sys/net/khttpd. Table 1 shows the sysctl parameters which can be set, along with a description of each (from the docs).

Table 1. Parameters Configurable through /proc/sys/net/khttpd

Conclusions

kHTTPd might be a viable solution for web sites with high traffic and mostly static web pages. Using kHTTPd is very easy, and tests with this setup have shown a dramatic increase in throughput compared to standard user space web servers. The code itself has proven very stable and safe. Once again, Linux is a technology leader. Right on!

email: mbar@cmp.com

Moshe Bar (mbar@cmp.com) is an Israeli system administrator and OS researcher, who started learning UNIX on a PDP-11 with AT&T Unix Release 6 back in 1981. He holds an M.Sc. in Computer Science. Visit Moshe's web site at http://www.moelabs.com/.

Archive Index Issue Table of Contents

Advanced search

Linux on Wheels: A New Opportunity

Linley Gwennap

Issue #76, August 2000

As in-dash computer technology evolves, will Linux become a contender for this popular new market?

Linux fans may be aware of the Empeg in-dash MP3 player, a Linux-powered computer that stores and plays back gigabytes of MP3 files in the car. (See "MP3 Linux Players" by Craig Knudsen, *LJ*, July 1999.) The Empeg player is an undeniably cool device, but it is just the tip of the proverbial iceberg. Within five years, almost every car will have a powerful computer that provides a variety of services. Linux is a contender to win a large share of this market, which has no dominant player and will generate tens of millions of sales per year.

Modern automobiles are already a seething mass of computing resources, packing anywhere from 30 to 80 networked microprocessors that control windows, doors, airbags, lights, gauges, brakes, engine timing, temperature and anything else that moves or changes. These inexpensive chips are too small and slow to run Linux. But now automobile equipment makers are planning to take the next step, turning the car into a "$30,000 wireless Internet device," says Mike Iannitti, director of Intel's In-Car Computing Division, with only a hint of a smile.

A Compelling Feature Set

Several concepts are converging to make an in-dash computer appealing. Various single-function devices exemplify these concepts today. The Empeg player, for instance, brings to the car the advantages of MP3, including instant access to a huge number of songs. Other digital music formats, such as digital FM radio and satellite radio, will become popular in the next few years, all requiring some sort of decompression software. A programmable platform will be flexible enough to handle new algorithms and protocols. Within five years, any car without these capabilities will be as out of date as a car with an AM radio and an 8-track player is today.

Navigation assistance is a second useful feature. Some high-end vehicles and rental cars come equipped with GPS systems that determine the car's location and plot a route to a specified destination, often providing audible directions (“turn left in 100 feet”). Anyone who has ever gotten lost while driving will appreciate the value of such a system. And unlike backseat drivers, it can be turned off.

A third area is safety and security. General Motors already has 150,000 customers for its OnStar system, which uses the GPS and a cell phone to contact a call center if the vehicle crashes or is stolen, providing the car's exact location. The call center can also unlock the car's doors if you forget your keys, and can help with other problems. Future systems could improve security by validating a voiceprint or fingerprint before allowing the engine to start.

Another helpful function is information access. This doesn't mean running a browser on the dashboard; rather, it is an intelligent mechanism for retrieving and delivering relevant information such as traffic reports, restaurant or ATM locations, or even just e-mail messages. The OnStar call center can direct a driver to a nearby hotel or restaurant. Other services already provide wireless Internet access.

Voice recognition will be a key feature in these car computers, as people can't type or press buttons at the same time they're trying to drive. By the end of this year, Delphi will begin shipping a Palm V docking station that allows drivers to access address and datebook entries using voice commands. The unit has a built-in cell phone that can be “dialed” using voice or by selecting an address book entry. Each of these functions is appealing to certain drivers. But the combination of these functions, particularly if the price is right, will be unstoppable. Acceptance of in-dash computers will be driven by two factors: platform standards and price.

Rapid Adoption Path

Today's first-generation products typically cost $800 to $1,000 and perform only one or two of the functions listed above. Current prices are bloated, because upfront development costs must be spread across a small number of units. As production volumes rise from tens of thousands to tens of millions, prices will drop dramatically. The cost of key components, such as microprocessors, flash memory and GPS receivers, is also falling due to normal semiconductor trends. For example, a GPS chipset that cost $100 two years ago sells for about $30 today, and will drop to $10 in another two years. These trends will help bring the price of an in-dash computer down to a few hundred dollars within the next few years.

The value of an in-dash computer will increase as vendors realize that all of the important functions require a fast microprocessor, a flexible operating system and a wireless interface. Combining digital music, navigation, information access, security and voice recognition into a single device will add little hardware cost but significant end-user value. The bigger difficulty is in creating the necessary software, particularly since new technologies and services are likely to continue to be developed after the car is purchased. Digital music formats are in flux and not likely to settle down for at least a few years. Information services will continue to evolve, and the broad deployment of car computers will undoubtedly spur a new round of innovation.

Ideally, the car computer should be built around a standard platform that allows third parties to develop applications. Without a standard, consumers will be confused and application development stunted. Microsoft hopes to play this role with Windows CE, which is used in Delphi's Palm V dock, Clarion's AutoPC and Visteon's ICES among others. But Linux's low cost, simplicity and open-source model should be attractive to auto equipment makers. WinCE faced little competition for these early systems, but more recently, a throng of embedded Linux vendors has risen to the challenge. The stakes are high. Worldwide auto sales totaled 54 million last year, about half the size of the desktop PC market. Full-function in-dash computers will appear as standard equipment in some model-year 2002 cars (just over a year from now) and will move rapidly through the product lines. Ford, for example, boldly claims that all its cars will have Internet capability within three years. This hot new market gives Linux the opportunity to establish itself as the platform of choice.

email: linleyg@linleygroup.com

Linley Gwennap (linleyg@linleygroup.com) is the founder and principal analyst of The Linley Group (http://www.linleygroup.com) analysis firm in Mt. View, California. He is a former editor in chief of Microprocessor Report.

Archive Index Issue Table of Contents

Advanced search

Organizing your $HOME

Marcel Gagné

Issue #76, August 2000

I have files I don't even recognize. And yet, I am afraid to delete them.

François, move those boxes down into the wine cellar, please. While you are down there, give the “Dom” a quarter turn as well. Oh, and bring up the '78 *La Tache* when you come back up. After all this re-arranging and straightening up, I think I will need to relax a bit.

Mon Dieu! Mes amis, I had not realized you were here. Forgive me and forgive this mess. With all the work that has been going on around here, I had not noticed it was opening time. Please, sit down. I will call down to François and have him bring something special from the cellar. On the house!

François! We have guests. A case of Dom Perignon for our friends!

You know, mes amis, when we cook with Linux, we can get a little carried away. After many experiments in the kitchen of your processor, your **$HOME** can become quite the mess, non? I have files I don't even recognize. And yet, I am afraid to delete them. What if they are important? What if I need them?

One solution is to move. Moving a user to a new directory is child's play with Linux. For instance, if I had a user called “natika” with a home directory of /home/natika, and I somehow felt that Natika would be better served in a new directory called /home/boullabaise, I would issue a command like this:

```
# usermod -d /home/boullabaise -m natika
```

The “-d” specifies the new directory, while the “-m” tells **usermod** to move everything from /home/natika to /home/boullabaise for the userid “natika”. If the directory does not already exist, it will be created. The problem is that everything is moved, and any clutter you may have had will follow you to your new **$HOME**.

Speaking of clutter ... were you ever curious about the vast array of .jpg files you've scattered across your system? Finding them in order to better organize them can be done with the **find** command.

```
# find / -name "*.jpg" -print
```

Do you want to time that command, mes amis? Depending on the power of your system, this can take awhile and you are likely to get a lot of “permission denied” messages (particularly as you traverse **/proc**) or when running as a non-privileged user. Luckily, there is a faster way. On most Linux systems, you have a process that runs once a day. That process builds a database of all the files on your system for quick and easy searching. The command is called **locate** or **slocate**. The process that runs on your system is located in /etc/cron.daily and called **slocate.cron**. If your system is not up 24 hours (it *is* a Linux system, non?), you can build the slocate database by running the cron script manually, or with the command **updatedb**. Let us try to find those .jpg files again.

```
# slocate jpg
```

Amazing, is it not? I should tell you that the find command is quite a bit more powerful than slocate, but if you need to lay your hands on a file quickly and you have no idea where it has gone, try slocate. However, if what you are trying to do is locate old files or particularly large files, then you should definitely use the find command. Here's an example where I look for anything I have not modified (the **-mtime** parameter) or accessed (the **-atime** parameter) in the last twelve months.

```
# find /data1/Marcel -size +1024 \
\( -mtime +365 -o -atime +365 \) -ls
```

The backslashes are escape characters, there to make sure our shell does not interpret them in ways we do not want it to, in this case, the open and close parentheses on the second line. The first line also has a backslash at the end. This is to indicate a line break, as the whole command will not fit neatly on one line of this page. Were you to type it exactly as shown without any backslashes, it would not work; however, the backslashes in line two are essential. Ah, but there is something else I have not told you. The above command also searches for files that are greater than 500KB in size. That is what the “-size +1024”, means, since the “1024” refers to 512-byte blocks. The **-ls** at the end of the command tells the system to do a “long” listing of any files it finds that fit my search criteria. You may want to pipe this output to your printer in order to better examine the results. Another option is to redirect it to a file, which you can later peruse and decide on what course of action to take.

Earlier, I mentioned that I'm afraid to delete any of these files because I don't always know what they are. While your humble chef may have been exaggerating a tiny bit, I can tell you how you can improve that search to help

you uncover the secrets behind those files. Your system contains a command called **file** which can deliver useful information on files and what they are. If I modify the find command by adding a **-exec** clause, I can yield even more information.

```
# find /data1/Marcel -size +1024  \
\( -mtime +365 -o -atime +365 \) -ls -exec file {} \;
```

The open and close braces that follow **-exec file** means that the list of files generated should be passed to whatever command follows the **-exec** option. The backslash followed by a semicolon at the end is required in this case.

find is a wonderful command in other ways. For instance, I can use it to look for **SUID** or **GUID** files, which is extremely useful for security reasons. It is simply a good idea to check your system for files that are set to execute as root and decide whether they should be there or not. I also use find to locate core files and editor temp files, those strange little files with the tilde (~) at the end of the file name. And, of course, working so much in the Linux kitchen means I have source files for hundreds of applications I have tried and tested for you, mes amis. Have a look at the following sample script which I have written to take all these ideas into consideration.

```
#!/bin/bash
#
# Locate files and report to me.
# Marcel Gagné, 2000
# search_log=/tmp/foundfiles
rm -f $search_log
touch $search_log
#
echo
"------------------------------ " >> $search_log
echo "Looking for big old files . . . " >> $search_log
echo "------------------------------ " >> $search_log
#
find /data1 -size +2048 \( -mtime +180 -o -atime +180
\)  \
   -ls -exec file {} \; >> $search_log
#
echo "------------------------------ " >> $search_log
echo "Looking SUID / GUID files . . . " >> $search_log
echo "------------------------------ " >> $search_log
#
find / -type f \( -perm -2000 -o -perm -4000 \) -ls >> $search_log
#
echo "------------------------------------------------ " >> $search_log echo
"Looking for core files or old editor files . . . " >> $search_log echo
"------------------------------------------------ " >> $search_log
#
find / \( -name core -o -name "*~" \) -print >> $search_log
#
echo "------------------------------ " >> $search_log
echo "All done!" >> $search_log
```

Of course, my **$HOME** (or my whole system, for that matter) is not the only place that needs organization. There are so many things to keep track of these days that Chef Marcel needs additional, Linux-style help. How about some electronic sticky notes? You already have 40 or 50 covering your monitor,

cubicle wall or desktop (the wooden one, as opposed to the virtual one on your monitor). Why not transport the concept to your Linux desktop? KDE comes with a little program called **knotes**. You can find it under the Utilities menu, or just type **knotes** from a terminal window. Unlike your classic paper sticky, you can also set alarms with these notes and they do not use up paper.

Yellow sticky notes are fine and quite frankly, I have found them to be extremely useful on a number of occasions. Unfortunately, they do not offer the flexibility of a planner or calendar. Once again, with your Linux system, you are ready and able to better start planning your life. If you are running GNOME, you can use the program **gnomecal** to keep yourself organized. KDE users have **korganizer** to keep them focused. We focus a great deal on both KDE and GNOME users, since desktop applications tend to be designed with ease of use in mind. However, GNOME and KDE are not the only alternatives, and many people use lighter, faster desktops. (For this article, I have been using Window Maker as my window manager.) To those looking for a nice cross-desktop alternative, I recommend **ical**, a clean, simple X window calendar program.

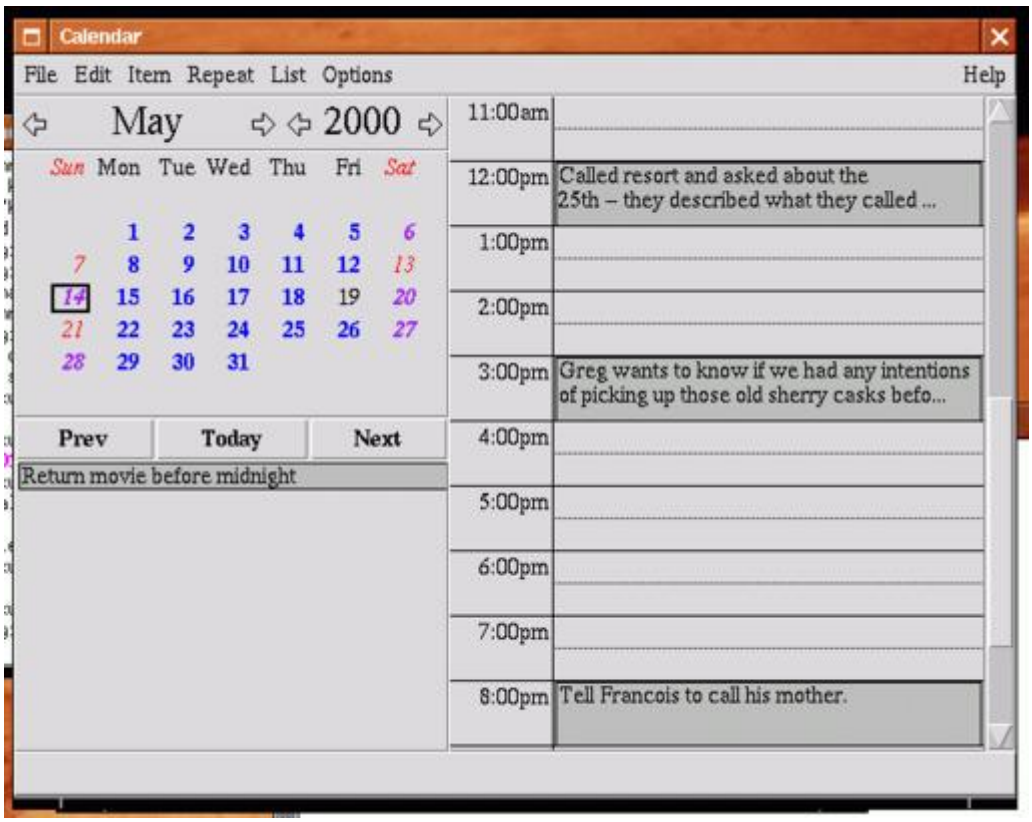


Figure 1. The ical Program

While not the most powerful calendaring software, ical stands on par with gnomecal mentioned earlier. This little program is likely already installed, and almost definitely on your CD-ROM distribution disk. Start the program by typing **ical**. The initial screen is a month-view calendar, a to-do list and a single-page day planner view. You can also share calendars with other members of your office by clicking on “File” and “Include calendar”.

