HOME PBX GIVES EXTENSIONS AND VOICE MAIL TO THE WHOLE FAMILY

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community

JANUARY 2006

Coming to DVD
Your home videos with Kino and DVDStyler

State of the art video production

At home with Linux

HOME PROJECTS

LINUX PHONE HOME

The Skype voice-over-IP phone service works great with a headset. But can you extend Skype to use the telephones in your house?

GET YOUR WINDOWS GAME ON WITH LINUX

PLUS

Broadcast Your Music From Network-Attached Storage To Your Stereo

A (Relatively) Cheap Terabyte Home Backup System

The competition doesn't stand a chance.

If you base deployment decisions on performance and price, Coyote Point's for you. We've cornered that market.

To prove it we asked The Tolly Group to evaluate our E350si application traffic manager against the competition. The results speak for themselves.

Throughput? Almost 40% more than others in our space. Cost of transactions per second? Up to four times less. Connection rate? In some cases, one-sixth the cost. One-sixth! And we're told Coyote Point is the #1 choice for today's open source networks.

But don't just take our word for it. Get the facts. Call 1.877.367.2696 or write info@coyotepoint.com for your free copy of the full Tolly Report.

www.coyotepoint.com

LINUX JOURNAL

JANUARY 2006 ISSUE 141

Coming NEXT MONTH

Linux Journal gets a new Editor in Chief, Nicholas Petreley. Nick has been a user and promoter of Linux since the dawn of time, or at least since 1994 when he first started using and then writing about Linux in the computer trade journal *InfoWorld*. Red Hat founder Bob Young credits his fair coverage of Linux in *InfoWorld* as one of the catalysts that helped Linux take over the IT world, and we agree.

Nick started his career in computing as an assembly language programmer for a signal processing and artificial intelligence research and development firm. In his former life at *InfoWorld*, he did everything from writing columns and articles to writing benchmark programs, and went on to manage product testing in the *InfoWorld* test center. Since then, Nick launched the original *LinuxWorld* magazine, helped start up the LinuxWorld tradeshow, created and first hosted the LinuxWorld keynote quiz show for geeks and nerds, The Golden Penguin Bowl, and has produced several Evans Data reports on Linux developers.

Now we've got him. To be honest, it was an easy catch. Nicholas Petreley has been one of *Linux Journal*'s biggest fans since he was first introduced to the magazine, and he's been itching to get involved with *Linux Journal* for years. To what can we look forward as we make this transition? We'll let Nick tell it himself in next month's From the Editor in Chief.

COVER PHOTO BY BUTCH MARTIN

diff -u

What's New in Kernel Development

Ahmad Reza Cheraghi has begun work on some infrastructure to allow the **kernel .config file** to be created automatically based on the system hardware. This project provides more of a framework for continued development than an actual fully fledged solution. Ahmad's intention is that over time, the various experts will add hardware detection rules to his infrastructure, so that `make autoconfig` will become more and more accurate. His work so far has gotten a mixed reaction. Folks like **Hua Zhong** are thrilled that this sort of thing is being done, because it will bring kernel compilation one step closer to the average user; however, folks like **Roman Zippel** seem skeptical that this framework could ever be completed and may languish in a partly usable state for a long time.

David Teigland has been trying to get **GFS2**, the clustered filesystem, into the kernel for a while now, with much controversy. Part of the problem has just been the name. The GFS developers claim to have tacked on the 2 at the end for their own reference, not realizing this would turn it into a whole new project that would have to win acceptance into the kernel all over again. GFS is already in the kernel, and an update to GFS would have a much easier time getting in than a new project. But others feel there is more to it than this. GFS2 is not fully compatible with the original GFS—specifically the disk format has changed between the two versions, making it impossible to mount GFS directory trees under GFS2. There are other differences, and **Andrew Morton** has been very reluctant to accept the new code without seeing some kind of justification from the authors. This justification, apparently, has not been forthcoming.

The **FUSE** (Filesystem in Userspace) filesystem has finally made it into the main kernel, after a long period of uncertainty. **Linus Torvalds** has never liked the project, saying that it just wasn't possible to separate the guts of a filesystem from the kernel's internals cleanly. But **Miklos Szeredi**, the principal developer, did finally answer most objections and showed convincingly that any remaining ugliness actually could not be avoided. This was apparently the real holdup for a long time, but after living for a while in Andrew Morton's -mm tree without requiring any major fixes, and without any other developer being able to suggest alternatives to the remaining weird hackery, Andrew and Linus were both convinced. FUSE made its first appearance in the main kernel tree in 2.6.14-rc1, and can be expected in 2.6.14.

There has been some concern that Andrew Morton will become overloaded by all the kernel work he's doing. As Linus Torvalds has said, "I actually worry that Andrew will at some point be where I was a couple of years ago—overworked and stressed out by just tons and tons of patches." It's true that a huge number of patches funnel through Andrew each day, but he says the sheer volume is not the biggest problem. It's the patches that don't actually work that cause him to waste hours. Putting out a release of his -mm tree is very time consuming, because it must boot on four different architectures and compile on at least seven. Any failure means tracking down the bogus patch and removing it, and then trying the whole thing again. In practice, this pushes the new release to the next day. Meanwhile, new patches keep flying in. In spite of this, Andrew says he doesn't feel overloaded, and he even has time to monitor and respond to the kernel bugs database and chase down maintainers to answer questions about various code submissions.

Greg Kroah-Hartman has handed maintainership of the **I2C Subsystem** off to **Jean Delvare**. The two had been working quite closely on it for some months, presumably preparing for this change, and now Greg has made it official. He still will be involved in I2C development to a lesser degree, and he also will remain the conduit for submitting I2C patches to Andrew Morton and Linus Torvalds for inclusion in their trees.

—ZACK BROWN

Geekcorps Wants You

Geekcorps (www.geekcorps.org) promotes economic growth in the developing world by sending highly skilled technology volunteers to teach communities how to use innovative and affordable information and communication technologies to solve development problems.

Geekcorps is in search of the few, the proud, the Linux Geeks who not only know Linux networking, including extensive management tools, but who also can teach French-speaking Malians what they're doing so the Malians can replicate and maintain the systems they design.

If you think you can do that, in French, then check out the Geekcorps Mali Connectivity Project in greater detail: mali.geekcorps.org. Geekcorps is teaching Malian radio stations and community centers how to work with Linux-based systems, connect the systems with Wi-Fi and VSAT, and maintain both the systems and connections with great ingenuity and minimal expense.

Volunteers would stay in Mali about four months, working on multiple projects and with local counterparts in a hands-on setting. Geekcorps can offer great benefits besides that warm and fuzzy feeling from doing good.

International airfare, expat-level lodging, decent living stipend and dedicated in-country staff are provided. Daily challenges and rewards are guaranteed.

Still interested? Then apply here: www.geekcorps.org.

—WAYAN VOTA

Tech Tip

If you have a lot of scripts, shell functions or aliases that depend on SSH, you're probably already using ssh-agent and ssh-add to store your identity so you don't have to retype your passphrase every time.

But what happens when you forget to run ssh-add, and you run one of your scripts that uses SSH? You get a passphrase prompt for every command that uses SSH transport, whether it's rsync, cvs, scp or unison.

To stop the prompts and give yourself a chance to enter the passphrase once, here's how a script can check to make sure you have an identity loaded before doing an `ssh`. Just use the return value of `ssh-add -L`, which is true if you have an identity loaded, false if you don't.

One line, as in a script:

```
ssh-add -L && rsync -a Images xenu.example.com:Images
```

To control a longer block of commands, as in a shell function:

```
sync_images()
{
  ssh-add -L || return 1
  rsync -a Images xenu.example.com:Images
  # more commands here if you like
}
```

Run one of those examples without an identity loaded, and you get a nice "The agent has no identities." error message instead of an annoying passphrase prompt.

—DON MARTI

routerOS:

routerBOARD.com

User Management

- support more than 3000 PPPoE or HotSpot clients
  - full RADIUS support for user parameters
    - tx/rx speed, address, filter rules
    - supports RADIUS real time modification of parameters while users are online
- Peer to Peer protocol control (P2P)
  - per client P2P tx/rx rules with burst support
  - P2P pool
  - complete blocking of P2P

Wireless AP and Backbone

- Wireless monitoring
  - Frequency scanning with detailed report
  - Raw wireless packet sniffer
    - streaming option to Ethereal analyzer
    - option to save to a file format supported by Ethereal
  - Snooper packet inspection
    - analyzes all raw frames received for wireless parameters
    - monitors a single channel or all channels

- Nstreme wireless polling protocol
  - no decrease in speed over long distances (as seen with the 802.11 ACK packet bottleneck)
  - polling improves speed and eliminates contention for access to the wireless bandwidth
  - access point control over Nstreme clients tx data to optimize use of the wireless medium
  - RADIUS support for the access control list including bandwidth settings for wireless clients

- Full 802.11a/b/g support

  The above is a brief description of a few features,

RouterBOARD 500 $140

- Linux Board Support Package (full Debian MIPS installation)
- 266-400MHz MIPS CPU
- 2 miniPCI (one on each side)
- 3 10/100 Ethernets MDI-X
- 64/128MB NAND storage
- PoE 802.3af standard and passive PoE (also 12V PoE)
- Compact Flash
- Low power
- 32MB DDR (64MB optional)
- 6-24V and 25-48V power mode available
- 2-3x faster for networking than the Geode SC1100 boards
- 200-300Mb/s aggregate throughput
- L3 RouterOS license included
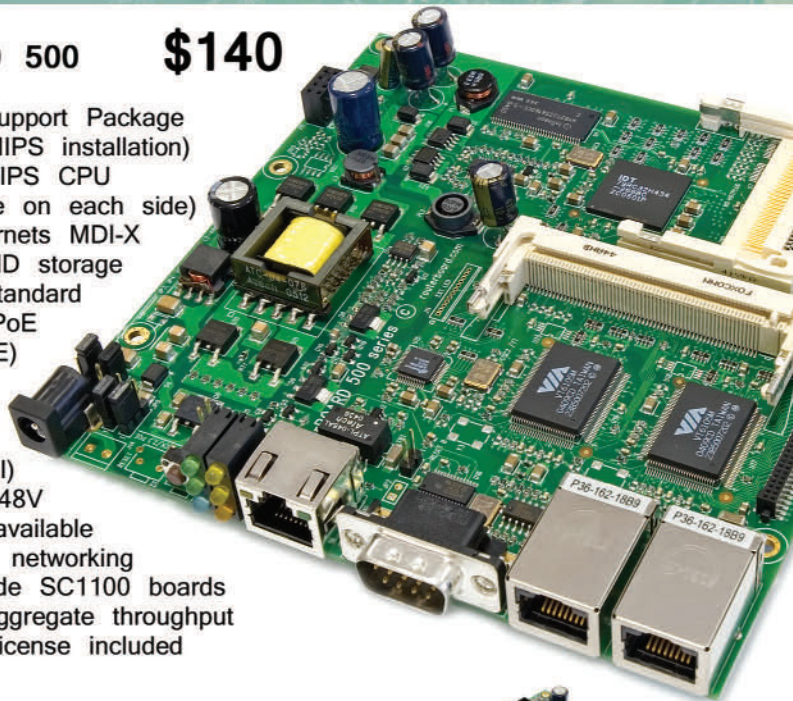
$95
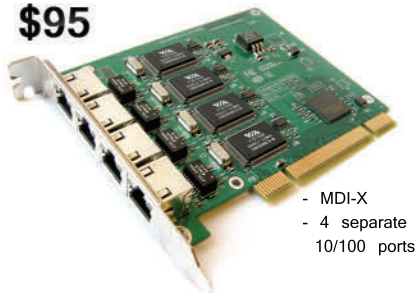
- MDI-X
- 4 separate 10/100 ports

RouterBOARD 44

For the Router Builder !

- rackmount servers and routers
- up to 24 Ethernet ports in a PC
- no more straight/cross cable problems
- server quality VIA VT6105 chips

$195
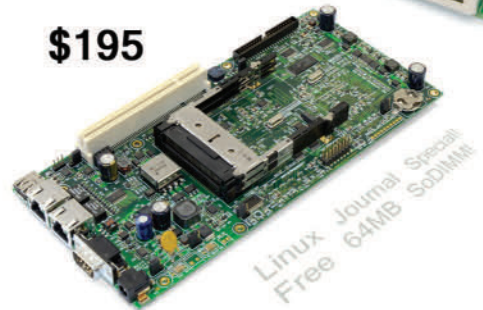
RouterBOARD 230

No feature left behind !

Integrated router with various interfaces. Use as an AP on a tower with up to 500ft PoE. Includes IDE/CF, miniPCI, USB, PCMCIA, UART, PCI, GPIO, LCD controller, Linux SDK, and more.

$120 $65
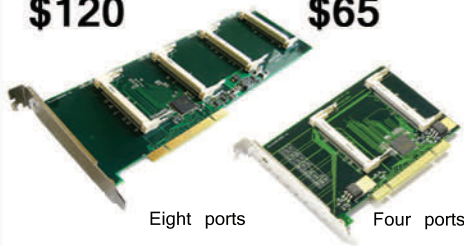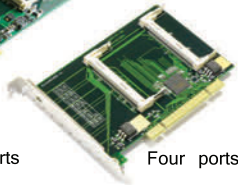
Eight ports   Four ports

RouterBOARD 11/14/18

Multi radio tower !

MiniPCI to PCI adapters for multi radio system. Tested with sixteen radios in one Router/AP.

$240
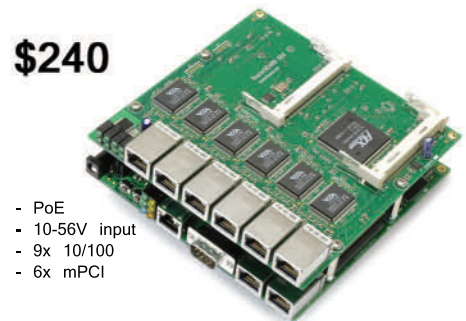
- PoE
- 10-56V input
- 9x 10/100
- 6x mPCI

RouterBOARD 500 & RouterBOARD 564

The Wireless Switchboard !

For a complete multi-radio tower system, the RouterBOARD 500 can carry a daughterboard (RouterBOARD 564) which adds six ethernets and four miniPCI.

MikroTik User Meeting 2006
Prague, Czech Republic, January 19-20

www.mikrotik.com   contact sales@routerboard.com or go to www.routerboard.com
all copyrights and logos belong to their respective owners

LJ INDEX JANUARY 2006

1. Percentage of Americans with broadband Internet in 2002: 21

2. Percentage of Americans with broadband Internet in 2005: 53

3. Percentage of Americans who never use the Internet in 2002: 23

4. Percentage of Americans who never use the Internet in 2005: 22

5. Millions of Linux users in the world: 29

6. Ranking of Antarctica among percentage using Linux: 1

7. Millions of breeding pairs of penguins in Antarctica: 10

8. Ranking of *March of the Penguins* among documentaries in 2005: 1

9. Ranking of *March of the Penguins* among all documentaries in US history: 2

10. 2005 projected Linux percentage of the SOE (server operating environment) in China: 6.1

11. Projected 2005 US dollar revenue for the Linux SOE in China: 19.9 million

12. Projected 2009 US dollar revenue for the Linux SOE in China: 117.6 million

13. Projected 2005–2009 CAGR (compound annual growth rate) for Linux SOE in China: 64

14. Number of available Linux versions: 386

15. Millions of PC users in 2003: 631.8

16. Millions of PC users in 2005: 661

17. Year the number of PC users will pass one billion: 2009

18. Billions of users not on the Net today: 5

19. Lowest target US dollar price point of Linux-based PC (sans monitor) from Novatium: 70

20. Expected Linux percentage share of the converged mobile phone operating systems market in 2009: 17

Sources:

1–4: Pew Internet & American Life

5, 6: Linuxcounter

7: New Zealand government

8, 9: BBC

10–13: International Data Corp.

14–20: CNET

—DOC SEARLS

They Said It

Q: Will Linux be around a hundred years from now?
A: Yes.

—ANDREW MORTON, FROM HIS TALKS ON THE LATEST LINUX LUNACY GEEK CRUISE (OCTOBER 2005)

Somebody knows every part of the kernel better than Linus or I do.

—ANDREW MORTON, FROM HIS TALKS ON THE LATEST LINUX LUNACY GEEK CRUISE (OCTOBER 2005)

I used to know how to do gnuplot, but that brain cell died.

—ANDREW MORTON, FROM HIS TALKS ON THE LATEST LINUX LUNACY GEEK CRUISE (OCTOBER 2005)

If there's one aspect of Linux that has led to its popularity, it is the ability of a suitably enthused individual to produce their own distribution. It has spawned thousands of different solutions and, in turn, has led to the creation of numer-

FREE SOFTWARE
FOUNDATION

be **FREE**

Join us at member.fsf.org

Is It True What They Say about Passwords?

The article "Two-Factor Authentication" by Corey Steele [*LJ*, November 2005] raised a few questions in my mind about using passwords alone.

Without using two-factor authentication, and with the better and quicker password-breaking tools, how long does it take to break a password?

And, how much more time does it take for every character added to a password?

Is my 15-character password much stronger than an 8-character password (assuming they both have been chosen with the same level of inherent "strength")?

How long of a password is needed to make the new better and quicker breaking tools impractical?

Please run some tests. I want to know.

PS: Is it really true that 90% of the passwords in use are on a short list of 100 passwords? I can't believe it.

--
Conrad Schuler

Ah, Memories

I found the article "Radio's Next Generation: Radii" by Dan Rasmussen, Paul Norton and Jon Morgan, about the Internet Radio [*LJ*, November 2005] quite interesting.

It reminds me of a time quite a few years ago when I interfaced a Commodore VIC-20 with a set of Moog Taurus II pedals and turned it into a MIDI controller. We actually used it to perform live at the Festival of Arts in Grand Rapids, Michigan. That was back in the 1980s! My brother played guitar and used the pedals for playing whatever (bass, strings and so on) on his Roland D10. I play drums. Boy, those were the days! I programmed the whole thing in 6502 assembly.

I would like to suggest, if you hadn't thought of it already, that you could get one of those Linux-based SBCs, similar to those advertised in *Linux Journal*, which cost not much more than an old laptop, and interface it with an old SoundBlaster 16 board and make the whole thing self-contained. Then you wouldn't have to play "Hide the Laptop" and you wouldn't have to wait for it to boot before listening to your favorite tunes.

Love that *Linux Journal* magazine. It's not all corporate fluff, but plenty of hacking and even some electronics now and then! I'm so glad someone still publishes this stuff!

--
Ken Peterson

Astronomy Heaven

As a new amateur astronomer and an avid Linux user, I was eager to somehow use both hobbies together. I had recently bought a rather nice Discovery 15-inch Dobsonian telescope. I also bought an Argo Navis Digital Telescope computer (DTC) to guide me to all the cool astronomical objects in the night sky, from Wildcard Innovations, and was anxious to use KStars, a KDE project, on my Linux laptop. To my dismay, the DTC wasn't compatible with KStars. The two systems would not allow me to see what my telescope was pointing at in the night sky. I had put both the DTC and KStars in the "Meade" emulation mode, but still no joy.

Linux, OSS and the GPL to the rescue! I had never gotten involved in an open-source project before but did have many great support experiences via OSS project listservers. So I thought I would give a shot to soliciting a modification to KStars. The KStars developers were fantastic, as I hear most OSS project maintainers are. I contacted the driver maintainer for KStars, Jasem Mutlaq, and he was very friendly and helpful. First I described my situation, and he gave me several ideas to try. They failed, so Jasem got the technical specs for the Argo Navis DTC from their Web site and almost immediately set about modifying an existing driver, the driver for Meade telescope. In a couple of days Jasem had something for me to try. I had never used Subversion before, so Jasem patiently talked me through using it to download and compile the latest KStars source code. We got close on the first try, me doing the test-

ing and Jasem making source mods. The second try was much more successful—the pointer on KStars was moving as I moved the telescope, but when it went off the screen, KStars didn't update the skymap properly. So after one more modification, Jasem had done it. I was in Linux, KStars and Astronomy heaven. I could move the telescope, and KStars, through coordinates passed to it via the DTC, was updating the skymap and very accurately showing me exactly what my telescope was pointing at in the night sky! I could actually look at the KStars screen to find objects to view. Jasem also informed me that the next release of KStars would include the modified Meade telescope driver renamed as "Argo Navis" for all to use and enjoy. Wow that was cool!

After everything was working, Jasem asked me for a little favor. He needed to debug his Sky Commander driver so I put my Argo Navis DTC in the Skycomm mode and operated it with KStars to help him with debugging the Sky Commander Digital Setting Circle unit. He didn't have one, and since my Argo Navis had a mode for it, I was more than happy to help. I gave him the requested debugging output from KStars and he was able to fix his driver.

This development experience was fun, pleasant, fast and most of all *free*. I learned a lot too. Try doing that with a proprietary software package!

By the way, Jasem lives in Kuwait. Talk about a distributed development environment! But the whole process worked *very* well.

--
Doug Phillipson

JBuilder vs. Netbeans

Greetings. I'm a new subscriber to your fine magazine, with my first issue being your November 2005 issue. I see that in that issue you have a letter from a reader promoting JBuilder. I have used JBuilder before, and I can agree that it's a fine IDE, but I wanted to point out that Netbeans is also free and just as good, if not better. Personally, having used both, I prefer Netbeans.

--
Shawn

Testing with Rails

Rails provides great tools for managing test data to build and refine an application. Here's how to use them. **BY REUVEN M. LERNER**

During the last few months, we have looked at Ruby on Rails, an open-source Web application framework written in the Ruby language meant to make Web/database development particularly easy. We already have looked at many facets of Rails development—the division of applications into model, view and controller components; subclassing ActiveRecord for automatic object/database mapping; and the use of Rails validators to ensure the integrity of data stored in the database.

Since starting to work with Rails several months ago, I have been impressed with its design and execution. Developing in Rails feels a bit different and funny at first, but you quickly can get into it, enjoying the fact that much of the tedium has been taken care of by ActiveRecord automatically.

Although Rails can reduce the amount of code we must write in our Web/database applications, it cannot reduce it to zero. And wherever there is code, there are bound to be bugs. As experienced Web/database developers know, testing these sorts of applications can be a bit tricky, because there is a disconnect between what the client sends and sees, and what actually happens with the server. It's not unusual, even today, for Web developers to debug their applications using a combination of print statements and error logs. Indeed, I am personally guilty of this on many occasions, partly out of habit and partly because this is often the best way to find problems with projects.

Managers and programmers alike know that it is cheapest and easiest to fix bugs as soon as they occur, and at an early stage of a project. But programmers often are reluctant to test their software, especially when such testing can be time consuming and tedious.

A relatively simple solution thus has emerged during the past few years, in which programmers were responsible for not only testing the software they had worked on, but also for writing the tests that would check the software in as many places as possible. Such unit testing can help ensure that each individual part of the system is robust, allowing us to depend on it when integrated into a full application.

This month, we look at the built-in Rails functionality for performing unit tests and whet our appetite for writing functional tests as well.

Test Database

Everything we have seen so far might seem reasonable, but there is a danger lurking beneath the surface. If we actually go ahead and test all of our code, we are likely to end up adding, modifying and deleting data in the actual database. On a serious production system, this could be more than inconvenient, it might cause untold problems.

If you have been following along since we first began to work with Rails, you might well remember that we defined three different databases for each of the projects we have worked with. In the case of the simple Weblog application we examined the last few months, we created three databases: blog_development, blog_test and blog_production. We completely ignored the _test and _development databases, concentrating solely on the _production version. Now that we will start testing our application, we will be using the _test database. Only when we are sure that our database and application have passed the test suite will we move it over into the production system.

If you have not already done so, create the test database and load its definitions. On my system, I created a blog user for PostgreSQL and executed the following:

```
$/usr/local/pgsql/bin/createdb -U blog blog_test
```

I then loaded the database definitions that I had saved in the blog/db directory, in a file called create.sql:

```
$ /usr/local/pgsql/bin/psql -U blog blog_test < blog/db/create.sql
```

This loads the table definitions. Assuming that create.sql was identical when I did the same with the development database, I can now assume that the development and test databases are defined in the same way.

But what if we have not been so good about updating create.sql with each modification we've made to our development database? Are we then forced to compare the two database structures manually, update create.sql and then re-import the definitions?

Luckily, the answer is no. Rails comes with a short program, clone_structure_to_test, that copies the structure of the development database to the test database. Note that it copies only the structure, not the contents. To invoke it, switch to the main application directory (blog, in our case) and use the rake, or Ruby make, program, which executes the appropriate section of the Rakefile in the current directory:

```
$ rake clone_structure_to_test
```

If the blog_test database does not yet exist, or if there are other issues, you will get an error message. Otherwise, you see only basic output, as I did:

```
[reuven@server blog]$ rake clone_structure_to_test
(in
/home/reuven/blog)
```

I encountered some initial problems with clone_structure_to_test, with the script claiming I was not the owner of the public schema in PostgreSQL. I got around this by giving the blog database user superuser permissions, which is necessary for the cloning process to work correctly:

```
$ /usr/local/pgsql/bin/psql blog_test

blog_development=# alter user blog createuser;
ALTER USER
```

MBX is the Industry Leader for Server Appliances

From design to delivery we are dedicated to building a partnership with you.

MBX RP-1013 Platform

- Intel® Pentium 4 630 Processor at 3.0 GHz
- 1U Rackmount Chassis 16.5" Deep
- 1GB PC4200 DDR2 Memory
- Maxtor 80GB Serial ATA Hard Drive
- Eight Gigabit NIC's, 4 ports with by-pass
- Optional 16x2 LCD with Keypad
- On-board Compact Flash Socket

- Branded With Your Color and Logo
- Custom OS and Software Install
- No Minimum Quantity Required
- 3 Year Warranty

$1,699 or lease for $52/mo.

MBX RP-2012 Platform

- Intel® Xeon Processor at 3.0D GHz
- 2U Rackmount Chassis
- 1GB PC3200 DDR Memory
- Maxtor 73GB SCSI Hard Drive
- Dual On-board Gigabit NIC's
- Custom OS and Software Install
- No Minimum Quantity Required
- 3 Year Warranty

$2,899 or lease for $89/mo.

MBX is the leader in custom appliances. Many premier application developers have chosen MBX as their manufacturing partner because of our experience, flexibility and accessibility. Visit our website or better yet, give us a call. Our phones are personally answered by experts ready to serve you.

MBX systems

www.mbx.com
1.800.939.0971

Now that we have our test database in place, we can start to write some tests. But where will we put them? Rails, in its typical style, already has defined a location for the tests and assumes we will follow the same convention as the author and other Rails users. This means looking in the blog/test directory, parallel with the blog/app and blog/db directories.

The blog/test directory contains four subdirectories and a single file of Ruby code, all of which are standard in a Rails application. The four subdirectories are fixtures, functional, mocks and unit, and refer to different parts of the testing mechanism that we are expected to create or modify.

Fixtures

Before we can begin testing, however, we need to overcome a problem: if we want to test our application, we should first populate the database tables with data. Moreover, we want it to be the same, consistent data each time we run our tests, so that we can know what is being tested. Rails solves this problem with fixtures, which automatically populate our test database before we want to test. The directory blog/test/fixtures in our system is where we can create such fixtures, generally using YAML, yet another markup language, whose structure is largely determined through indentation.

We create one YAML file for each database table we want to test. Our database contains only one table, so we have to create only a single YAML file, blogs.yml. Sure enough, when we look at the blog/test/fixtures directory, we see that there is already such a file there, demonstrating how we create our fixtures and pointing us to documentation at ar.rubyonrails.org/classes/Fixtures.html, in case we don't completely understand how fixtures work.

We now create one or more entries in our blog.yml file, each of which corresponds to a single row in the Blogs database table, corresponding in turn to a single instance of the Blog object defined in our blog/app/models directory. As a reminder, our table definition is as follows:

```
CREATE TABLE Blogs (
id            SERIAL    NOT NULL,
title         TEXT      NOT NULL,
contents      TEXT      NOT NULL,
posted_at     TIMESTAMP       NOT NULL  DEFAULT NOW(),

 PRIMARY KEY(id)
);
```

Here is how we could create two fixtures for this table:

```
blog_entry_one:
    id:    1
    title:   My first entry!
    contents:  It was a dark and stormy night, and
I forgot my umbrella.  So I decided to tell the world
on my blog.
    posted_at: 2005-Sep-1 22:00:00

blog_entry_two:
    id: 2
    title:   My second entry!
    contents:  It was much nicer this morning.
    posted_at: 2005-Sep-1 22:00:00
```

This is the equivalent of two INSERT statements. Given that INSERT is standard SQL, why would we prefer to use fixtures?

To begin with, fixtures ensure that we always start with the same baseline data. It's terribly frustrating to run tests, only to have them fail because of a uniqueness constraint that was triggered by duplicate data.

The second reason is that it allows us to test our database objects not only in the database itself, but also from a source. That is, the Rails test system loads the data in our YAML file into the database, and then accesses them via our model objects. It then loads the values from our YAML file a second time, making them available via a hash. We can then compare the two, ensuring that data was imported correctly before we begin to test more sophisticated methods.

Unit Tests

With our fixture in place, we now can begin to execute the first of our unit tests. Unit tests check individual methods and objects. If an application's components pass a complete set of unit tests, there is still room for error. However, those errors tend to be from the integration of the units, which are covered by a different set of tests.

Not surprisingly, unit tests are defined inside of the test/unit directory within our main application directory. Rails automatically creates a unit test file for each model class you have defined; thus, my test/unit directory contains a file named blog_test.rb, corresponding to the Blog class defined in app/models/blog.rb. (Remember: Rails model objects have singular names and refer to database tables that have plural names.)

By default, the file contains a skeleton for our unit tests:

```
require File.dirname(__FILE__) + '/../test_helper'

class BlogTest < Test::Unit::TestCase
  fixtures :blogs

  def setup
@blog = Blog.find(1)
  end

  # Replace this with your real tests.
  def test_truth
assert_kind_of Blog,  @blog
  end
end
```

The first line helps to bootstrap the test mechanism and is not of immediate interest. But then we see that we are defining a class (called BlogTest, using the Rails naming convention that we would expect in a file called blog_test.rb). BlogTest is a subclass of Test::Unit::TestCase, which comes with Rails, and provides us with a number of different test-related features.

The definition of BlogTest begins with a declaration, fixtures, whose value is :blogs—an indication that when Rails wants to test our Blog object with the BlogTest object, it should first populate the test database with fixtures defined in test/fixtures/blogs.yml. If we are interested in using multiple fixtures, we can name them as well:

```
fixtures :blogs, :foo, :bar
```

WARNING:

TO THOSE CONSIDERING PERFORMANCE ENHANCEMENT

BEFORE

AFTER
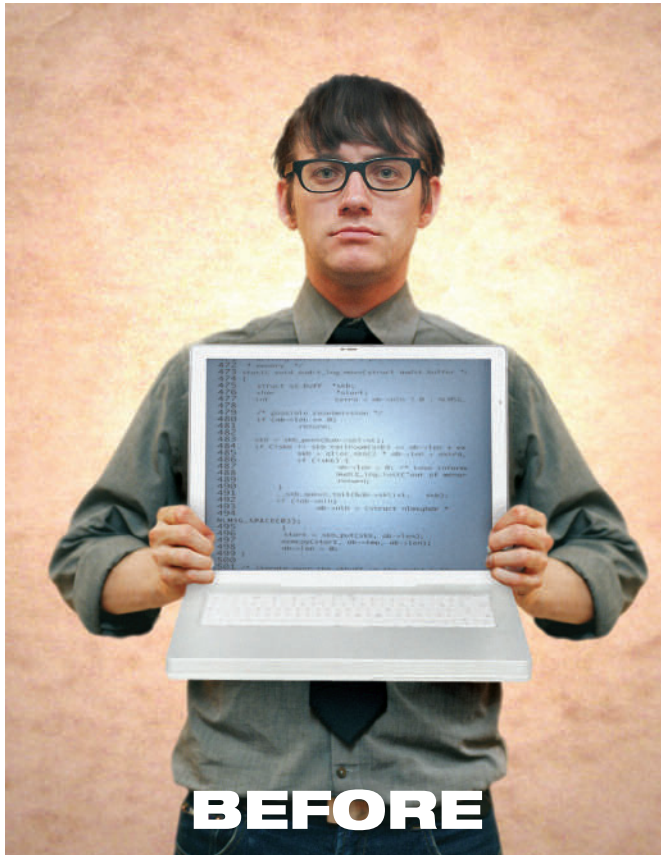
Natural ability and ordinary debuggers can take you just so far. That's why you need NightStar LX™. An integrated suite of tools that gives you full visibility into your Linux® application. You can debug, monitor, analyze and tune at application speed, so you see real execution behavior. Plus, you'll reduce test time and lower costs. NightStar LX. Experience real power for a change.

NightProbe   NightTrace   NightView   NightTune

The NightStar LX suite is extremely habit-forming, and may cause feelings of euphoria.

NIGHTSTAR LX™
Serious tools for serious apps

See if NightStar LX is right for you.
Download a **Free Trial** at
www.ccur.com/nightstar/LX

800-666-4544 • 954-974-1700
Email us at nightstar@ccur.com

concurrent

Two methods are defined for us by default, called setup and test_truth. The setup method, as you might expect from its name, gets things ready for our tests. In this particular case, it invokes Blog.find(), giving it a parameter of 1. In other words, it retrieves the object whose primary key is 1 (that is, blog_entry_one) and puts it into @blog. Perl programmers might need to remember that in Ruby, @blog is an instance variable, not necessarily an array. In this particular case, @blog contains a single instance of Blog—the object that Blog.find(1) returned.

The other method, test_truth, then uses one of the predefined test assertions that comes with Rails. In this particular case, we use the assert_kind_of assertion to check that @blog is an instance of Blog.

To run this initial version of our tests, we simply say:

```
$ ruby blog_test.rb
```

As the tests run, we get a status report. If all goes well, the output should look like this:

```
[reuven@server unit]$ ruby blog_test.rb
Loaded suite blog_test
Started
.
Finished in 19.066227 seconds.

1 tests, 1 assertions, 0 failures, 0 errors
```

The first few times that I ran these tests, I received error messages. The first problem was that I had failed to change blogs.yml from its original, default state, without defining anything but the id primary key field. Because of the integrity constraints that I had put on the Blogs table, PostgreSQL stopped the test, indicating that it would not allow NULL values in the title field.

The second time I tried to run the tests, Rails picked up an error in my YAML file, reminding me that the YAML format requires consistent indentation with spaces and without any tab characters.

Rails wisely distinguishes between failures and errors; both of the above were classified as errors, letting me focus on the overall test environment rather than a particular method.

We can add additional tests by defining new methods. For example, let's check that the title of @blog matches the value we put in the YAML file. We can add the following to the BlogTest class definition:

```
def test_title
assert_equal @blogs["blog_entry_one"]["title"], @blog.title
end
```

Notice how our test begins with the characters test_. This tells Rails that this method should be part of our test suite. Because each individual method is counted separately, it is probably best to have a large number of test methods, each of which contains a small number of assertions. There is no technical reason why you cannot put a large number of assertions in the same method, but it means you might have a tougher time understanding just where the problem lies.

In this case, we are using assert_equal to check that two quantities are equal. Pay close attention to the very similar names, and

you will see what we are doing. The first item to test for equality is @blogs["blog_entry_one"]["title"]. The @blogs hash is created automatically for us by the test suite, and (as mentioned earlier) contains the entire YAML fixture definition file. If @blogs contains the entire YAML definition, then @blogs["blog_entry_one"] contains the first fixture, and @blogs["blog_entry_one"]["title"] contains the title of the first one.

@blog, by contrast, has a singular name because it contains only a single instance of Blog. And like all good objects descended from ActiveRecord, we can use a method to retrieve the contents of a field—in this case, @blog.title. So, the bottom line is that this test helps us check that the title is the same.

More Testing

The above are only two of the types of tests you might want to use on your system. Rails comes with a large collection of assertions, allowing you to test your models in a great variety of ways.

Remember that methods are just one part of the testing equation; you also will want to have appropriate integrity constraints and checks in your table definitions, and a wide variety of inputs to ensure that you are checking many different possibilities. One way to create a large number of fixtures is by creating them dynamically, using the same syntax (known as ERb, or Embedded Ruby) that is used in Rails views.

As I mentioned above, functional tests are another important element in any application's test suite. Functional tests, which operate against Rail controllers, work similarly to our unit tests—in the tests/functional directory, with one test object per controller, and with a test_ method for each method in the controller object. Testing models ensures that your data is going to be robust; testing controllers ensures that no matter what inputs you receive from users via the Web, the application will handle the situation gracefully.

Finally, Rails makes it easy to create mock objects, allowing us easily to pretend that an object has been created. For example, we might want to pretend that a credit-card transaction has gone through, or that we have sent e-mail to 50,000 users of our system, without actually carrying out the task.

Conclusion

Web applications are becoming large and sophisticated enough that they demand disciplined testing techniques to avoid unforeseen problems. Ruby on Rails comes with an integrated test system that makes it easy to create and use tests at all levels—database, model objects and controller objects. It shouldn't come as any surprise that many Ruby developers are fans of test-driven development, in part because Ruby and the Rails environment make it so easy to accomplish. If you are going to develop with Rails, it's worth taking the extra time to add tests into your application. It's easy to do, and it will save you a great deal of time later on.

Resources for this article: www.linuxjournal.com/article/8631.■

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

Easy I/O with IO Channels

IO Channels, a feature of the Glib library, make portable I/O simple and efficient. In this article, Robert shows how. **BY ROBERT LOVE**

Glib's IO Channels provide several powerful features:

- Buffered I/O: as with C's Standard Library, IO Channels provide user-side buffering, which minimizes the number of system call invocations and ensures that I/O is performed in optimally sized units.

- Portability: IO Channels are portable, working in various UNIX systems as well as Windows.

- Simple yet efficient I/O routines: helper routines to make common programming chores, such as "read exactly one line" or "read the whole file" easy.

- Main loop integration: integration into the Glib main loop means multiplexed I/O and event-driven programming is easy.

Although designed for powerful and complex GNOME applications, Glib is actually a separate library from GNOME and readily usable from any C application.

Main Loops

A main loop, sometimes called an event loop, allows a single-threaded process to wait for and handle events originating from multiple sources. Most GUI programmers are familiar with main loops: they allow event-driven GUI programming to register callback functions that are invoked in response to events, such as a button press or window close. The Gtk+ main loop is built on top of Glib's main loop.

The Glib main loop is implemented using multiplexing I/O—in Linux, via the poll() system call. Events are associated with file descriptors, which are watched via poll(). In this manner, the application need not check incessantly for new events but can sleep, consuming no processor time, so long as there is no activity.

Glib's main loop associates a callback function with each event. When the event occurs, the main loop will call back into the given function. Callbacks are invoked in the order that their associated event occurs, although priority may be given to events to change this order. Because multiple events may be watched and multiple callback functions registered, even a single-threaded process can juggle numerous events.

Glib

The Glib library is GNOME's base library, providing portability wrappers and a set of helper functions to make programming in C less arduous. Although part of GNOME, Glib is very much usable on its own, and many non-GNOME projects do indeed utilize Glib without touching any other parts of GNOME. In fact, Glib is even beneficial to console applications. This article makes no assumptions about the use of other GNOME components; the interfaces covered work equally well in a complex GNOME application and a simple console program.

Compiling an application with the requisite Glib support is made easy by the pkg-config program. You can build a binary gio from the source file gio.c with the following command:

```
gcc -Wall -O2 \
        `pkg-config --cflags --libs glib-2.0` \
        -o gio \
        gio.c
```

IO Channels

An IO Channel is represented by the GIOChannel data structure. Its fields are private, and it is accessed using only the official IO Channel interfaces.

Each IO Channel is associated with a single file or other "file-like" object. On Linux, an IO Channel can be associated with any open file descriptor, including sockets and pipes. Once associated, the IO Channel is used to access the file.

Watches are created against a given IO Channel, along with a set of events on which to wait for and a callback function to invoke in response. The watches then integrate with Glib's main loop. When an event occurs—say, a socket has new data available for reading—the watch is triggered and the callback is automatically invoked.

The watch lies at the heart of the power of IO Channels: applications can create multiple watches and integrate them, along with numerous other events, into the Glib main loop, providing event-driven programming to even simple single-threaded applications.

Getting Started

Listing 1 is a complete and working console application that uses IO Channels to communicate across two pipes. It creates two IO Channels, one for the read side of the pipe and another for the write side of the pipe. It then registers watches for these two IO Channels. One watch invokes a callback, gio_in(), when the pipe is available for reading (that is, when a read from the read side of the pipe will not block). The other watch invokes a callback, gio_out(), when the pipe is available for writing (that is, when a write to the write side of the pipe will not block). The gio_out() callback writes a small message into the pipe. The gio_in() callback reads the available data from the pipe and prints it to standard out.

To be sure, this is an example rooted solely in explanation. It is silly to operate a pipe like this in a single application. Further, the program will continually read from and write to the pipe (you can kill the process with Ctrl-C). Nonetheless, this example serves a good purpose: it demonstrates event-driven programming and the utility of a main loop multiplexing I/O. The natural extension of this program would be to separate it into two processes, a consumer and a producer, and actually communicate interprocess over the pipe. Add a hand-

Listing 1. A Complete and Working Console Application That Uses IO Channels to Communicate across Two Pipes

```c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <string.h>

#include <glib.h>

static gboolean
gio_in (GIOChannel *gio, GIOCondition condition, gpointer data)
{
        GIOStatus ret;
        GError *err = NULL;
        gchar *msg;
        gsize len;

        if (condition & G_IO_HUP)
                g_error ("Read end of pipe died!\n");

        ret = g_io_channel_read_line (gio, &msg, &len, NULL, &err);
        if (ret == G_IO_STATUS_ERROR)
                g_error ("Error reading: %s\n", err->message);

        printf ("Read %u bytes: %s\n", len, msg);

        g_free (msg);

        return TRUE;
}

static gboolean
gio_out (GIOChannel *gio, GIOCondition condition, gpointer data)
{
        const gchar *msg = "The price of greatness is responsibility.\n";
        GIOStatus ret;
        GError *err = NULL;
        gsize len;

        if (condition & G_IO_HUP)
                g_error ("Write end of pipe died!\n");
```

```c
        ret = g_io_channel_write_chars (gio, msg, -1, &len, &err);
        if (ret == G_IO_STATUS_ERROR)
                g_error ("Error writing: %s\n", err->message);

        printf ("Wrote %u bytes.\n", len);

        return TRUE;
}

void
init_channels (void)
{
        GIOChannel *gio_read, *gio_write;
        int fd[2], ret;

        ret = pipe (fd);
        if (ret == -1)
                g_error ("Creating pipe failed: %s\n", strerror (errno));

        gio_read = g_io_channel_unix_new (fd[0]);
        gio_write = g_io_channel_unix_new (fd[1]);
        if (!gio_read || !gio_write)
                g_error ("Cannot create new GIOChannel!\n");

        if (!g_io_add_watch (gio_read, G_IO_IN | G_IO_HUP, gio_in, NULL))
                g_error ("Cannot add watch on GIOChannel!\n");
        if (!g_io_add_watch (gio_write, G_IO_OUT | G_IO_HUP, gio_out,
NULL))
                g_error ("Cannot add watch on GIOChannel!\n");
}

int
main (void)
{
        GMainLoop *loop = g_main_loop_new (NULL, FALSE);
        init_channels ();
        g_main_loop_run (loop); /* Wheee! */
        return 0;
}
```

ful of other IO Channels, some GUI events, a few timers, and so on, to the main loop, and you will have a real program!

Creating an IO Channel

There are two ways to create a new IO Channel. The easiest method creates the IO Channel from an existing open file descriptor. The file descriptor can map to any object, including sockets and pipes:

```c
GIOChannel *gio;

gio = g_io_channel_unix_new (fd);
if (!gio)
        g_error ("Error creating new GIOChannel!\n");
```

As its name suggests, this function is UNIX-specific. Another method is available for creating an IO Channel in a platform-independent manner:

```c
GIOChannel *gio;
GError *err = NULL;

gio = g_io_channel_new_file ("/etc/passwd"
                             "r",
                             &err);
if (!gio)
        g_error ("Error creating new GIOChannel: %s\n",
                err->msg);
```

The second parameter specifies the mode with which to open the file: one of r, w, r+, w+, a or a+. These values have the same meaning as with fopen(). For example, in this code snippet, we are asking to create a read-only IO Channel.

In our example program in Listing 1, we create two IO

Tools to build. Tools to tune.
Tools to transform.

You're building software applications that will change the
way the world works, plays and interacts. We're right there with you.
With products that help your applications run better and faster.
Compilers. Tuners. Threaders. Debuggers. And much more.
Our products help you integrate software innovation with Intel
platform technologies—from idea to optimization.

Intel® | software network

Check us out at www.intel.com/software/products

intel.

Channels using g_io_channel_unix_new(), one for each end of the pipe.

Creating a Watch

Given an IO Channel, creating a watch is easy:

```
guint ret;

ret = g_io_add_watch (gio, G_IO_IN, callback, NULL);
if (!ret)
        g_error ("Error creating watch!\n");
```

The first parameter, gio, is the IO Channel we want to watch. The second parameter is a mask of one or more conditions for which to watch. The condition G_IO_IN is true when there is data to be read without blocking. Other conditions are G_IO_OUT (data can be written without blocking), G_IO_PRI (urgent data is available to read), G_IO_ERR (an error occurred) and G_IO_HUP (the connection was hung up). The third parameter is the callback function that the Glib main loop will invoke when the event occurs.

Watch callbacks take the following form:

```
gboolean callback (GIOChannel *gio,
                   GIOCondition condition,
                   gpointer data);
```

where gio is the applicable IO Channel, condition is a bitmask of the triggered events and data is the last argument given to g_io_add_watch().

If the return value of the callback is FALSE, the watch is automatically removed.

In our example program in Listing 1, we create two watches, one for each of our IO Channels.

Reading from an IO Channel

The Glib library provides three basic interfaces for reading from an IO Channel.

The first, g_io_channel_read_chars(), is used to read a specific number of characters from an IO Channel into a pre-allocated buffer:

```
GIOStatus g_io_read_chars (GIOChannel *channel,
                           gchar *buf,
                           gsize count,
                           gsize *bytes_read,
                           GError **error);
```

This function reads up to *count* bytes from the IO Channel channel into the buffer buf. Upon successful return, bytes_read will point to the number of bytes actually read. On failure, error will point to a GError structure.

The return value is an integer with one of four values: G_IO_STATUS_ERROR (an error occurred), G_IO_STATUS_NORMAL (success), G_IO_STATUS_EOF (end-of-file was reached) or G_IO_STATUS_AGAIN (resource temporarily unavailable, try again).

The second interface, g_io_channel_read_line(), is used to read an entire line from a given IO Channel. It will not return until a newline-delimited line is read:

```
GIOStatus
g_io_channel_read_line (GIOChannel *channel,
                        gchar **str_return,
                        gsize *length,
                        gsize *terminator_pos,
                        GError **error);
```

Upon successful return, str_return will contain a pointer to a newly allocated block of memory of *length* bytes. terminator_pos is the offset into str_return of the terminating character. The data returned by this function must be freed via a call to g_free().

The final interface, g_io_channel_read_to_end(), reads all remaining data from the file into the given buffer:

```
GIOStatus g_io_channel_read_to_end (GIOChannel *channel,
                                    gchar **str_return,
                                    gsize *length,
                                    GError **error);
```

Upon successful return, str_return will contain a pointer to a newly allocated block of memory of *length* bytes, which must be freed via g_free().

This function should not be used on IO Channels that map to file descriptors that do not necessarily return end-of-file when exhausted of data. For example, we could not use this function in our example program, because pipes do not return end-of-file until the other side has closed its end of the connection. Thus, our example would block indefinitely if we use g_io_channel_read_to_end().

Instead, in our example, we use g_io_channel_read_line() to read an entire line from the pipe.

Writing to an IO Channel

Glib provides a single interface for writing to an IO Channel:

```
GIOStatus
g_io_channel_write_chars (GIOChannel *channel,
                          const gchar *buf,
                          gssize count,
                          gsize *bytes_written,
                          GError **error);
```

A successful call to g_io_channel_write_chars() will write up to *count* bytes from the buffer pointed at by buf into the file represented by the IO Channel channel. If count is negative one, buf will be treated as a NULL-delimited string. On return, bytes_written contains the actual number of bytes written.

Like C's Standard I/O Library, IO Channels perform buffered I/O to optimize performance. Thus, a write request may not actually be submitted to the kernel after each call to g_io_channel_write_chars(). Instead, glib may wait until a sufficiently large buffer is full and then submit the write request, in one large swoop.

The g_io_channel_flush() function is used to force a submission of any pending write requests to the kernel:

```
GIOStatus
g_io_channel_flush (GIOChannel *channel,
                    GError **error);
```

Boost Performance on Linux Clusters

Create, Analyze and Optimize High-Performance Applications on Clusters of Intel® Processor-based Systems
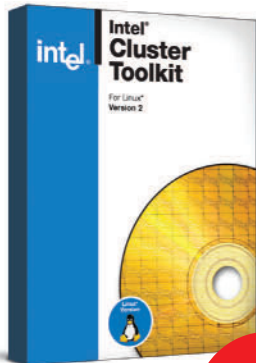
Introducing the Intel® Cluster Toolkit 2.0 for Linux

These tools address every stage of the development process, enabling you to achieve optimized performance for Intel® processor-based cluster systems.
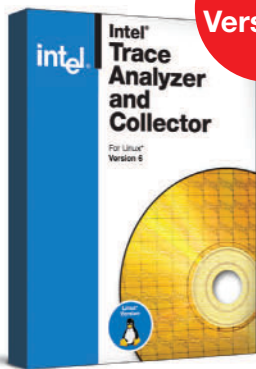
Intel® Trace Analyzer and Collector 6.0
Optimized MPI Performance Analysis

Intel® Trace Analyzer and Collector bundle together graphical analysis and runtime event traces, providing maximum performance to cluster application developers.

- Navigate through the trace data at different levels of abstraction (cluster, node, and process) to address data scalability.
- View important aspects of application runtime behavior in detail and in the aggregate.
- Low-overhead, event-based tracing to profile the behavior of multithreaded applications with and without a Message-Passing Interface (MPI).

NEW Versions!
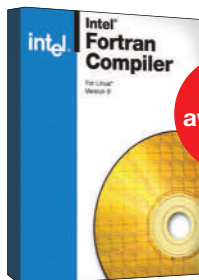
Intel® Fortran Compiler 9.0 for Linux

Compatible with Linux* development tools most programmers already use. A powerful combination of Compaq Visual Fortran (CVF) front-end and Intel® Fortran back-end, along with substantial CVF source code compatibility.

9.0 available NOW!

Intel® MPI Library for Linux

Use this high-performance message-passing library to develop applications that can run on multiple cluster fabric interconnects chosen by the user at run-time.
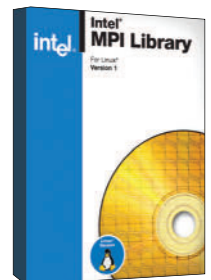
Programmer's Paradise®

To order or to request additional information
Call: 800-423-9990
Email: intel@programmers.com

FREE trials available at:
programmersparadise.com/intel

YOU SAVE UP TO $73!	Paradise #	Retail	Discount
Intel® Cluster Toolkit for Linux 2.0	I23 0E70	$749.00	$675.99
Intel® Trace Analyzer and Collector 6.0	I23 0E8G	$499.00	$461.99
Intel® Fortran Compiler 9.0 for Linux	I23 0BL3	$699.00	$646.99
Intel® MPI Library 1.0	I23 0BFP	$499.00	$461.99

Buffering can be turned off altogether, if so desired:

```
g_io_channel_set_encoding (gio, NULL, NULL);
g_io_channel_set_buffered (gio, FALSE);
```

The second function disables buffering. The first sets the IO Channel's encoding to NULL, which is required for disabling buffering.

In our example program, we use g_io_channel_write_chars() to write a small string to the pipe. We do not mess with the buffering of the IO Channel.

Seeking Around an IO Channel

As a file is written to or read from, Glib automatically updates the application's position within the file. When a file is first opened, the position is set to the start. Read four bytes, and the position is subsequently set to the fifth byte. This is the file I/O behavior nearly all developers are familiar with.

Sometimes, however, an application wants to seek around the file on its own. For that, Glib provides a single function:

```
GIOStatus
g_io_channel_seek_position (GIOChannel *channel,
                            gint64 offset,
                            GSeekType type,
                            GError **error);
```

A successful call will change the current position in the file as described by offset and type.

The type parameter is one of G_SEEK_CUR, G_SEEK_SET or G_SEEK_END. G_SEEK_CUR asks that the file's position be updated to offset bytes from the current position. G_SEEK_SET asks that the file's position be set to offset. Thus, the following code sets the file position to the start of the file:

```
GIOStatus ret;
GError err;

ret = g_io_channel_seek_position (gio,
                                  0,
                                  G_SEEK_SET,
                                  &err);
if (ret)
        g_error ("Error seeking: %s\n",
                 err->message);
```

Finally, G_SEEK_END asks that the file's position be set to offset bytes from the end of the file.

We do not use g_io_channel_seek_position() in our sample application, because even if we had a reason to, pipes are not seekable.

Closing an IO Channel

When done with an IO Channel, it is destroyed and the file is closed via a call to g_io_channel_shutdown:

```
GIOStatus
g_io_channel_shutdown (GIOChannel *channel,
                       gboolean flush,
                       GError **err);
```

If flush is TRUE, any pending I/O is first flushed.

Deprecated Functions

Several older IO Channel functions are provided by Glib: g_io_channel_read(), g_io_channel_write() and g_io_channel_seek(). The functions discussed in this article have replaced these older, deprecated functions and should be used instead. Particularly, *never* mix these old functions with the other functions on the same IO Channel.

Putting It All Together

We have now covered all of the IO Channel interfaces used in Listing 1. Only two details go unexplained. First, in our example program, we create a pipe in the usual manner:

```
int fd[2], ret;

ret = pipe (fd);
if (ret)
        g_error ("Creating pipe failed: %s\n",
                 strerror (errno));
```

Second, our main() function initializes the Glib main loop, calls our function to initialize the IO Channels and then runs the main loop:

```
int
main (void)
{
        GMainLoop *loop;

        loop = g_main_loop_new (NULL, FALSE);
        init_channels ();
        g_main_loop_run (loop);     /* Wheee! */
        return 0;
}
```

The g_main_loop_new() function creates a new main loop and returns a pointer to a new GMainLoop structure. Once ready to start the main loop running, we call g_main_loop_run() and let Glib handle the rest.

Conclusion

Applications that use IO Channels combine a portable multiplexed I/O solution with smart buffering and Glib main loop integration. The result is a solution that allows applications to juggle I/O among hundreds of file descriptors. A graphical network client can manage all of its open sockets, handle new connections seamlessly, juggle a dozen open files and respond to numerous GUI events from a single place and with a single thread.

Glib's IO Channels make I/O easy, efficient, and—gasp—even fun!

Resources for this article: www.linuxjournal.com/article/8632.

Robert Love is a senior kernel hacker in Novell's Ximian Desktop group and the author of *Linux Kernel Development* (SAMS 2005), now in its second edition. He holds degrees in CS and Mathematics from the University of Florida. Robert lives in Cambridge, Massachusetts.

Is Your $HOME a Money, er, Messy Pit?
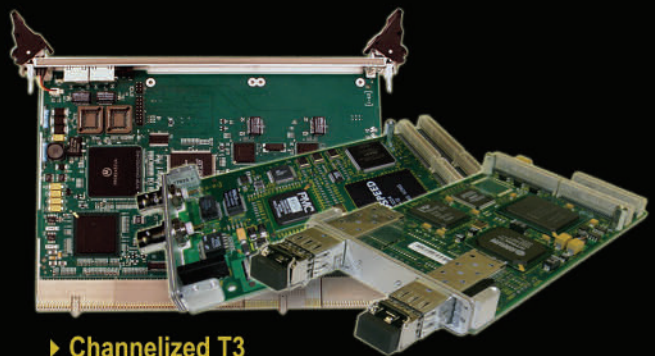
How to unclutter your $HOME or even your whole disk and keep it uncluttered. **BY MARCEL GAGNÉ**

That's very amusing, François. I can't remember when I've I laughed so much. *Quoi?* I'm sorry, *mon ami*, I did not realize that you were serious. It's true that most people will tell you that their home is a money pit, but that is not the reason the environment variable for a user's home directory is $HOME. The dollar sign in front of the variable name is simply how these things are accessed by the shell. Now, don't look so sad, François. It really was very funny, and if I were you, I would write that down and use it someday.

Since you mentioned it, however, your $HOME can become expensive in other ways. People always seem to be able to find some kind of construction project for their homes, whether it be remodeling a bathroom, updating a kitchen or building a new deck. So it seems to be a similar case when running their Linux systems. I'm guilty of it myself, *mon ami*. During the course of any given year, I download hundreds of packages that I compile and install. Sometimes, I uninstall them and sometimes I forget about them, deleting the source folder and its accompanying uninstall scripts. The result is unused programs, forgotten programs and hanging files without a purpose. Because most Linux software is free, your Linux home isn't a money pit, but it can be a messy pit. All of this fits in nicely with tonight's menu.

We'll talk later, François—the door is opening and our guests are arriving. *Vite!* Direct them to their tables. Welcome, everyone, to *Chez Marcel*, home of the finest in Linux fare, not to mention the world's greatest wine cellar. François will show you to your tables, and then he will fetch the wine. I believe the 2003 Clarendon Hills Brookman Syrah is already drinking well. François, you will find a case in the East wing of the cellar.

Those of you who visit *Chez Marcel* on a regular basis are very familiar with the extract-and-build five-step. As a quick recap, let's consider a package called ftl-drive-3.1.tar.gz. To build and install the package using the extract-and-build five-step, you would use the following commands:

```
tar -xzvf ftl-drive-3.1.tar.gz
cd ftl-drive-3.1
./configure
make
su -c "make install"
```

This tends to work quite well, and if you decide that this package doesn't do what you want it to do, or that you no longer need it on the system, the way to install it is to go back into the source directory and type the following:

```
su -c "make uninstall"
```

Sounds simple enough, except that many source packages do not include an uninstall script. This is what makes package managers like RPM or DEB so much nicer, and it's why I suggest that you look for precompiled packages whenever possible. The package managers keep track of what files were installed where and handle the messy process of removing everything that makes up an installed package. It would be truly fantastic if you could do the same with source packages. That's where Felipe Eduardo Sánchez Díaz Durán's CheckInstall comes into play. To get your copy, go to the CheckInstall Web site (see the on-line Resources). Source is available (though you likely won't need it), as well as RPM, DEB and Slackware packages. The advantage of the binary packages (other than simplicity) is that they are already configured for your distribution's package manager. Why is this important? Keep reading.

CheckInstall makes it possible to generate DEB or RPM packages from source code bundles. The idea is that you run the first four steps of the extract-and-build five-step and run the `checkinstall` program instead of the final `su -c "make install"`. Because the result is a precompiled package, your system's package manager takes care of the installation and removal of the packages. Let's pretend that you already have gone through the extract of a source package, right through to the `make` stage. Now, type the following:

```
su -c "checkinstall"
```

The first time you run checkinstall, you will get the following message:

```
The package documentation directory ./doc-pak does not exist.
Should I create a default set of package docs?  [y]:
```

The correct answer here is y, or simply press Enter to accept the default. The next step puts together the documentation package. In some cases, you may get a message that no documentation was included with the package. There's no need to worry if this happens. Just before things really start happening here, you'll be asked for a description of the package. As I write this, I am using CheckInstall with a kids' concentration game I found (see Resources), so my description looks something like this:

```
Please write a description for the package.
End your description with an empty line or EOF.
>> Concentration game for kids


***************************************
**** RPM package creation selected ***
***************************************

This package will be built according to these values:
```

```
1 -   Summary: [ Concentration game for kids ]
2 -   Name:     [ concentration ]
3 -   Version: [ 1.1 ]
4 -   Release: [ 1 ]
5 -   License: [ GPL ]
6 -   Group:    [ checkinstall ]
7 -   Architecture: [ i386 ]
8 -   Source location: [ concentration-1.1 ]
9 -   Alternate source location: [   ]
10 - Requires: [   ]
11 - Provides: [ concentration ]


Enter a number to change any of them or press ENTER to continue:
```

Once you pass this point, a lot of information will fly past you on the screen, followed by the creation of the RPM package itself. Assuming everything worked out well, you should see a message similar to the one I received when building the above game:

```
Done. The new package has been saved to

 /usr/src/RPM/RPMS/i386/concentration-1.1-1.i386.rpm
 You can install it in your system anytime using:

      rpm -i concentration-1.1-1.i386.rpm
```

Should you decide that you want to uninstall this package, the standard RPM tools will do the trick, either graphically or with the command line:

```
rpm -e concentration
```

For those who prefer to avoid the command line whenever possible, you might want to check out Ingo Stieran's KInstaller, an install program geared specifically to KDE. You can get a copy of KInstaller from the Web site (see Resources). It's kind of interesting that building a package to avoid the command line is done at the command line, but this is a pretty straightforward (you guessed it) extract-and-build five-step.

When you first start the program, you are treated to a setup wizard (Figure 1). Because this is essentially an installation tool for other KDE programs, your first step is to identify the KDE base directory. The easiest thing to do here is to click the Guess button. The field you should pay more attention to, however, is Storage dir. By default, KInstaller needs to know where you are going to extract and perform the installs for your packages. That directory is /tmp, but that may not be the best choice for everyone. Because I do a lot of package testing, I have a rather large partition, /mnt/bigdrive, that I use exclusively for this purpose. Choose something appropriate for your system. Finally, look at the check box labeled Move source packages to storage directory after installation. If you want to keep a copy of your source packages, check this box before continuing.

When you click Finish, KInstaller starts. You'll see a two-tabbed window with the Installation tab selected by default. Near the top is a field labeled Select, which you will use to navigate to a source package. Once you press the Install button, the log window below starts filling up with the details of the build, from extraction to compilation (Figure 2). It's all
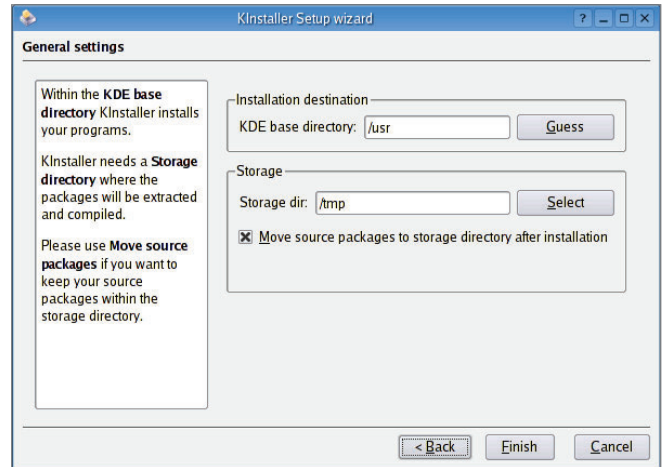
Figure 1. KInstaller starts with a simple configuration wizard.
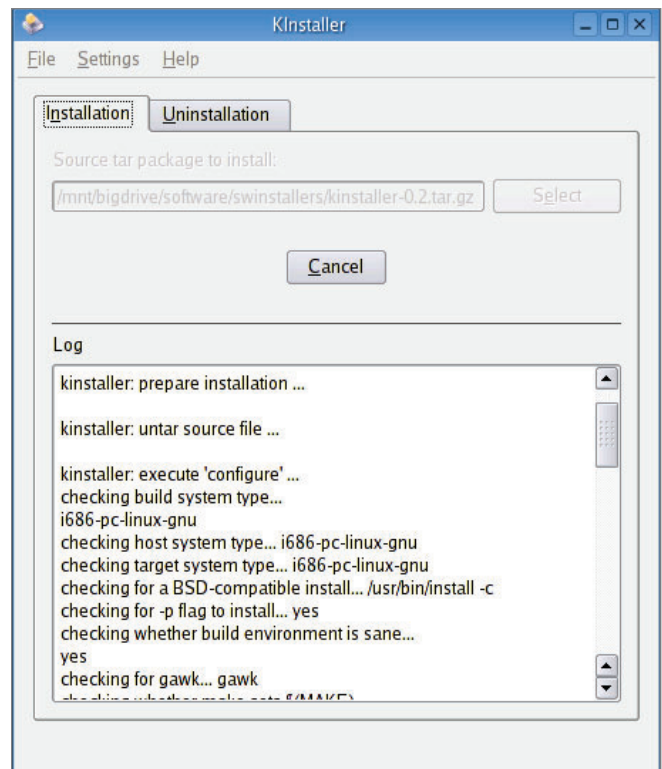
Figure 2. KInstaller handles every step of the extract-and-build five-step.

done automagically. When the compilation completes, the install process launches and you'll get a pop-up box asking you to enter the root password.