Some time ago, your humble chef admitted he needed even more help than ical alone (or electronic yellow stickies, for that matter) could provide. He needed something portable, something that followed him everywhere. That help comes in the shape of his Palm Pilot. While the friendly folks at 3Com were good enough to create this little wonder, they did not provide Linux software for backups and hot-syncs. Luckily, the Linux community of talented developers have come through to help your humble restauranteur stay organized as well as backed up.

Each and every tool I mention from here on requires the **pilot-link** suite of utilities. You should not have to go hunting for this package, since it is likely already installed on your system. If not, check your distribution CD. One sure way to find out is to try the command **read-ical**. **read-ical** is an interface program that will read your Palm Pilot information and load it into a calendar file that can be used by ical. Here's the format for the command.

```
read-ical /dev/pilot calendar_filename
```

In the case of my system, **/dev/pilot** is a symbolic link to **/dev/ttyS0**. I created that link with this command:

```
ln -s /dev/ttyS0 /dev/pilot
```

After issuing the read-ical command, you should click on the “hot-sync” button on your Palm Pilot's cradle. Be warned, mes amis. This process is *not* a true hot-sync. Each time you execute this command, the calendar file name you specified above is overwritten. For this reason, you should use a different file than your normal everyday calendar file (or your company-wide file). I used a file I called pilotcal. To use it with ical, I simply clicked “File”, then “Include Calendar”. I specified the file named “pilotcal”, and suddenly, all my appointments and to-do items were available.

Before I move on to what I think of as “Super Stickies”, I should mention that the KDE Korganizer program also allows you the means of sync'ing with your Palm, as does another cool KDE program called Kpilot. I have used Kpilot for quite some time now, but lately I've discovered another player in this field, something called J-Pilot. This one won't be on your CD, so you will have to visit the web site (see Resources). Looking very much like its inspiration, the Palm desktop software distributed for *that other OS*, J-Pilot mimics what is already a familiar interface for users of the Palm Pilot. Written by Judd Montgomery, J-Pilot is available as a source tarball or an RPM (Red Hat, Caldera, Slackware 7).

But what about combining the venerable yellow sticky with the flexibility of the Palm Pilot? Is there nowhere to turn? Luckily, Michael J. Hammel (of Graphics Muse fame and a frequent contributor to this magazine) has written

XNotesPlus. This is a nifty little package that extends the power of the sticky well beyond that afforded by knotes.

You can pick up the source, a GTK version or a Motif/LessTif version from the Graphics Muse web site. I picked up the GTK version, which at the time of this writing was version 3.2.2. Installation of XNotesPlus is a breeze. After untarring and unzipping the distribution, simply run the install script from the distribution directory. The steps are as follows:

```
tar -xzvf XNotesPlus-v3.2.2-Linux-gtk.tar.gz
cd xnotes
./install
```

The script will confirm with you the location of your xnotes database, as well as the path name to the executable. You will also get instructions to add the following lines to your .bash_profile so that you can sync with your Palm Pilot.

```
export PILOTPORT=/dev/pilot
export PILOTRATE=57600
```

To use XNotesPlus, issue the command **xnotes+**.

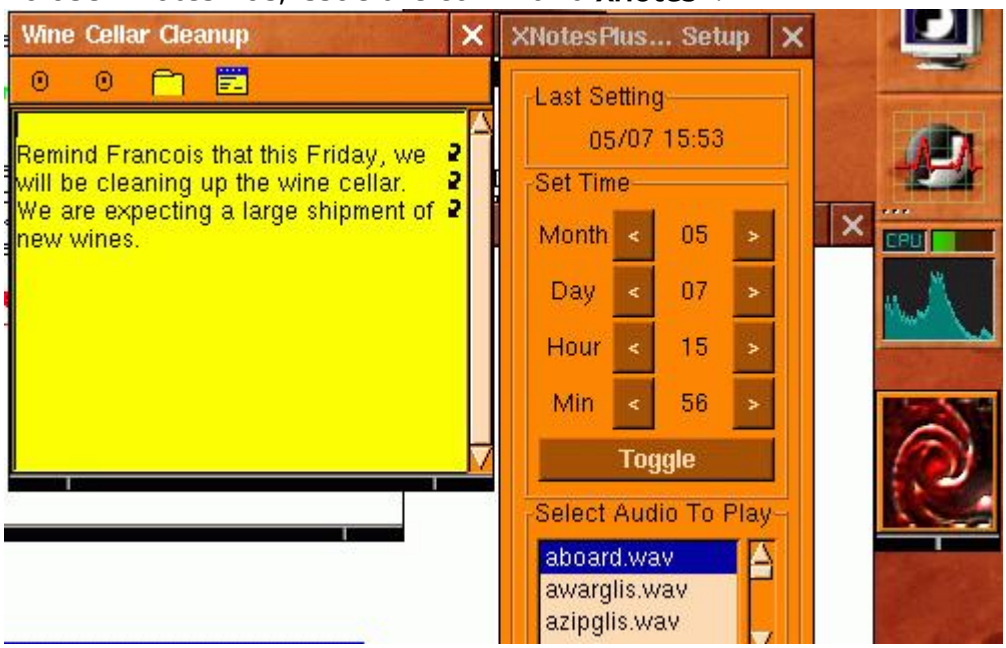


Figure 2. Setting an Alarm with XNotesPlus

C'est vrai. There is a plus sign at the end of that command. The application appears with a nice little Window Maker-like icon which you right-click to do various things (it has a little swirl that looks sort of like a galaxy). Among the features you'll find (other than covering your virtual desktop in yellow) is a Palm Pilot sync utility, the ability to print envelopes from your address book, set alarms with little musical reminders, e-mail notes, or, if you really don't want to clutter your desktop, you can even hide the notes.

Before I leave XNotesPlus, I should point out that while it is free software and source is provided, it is not GPL'ed. Read the accompanying “COPYING” file for license details.

Well, mes amis, it is that time once again, when we must part. Next time you visit, I hope to be a little more organized. While it is good for you to sip free champagne, it is not so good for me, non? Never mind; enjoy. In fact, have François pour you a final glass before we close the door. When next we meet, your table will be ready here at Chez Marcel.

A vôtre santé! Bon appétit!

Resources

email: mggagne@salmar.com

Marcel Gagné (mggagne@salmar.com) lives in Mississauga, Ontario. In real life, he is president of Salmar Consulting Inc., a systems integration and network consulting firm. He is also a pilot, writes science fiction and fantasy and edits TransVersions, a science fiction, fantasy and horror magazine. He loves Linux and all flavors of UNIX and will even admit it in public. You can discover much from his web site at http://www.salmar.com/.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Session Management with Mason

Reuven M. Lerner

Issue #76, August 2000

This Perl-based web helper and MySQL work together to let you quickly build a user registration system for your web site.

Over the last two months, we looked at the Perl-based web development known as Mason. Mason, written and maintained by Jonathan Swartz, is based on the idea of “components”, flexible templates that can contain HTML, Perl code, or a bit of each. Mason makes it possible to create large, dynamic web sites in a relatively short period of time. Moreover, it lends itself to easy and extensive code reuse, removing many of the common maintenance issues associated with web sites.

Because Mason is traditionally run on top of **mod_perl**, an Apache module that places a complete Perl binary inside of the web server, it can take advantage of other Perl modules developed for mod_perl. One particularly useful example of such a module is **Apache::Session**, which makes it possible to get around some of the problems associated with HTTP's statelessness.

This month, we will look at Apache::Session, using it to create a simple user registration system based on Mason. This system can make it relatively simple to create a personalized web site, connecting information in a relational database with a particular user.

Session Management

HTTP was designed to be a lightweight protocol, with each transaction taking a minimum amount of time. As a result, it is fairly minimalist, with each connection consisting of a single request-response pair. (Modern versions of HTTP support multiple request-response pairs within a single transaction, but my impression is that the single-transaction version 1.0 is still the norm.)

In this model, the HTTP client connects to the server, sends a request and an optional parameter, and then one or more headers that describe the browser's capabilities. The HTTP server then returns one or more headers describing the response, followed by the response itself. The response can be an HTML-formatted text document, an image, or an error message indicating that the request could not be fulfilled. After the server sends this response, it closes the connection.

Because each HTTP transaction takes place in a vacuum, without any information from other transactions, it is difficult to keep track of a user's actions. The web has no sense of “logging in” or “logging out”, unlike a traditional computer environment. It is impossible to know whether five HTTP requests were initiated by five separate users on the same computer, or one user interested in five different URLs.

Two main techniques get around these problems. The first, called “cookies”, allows the server to store a name-value pair on the user's computer. The cookie is set in a “Set-Cookie” header at the beginning of the server's HTTP response. Every time the browser returns to a site within this server's domain, it sends a “Cookie” header as part of the request, with the name-value pair that was previously stored. Cookies are limited in length, can be deleted by a browser at any time and can easily be inspected and modified by a user.

Another technique, which we will not explore this month, involves the use of a URL's “path_info” segment. For example, consider the URL www.example.com/cgi-bin/foo.pl/abc/def. If /cgi-bin/foo.pl exists on the server, then /abc/def is passed as an additional argument that exists separately from any name-value pairs submitted from the client.

While neither cookies nor path_info is a perfect solution for the issue of state on the Web, they are sufficient for most needs. However, these solutions address only the problems with HTTP; they don't provide a means for giving our programs a sense of state.

Apache::Session bridges this gap, making it possible to associate arbitrary information along with a user. (We will soon discover that things are not quite this simple, but the overall principle is sound.) **Apache::Session**, which is available from CPAN, works with either cookies or path_info, and can store information using mechanisms ranging from ASCII files to relational databases. It is designed to work with mod_perl, and thus works with Mason; the documentation indicates that Apache::Session should also work under CGI, although I have not tested this claim.

Because of their versatility and speed, and because Apache::Session works best when associated with additional information in a relational database, we will use MySQL for our back end, called the “object store” in the module's documentation. In order to do this, we will need to create a table named “sessions” in our database, which looks something like this:

```
CREATE TABLE sessions (
    id CHAR(16),
    length INT(11),
    a_session TEXT
);
```

Apache::Session requires the table to be named **sessions** and that it contain three columns: an **id** column of type **CHAR(16)**, a **length** column of type **INT(11)** and an **a_session** column of type TEXT or BLOB, which can contain any amount of binary data.

Each unique session is identified by a unique 16-character string, stored in the id column. The actual session data is stored in the a_session column, in the “nfreeze” format defined by the Storable module. (Storable is also available from CPAN.)

Apache::Storable and Mason

Each time a user's browser sends an HTTP request to the web server, it sends whatever cookies have been stored by that domain. So if a cookie was set by cnn.com, my browser will return only that cookie—which is, after all, simply a name-value pair—when I visit http://www.cnn.com/ again.

The cookie version of **Apache::Storable** takes advantage of this by storing a unique identifier in a cookie. This unique identifier corresponds to the id column in the sessions table. This allows us to retrieve any data that have been stored in a_session. Because a_session is defined to be infinitely long, the amount of data we can store is limited only by our database and our file system.

Data stored in table sessions by Apache::Session is available to programs via the global **%session** hash. **%session** is created anew for each incoming HTTP request, and refers to only the data stored in a_session. Storing something in %session places it in the a_session column, and retrieving something from %session gets the value from a_session. Assuming that the variables **$first_name**, **$last_name** and **$email** contain the appropriate pieces of information, we could store them reliably with the following lines of Perl:

```
$session{first_name} = $first_name;
$session{last_name} = $last_name;
$session{email} = $email;
```

Since each user (actually, each session) is stored in a separate row of the database, we do not need to worry about users clashing with each other.

In order for sessions to work, we must make a connection between the **Apache::Session::DBI** module and the corresponding sessions table on disk. This connection must take into account three different possibilities: (a) that the user sends us a valid ID cookie, (b) that the user sends us an invalid ID cookie, and (c) that the user sends us no ID cookie at all.

The first case is the easiest; the program merely needs to re-establish the connection between %session and the appropriate row in sessions, using Perl's "tie" mechanism. In the second case, the program must create a new session if it could not re-establish a previous one. And if the user sends no cookie at all, then we must create a new row in sessions, attach a unique ID to it and send that unique ID to the user's browser in the form of a cookie.

When working with Mason, we put all this in our start-up file. This file, which the Mason documentation calls handler.pl (but which I prefer to call mason.pl), defines all of Mason's main behaviors and allows us to define global variables that other elements of the system will require. Defining %session in mason.pl also ensures that it is available in all Mason components. See Listing 1 for a simple example of mason.pl for a site that wants to include sessions. (Much of Listing 1 comes straight from the Mason documentation.)

Listing 1

The most important part of this file is a call to Perl's **eval** command. **eval** comes in two forms, one of which takes a code block as an argument, and forms as a primitive form of error-checking. Inside our code block, we attempt to use Perl's **tie** command to connect the hash **%HTML::Mason::Commands::session** to the Apache::Session::DBI module. Tying these two together means that the default storage and retrieval mechanism associated with hashes no longer applies for %session—when we retrieve or modify its value, one or more methods in Apache::Session::DBI will take over:

```
eval {
    tie %HTML::Mason::Commands::session, 'Apache::Session::DBI',
        ($cookies{'AF_SID'} ? $cookies{'AF_SID'}->value() : undef),
        {
         DataSource => $dbsource,
         UserName => $dbuser,
         Password => $dbpass
        };
};
```

If this eval is unsuccessful, the variable **$@** will contain the error message. Here, we test to see if the object exists in the data store. If so, then we assign the user a new session:

```
if ( $@ )
{
    if ( $@ =~ m#^Object does not exist in the data store# )
    {
        tie %HTML::Mason::Commands::session,
'Apache::Session::DBI',
            undef,
            {
             DataSource => $dbsource,
             UserName => $dbuser,
             Password => $dbpass
            };
        undef $cookies{'AF_SID'};
       }
}
```

Finally, if the user does not pass us any identifying **AF_SID** cookie at all, we create a new one and tell **mod_perl** to send it along with the rest of the outgoing headers:

```
if ( !$cookies{'AF_SID'} )
{
    my $cookie =
       new CGI::Cookie(-name => 'AF_SID',
                       -value =>
                       $HTML::Mason::Commands::session{_session_id},
                       -path => '/',);
    $r->header_out('Set-Cookie', => $cookie);
}
```

Once these are in place, any Mason component can store and retrieve information in %session. **Apache::Session's** use of the Storable module means that references and complex data structures (such as arrays of arrays, and hashes of hashes) can be stored in %session without us having to worry about losing data.