Once you have done a few of these, take a moment to look at those tabs again, and you'll see that one of them is labeled Uninstallation. Every package you have installed using KInstaller will have an entry here. To uninstall the package (regardless of whether an uninstall script was provided), select the package you want removed from the list and click the Uninstall button. Once again, a pop-up window will ask you for the root password to go ahead and remove the package. KInstaller does all this without using checkinstall. The package tracks installs on its own. It is also, as I mentioned earlier,

geared to KDE packages.

While François attends to what appears to be an unusually large number of empty glasses—it seems, *mes amis*, that you are enjoying the wine as much as I am—I would like to show you another package in the style of KInstaller. Check out Javier Campos Morales' KConfigure. KConfigure isn't geared specifically to KDE packages, despite the very KDE-ish sounding name. What makes this package interesting is the way it handles the process of uninstalling software; it uses our old friend, checkinstall.

To start working with KConfigure, visit the Web site at SourceForge (see Resources). Source is available (and yes, it is a simple extract-and-build five-step). When you run the program, a basic two-paned window appears with a small number of icons running along the top. The icons have familiar-sounding names like Configure, Make and Install.

This is where things get interesting, so let's pause for a moment and look at KConfigure's configuration menu (Figure 3). I'll let you explore the Fonts and Miscellaneous settings on your own. For now, I want to concentrate on the Check Install Settings. To use the checkinstall program, click the Use Checkinstall check box and make sure you have the right path to the checkinstall executable (on my system, it resides in /usr/bin). I make a point of mentioning this since the default entered by KConfigure is /usr/sbin. When you are happy with your settings, click OK.
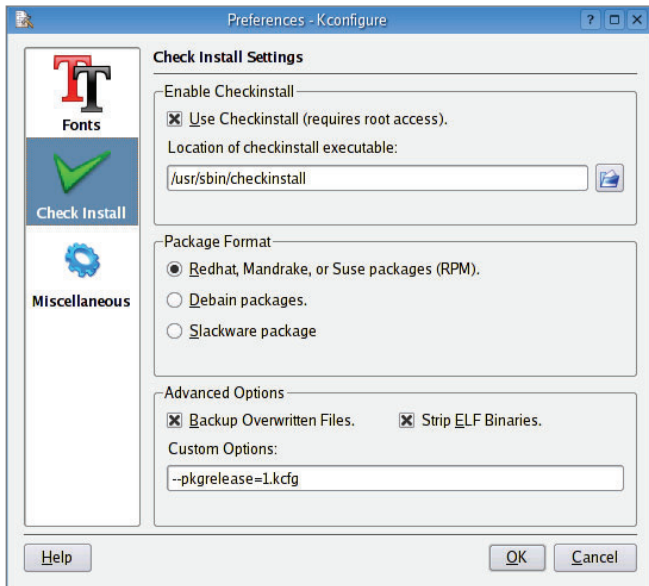
Figure 3. KConfigure's settings allow you to use checkinstall to build packages.

Time to build a package. Click Open and select a source bundle. KConfigure now shows you a window with a listing of the files that are about to be extracted. The default directory for the extract and build is the directory from which you selected the bundle, but you can override that by clicking the Extract files in... radio button and selecting another location. Click OK when you are happy with your choice, and a few seconds later a dialog box will inform you that the package was extracted successfully. Now you are ready to click the Configure button. A log of the configure process appears in the top-half window with any errors displayed in the bottom half. The same display format applies when you finally click the Make button (Figure 4).
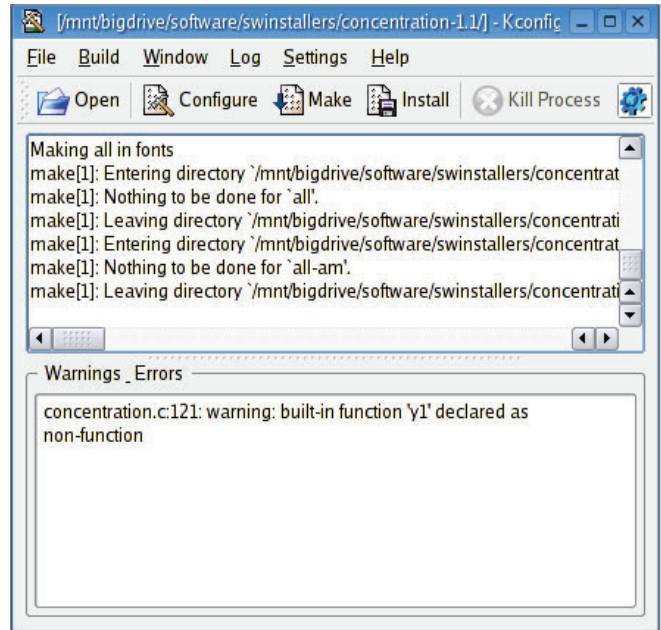
Figure 4. Details, details—a KConfigure build displays the work happening in the background.

Once the configure and make process has completed, you could just go ahead and click Install. Assuming that you have checkinstall and that you want to use it, make sure that you click Build on the menu bar and select Check Install from the Install With submenu. Now, go ahead and click Install. A prompt appears asking you to confirm the root password before continuing. After doing so, the RPM package is created. Make sure you check the bottom of the log window where the path to your newly created RPM package will be. On my system, that was:

```
/usr/src/RPM/RPMS/i386/concentration-1.1-1.kcfg.i386.rpm
```

Mon Dieu! The time has come and gone so quickly, and it is already closing time. Surely there must be an open-source programmer somewhere working on a program to extend time. We could even use one of the tools covered today to build and install the package, *non?* In the meantime, we shall have to live with the clocks we have. Besides, we can easily keep the doors open a little longer while we enjoy one last glass of wine. Please raise your glasses, *mes amis*, and let us all drink to one another's health. *A votre santé! Bon appétit!*

Resources for this article: www.linuxjournal.com/article/8634.■

Marcel Gagné is an award-winning writer living in Mississauga, Ontario. He is the author of the all new *Moving to Linux: Kiss The Blue Screen of Death Goodbye!*, 2nd edition (ISBN 0-321-35640-3), his fourth book from Addison-Wesley. He also makes regular television appearances as Call for Help's Linux guy. Marcel is also a pilot, a past Top-40 disc jockey, writes science fiction and fantasy, and folds a mean Origami T-Rex. He can be reached via e-mail at mggagne@salmar.com. You can discover lots of other things (including great Wine links) from his Web site at **www.marcelgagne.com**.

MORE SPACE. LESS MONEY.

UNLIMITED AFFORDABLE NETWORK STORAGE

Everybody needs more space. And they need to spend less money. What if you can both have more space and spend less money?

What if you could put 7½ terabytes in only 3 rack units? What if that 7½ terabytes cost less than $10,000? Including the SATA disk drives. Imagine if you could glue it all together with a RAID appliance into one system. What if you could add as much storage as you wanted, one shelf at a time, and never have to `fork-lift' anything?

Coraid's new SATA EtherDrive Storage allows you to do just that. Using industry standard SATA disk drives, EtherDrive Storage connects disks directly to your Ethernet network. Each disk appears as a local drive to any Linux, FreeBSD or Solaris system using our open ATA-over-Ethernet (AoE) protocol. Since the disks just appear as local drives you already know how to use them.

The EtherDrive® SATA Storage Shelf is a 3U rack-mount network appliance that contains 15 SATA drive slots. Its triple redundant power supply protects you from your most likely failure. Its dual Gb Ethernet interfaces allow your data to go fast; 200MB per second. And at a very affordable price. List price for the EtherDrive Storage Shelf, without disks, is only $3,995.

Our companion product, the RAIDBlade RAID controller, allows a virtually unlimited number of Storage Shelves to be combined into a set of logical AoE storage devices.

Now you can have unlimited storage at a very affordable price. For complete information, visit our website at www.coraid.com, or call, toll-free, 1-877-548-7200. And we'll show how we've made network storage so affordable, you can have all the space you want.

www.coraid.com
info@coraid.com
1.706.548.7200

CORAID

Exploring Pipes, Test and Flow Control

More shell script programming building blocks, and finally, our first small script. **BY DAVE TAYLOR**

L ast month we started out easy, with a discussion of file redirection. This month I continue to talk about the basic building blocks of shell script programming by exploring pipes, then we jump into some basic programming statements so we can move into an interesting programming project.

Many people who start working with the Linux command line don't realize that it is unlike the world of the graphical interface, where programs are all standalone entities that can't really interact with each other (that is to say, Photoshop can't feed output directly to Microsoft Word, if you're a Windows person, or in Linux terms, The GIMP can't easily interact with OpenOffice.org). We've been taught to think of programs as autonomous, but when you're on the Linux command line, programs can all communicate with each other.

This is a real boon, because it means that instead of having roughly 1,800 different commands available, you actually have the equivalent of millions of different commands that can be put together to do just about anything you can imagine.

The key is the | (pipe) symbol, which hooks the output of the first command to the input of the second. For example, want to know how many files you have in your home directory? The simple solution is:

```
ls | wc -l
```

which is invoking the ls command to list files, but instead of displaying the output on the screen, it's actually fed to the wc (word count) program, with the -l option indicating that we want to have a count of the number of lines in the input stream, rather than the number of words or characters.

Now, here's something a bit more complex. Let's say that you want to know how many files you have that were last modified in August. When you use the `ls -l` command, you notice that the lines are typified by the following:

```
drwxr-xr-x   11 taylor  taylor   374 Aug  16 21:57 ConnectSafely
```

This provides lots of information, but all we care about is that the month of last modification is shown as a three-letter

abbreviation surrounded by spaces. The grep command makes it easy to match only specific patterns in the input stream, so now let's build a three-part pipeline that lists all files in the current directory, screens out everything that isn't from Aug, and then counts how many lines remain:

```
ls -l | grep " Aug " | wc -l
```

See how that works? You should be thinking that people with even a rudimentary grasp of the standard 20 or 30 Linux commands has a powerful interactive environment at their beck and call. You'd be right! (Note that if you don't have files that are old enough, you won't see the month name in the `ls -l` output. Move to an older directory, like /etc, and try the command again; odds are you'll find sufficient old files in that directory instead.)

You can even have a pipeline that has its final output saved to a file by simply adding a redirect to the end of the pipeline:

```
ls -l | grep " Aug " > files.from.August
```

And, with the use of the little-known tee command, you can even save a copy of the data stream in the middle of a pipeline too:

```
ls -l | grep " Aug " | tee aug.output | wc -l
```

Here we have the same output as earlier, but now a copy of the intermediate results are neatly tucked away in the file aug.output. (A helpful tip too: you can use `tee /dev/tty` and have a copy of intermediate output shown on screen, even though it's also being fed to the next step in the pipeline.)

Thousands of Linux commands are accessible from the command line, and all but a small percentage are easily added to a command pipe. Given that a typical command also has at least a half-dozen different options to change its behavior, you can get a sense for just how rich the command-line environment is and why so many Linux power users and administrators still eschew the GUI for most of their work.

Flow Control and the Test Command

The next building block with shell scripting is flow control. This is an essential ingredient of any programming language, from the obscure APL to the now-pedestrian BASIC. Fortunately, there are a number of flow control elements available for shell script programmers, ranging from the most rudimentary if-then-else-fi to the more sophisticated while-do-end and repeat-until blocks and switch-case-end.

To look at flow control, it's necessary that we detour for a few minutes and talk about one of the most important commands for shell script programmers: test. The test command often is the program that evaluates conditional statements and ascertains whether the result is TRUE or FALSE, obviously a key capability for any sort of conditional flow control.

Believe it or not, the test command is linked to the [command, which is why you can write conditional statements one of two ways, as exemplified here:

```
if test -f filename
if [ -f filename ]
```

Flexibility to power the enterprise.

intel inside
XEON™

From mail servers to databases, ZT X9000 series servers powered by the 64-bit Intel® Xeon™ Processor can run the full range of 32-bit applications and offer extended flexibility for your 64-bit needs. So you can create powerful, all-purpose IT infrastructure that enhances business agility – and the bottom line.

ZT Optimum 1U Server X9535

Intel® Xeon™ Processor 3.20 GHz
(2MB L2 Cache, 3.20 GHz, 800MHz FSB)
- Intel® E7320 Chipset Server Board
- 1GB ECC Registered DDR 333 SDRAM (Up to 16GB)
- 4 x Seagate® 250GB SATA Hard Drive (Total 1TB Storage)
- 4 x 1" Hot-swap SATA Drive Bays
- 4 Channel High Performance SATA Controller
  (RAID 0,1,5,10,JBOD Support)
- Slim CD-ROM & Slim Floppy Drive
- Integrated 2-port SATA Controller (RAID 0, 1 Support)
- Integrated Marvell 4-port SATA Controller (RAID 0, 1, JBOD Support)
- 2 x Intel® 10/100/1000 Gigabit Network Controller
- 1U Rackmount Chassis W/500W Cold-swap Power Supply
- 3-Year Limited Warranty

Free Shipping ! **$2,599**

ZT Revolution 2U Server X9543

Dual Intel® Xeon™ Processors 3 GHz
(2MB L2 Cache, 3 GHz, 64bit, 800MHz FSB)
- Intel® E7520 Chipset Server Board
- 1GB ECC Registered DDR 333MHz SDRAM (Up to 16GB)
- 1 x Seagate® 36GB 15,000rpm SAS Hard Drive (O/S)
- 3 x Seagate® 73GB 15,000rpm SAS Hard Drive (Raid 5)
- 6 x 1" Hot-swap SAS Drive Bays
- DVD-RW & Floppy Drive
- 64bit High Performance SAS RAID Controller
- 2 x Intel® 82541GI Single Port Gigabit Controller
- 2U Rackmount Chassis w/550W Power Supply
- Super Doctor III Server Management Software
- 3-Year Limited Warranty

$3,999

ZT Revolution 4U Server X9544

Dual Intel® Xeon™ Processors 3 GHz
(2MB L2 Cache, 3 GHz, 64bit, 800MHz FSB)
- Intel® E7520 Chipset Server Board
- 1GB ECC Registered DDR 333MHz SDRAM (Up to 16GB)
- 3 x Seagate® 73GB 15,000rpm SAS Hard Drive (Raid 5)
- 4 x 250GB ESATA 8MB Cache Hard Drive (Raid 5)
- 8 x 1" Hot-swap SAS / ESATA Drive Bays
- 64bit High Performance SAS / ESATA RAID Controller
- DVD-RW & Floppy Drive
- 2 x Intel® 82541GI Single Port Gigabit Controller
- 4U Rackmount Chassis w/650W
  Redundant-Cooling Power Supply
- Super Doctor III Server Management Software
- 3-Year Limited Warranty

$4,499

1. OEM Computer Manufacturer

- 3 year warranty with lifetime tech support
- Reseller and volume pricing available.
- Personal attention (Dedicated Technical Sales Team)
- Call now to customize using the latest technology

Find out how ZT Insider Program can help maximize your Business Solution

Go to **ztgroup.com/go/linuxjournal**

Call **866- ZTGROUP** (866-984-7687)   promote code : LJ0106

ZT GROUP

This particular conditional tests to see if filename is indeed a file (the -f test). If you use the more readable [notation, you are required to include the closing] symbol too; whereas, if you use test overtly, you can skip any closing symbol on a conditional.

Tip: using the [symbol has a second benefit. Many modern shells have a version of the test command built in to the shell itself, considerably speeding up shell script execution. Using the [symbol ensures you'll use the built-in version if available, but explicitly calling test means that you'll likely not have that performance enhancement when running your scripts.

The test command has at least 30 different options, and it's critical that you become familiar with them, so you can understand how to test two alphanumeric strings (for example, filenames) versus how you might test numeric values (file sizes) or even perform a bewildering set of file and directory tests, including tests for execute permission, nonzero size, whether it's a pipe or socket and many more possibilities. To begin learning more about this command, type `man test` in your terminal.

Armed with the test command, a standard if-then conditional is structured as shown:

```
if [ condition ]
then
        statement block if condition is true
else
        statement block if condition is false
fi
```

Oftentimes, you'll see programmers use a small shorthand by adding a semicolon, so that the first two lines are instead written as:

```
if [ condition ] ; then
```

Here's a quick example of how this might be used before I run out of space in this issue:

```
if [ -w . ] ; then
   echo "I can write to the current directory. "
else
   echo "I cannot write to the current directory. "
fi
```

As you can see, this offers a quick way to test whether you have write permission to the current directory. Type it in to an editor (vi or emacs, whatever you prefer) and save it in your home directory as dir.write.sh, then you can use `cd` to move to different directories and run this first shell script by typing `sh ~/dir.write.sh` to see whether you have write permission in that directory.

Out of space. Next month, we'll spend more time looking at conditional statements and flow control and start noodling on how to write a rudimentary blackjack game as a shell script. See you then!▪

Dave Taylor is a 25-year veteran of UNIX, creator of The Elm Mail System, and most recently author of both the best-selling *Wicked Cool Shell Scripts* and *Teach Yourself Unix in 24 Hours,* among his 16 technical books. His main Web site is at www.intuitive.com.

Single Sign-On and the Corporate Directory, Part II

Ti Leggett continues his series on building a secure corporate directory. **BY TI LEGGETT**

Welcome back! Last month we blasted through getting your single sign-on and corporate directory infrastructure up and running. In this article, we configure some Linux clients and Mac OS X clients to make full use of last month's hard work. We don't have as many pieces to investigate this time, but we still have a lot to cover, so let's get to it! Again, all the configuration files are available from the on-line Resources.

In this article, we explore how to connect Gentoo Linux and Red Hat Enterprise Linux (RHEL) v3 and v4, but most Linux clients should be about the same to configure, with minor differences. We also cover Mac OS X v10.4 (aka Tiger) client integration. In a later article, we'll explain how to configure Microsoft Windows clients to use the authentication and authorization system, since it depends on configuring and setting up the Samba package.

For single sign-on, Linux and Tiger clients need a Kerberos host keytab. We create one the same way we created other keytabs. For both Linux and Tiger clients, this keytab is kept in /etc/krb5.keytab.

Linux Client Configuration

Not all users, especially remote laptop users, will want or be able to have their machine incorporated fully into the Kerberos realm. Unless you have complete control of all the machines from which users will connect, you need to allow traditional password authentication. While this defeats some of the security of using Kerberos by having to transmit passwords over the Net, as long as you are aware of this as a sysadmin and take precautions to limit its use, you're no worse off than if you had not chosen to use Kerberos. Kerberos still has a few advantages over mechanisms such as /etc/passwd, NIS or LDAP storage of passwords. It's much simpler to enforce password rules with Kerberos than with the other mechanisms, and the storage of the passwords is more secure in a Kerberos database. Make sure to review "Centralized Authentication with Kerberos 5, Part I" by Alf Wachsmann (see Resources) on enabling Kerberos PAM authentication.

In "OpenLDAP Everywhere" by Craig Swanson and Matt Lung (see Resources), the authors touched on /etc/nsswitch.conf /etc/ldap.conf and /etc/openldap/ldap.conf. We're going to tweak these files to refine them for speed and security. First, let's look at /etc/openldap/ldap.conf. This file defines defaults for the OpenLDAP command-line tools, such as ldapadd and ldapsearch. Our /etc/openldap/ldap.conf file looks like Listing 1.

For more options and more information, refer to ldap.conf(5). Because there's no way to specify that we want to use StartTLS in /etc/openldap/ldap.conf, we must explicitly use

---

Listing 1. /etc/openldap/ldap.conf

```
BASE "o=ci,dc=example,dc=com"
URI ldaps://ldap.example.com ldaps://kdc.example.com
TLS_CACERTDIR /etc/ssl/certs
TLS_REQCERT allow
```

Listing 2. /etc/ldap.conf

```
host ldap.example.com kdc.example.com
base o=ci,dc=example,dc=com

ssl start_tls
tls_checkpeer no
tls_cacertfile /etc/ssl/certs/ci-cert.pem

nss_base_passwd ou=people,o=ci,dc=example,dc=com
nss_base_group ou=group,o=ci,dc=example,dc=com
nss_base_hosts ou=hosts,o=ci,dc=example,dc=com
nss_base_services ou=services,o=ci,dc=example,dc=com
nss_base_netgroup ou=netgroups,o=ci,dc=example,dc=com
```

---

an ldaps:// URL. Now we can simply issue `ldapsearch`, which uses SASL authentication by default and then refers to /etc/openldap/ldap.conf for the default host and base.

Let's now work on getting name service switch (NSS) working properly. Make sure the nss_ldap package is installed. Also, be aware that older versions of this package didn't handle LDAP storage of some services such as netgroups, so try to get the latest version you can. First configure the nss_ldap package itself by editing /etc/ldap.conf. This file differs from /etc/openldap/ldap.conf, which is only for the OpenLDAP tools, whereas /etc/ldap.conf is the configuration file for nss_ldap. Listing 2 shows what /etc/ldap.conf should look like.

Let's look at what these different lines do, as there is no man page for this file. The first two lines define the LDAP servers to contact and the base from which to search. The next three lines explain how to initiate the TLS connection. As you can see, nss_ldap is aware of StartTLS, so we use that method to initiate a TLS connection. The last lines describe the bases to search from for various nsswitch controlled attributes. We set these for performance reasons. There's no need to have to search the whole directory for a given user name if all of our user names are under one

branch of the tree. For example, nss_base_passwd specifies the base for searches traditionally associated with information in /etc/passwd. If, however, you do store user names in multiple branches of the directory, you can't use this option. A slew of other options can be defined in this file. To explore these options, check with the example ldap.conf that comes with the nss_ldap package.

Make sure you have the CA certificate in /etc/ssl/certs, and run c_rehash. This process needs to be done for any machine that will be contacting the LDAP server for any information over an SSL connection.

Next, edit /etc/nsswitch.conf to specify looking in LDAP for information. Don't put ldap as the first entry for hosts; if you do, you won't be able to resolve the LDAP server's hostname. If you had added a user locally to /etc/passwd or /etc/shadow that you wanted to be served out of LDAP, remove it or comment it out. Now you can test that it's working:

```
# getent passwd leggett
leggett:x:1001:100:Ti Leggett:/home/leggett:/bin/bash

# id leggett
uid=1001(leggett) gid=100(users)
```

If both of these commands work, you're ready to go. Some programs require a restart in order to recognize changes to /etc/nsswitch.conf. OpenSSH is one of these, so restart sshd and then attempt to slogin.

Everything we've done so far has led us to enabling Gentoo and RHEL clients, but let's review what needs to be in place.

The following files are needed for Kerberos authorization:

- /etc/krb5.conf

- /etc/krb5.keytab

- /etc/pam.d/system-auth

And the following files define OpenLDAP account management:

- /etc/openldap/ldap.conf

- /etc/ldap.conf

- /etc/nsswitch.conf

- /etc/ssl/certs/ci-cert.pem (Gentoo)

- /usr/share/ssl/certs/ci-cert.pem (RHEL)

There's one caveat for RHEL 4 at the time of this writing. A bug occurs when you use hostnames in /etc/ldap.conf instead of IPs, so have LDAP as a lookup for the hosts database in /etc/nsswitch.conf and use DHCP to get client IPs. If you notice that bringing up your network devices causes a segfault in dhclient, change hostnames to IPs in /etc/ldap.conf.

Configuring sshd for single sign-on under Gentoo and RHEL v4 is a snap. You need to make sure the following options are set in /etc/ssh/sshd_config:

```
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
UsePAM yes
```

Once you've set these, make sure to restart sshd.

Unfortunately, RHEL v3's sshd supports an older GSSAPI mechanism called gssapi, which is susceptible to a man-in-the-middle attack. It's been replaced by the gssapi-with-mic mechanism, which is what both RHEL v4 and Gentoo use. If you're unsure which mechanism your sshd supports, simply enable GSSAPI authentication in the sshd_config file and then attempt to ssh in using the verbose flag. You'll get a report of all the mechanisms that sshd supports. If you attempt to ssh from a client that uses one mechanism to a server that uses the other, you'll be prompted for your password. This is because the credential passing is done slightly differently, and incompatibly, with each mechanism.

Our ultimate goal is for users to have to type their password only once per day and never have that password go across the wire. So why go through all this trouble if your users are sending their password over the wire every time they check e-mail? Fortunately, more e-mail clients are starting to support the GSSAPI mechanism. Unfortunately, if you're a Mozilla Thunderbird user, you're out of luck (as of this writing). A few other options exist, however; for example, both KDE's KMail v1.8 and Ximian Evolution v2.2 have GSSAPI support. I've never used KMail, so I'll stick to what I know. Configuring Evolution to use GSSAPI is simple. Just select GSSAPI as the authentication type under both the Receiving and Sending Email tabs (Figure 1). If you set the Use Secure Connection option to Whenever Possible, Evolution uses StartTLS to secure the data transport.

Figure 1. Evolution 2.2 Account Preferences

Mac OS X Clients

Starting with Tiger, Apple has Kerberized almost all portions of the operating system. If you need to incorporate Panther (v10.3) clients into your infrastructure, contact me for information; a good deal is required. Tiger, however, is relatively easy.

Start by editing the file /Library/Preferences/edu.mit.Kerberos. This file is quite similar to its Linux counterpart, /etc/krb5.conf, with some very minor changes. Ours looks like Listing 3.

Listing 3. /Library/Preferences/edu.mit.Kerberos

```
[libdefaults]
 ticket_lifetime = 600
 default_realm = CI.EXAMPLE.COM
 default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
 default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
 dnsfallback = no

[realms]
 CI.EXAMPLE.COM = {
  kdc = kdc.example.com:88
  kdc = ldap.example.com:88
  admin_server = kdc.example.com:749
 }

[domain_realm]
 .example.com = CI.EXAMPLE.COM
 example.com = CI.EXAMPLE.COM
```

With Kerberos configured, the next step is to create a keytab for the host and place it in /etc/krb5.keytab. You can run kadmin from the OS X client, but because the version that comes with v10.4 is slightly buggy, you need to append the -O option:

```
#/usr/sbin/kadmin -p <admin princ> -O
```

That's it for Kerberos authentication with Tiger.

Unfortunately, at the time of this writing, a bug causes the machine's authentication subsystem to hang when a network user attempts to use sudo while having valid Kerberos credentials. I'm working with Apple to resolve this issue, so check back with me to find the solution.

Mac OS X does not use nsswitch for resolution of name services. Instead, it uses what it calls Directory Services. I explain how to modify Directory Services via a GUI called Directory Access, but ultimately the GUI makes modifications to two files: /Library/Preferences/DirectoryService/DSLDAPv3PlugInConfig.plist and /Library/Preferences/DirectoryService/SearchNodeConfig.plist.

The GUI utility can be found in /Applications/Utilities. First, enable the LDAPv3 plugin, then select it and click the Configure button. Once inside, click the Show Options drop-down, and then click New to define a new LDAP server. Enter your LDAP server's name, make sure all three check boxes are checked (Figure 2), and click Continue. Next, choose RFC 2307 (Unix) as the template, enter your LDAP base, click Continue and define a Configuration

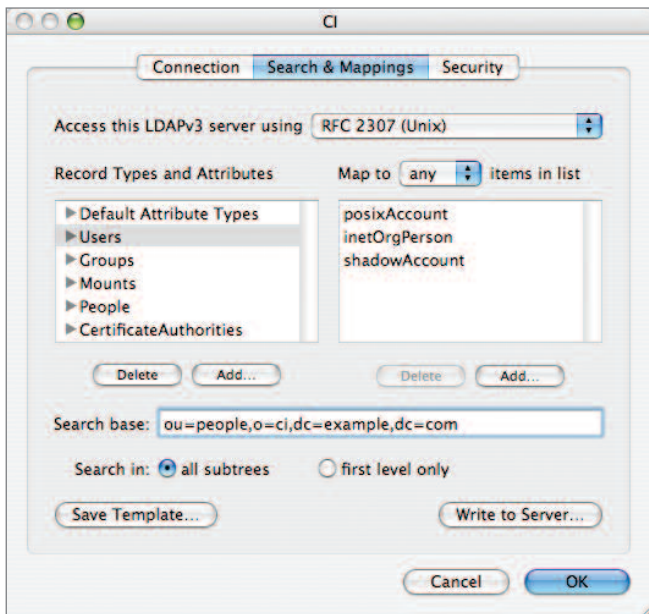Figure 2. Creating a New OS X LDAP Directory Connection

Figure 3. Refining OS X LDAP Searches

Name. That's it!

You can refine how Directory Services searches the directory, much as you can with /etc/ldap.conf under Linux. If you highlight the directory service and click Edit, it brings up some more advanced options. Click on the Search & Mappings tab. There you'll see a list labeled Record Types and Attributes. For each of those you can select and then define a more specific Search base (Figure 3). Click Ok twice, and then Apply to save your changes.

Of course, you'll want to verify that your directory changes are correct and working. OS X has a command-line utility, dscl, that is used to query not only an LDAP directory but also a NetInfo directory, NIS directory or any other directory listed in Directory Access. First, we should make sure we can query our LDAP server directly:

```
# dscl localhost list \
  /LDAPv3/ldap.example.com/Users
```

If you run dscl without any options, you are given usage instructions and left at an interactive prompt. Here are two more examples of using dscl:

```
# dscl localhost list /Search/Users
# dscl localhost read /Search/Users/leggett
```

Here we use the /Search directory, which acts on all enabled directories. So, if you have local users in a NetInfo directory and also LDAP users, the /Search acts on both of those directories, merging the results for display. The second example uses the read action to show the detailed information of the leaf of the branch specified, /Search/Users/leggett in this case. The dscl utility can be helpful when all you have is console access to your OS X machine.

Having verified that our LDAP users are available, we need to create local home directories for the LDAP users we've just enabled:

```
# install -d -o leggett /Users/leggett
# ln -sf /Users /home
```

OS X v10.4 has some finer-grained policies than standard POSIX access for controlling access to certain aspects of the OS. By default, members of the group admin have administrator privileges. However, this group is locally stored in each machine's NetInfo directory, and I've been told it's very bad to remove or rename this group. You also can't override this group with an LDAP group named admin, because the search order for directories always searches the local NetInfo directory first. To use an LDAP group to replace the local group role, you need to edit the file /etc/authorization. This is a standard Apple plist-formatted file, and it defines roles for different aspects of the system. If you change lines that look like:

```
<key>group</key>
<string>admin</string>
```

to:

```
<key>group</key>
<string>ldap-admins</string>
```

for the different rights, you'll enable those users in the ldap-admins group to be administrators on that machine. These rights are different from sudo privileges, which are defined in /etc/sudoers. These rights control rights to tasks such as installing software and modifying system preferences.

At this point we should be able to log in as user leggett. Tiger's sshd supports both GSSAPI mechanisms, gssapi and gssapi-with-mic. Previous OS X versions supported only gssapi, so a password was required when logging in to or from an OS X client. SSO support is enabled out of the box for sshd, so there's no config files to edit.

As I stated earlier, with v10.4, almost all of OS X's built-in services and applications are Kerberized, including Apple's Mail.app. If you're running your own CA or using self-signed certificates, you need to import your CA's certificate into the System keychain first, so Mail.app won't complain when connecting to self-signed SSL-enabled services like IMAP and SMTP. Copy the CA cert to the OS X client and then run certtool:

```
sudo certtool i ci-cert.pem v \
 k=/System/Library/Keychains/X509Anchors
```

Now you're ready to start Mail.app. The trick to enabling GSSAPI during the account creation process is to fill in the user name and leave the password blank. If you don't already have valid credentials, it will prompt you for your Kerberos password. Once the account is created, go back and enable SSL for IMAP. By default it's not enabled, and Mail.app doesn't give you the choice at account creation time.

All versions of OS X since v10.3 ship with a GUI application for managing Kerberos credentials named Kerberos.app (Figure 4), but it's buried in /System/Library/CoreServices. You can add this useful app to your dock and have it start at boot. It can automatically renew your credentials when they're expiring and easily show you how much longer your credentials are valid, among other useful features.

Many of Apple's services and applications are fully Kerberized, including Safari, VPN, Xgrid and AFP, making your Apple users and administrators first-class citizens in your network.