What Do We Store?

Just because we *can* store anything in **%session** does not mean we necessarily should. For instance, a site that wants to keep track of users' names and e-mail addresses could potentially store this information in %session. While doing so makes the information readily available from within Mason components, it creates other problems. For instance, it would be difficult to retrieve the rows of “sessions” and use them to create a mass mailing to subscribers' e-mail addresses.

For this reason, I generally use Apache::Session to store only one value, the primary key associated with the user's row in a **Users** table. (There are other ways to accomplish the same task, such as including the user's unique 16-character ID field in the Users table and adding a “UNIQUE” constraint on it.) If **$session{user_id}** exists, then we can assume the user has previously registered, and use that value to retrieve other information from Users. If $session{user_id} does not exist, then we assume the user is new to our system.

Here is one possible definition for a Users table which we can use in this way:

```
CREATE TABLE Users (
    user_id MEDIUMINT AUTO_INCREMENT,
    username VARCHAR(30) NOT NULL,
    email VARCHAR(50) NOT NULL,
    password VARCHAR(20) NOT NULL,
    password_hint VARCHAR(60) NOT NULL,
    PRIMARY KEY(user_id),
    UNIQUE(username),
    UNIQUE(email)
);
```

We define all of the columns in this database as NOT NULL, meaning that they are mandatory fields. Aside from the user's unique ID (which is automatically generated by MySQL), user name and e-mail address, we require a password and a password hint. As we will see, these will allow us to create a full login system, and to handle some of the problems associated with HTTP cookies.

Registration Components

Now that we have defined a Users table, it is time to define some Mason components. Some of these components will be similar to subroutine, and others will be similar to HTML fragments. As we saw last month, both are acceptable (and welcome) types of Mason components. I typically use an .html suffix on top-level components that are visible to the user, and a .comp suffix on others—but you may wish to set up your own conventions.

Before we do anything else, we will need a component that allows us to connect to the database, and to retrieve a database handle (traditionally known as **$dbh**). Because Mason typically runs under mod_perl, we will take advantage of the **Apache::DBI** module, which keeps a database connection open even after an HTTP request has been served. Reusing database connections in this way dramatically increases the speed of our application, since logging in to a database can be relatively slow.

Listing 2 contains a simple Mason component that connects to the database and returns a valid **$dbh**. By putting this functionality inside one component, we avoid having to include that code inside every other component on the site. Moreover, it means that if we have to modify the data source name (“DSN” in Perl lingo), we can do so by changing one file.

Listing 2

Notice how database-connect consists solely of **<%perl>** and **<%once>** sections, without any HTML. This is an example of a component that acts purely as a Perl subroutine, returning a value to its caller. By contrast, Listing 3 contains register-form.html, a top-level component that contains only a few lines of Perl.

The majority of register-form.html is straight HTML, and can be written by a graphic designer, rather than a programmer.

Listing 3

Registering is a relatively straightforward process. Information typed into register-form.html is sent to register.html (see Listing 4). The latter retrieves the name-value pairs from the form, placing them into scalar variables using the Mason **<%args>** section. If one or more elements are missing, register.html gives the user an error message indicating that the information needs to be updated.

Listing 4

If the user's registration information appears to be complete, register.html performs a quick SELECT to ensure that the user name will indeed be unique. True, we have defined the table such that a user name must be unique, but we would rather produce a nice-looking error message for our users than display an error message from the database.

Note that this code creates a race condition; it is possible that two users could try to register with the same user name simultaneously. Both would be told that the user name is available, and yet only one would be allowed to insert the requested user name. Databases that support transactions, such as PostgreSQL, can avoid this problem by wrapping the **SELECT** and the following **INSERT** into a single transaction, which can then be rolled back if there is an error.

Listing 5

register-form.html attempts to be somewhat helpful, reminding users if they are already logged in. (After all, there usually isn't any reason to register if you're already logged in.) It uses the component get-user-info.comp (see Listing 5), which takes one argument (a user ID) and returns a hash reference describing the user with that ID. Since user IDs are stored in %session with the **user_id** key, we can retrieve a hash reference with user information as follows:

```
my $user_info = $m->comp("get-user-info.comp",
                         user_id => $session{user_id});
```

If **$session{user_id}** is undefined—that is, if the user has no session—then get-user-info.comp returns undef. Otherwise, a program can retrieve information for the user with the hash reference's keys. Indeed, the top of register-form.html demonstrates this:

```
% if ($user_info) {
<P>You are currently logged in as <b><% $user_info->{username} %></b>. Do
you really want to register?</P>
% } else {
<P>You are not logged in. Go ahead and register!</P>
% }
```

Logging In and Out

register.html automatically logs in a user. By this, we mean that it sets the value of **$session{user_id}** to a valid primary key for the Users table. When $session{user_id} is set, a user is said to be logged in; when it is undefined, the user is not.

Listing 6

Logging out a user, then, is as simple as undefining the value $session{user_id}. We do exactly this in Listing 6, logout.html. Once a user visits this page, he or she is no longer logged in. Note that the line

```
undef $session{user_id};
```

does not remove the user_id key from %session. Rather, it assigns the undefined value to $session{user_id}.

If a user fails to log out, then the session will remain active for as long as the session cookie exists. Cookies are normally assigned an expiration date when they are created, indicating the maximum date on which they should be transmitted to a server. If no expiration date is mentioned, the cookie should disappear when the user exits from the browser. Session cookies are normally set with the latter expiration date, forcing them to disappear when the user quits from the browser.

However, this doesn't mean that users can ignore the “logout” button. On the contrary, someone who fails to log out is effectively saying that any HTTP requests originating from a particular computer should be attributed to his or her user name. In a typical office, where everyone has their own computer, this might not be a serious issue. However, a student in a recent class I taught told me that she was able to read someone else's e-mail at an Internet cafe, because Yahoo! Mail had failed to log out the previous user.

If the information is particularly sensitive, you might want to force users to re-register every 15 or 30 minutes. Simply set the cookie expiration date and time to be something in the very near future, and the cookies will expire automatically.

Logging in is slightly more complicated, in that we must ask the user for a user name and password. These pieces of information, supplied from login-form.html (listing 7), are passed to the **login.html** component (listing 8). **login.html** performs two tasks: it submits a **SELECT** query to the database, requesting the user_id column for the submitted user name and password. If no such row exists, **$sth->fetchrow_array** returns undef, and we thus know that the user does not exist. If it does exist, then we retrieve all the relevant information about this user into a hash reference and set $session{user_id} to the newly rediscovered user ID. This restores the session information to the user's browser, which sets it in a cookie (or path_info, as appropriate).

Listing 7

Listing 8

While there is no room to discuss it here, it would obviously not be very difficult to create a “password-remind.html” component which allows users to retrieve their password using the hint they entered in the initial registration form.

Of course, personalized sites are rather uninteresting if they store only the user's name and e-mail address. Things get much more interesting if the site keeps track of users' interests, birthdays and stock portfolios. But once we have a unique ID that represents this user—the user_id column in Users—we can create as many tables as we like, identifying each user with their primary key.

Conclusion

Session management can be a tricky subject when working with the Web, since it means using a stateless connection for something it was never intended to do. With the help of Mason and Apache::Session, it is not difficult to develop a personalized site which keeps track of users' interests and customizes the site's output accordingly.

Resources

Reuven M. Lerner , an Internet and Web consultant, moved to Modi'in, Israel following his November marriage to Shira Friedman-Lerner. His book Core Perl will be published by Prentice-Hall in the spring. Reuven can be reached at

reuven@lerner.co.il. The ATF home page, including archives, source code and discussion forums, is at http://www.lerner.co.il/atf/.

Archive Index Issue Table of Contents

Advanced search

LINUX JOURNAL

Focus on Software

David A. Bandel

Issue #76, August 2000

Bartleby, smtm, ttnews and more.

How have you weathered the recent (hopefully finished) spate of e-mail worms (mistakenly called virii) passing among our Microsoft-using associates? Linux could just as easily be the target. While its environment is much more hostile, I still see too many posts to mail-list servers where the sender address is “root”; these folks are very susceptible, and so are the files of many non-privileged users. It could happen to you. By default, most distributions of Linux are wide open. Even those users who know it's not wise to read mail as root may have no concept of how to secure their system from Net vandals. True, .vbx scripts can't hurt us, but many of the services run by default systems are able to. You can take comfort in the fact that the Linux/UNIX community has helped by adding new rules to Sendmail to strip off .vbx attachments that might have wiped out hundreds of thousands of systems. Be safe; shut off services you don't need, and make sure those you do need are current.

Bartleby: http://www.dahak.com/bartleby/

The author says the purpose of this program is to easily log changes to systems. All you do is send a very short e-mail to sysadmin (a one-line echo statement will work well). However, **bartleby** works great for many other things as well, e.g., a quick note when a check is sent off, or when other daily tasks are completed. Messages can easily be categorized by using a server name as a note category (perhaps using a client's name). This message is parsed and inserted into an SQL database, which can be accessed via a web browser. Look for more improvements to this program soon. It requires Perl, MySQL (or another relational database), DBD/DBI Perl modules, a web server that can run cgi scripts, an MTA (Sendmail, smail, etc.) and a web browser (Lynx, Netscape, et al.).

smtm: rosebud.sps.queensu.ca/~edd/code/smtm.html

Show me the money! This particular utility will fetch quotes from your list of stocks, so you can see how badly or well they are doing. Display is configurable and will show price, last change, volume traded and more. Negative price changes are reflected in red. It requires Perl 5 with the following Perl modules: English, Date::Manip, File::Spec, Getopt::Long, HTTP::Request::Common, IO::File, POSIX, Tk, Tk::Balloon and Tk::FileSelect.

ttnews: www.student.lu.se/~etn97ksi/sim_home/ttnews.html

This news ticker is for those who crave information overload. Even I couldn't handle all the news it gave me with a default install, and that's a lot of news. Fortunately, **ttnews** comes with information on how to configure it, and it's not difficult to do. It requires libqt2, libX11, libXext, libstdc++, libm, glibc, bash and Lynx.

arping: synscan.nss.nu/programs.php3

I travel to client sites often and like being able to connect my laptop to their network and run tests against the system I'm working on, grab files off my system and more (including keep up with my e-mail while programs compile). To do so, I usually have to bug the administrator to give me an unused IP if he's not running DHCP (permission to connect is usually easy to obtain, but an IP is much harder). Now there's a better way. This little utility allows me to ping a couple of IPs before I grab one that's already in use and stomp on it. It requires libnet and glibc.

sendEmail:

marvin.criadvantage.com/caspian/Software/SendEmail/default.php

I'd rather not install programs like Sendmail on a firewall or other “secure” system; I like to keep what's on these systems to a minimum. While Perl (which is needed for this program) isn't exactly small, I use it for other things anyway, so I've found **sendEmail** to be a useful addition to my bag of tricks. The sendEmail utility is very powerful. In fact, it looks like the next best thing to connecting directly to the Sendmail server to which mail is delivered. While it won't defeat a closed relay, it will allow you to rewrite the header in just about any way you might wish. It requires Perl and the Perl Socket module.

SICKnotes: sourceforge.net/project/?group_id=3942

SICKnotes can provide you with a web-based view of your system inventory, configuration and knowledge base. It might be overkill for a small home network; however, most businesses need the kind of information this system stores. And while you may have this information already, it is most likely stored

in several places rather than one (and possibly not available from just any network-connected system). SICKnotes is an IT information system worth looking at, since it can probably be reconfigured easily to almost any department. It requires Python, MySQLdb Python module, MySQL, a web server and a web browser.

cidr: home.netcom.com/~naym/cidr.html

Ever have one of those days where you just can't seem to remember the netmask for a /25 address, or maybe the network and broadcast addresses that correspond to it? Perhaps I'm just brain-fried, but I sometimes find myself staring at a screen where I have already typed **ifconfig eth0 192.168.0.6 netmask** and thinking, “Gee, I know this, I must use it every other day, so why can't I remember it now?” Well, just feed **cidr** the basic information, IP address and /25 or whatever, and the program provides the rest. At this rate, I'll be able to stop thinking completely in another two or three years. It requires glibc.

eatcookies: http://www.ii.uib.no/~otto/eatcookies/

Don't like cookies? Well **eatcookies** is one hungry little program that will gobble them up for good. Run it once—it will eat all your Netscape cookies (at least those you allow) and won't leave any crumbs. The program comes with a Netscape loader that automatically runs eatcookies after Netscape exits, cleaning your cookie file after each use. You can adjust eatcookies' diet via its diet file. This may be little consolation, since some companies have figured out how to keep your cookies on a third host where you can't get to them, but that technology isn't widely used yet. It requires Perl.

email: dbandel@pananix.com

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

Archive Index Issue Table of Contents

Advanced search

SOC it To 'Ya!

Rick Lehrbaum

Issue #76, August 2000

A host of system-on-chip silicon devices will help push embedded Linux into the post-PC era.

We've all heard about the coming “post-PC” era. Are we there yet? Is it just around the corner? Or is the post-PC era just a bunch of hype? After all, most of us still use conventional PCs at work and home to do our computing.

First, realize that the opportunity for computerized devices which aren't PCs is practically limitless. International Data Corporation (IDC) reports that out of the nearly two billion microprocessor chips manufactured each year, over 95% go into non-PC “embedded” devices. Today, most of this figure represents low-level control tasks in vending machines, cars, test instruments, sprinkler systems, etc.

But the smartness and interconnectedness of the myriad of computerized devices that surround us will soon increase dramatically. Recent technology advances make it possible to embed PC-level computing, communications and display capabilities within common appliances. They're also enabling the creation of many new kinds of electronic gadgets—a process that is currently happening at a furious pace in thousands of ongoing projects. Which means nine months from now (products take roughly the same time to gestate as human babies), the results of this frenzy of post-PC development will begin to emerge in a big way.

So here's my prediction: 2001 will be “the year of the post-PC”.

Each technology era tends to be characterized by a handful of key, new technology threads that are woven into its fabric. In the case of the post-PC era, those threads are likely to be:

- the Internet

- Wireless LANs
- Embedded Linux
- Voice recognition
- “System-on-chip” integrated circuits

Let's zoom in on one of them: system-on-chip (SOC) integrated circuits.

System-on-Chip: The Holy Grail of Embedded Systems

Chip companies have long sought to develop the means to build entire systems on one piece of silicon. Imagine a single super-high-density chip that you could program to perform all the electronic functions your system needs. We're not there yet, but we've certainly made a lot of progress.