Figure 4. OS X Kerberos Application

Wrapping Up

By now you're probably starting to realize the enormous potential of LDAP directories and Kerberos authentication. You have a powerful and scalable infrastructure as well as clients making full use of it. In my next article, we'll discuss how to integrate in one more type of client, Microsoft Windows. Until then, enjoy the fruits of your labor!

Resources for this article:
www.linuxjournal.com/article/8636.

Ti Leggett (leggett@mcs.anl.gov) is a systems administrator for the Futures Laboratory of the Mathematics and Computer Science Division at Argonne National Laboratory. He also has a joint appointment with the Computation Institute at the University of Chicago.

Reclaim lost time

3:25 pm

◁ S M T W **T** F S ▷

Tasks > Daily

09:00-10:00	☑ Provision 50 new servers & desktops
10:00-11:00	☑ Migrate 40 old servers to new hardware
11:00-11:30	☑ Deploy entire application suite
11:30-01:30	☑ Long lunch
01:30-02:00	☑ Audit all changes made everywhere, anytime
02:00-03:30	☑ Conduct disaster recovery exercise
03:30-05:00	☐ **CATCH SOME RAYS**

NEW TASK   DETAILS   DELETE

Levanta Intrepid ™

The world's first Linux management appliance

Plug the Levanta Intrepid ™ into your network and perform the most important Linux management tasks in a fraction of the time you spend now. And gain power and flexibility that you've never had before:

- **Fast & Portable:** Provision servers or workstations practically anywhere, anytime – in minutes. Swap them around, mix it up.
- **Flexible:** Supports commodity hardware, blades, virtual machines, and even mainframes.
- **Out of the Box:** Includes pre-defined templates for servers, workstations, & software stacks. Or create your own.
- **Total Control:** Track any file changes, by any means, at any time. And undo them at will.
- **Disaster Recovery:** Bring dead machines quickly back to life, even if they're unbootable.

Based upon technology that's already been proven in Fortune 500 enterprise data centers. Now available in a box, priced for smaller environments. **Just plug it in and go.**

30-Day
Money-Back Guarantee
Order online by 1/31/06
Get $500 Off
Enter PROMO CODE: LJ0106

LEVANTA®
www.levanta.com
1.877.LEVANTA

WINNER
L∆NUXWORLD
CONFERENCE & EXPO
Most Innovative
Hardware Solution

Making IT Work

How one company and its customers survived
Hurricane Katrina—and the lessons they teach the
rest of us. **BY DOC SEARLS**

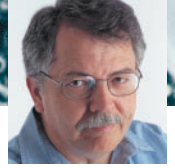**W**hen Hurricane Katrina slammed New Orleans and the Gulf Coast in August 2005, it quickly became the worst natural disaster in US history. After the levees broke and the Gulf of Mexico poured into New Orleans, law and order failed along with the civic graces we call infrastructure. As the waters rose, electric service went out, along with phone, cell, cable and tap water. TV and radio stations went silent. So did communications among police, fire and emergency services. Hospitals struggled to survive along with their patients. On the roofs of submerged houses and the shores of drowned parishes, citizens waited for help from services rendered just as helpless—while looters emptied stores and snipers shot at rescuers. For the next fortnight, New Orleans was Mogadishu.

Yet one beacon of infrastructural persistence stood like a lighthouse above the storm and the tides: DirectNIC, the domain name registrar and site hosting service operated by Intercosmos Media Group, Inc., a company founded by Sigmund "Sig" Solares, who is also its CEO. Sig was born, raised and educated in New Orleans, earning his BA, law degree and MBA from Loyola University there. He is also a determined and resourceful businessman with a loyalty to his company, his customers and his city that verges on the absolute.

In the course of keeping DirectNIC up, Sig and his crew also opened a lifeline to the world, serving as a firsthand source of zero-BS news and photojournalism while also providing much-needed technical help. The survival of DirectNIC is a triumphant story of open source in every meaning of the phrase. It may also be the best DIY-IT (Do-It-Yourself IT) story of all time: a shining example of how to survive a true worst-case scenario.

DirectNIC isn't a small IT potato. According to Netcraft (see the on-line Resources), DirectNIC is the world's #11 domain name registrar, with more than 400,000 customers and 1.1 million domains. It's also a large hosting service taking up 6,800 square feet on the 9th and 10th floors of an office building on Lafayette Square, close to the Superdome and the shores of the city's drowned districts.

Not coincidentally, DirectNIC runs on Linux. As soon as I saw that was the case, I began trying to get in touch with Sig. He finally found time to give me a 45-minute interview on September 7, 2005, while the disaster was still going on. (To place it in a timeline, the levees were still breached, the city was still under water, chaos still ruled and five days would pass before FEMA director Michael Brown resigned.) We did that interview as background for this column. We also made it a podcast you can download from IT Conversations (see Resources).

"When we first started out, we had to go with Linux because we didn't have the money for any other options", Sig explained. He continued:

> We hired programmers who were familiar with Linux. That's just the direction we went. It's made it easy to find employees. We use Red Hat. Everything of course is Apache, MySQL, PHP. We have 17 programmers, and we try to get everything done with PHP and MySQL. If fact, back when they were small, MySQL had a couple of meetings each year. They used our training facility and did their training here. We're big on hiring their training people to get our programmers up to speed.

Thus, what was already clear on the Web became vivid on the phone: Sig and his team are resourceful and determined in large and equal measures. "We have our own 6,800 square-foot data center. It was built for us by Enron, which went in to bankruptcy. We went to bankruptcy court and said 'We'll give you 5 cents on the dollar for it.' And they said, 'We'll take it.'"

As for dependencies, DirectNIC is closer with its customers than with its vendors:

> We've always bought HP servers. We used to have white boxes; but we had some problems, so since 2000 we've been using HP. We've been very happy with those servers. MySQL has always helped us with technical questions. We have a support package with them. But in this case, we really didn't need help from them. In fact, in this case, we didn't turn to any of our vendors. We actually turned to a couple of our customers who were in the area, and we needed diesel, so she went and bought a (Hummer) H1. She's like, "Look, I'm going to make sure you can get diesel in there. And you'll also have a cool vehicle afterwards."

As Katrina approached and orders came to evacuate, Sig made his position clear:

> I told all my IT guys, "I'm not asking anybody to stay. If you're going to stay, and want to come to the office, great. I'm going to be here." Myself, I stayed. Donny Simonson, a Senior Vice President—he's in charge of all our programmers—he stayed. All of our net ops guys left town. Then Michael Barnett, a friend of mine—he never worked for the company before—he wanted to stay at the office. He already had a Live Journal blog where he called himself the Interdictor.

In fact, Michael Barnett is a former Green Beret who clearly felt his experience and skills would come in handy. His title became Crisis Manager, and his early Katrina entries make clear that managing was exactly what he intended to do:

> Sunday, Aug. 28, 12:01 p.m. We're on the 10th and 11th floor of a corporate high-rise on Poydras Ave., right near St. Charles.

We have generators and tons of food and water. It is five of us total. I am not sure how the Internet connection will be affected. I have a camera and my gun.

Monday, Aug. 29, 5:12 a.m. Welcome to ground zero.

An hour later, Hurricane Katrina made landfall in Louisiana as a Category 4 storm with winds over 145 mph. That same morning, the 17th Street Canal levee was breached. And the Gulf of Mexico poured into New Orleans through Lake Pontchatrain.

DirectNIC also took a direct hit, as rain and wind broke through the windows of DirectNIC's facility. "You never expect to have water come into your data center when it's on the 10th floor of a building", Sig said. Yet, they were prepared. Sort of. Sig explains:

We're always looking for angles.

Any time we make T-shirts, it's a sure sign it's going to be a failed business project. Luckily we had a failed project so we had 2,000 of these T-shirts. They were perfect for sandbags....

First, we created a barricade of T-shirts. We grabbed buckets and mops. We also grabbed a wet-vac to suck up the water. So we had six or seven people who were doing nothing but sucking up the water. Meanwhile we could hear the hurricane break the glass behind the wall that we had....And any time the glass breaks, that's going to allow more water through the walls...and head toward our servers.

Today we have cardboard over the vents just to keep the air conditioning in. The storm shutters aren't going to keep the AC in.

The generator is on the 9th floor and we're on the 10th floor. We also have a 1,500-gallon diesel tank on the 10th floor. We have 120 tons of coolers for the air conditioning, and blowers, on the 11th floor.

DirectNIC could stay up only by keeping the generators going and the AC cooling the servers. Also by making sure their fiber optic connections to the Net were working. "We knew we'd lose OC-3s from time to time", Sig said. "So we had four OC-3s when we knew we could get by with only two." (OC-3 stands for Optical Carrier level 3, the highest currently provided. An OC-3 line carries three times the base OC-1 rate of 51.84Mbps, or 155.52Mbps.) He continued:

We had to shut a couple things down, but we had at least one OC-3 in operation all through the entire time....

We were relying on three different providers for four OC-3s. Two of them went down. The third was close to running out of fuel, so we had to find a way to get fuel to their facility. One of my customers got on a fuel truck with the National Guard and went down the street to make sure the fuel got to the facility....The provider told us they were going to run out of fuel if they didn't get it by Saturday. So...Brian got on the truck with the National Guard and made sure the fuel got to the building. It

was near the Superdome, where the flooding was pretty bad. There was no way we could take a truck there. You had to have one of the military high water vehicles to get the diesel there.

The fiber is all underground, but was fine. Most of the generators were above ground and were fine; but the problem was getting fuel to the generators. Our generator is on the 9th floor. Another one was a couple floors up. So the biggest problem was just getting the fuel to the building, and also getting past all of the military, to convince them that there was enough reason. You have to get approval from City Hall to get fuel moved where you need it.

Meanwhile, the absence of net ops workers put a squeeze on the small remaining staff:

During this whole process, we did have a lot of questions. And one of the problems we had was all of our employees who left....We couldn't reach them by cell phone. The main method of communication was through the blog, and through e-mail and stuff like that. We'd say, "Okay, here are problems that we don't know how to figure out because the guys who usually figure this stuff out are far, far away." So we were able to get a lot of help in that manner.

For example, "We were so worried that we would not get diesel for several weeks that we had to figure out, 'How can we lessen the load so that we can stretch out our diesel for as long as possible?' We got a lot of information on that, straight from comments in the blog."

Meanwhile, Michael Barnett's Interdictor became a virtual port in a real-world storm. While the blog served as a half-duplex one-to-any help line for DirectNIC, it also served as on-the-ground zero-BS source of Solid Information about what was actually happening in deepest New Orleans. Michael did most of the posting and had no time for small talk. Every post was a punch to the fat belly of the media beast that was caught no less flatfooted than the local, state and federal offices quickly proved to be. For example, this entry on August 30:

I keep being told that CNN and the *Slate* reported our "moods" as something other than upbeat. The city is falling apart, no doubt. The looting is rampant. Just take a look at the cam and you'll see them breaking into that hotel and taking everything. The water is still creeping in. But you know what? My team's mood is not negative. We're focused. We've got things that need doing and we're gonna get them done. That's all there is to it. We need diesel. We'll find some. We have people depending on us and we are not going to let them down. That's all there is to it.

And if that's not enough to put our situation in perspective, just remember we live here.

And, the next day:

If you're watching the cam you'll note that there are WHITE people, BLACK people, and HISPANIC people looting. It's interesting to note that I see no ASIANS looting, but I'll leave that observation to the sociologists to explain. Back to the point,

don't bring your PC bullshit to this blog. This crisis is not about race, but about inhumanity. Got it?

As a result, The Interdictor took on a media life of its own. On September 1, six days into the Katrina disaster, Michael wrote, "This blog was never intended to be some kind of springboard to fame. I had no idea it would be... well, be what it is now. It's nothing short of stunning...." That post alone had more than 500 comments. Many came in via links from the *Times* of London and other bigtime publications, as well as thousands of other blogs.

Yet media stardom did nothing to distract from DirectNIC's primary mission. Here's the first thing Michael posted on September 2:

Outpost Crystal has had a long day. Alpha Team is gonna grab a few hours of sleep. You guys are great and all your support is pure motivation to keep plowing ahead. We won't quit, I promise you. We're expecting today (Friday) to be our most physically demanding day so far. Sig and I will be pushing 18 55-gallon drums of diesel up a steep parking garage incline to the 9th floor generator.

And the next day:

The three guys you just saw on the Webcam are Jeff and Hank of Data Protection Services (one of our customers who swore to their customers that they would not go down because of this storm), and Doctor Tom, an anesthesiologist. These three men heroically found a way to deliver us fuel and supplies into this disaster zone. I say heroically, because the amount of effort it took them to coordinate a way to get a container and fuel and a route in to the city in the face of persistent danger on the streets was absolutely off the charts. Their customers need to know the lengths that these men went to in order to get the job done. I listened to their accounts and I was absolutely impressed by the initiative and resourcefulness of these guys.

The result, one month later (as I'm writing this, in early October 2005), is a growing body of wisdom about DIY-IT and both preparing and recovering from natural disasters. Sig explains:

We've been through hurricanes before. We said "Okay, we need to hurry up and get some diesel, and have diesel on standby, and have the standby tank completely filled." The first time we had only one pump. And a diesel pump is going to burn out, and you have to wait for it to cool off. So we learned a lot because we've already been around a couple of times. But nothing nearly this scale. So this time it was definitely a trial by fire.

We've done migrations before. But when we've migrated from one data center to another, the biggest migration we had was five racks. We did five racks across the country. This time we had a 6,800 square-foot data center. Things just end up so much more complicated. Especially when you throw in the fact that now most of your employees you can't reach. We did have seven to ten guys who do a lot of telecommuting work. That's one of the things that really helped us out. One of the owners lives in Naples, Florida. He handled all of the coordination of migrating stuff away from this facility. So he just worked around the clock for a week, starting with the most critical stuff and working his way down. We have a lot of customers who aren't worried about having a solution redundant across two data centers. They'll pay us $15 for a domain and we host it for free. So in those cases we don't provide two different solutions. So if this data center were to go down, they'd be down. But once we were faced with the prospect of the data center going away, we wanted to do everything we could to prevent that from happening.

I've just been focused on making sure everything has been under control.

Sig is as plain about his company's role as a news source as he is about its

role in the marketplace: "We've definitely tried to put out as much information as possible while doing everything that has to be done."

DirectNIC also worked to help communications inside New Orleans as well. Sig reports:

> There are so many different organizations. I've talked with Alabama police, Florida police, reserves....The communications between the different groups are not very good. [This is one] of the things we're trying to help. The city's phones (mayor's office and so on) are handled by one of our customers. So if our facility were to go down, the city wouldn't have any phones. NOPD lost so much of their capacity to communicate as a result of Katrina. We had a police officer who stayed with us and went 24 hours without hearing from command.

Exactly one month after I spoke with Sig, he published an open e-mail on the Interdictor, which he and Michael Barnett had transformed into the Survival of New Orleans blog, with a new URL: mgno.com. That e-mail contains such valuable and hard-won lessons that I can find no way to edit it down. Here it is:

> Recently I received an e-mail from a customer who was worried about domain names that he registered at DirectNIC because of

the problems in New Orleans. Let me make one thing perfectly clear: Every other business in New Orleans may fail; nevertheless, Intercosmos Media Group, Inc., operators of DirectNIC, will continue on successfully in New Orleans and/or elsewhere without interruptions to our clients.

New Orleans has never been known as a pro-business location. Despite this, we have thrived in the Crescent City. Prior to Katrina, we were one of the most successful businesses in New Orleans. A few days after Hurricane Katrina hit, only two offices had their lights powered on the main street (Poydras Street), in the heart of the CBD. Those two offices were Bell South and Intercosmos Media Group, Inc. It was no accident that we continued to operate.

At Intercosmos, we did nothing special to prepare for Hurricane Katrina. We had already prepared for dozens of prior hurricanes and we already had our disaster preparations in place. The following is a very short summary of how we have prepared for any catastrophe:

1. Require Excellence from Employees. A small excellent team can accomplish anything. If you have the right team, then you can survive and actually thrive during any disaster. We have spent years putting together the right team of employees to operate under any circumstance.

2. Built an Excessively Large War Chest. At Intercosmos, we have never borrowed money to facilitate growth. We always focused on increasing liquid assets while maintaining steady profitable growth. For this strategy, we just followed Bill Gates who likes to keep a large amount of cash on hand for a rainy day. Because of this we are well positioned to weather any storm or series of storms.

3. Secured a $3 Million Line of Credit. We secured this line of credit despite the fact that we had and still have no intention of using this line of credit. The line of credit is merely a safety net.

4. Lots of Insurance. We are insurance junkies at Intercosmos. Nevertheless, as a result of Hurricane Katrina and our preparedness, we will have fairly limited claims to make.

5. Telecommuter Plan in Place. Prior to Hurricane Katrina hitting, close to 50% of our employees had telecommuting agreements in place with the company. When any hurricane hits and employees evacuate, we plan for employees to telecommute through the use of Internet-based (VoIP) phones, wireless Internet and laptops.

After Hurricane Katrina hit:

1. Increased levels of redundancy. Now we have servers in four secure, distant facilities instead of just in two facilities.

2. Increased profitability and cash on hand. Our business does not rely on areas hit by Katrina. Well over 99% of our business comes from individuals and businesses outside of Louisiana. As a result, we were actually more profitable in the month after Katrina hit than we were in the month prior

to Katrina hitting.

3. We started to use an 8,000-square-foot building that we have in another state as a re-location work space for employees. Employees who cannot work in one of our buildings in New Orleans can work as a telecommuter or from our other office.

What can you do to help?

We are passionate about our city. We want to see New Orleans rise again. However, our fate is not tied to New Orleans. We are trying to raise awareness of problems in New Orleans because we want the rebuilding effort to succeed and not get tied down in politics as usual. If you would like to help us on this mission, we ask that you join in the conversations at the message board we launched at nola.us.

Sincerely,

Sigmund Solares Chief Executive Officer Intercosmos Media Group, Inc.

My wife divides the world into "savers" and "spenders". Her theory is that most of us are born as one or the other. She's a saver. I'm a spender. Our four kids include two of each. Phil

Hughes, our founder and publisher here at *Linux Journal*, is a saver. Near as I can tell, so is Linus Torvalds. After hanging out with Andrew Morton (the kernel maintainer that Linus calls his "right-hand man") on last fall's Linux Lunacy Geek Cruise, I'm convinced that he's a saver too.

So, clearly, is Sig Solares.

As my wife explains it, the difference between savers and spenders is debt. Especially credit-card debt. She didn't want to marry me until I carried no credit-card debt and began paying my bills in their entirety every month. I've done that for so long now that it gives me chills to think about going back to my old ways. I may be a born spender, but I'm not a practicing one. At least not when it comes to debt.

Linux, I believe, is the saver's operating system. The values of saving are also embodied in the development and application practices of every free software and open-source project.

Surviving disasters is about saving too. That's the lesson that transcends politics, race, class and every other issue that divides us. When the going gets tough and the tough get going, the ones that go farthest will be the ones who saved most in the first place.

Resources for this article: www.linuxjournal.com/article/8637.

Doc Searls is Senior Editor of *Linux Journal*.

Running Windows Games in Linux

Windows gaming on Linux has its ups and (mostly) downs, but there are viable options for some games. **BY DEE-ANN LEBLANC**

More than 300 games are available for Linux today. However, plenty more are out there that are available only to Windows users, and if there's one thing Linux users don't like to accept, it's that we cannot do something everyone else can do. To that extent, a number of products and projects exist that make it possible to play Windows games under Linux (and to run other Windows programs as well). A quick list of these options include the WINE Project, Win4Lin, CrossOver Office, TransGaming's Cedega, VMware and the simple fact that some games, such as many of those from Id Software (creator of *DOOM*, *Quake* and so on) actually have binaries available that let you run the game natively under Linux.

Let's take a look at each of these options, see how it fits into the big picture and how likely it is to run the types of games you like, keeping in mind that sometimes the games you'll be able to use are older games rather than newer ones.

Figure 1. *Bejeweled,* being played inside Windows XP under Win4Lin Pro.

WINE—The Free Solution

The WINE Project's software (see the on-line Resources) allows people to run MS-DOS and 32-bit Windows applications under Linux and the free BSDs. Whether WINE will run your particular game is heavily dependent on a number of factors. Because this is free software, your best bet is simply to give it a try and see if it works. Depending on which Linux distribution you're using, WINE may be difficult or simple to install. You may find that WINE is bundled in your distribution's core or external package repositories.

Those who don't have this option can go to the WINE site and click the Download button. You may find a prebuilt package for your distribution there. If there isn't a package available, follow one of the WINE Source Downloads links and then follow the instructions in the README file for how to build this program.

Once you have WINE installed, attempting to run a Windows program using this tool is actually not that difficult. First you need to get the program onto your system, whether by mounting the CD-ROM or DVD-ROM containing it, or by downloading or copying the software onto your machine. Then, if this is software you need to install, you need to determine which program launches the installation—this is typically INSTALL.EXE or SETUP.EXE. From inside the directory where the program lives, type `wine program`. For example, `wine INSTALL.EXE`.

More often than not, the installation program will launch properly. You might even be able to install the software fully using WINE. If that's the case, don't get too excited yet. The real test is whether the software will run. Programs installed inside of WINE are placed in ~/.wine/drive_c/Program Files/. You also may have launcher icons on your desktop. If you're starting the program by hand, navigate to the directory containing the program, and then type `wine program` just as you did before. So, if you were trying to see if you could get the game *Bejeweled* running, you might change to ~/.wine/drive_c/Program Files/PopCap Games/Bejeweled, and type `wine WinBej.exe`. This is the moment of truth. It either will run or it won't. If it does, rejoice! If not, you can search the Web for tips on how to get the particular game running under WINE, or you can try another tool.

Unfortunately, I've had very poor luck

Figure 2. *SimCity2000*, being played under CrossOver Office 5.0beta1.

getting games to run under WINE, personally—much of that is due to a refusal to spend four days hunting down every possible configuration tweak. I'm sure someone will take away my official Geek card for that. Frankly, I have better things to do. The closest I got this time was pulling out my *Classic Text Adventure Masterpieces* CD with old Infocom games. The installer worked, and the games would launch, but then they died. Ah well, it was a bit of nostalgia anyway.

Win4Lin—Running Windows inside Linux

Win4Lin is a commercial product (see Resources), and this section focuses on Win4Lin Pro, which allows you to install Windows 2000 or XP inside a Win4Lin framework, which is installed on a Linux system. Essentially, it lets you run a Windows machine inside a Linux machine. Although this product is meant primarily as a business tool, if you have it for "practical" purposes, then why not try using it to run games that you can't get to run in another way? Because this is a commercial tool with plenty of documentation, installation instructions and starting Win4Lin are left to the manuals.

Now, first off, on an older machine like an Athlon 1.2GHz with 1GB of RAM, don't even bother. The virtual Windows XP box functions at a snail's pace for just about any operation, even opening folders. Playing any games except for untimed casual ones (for example, *Bejeweled*, which did work for me) is essentially impossible due to the

performance hit—though you get this hit only inside the Win4Lin session, the rest of the machine functions normally. Mind you, one of the older games I always test under emulators, because I kind of miss it, is *SimTower*, and this is the first time I've gotten it to run. However, *World of Warcraft* couldn't start its install, because the installation program gave an error claiming it couldn't find a data file.

Note that to get *Bejeweled* to work, I had to copy the files off of the CD-ROM and install straight from my hard drive. The CD is both a data and music CD, and both Linux and Windows seem to get fussy with it from time to time.

CrossOver Office

CrossOver Office is, again, a commercial product designed for running "serious" Windows programs, but that doesn't mean you can't use this software at least to attempt to play games. CrossOver Office is available from the CodeWeaver's Web site (see Resources), and once you have it installed, you can convince it to try installing any Windows program that isn't on its list by launching its Install Windows Software tool and clicking the Install unsupported software button.

As with the others, don't get too excited if you can get a game installed. That doesn't mean it will run. For example, *World of Warcraft* does manage to install under CrossOver Office—mind you, the text on the installer's buttons is almost too tiny to read—but *SimTower*'s installer malfunctioned and wouldn't

work. I found an old *RISK* CD lying around and discovered to my amusement that it refused to install because it works only under Windows 95.

In fact, *World of Warcraft*'s opening movie plays under CrossOver Office as well, and the game starts and is able to start downloading patches. Although it crashed at this point, frankly I found it impressive that the software even got that far. That encouraged me to pull out something older. I tried *The Sims* but that made the CrossOver Office installer decide that it suddenly couldn't access the hard drive. *SimCity 2000* not only installed, but actually plays, albeit a bit slowly on this system.

VMware

Due primarily to the expense (VMware Workstation costs nearly $200 US), most people who use VMware are doing so for work-related reasons. However, again, if you have this virtual machine tool lying around, and your computer is powerful enough to run a VMware session quickly enough to play mainstream games inside without problems, then this is another avenue to explore. On the same machine in which Win4Lin with Windows XP crawled, Windows XP under VMware runs at perhaps twice the speed.

VMware so far is the only one out of the bunch that was able to launch the installer for *The Sims*. Not only could VMware install it, it could actually run the game.

Then I figured it was time for the big test, *World of Warcraft*. First I had to allocate more hard drive space to my VMware session, which involved figuring out how to get Windows XP to see and use the new drive (a process I did not find intuitive at all, not being a Windows XP user aside from the occasional screenshot). Once the installation was complete, I tried to launch *World of Warcraft* and was told that 3-D support couldn't be started, so I finally got around to installing the VMware Tools package, which is supposed to—among other things—improve graphics performance. Although some might claim that VMware doesn't support Accelerated 3D, this is actually no longer true. However, that support is "experimental", but it doesn't get much more experimental

Figure 3. *The Sims*, being played within Windows XP in VMware 5.

than this, so it's worth trying.

To turn on Accelerated 3D, it's important first to shut down the virtual machine. Once this is done, it's time to edit the .vmx file for the instance. The VMware documentation recommends adding the following three lines to the file:

```
mks.enable3d = TRUE
svga.vramSize = 67108864
vmmouse.present = FALSE
```

Once this file is saved and closed, go to the VMware window and select Edit→Preferences. In the Preferences dialog box, choose the Input tab, and click the Ungrab when cursor leaves window check box to remove the check mark. Doing so will make sure that your games don't run into confusion over the mouse pointer. Click OK to save the setting. With this done, it's time to bring the machine back up and try *World of Warcraft* again. The game detects the "hardware change" and offers to reload default settings. Unfortunately, it's not enough, though at least this time there was an obvious attempt to start the game.

Hoping that an update of the rarely used XP session will help, I submit myself

to the ritual of update, reboot, update, reboot and so on. Unfortunately, it wasn't enough. Too bad—*World of Warcraft* gets tantalizingly close to starting.

Now, can VMware use the half-and-half CD with *Bejeweled*? Yes and no. It recognizes that both parts exist and allows accessing the files, but it can't actually run the installer and doesn't seem aware of all of the files on the CD. The game does install and run though from files copied off of the CD and onto the hard drive earlier. *Bejeweled* will not run in hardware-accelerated mode either, so the experimental feature isn't quite there yet. Still, as it improves, the chance of being able to use VMware for higher-end Windows games does too. Hopefully by that point, however, there will be more mainstream games available for Linux natively.

TransGaming

I've saved the most appropriate choice for last among the Windows-gaming options, so this section can end on an upbeat note. TransGaming's Cedega product (see Resources) is essentially a subscription service where you pay a monthly fee for access to the latest versions of the program in binary form, the ability to vote on the games
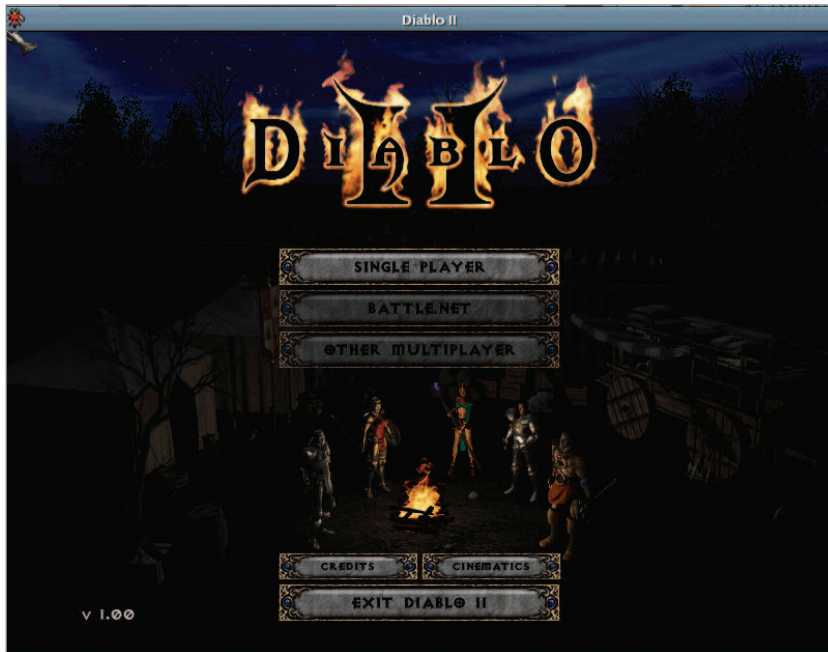
Figure 4. *Diablo II* in windowed mode through Cedega.

that you would like to see prioritized and more. Because this product focuses on games and implementing the DirectX APIs and other Windows features heavily used by game programmers, the likelihood of a Windows game working under Cedega should be better than under the other options. However, it is not guaranteed.

I'll give the Point2Play interface a shot even though I tend to have hit-and-miss luck with it. On Fedora Core 4, my CD drive shows up as /media/cdrecorder, and even when a disk is mounted onto the system, Point2Play can't see it—even though running the built-in Point2Play tests says that my CD drive is fine. A quick `ln -s /media/recorder /mnt/cdrom` fixed that problem. However, the Install button still doesn't become visible, so I gave up. There's no other fixes listed in the documentation that I haven't already tried.

There's no Cedega to install the program directly without bothering with the extra GUI. To do so, I mount the CD (in this case, the first CD-ROM for *World of Warcraft*) and change to its base directory. Then I type `cedega Installer.exe` and immediately get hit with a stream of errors. Going to the TransGaming Forums and running keyword searches doesn't help, so I post a query; we'll see what comes of that. I had *World of Warcraft* working under Cedega and Fedora Core 3 so I know it's doable.

Instead, I'll try *Diablo II*. Point2Play still won't see the CD, so I go to /media/cdrecorder and type `cedega install.exe` to launch the installer, and it launches just fine. When it gets to the video tests, it recommends Direct3D: DirectDraw HAL, so it does pick up the 3-D functionality on the system. The game also launches fine, though if I run it in windowed mode, I can't click on any other windows or it crashes. Again, none of the potential fixes I find on the boards helps with this.

Summary

There is no perfect solution for playing Windows games

under Linux. The best solution is to look and see whether a Linux binary is provided for the game, or to go find games that are written to play under Linux. Id Software and Epic Games both release Linux binaries for their games. Keep in mind that if you opt to use a solution such as Win4Lin or VMware, you have to own a valid copy of the Windows version you intend to use. Solutions such as WINE, CrossOver Office and Cedega implement the APIs without requiring the operating system to be installed.

Still, as you can see, there are many options if you are really determined to play a Windows game in Linux without having to dual-boot.

Resources for this article:
www.linuxjournal.com/article/8640.

Dee-Ann LeBlanc is the award-winning author of 13 computer books (mostly focused on Linux) as well as an award-winning technical journalist with more than 200 articles behind her. Her latest book is *Linux for Dummies, 6th Edition*, and you can learn more about her at **www.Dee-AnnLeBlanc.com**.

Creating a Home PBX Using Asterisk and Digium

Use Asterisk and Digium to give all your family members their own phone extension and voice mail.

BY JAMES TURNER

Some open-source projects have a certain perverse economic madness about them. For example, as I wrote about last month, you can use $600–$700 US worth of hardware and MythTV to re-create the TiVo box you can get at Best Buy for half the price. Of course, there are philosophical and technical reasons for wanting to use MythTV, but for most of the general public, it really doesn't make sense.

On the other hand, you occasionally run across an open-source product that is, in fact, a much cheaper solution than the commercial offerings in the same space. One example is Asterisk, which bills itself as the open-source PBX. A PBX, for those not in the know, is a Private Branch Exchange. It's essentially a mini-phone company inside a business or (as in our example here) a home. A certain number of outside lines come into the building, but there can be many more individual extensions that can call each other or request an outside line. Anyone who's worked at a company of even moderate size should be familiar with the concept (dial 9 to get an outside line—ring any bells?).