As a hardware designer, I've personally experienced much of that evolution. In 1980, the floppy controllers I designed required entire boards full of chips. Soon, the designs compressed into two chips. Then one. Ultimately, the whole thing literally vanished—sucked into a super-I/O chip! Just a mere speck of silicon and a few interface pins are all that remain.

As Moore's Law continued to work its magic, the half-dozen circuit boards of the original PC eventually met similar fates. One by one, each board-level controller became several chips, one chip, and finally a fraction of a multi-function IC. By the end of the '90s, the functions of a PC had been reduced to a handful of chips. Would the final result be a single-chip PC?

The obvious answer is “yes”. But the actual answer, for the moment, is probably “no”. Why? It turns out there are still laws of physics to contend with. Intel started pushing back on the PC-on-a-chip idea several years ago, arguing that it's neither efficient nor practical to implement everything on one piece of silicon. Some partitioning of technologies, they reasoned, is necessary in order to extract maximum benefits from the silicon and to achieve the best cost and performance.

General-purpose PCs, and larger computers used as servers or control systems, demand maximized CPU performance. Device interfaces like Ethernet, sound and LCD controllers have more specialized needs. These two priorities—CPU performance and device interface—tend to make conflicting demands on the silicon. That's why Intel has opted not to build its Pentium processors with on-chip video, Ethernet and sound functions.

On the other hand, when you set out to create a web pad, set-top box, Internet radio or smart vending machine, you face an entirely different set of challenges. After all, you're designing an appliance, not a PC. These embedded

applications tend to be interface-intensive, rather than compute-intensive. So you probably don't need Intel's latest Pentium. In the past, you may have used a single-chip microcontroller (8051, 68HC11, etc.). Today, there's an exciting new alternative: the post-PC, Linux-oriented system-on-chip. That's quite a mouthful! Let me explain.

- post-PC: Embedded applications don't need to be “Wintel” compatible, since they're not trying to be general-purpose PCs. Their CPUs can be x86 variants, but there can be big advantages to RISC processors. Why? Because RISC CPUs tend to be more efficient in their use of silicon resources. That can result in higher performance and functional integration, at lower power consumption and cost. Which is all great, provided you've got the operating system software and tools you need to make the device work the way you want. Which leads us to ...
- Linux-oriented: In my opinion, embedded Linux will be a key enabler of the post-PC era. Why? One reason is that Linux isn't wedded to just one CPU architecture. Another is that Linux is highly scalable, modular and flexible—which makes it well-suited to the extreme diversity of embedded systems. Then, there's the fact that Linux is open source, so it's much easier to get your embedded widget to act like it's supposed to. Also, don't forget the Linux royalty model—zero (or nearly so)—which makes all this great stuff affordable in even the simplest of devices.
- system-on-chip: Right now, a whole new class of system-on-chip processors is emerging. They're popping up everywhere, on an almost weekly basis. These highly integrated and easy-to-design-with little tidbits of silicon now contain powerful 32-bit CPUs, along with a feast of built-in peripheral interfaces. Best of all, they're nearly all supported with ready-to-run embedded Linux.

A Post-PC SOC Checklist

Here's my checklist of minimal requirements for a post-PC SOC:

- 32-bit CPU
- Built-in interface to RAM and ROM
- Built-in DMA, interrupt and timing controllers
- Built-in interface to disk or flash memory
- Built-in Ethernet and/or LCD/CRT interface
- Built-in serial and parallel ports and/or USB
- Full embedded Linux support

Why require either built-in Ethernet or a display controller? That's because some SOCs go in “black box” devices that don't need displays, such as firewalls, specialized servers or gateways, while others go in user-interactive devices like

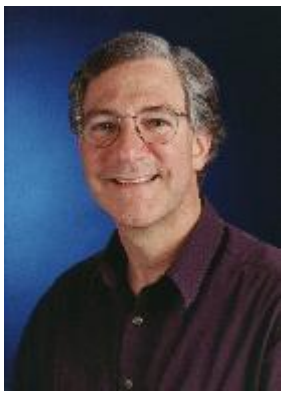
web pads and vending machines. By including at least one of these two popular external world interfaces, the SOC implements all the key functions of the required embedded computer.

Are there any SOCs that currently meet all the requirements on my checklist? You bet! For several months, I've been gathering info on post-PC Linux-oriented SOCs. Here's a sampling of what I've found.

- STMicroelectronics STPC Industrial—contains an 80MHz x86 CPU, plus a CRT/LCD display controller, PCMCIA and serial/mouse/keyboard ports. The STPC Consumer is another version, with a slightly different mix of features. www.linuxdevices.com/links/LK9677697611
- Motorola PowerPC MPC823e—contains a 75MHz PowerPC CPU, a sophisticated communications signal processor, plus a CRT/LCD display controller, PCMCIA, seven serial ports, USB, I2C and SPI. www.linuxdevices.com/links/LK9083681498
- IBM PowerPC 405GP—contains a 266MHz PowerPC CPU, plus 10/100 Ethernet, serial and parallel ports and I2C. www.linuxdevices.com/links/LK5925887918
- NETsilicon NET+ARM—contains a 40-MIPS ARM7TDMI CPU, plus 10/100 Ethernet, two high-speed sync/async serial ports with HDLC and SPI support, four IEEE-1284 parallel ports and 24 digital I/O pins. www.linuxdevices.com/links/LK7209982157
- Aplio/TRIO—contains a 20-MIPS ARM7TDMI CPU, a pair of 40-MIPS DSPs, plus two serial ports, SPI, a pair of CODECs, 10/100 Ethernet, USB and Flash memory interface. The DSPs and CODECs provide software modem, audio and voice functions. www.linuxdevices.com/links/LK3569182901
- Axis ETRAX—contains a 100-MIPS RISC CPU, plus 10/100 Ethernet, IDE, SCSI, two IEEE-1284 parallel ports and four high-speed serial ports. www.linuxdevices.com/links/LK5309816751
- Intel SA-1110—contains a 206MHz StrongARM RISC CPU, plus an LCD display controller, four serial channels, IRdA, USB slave and 48 digital I/O lines. www.linuxdevices.com/links/LK7515567179
- NEC Vr4181—contains a 66MHz 64-bit MIPS RISC CPU, plus interfaces for LCD display, CompactFlash, a serial port, IrDA, keyboard, USB, touch panel and audio in/out. www.linuxdevices.com/links/LK4338659220

Bear in mind, this list represents the tip of the SOC iceberg. New SOCs are announced continually, so check for the latest information at LinuxDevices.com. Use the site's search function, with “system-on-chip” as the keyword.

The post-PC era is just around the corner and there's little doubt that embedded SOCs, combined with embedded Linux, will be two of its principal enablers. So get ready for some exciting changes in the electronic gadgets that surround us, as those devices become much more intelligent and much more connected.



email: rick@linuxdevices.com

Rick Lehrbaum (rick@linuxdevices.com) created the LinuxDevices.com "embedded Linux portal", which recently became part of the ZDNet Linux Resource Center. Rick has worked in the field of embedded systems since 1979. He co-founded Ampro Computers, founded the PC/104 Consortium and was instrumental in launching the Embedded Linux Consortium.

Archive Index Issue Table of Contents

Advanced search

Hold the Back Page

Stan Kelly-Bootle

Issue #76, August 2000

On Eric S. Raymond's The Cathedral and The Bazaar

Extending the mnemonic, disyllabic fame of A&M (Hymns Ancient and Modern), S&M (don't ask) and K&R (Kernighan and Ritchie), I propose C&B for Eric S. Raymond's *The Cathedral & The Bazaar*, O'Reilly, 1999. Here's a book that will survive Guy Kawasaki's hyper-blurb: “The most important book about technology today, with implications that go far beyond programming.” It may even, in future editions, distance itself from Bob Young's foreword, written before the Linux Red Hat CEO started challenging Bill Gates as Mister Midas #1.

As a longstanding Jolt judge in the book category (since 1991, in fact, when *Software Development* magazine initiated these prestigious Academy-type awards; see note 1), I've been judiciously cautious in my reviewing strategy, resisting all but the most perverse publicational PR sexual enticements. My critical objectivity has been further strained by the fact that, over the years, I've been personally involved with many of the writers under scrutiny. Also, authors are more aware of the hard slog involved than are unpublished doryhores (see note 2) and tend to review with more sympathy and fewer sour grapes. A classic example from the early UNIX days was the late Professor Jim Joyce of UCB. He never, alas, got around to writing the “perfect” UNIX book, but gained infamy with his deflating reviews of other authors' attempts. Of one, he wrote, “I won't say there's a mistake on every page, but there are certainly more mistakes than pages.”

Yet in praising C&B, I swear I'm uninfluenced by a feeling of friendship and gratitude to Eric Raymond stretching back to his seminal work (with Guy L. Steele, Jr. and others) on the weird delights of hacker culture and its lexicon. (Eric also kindly reviewed my *Devil's DP Dictionary* back in 1981, and encouraged my update, *The Computer Contradictionary* (1995)--but again, I deny any cozy back scratching!)

C&B is, first off, a great read, written for a wide audience including many who when polled “Which OS do you use?” answer “What's an OS?” The main theme, as you must know, is the OSS (Open Source Software, see note 3) revolution: what, why, how and whence. In particular, the theme is the unpredictable successes of GNU, Linux and Apache, based on an approach that seems to defy the sacred rules of cathedral-capitalism (or what Radio Tirana used to call the “howling Wall Street jackals”, see Note 4).

The gist is that many thousands of programmers are contributing their time and skills on a voluntary, unpaid basis. The motivations vary, but one that Eric stresses is the traditional hacker *egoboo* (“egoboost”), and I suppose, an appearance on my new TV show, “Homes of the Poor and Famous”.

Even stranger, and generating mixed feelings among the OSS purists, is how OSS has, in many instances, formed a symbiotic relation with the big-money bottom-liners who are rushing to exploit the Linux phenomenon. (At the recent ACM2000 Awards, Apache won the coveted IBM Systems' prize.)

The wonderful thing about C&B, unlike the spate of bland software-development and people-ware books, is the violent controversy generated, some of which rises beyond the boring anecdotal to the exciting *ad hominem*. The most detailed attack on C&B has come from Bertrand Meyer (another esteemed friend, mon dieu), available at www.sdmagazine.com/features/2000/03/f4.shtml.

I've looked on OSS from both sides—and what strikes me is how often each side ignores ecumenical approaches from the other. Bertrand declares, “Accept that both commercial and free software have a role to play and that neither will ever go away.” Eric makes similar statements. But where's the controversial fun in that?

Notes

Stan Kelly-Bootle (skb@atdial.net) has been computing on and off since his EDSAC I (Cambridge University, UK) days in the 1950s. He has commented on the unchanging DP scene in many columns (“More than the effin' Parthenon”--Meilir Page-Jones) and books, including *The Computer Contradictionary* (MIT Press) and *UNIX Complete* (Sybex).

upFRONT

Various

Issue #76, August 2000

Stop the Presses, *LJ* Index and more.

THEY SAID IT

“Society is always taken by surprise at any new example of common sense.”

—Ralph Waldo Emerson

“You can never entirely stop being what you once were. That's why it's important to be the right person today, and not put it off till tomorrow.”

—Larry Wall

“I'm what you get when an eleven-year-old kid decides he wants to grow up to be a Heinlein character.”

—Eric S. Raymond

“FEATURE”

—VW Bug license plate spotted by ESR in California

“Kill processes, not people.”

—Larry Cohen

“You can make millions without exploiting. But not billions.”

—Alan Kay

“In battle, exploiting a situation deserves a medal. In business, exploiting a situation deserves condemnation.”

—Craig Burton

“Trust always breaks down first over money.”

—David Hodskins

“Web browsers are fragile assemblies of bugs, held together with Hello Kitty stickers. They tend to have problems with complicated pages, especially if they're long. Some of my pages are pretty long, and I want them to work with any web browser on any computer, and so I've been very careful not to do anything unkosher, daring, or cool.”

—James “Kibo” Parry

THE UNTAKEN

Last September, we ran a little piece called “Barrel Scrapings” that listed a couple dozen domain names that were still untaken. Since then, the domain name business has turned into an e-land rush. At the current rate, several new domains are claimed every second. That's a lot of “brands”, all bent on capturing your eyeballs, or whatever it is that e-marketers are after these days.

So we thought we'd step back into the fray to see just how insane this business has become.

First, we took a look at some of the names we suggested almost a year ago. “Earwig.com” is taken. So are “stoptalking.com”, “toygod.com” and “cashbird.com”. But “coaptathetic.com”, “bedkill.com”, “buttcramp.com”, “neithersex.com”, “petsurface.com” and “barfwash.com” are all still available. Hope lives.

In fact, hope abounds. Thanks to the good folks at Register.com, we found many domain names already taken, like “feces.com”, are blessedly unconfused by a host of derivative names. So Register.com kindly lists a pile of those names, each produced by adding an e-cliché prefix (e.g., “about”, “quick” and “cyber”) or suffix (e.g., “biz”, “search” and “dot”). So, if you're disappointed by missing your chance with “crap.com”, you can choose from any or all of the following:

aboutcrap.comcrapdot.comfreshcrap.comcrapmeat.comcrapforge.comcocrap.comcrapauctio

Sadly, mycrap, cybercrap, ecrap, quickcrap, crapstore, crapsite and b2bcrap—all .coms—are taken.

But to prove there is still an abundance of untaken names, here are just a few that remain unclaimed.

butthook.compalmhair.comhopeloss.comflyseason.comceleprosy.combuttsleeve.comworses

—Doc Searls

NEOLOGISTICS

A couple words from our resident linguist.

Tune in, turn on, click out.

What's the opposite of “sticky”? Try “clicky”. A clicky site is one with plenty of interesting hyperlinks to elsewhere. What better to support than curiosity?

Mo fo

A few weeks ago I was talking with Tim O'Reilly about the noun “information”, which lately has been replaced with “content”. Not that we've lost much. Both now carry the meaning of cargo: something we aggregate, store, move, address and deliver. Where we're moving, clearly, is away from real meaning.

Information, we noticed, derives from the verb *inform*, which derives from the verb *form*. In conversation, we don't just “deliver information” back and forth. We form each other. When I learn something new from you and what I learn is meaningful—that is, I can't forget it—you have literally formed me. In other words, we are authors of each other. What's more, we are in the market to be formed. We demand it. Otherwise, we wouldn't learn a damn thing.

That's why it's misleading to conceive of information as a substance we “deliver” to each other (or worse, to an “audience” of “eyeballs”). When we do that, we insult the verb at information's heart.

So how can we start to recover some of the meaning lost when information becomes content? I suggest abbreviating “information” to “of”. When we're in the market for something meaningful, we don't want content, or even information. We want the real deal, pure and uncut. Give us of.

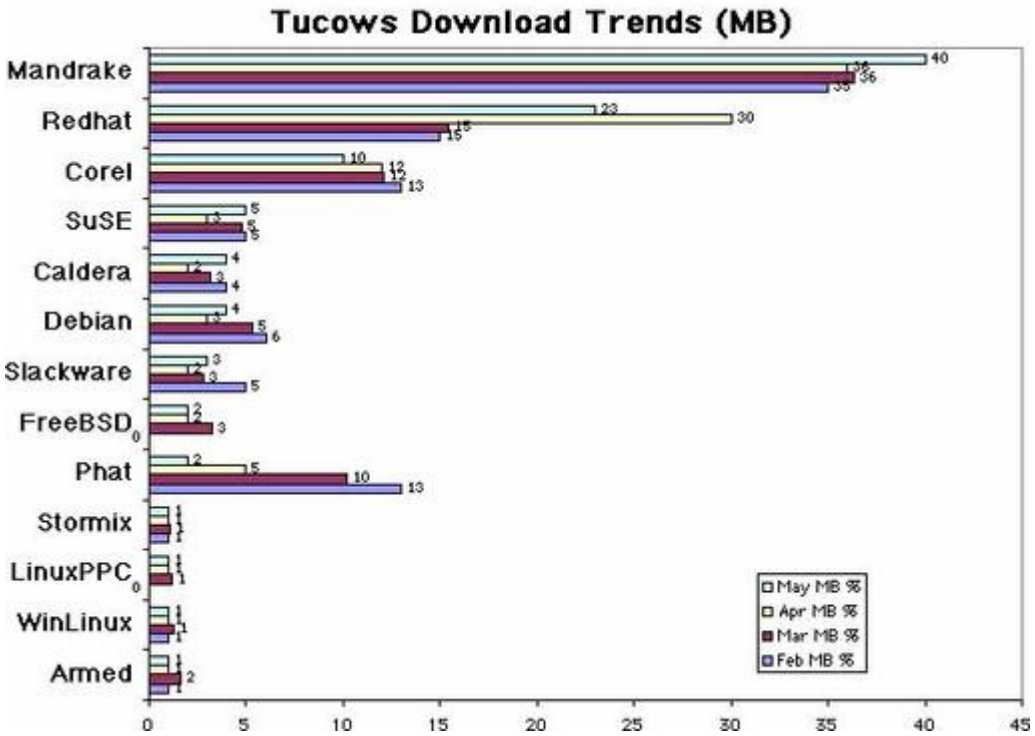
Okay, “clicky” has a better chance. But time will of.

—Doc Searls

TUCOWS DOWNLOADS

After a slight slip due to a strong Red Hat surge in April, Mandrake bounced back to increase its lead in Linux distribution downloads from Tucows in May. Measured through May 25, Mandrake was first with 40%, followed by Red Hat with 23%. That's a gain of 4% for Mandrake and a drop of 7% for Red Hat. Next was Corel, which dropped 2 to 10%. SuSE gained 2 and reached 5%. Slackware had 3%, FreeBSD had 2 and so did Phat, which continued its drop from a high of 13% in February. The rest were at 1% or less.

Now for an interesting statistical effect. Astute readers will remember that last month, Red Hat took the lead, with 33% to Mandrake's 31%. Like this month's May figures, April was then tabulated through only the first 25 days. What a difference just five days makes. In this month's report, we look at all 30 days of April, and the difference pushes Red Hat down 3% and Mandrake up 5%. This leaves Red Hat in second place and Mandrake in first: exactly the positions they've maintained since February. Tune in next month to see if those last five days make any difference in the final May numbers.



Smile, you're on a 1-pixel camera!

Web pages use a publishing metaphor. They are pages, after all. We write, open, read and bookmark them. That's one reason we assume that when a page downloads from a server, it's a one-way deal. The HTML describes the page, lays out the print, loads the graphics onto the page and into the cache.

There is at least the presumption of privacy. After all, reading is a personal (even an intimate) act. At times when interaction is required, such as filling out

a form, there's a “submit” button that sends information back to the other end of the line. We're still in control.

And okay, we know about cookies and what they do. If we're especially vigilant, we either refuse to accept them, or we go through the whole pile and weed out the suspicious ones.

But watch out. Big Brander is watching you. And not just with cookies.

It turns out that some companies are spying on you and your web travels by using invisible 1x1-pixel transparent GIFs. These are in-line images downloaded from elsewhere, so the server-browser dialog can initiate covert reconnaissance on you and your subsequent surfage. You see no ad, and suspect nothing. The cookie alert doesn't go off. But the bug—as in bugging device—has been planted.

Who's doing the planting? Usually, a company that wants to learn something about you. Most of the time it's an advertising service that wants to “target” you with banners, e-mail spam or whatever. But they can serve all kinds of purposes, known and unknown. Don Marti found one on a Fed Ex page. (Presumably, they want to track customers the way customers want to track a package.) Richard Smith, the leading source of information on Web Bugs, found two on Quicken's home page, both providing “hit” information to advertising companies.

According to Smith's Web Bug FAQ www.tiac.net/users/smiths/privacy/wbfaq.htm, here is the information a Web Bug sends back to its server:

- the IP address of the computer that fetched the Web Bug
- the URL of the page the Web Bug is located on
- the URL of the Web Bug image
- the time the Web Bug was viewed
- the type of browser that fetched the Web Bug image
- a previously set cookie value

Of course, any graphic can serve the same purpose. That's why the only way you can see a Web Bug is to view a page's HTML source. But Smith also provides another way to at least discover what sites are playing slight-of-pixel games: The Web Bug Search Page, www.tiac.net/users/smiths/privacy/wbfind.htm. It shows where each of ten bugging companies are sneaking in their little spies. Makes for interesting (and creepy) reading.

On the matter of security, Don Marti gets the last word: “When a site tries to violate users' common-sense expectation of privacy, it should be the system

administrator's responsibility to protect the user unless the user requests otherwise. Web ad banners are a security hole."

—Doc Searls

Uncollapsing Open Source Distinctions: a Conversation with Craig Burton

When he was at Novell in the 1980s, Craig Burton changed the concept of networking by making Netware hardware-independent. Later, at The Burton Group, he drove a new understanding of networks—as services (file, print, directory, security, management) rather than "pipes & protocols".

Lately, he has been turning his attentions to open source. He consults for a number of open-source companies, plus companies looking for open-source strategies. As always, his thinking is highly unique, independent and not part of anybody's orthodoxy. We caught up with him for a brief interview after he moderated a panel in April at *Linux Journal's* "Linux For Suits" event at Internet World. The interview moved forward by e-mail in May and June.

Doc: From what I gather, you like open source, but not the way we talk about it.

Craig: I recognize open source as a fact of business life that we need to get our heads around, but we won't make much progress until we start pulling apart a lot of distinctions that are currently collapsed. Until we do, "open source" will just be a buzz phrase instead of a fully rational strategy.

Doc: You mean for vendors?

Craig: I mean for every company that wants to operate in the world we're all trying to build here. If you want to survive in the long run, you need an open-source strategy. But you won't be able to implement one if you collapse a bunch of highly discrete distinctions.

Doc: Such as?

Craig: Accessibility and ownership. These are very different sets of distinctions, and they are entirely collapsed when most of us talk about them. The opposite of open is closed, not proprietary. The opposite of proprietary is public domain, not open.

Accessibility is an extremely important issue, and it is not just about source code. There are questions of source, binaries, licensing and derivative works. There are questions of protocols and APIs. There are questions of cost. It is important to make distinctions among the variables here. As an industry, we have collapsed the issues of accessibility and ownership.

Here is a matrix that separates those issues but keeps them in the same discussion by positioning them orthogonally. The accessibility axis runs from Closed to Open, while the ownership axis runs from Proprietary to Public Domain.

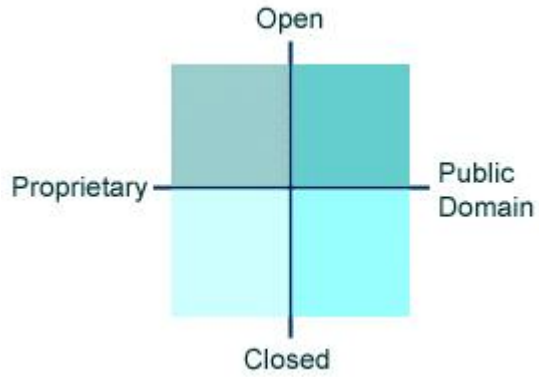


Figure 1. Burtonian Matrix

This is the matrix we need to understand, given any technology. Microsoft has reluctantly participated in the open-source corner of that matrix, and over time has moved a lot of technology into the open/public domain quadrant. And a lot of stuff they haven't. But the only significant difference between the Open Source community's products and Microsoft's is the amount of core technology that is made available in the open source half of that matrix.

Doc: So you don't believe that because most of Microsoft's familiar software is closed source, that it sucks?

Craig: First, some of it sucks and some of it doesn't. In spite of the open source movement, customers are buying it and using it. Further, there are other companies that have leveraged the Microsoft model to make money. On the other hand, there is the constant threat to customers that Microsoft will try to lock them in, and there is the constant threat to Independent Software Vendors (ISVs) that Microsoft will encroach on their businesses. Despite these threats, however, the industry continues to grow and thrive. As a model for making money—for lots of companies—we have seen nothing comparable yet from the Open Source world.

Doc: But we have seen a number of money-making models outlined by Eric Raymond, first in “The Magic Cauldron” and then in the book version of The Cathedral and the Bazaar.

Craig: Yet by Eric Raymond's own description (in “Homesteading the Noosphere”), an anti-commercial bias is part of the “hacker ideology” out of which the Open Source movement has grown. That's one reason that the distinction he makes between a cathedral and a bazaar is more accurately one

between a cathedral and a cult. A bazaar is, if nothing else, a marketplace. And marketplaces are by definition hospitable to business. So the culture of Open Source has a strong steak of inhospitality to business. A lot of this comes from the original Free Software movement and persists as a legacy in the GPL license, which is in many ways the least open and business-friendly of all the open-source licenses certified by Eric's organization, the Open Source Initiative. The main reason I point this out is that this anti-commercial bias accounts for the collapsing of distinctions here, because this bias introduces a third axis—a moral one that runs from bad to good. You can see it running from the lower left to the upper right. Proprietary and Closed are Bad. Open and Public Domain are Good. This is fine if all you want to do is hack code. But if you want to do business, you've got to face rational choices that are all over this matrix, whether you're a supplier or a customer.

Doc: Did I hear you say that the GPL is really a moral statement?

Craig: Moral and political. To me, the GPL is as much a political statement as a licensing agreement. There's a bunch of stuff in there that's superfluous to licensing concerns. This mixing political with licensing views is what causes businesses to be suspicious of open source.

Doc: But historically, the Open Source movement has tried to move away from the Free Software movement's anti-commercial rhetoric and policies.

Craig: Absolutely. The Open Source movement has a tall order, in attempting to commercialize the Free Software movement. We owe a lot to Eric Raymond for attempting to make free software palatable to business—and succeeding. Now it's clear that Open Source is a movement that will eventually include everybody. But that doesn't mean all the code in the world will be open source. Or public domain.

Doc: The important thing, then, is that both suppliers and customers need to factor open source into their strategies.

Craig: Exactly. Every business needs an open-source strategy. But the choices should not—and will not—be made on moral grounds. There are many combinations of choices that are all over this map. And you see them when you look at some of the relatively proprietary and/or closed products, or product components, being sold by card-carrying members of the Open Source community.

Doc: Such as all three desktop office suites for Linux. Applix, Sun and Corel all make closed-source suites.

Craig: Exactly. And they're all clones of Microsoft Office to a significant degree. But let's stay with the core issue here. Software varies greatly in accessibility, and open source isn't the only variable. In many cases, you can get at the APIs but not the source code. That makes the software somewhat more open than if the APIs and the source were closed. Beyond the API, there is the case—as in the UDA for Microsoft—where customers and competitors can add value to the Microsoft model without having source.

Doc: Do you think Microsoft has any understanding of this matrix, or uses it strategically?

Craig: In fact, Microsoft has moved quite a few things to the upper-right quadrant of the matrix. The Universal Data Access model provides a framework for accessibility to core Windows constructs. UDA is not open-source-based, but it provides programmatic interoperability with the Windows architecture. While I think Microsoft could do a lot more—including providing an open-source-based strategy to make Windows accessible—it has made large strides to open up Windows, and we shouldn't ignore that fact for the simple reason that developers and customers have taken advantage of it. Many, many companies have figured out how to make money off of Microsoft's accessibility model.

Doc: But can Microsoft have a truly strategic open-source strategy if it is hostile to open-source software and the Open Source movement?

Craig: No, it can't. Microsoft's combativeness, its hostility to everything it perceives as a threat, works against it here. But the same thing goes for the combative folks in the Open Source community. Both sides fail to understand the reasons for the other's successes. We have to get past that, and a good way to start is by getting clear about what some of these distinctions really mean. Without that clarity, we see gray as black or white.

Doc: How, for example?

Craig: Many Linux distributions are chock full of stuff that's both closed and proprietary. We just excuse it because it's bundled with something that is called “open”. We see gray as white. Good guys wear white hats and sell white code. But that's not always the case.

Doc: Give me an example of closed source code that passes for open.

Craig: Security components that use PKI—Public Key Infrastructure. We're talking about copyrighted, patented stuff. Proprietary and closed as can be.

Doc: You mean RSA?

Craig: Right. The distributions can't open source the RSA code. And it's all over the place.

Doc: Let's go to the big picture here. I watched you change the whole networking conversation back in the '80s, from one about wires and protocols to one about services. Are we at a similar cusp at this point in time?

Craig: Yes. For some reason, there's this idea that the Net is a finished thing. In fact, it's only beginning. One of its virtues is that it's still wide open. And it would help if more of us understood that openness—the ability of infrastructural software to interoperate without interference from anybody's agenda—is what enables growth. Over time, as Microsoft is discovering, the challenge is moving infrastructural code to the upper-right corner of this matrix. There is a big movement right now to copyright and patent everything you can name, but in the long run, that's wasted effort. It's more rocks in the stream of progress. History will flow around these obstacles. The real question companies need to ask is not “how do we patent and copyright and protect our stuff” but “how do we make money and do business while we gradually move toward that upper-right quadrant?” To do that, you need to see the Net as something that still desperately needs a lot of infrastructure.

Doc: Such as?

Craig: Directory and security services, which go hand in hand, are still very primitive. The Linux community is still using Sun's yellow pages (now called NIS) for directory services. Within Linux itself, it's old UNIX legacy stuff that hasn't changed in many years. Before we can have real privacy, and real secure communication, and real useful directory services—which we'll need to keep track of a world of abundant, changing and distributed objects—we're going to have to evolve past that legacy.

Doc: Novell wants us to use NDS. Microsoft wants us to use Active Directory.

Craig: And neither will be adopted by the Linux community, which is necessary for this whole thing to move forward. They're both proprietary and in the way. Rocks in the stream.

Doc: How do you conceive the Net? What's its conceptual architecture?