Unfortunately, as handy as PBXes are, they are also mucho dinero. Thankfully, Asterisk can provide a full-function PBX with voice mail and even VoIP bridging, without breaking the bank. It's cheap enough that it even makes sense in a residential setting, assuming your home is large enough that you spend half your time shouting at the spouse to pick up the phone. With that in mind, let's see how you might set up Asterisk in a home environment.

To begin, you'll need a garden-variety Linux server with a spare PCI slot. Asterisk has been reported to have performance issues running with the X Window System, so you'll probably want to dedicate a server just for the purpose. You don't need an absolute speed demon of a box, but low-balling your processor probably isn't a great idea either. You'll need some disk space to store voice mail, but Asterisk is fairly efficient at compressing sound, so a 40GB drive should be more than enough for the OS, swap and voice data.

Phone Calls Are in the Cards for You

You will need one piece of specialized hardware, a Digium Wildcard TDM400P. This card allows you to hook traditional handsets and POTS (plain-old telephone service) lines to your server. It's what Asterisk uses to receive and send calls over the
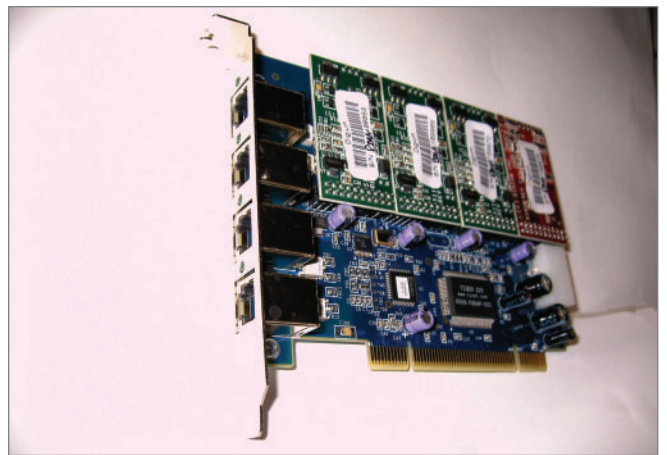

Figure 1. The TDM400P provides four ports to mix and match between external lines and extensions.

phone network, and to ring the various PBX lines inside the house. A TDM400P has four available ports—you can mix and match them between outside lines and internal handsets. So, for example, you could configure it to have one outside line and three internal lines. You need to order the card with the mix you want; you can't change a line from outside to inside. Fully loaded, the cards go for about $250 US retail. Basically, to figure out how many you'll need, add up the number of outside lines you have or intend to get plus the number of internal extensions you plan to provision (you can have more than one handset on a single extension), then divide by four. So, two outside lines plus six extensions would be eight ports, or two cards. In a corporate setting, you'd probably be bringing in your outside lines via a T1 line, and Digium makes cards for that purpose as well, but since you're probably not running a call center out of your Tudor mini-mansion, the TDM400P will probably be all you need.

At this point, you need to set up your OS and the Asterisk software. I know the almost irresistible temptation is to install your favorite distro and then try to install the Zaptel drivers needed for the TDM400P and the Asterisk software from source. Resist, resist! I spent a good three days trying to get things working that way. First I couldn't get the drivers running right, then I had to configure udev support for those Zaptel

devices, then spend a good day poring over the available documentation for Asterisk trying to get the software to talk to the card. I even had an engineer from Digium logged in to my system poking around for an hour, and he couldn't get it to work either. (This, by the way, is a really nice service offered by Digium to folks who purchase their cards; they'll assist in the setup and basic configuration of an Asterisk system.)

Feeling @Home with Asterisk

Finally, I saw the light (possibly stars brought on by repeated pounding of my head on the desk), and downloaded the ISO of Asterisk@Home, a complete Linux distribution that comes up running right out of the box. An initial caveat: the install procedure is extremely automatic, to the extent that it will repartition and reformat your disk without a second glance. This is definitely not something you want to "trial install" on a machine with important data sitting on it.

Once the install is finished (about 20 minutes on a fast machine), you should be able to browse (from another machine) to a newly installed Web server on your Asterisk host. Clicking on the Asterisk Management Portal link brings you to the main AMP screen. From here, you can set up your entire PBX using a well-documented and simple-to-use
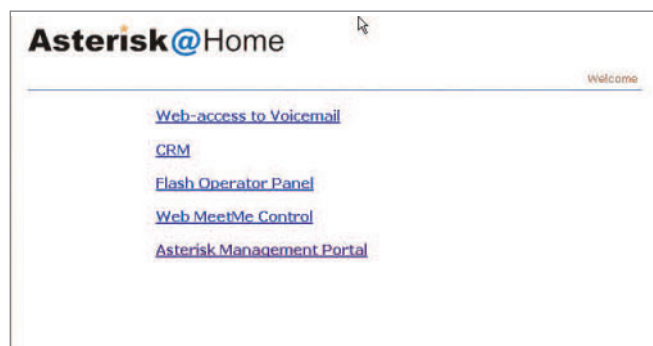

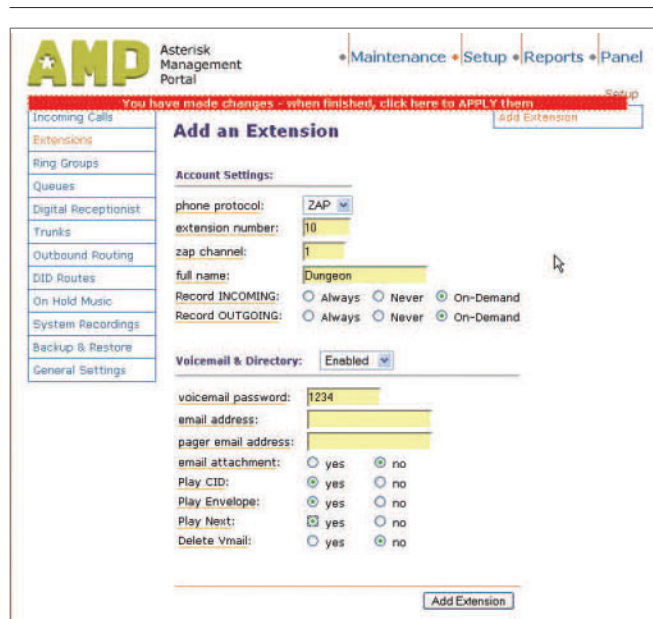Figure 2. The Asterisk@Home Main Screen

Figure 3. Adding an Extension

Figure 4. Configuring Incoming Calls
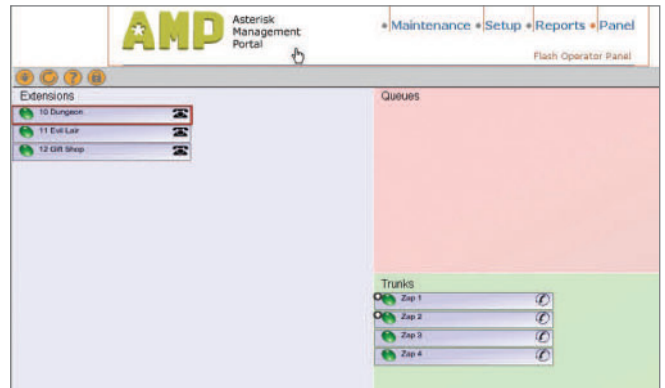
Figure 5. Using the Operator Panel, you can drop and drag phone calls.

GUI interface.

For example, adding an extension is as simple as a couple mouse clicks. In Figure 3, we're associating port 1 on the

Digium card with extension 10. We also can configure the voice-mail options at the same time. Recording automated voice response messages is just a matter of calling an extension from one of the system phones and talking.

Incoming calls can be routed to a set of phones or to an operator. You also can set different rules for business and after-hours calls. Certain lines can be restricted from making long-distance phone calls or even from calling outside the house at all. Asterisk@Home also comes with an application called Flash Operator Panel, which lets authorized users see which lines are in use, transfer calls to different extensions and generally perform the functions normally handled by a receptionist at a company. It's probably overkill for a simple home application, but it comes as part of the package and may prove useful on occasion.

The Bottom Line

Asterisk can seem like a foreboding project to undertake, if you make the mistake of reading the manual and trying to do it yourself. But with some relatively inexpensive hardware and the right Linux distribution, you can have a home PBX screening your calls in less than an hour.

Resources for this article: www.linuxjournal.com/article/8633.∎

James Turner is Product Review Editor for *Linux Journal*. He has written two books on Open Source Java development and is a Senior Software Engineer with Axis Technology, LLC.

Linux Video Production: the State of the Art

From box-office hits to home movies, Linux is ready, set, action. **BY DAN SAWYER**

From the operating system that gave you *Lord of the Rings*, *Spiderman* and pretty much every other film that's made these days, comes the Holy Grail of Linux computing: the home and small office video production pipeline. Unlike the object of Monty Python's ill-fated quest, however, this long-sought treasure actually exists and is finally within reach.

The problem has been one of great frustration for many potential adopters: what do I do about my home videos, my hobby films, my presentation videos? These days, even the greenest newbies can bumble their way through assembling a video in the tinkertoy-like Windows Movie Maker, and Mac users are even better off. They have a proper consumer video editing application: iMovie.

Linux has, for years, been growing into a major standard in the Hollywood studio pipeline. A cursory glance at the CinePaint Web site reveals a catalog of films that could not have been produced on time or budget (if at all) without a Linux infrastructure. Without Linux, the *Star Wars* prequels would have had a different look, and there would have been no *Lord of the Rings* trilogy. From its early days on the render-farms to its coming of age in *Shrek*, Linux has proved time and again its effectiveness, power and stability to the major studios.

None of which does the hobbyist user much good. Typically, filmmaking software for Linux is dearly bought; Eyeon's DFX compositing system, the most reasonably priced of any professional Linux-based compositing systems, weighs in at a hefty $1,295 US per seat, and it's the least capable player in the field. The home producer historically sits in the worst possible spot: atop a free and exceedingly capable operating system that was powering the major motion picture studios, without any way to get in on the fun short of writing a lot of scripts and working from the command line a great deal.

During the last three years, that situation has changed. Although the situation is not yet ideal, the FLOSS end of the Linux universe now allows for a near-complete, end-user-oriented production pipeline. I know, because during the last four years, I've built up a production studio running entirely on Linux—with one gap in the pipeline—and I make my living using it.

Let's say you want to edit your brother's wedding video to produce a short film with an alien sitting in the audience as a guest. In the Windows and Mac worlds, several companies offer end-to-end commercial solutions that take you from

acquisition through delivery-format authoring, covering (in no particular order) editing, titling, compositing, color correction, sound/score sequencing and DVD mastering. 3-D graphics applications are easily obtainable from a variety of other companies, and for a variety of prices. However, the companies producing such software will continue ignoring the Linux market until their high-end customers force them to produce Linux versions of their software (as happened in the 1990s with the IRIX version of Photoshop).

Although it would be nice to have such tools, we're not going to get them. And, I can finally say with confidence, the day approaches when we won't need them, either.

To get a home or small-office production studio running, you need most of the pipeline shown in Figure 1.
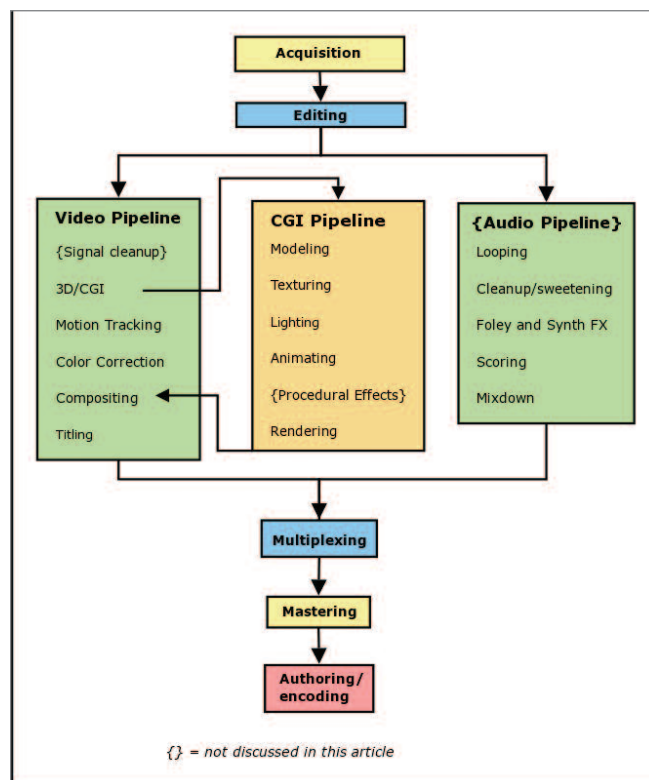
Figure 1. Production Pipeline

Not everything will be needed for every project, and indeed most projects will not approach this level of sophistication. Nevertheless, our Alien Wedding Guest project will give it a thorough workout, enough to show the weaknesses in the Linux pipeline as well as its strengths. To that end, we do an overview of the basic post-production process.

Acquisition—Kino, Cinelerra, XFS

When choosing your acquisition format, your originating medium can be just about any sort of video or film (see the Acquisition Means and Formats sidebar).

Because the focus here is on home and small business, it's safe to assume that the most common format for the next couple of years is going to be MiniDV. Getting the footage into your computer is fairly straightforward—DV comes in over the 1394 (FireWire) port, which is well supported in all modern kernels, and dumps straight to the hard drive. Be sure you have plenty of disk space—DV clocks in at approximately 12.5GB/hr. In either case, either Kino and Cinelerra will capture for you neatly, and variety of command-line tools exist as well.

There is a major technical issue to keep at the front of your mind: video compression. Pretty much any video you import will have undergone a compression pass. Edits don't usually require a second pass, so this isn't much of a problem if your plan is to do a quick-and-dirty edit, perhaps throw in a title, and output your video again. But the moment you start changing the video—effects, transitions, color correction, CGI—it has to be recompressed. Just as happened in the analog world when dubbing tapes from copies of copies, generation loss is a palpable problem. The cardinal rule, then, is to minimize further compression passes on your footage. If your system is properly managed, you can usually get away with only one additional compression pass; the one that you do on final authoring of the project for its delivery format.

You also will need to look to your filesystem. Simply put, video takes a lot of space, and it requires a high level of disk throughput and processing power. The faster your filesystem, the better the performance in your pipeline will be, and the fewer dropped frames you'll have. At the time of this writing, XFS is the fastest for this type of work, although the next generation of ReiserFS promises to give it a run for its money.

Editing—Kino, Cuisine, Cinelerra, Diva, Blender, MainActor

Once imported, you'll need to edit your footage. Take heed: there's more to editing than simply setting clips in order. The editing process involves timing decisions and asset management (footage selection, logging and commenting). All these are essential if you want to have a workable setup and tell a coherent story. There's nothing worse than being caught in the middle of a project and not being able to find that one particular shot that'll make or break the scene. Your footage organization (which can be as simple as a well-ordered directory structure) should be obvious, scalable and flexible. It'll need to grow with you, so put a bit of thought into how you want it set up at the outset.

The choice of editor is going to be determined both by the format of the footage you're working with, and by the way you structure your work. Stability, performance, interface and interoperability all need to be weighed.

At the time of this writing, the Linux editing field is in a state of dramatic upheaval, caused in no small part by Google's Summer of Code. By funding the FFmpeg/Blender integration project and the gstreamer-based Diva, they've put major cracks in a dam already fixing to burst. By the time of publication or soon thereafter, there should be five viable FLOSS choices for editing platforms: Kino, Cuisine, Cinelerra, Diva and Blender (Table 1). Of course, if you're really wanting a commercial solution for less than $2,500 US/seat, there is always MainActor, a program that amazes all who use it with its continued success as the Ford Edsel of video editors.

At the time of this writing, Diva looks promising, but isn't releasing proper packages yet; Blender is in the midst of a major development cycle to bring its editor up to par; Cinelerra is incredibly full-featured but barely usable, due to serious interface design and stability problems (any program that proudly lists, in its version 2.0 changelog, "Fewer lockups when resizing video window" does not make it into my toolbox).

3-D Computer-Generated Imagery (CGI)—Blender, The GIMP

In our imaginary Alien Wedding Guest project, we're going to need some fairly sophisticated CGI: a spaceship hovering in the background and an alien being of some sort sitting in the audience or standing off to one side seems a bare minimum.

Program	Formats	multitrack	Asset management (1-10)	Open format EDL	Stability (1-10)	Workflow (1-10)	Effects (1-10)	Speed
Kino	DV	N	4	Y	9	9	7	10
Cuisine	DV	Y	10	Y	9	10	3	7
Cinelerra	Quicktime and others	Y	7	N	2	6	10	7
Blender	FFMPEG	Y	2	N	10	6	5	8
Diva	gstreamer	?	?	?	?	?	?	?
Mainactor	AVI/MPEG	Y	6	N	4	4	10	6

Table 1. Editing Platforms

linux phone systems 101

lesson one
... choose the right team

The first step in any project is assembling the right team for the job. Let the experts at Telephonyware guide your Linux™ phone system project by helping you select the best hardware and software, and by providing the very best in service and support.

Take the guess work out of VoIP, choose a partner you can trust — Telephonyware.

lesson two
... get the right gear

For your VoIP project to be successful, you need the right gear! Let Telephonyware take the worry out of selecting the right hardware and software for the job.

We sell and support a full range of IP phones, analog and digital telephony cards, analog telephone adapters (ATAs), power over ethernet midspans and switches, and many more quality products. Our range is hand picked from the best manufacturers, and our helpful staff have used every product we sell.

lesson three
... put it all together

When it's time to turn plans into reality, Telephonyware is the right partner to take you from idea to completion. Our network of service partners, and excellent in–house support, give you the confidence you need, at a price you can afford.

Whether you're an experienced consultant deploying VoIP for your customers, a business replacing a phone system, or just looking for an IP phone or an ATA, Telephonyware will help you put it all together.

POLYCOM®

Telephonyware sells, supports and recommends the full range of Polycom IP phones.

Polycom's range of IP phones have been engineered to deliver a superb communications experience. They provide outstanding sound quality, advanced functionality, ease of use, simplified configuration and upgrades, and protection of your investment over time.

The Polycom VoIP portfolio includes the SoundPoint® IP family of desktop phones, an attendant console based on the SoundPoint IP 601 and Expansion Module, and the SoundStation® IP 4000 conference phone.

Both the SoundPoint range of desk phones, and the SoundStation range of conference phones seamlessly integrate with your IP PBX or softswitch application.

The phone's intuitive user interface offers dedicated, single button access to common telephony features. The high-resolution display delivers content for call information, multiple languages, directory access, system status and future applications.

The entire range of Polycom phones and accessories can be purchased by calling us, or directly though our web site.

Telephonyware is proud to be a Polycom Certified Channel Partner.

CERTIFIED RESELLER
POLYCOM
VOICE VoIP
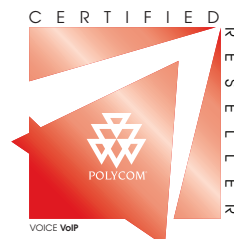
... visit **www.telephonyware.com/polycom** for more info

TELEPHONY WARE

For online orders or more info, please visit us at **www.telephonyware.com/lj**

Call us on **(866) 864-2304** or write to **sales@telephonyware.com**

Constructing these objects in 3-D space (modeling), adding color and bumps (texturing), animating their movement, and rendering out a finished video for blending with the original wedding footage covers the basic 3-D pipeline, as can be seen in our initial flowchart.

There are a variety of FLOSS 3-D programs available for Linux, most of them highly capable, and most of them require a good amount of scripting to get working in a pipeline. One program stands out far and away from the pack in features, interoperability and usability: Blender.

Where programs like POV-Ray are essentially renderers that depend on scripts or external programs for their grist, Blender is an end-to-end 3-D solution that plays nicely with a variety of renderers and file formats. In terms of capabilities, Blender aims to be an open-source Maya, and in the last 18 months, it has made amazing strides to that end with no sign of slowing down.

One of its most significant assets is its user base. Among the video and graphics applications available for Linux, it stands alone—far beyond even The GIMP—in the energy and vitality of its community. Tutorials and helpful forums abound, making its professionally oriented interface quickly accessible for all the basic tasks involved in small projects.

The GIMP also plays an indispensable role in the 3-D process as a texture creation system. Creating bump, reflection, specular and dirt maps from photos or painting texture layers from whole cloth, there isn't anything on Linux that even comes close in terms of versatility and raw power. What's more, by the time this article is published, GIMP modifications should update Blender textures in real time.

Motion Tracking—Voodoo

Motion tracking is matching the movement of computer-generated elements to your real-world footage. It's also used for stabilizing shaky handheld video. A number of motion trackers have come and gone for Linux, and until an open-source solution is developed, this will likely remain the case, as university research projects are bought up by commercial interests for development and sale to movie studios. At the moment, the best and only player on the field is Voodoo, put out by Digilab at the University of Hanover. It has the added advantage of outputting tracking data to a Blender-readable script. In practice, the export leaves a lot to be desired, but fortunately Ian Gowen has written a conversion utility that translates Voodoo's clumsy export into a very clean script for Blender (www.blenderwars.com/downloads/voodoo_convert.py).

Voodoo is not a perfect solution; it seems particularly to confuse slow zooms and slow rotation, but despite its minor defects, it's still a capable little program that's fairly well documented and licensed for free use by all. I don't doubt that sometime in the future it will be commercialized, but until then we have it to do with as we please (alas, we cannot have the source code).

Color Correction and Compositing—Jahshaka, Cinelerra, Blender, ImageMagick, CinePaint Glasgow, Mattelab

Once you've edited your footage, nailed your CG and tracked your camera, the time comes to marry them all together. Unfortunately, this is where we hit the major hole in the pipeline. There is, simply put, no straightforward way to do compositing on Linux in any straightforward sense. A well-outfitted, if basic, compositing system would provide for HSV adjustment, level calls, histograms, curve adjustments and brightness/contrast controls for color correcting the footage. A *really* competent system also supplies rotosplines for animated masking and two or three different keyers operating in different color spaces (essential if you want to do blue screen work with MiniDV). Finally, any compositor needs to do multilayer overlays, and do them well.

Our Alien Wedding Guest certainly requires this; sitting an alien in a chair with his spaceship hovering in the blue sky behind a tree requires taking rendered 3-D animation footage of the alien and his ship, using rotosplines to create a foreground mask for the chairs in front of the alien, and using the color keyer to treat the blue sky as a blue screen, laying the spaceship in behind the tree. You then apply color adjustments to each layer so that they all blend together nicely.

As you can see, it's possible to cobble together a more-or-less full suite using the various tools, but it's not a pretty sight. However, this is changing.

Blender is slated for a full-compositing system implementation for Project Orange, which requires Blender to become a full-fledged high bit-depth editing and compositing system; but that isn't due until February 2006 or so. Jahshaka is introducing color keying and splines into its RC2 release, in the fall of 2005/winter of 2006. CinePaint Glasgow's first release is scheduled for December 2005. And MatteLab may soon be extended and developed into a CinePaint or Blender plugin, as well as getting even better keying features.

All of these projects display great promise, and some of them will bear fruit this year, but at the moment, there just ain't no way to pull this off without doing a fair bit of coding, linking GIMP with ImageMagick with FFmpeg and working

Program	Color correction (1-10)	rotosplines	Keying ability	Versatility (1-10)	Stability (1-10)	Workflow (1-10)	Effects (1-10)	Interface (1-10)
Jahshaka	6	N	0	6	9	6	6	8
Mattelab	0	N	8	1	10	10	3	9
Cinelerra	9.5	Y	6	10	2	6	10	5
Blender	6	N	6	8	10	8.5	5	7
Cinepaint Glasgow	9.5	?	?	?	?	9	10	8
Image Magick	10	N	8	10	10	4	10	Command line

Table 2. Compositing Tools

ACQUISITION MEANS AND FORMATS

Linux's video pipeline can handle just about anything you throw at it, although some formats will limit your choice of editing software. The available formats include, but are not limited to:

■ Standard Definition: DV25 (MiniDV and Digital8): the most popular consumer and prosumer format both for its convenience and the quality of its image in the higher-end cameras; its compression makes some compositing work, such as color keying, problematic (though not impossible). MiniDV is captured through the 1394 port.

■ Beta/Digibeta: common in broadcast, rare everywhere else, this format is the granddaddy of TV video. Beta is captured through an analog tuner card (V4L-compliant with 720x540 capture capability), DigiBeta comes in through SDI and requires specialized capture hardware for optimal signal preservation.

■ VHS and derivatives: analog consumer mediums, lower resolution than MiniDV, must be captured through a V4L-compliant tuner card.

■ High Definition (HDV): this is the new consumer hi-def format being sold by Sony and JVC on some of their prosumer cameras. They record to MiniDV tape a highly compressed MPEG-2 signal having roughly the same bitrate as standard-definition DV25. It captures over 1394 or USB 2.0. At this

point, it is unclear whether this format will be very useful or have any meaningful advantage over progressive-scan SD miniDV.

■ Hi-Def: the CineAlta and other high-end motion picture cameras are Hi-Def platforms. Capture is accomplished through Hi-Def capture cards of various sorts, available through Linux Media Arts (lmahd.com) and Specsoft. This format requires special high-speed disk arrays and capture software, both of which are available through Specsoft. Look out though—the price on the cameras begins at $50K for the cheapies.

■ Film: 8, 16, 35 and 70mm and all their various incarnations. The old-fashioned chemical emulsion is expensive, but can't be beat for charm and (in some cases) for look. Film can be brought into Linux in one of three ways: 1) Telecine: projecting a film for recording by a video camera, after which capture proceeds according to the dictates of the video format. 2) Home-brew film scanning: using a DigitalSLR, the film is recorded frame by frame. Some creative electronics work can automate the process. Time consuming, but it does give you 10-bit or better color depth. 3) Professional film scanning: done with very large machines, at very high prices, in very large buildings, which are usually located quite a long way away.

mostly through the command line. This approach is extremely powerful, but it's far from ideal for the hobbyist.

Titling—Jahshaka, Kino, Blender
Titling, from the simple to the complex—anything from a quick fade-in/fade-out in Helvetica to a complicated short film of a title sequence—is the final step in the video part of our pipeline. Complicated title sequences of the sort seen in *SE7EN* are technically the domain of motion graphics software rather than simple titling programs.

If you want to deal with complicated motion graphics without going all the way and just creating your title sequences in Blender (which can be fun, but is rather like swatting a fly with a sledgehammer), Jahshaka is pretty much where it's at.

If simple fade-in/fade-out titles with dropped shadows are more your forté, you'll be perfectly happy dealing with Kino's DV Titler plugin.

Mixdown and Multiplexing
Once your effects are all done, and you've gotten your soundtrack mixed and timed the way you want, it's time to mixdown to the tracks that you'll marry back to your video stream. Assuming that you're going back to DV, it'll need to be a stereo mixdown (as indeed one audio track should be even if you're going to a surround-mixed DVD for final output), which can then just be dubbed back in your editor of choice. Once your mixdown is done, you're ready to marry the audio back to the video (multiplexing). This is accomplished in your video editor by lining the track up with the video and overdubbing the original audio.

Full discussion of the sound tools available for Linux can be found in the excellent articles by Dave Phillips in

DISTILLATION

Here, then, are the programs discussed, with the ones that are used regularly in my own studio pipeline in boldface.

- ■ Acquisition: **Kino**, Cuisine, **Cinelerra**, Diva, MainActor

- ■ Editing: **Kino**, **Cuisine**, Cinelerra, Diva, Blender, MainActor

- ■ 3-D Computer-Generated Imagery (CGI), modeling/texturing/animation/rendering: **Blender**, **The GIMP**, YAFRAY

- ■ Motion tracking: **Voodoo**

- ■ Color correction and compositing: **Jahshaka**, Cinelerra, **Blender**, ImageMagick, **GIMP**, CinePaint Glasgow, **MatteLab**

- ■ Titling: **Jahshaka**, Kino, **Blender**

- ■ Looping, cleanup, sweetening and FX editing: **Audacity**, **Ardour**, ReZound, Sweep

- ■ Music: **Rosegarden** (scoring) and **Ardour** (recording)

- ■ Authoring: **QDVDAuthor**, kencoder, **konverter**, Gmencoder, **Kino**

A FEW THOUGHTS ON SOUND

A video pipeline deals with a particular sort of software, but it's not the whole story when it comes to media production. The process of sound design for video is at least as involved as the video side of the equation.

In brief, you'll need a sound effects editor, a synthesizer, a mixer, a recorder and a scoring program. I've found the greatest productivity out of the fewest number of programs using Audacity, Ardour and Rosegarden in tandem, while keeping a handful of synths (like TerminatorX) around for use in a pinch. The process of sound production for a video or motion picture is a story in itself that deserves telling, but not today. In the meantime, a full discussion of the sound tools available for Linux can be found in the excellent articles by Dave Phillips in past issues of *Linux Journal* (the September 2004 issue has a good technical overview of many contenders, and the September 2005 issue has an extensive article on Rosegarden).

past issues of *Linux Journal* (there are far too many to enumerate here, but a search at the *Linux Journal* Web site will yield good results). For our purposes here, remember that close attention needs to be paid to preserving sound sync. Make sure that your finishing dialog track is timed the same way your source track was, and you'll be most of the way there.

Mastering

Using your editing program, print the video back to the format that will result in the fewest possible compression passes. If your originating medium was DV and you've done everything properly to this point, that should mean you have a total of one recompression on your footage and you can go out to a DV deck with little noticeable generation loss. Even better, take your multiplexed file and back it up, un-recompressed, to a tape or optical backup system.

Authoring—QDVDAuthor

Once you've multiplexed, you can use QDVDAuthor (see my article in the December 2005 issue) to author a well-tricked-out DVD for delivery to your friends and/or customers. Or, you can use one of the variety of front ends for mencoder, mjpegtools, or FFmpeg to compress your video for Web delivery or storage on your home machine. Kino includes such a front end, and other good ones include kencoder, konverter and Gmencoder. Whether to DVD or VHS or for Web delivery, this is the end point of the pipeline.

So, in short, it is now possible, with a little work, to get a competent and usable end-to-end video production studio working in your home or business, running solely on Linux. The few holes left in the pipeline, particularly in the compositing arena, are quickly being filled in and should be in much better shape by mid-year. We're on the cusp of a breakout year

OPEN-SOURCE BLUE SCREENING

Of special note in the open-source compositing field is the newcomer MatteLab (**www.nccn.net/~w_rosky/evan/ evan/programs/mattelab/index.html**), a Java-based one-task utility that is very nicely designed and written by 16-year-old high-school student Evan Rosky. Although, at the moment, it does not support multipass keying, it is the best keyer I've yet seen in the open-source space on Linux. Outputting as it does to transparency-enabled PNGs, it allows for multiple instances of single passes to be run on the footage without quality loss, and when used in conjunction with Blender or Jahshaka for compositing, it is a very powerful tool indeed.

in the media creation field.

During the next few months, I'll be focusing specifically on potential problems in this pipeline, and keep you updated on new developments in both hardware and software. Take heart, fellow producer. There is now a video-production oasis in the open-source desert.■

Dan Sawyer is a freelance director/producer running the backbone of his small studio on Linux. He has been an enthusiastic advocate for free and open-source software since the late 1990s, when he founded the Blenderwars filmmaking community (**www.blenderwars.com**). Current projects include the independent SF feature *Hunting Kestral* (**www.blenderwars.com/kestralmannix**) and *The Psyche Project*, a fine-art photography book centering on strong women in myth.

Hear Yourself Think Again!

WhisperStation™

Originally designed for a group of power hungry, demanding engineers in the automotive industry, WhisperStation™ incorporates dual 64-bit AMD Opteron™ or Intel® EM64T™ processors, ultra-quiet fans and power supplies, plus internal sound-proofing that produce a powerful, but silent, computational platform. The WhisperStation™ comes standard with 2 GB high speed memory, an NVIDIA FX1300 PCI Express graphics adapter, and 20" LCD display. It can be configured to your exact specifications with either Linux or Windows, and specialized applications including Mercury's AmiraMOL,™ PathScale's EKO Compiler Suite or the Intel Performance Tools. RAID is also available. WhisperStation™ will also make a system administrator very happy, when used as a master node for a Microway cluster! Visit www.microway.com for more technical information.