Craig: I see the Net as a world we might see as a bubble. A sphere. It's growing larger and larger, and yet inside, every point in that sphere is visible to every other one. That's the architecture of a sphere. Nothing stands between any two points. That's its virtue: it's empty in the middle. The distance between any two points is functionally zero, and not just because they can see each other, but

because nothing interferes with operation between any two points. There's a word I like for what's going on here: terraform. It's the verb for creating a world. That's what we're making here: a new world. Now the question is, what are we going to do to cause planetary existence? How can we terraform this new world in a way that works for the world and not just ourselves? Remember, this thing is in outer space. It's not connected to anything else. If you want to live here, you have to bring your own sustaining structures, foods and the rest of it.

Doc: Is this something like what Larry Lessig calls “end to end” architecture?

Craig: I haven't read much of Lessig's work yet, but I believe we're consistent here. But since we're talking about authors, I also like Carl Shapiro and Hal Varian's *Information Rules*, which makes pragmatic distinctions between the economy of building things and the economy of building information. The old economy was driven by economies of scale, including time, resources and materials. The Open Source community claims this is outdated, and they're right. But now what? These guys say that the economics of information invite respect for network effects, which can be increased by leveraging the value of intellectual property, rather than by protecting that property. A key to understanding the economies of networks is seeing that networks are comprised of relationships. That favors companies that build and maintain relationships through one of your favorite subjects: conversations.

Doc: Who are going to be the leaders in your view of the Web?

Craig: I have six theses—as opposed to ninety-five—that I use as the basis for determining the leadership of the next-generation web companies:

- No single vendor can own the web infrastructure. Therefore, inclusion is imperative.
- Distinguish accessibility from ownership.
- Standards are independent from these distinctions.
- The market drives accessibility requirements, not by credo, license agreements or moral positions.
- An open-source-based model requires all core technology to become more accessible over time.
- Those who—pragmatically and not moralistically—know when to put what in the upper half of the matrix will define leadership of the next generation of the web. --Doc Searls

SMILE, YOU'RE ON A 1-PIXEL CAMERA!

Web pages use a publishing metaphor. They are pages, after all. We write, open, read and bookmark them. That's one reason why we assume that when a

page downloads from a server, it's a one-way deal. The HTML describes the page, lays out the print, loads the graphics onto the page and into the cache.

There is at least the presumption of privacy. After all, reading is a personal (even an intimate) act. At times when interaction is required, such as when we fill out a form, there's a “submit” button that sends information back to the other end of the line. We're still in control.

And okay, we know about cookies and what they do. If we especially vigilant, we either refuse to accept them or go through the whole pile and weed out the suspicious ones.

But watch out. Big Brander is watching you. And not just with cookies.

It turns out that some companies are spying on you and your Web travels by using invisible 1x1 pixel transparent GIFs. These are in-line images downloaded from elsewhere, so the server-browser dialog can initiate covert reconnaissance on you and your subsequent surfage. You see no ad and suspect nothing. The cookie alert doesn't go off. But the bug—as in bugging device-has been planted.

Who's doing the planting? Usually a company that wants to learn something about you. Most of the time it's an advertising service that wants to “target” you with banners, e-mail spam or whatever. But they can serve all kinds of purposes, known and unknown. Don Marti found one on a Fed Ex page. (Presumably they want to track customers the way customers want to track a package.) Richard Smith, the leading source of information on Web Bugs, found two on Quicken's home page, both to provide “hit” information to advertising companies.

According to Smith's Web Bug FAQ www.tiac.net/users/smiths/privacy/wbfaq.htm, here is the information a Web Bug sends back to its server:

1. The IP address of the computer that fetched the Web Bug
2. The URL of the page that the Web Bug is located on
3. The URL of the Web Bug image
4. The time the Web Bug was viewed
5. The type of browser that fetched the Web Bug image
6. A previously set cookie value

Of course any graphic can serve the same purpose. That's why the only way you can see a Web bug is to view a page's HTML source. But Smith also provides another way to at least discover what sites are playing slight-of-pixel games: The Web Bug Search Page www.tiac.net/users/smiths/privacy/

wbfind.htm. It shows where each of ten bugging companies are sneaking in their little spies. Makes for interesting (and creepy) reading.

On the matter of security, Don Marti gets the last word: “When a site tries to violate users' common-sense expectation of privacy, it should be the system administrator's responsibility to protect the user unless the user requests otherwise. Web ad banners are a security hole.”

—Doc Searls

LJ INDEX—August 2000

1. Percentage of public relations professionals who admit to lying on the job: **25**
2. Percentage of public relations professionals who say they are not always able to confirm the validity of information conveyed to reporters on behalf of clients: **62**
3. Percentage of web sites with personal server headers: **5.7**
4. Number of Linux servers that self-disclose their distribution brand: **0**
5. Number of Linux servers found by Netcraft to disclose their distribution through Apache's personal server header: **850,000**
6. Percentage of those servers that identify their distribution as Red Hat: **72**
7. Percentage of those servers that identify their distribution as SuSE: **10**
8. Percentage of those servers that identify their distribution as Debian: **9**
9. Number of other distributions with more than 3%: **0**
10. Market cap of the entire Linux category in July 1999: **$0**
11. Market cap of Red Hat at its peak in November 1999: **$22.5 billion**
12. Red Hat revenues in the quarter ending November 1999: **$5.7 million**
13. Market cap of Red Hat on June 1, 2000: **$2.8 billion**
14. Red Hat revenues in the quarter ending February 2000: **$13.1 million**
15. Market cap of thirteen “Tier 1” Linux companies on June 1, 2000: **$8 billion**
16. Percentage of girls at popular teen web site who say they have had sex by the age of fifteen: **35**
17. Percentage of girls at popular teen web site who say virginity is in: **34**
18. Percentage of visits to popular teen web site that use Linux: **.22**

Sources

- 1-2: PRWeek, according to Reuters
- 3-9: Netcraft, http://www.netcraft.com/
- 10-15: Linux Weekly News and public financial information sources

- 16-18: http://www.smartgirl.com/

THE BUZZ

What were people talking about in May and early June? Below is a sampling of some of the hotter news stories over the past few weeks, as reported in "The Rookery", *Linux Journal's* on-line source for news, notes, quotes and reports from the field (updated daily and found at our web site, http://www.linuxjournal.com/):

- Python moves to BeOpen.com, and Guido van Rossum is appointed director of PythonLabs.
- Gateway and AOL become Transmeta's first big-time customers. Look for a variety of Internet devices running on the Crusoe chip later this year.
- Google adopting Linux as its operating system of choice.
- Microsoft being given the OK to tie up the court system in years upon years of appeals, trying to avoid the break up.
- Loki Entertainment Software releases Descent 3.
- The European Union looks to be more strict with software patents.
- Kevin Mitnick being courted as security advisor.
- Poor Corel!
- Debian's General Resolution, an amendment to the Debian Social Contract, which hopes to minimize the impact of non-free software on Debian GNU/Linux.
- Linux being pre-installed on the IBM ThinkPad.

STOP THE PRESSES: InterBase Redux

InterBase 6 is out, and the cross-platform relational database is now officially in the Open Source world. The source code has been released and so has a new company by the same name, created for the purpose of serving the businesses that put InterBase code to use.

The president of the new company is Ann Harrison, who co-founded the original InterBase with Jim Starkey, the product's original creator. Starkey also has a position with the new company as "technology software architectural advisor". This allows him to work inside the Open Source development community that began growing around InterBase even before the code was released.

Back in January, after Inprise (a.k.a. Borland) promised to open source InterBase, *Linux Journal* interviewed Inprise CEO Dale Fuller on the subject. That interview drew a record response from readers, most of whom expressed

enthusiasm for the move. Since then, interest in InterBase has grown, while both Inprise and InterBase have gone through a number of gyrations.

The intent to spin out a new company was announced in February, around the time Inprise announced plans to merge with Corel. Interest in InterBase grew, while the nascent company labored to craft both a business plan and a licensing strategy. The merger was called off in May, by which time free InterBase binaries were being downloaded at the rate of 3,000 per week (that's more than 50,000 as of late June).

By July, however, the new company was moving forward. “Basically, we're a start-up,” Ann Harrison says. The company is 19% owned by Inprise/Borland's venture fund and expects to grow quickly, along with the adoption curve of Linux in enterprise, which is moving rapidly. “Linux is a phenomenon much like the Internet five years ago, and it's bringing up a lot of the same kind of easy choices,” Harrison says. “People will say, `Why should I bother fighting with the file system and trying to write my own index code? Here's this free database. It just works. Why don't I pick that up and use it?”' Suddenly, it becomes the default way of storing data.

The business model is to sell add-ons and services around the product. These include training, support, documentation and third-party packages such as an ODBC driver, a Java driver and a product for connecting with development tools. The company does not plan to do any development of its own, but instead serve and back InterBase development and deployment, by both the development community and enterprise customers.

Kylix will help. Kylix is the code name for Inprise/Borland's family of visual development tools for Linux, due for release this summer. “We have a lot of tools that work well with Kylix, so when it's released, that will give us another boost,” Harrison says.

The company's goal is not modest. “Some day there will be a popular spreadsheet for Linux desktops, with lots of users in the enterprise. Where will that data live? On InterBase, we expect. Our goal is not to own an organization, but rather to own all the organization's applications, from the standpoint of the data store.”

While InterBase never scored big as a commercial product, it did develop a loyal following over the years starting in 1984, when the company was founded on an idea Jim Starkey had in the shower, and initial capitalization of $243.50. After early success with selling to the UNIX workstation market, the company was sold to Ashton-Tate, which was sold to Borland, which became Inprise (now Inprise/Borland, or vice versa—it's hard to keep the old name down). Restarting

the same company with two of the founders is an interesting second-chance story.

InterBase does face competition in the category. PostgreSQL has been open source for many years, and also has a loyal following. And the incumbent relational databases are certainly not going to go away easily.

But Linux is being adopted quite easily by countless enterprises. If InterBase can surf that wave, it should do pretty well.

—Doc Searls

Archive Index Issue Table of Contents

Advanced search

Do Manufacturers Have Any Responsibility?

Peter H. Salus

Issue #76, August 2000

This spring's outbreak of viruses has caused me to think about the burden that should be borne by a manufacturer. There is both a moral and a legal accountability any manufacturer has to its customers.

I'll readily admit that I don't use any Microsoft products: in fact, none of my machines has a DOS or a Windows partition. And I think early 1997 was the last time I sat down at an x86 box running Windows. But this spring's outbreak of viruses has caused me to think about the burden that should be borne by a manufacturer.

Any manufacturer has both a moral and a legal accountability to its customers. The various manufacturers of automobiles, children's toys and tobacco products, for example, have learned painful lessons about accountability through product liability suits.

“ILOVEYOU” and “Killer Resume” exploit holes in Microsoft Outlook that any less-avaricious company would have tried to stop several software generations ago. (I might note that the first RFC concerning security is 602 (December 1973), and was written by Bob Metcalfe, the inventor of Ethernet.) In my view, Microsoft has willfully ignored customer security for at least 20 years.

As Gene Spafford (Purdue University) has pointed out, even though Microsoft (and the tobacco companies) sell products that customers appear to want to buy,

> does that make the tobacco companies less culpable for selling a product they know to be dangerous? Does it matter that the consumers shell out money willingly for the product? (Even those who have some idea of the danger believe they have no control or choice.)
>
> There is a fundamental question involved in the area of informed consent. If the consumers actually

> understood the technology and the risks posed by their choices, and if they actually were able to make an unconstrained choice, would they make the purchases? If not, there is a moral (and potentially, legal) obligation for the vendor to make wise decisions on their behalf.

Gerald Shifrin (on the IP mailing list) noted:

> As an ordinary non-attorney consumer of computer products, it seems reasonable to me to expect that my software should ask permission before sending email to everyone in my address book or performing a mass deletion or modification of my files. If vendors like Microsoft allow or assist unsolicited foreign email to perform these acts, they are, at least in my mind, guilty of gross negligence.

Well, yes.

In fact, I found it puzzling that after “Melissa” and then “ILOVEYOU” and now “Killer Resume”, the newspapers aren't noting mass filings of product liability suits against Microsoft. If the world's economy can be brought to its knees by very simple code delivered via e-mail to PCs running Outlook or Observer or Explorer, then parts of that economy should be holding the manufacturer responsible.

Kevin G. Barkes (on the IP mailing list) posted:

> Another real-world analogy: you're tooling down the Interstate in your Chevy and hit a bump in the road. The doors fall off and the engine explodes. You have the ambulance driver stop at the dealership on the way to the trauma center so you can chew out the service manager. He sneers at you condescendingly and points to a paragraph of six-point type buried in a totally unrelated portion of the owners' manual:
>
> The doors of your car will fall off and the engine will explode when you hit a bump while traveling on an Interstate highway. One of our engineers thought this feature would be neat and we have added it at no extra charge to you. If you disagree (you weenie), you can disable this feature by performing the following procedure. First, obtain three chickens, two brown recluse spiders, a length of nylon rope and a virgin ...

Barkes points out that “Melissa” and “ILOVEYOU” were “badly-written programs created by rank amateurs”. What would happen if a really malicious first-rate programmer wanted to target Microsoft Outlook or Outlook Express?

Another list member said this about Spafford's posting:

> To further agree with Gene's point about tobacco and “what consumers want”, let me suggest that at any one point the market offers only a tiny subset of what is possible to create for consumers. Mere selection cannot create possibilities that are not developed or invented. Monopolies distort the creation of selections —in particular in systems' properties like security.

> Because of its installed base dominance, Microsoft's primary drive for innovation comes from a need to motivate an orderly “upgrade” revenue stream, while at the same time blocking competitors from entering the market to take that revenue away. That means innovations will be small, incremental, and extremely easy for customers to adopt.

By the time you read this, all questions on the DoJ's case against Microsoft will have been answered, pending a decade of appeals. But even a partly knowledgeable reaction on the part of customers may well put Microsoft into the product liability dock, and send Outlook and its kin the way of the Chevy Corvair.

> *Peter H. Salus, the author of* A Quarter Century of UNIX and *Casting the Net*, is an *LJ* contributing editor. He can be reached at peter@usenix.org.

email: peter@usenix.org

Archive Index Issue Table of Contents

Advanced search

The Shrinking Subject

Doc Searls

Issue #76, August 2000

Microsoft is a conversational black hole. Drop the subject into the middle of a room and it sucks everybody into a useless place from which no light can escape. --Don Norman

Monthly columns about volatile industries are always risky, because you write them a month or two before they come out. This means every topic is a gamble that the world two months hence won't be much different than right now. This time, I'm betting on a real long shot: that when you read this, Microsoft will still be one company—and will eventually prevail against the government's antitrust case. I'm also betting on a much more sure shot: that the trial's outcome won't matter much to the world we're building on open-source infrastructure.

Microsoft is a risky subject in any case, but it's especially risky when you have anything nice to say about them and your audience is a bunch of Linux folks. It's like going to Miami to talk up Fidel. So I'll try to be as rational and brief as I can, at least about the nice stuff.

I'll start by positing a thesis: that Microsoft's success has many good reasons and one bad one. All the good reasons reduce to orthodox marketing, which is not about “delivering messages” and “capturing eyeballs”, but rather about finding what customers want and giving it to them, even if it's talking paper clips. The bad reason is behavioral. Microsoft is often brutal, to its partners as well as its enemies.

The first reason explains the success of Microsoft's two cornerstone product families, Windows and Office. Both are what you get when a large, aggressive desktop software company does what orthodox marketing demands (in the words of Theodore Levitt): “satisfy the customer, no matter what.” For decades, Microsoft took that advice to a whole new level, while its opponents ignored it.

In 1996, when I spoke at a small retreat of computer industry types, my topic was “Markets are Conversations” (which later became the first thesis of *The Cluetrain Manifesto*). In that talk, I said Microsoft succeeded by being the only leading PC software company that made sure customer feedback informed—and even drove—engineering. Afterwards, one of Microsoft's top people took me aside and said, “That's the first time I've heard us credited with Marketing 101.” In fact, he said, “we do it to a fault.” That's why “Excel is full of stuff that exactly one customer asked for.”

Later, an Apple executive told me, “we hate our users. That's why we try to scrape them off on resellers.” Then he told me the story of a frustrated tech support group that chartered an airplane from Austin to San Jose, drove to Cupertino and presented the CEO with a long list of product faults that users complained about and engineering never fixed.

Bear in mind that both Microsoft's and Apple's true customers are intermediaries (resellers and OEMs), not users. That means listening to those users is highly optional. Microsoft deserves credit for at least being curious, whether or not their tech support sucked. (I should add that I never got poor tech support out of either Microsoft or Apple.)

Later, when I consulted a Seattle company that had a close working relationship with Microsoft, they agreed that Microsoft did a good job of closing the loop between tech support and engineering. But they added this: “Since so many tech support calls come from dumb users, you get dumb features.” Exhibit A: the talking paper clip. This also squares with Steve Jobs' four-word indictment of Microsoft: “they have no taste.” (Which says as much about Steve as it does about Bill.)

So how could a company that listens so closely to users fail to understand the antitrust rules by which increasingly dominant companies are supposed to compete?

Because Microsoft isn't just a competitive company. It's a combative one. There's a huge difference. Competitive companies conceive of business in terms of sports. Their conceptual metaphors are “business is sports” and “markets are playing fields”. Combative companies conceive of business in less-civilized terms. Their conceptual metaphors are “business is war” and “markets are battlefields”.

By these metaphors, “competitiveness” and “combativeness” mean very different things, even though both use nearly identical vocabularies. Both will “attack”, “defend”, “flank”, “command”, “dominate”, “control”, “retrench” and so

on. But the concepts are very different. That difference is both subtle and absolute:

- In sports, fairness is all. In war, all is fair.
- Sports is full of rules. War has no rules. (Ask the people of Sarajevo and Hiroshima.)
- Playing fields are always level. Battlefields are rarely level.
- Sports is civilized. War is barbaric.
- Sports end with victory or defeat—but both sides live to play again. Wars end with triumph and dominion or death or surrender.

Business is not regulated by rules of war. The “competitiveness” about which business law speaks does not comprehend the right of one company to “cut off the air supply” of another—unless, of course, it's “fair”. Microsoft tried mightily in The Trial to convince Judge Thomas Penfield Jackson that the computer and Internet worlds were battlefields, and that every competitor's very existence was under constant threat. Microsoft failed because Judge Jackson and the entire regulatory system conceive of business as sports, not war. There was no way Jackson could buy Microsoft's argument. There are rules for competition, especially between monopolies and less-advantaged companies. Microsoft broke them. And Judge Jackson's response was to break Microsoft. Literally.

To a combative company like Microsoft, a breakup is a death penalty and utterly unacceptable. But Judge Jackson doesn't have the final power to execute Microsoft, which belongs to the appeals court or the Supreme Court. He does, however, have the power to save Microsoft, if he makes enough mistakes.

It is significant that throughout this whole ordeal, Microsoft has refused to settle, or even to admit the obvious fact that they play hardball. This was especially obvious during one of the trial's most ludicrous moments: when Robert Muglia, a Microsoft executive, insisted that when Bill Gates wrote that he was “hard-core about NOT supporting” a hunk of Java technology, the words meant something other than what they said. Jackson bought none of it, and finally yelled “No!” and “Stop!” at the witness. “By asking the judge to swallow such absurdities as Muglia's testimony,” wrote Joe Nocera in *Fortune*, “Microsoft instead gave Jackson an excuse to swallow nothing.”

This looked like Muhammad Ali's rope-a-dope strategy to me. Stay on the defensive, dodge constantly, and look for the other guy to make mistakes. The prosecution had more than the burden of proof here; they had the burden of their own rules. If they broke enough of them, Microsoft reckoned, the final ruling would go to the defendant. Indeed, Microsoft attorneys have said for some time that they expected to win on appeal.

But this isn't just a story about Microsoft. It's also about Netscape. The judgment against Microsoft was based mostly on that company's behavior toward Netscape, whose air supply Microsoft allegedly wished to cut off. Nobody, however, including Microsoft, gives due credit to Netscape for choking itself. Or to the press, for covering “The Browser War” as anything but a sports event.

True, Microsoft really was out to hurt Netscape. I remember Jim Clark, onstage at the first Netscape conference in 1996, responding to a question about “polarizing” remarks made by Netscape people about Microsoft. “They're out to kill us!” Clark yelled back. “That has a polarizing effect!” Quite true.

But Netscape was a worthy opponent, blessed with enormous advantages. It made the first popular browser, which quickly achieved nearly unanimous usage. Browsing was understood on Netscape's terms. The whole company took sides with the Net, which offered enormous strategic advantages in an increasingly Net-based business world. And Netscape used some of those advantages very cleverly.

Unfortunately, they got caught up in the war story.

By the time Microsoft noticed that the Net mattered, the press had already cast the company in the Evil Empire role it had done much to earn, thanks to its merciless (or, in Gates' own words, “hard core”) marketing.

Here's how it looked to me in early '96, when I wrote about it for a web 'zine:

> I learn from the papers that the desktop world has fallen under the iron grip of the most wealthy and powerful warlord in the galaxy. With boundless greed for money and control, Bill Gates of Microsoft now seeks to extend his evil empire across all of cyberspace.

The galaxy's only hope is a small but popular rebel force called Netscape. Led by a young pilot (Marc Andreessen as Luke Skywalker), a noble elder (Jim Clark as Obi-wan Kanobe) and a cocky veteran (Jim Barksdale as Han Solo), Netscape's mission is joined by the crafty and resourceful Java People from Sun.

Heavy with portent, the headlines tromp across the pages (cue the Death Star music—dum dum dum, dum da dum, dum da dummm)...

- “MICROSOFT TAKES WAR TO THE NET: Software giant plots defensive course based on openness”

- “MICROSOFT UNVEILS INTERNET STRATEGY: Stage set for battle with Netscape”
- “MICROSOFT, SUN FACE OFF IN INTERNET RING”
- “MICROSOFT STORMS THE WEB”

The mind's eye conjures a vision of The Emperor, deep in the half-built Death Star of Microsoft's new Internet Strategy, looking across space at the Rebel fleet, his face twisted with contempt. “Your puny forces cannot win against this fully operational battle station!” he growls.

But the rebels are confident. “In a fight between a bear and an alligator, what determines the victor is the terrain,” Marc Andreessen says. “What Microsoft just did was move into our terrain.”

And Microsoft knows its strengths. December 7th, *The Wall Street Journal* writes, Bill Gates “issued a thinly veiled warning to Netscape and other upstarts that included a reference to the Pearl Harbor attack on the same date in 1941.”

Exciting stuff. But is there really a war going on? Should there be?

No, but both Microsoft and Netscape operated on exactly that metaphor. And at first, Netscape used it brilliantly.

One extremely clever move was buying, opening and ubiquitizing LDAP, the Lightweight Directory Access Protocol. In an interview for *Linux Journal* in 1996, Craig Burton said,

> Microsoft built its entire services strategy on what it thought was a titanium vise. One side was an object-oriented file system called OSS which collapsed the directory into the file system. This was Cairo (now Windows 2000). The other side was a distributed application development framework called OLE (now ActiveX), which they owned lock, stock and barrel. They would squeeze those together and the Netscapes of the world would squoosh like jello. But the Internet blew the jaws of that vise apart. When Microsoft tightened the jaws of that vice, they bent wide open. The world has shifted, and Microsoft is not going to dominate it.

LDAP was a strategic move that helped shift the world.

For a year or two, Netscape looked like it could do no wrong. It was a Miata being chased down a mountain road by a tractor trailer. As long as it moved fast and looked ahead, there was no problem with the truck behind. But at

some point, Netscape got fixated on the rear-view mirror. That's where they were looking when they drove off the cliff.

Why did they do that?

1. They forgot where they came from: the hacker community that had for years been developing the Net as a free and open place—one hospitable to business, but not constrained by anybody's business agenda. The browser was born free, like Apache, Sendmail and other developments that framed the Net's infrastructure. The decision to charge for the browser—especially while still offering it for free—put Netscape in a terminal business from the start.
2. They got caught up in transient market's fashions, which were all about leveraging pre-Web business models into an environment that wouldn't support them. Mostly, they changed the browser from a tool of Demand (browsing) to an instrument of Supply. They added channels during the “push” craze. They portalized their web site. They turned the location bar into a search term window for a separate domain directory, to be populated by the identities of companies that paid to be put there (a major insult to the user's intentions). Worst of all, they bloated the browser from a compact, single-purpose tool to an immense contraption that eventually included authoring software, a newsgroup reader, a conferencing system and an e-mail client—all of which were done better by stand-alone applications.
3. They became arrogant and presumptuous about their advantages. At one point, Marc Andreessen said an OS was “just a device driver”.
4. Their engineering went to hell. By the time Netscape was sold (at top dollar) to AOL, the dirty secret was that its browser code was a big kluge and had been for a long time. Jamie Zawinski (one of the company's first and best-known engineers) put it bluntly: “Netscape was shipping garbage, and shipping it late.” Not exactly competitive.
5. They lost touch with their first and best market: those customers who had actually paid for that damn browser.

It was painful to watch. Netscape was family for me (a close relative was highly placed in the company). I was such a partisan that I paid for that lousy browser the whole time they charged for it, and I never got useful tech support. One time, a tech support person told me the browser “didn't matter” because the big money was all in server software.

Making browsers free (in cost, if not in code) was a brilliant and insightful move by Microsoft that was utterly lost on Netscape, which was unwilling to play smart chess by making a reciprocal sacrifice. It was a suicidal mistake. The company we loved and rooted for is now history.

Its ghost may still have revenge on Microsoft. It was Netscape, more than any other company, that pushed the antitrust case against Microsoft, and then served as the exemplary victim of Microsoft's rapacious behavior. If the Feds bust up Microsoft, Netscape deserves more credit than any other company.

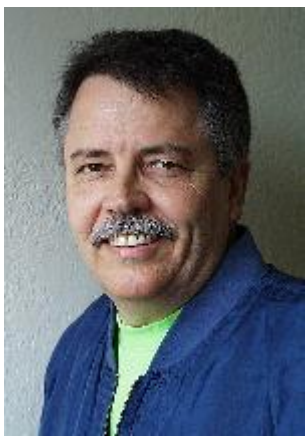
But let's have some perspective here. What happened between Microsoft and Netscape may make interesting stories, whether told in terms of sports or war. But the story that matters can't be told in those terms. It needs a better metaphor.

My favorite, of course, is conversation. As a topic of conversation, Microsoft has been shrinking for the last five years, regardless of what happened to Netscape or what revenge the Feds carry out. There are too many other, more interesting subjects to talk about.

Craig Burton, Brian Behlendorf and Tim O'Reilly all prefer the environmental metaphor. That's what I've learned from conversations with each of them. The wide-open world each of them envisions is a habitat where development tools, methods and strategies matter more than the companies that employ them.

Programmers like metaphors about buildings. They “architect”, “design” and “build” with “tools”. And it is individual programmers, more than any corporate supplier, who are designing, architecting and building this new world, using free software tools, Linux building materials and open-source building methods.

If Microsoft can help, whether as one company or as two, that's great. If they can't, it's probably because they're too busy looking for enemies. And giving us less and less to talk about.



Doc Searls (info@linuxjournal.com) is Senior Editor of *Linux Journal* and co-author of *The Cluetrain Manifesto*.

Archive Index Issue Table of Contents

Advanced search

Best of Technical Support

Various

Issue #76, August 2000

Our experts answer your technical questions.

Two Monitors?

Is it possible in Linux (I don't care which distribution, I mean the system architecture) to have two screens, i.e., two monitors on the same machine? Also with two graphics cards, of course. Is it possible to have them running together, either in text mode or serving X? And is it possible to have only one actual X desktop, but two virtual ones projected over two different screens? —Eduardo Garcia, egarcia@msl.es

The “two screens” support you've mentioned is called multi-headed display support. It is the ability to use two (three and so on) monitors simultaneously and usually make them act as one huge virtual desktop. The commercial X servers such as Accelerated X (http://www.xig.com/) and Metro-X (http://www.metrolink.com/) already have such support, and recently XFree supports this too (but it is not stable enough, in my opinion). Be aware, though, that your hardware (graphics card) must also support this. —Mario de Mello Bittencourt Neto, mneto@argo.com.br

With XFree 4.0, you can run X on two cards and two monitors, either as independent sessions or as one big screen (Xinerama). It should also be possible to get XFree 4.0 to open one X session on the secondary screen (i.e., not the one Linux is displaying console messages on) and keep text mode in the other monitor, but there isn't a convenient way to switch back and forth between the two screens (you'd have to use **CTRL+ALT+FN**). Frame buffer (FB) support will also work with two video cards, so my guess is that the FB application gets to say which FB display to write on. —Marc Merlin, marc_bts@valinux.com

As installation of XFree86 4.0 is not for the faint-hearted, you may want to hold off until RPM packages start to appear before trying it. —Erik Ratcliffe, erik@calderasystems.com

It depends on your X server software. XFree86 version 4 (which is more recent than the version included in your Red Hat distribution) has some support for multi-headed configurations; you can check their release notes under http://www.xfree86.org/. —Scott Maxwell, maxwell@ScottMaxwell.org

fopen

What is the scope of **fopen**? That is, when you use this system call, where exactly does it look for the file we want to open?