Experience the "Sound of Silence".
Call our tech sales team at 508-746-7341 and design your WhisperStation™ today.

AMD 64 Opteron™

intel inside XEON

Microway®
Technology you can count on sm

Build a Home Terabyte Backup System Using Linux

Build a low-cost, terabyte-sized backup server using Linux and back up your digital audio files, digital images and digital movie recordings. **BY DUNCAN NAPIER**

Aterabyte-plus backup and storage system is now an affordable option for Linux users. This article discusses options for building and configuring an inexpensive, expandable, Linux-based backup server.

Server Design

High-capacity disk drives are now widely available at prices that are incredibly cheap compared to those of only a few years ago. In addition, with so many Linux users now ripping CDs to disk, saving images from their digital cameras and recording video using digital camcorders and DVRs, such as MythTV, the need for backing up and archiving large amounts of data is becoming critical. Losing pictures and videos of your kids—or your audio music library—because of a disk crash would be a catastrophe. Fortunately, a high-capacity, Linux-based backup server can be built easily and cheaply using inexpensive disk drives and free software.

Virtually any home PC can meet the basic requirements for a backup server. If you have long backup windows or relatively small amounts of data, a slow computer is not an obstacle. Make sure your network is fast enough to transfer data within your backup window. For older equipment, the bottleneck for backups can be the disk data transfer bandwidth (30–150Mbps depending on disk technology).

Many consumer-level computers do not have cooling capacity for more than two internal hard disks. Most motherboards support a maximum of four onboard disks (often four ATA/IDE devices, but the two ATA/IDE and two SATA combination is becoming common). External USB high-capacity drives are also available. If your computer is older and has USB1, purchase an inexpensive USB2 PCI expansion card, which is ten times faster.

SCSI has fewer limitations, but it is expensive and has tended to lock purchasers in to "flavor-of-the-month" SCSI technologies. One option for disk expansion and upgrade is the Host Bus Adaptor (HBA), such as those made by Promise Technology. An HBA is a disk controller on a PCI expansion card. HBAs typically require no additional software, have their own BIOS and are not constrained by PC BIOS limits on disk size. HBAs let you put large disks (more than 120GB) into systems with legacy BIOSes, upgrade from ATA-33 to ATA-150 or mix ATA and SATA disks.

You may want to consider purchasing a dedicated fileserv-er. A bare-bones server capable of holding six disks (fully pre-assembled, no disks or OS) can cost less than $1,500 US. With this initial investment, you can expand disk space as needed for less than $0.80 per GB or grow by plugging in USB disks. Once you have decided how many disks you need, consider their space, cooling and noise requirements. Figure 1 shows an example of a backup system build from an old server. The system has well over a terabyte of storage capacity.

Figure 1. Storage array build from an old server (capacity of nine IDE disks, including five in a converted SCSI RAID stack). Additional IDE spots added with Promise HBA.

Even if you choose to build a server from scratch and populate it with high-capacity disks, you can expect costs for your terabyte-plus backup server still to be minimal in terms of its per-gigabyte price. This is because storage costs have decreased so dramatically. Table 1 provides a variety of different configurations for a backup server, along with estimated prices per gigabyte for each (note: prices are estimates and do not include taxes or shipping costs). As you can see from the table, costs for a new server equipped with more than two terabytes of storage can be built for a cost of less than $1.50 per gigabyte. That will back up a lot of home movies, digital pictures and music files!

Building the Server and Adding Additional Disk Drives

During the past few years, I have built backup servers using

Red Hat Linux 9, but you can use any flavor of Linux. I use Red Hat 9 because it is stable, free, currently maintained (Fedora Legacy Project) and simple to install and configure. If you buy a new computer, you may have to use a more current version of Linux. I generally do not use RAID for low-budget systems where cost is paramount, but it is worth considering.

Software requirements for a Linux backup server are minimal. Basic network administration utilities (including the secure shell, SSH, and secure shell dæmon, sshd) and rsync are required. rsync is a fast, incremental duplication/synchronization utility that comes with most Linux distributions. With SSH and rsync, you can carry out virtually all basic backup tasks. It is advantageous for a backup server also to be a fileserver, so I install Samba, the SMB fileserver as well. I use Samba because it is the default fileserver for MS Windows clients, and it also is readily accessible by any UNIX system (including Mac OS X) using a Samba client. If you have a homogenous UNIX network, you can use NFS, which I will not discuss here.

If you need to attach additional disks to your server, begin by making sure you have enough data (IDE/SATA/SCSI) cables and power lines to accommodate the expansion. Ensure that your drive is Linux-compatible (although most are). Turn off the power to your computer and disconnect the power cable. Physically attach the disk(s) to your computer. Linux should recognize the new disk(s) on boot. If your drive is not recognized, your disk is incompatible or you need to locate and install a driver for it. Check boot messages for new drives using the `dmesg` command. The boot message for an IDE drive may look like this:

```
hdb: ST3400832A, ATA DISK drive
```

All IDE/ATA (and some SATA) drives have the designation hdx, where the x is replaced with a letter of the alphabet (b in this case). Similarly, adding new USB or SCSI (and some SATA) disks gives boot messages indicating a new drive designation sdx, where the x is replaced by the appropriate letter.

Most Linux distributions come with a GUI disk manager. These disk managers let you define and format partitions (I generally use one partition per backup disk), assign mount-

points (for example, /data1, /data2) and mount the partition. The process also can be done from the command line using fdisk to create partitions.

Software Configuration—rsync and SSH

rsync is included in most Linux distributions. You need rsync and SSH on both your backup client and server. Check to see whether rsync installed by typing `rsync` at the command prompt or check your list of installed packages. If you cannot find a binary distribution for your package, you can download the source code for rsync by following links on the rsync home page (see the on-line Resources).

The simplest way to run rsync over a network is as a standalone application using SSH for authentication. You can run rsync as a dæmon with more features, but you won't need to in this case. I illustrate this here with a backup client named foo and a server named bar.

To replicate the directory /home on Linux machine foo with directory /data1/foo of backup server bar from client foo using rsync and SSH, type:

```
rsync -az  /home  -e ssh bob@bar:/data1/foo
```

You will be prompted for user bob's password, and then the foo /home directories are replicated to /data1/foo/home on bar (bob needs an account on the server and write permission for /data1/foo).

To avoid having to type bob's password each time, create a private/public key pair for SSH authentication without a password. This allows you to automate the login process.

You can run a script on foo to replicate foo on bar using bob's account on bar. You should read the documentation for rsync, which has numerous features (more than 70 command-line options). In particular, the -delete option can have disastrous consequences if misused. Listing 1 shows a seven-day incremental backup. Files altered or deleted on each day of the week are deposited in directories named for the day (set by -backup-dir). The most recent backup is stored in the directory current.

If you prefer a compressed archive format, you still can run

CREATING NEW PARTITIONS

To create new partitions on hdb (above), type:

```
fdisk /dev/hdb
```

Type **m** at the fdisk prompt for a help summary. Typing **n** at the prompt asks about the new partition we are creating:

```
Command action
   e   extended
   p   primary partition (1-4)
p
```

For a single primary partition, type in **p**:

```
Partition number (1-4):1
```

You are then prompted for a partition number (type **1** for a single partition). Next, set the partition size by determining the first and last cylinder. Because we are using the whole disk, you should be able to select the default values (the first and last cylinders):

```
First cylinder (1-48641, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-48641, default 48641):
Using default value 48641
```

Type **w** to write the partition table. You now have a partition, /dev/hdb1, that occupies the whole disk.

Next, format the partition in the filesystem of choice (mine is in the ext3 format) using the mkfs command:

```
mkfs -t ext3 /dev/hdb1
```

Create a mountpoint for the new partition of your new disk (I'll call it /data1):

```
mkdir /data1
```

Mount the newly created ext3 partition:

```
mount -t ext3 /dev/hdb1 /data1
```

And, test reading and writing. Finally, add a line in /etc/fstab, the mount table, to mount automatically during the boot process:

```
# Device              mountpoint      fstype   options      freq   pass_no
/dev/hdb1             /data1          ext3     defaults        1      2
```

`tar` for a full backup over the network:

```
tar cvfz - /home | ssh bob@bar dd of=/data1/foo/current.tar.gz
```

and use the -newer option for an incremental tar backup.

rsync is more efficient than the tar command, because rsync copies only the differences between the current and previous copy of the data.

You can get by with rsync and SSH on most platforms (including MS Windows), but in reality, a fileserver setup is preferable, especially if you are running MS Windows clients. For MS Windows machines, a Windows backup application is preferable. The easiest way to do this is to run the backup to write to a share on the Samba server.

Software Configuration—Samba

If your Linux installation supports SMB file sharing, Samba is probably installed. If not, binaries are included with virtually all distributions. If this isn't the case with your distribution, or if you prefer to use the very latest Samba version, download the source code and compile and install. Official Samba distributions are available from the Samba home page (see Resources). Refer to the documentation there for installing and initially configuring Samba.

Once your backup server has Samba server installed, all Samba configurations are made by editing the smb.conf file, which is usually in /etc/samba/smb.conf or /usr/local/samba/lib/smb.conf. Graphical configuration utilities like SWAT usually are included with Samba. See your documentation for information about starting or stopping Samba. You should configure your server to ensure that Samba starts when the server initially boots up.

Following our backup example above, on server bar, set up a simple smb.conf file or try appending the section below to the existing smb.conf file to define a share called bob:

```
[bob]
    comment = foo backup account
    path = /data1/foo
    valid users = bob
    public = no
    writable = yes
```

Next, add bob with any secure password as a Samba user

```
#!/bin/sh

# This script does backups of foo to the backup server bar
# in a 7 day rotating incremental backup.
# Based on script by Andrew Tridgell

# directory to backup
BDIR=/home

# Remote directory on backup server
BACKUP_HOME=/data1/foo

# Backup login account on remote server
BACKUP_LOGIN=bob

# the name of the backup server
BSERVER=bar
BACKUPDIR=`date +%A`
OPTS="--force --ignore-errors --delete --backup
-backup-dir=$BACKUP_HOME/$BACKUPDIR -av"

export PATH=$PATH:/bin:/usr/bin:/usr/local/bin

# Dump output to backup file
date > /var/log/backup.$BACKUPDIR.log

# the following line clears the last week's incremental directo-
ry
[ -d /tmp/emptydir ] || mkdir /tmp/emptydir
rsync --delete -a /tmp/emptydir/
BACKUP_LOGIN@$BSERVER:$BACKUP_HOME/$BACKUPDIR/
rmdir /tmp/emptydir

# now the actual transfer
rsync $OPTS $BDIR BACKUP_LOGIN@$BSERVER:$BACKUP_HOME/current >>
/var/log/backup.$BACKUPDIR.log
```

GENERATING THE KEY PAIR

On the machine you want to log in to (logged on as bob on bar in this case), type `ssh-keygen -d` to generate the key pair. Enter a password if the key will be accessible/readable to other users. Otherwise, press Return. Change into the .ssh directory and copy the public key to the allowed list:

```
cd ~/.ssh
cp id_dsa.pub authorized_keys2
```

Copy the private key to the .ssh directory of the account on the machine you will be logging in from (for example, root user on foo). Remove the private key from bar (the machine you want to log on to):

```
scp id_dsa root@foo:~/.ssh/id_dsa
rm id_dsa
```

On the machine you're logging in from, start the SSH agent, and add the key to the agent's list (`ssh-add` asks for a password if you typed one in the first step above):

```
eval `ssh-agent`
ssh-add
```

You can now log in to account bob bar from foo without a password:

```
ssh bob@foo
```

You also can use Samba to serve files to most UNIX or Mac OS X clients. The smb client is installed by default in Mac OS X. In Linux distributions, make sure that the smb client package is installed. The smb share should be mounted onto the /backup mountpoint of machine foo:

```
mount -t smbfs -o username=bob,password=somepassword //bar/foo /backup
```

To have the backup drive mount when the system boots, place a line such as the following in /etc/fstab:

```
//bar/data1/foo    /backup    smbfs  rw,username=bob,password=somepassword  0 0
```

Adding Off-Site Backup for Additional Security

To add an additional level of security, you may consider adding a second server to your overall backup plans consisting of a server that exists off-site, away from the home or office location where your primary backup server is located. This allows you to mirror your backup server to an off-site location once a week. That way, if you have a fire or some other catastrophe at your primary location, your data still will be available. Figure 2 shows a sample configuration for this setup.

Listing 2 is a basic script that mirrors the server bar with an off-site mirror baroffsite using rsync. Always set up backups to

(bob must have a Linux account as well as permission to read/write the /data1/foo directory):

```
smbpasswd -a bob
New SMB password: somepassword
Retype new SMB password: somepassword
Added user bob
```

For MS Windows clients, map the share \\bar\bob as a network drive in MS Windows using the user name bob and the SMB password for the bob Samba account. You then should be able to run backups to the mapped network drive. I typically use the free ntbackup software and set it up to write .bkf files to network storage. ntbackup comes free with Windows 2000 and XP and can run automated, regularly scheduled backups from the Windows client. Windows client-based backups have the advantage of backing up the entire state of the system (including the Windows registry).

Figure 2. Example backup scenario with client foo, backup server bar and off-site mirror baroffsite.

run automatically and on a regular schedule. Always keep logs of your backups, and always check the backup logs.

Monitoring Your Backup Server

In order to monitor your backup process and make sure your backups are running as scheduled (and that your backup server hasn't run out of disk space), it's important to put some automated monitoring and reporting into place. Listing 3 is a simple script that can be set up to run periodically via cron and send you a summary of the backups that have occurred and how much disk space is remaining on each of your partitions.

Resources for this article: www.linuxjournal.com/article/8635.■

Duncan Napier works as computer and instrumentation consultant in the Vancouver area of British Columbia.

Listing 2. rsync Mirroring bar to baroffsite

```
#!/bin/sh

# Mirror /data1 on bar to /data1/bar on baroffsite.

#Backup directory on bar
BACKUP=/data1
#Backup directory on baroffsite
BACKUP_OFF=/data1/bar

# Give the day of week as name of backup
BACKUPNAME=`date +%A`

# Offsite server
BSERVER=baroffsite
# Backup account on backup server
BAC_ACC=backup

date >  /var/log/backup.$BACKUPNAME.log

/usr/bin/rsync -avz --delete -e ssh $BACKUP $BAC_ACC@$BSERVER:$BACKUPOFF
>>  /var/log/backup.$BACKUPNAME.log

# Email the log to administrator

cat /var/log/backup.$BACKUPNAME.log | mail -s 'Mirror Check'
backup_guy@mycompany.com
```

Listing 3. Simple Timestamp and Disk Space Lister

```
#!/bin/sh

# Check space on partitions
# List timestamps in chronological order

BACKUPS=/data1      #Identify directories to check
# Give the day of week as name of backup
BACKUPNAME=`date +%A`

#Timestamp
date >  /var/log/backup.$BACKUPNAME.log
# Disk space on partitions
df -k > /var/log/backup.$BACKUPNAME.log
echo ' ' >>  /var/log/backup.$BACKUPNAME.log

#List timestamps on backup server
# ls -lRt is much more verbose
ls -lt $BACKUPS/* >>  /var/log/backup.$BACKUPNAME.log

# Email the log to administrator

cat  /var/log/backup.$BACKUPNAME.log | mail -s 'Backup Check'
backup_guy@mycompany.com
```

Creating DVDs with Kino and DVDStyler

How to use DVDStyler, a DVD-authoring solution that complements Kino. **BY PHILIP W. RAYMOND**

As a user of Linux for nearly two years, one thing I have found is that it has been lacking in software solutions for video production. I'm happy to report that this situation has changed for the better. For the past six months, I've have been using Kino 0.7.5 for my video editing with results equal to and in some ways better than what's offered in some Windows video-editing solutions or Apple's iMovie. Now there is a DVD-authoring solution that complements Kino by being user-friendly and creating authored DVDs that will play on most post-2000 DVD players. It's called DVDStyler, and in this article, I show you how to use it and some of the options it offers to create as simple or elaborate a menu as you want for your video DVDs.

First, you need a few dependencies in place before you install DVDStyler: wxGTK, DVDAuthor, MJPEG Tools, MPEG Toolbox (mpgtx), Xine or Kaffeine (optional, for pre-viewing video) and Totem (optional, for generation of thumb-nails). Any of these can be found at your favorite RPM or Debian package resource, like those provided in the on-line Resources for this article. DVDStyler creates only video DVDs. It does not create video on CD, better known as VCD and SVCD. If you don't have a DVD burner, you're unfortu-

nately out of luck as far as using DVDStyler. Now, let's get to the business of creating a DVD video after editing a video in Kino.

In Figure 1, you can see the Kino export screen. If you're using Kino 0.7.5 or higher, you already should have the dependencies needed to create a DVD-quality MPEG-2 video file, as you can see in the video and audio windows. Make sure you export to a destination where you can find the file easily. This destination is where DVDStyler will find your video for authoring later. Next, click the export button and the conversion to MPEG-2 begins. Depending on the length of your video, this process could be lengthy. Expect it to take roughly two minutes of rendering for every minute of video. So a half-hour movie would take about an hour to render. Obviously, processor speed figures somewhat into this too. Once rendering is completed, you can move on to using DVDStyler to author your movie.

Figure 2. DVDStyler starts with a blank main screen and a selection of menu backgrounds.

Figure 2 shows what you see when you first click on DVDStyler. Default menu backgrounds are available if you don't have a still picture or a video clip of your own to use (more on this later). First, go to the top of the page and set the

Figure 1. The first step in making a DVD is to export your movie from Kino in MPEG-2 format.

configuration of DVDStyler. Click Configuration, then Settings, and choose your language and default format (NTSC for North America, PAL for most other parts of planet Earth).
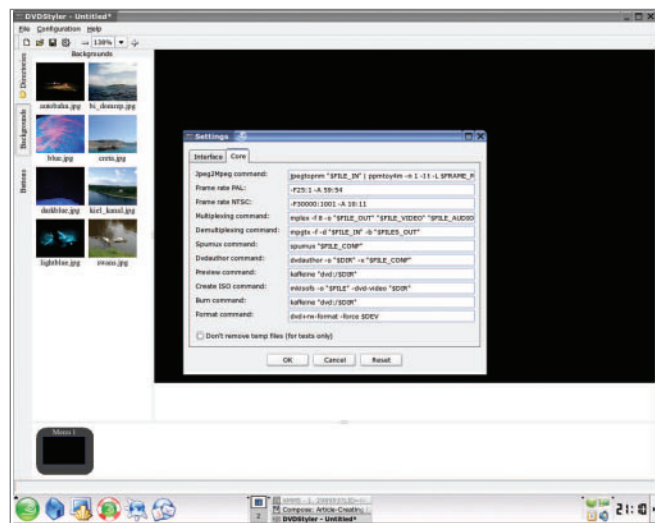
Figure 3. The Core settings tab lets you set your programs of choice for previewing video and burning to DVD.

Next, click on the Core tab to see all the commands and dependencies that are used. The preview and burn commands use Xine by default for previewing, but if you're like me and use Kaffeine, simply type in Kaffeine where it says Xine, so you can preview before burning. Now you are ready to begin inserting your files for menu creation.
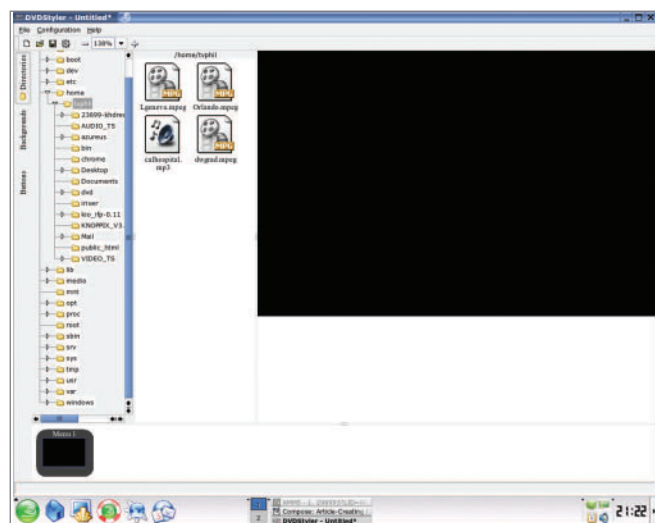
Figure 4. Now you're ready to start building a DVD. Drag the video files you want to use to the time line at the bottom of the screen.

Click on the top-left corner, under the word File on the white square icon to start a new project. If you were returning to a project you already started, you would click the yellow folder icon next to it to find your saved project. Next, on the left side, click on the Directories tab, and you should see all the media files in the current folder that are compatible for DVD creation in DVDStyler. Simply click and drag any of

those files you want to use in your DVD to the bottom of the screen to the right of the Menu 1 thumbnail, in the sequence you want, from left to right. This area is commonly referred to as the time line. Now you are ready to create a background for your menu.

There are several ways to do this, and each way has nearly

Figure 5. You can use a still photo or a video clip for a menu background.

infinite possibilities for background creation. First, if you want to do it the easiest way (the best choice if you've never authored a DVD before), simply choose one of the JPEG pictures offered with DVDStyler. Left-click once on your choice, and then drag it over to the left and into the main screen area. If you want to use a still photo or a video clip for a background, it involves a little more work—here's how.
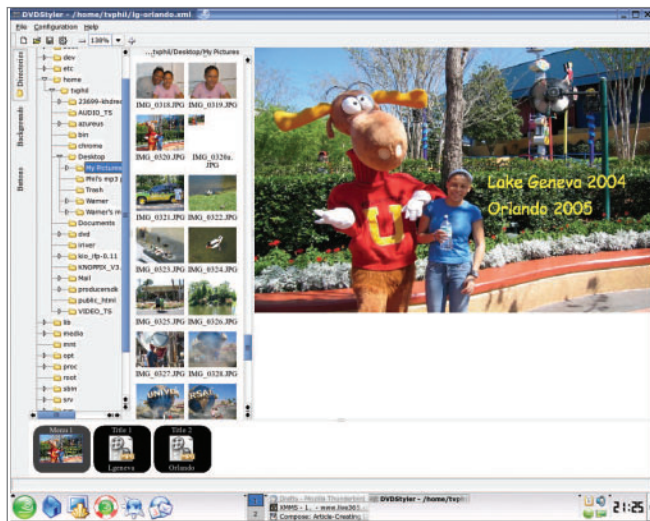
Figure 6. Select the photo you want to use for a menu background, then drag it to the main screen.

For a still photo, go to the Directories tab again and open the folder where your pictures reside. In my example in Figure 6, you see the open My Pictures folder and all of the pictures in that folder. I simply chose the one I wanted to use, left-clicked once and dragged it over to the main screen area, exactly like we did in the first example. The small one (320a) is the one I used for the background.

The reason it is smaller is because I had to resize it using The GIMP before I could use it as a background, as the original size was much larger than what would fit for a menu background. Any still photo needs to be the same size as the NTSC ratio (or PAL) video ratio. In the case of NTSC, it's 720 x 480. If you use a larger photo, you'll end up with some (or a lot) of it framed off. In a DVD menu, a still photo is simply seen as a still frame of video. If you use a pre-edited video clip for your background (something short, 10–60 seconds long), the process is a little different. Simply click on the Directories tab, navigate the tree to the folder where your video clips are located, and then open that folder, double-click on the video clip and you will see the video's file path in your main screen area. You won't actually see the video play at this point, only the words displaying the file path. When your DVD is authored, you will see the video in the background.

Now, it's time to talk about the most important function in DVD authoring, creating buttons. The buttons are what you and other viewers of DVD will navigate to on the menu to trigger the playback of whichever video clip they want on the menu. The first step in creating a button is to click on the Buttons tab. Then, click and drag the word button to the right of the directory and move it to wherever you want to place the button on the background. Don't worry if you change your

mind on the placement later, you still can move it somewhere else on the background and not affect the properties of the button itself. Once your button is on the background, right-click on the word button and choose Properties. You will then see what's shown in Figure 7. Where it says action-jump titles, it is referring to the titles of the movie clips you chose and put on the time line at the bottom of the screen earlier.
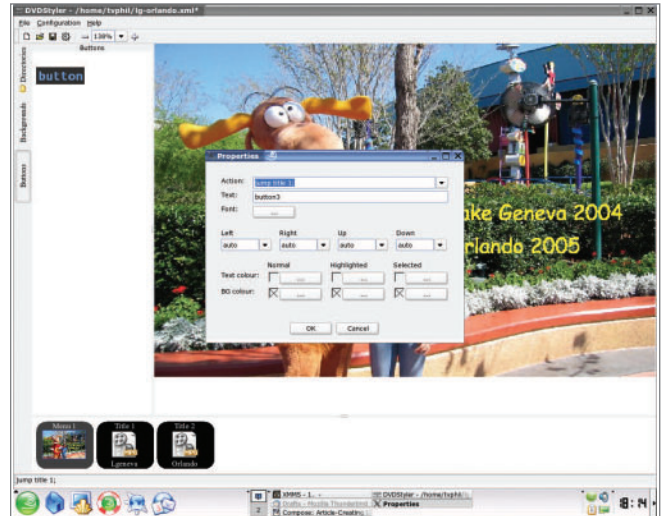
Figure 7. The button Properties dialog lets you set the text and associated actions for a button.

Click on the scrollbar to the right of jump title and you'll see all of the movie titles on your time lime to be represented by button 1. Button 1 doesn't have to be tied to title 1. It can trigger any movie title on the time line; it's your choice. However, you may find it easier to keep buttons and titles tied together chronologically. After you choose which title you want for the button, you also can choose what color background to have behind the text of your button, the color and font for your text and whether it becomes highlighted when navigated to and after it is clicked on when played back on a DVD player.
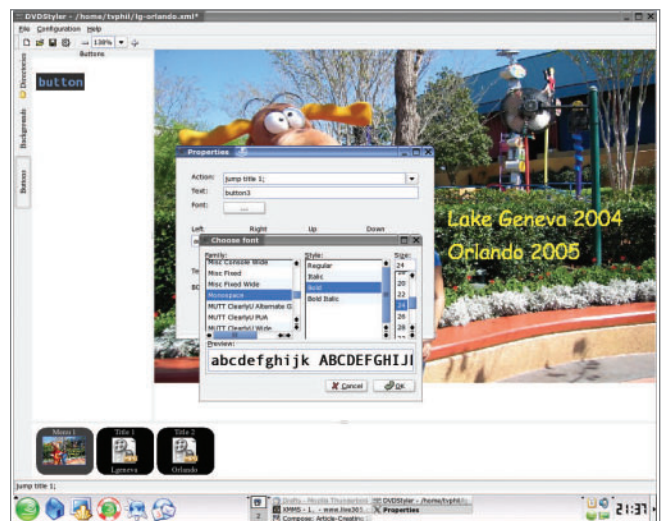
Figure 8. You can set the colors and font for text from the same dialog.

These choices are made right below where the text was entered. In the area called text, it says Title 1 by default. You will want to change that to a name that makes sense to the viewer of the video. One more thing to remember, if you have to stop in the middle of authoring, make sure you click the Save icon in the upper-left corner of the screen before you exit or all of your hard work will be lost. That's it. Now, if you're a more-advanced user, you can create title sets, submenus and make the video background loop. Your options are plentiful. The DVDStyler user guide (a free download from the DVDStyler Web site) goes into great detail explaining how to do everything to create a simple or complicated DVD menu. That's what I think is so great about this application. It's simple enough to use for a beginner, but it also has a lot of more-advanced features that can be used if you know how.
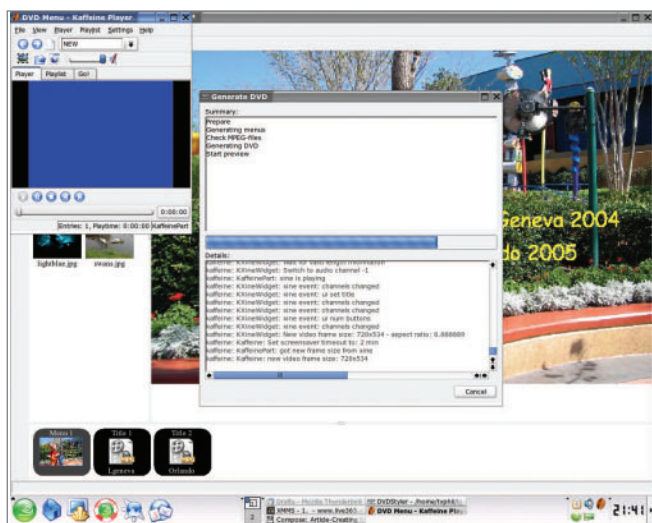
Figure 9. When you're ready to burn the DVD, the Generate DVD dialog keeps track of your progress.

Let's burn! The DVD that is. Simply click on the gear-wheel icon below the word Configuration or click on File in the upper-left corner and scroll down to the command burn DVD. Here you either can preview your work before burning or you can go straight to burning the DVD. When you do, you'll see displayed all the commands that DVDStyler is giving to DVDAuthor to initiate the burning process. If you haven't already inserted your DVD disk into your DVD burner, do it now.

When you insert the DVD, if you get a prompt like I do, asking if you want to start K3b, answer no. K3b is another disk-burning application that many Linux distributions use for burning data DVDs and CDs and audio CDs. It doesn't, however, author video DVDs for DVD player playback. DVDStyler does all the burning by itself. If you're using a distribution that has hotplug, you don't have to mount or unmount your DVD burner. Burning a DVD takes much longer than burning a CD, because it stores much more data. The speed varies with different burners; some go faster than others. Mine is about two years old and burns DVDs only at 2x speed. For me, it takes roughly two minutes to burn every minute of video. A DVD holds only

about 58 minutes of MPEG-2-quality video. So for me, it takes about two hours to burn a 58-minute video DVD. If you have a newer 8x speed DVD burner, obviously it will go much faster. Once the DVD is done burning, it will eject itself. Then, go play it on your DVD player. Most players made after 2000 will play it. If yours is older than that, don't be surprised if it won't play. If this happens, go out and buy the cheapest DVD player you can find ($30), and it will play. If you don't want to do that, you always can play it on your computer and navigate the video clips the same way you would on a TV screen.

Video DVD authoring for Linux is a reality, and best of all, it's pretty easy, even for newbie videographers. DVDStyler is one of the Linux applications I have felt compelled to donate to, to help advance this already good application. It's that good!

Resources for this article: www.linuxjournal.com/article/8641.∎

Philip W. Raymond has been a user of Linux for two years and has worked professionally as a Broadcast Technician for 30 years, the last 21 of those years at WFLD-Fox Television in Chicago. He also can be heard wasting time on his Internet radio station, **www.bsidehits.com**. He can be contacted at tvphil@yahoo.com.

Wireless Home Music Broadcasting—Modifying the NSLU2 to Unleash Your Music!

Start with a storage-attached network device to deliver entertainment to your home.

BY JOHN MACMICHAEL

Like many consumers, the size of my collection of music CDs has spiraled out of control during the past few years. As hard drive prices dropped and CD ripping speeds increased, I jumped at the chance to rip my CDs into MP3 and Ogg Vorbis format; the pile of unorganized CDs became a thing of the past and were relegated to a box in the back of my closet. Although ripping the music collection to my computer eliminated the stack of CDs, it presented new problems. Several members of my family share the computer on which my music was stored; this meant I had to interrupt my son's game of WINE-powered *Deus Ex* if I wanted to queue up an album. Worse yet, I did not have a method of delivering the music from my computer to my stereo on the other side of the house in a low-loss and cable-free method. I cringed at the thought of listening to music blasting from my inexpensive computer speakers in our small computer room. I waited for a product that would be the solution to my problem, but none emerged. I eventually decided to string together several pieces of technology and build my perfect solution. In the end, I desired a setup that would allow me to move my music library off my computer and onto Network-Attached Storage (NAS), play my music with minimal signal loss and remove the necessity of utilizing the family computer as a middleman.

I have always been enamored with projects that extend the capability of computer hardware beyond the vision of the original product engineers. With this as a backdrop, my goal was to chain together several pieces of existing hardware and software to build the most optimal utilization of my audio collection. What I describe is a method to store my music library on a Linux-enabled Linksys NSLU2 NAS device, advertise the music library via mt-daapd, a program that acts as an iTunes (DAAP) server for Linux and other POSIX UNIXes and send the music to my stereo via a Roku Labs SoundBridge, a network music player. There are certainly several ways that this project could have been completed, substitutions could have been made—a full-size Linux server for the Linksys NSLU2, a long audio cable from the server to the home stereo—however, I found this to be the most elegant and practical solution. What surprised me was that I found a thriving community of users that were yearning for the same type of configuration.