—Kumhaar, kumhaar@yahoo.com

If you supply an absolute file name (that is, one that starts with a "/"), then fopen starts at the root directory. Otherwise, the file name is relative, so fopen starts looking for the file in the process's current working directory. This directory is initially the same as the parent process's current working directory (so if you ran the program from a shell, it's whatever directory you were in when you ran it), but the current working directory can be changed by calling **chdir** or **fchdir**. By the way, as a pedantic note, fopen is not a system call; it's a C library function. **fopen** does part of its work in terms of the system **call open**. —Scott Maxwell, maxwell@ScottMaxwell.org

The fopen library function is the analog of the low-level open system call. You use it mainly for files and terminal input/output. When you need explicit control over devices, you are better off with the low-level system calls, as they eliminate potentially undesirable side effects from libraries, like input/output buffering.

If successful, fopen returns a non-null FILE * pointer. If it fails, it returns the value NULL, defined in stdio.h.

fopen uses the open system call. Here is how the open system call works:

1. When the kernel receives an open system call, it starts the function called sys_open. You can find the code in the kernel source in fs/open.c. **sys_open (const char * filename, int flags, int mode)**

2. From the file name, sys_open will try to get the associated inode structure. This inode structure is located in the directory where the file is (the directory is a special file). To get the inode of the directory with the relevant information, sys_open will have to recurse by starting to read the current directory to get the inode of the relative directory, and so on.

If the file name starts with a /, the process is the same, except that it will start with inode 2 (inode number of /) on the root partition.

3. Once the inode of the file is found, sys_open will read the file operation associated with the file's inode, and run the open method associated with that inode/file.

This open method may be related to a device module if the file is a device (see major number and /proc/devices) or to a specific file system (**df -k *filename* -> proc swap ext2** ...)

4. This open method returns a “struct file *” which is associated with a file descriptor.

sys_open will return a file descriptor, an integer greater than 0. If sys_open fails, it returns an integer less than 0. fopen will then associate this file descriptor (int) to a file stream (FILE *).

I hope this helps. —Emmanuel-Robert Mayssat, emayssat@valinux.com

fopen(2) takes two parameters. The first is the file to open, and the second tells the system to open it for reading, writing, reading and writing, and other options. If you do not specify a path to the file to open, fopen() will look for that file in the current directory. —Chad Robinson, Chad.Robinson@brt.com

Feel free to refer to the manual pages: use Chapter 2 for the system calls (e.g., **man 2 open**) and Chapter 3 for library functions (**man 3 fopen**). —Alessandro Rubini, alessandro.rubini@linux.it

Signal-7

I have tried unsuccessfully to install Linux 6.1. Here's what happens:

I put in the CD and the boot disk. The Linux program decompresses all the “running/sbin/loader”. Then it waits a few minutes and says:

```
"install exited abnormally -- received signal 7"
sending termination signals.
```

I have asked everyone I know what “signal 7” means, but to no avail.

—Michael C. Fields, kindra@2fords.net

Signal 7 is “bus error”, as reported by /usr/include/asm/signal.h (included by /usr/include/signal.h, the first place to look). This is most likely a hardware problem on your box, similar to the “signal 11” (segmentation fault) problem.

Please refer to the sig11 FAQ at http://www.bitwizard.nl/. —Alessandro Rubini, alessandro.rubini@linux.it

You fail to mention which distribution you are trying to install, but luckily, my crystal ball is telling me that it would be Red Hat, or a derivative thereof. Signal 7 is SIGBUS, which means that there are some hardware/driver issues. I'd try a newer version of Red Hat or a different distribution in case a different kernel helps on your machine. —Marc Merlin, marc_bts@valinux.com

parport (Parallel Port)

I have been asked by my company to write some code to write to and read from a special printer. As an example, if I shove the string "^[I" out the parallel port, I should be able to read back in "TP96". However, I am as new as they come. I have been looking at all the parport documentation, but I don't understand. Is parport already part of the kernel? Do I #include parport in my own program? How do I put all the pieces together to get this one seemingly simple task accomplished?

—Michael J. Conroy, m_j_conroy@yahoo.com

Two web sites may help you. The first contains general parallel port programming information: www.lvr.com/parport.htm. The second contains information on I/O port programming under Linux: metalab.unc.edu/mdw/HOWTO/mini/IO-Port-Programming.html. —Chad Robinson, Chad.Robinson@brt.com

If you want to know more about parport, take a look at the file Documentation/parport.txt in the kernel sources hierarchy. For your personal use, you should use low-level inb/outb functions (try **man outb**). There is an excellent (of course :-)) article by Alessandro Rubini in *LJ* #47 about using parallel port. —Pierre Ficheux, pficheux@wanadoo.fr

Yes, it's a compile time option.

```
moremagic:~# cat /proc/parport/0/hardware
base:   0x378
irq:    none
dma:    none
modes:  SPP,ECP,ECPEPP,ECPPS2
moremagic:~# cat /proc/parport/0/autoprobe
MODEL:Unknown device;
MANUFACTURER:Unknown vendor;
```

I've never done parallel port programming, but you can write to /dev/lp0 (or lp1, lp2...) and you may be able to read back from it. One thing you can do for more information is look at the source code from lpd. —Marc Merlin, marc_bts@valinux.com

Athlon

Can Linux be installed on an Athlon platform? —Mark MacWilliam, mMacWilliam@infocell.com

Because the Athlon processor has the same instruction set (with enhancements) as the x86 processor series, Linux should run on it without problems. —Chad Robinson, Chad.Robinson@brt.com

Yes. The 2.2.14 kernel (or better) recognizes the Athlon, and 2.3.x (soon to be 2.4) has optimized support for the Athlon. —Marc Merlin, marc_bts@valinux.com

I am typing this reply using my brand-new 700MHz Athlon system (running Linux, of course!). So not only can Linux be installed on an Athlon, I can add that it runs like a dream. —Scott Maxwell, maxwell@ScottMaxwell.org

Trying to Install OpenLinux

I am having problems installing Caldera OpenLinux 2.3 on a PC Pentium III 450, with a SIS 620 video card and operating under Windows 98. I succeeded in making the partition on the disk for where to install OpenLinux, but when trying to install it, first it appears on the screen where the recognition of the hardware of the machine is made, and then the screen becomes frozen after showing vertical gray lines. I don't know what the problem could be. Would you help me?

—Susana Diaz, subediaz@satlink.com

The SiS 620 chipset is supported by the SVGA server, but for whatever reason, it will not run in 16-color VGA mode (which is what OpenLinux's graphic installer uses). What you'll need to do is use the character-mode installation, then run either lizardx, XF86Setup or xf86config after the installation is done to configure the X Window System. The disk images for the character-mode installation are found in the /col/launch/lisa/floppy directory. Use RAWRITE (in DOS/Windows) or **dd** (in UNIX/Linux) to transfer the install.144 image to a formatted 1.44MB diskette. Use the new diskette to launch the installation. —Erik Ratcliffe, erik@calderasystems.com

Many on-line help resources are available on the SSC web pages. Sunsite mirror sites, FAQs and HOWTOs can all be found at http://www.linuxjournal.com/.

Archive Index Issue Table of Contents

Advanced search

New Products

Ellen M. Dahl

Issue #76, August 2000

Linux Office Solutions, CodeWizard 3.1, eEMU and more.

Linux Office Solutions

VistaSource, a new, wholly owned subsidiary of Applix, Inc., has released three cornerstone products, Applixware Words, Spreadsheets and Office for Linux v5.0, in conjunction with the launch of the new company. The suite, Office, includes an extensive filtering capability to allow document exchange with Microsoft Office 2000, Corel WordPerfect and other file formats. Words for Linux and Spreadsheets for Linux are now available as two separate products.

Contact: VistaSource Inc., 114 Turnpike Rd., Westboro, MA 01581, 508-870-0300, 508-366-4873 (fax), info@vistasource.com, http://www.vistasource.com/.

CodeWizard 3.1

Originally developed for C++ developers, CodeWizard is a static analysis tool which automatically enforces industry-respected coding standards. The latest version has additional rules for C developers and enhanced customization capabilities for the RuleWizard module and new metric rules in its rule set.

Contact: ParaSoft, 2031 S. Myrtle Ave., Monrovia, CA 91016, 888-305-0041, 626-305-3036 (fax), info@parasoft.com, http://www.parasoft.com/.

eEMU

eEMU (enterprise Event Management Utility) is a simple yet powerful event-management framework. The commercial version features high-speed processing, low disk-space requirements, improved security and other useful enterprise environment features. It comes with an enhanced event browser

and monitoring agent kits for various platforms. The server software is available for Linux, Solaris and Windows NT. Technical support is included.

Contact: Jarrix Systems Pty Ltd, PO Box 8, West Ryde 1685, NSW, Australia, +61-2-9809-1261 (fax), info@jarrix.com.au, http://www.jarrix.com.au/.

RIA Server

Rackmount Integrated Applications (RIA) server is the world's smallest rackmount server and a development platform for high-density Internet and telephony applications. For appliance-based needs, RIA can be pre-loaded with Linux, Apache, SendMail and Crystal's exclusive Web Utility, which allows browser control of systems settings, application settings, management of users and virtual sites, as well as faceplate LCD programming. Up to 52 individual units can be placed in a standard 7-foot rack.

Contact: Crystal Group Inc., 850 Kacena Road, Hiawatha, IA 52233-1204, 800-378-1636, 319-393-2338 (fax), http://www.crystalpc.com/.

Exile III: Ruined World

The Linux part of Exile III: Ruined World, an epic fantasy role-playing game, features an involving, intricate plot, an enormous, detailed and changing world, and an elegant, carefully designed game system and interface, with comprehensive on-line help. Exile III for Linux requires glibc 2.1 or later.

Contact: Spiderweb Software, PO Box 85659, Seattle, WA 98145, 206-789-4438, 206-789-6016 (fax), spidweb@spiderwebsoftware.com, www.spiderwebsoftware.com/exile3/linuxexile3.html.

User Management in MandrakeSoft 7.1

MandrakeSoft now bundles Enlighten's cross-platform User Management Tool with the MandrakeSoft 7.1 distribution, adding sophisticated system management capability to MandrakeSoft Linux, allowing customers to manage all their users simultaneously from a single graphical console. Enlighten User Management features account creation, configuration, and password management of network user accounts and group membership.

Contact: MandrakeSoft, Inc., 2400 N. Lincoln Ave., Altadena, CA 91001, http://www.linux-mandrake.com/.

Contact: Enlighten Software Solutions, Inc., 999 Baker Way, Fifth Floor, San Mateo, CA 94404, 650-578-0700, 650-578-0118 (fax), info@sftw.com, http://www.enlightendsm.com/.

HostML and ViewML

The HostML Application Server captures and translates large and complex UNIX and IBM application data to XML format, which is then re-rendered using ViewML on a range of systems, including hand-helds and palmtops running Linux, Microsoft Windows CE, Palm, Java and ActiveX. ViewML is based on an open-source project known as Microwindows, which provides a small graphical windowing system for embedded Linux and other platforms.

Contact: Century Software, 5284 South Commerce Dr., Suite C-134, Salt Lake City, UT 84107, 801-268-3088, 801-268-2772 (fax), sales@centurysoftware.com, http://www.centurysoftware.com/.

Flipper Graph Control 2.0

Flipper Graph Control version 2.0, a significant upgrade to ProWorks' ActiveX charting control, features enhanced compatibility with the Web, increased flexibility for scientific and financial charts, an improved look and feel and greater customizing capability. Flipper Graph Control can now download saved data over the Internet and integrate it into ASP or HTML web pages, enabling management and display of off-site data. New graphing functions were added for polynomial curve fitting, moving averages and setting axis data aspect ratios.

Contact: ProWorks LLC, 2371 NW Maser Dr., Corvallis, OR 97330, 541-752-9885, 541-752-9886 (fax), http://www.proworks.com/.

RTAI v1.3

RTAI version 1.3 provides enhanced capabilities and services, which simultaneously extend the ability of RTAI to meet the needs of complex real-time applications and make it easier for developers to write and debug them. Available under the LGPL, RTAI now includes dynamic memory allocation, a /proc interface, an enhanced LXRT-Informed (LinuX RT) module and Perl bindings for soft real-time task development.

Contact: RTAI, mantegazza@aero.polimi.it, http://www.rtai.org/.

eServer.group

Scalable eServer.group features application-level clustering and load balancing for enterprise mail and web servers and provides network managers with distributed cluster controllers for load balancing between servers. eServer.group is available as stand-alone software, as a plug-in for third-party applications such as the Apache web server or database engines, and as a

complete hardware/software solution. Features include application-level failover/redundancy, distributed cluster controller and application-level monitoring and management.

Contact: Technauts Inc., 501 James Jackson Ave., Cary, NC 27513, 877-373-7837 (toll-free), 919-388-2060 (fax), info@technauts.com, http://www.technauts.com/.

VCOM on Linux

VCOM is a message-oriented middleware solution for enterprise-wide application integration projects undertaken in distributed and heterogeneous computing environments. Using VCOM, applications can exchange messages across platforms, providing application programmers with an industry-leading, easy-to-use, scalable and robust solution.

Contact: NetSys Software Group AB, N<\#228>mndemansgatan 3, S-431 85 Mölndal, Sweden, +46-31-720-60-00, +46-31-27-50-10 (fax), info@netsys.se, http://www.netsys.se/.

RM1U-AXe and RM2U-AXi-C

Rave Computer Association's RackMount 1U-AXe features an original Sun UltraAXe motherboard integrated into a 1U rackmount chassis. The RackMount 2U-AXi-C features an original Sun UltraAXi motherboard with upgradable UltraSPARC-IIi processor integrated into Rave's new 2U rackmount chassis. Storage capacity for the 2U is found in four front-exposed, removable 3.5-inch low-profile drive bays, plus one 3.5-inch and one 5.25-inch drive bay.

Contact: Rave Computer Association, Inc., 36960 Metro Court, Sterling Heights, MI 48312, 800-966-7283, 810-939-7431 (fax), info@rave.com, http://www.rave.com/.

TowerJ 3.5

TowerJ release 3.5 is the first high-performance Java 2 Java Virtual Machine for Intel-based Windows NT and Linux. The Java deployment solution provides an extra level of security for organizations deploying Java-based e-business applications. Tower offers a free 15-day evaluation with full technical support to qualified organizations.

Contact: Tower Technology Corporation, Centennial Towers, 505 E. Huntland Dr., Suite 530, Austin, TX 78752, 800-285-5124, 512-452-1721 (fax), tower@twr.com, http://www.towerj.com/.

X-Win32 v5.0

X-Win32 version 5.0 for Windows 95/98 and Windows NT/2000 computers offers a redesigned user interface and new features, including thin-client access to Linux/UNIX servers. The Multiple XDMCP Sessions feature allows users to connect to two or more Linux/UNIX systems simultaneously; run applications from those servers in different windows on their Windows desktop; and copy/paste data between the Windows and X applications. Windows Terminal Server Support allows administrators to install X-Win32 on a server.

Contact: StarNet Communications Corporation, 1270 Oakmead Pkwy. #301, Sunnyvale, CA 94086, 408-739-0881x18, 408-739-0936 (fax), sales@starnet.com, www.starnet.com/download.

Archive Index Issue Table of Contents

Advanced search