Network-Attached Storage, Getting to Know the Linksys NSLU2

The first order of business was to determine a method of moving my music library from my computer and onto a networked storage medium. My initial plan was to convert my aging Pentium III tower into a Linux file and media server that would enable music video across the network while being flexible enough to implement any other services. I scrapped this idea as I was concerned about the power consumption of what would be a mostly idle device, powering up a server just to hear the latest music from the Beastie Boys did not seem like a viable option. Additionally, the space in my tiny computer room is quite limited, and another machine would have added to the heat and noise levels.

I found the perfect alternative in the Linksys NSLU2 Network Link Storage Unit, a NAS device that allows the connection of two external USB 2.0 hard or Flash drives, connecting them to the network via a 10/100 Ethernet connection. The NSLU2 has a Web-enabled interface that allows configuration of file and data sharing across the network. The NSLU2 is packaged in a convenient device that is smaller than my external hard drive, weighs in at less than 6 ounces, consumes less than 9W during normal usage and is easy to stash on a bookshelf or in an unused corner of your computer space. Powered by a modest Intel IXP420 ARM CPU running at 133MHz with 32MB of internal Flash RAM, I had found a storage device but had not determined a method to serve the files.

NSLU2 and the Open Source Community

Several Linksys devices use Linux as the underlying OS; as such, the release of their source code is mandated by the GPL license. Armies of Linux programmers and developers have embraced the Linksys product line to include the NSLU2 (affectionately termed the Slug), and have extended the functionality of these devices. Several flavors or "streams" of replacement firmware have appeared that extend the functionality of the unit. My choice for firmware replacement, and by far the most user-friendly, is the Unslung stream that retains the original product functionality, retains compatibility with stock Linksys firmware, yet allows users to add programs and functionality to the NSLU2. Another popular choice is the OpenSlug firmware, a stream produced with OpenEmbedded, which contains a completely customizable Linux kernel and

root filesystem. OpenSlug installs a minimal OS footprint that is enough to start the hardware and allow for the installation of additional software. This allows for a highly configurable and extendable installation and encourages functionality tailoring based upon user needs—think of it as the Slackware of the NSLU2 firmware world. Both Unslung and OpenSlug are hosted on the NSLU2 Linux Wiki Web site (see the on-line Resources) and have a throng of supporters. At the time of this writing, the latest Unslung firmware is 5.5 and is considered stable. Upgrading the NSLU2 with the Unslung functionality is done in two parts: installing the firmware to the NSLU2 and then "unslinging" the attached USB hard disk or Flash device. This process is straightforward and well documented in the README file included with the firmware. The directions should be followed strictly throughout the process to avoid "bricking" the NSLU2. Upgrading the firmware is as simple as accessing the device Web interface and utilizing the Upgrade Firmware utility. Once completed, the external disk must be unslung. During unslinging, the original root NSLU2 jffs2 filesystem is copied onto the external USB disk and additional directories are added. Each time the device boots after the firmware has been upgraded and the attached disk is unslung, it runs a script that directs the NSLU2 to load a root filesystem stored on the external drive instead of the filesystem in the NSLU2 Flash memory. This allows the user to store code for use by the NSLU2 on the hard disk instead of within the device RAM.

Although programs can be cross-compiled for the NSLU2, more than 6,500 packages have already been ported. Most users will find it much easier to install software using the ipkg system; apt users will find this familiar. The ipkg system allows for network downloads, satisfies package dependencies and manages the installation, upgrade and removal of programs. After upgrading the NSLU2, the ipkg system needs to be configured correctly and the feeds necessary to download new packages need to be updated. After telnetting into the NSLU2, the following commands are issued:

```
#ipkg update
#ipkg install unslung-feeds
# ipkg update
```

Common ipkg commands are `ipkg list`, `ipkg status` and `ipkg install <package name>` to install a particular program. These packages allow the NSLU2 to become a truly capable Linux device; users have reported using the NSLU2 as Web, file, mail, asterisk, media, squid and a host of other server applications.

Continuing the Project—mt-daapd
To transform the NSLU2 from a simple

NAS into a powerhouse music server, the program mt-daapd, written and maintained by Ron Pedde, is utilized. mt-daapd acts as an iTunes server; it is a multithreaded implementation of Apple's DAAP protocol that advertises music libraries for use by iTunes-compatible clients. Although mt-daapd could have resided on a Linux server, I chose to use the NSLU2 to provide music on demand to my network while meeting my low-power and space-saving desires. Ron actively maintains the mt-daapd program and moderates questions and discussions regarding the program on his Web site (see Resources); there is an active contingent of users that share knowledge and help new users in the site forums.

Since I anticipate streaming media to a Roku Labs SoundBridge, I know that I am limited not by the protocols that mt-daapd supports but rather by whether the downstream client supports playing a file format via DAAP protocol. The SoundBridge will play MP3, non-fairplay lossless aac, aiff, WAV formats and Shoutcast streams advertised by mt-daapd. Had I intended to play the music via Ourtunes, an open-source, cross-platform Java iTunes client, I could have had mt-daapd serve up MP3, WAV and OGG formats via the DAAP protocol. There are also filesystem plugins (gnome-vfs-daap and kde daap ioslave) that allow Rhythmbox and JuK to play music advertised by mt-daapd. Ron noted that the most bleeding-edge "nightly version" of mt-daapd supports server-side conversion; this allows for conversion of wma, OGG and flac file types

into .wav format and streaming to the downstream client and enables any client that can play a .wav file to play the converted file. One important caveat is that mt-daapd cannot broadcast aac DRM files that have been purchased from the iTunes music store and are digitally protected.

With this information in hand, we install mt-daapd on the NSLU2. If the user is content with the most recent stable release (0.2.3 at the time of this writing) installation is as simple as telnetting into the NSLU2 and issuing `#ipkg install mt-daapd` to download and install the program and any dependencies. To continue the process, the user navigates to the NSLU2 default network share /DISK 1/public (or /share/hdd/data/public from the console) and creates a subfolder /mp3 in which to store the music files. The server is then restarted and the music database initialized by issuing the following command `#/opt/etc/init.d/S60mt-daapd`. This script restarts mt-daapd if the NSLU2 is ever rebooted. The NSLU2 is now an iTunes (DAAP) server. To test this, open an iTunes client attached to the network, and you will note a new blue entry titled mt-daapd on the left-hand side between the Music Store and 90's Music. The name of your iTunes server, passwords, directories and other variables can be configured by editing /opt/etc/mt-daapd/mtdaapd.conf.

That covers the easy way, but what about the more-involved method? Less-stable nightly packages with Ron's newest feature set are available for download on the mt-daapd

Web site. Ron ominously points out that the nightlies are "development code...significantly less tested than the stable code, and very likely won't work". Ominous or not, the nightly packages have never let me down. As of this writing, the most recent nightly packet is mt-daapd_0.2.2-1_armeb.ipk. To install the nightly package, download a copy of the nightly package and copy it into a directory on the NSLU2; most users choose the $HOME directory. After telnetting into the NSLU2 and navigating back to the directory that contains the .ipk file, the command `#ipkg install mt-daapd_0.2.2-1_armeb.ipk` downloads any dependencies and installs mt-daapd.

That's it! Either method has resulted in configuring the NSLU2 as an iTunes (DAAP) server to any one device on the local network. Before moving on, install a few packages that will enhance your quality of life with the NSLU2. I suggest three: the Bash shell, DropBear for a lightweight secure shell access and wget for downloading files from the console. Each can be readily installed from the command line by typing `ipkg install xxx` and substituting bash, dropbear and wget for xxx.

Final Steps—Streaming Music on the Roku Labs SoundBridge

At this point, the NSLU2 is patiently waiting to serve music on your network. Although a Windows or Mac client could be used to access the music, the original intent was to play the music in pristine quality through my home stereo system. I could have snaked a cable from the computer to the stereo, but the more attractive solution is the Roku Labs SoundBridge. The device is configured with a standard wired Ethernet port and optional CompactFlash 802.11b adapter; I chose to connect to my network with the latter. After connecting a digital optical Toslink fibre connection from the SoundBridge to my stereo, I powered up the unit and entered my wireless network and encryption data with the handy remote (a significant downside is that the device only supports WEP encryption). A quick check to ensure that an IP addresses was issued from the DHCP server, and the SoundBridge is in business. The SoundBridge automatically detected my mt-daapd library, and I used the handy remote control to select and play music from the NSLU2. mt-daapd supports static and smart playlists and passes these to the SoundBridge after a bit of configuration of mt-daapd. The device will also play saved podcasts and Internet radio stations; my NSLU2 has been successfully serving music via the DAAP protocol to my home stereo for several months. It is an incredibly reliable and efficient method of providing music across the local network that has been made possible by an army of developers and enthusiasts in the Linux community.

Resources for this article: www.linuxjournal.com/article/8643.∎

John MacMichael (CISSP, GSEC, CWNA) is a Naval Officer and Information Professional who works in the field of Information Assurance. He considers himself a journeyman Linux user and utilizes a variety of distributions both at work and home, including Slackware, Debian, Red Hat and several live distros; he has yet to find his favorite. He invites your comments at johnny@757.org.

Why is LPI the Global Standard in Linux Certification?

Trusted.

All Linux Professional Institute certification programs are created using extensive community input, combined with rigorous psychometric scrutiny and professional delivery. We test the whole continuum of important Linux skills - we don't just focus on small, subjective tasks. LPI exams are not simply an afterthought used to help sell something else. LPI is a non-profit group that does not sell software, training or books. Our programs and policies are designed to meet educational requirements, not marketing.

Accessible.

LPI exams are available in seven languages, at more than 7,000 locations, in more than 100 countries. You take LPI exams when you want, where you want. In addition, special exam lab events around the world make our program even more affordable. And because we don't make exclusive partnerships, LPI is supported by a broad range of testing centers, book publishers and innovative suppliers of preparation materials.

Independent.

You switched to Linux to get away from single-vendor dependence. So why trade one form of vendor lock-in for another? LPI's program follows the LSB specification, so people who pass our tests can work on all major distributions. Because of its strong grass-roots base and corporate support both inside and outside the world of open source, LPI goes beyond "vendor-neutral" to truly address community needs.

LPI is IT certification done RIGHT!

For more information, please contact us at Info@lpi.org or visit us at www.lpi.org.

Linux Professional Institute

Build a Skype Server for Your Home Phone System

Build this Skype server to provide 24/7 phone service through regular phone handsets in your home—and save a bundle of money in the process!

BY ANDREW SHEPPARD

One irritating feature of Skype is that it must be running on a computer for you to make and receive calls. That is, when your computer is off, Skype doesn't work. Moreover, when you run Skype on the computer you use day in and day out, Skype's performance (call quality, reliability and so forth) can suffer if you are doing other things that deprive it of the runtime resources it needs.

My solution was to build a Skype server that provides 24/7 phone service with the minimum of hassle and fuss. By dumping your regular phone company and taking back control of your home phone wiring using a Skype server, you will have not only a phone system with nearly the same capabilities as before—indeed, in some ways better—you will also save a bundle of money! In my case, I save a little less than $700 US each year (this year, next year, and the year after that, and so on), or about 82% off of my old phone bill.

Using a Skype server plugged in to the existing copper phone wiring of your home means that you can lift a receiver anywhere in your home, at any time, and get a regular dial tone. Incoming calls either from Skype users or regular phones ring all handsets throughout your home. Basically, you can make Skype behave like a regular phone line, but at a tiny fraction of the cost.

You have three choices when building a Skype server: buy a new computer, build a new computer or convert an old machine you have conveniently at hand. This article shows you how to build a new computer from scratch to act as a Skype server. However, whichever path you take, the configuration is the same and is covered in this article.

Skype is not an all-or-nothing proposition, as you can mix and match Skype with your existing phone system, and run the new alongside the old in parallel. That way you have the comfort of having a regular land line and, at the same time, reap the benefits of Skype, such as free Skype-to-Skype calls, and long-distance and international calls at very low rates. This is the approach this article takes, and the configuration you should be aiming for should look something like that in Figure 1. Keeping one of your regular phone lines neatly sidesteps issues such as 911, 411, regular fax and alarm system monitoring (make sure the regular phone line you keep is the one used by your home alarm).

The setup shown in Figure 1 also simplifies the configuration of your Skype server a good deal. Indeed, making multiple instances of Skype run under Linux to support multiple phone lines is another article in itself!
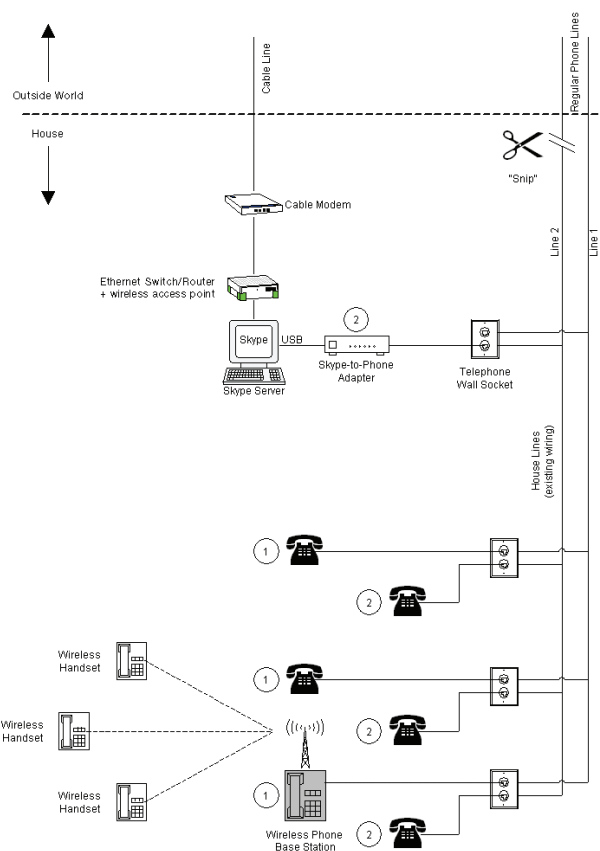

Figure 1. Using a Skype Server to Replace an Existing Regular Phone Line

Build a Skype Server

Whether you buy, build new or piece together a Skype server from computer parts you have at hand, you must first make sure that what you end up with will meet Skype's minimum software and hardware requirements, which are:

■ Fedora Core 3 (Skype also supports SUSE 9, Mandriva 10.1 and Debian 3 or newer. However, Linux support for Skype

add-on hardware is presently extremely limited. In the case of the SkypeMate software used in this article, it is limited to Fedora Core 3 only).

- 400MHz processor.

- 128MB of RAM.

- 10MB of disk space

- OSS-compatible sound device (or ALSA with OSS-compatibility layer enabled).

- Broadband Internet connection.

Pay particular attention to the fact that these are minimum hardware requirements for a single phone line. If you scale these requirements in proportion to the number of phone lines you want your Skype server to support in the long run, you won't go far wrong. You might even want to build in some margin for future expansion. Skype is advancing at a phenomenal rate, with each new release bringing new features and improvements to existing features. All of this new functionality must surely come at the cost of increased hardware resources.

For my Skype server, I decided to build a new machine that would be small, both in terms of its physical size and its power consumption (as it runs 24 hours a day, 365 days a year). The specification, and cost, of my Skype server is shown in Table 1. Remember, a Skype server needs no mouse, keyboard, monitor, CD-ROM or floppy drive—other than at the time of its configuration.

Table 1. Typical Cost of Building a New Skype Server from Scratch	
Component	**Cost (US)**
IN-WIN BT610P.180BFU2 Black steel MicroATX computer case, 180W power supply	$39.99
BIOSTAR M7VIG400 MicroATX motherboard with AMD Duron 800 mobile CPU	$69.00
OCZ value series 512MB (2 x 256MB) 184-pin unbuffered PC 2700 DDR SDRAM	$43.75
10GB Hard disk drive (used—salvaged from an old system)	Free
Skype-to-Phone USB adapter (Figure 2)	$43.90
Linux operating system	Free
Total cost	**$196.64**

Figure 2. A Typical USB Skype-to-Phone Adapter

Building your Skype server requires that you assemble it from the parts. I won't cover the nitty-gritty details as there are plenty of on-line resources to help you in this task; for example, there's a step-by-step guide to building your own PC at PCMechanic.

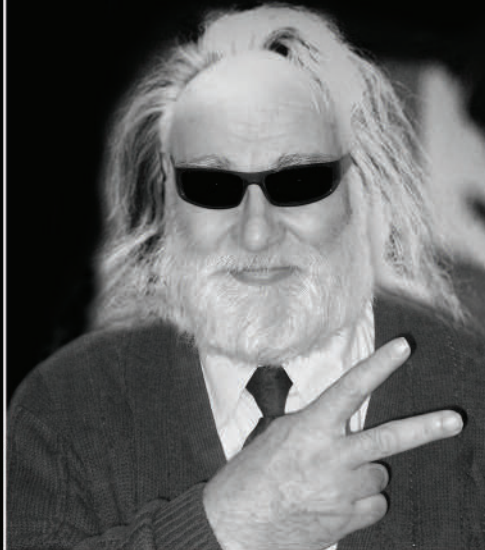### Configure Linux to Work with Skype

I'll assume that because you're a *Linux Journal* reader, getting Fedora Core 3 up and running on your Skype server is a no-brainer. The only important thing to remember is that Skype is a Qt application (though it's also available in a version with Qt statically linked), and the Skype API uses D-BUS. Also, disable the screensaver (after all, there won't be any screen to "save") and power standby features as these may interfere with Skype.

Here's a step-by-step guide to setting up Linux to work with Skype (it assumes you have set up a Linux user account named skype for the purpose):

1. With your Skype server powered off, plug your Skype-to-Phone adapter in to your server using a USB cable and, for test purposes, connect its TEL socket to a regular phone handset.

2. Power up the server and log in to Linux as skype.

3. Download and install Skype for Linux. If you don't install from an RPM, you will have to add this file by hand, /etc/dbus-1/system.d/skype.conf:

```
<!DOCTYPE busconfig PUBLIC "-//freedesktop//DTD D-BUS Bus Configuration 1.0//EN"
"http://www.freedesktop.org/standards/dbus/1.0/busconfig.dtd">
<busconfig>
<!- skype.conf -->
<policy context="default">
<allow own="com.Skype.API"/>
<allow send_destination="com.Skype.API"/>
<allow receive_sender="com.Skype.API"/>
<allow send_path="/com/Skype"/>
</policy>
</busconfig>
```

4. Start Skype and then log in. Next (steps 5 and 6) configure Skype.

5. Make sure Skype starts automatically at login (select Skype→Tools→Options→Privacy, and then check the box opposite Remember my password).

6. As you want your Skype server to provide 24/7 phone service, you will want other Skype users to see your on-line status as always Online. Select Skype→Tools→Options→ General, and then set Show me as "Away" when inactive to 0 instead of 5 minutes. Set Show me as "Not Available" when inactive to 0 instead of 20 minutes. Zero, in this case, means infinity or never.

7. Switch to superuser mode by entering the command su and the root password.

8. Download install-SkypeMate.zip and unzip it to get the file install-SkypeMate (this assumes your Skype-to-Phone adapter is compatible with the SkypeMate software—check before you buy).

9. Change the permissions for install-SkypeMate to make it executable (skype@fc3:~$ is the command prompt; what follows is the command you should enter):

```
skype@fc3:~$ chmod +x install-SkypeMate
```

10. Run the SkypeMate install program:

```
skype@fc3:~$ ./install-SkypeMate
```

11. Exit superuser mode and reboot the Skype server. Log in again as skype.

12. Double-click the SkypeMate icon on your desktop (which

points to /usr/bin/SkypeMate). Skype will pop up a window asking you to give SkypeMate permission to use its API to control Skype (Figure 3). Check the box Do not ask me again, and then click Yes (that way, you won't be asked to give permission again).
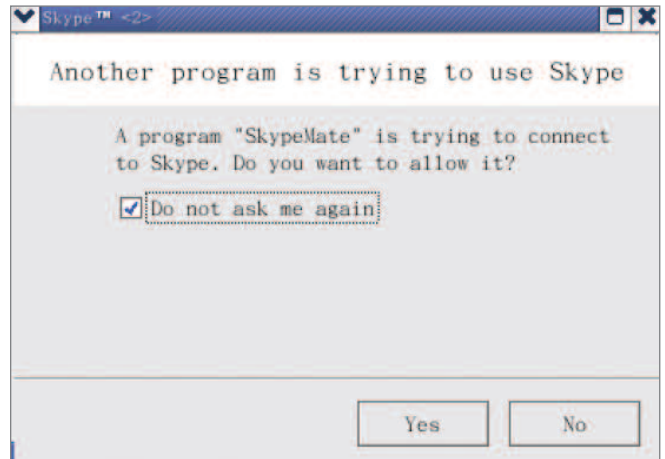

Figure 3. Giving Permission for SkypeMate to Connect to Skype through Its API

13. Select your USB Skype-to-Phone adapter as the audio device for calls (select Skype→Tools→Options→ Hand/Headsets, and then under Audio Devices select the appropriate device from the pull-down list).

14. For your convenience when dialing, you may want to set up speed-dial numbers for your contacts list. That way, you can pick up a phone handset and simply dial, say 10#, to call a specific contact.

15. Test Skype by calling the echo123 call-testing service.

If you want to make calls to regular phones, you will have to sign up your Skype server account for SkypeOut, and if you want to receive incoming calls from regular phones, you will have to sign up for SkypeIn. Both services are available at Skype's cheap rates.

Installing Your Skype Server

Where are you going to locate your Skype server? Ideally, it should be somewhere with access to power, good ventilation, an Internet connection, your regular phone lines (RJ11 sockets) and out of sight. My choice was to install my Skype server in my basement (Figure 4), which is possibly the ideal location, but not necessarily one open to everybody. If your choices are more limited, that's all the more reason to think long and hard about where to put your Skype server once it's built.

Here's a step-by-step guide to installing your Skype server in your home:

1. Cancel one of your regular phone lines (not the one that serves your home alarm system).

2. Cut the incoming phone line that has been canceled where

Figure 4. Skype server to provide 24/7 phone service. Server is at the top of the photo, cable modem and wireless router are to the left, and the patch board for household phone wires is to the right. In the center of the picture is the power distribution cabinet for my house.
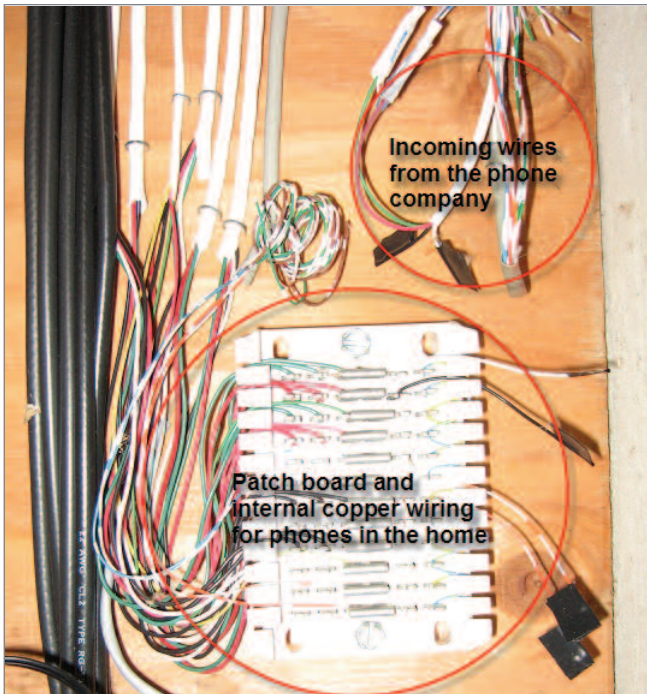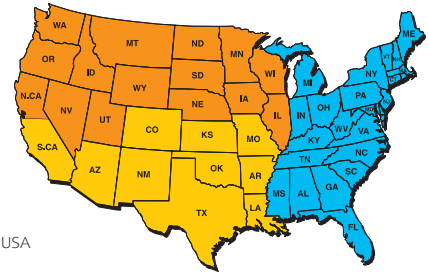

Incoming wires from the phone company

Patch board and internal copper wiring for phones in the home

Figure 5. Phone Patch Board

it enters your home (see sidebar).

3. Connect the Skype-to-Phone adapter to all the handsets of your canceled phone line by connecting its TEL socket to the wall socket of the canceled line using a regular phone cable having RJ11 sockets at both ends.

4. Test Skype again using the handsets plugged in to the canceled line.

5. Power down your server and remove any borrowed hardware that was used during its configuration, but that is not needed for its operation, such as a CD-ROM and floppy drive.

6. Move the server to its new location. Plug in all the cables and connectors, then power it on.

SAFELY CUTTING YOUR PHONE LINE

To cut your incoming phone wires safely, you need a pair of wire cutters (or sharp scissors) with insulated handles, a roll of electrical insulation tape, and you need to follow a simple procedure to do the cutting (Figure 6).

Your phone wires are powered by the telephone company; that's why your regular telephone works during a power cut. Now, even if you've had the phone company disconnect you, your phone wires may or may not still have electrical power. Consequently, you must be careful not to short the wires when cutting them. Even a momentary short can be bad news. This can happen if you cut two or more wires at the same time (step 1). The solution is simple, cut only one wire at a time (step 2). As you cut wires, you should insulate their exposed ends with electrical tape (step 3).
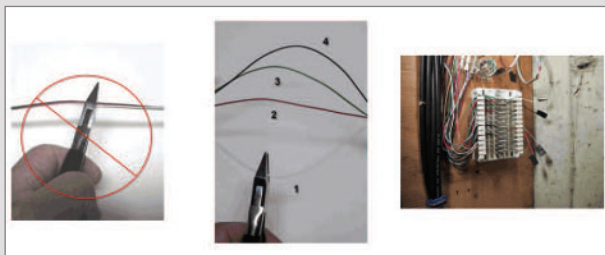

Figure 6. Steps for cutting your phone lines: step 1) don't cut all the wires at once; step 2) cut wires one by one; step 3) insulate ends of cut wires as you go.

Now, if this activity looks to be beyond your comfort zone, you always have the option of calling the phone company and having them send a technician to do the job for you. Even after paying for this to be done, your Skype savings will most likely recoup the cost in very little time.

7. Log in and test Skype once more.

8. Remove the mouse, keyboard and monitor.

9. If all has gone well, you now have 24/7 phone service on one phone line provided exclusively by Skype.

For those readers who are security-conscious, and don't trust Skype as an application, building a Skype server has an added advantage. By placing the Skype server on your Internet connection outside your firewall, you gain the peace of mind that should a hacker break in to your server, or compromise Skype somehow, then as the server interfaces with nothing more than a copper phone line into your home beyond the firewall, any damage will necessarily be contained and limited to the Skype server outside the firewall. Worst case is that you'll need to do a reinstall on your Skype server, and perhaps, a better job of locking it down from a security point of view so that it can protect itself. Indeed, if your firewall is sufficiently restrictive that Skype won't work on the inside, then placing your Skype server outside your firewall is the only way in which you can take advantage of Skype's phone services.

Working Out Your Savings from Using a Skype Server

Phone bills have a lot in common with taxes. Both are mind numbingly complex, and both take a lot and give little in return! To help work out your potential savings from switching to Skype for your phone services, in whole or in part, I've written a spreadsheet that simplifies the process and can be downloaded from the Elpis Web site (see the on-line Resources).

You should factor in the cost of building and running a Skype server into your savings analysis. Running costs will depend on the machine that you choose to use in order to run Skype 24/7. An old clunker of a machine may consume so much power that it would be worthwhile to build a new machine in the long run. As always, run the numbers and make some decisions.

Let's look at the cost of running a small Skype server 24 hours a day, 365 days a year. Without a monitor and other peripherals to guzzle power, a small modern computer typically consumes between 50W and 100W. If your cost of electricity is $0.10 per kWh, then the annual cost of running your Skype server is between $44 and $88. These are just ballpark numbers and you'll no doubt do your own, but it does show that the cost of running a Skype server 24/7 is not insignificant. (Cost = power consumed in kW x 365 days x 24 hours x cost per kWh, where a power consumption of 50W is 0.05kW and 100W is 0.1kW.)

Resources for this article: www.linuxjournal.com/article/8644.■

Andrew Sheppard is the author of the book *Skype Hacks* (ISBN: 05-9610-1899) published by O'Reilly (**www.oreilly.com**), and the editor of Elpis' *Skype Power User Magazine* for Elpis Publishing Limited (**www.elpispublishing.com**). He can be contacted at shep@elpispublishing.com.

Dell's Dimension E510n

Tired of shelling out those hidden Microsoft license fees bundled into the name-brand desktops you buy, only to blow away XP and install Linux instead? Dell's Dimension E510n ships with an absolutely blank hard drive and no pesky tribute to Redmond, Washington in the price. Systems start at $849 US, which will nab you a Pentium 4 630, 512MB of DDR2 RAM, an ATI RADEON X300SE graphics card and an 80GB SATA drive, all in a mini-tower design that Dell claims reduces noise. According to a Dell representative, Linux users will see a savings of $30–$50 US per unit over equivalent systems packages with XP Home.

Micro/sys SBC1670
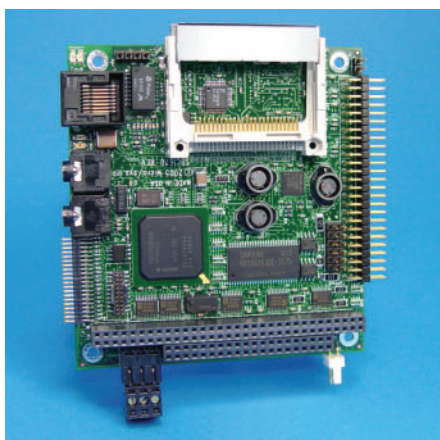
Micro/sys wants to bring Linux into the point-of-sale arena. To that end, they've released the SBC1670, a single-board computer with an integrated flat-panel display interface. The 520MHz ARM processor is coupled with an 800x600 LCD interface, 10/100baseT Ethernet, five serial ports, CompactFlash slot and a keyboard interface. Compatible with the PC/104 standard, it's ready to take its place in cash registers across the world.

QCD Microsystems' Interstructures Product
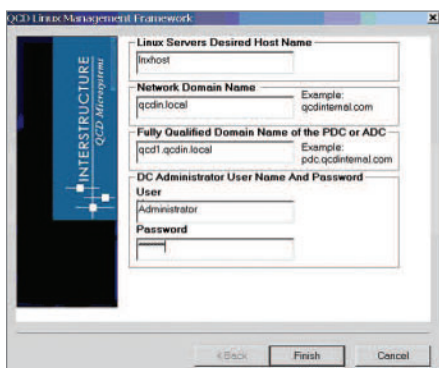
Administering a Samba Primary Domain Controller (PDC) can tax the skills of even a seasoned Linux sysadmin. QCD Microsystems has added PDC management to their Interstructures product, allowing them to be created using a GUI-based interface. According to QCD, users with no previous Samba training will be able to set up a new PDC within minutes. The Interstructures product already provides a graphical front end for tasks such as DNS, DHCP, firewall and Sendmail configuration.

SWsoft Virtuozzo Virtual Private Server

Continuing the industry trend of 64-bit virtualization support, SWsoft has added Opteron and Intel Extended Memory 64 support to their Virtuozzo product, both in host and guest modes. Virtuozzo creates what SWsoft calls Virtual Private Servers, allowing a single machine to host many virtual servers, each with its own IP addresses and securely separated from the other sites being hosted. SWsoft claims near-zero overhead for their software.

Safedesk Solutions' Enterprise Server 2.0

Safedesk Solutions has released the 2.0 version of their Enterprise Server, which offers both Linux-based and Microsoft-based solutions to the thin-client environment. Based on SUSE Linux, it provides both a desktop and remote software execution to thin-client users, as well as access to the same environment through a Web browser from anywhere in the world.

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

Circuit Design on Your Linux Box Using gEDA

Use your Linux box for professional-quality printed circuit board design using CAD tools created by the gEDA Project.

BY STUART BRORSON

Alot of attention—and hype—has focused on bringing traditional office-productivity programs, such as the OpenOffice.org suite, to Linux. However, another important—and far less-hyped—area where Linux's desktop abilities come to the fore is in engineering software, and in particular, CAD (computer-aided design).

Non-engineers tend to think of the term CAD as referring to mechanical design software, and they are partially right. We are used to seeing complicated drawings of mechanical assemblies appearing on computer screens in advertising and television. However, CAD doesn't mean only mechanical design. Electronics designers also long have used computer-based design tools to help them perform their design tasks.

The area of CAD software used for electronics design is typically referred to as EDA, short for electronic design automation. Happily for Linux users, many EDA applications—both proprietary and open source—are now available for Linux. Perhaps the largest and oldest open-source project is the gEDA Project. The gEDA Project is an active community of hackers who have developed high-quality EDA applications for many different electronic design tasks. In this article, we examine a set of gEDA programs you can use to design your own circuit boards from scratch.

EDA Overview

Designing circuits using EDA tools is analogous to creating software using traditional software engineering tools. That is, rather than using only one program to accomplish a job, electrical engineers will use a wide variety of tools, each tailored to accomplish its particular task efficiently. They use each tool in a sequence as they progress from start to finish through their design work, a concept known as design flow. This concept should be familiar to software engineers who might first use a flowcharting tool, then an editor, followed by a compiler, then a debugger, as they create applications ready for distribution.
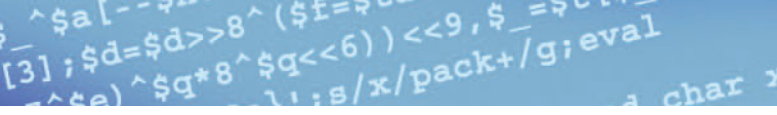
Different types of circuit design use different flows. For example, if you work on digital electronics, you might code your design using a hardware description language such as Verilog, and then use a Verilog compiler followed by a waveform analyzer to simulate and analyze it. On the other hand, if you are doing analog circuit design, you might draw your circuit using a schematic capture program, and subsequently use a circuit simulation program, such as

SPICE, to verify its correctness before proceeding to create a PC board. Other, more-complicated designs might require additional steps in the design flow.

In this article, our interest is to create a garden-variety printed circuit board (PCB). To create a simple circuit board using any EDA toolset—whether proprietary or open source—the design flow looks like this:

1. Gather information about your design's requirements, as well as find data sheets for available parts.

2. Create a high-level block diagram of your design, using either paper and pencil or a general-purpose drawing program such as Dia.

3. Enter your detailed design into the computer, and draw its schematic using a special-purpose schematic capture program.

4. Netlist your design's schematic, meaning that you export a so-called netlist that captures all your design's components and connections in a special-purpose file format.

5. Read the netlist into a layout editor, which is a special-purpose drawing program for creating PCBs. Use the layout editor to create a drawing of your PC board's physical layout.

6. Export Gerber files from your layout program. The Gerber files are industry-standard files that describe your PCB to your PCB manufacturer.

7. Send the Gerber files to a PCB fabricator.

8. Receive your freshly manufactured PCBs, populate them with components and test them out.

At any point in this flow, you might have to go back and repeat a step if you find an error in a preceding step. Again, this is similar to software engineering, where you have to go back to edit and recompile your program if you find that it has a runtime error. The only difference is that if you take a PCB design to its completion by ordering boards, and then you find a design error, you're stuck with a batch of bad boards—you can't simply delete your work and recompile.

Currently, the design tools in the open-source gEDA Suite will support almost any kind of low- to mid-level electronic design task, offering the same power as tools costing thousands of dollars.

Origins and History of gEDA

The gEDA Project was originated by Ales Hvezda in 1998. After graduating from the University of New Mexico, he became interested in robotics. Ales wanted to design circuits for robots using his Linux box, but found that no suitable applications running on Linux existed. Consequently, like so many open-source developers active in other areas, he felt the itch to write his own EDA suite. Ales scratched the itch by writing a schematic capture program and a rudimentary netlister. He also created a Web site to share his efforts, which he named the gEDA Project, because he wanted to release his EDA creations under the GPL.

Very quickly, a small team of developers joined him and started patching and adding to the gEDA tools. Others started contributing tools of their own, some of them modest design utilities and others major software projects in their own right. In the latter category fall the ngspice Project, which aims to develop an open-source implementation of the popular analog simulator SPICE; Gnucap, an analog/digital circuit simulator; and Icarus Verilog, a tool for digital logic generation and simulation. Finally, a very important allied program is pcb—the printed circuit board layout editor.

Because his idea was met with such enthusiasm, and so many tools allied themselves with the gEDA Project, Ales decided to rename his original suite of tools gEDA/gaf, an acronym for gschem and friends. Meanwhile, the term gEDA Suite has come to designate the entire ensemble of open-source EDA tools that have allied themselves with the gEDA Project. A main distinction between gEDA/gaf and the other tools is that the gaf tools tend to be used at the front-end stages of a design, while the other tools are used in later stages.

Since he started it in 1998, gEDA/gaf has grown to a full suite of front-end EDA applications. Besides the core programs gschem and gnetlist, there are almost two-dozen contributed utility programs, including an attribute manager, a symbol checker, an refdes generator (or component numberer), symbol generators and file format conversion utilities. Meanwhile, the various back-end tools, such as pcb, ngspice and Gnucap have also evolved and matured, enabling a variety of different flows adapted to different design types. Currently, the design tools in the open-source gEDA Suite will support almost any kind of low- to mid-level electronic design task, offering the same power as tools costing thousands of dollars.

When discussing the future of EDA on Linux with my friends and colleagues, I like to point out that Sun Microsystems became the corporate giant it is today by offering a great platform for engineering design. Linux can follow the same path to greatness because it, too, offers a powerful platform for engineering applications. Engineers are a natural customer base for Linux as they are smart, technology-savvy and actually want to use UNIX-native power tools such as TCL, Perl and Python to automate their work. Commercial EDA vendors have just begun to realize the customer leverage they can get from the Linux platform, and they are moving fast to serve that market. However, open-source developers recognized the importance of Linux to electronic design many years ago, and the maturity of the gEDA Suite is proof of their early insight.

Schematic Capture—gschem

A good way to understand how gEDA is used is to examine its individual pieces in the context of the overall design flow. The first design step involves schematic capture—that is, using a specialized drawing program to draw a schematic representation of your circuit. The gEDA Suite's schematic capture program is called gschem.

gschem is usually invoked from the command line; once started, it opens up a GUI composed of a drawing window surrounded by all the menus and buttons necessary to draw a schematic. gschem, like any schematic capture program, has a number of built-in graphical primitives corresponding to wires, component pins, resistors, capacitors, transistors and other items you need to connect when creating a circuit design. A screenshot of a typical session with gschem is shown in Figure 1.

As for electronic devices, gschem maintains a library of component symbols, which are drawings of individual circuit elements such as resistors, ICs, connectors and anything else you might want to place on your schematic. Each symbol is stored as an ASCII file; when you place a component symbol into your schematic, the corresponding symbol file is opened up and the information contained in it is used to draw the symbol on your screen.
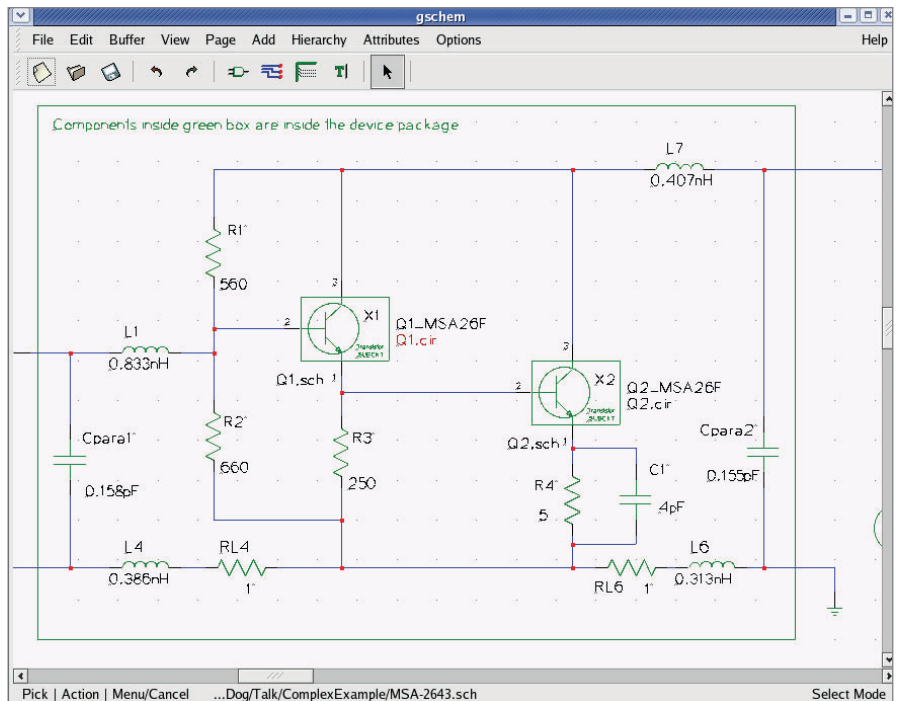

Figure 1. gschem in action. Here is a section of a two-transistor microwave amplifier being drawn in preparation for SPICE simulation.

Currently, gschem's symbol library holds more than 2,000 component symbols, including symbols for most common electronic parts. However, engineers commonly need to create new symbols for their designs, because it is likely that not all the parts they want to use are present in the symbol library. Therefore, gschem—like all schematic capture programs—incorporates a symbol editor, which allows users to create and save their own symbols, which they can then use in any design.

gschem understands electrical connectivity, an important property for any schematic capture program. That is, wires (called nets in EDA parlance) know that they can connect only to component pins and other nets. When two nets are connected together, gschem knows to draw a large dot at the connection point, indicating to the user that a connection exists at that point.

gschem enables engineers to attach attributes to each component, which is an important part of creating a design. For example, if you have a 499-ohm resistor in your schematic, gschem lets you place a resistor symbol from the library, double-click on the resistor and

then attach a value=499 to the resistor itself. Later, when the design is netlisted, the component's attributes are written into the netlist file and made available for other programs.

Finally, gschem saves your design in a well-documented ASCII format. There are many advantages to ASCII file formats; readers of *Linux Journal* will appreciate that ASCII files can be parsed and manipulated using scripting languages, including Perl and Python. Scripts facilitate labor-saving design tasks like automated symbol generation and schematic merging. Many proprietary EDA programs do not use ASCII file formats because they are interested in locking in customers. Open-source EDA advocates believe that open file formats are a key superiority of toolsets like gEDA.

Netlisting—gnetlist

After you have captured your schematic, the next step in the design flow is to create a netlist. gnetlist is the gEDA/gaf program used to generate netlists from your schematic files. gnetlist is a command-line utility; when you run it, it generates output netlist files and also displays diagnostic information in your

terminal window.

So what's a netlist? A netlist is a file holding your design's connectivity information in a structured format suitable for machine processing. Many different types of netlist exist; each represents a file format optimized for a particular type of subsequent processing. For example, SPICE analog simulators read files written in the SPICE netlist format, which calls out connections between analog components, as well as specifies the values of each component's parameters, such as a resistor's resistance. As another example, netlists used as the input to lay out programs typically hold information about each component's PCB footprint, which is the metalization pattern on the circuit board to which the component is soldered, as well as connectivity information between all component pins.

gnetlist is designed in a unique way. It incorporates a front end written in C that reads and parses your schematic files. Once the read-in is complete, gnetlist invokes a back-end netlist generator written in Scheme. The back end is specific to the desired output netlist. The back end to use is specified via a command-line flag when you invoke gnetlist. gnetlist was designed this way to facilitate easy extensibility. Users who want to create new netlisters simply need to write a Scheme program implementing their desired netlister; they don't need to learn C or fool around with the internals of reading or parsing schematic files.

At the time of this writing, gnetlist can output more than 20 netlist formats. Among the important netlist types output by gnetlist is SPICE. The powerful gEDA SPICE netlister spice-sdb supports the inclusion of vendor SPICE models into your spice netlist. It has proven very popular with EE students worldwide, perhaps because it is well documented in a HOWTO available on the Web. Also, netlisters for several different layout tools exist. Finally, gnetlist is also used for BOM (bill of materials) generation and DRC (design rule checking) using any of several back ends crafted to achieve these goals.

Important to PCB design is the question of how to translate a gschem schematic into a format suitable for layout using the open-source layout program pcb. Although this can be done using gnetlist alone, the procedure is complicated. Therefore, Bill Wilson made a recent contribution to the gEDA Project by writing gsch2pcb, a C utility that wraps gnetlist and outputs the correct files to read into pcb for layout. gsch2pcb is a key addition to the gEDA Suite because it makes the transition from a gschem schematic to a pcb layout easy, and it also illustrates the vibrancy of the gEDA on-line community.

Layout—pcb

Once the initial layout files are created using gsch2pcb, you can lay out your design. This involves using a layout editor—a specialized drawing program to draw metal tracks, components, drill holes and other structures onto your circuit board. The PCB layout tool used with gEDA is called, appropriately enough, pcb. pcb usually is invoked from the command line; once running, it presents a drawing window accompanied by all the widgets and tools necessary to draw your circuit board. A screenshot of pcb in action is shown in Figure 2.

The history of pcb is quite interesting. It originally was written by Thomas Nau in 1990 for the Atari ST. Thomas ported pcb to UNIX in 1994 and used the Xaw (X11) widget set for its GUI. In about 1998, Harry Eaton took over maintaining the program, and—among many other contributions—implemented the ability to output Gerber files. pcb was placed on Sourceforge.net about two years ago, and it is currently maintained by Harry, D. J. Delorie (of djgcc fame) and Dan McMahill. Most recently, Bill Wilson (author of gsch2pcb) updated pcb's GUI to use GTK+, a very welcome modernization.

Creating a circuit board layout using pcb, as with any layout editor, involves first placing the component footprints and then routing the metal connections—called tracks or traces—between the pins of the footprint. pcb allows you to define the track width to use, which is important when, for example, drawing power (usually thick) traces, as opposed to signal (usually thin) ones.

As for component footprints, pcb supports two different footprint libraries: a legacy library based upon the M4 macro language and a newer library (newlib), which defines footprints via an ASCII file defining all graphical elements composing the footprint, such as metal pads and rings, drill holes, silk-screened text and so on. When rendering your layout, pcb uses footprints from either library to draw the footprint required by each component; the footprints used are those called out by the footprint attribute specified in your gschem schematic.

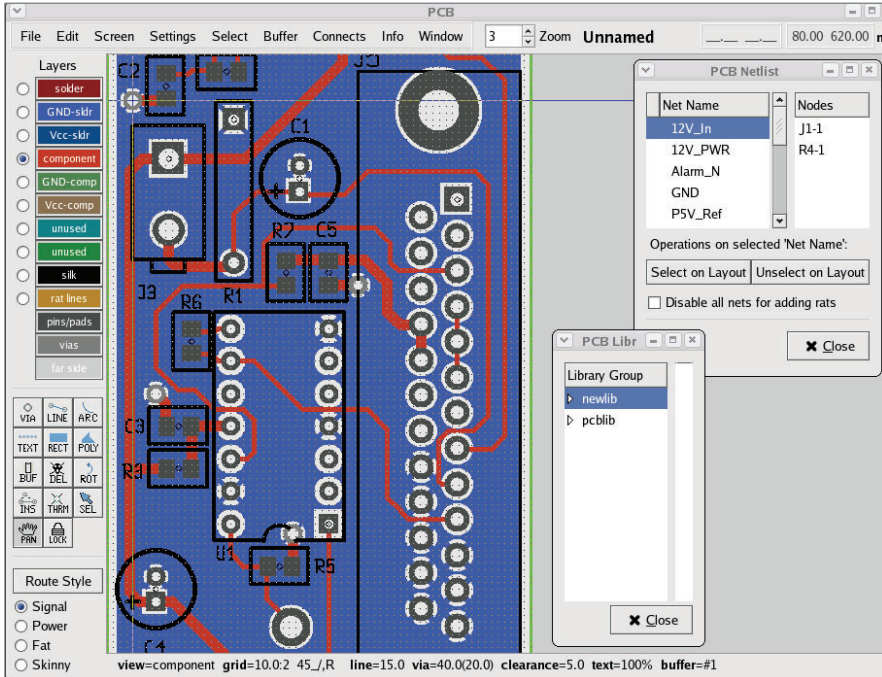Since pcb's newlib defines footprints using an ASCII

Figure 2. A board layout in pcb. The red lines represent metal traces connecting the components on the top layer of the board; the large blue area corresponds to a ground plane on the back layer of the board. A number of component footprints also are visible.
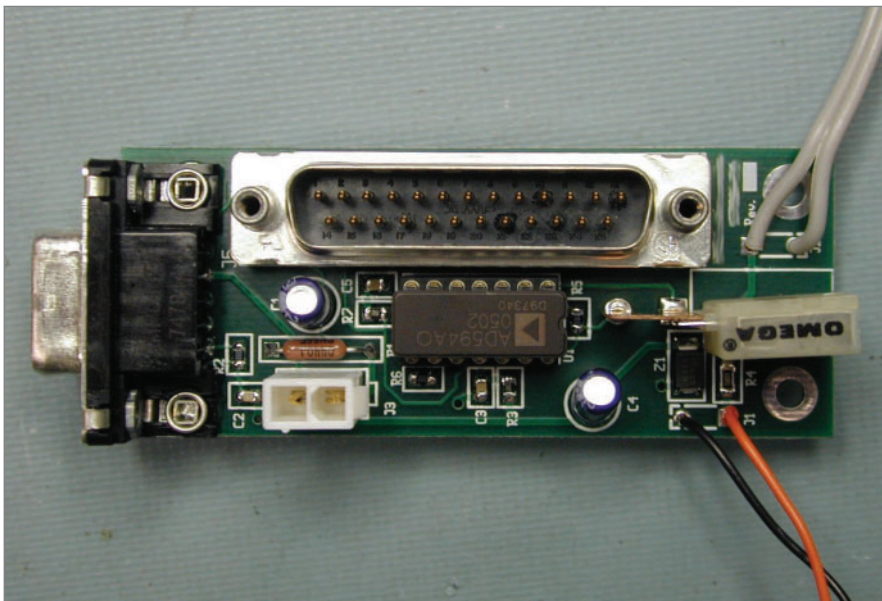

Figure 3. A sensor board created using the gEDA Suite. As is evident, pcb can handle a wide variety of component types.

file format, automated generation of footprints using scripts is possible. To this end, another member of the gEDA community, John Luciani, has created a large collection of useful pcb footprints using Perl scripts; both scripts and the generated pcb footprints are available for free download from his Web site (see the on-line Resources).

pcb supports routing on up to eight layers, meaning that you can draw metal connections on any of up to eight separate layers on the PCB itself. This is important for enabling high-density component placements, which are the norm for modern, compact designs. Connections between tracks on different PCB layers are done by running a pair of tracks to a via, which is a hole drilled through the PCB and subsequently plated with metal, thereby electrically connecting tracks on one layer with tracks on another.

Once you've completely laid out your board using pcb, you can generate Gerber files, which is an industry-standard representation of your board's layout. An assembly drawing, drill file and pick-and-place file also are automatically created when you generate your Gerber files. Send all these files to any PCB fabrication house, and soon you will receive professional-quality PCBs designed by you on your Linux box!

A Finished Board

Once your bare PCBs come back, you either can stuff (assemble) them yourself or send them to an assembler to complete the job for you. Shown in Figure 3 is an example PC board created using the gEDA tools. This board is the same as that shown in Figure 2. It is a two-layer board that aggregates signals from several sensors and routes them to an A/D module. This example board is not particularly large or complex; larger and more-complicated boards are regularly done using the gEDA tools. However, it does show a wide variety of component types: several through-hole connectors, surface-mount and through-hole devices, a 14-pin DIP in a socket, as well as holes and other elements. This illustrates the ability of pcb to handle many different types of electrical components. To see more boards done using the gEDA Suite, look at the featured project on the gEDA Web site, or do a quick Google search. The variety of possible circuit boards is limited only by your imagination!

Resources for this article:
www.linuxjournal.com/article/8530. ■

Stuart Brorson has been an avid Linux user since 1994 and became a contributor to the gEDA Project in 2003. By day, Stuart is a professional electrical engineer involved in designing scientific instruments for spectroscopy.

gEvas: the GTK+2 to Evas Bridge

Learn about some of the Enlightenment Foundation Libraries and how to use them with gEvas for high performance in your next GTK+2 application.

BY BEN MARTIN

The Evas library provides a canvas for quickly rendering raster graphics with alpha blending support. Evas is part of the Enlightenment Foundation Libraries (EFL), a suite of libraries originally built to support Enlightenment DR 17. Other libraries in the EFL that complement Evas are the Edje and Embryo combination and the Emotion library. Edje allows you to wrap up fonts, graphics and functionality into portable theme-like files. Embryo is a simple yet Turing complete scripting language that enables simple scripts to be embedded into Edje files. Emotion allows you to have many video streams as first-class canvas objects. This means you can alpha blend video, move video objects around, resize them and layer them in the canvas while playing.

gEvas is a wrapper and glue library built to allow Evas to be used from GTK+2.x applications easily. In order to motivate you to try gEvas, I present here the points that originally motivated me to use Evas and subsequently create gEvas. My major motivations to use Evas were its simple API and great render speed for an alpha-blended canvas.

Making claims of high render speed unfortunately requires a brief trip into the benchmarking world. The evas_bench application from the Evas distribution involves many pixmap canvas items, scaling and blending of pixmap items and text elements. The image scaling is nowhere near as extreme as the above resize benchmarks. I have ported evas_bench to using GNOME Canvas.

Figure 1 shows a screenshot of the GNOME Canvas port of evas_bench. I have also created some simpler tests for the canvas scaling algorithm both with and without forcing alpha blends on each frame. For the non-alpha-blended

version, a leaf image is resized from larger than full-canvas width to 0 by 0 and back again in a loop; see Figure 2. For the alpha-blended version, a red rectangle image of the same size as the leaf image was used ranging from 0 alpha in the top left to full alpha in the bottom.

As stated above, I created gEvas, and some readers may notice that I am a part of the Enlightenment developers team. Although this is true, I have gone to lengths to ensure that the benchmarks are not biased. The benchmark source code is available (see the on-line Resources). Those reading the source, please excuse the excessive use of less-than-optimal coding conventions used in the quick code hack.

For those unfamiliar with GNOME Canvas, it has two rendering back ends. From the GNOME Canvas developer documentation (see Resources): "...It [GNOME Canvas] offers a choice of two rendering back ends, one based on Xlib for extremely fast display, and another based on Libart, a sophisticated, anti-aliased, alpha-compositing engine." Evas attempts to provide the best of both worlds. I benchmark Evas against both GNOME Canvas rendering engines.

There are also reasons why Evas might not be an appropriate choice. GNOME Canvas supports a Bezier path canvas item that, at present, Evas does not support. In addition, Evas and gEvas are less likely than GNOME Canvas to be preinstalled.

Evas may get beziers in the future. The summary of this point from Raster (aka Carsten Haitzler) was: "If you are after a vector editor suite or something, GNOME Canvas is better. If you want real time fast, alpha-blended object display on all targets, Evas is a good choice."

Evas itself supports multiple back-end render targets, including the framebuffer, XLib and OpenGL. At present, gEvas uses only the XLib Evas back end. However,

Figure 1. GNOME Canvas port of evas_bench tool. Many images and text are moved and resized around the canvas.

Figure 2. gEvas version of gnome_canvas scaling and compositioning benchmark. The leaf is resized gradually to retract to the top left of the canvas and then return to this original size again.

as GTK+2 can run on the framebuffer, you should be able to use gEvas on the framebuffer too, but this is untested.

Both GNOME Canvas and Evas share a similar data model. The Qt QCanvas data model is different enough to make clean benchmark comparisons difficult. The first major difference that makes comparisons difficult is how QCanvas handles images. To put an image on the canvas, you create a QCanvasSprite with a single frame. To scale that image, you then use QImage::scale() or

QImage::smoothScale(), which gives you back an image that you can use to update your sprite. This puts image scaling and the cache handling for scaled images into the client application. Both Evas and GNOME Canvas allow canvas objects to be resized directly and thus take responsibility for handling the caching of scaled images for you.

The second difference is that Qt gives you control over the update tile size. The Qt documentation recommends: "A good rule of thumb is that the size should be a bit smaller than the average canvas item size. If you have moving objects, the chunk size should be a bit smaller than the average size of the moving items."

Due to the data model differences, I haven't created a Qt port of evas_bench as yet. I did create a canvas scaling and blending client, although there are some canvas design issues affecting a clean comparison.

With Qt moving the cache policy fully in the client, I have chosen to cache all scaled images in the first iteration of scaling and reset the benchmark start timer for further iterations that will use only the cache.

So keep in mind that the Qt resize benchmark is performed with all resized images pre-cached and many user-specified tile sizes are used for benchmarking. This effectively should give QCanvas a huge speed advantage over Evas and GNOME Canvas.

Let's Race Them

The results should be similar on a relative basis no matter what hardware is used. For the sake of completeness, my testing CPU was an AMD XP-Mobile running at 2.4GHz with 200MHz FSB, with 1GB of RAM at 400MHz dual-channel cas222 and and NVIDIA 5900 video card. The software that may affect performance includes xorg-x11-6.8.2-1.FC3.13, either GCC 4.0.0 20050308 (Red Hat 4.0.0-0.32) or GCC 3.4.3. X11 was configured with TwinView with one 1024x768 and one 1600x1200 screen, both running 85Hz in 32-bit color. TwinView should not affect runtimes because all canvases are using software render paths that should be more sensitive to CPU/RAM speed.

Using client libraries qt-3.3.4, libgnomecanvas-2.10.0 recompiled with below CFLAGS, Evas CVS checked out on May 28, 2005. Evas was compiled with GCC 3.4.3 with the below CFLAGS. Benchmark compiled code CFLAGS and CXXFLAGS are generally:

```
-O3 -march=athlon-xp -fomit-frame-pointer
```

I benchmarked the qt-canvas-resize client separately because of the image caching distinction and chunk-size optimizations mentioned above. Shown in Table 1 are the benchmarks for qt-canvas-resize, where the Qt part of the main loop consists of:

Table 1. qt-canvas-resize Benchmarks with Different Tile Sizes			
Application	Chunk size	Leaf image FPS	Alpha rectangle FPS
qt-canvas-resize	default	114	72
qt-canvas-resize	32	128	80
qt-canvas-resize	64	136	82
qt-canvas-resize	128	142	81

Table 2. GNOME Canvas vs. (g)Evas in the Resize Benchmark		
Application	Leaf image FPS	Alpha rectangle FPS
gnome-canvas-resize	21	21
gnome-canvas-resize --aa	149	127
evas-resize	190	184
gevas-resize	185	177

```
QCanvasSprite* leaf_sprite = ...;
QCanvasPixmapArray* leaf_tiles = ...;
while( running )
{
  while( app->hasPendingEvents() )
    app->processEvents();
  QImage im = ... from cache ...;
  QCanvasPixmap* qpix = new QCanvasPixmap( im );
  leaf_tiles->setImage( 0, qpix );
  leaf_sprite->setFrame(0);
  canvas->update();
}
```

The client has several command-line options: --alpha-blend-image is used to alpha-blend the red rectangle instead of the leaf, and --chunk-size is used to specify a non-default chunk size. The --alpha-blend-image option is common to qt-canvas-resize, gnome-canvas-resize and (g)evas-resize.

Running the default chunk size leaf image qt-canvas-resize through valgrind's callgrind for a few minutes reveals that QCanvas::update() gets 30% overall runtime, with 59% of runtime used by QCanvasPixmap::init(). So the benchmark could be considerably better if the pre-cached images are stored in a QCanvasPixmapArray for the sprite.

To test this level of pre-caching, I added the -Z option to put all of the cache images into a single QCanvasPixmapArray, which is the backing for the QCanvasSprite. With this optimization, 559 FPS can be achieved with 78% of runtime in QCanvas::update() and 7% in QCanvasSprite::setFrame(). It has to be noted that this level of pre-caching presents a unfair advantage for QCanvas for rendering speed.

The GNOME Canvas client for image scale and blend is

Table 3. evas_bench and Its GNOME Canvas Port Head to Head

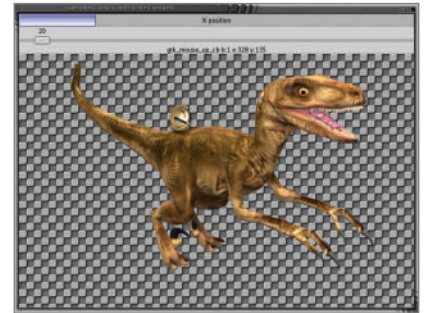| Application	FPS	EVAS_BENCH
gnome-canvas-port-evas-bench --aa	90	1.49
evas_software_x11_main	164	2.75
evas_software_x11_main --smooth-off-for-some	200	3.32
evas_buffer_test	290	4.83

Figure 3. Connecting Evas and GTK+ signals. The dinosaur image can be dragged around either directly or by moving the slider bar.

gnome-canvas-resize, which has an --aa option to choose the GNOME Canvas alpha-blending back end. evas-resize has no custom options.

Without the --aa option, gnome-canvas-resize spends 99% of its time inside gtk_widget_send_expose(), which is called one way or another from g_main_context_iteration(). I don't think that the non-aa GTK+2 engine likes being used in a flat-out benchmark manner. Using callgrind on the --aa GNOME Canvas back end finds 96% of its time spent in gtk_widget_send_expose(), although now we can see 66% of the time in gdk_pixbuf_composite() is called indirectly from gtk_widget_send_expose().

evas-resize is spending 99% of its time in evas_render_updates(). Of the functions called from evas_render_updates(), 91% is spent scaling functions.

I also ported the evas-resize to using gEvas and its API calls. Although there is some speed loss due to the GTK+ signal glue and other gEvas trimmings, the loss is not too significant.

This comparison shows that for outright image scaling, GNOME Canvas and QCanvas are similar and both are slower than Evas. Evas gains further ground when the scaled image also needs to be alpha-blended to the background.

I modified the original evas_bench application to remove the use of features not easy to replicate in GNOME Canvas. Other features became optional to measure their impact on overall performance. The setting of clip zones in Evas is not an easy thing to port to GNOME Canvas, so these were disabled in the evas_bench. Smooth scaling also can have a strong impact on performance, so an option to turn that off for the Evas version was added.

It should be noted that Evas doesn't currently implement a cache for scaled images. So every frame in the Evas benchmarks is performing an image scale and blend.

The evas_buffer_test client performs the same work but only renders the output to a 32-bit RGBA image buffer in memory.

gEvas

At its core, gEvas provides five things: it tells Evas when to repaint itself, assists in gluing Evas events and glib2 signals together, handles Edje timer calls to support animation, helps Evas play nice with GTK+2 widgets and codes to assist in Evas usage. Because Evas also is being targeted at embedded systems, some handy code is left out of the core Evas to make it lean. Because gEvas is desktop-targeted, it adds some handy functionality for desktop applications.

The following code creates a gEvas canvas inside a scrollable area and attaches it to a GTK+2 window. As not every scrollable gEvas will want to allow the middle button to drag the canvas position—as in The GIMP—you have to set this up outside of gevas_new_gtkscrolledwindow():

```
GtkWidget* window = 0;
GtkWidget* scw    = 0;
GtkWidget* gevas  = 0;

window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
gevas_new_gtkscrolledwindow(
    (GtkgEvas**)(&gevas), &scw );
gtk_container_add(GTK_CONTAINER(window), scw);

gtk_scrolled_window_set_policy(
    GTK_SCROLLED_WINDOW(scw),
    GTK_POLICY_AUTOMATIC, GTK_POLICY_AUTOMATIC);

gevas_set_middleb_scrolls(GTK_GEVAS(gevas), 1,
  gtk_scrolled_window_get_hadjustment(
      GTK_SCROLLED_WINDOW(scw)),
  gtk_scrolled_window_get_vadjustment(
      GTK_SCROLLED_WINDOW(scw)));
```

The gEvas API for creating objects takes a leaf from standard GTK+ coding, having some methods attached to a general GtkgEvasObj class. Other special classes, such as GtkgEvasImage, are derived from GtkgEvasObj. Unfortunately, this also brings the usual cast-heavy style of ANSI C GTK+ programming.

The following creates an image showing your PNG file at its original width and height. We then move the image and raise its layer in the canvas:

```
GtkgEvasImage* gi;
GtkgEvasObj* go;

gi = gevasimage_new_from_metadata(
        GTK_GEVAS(gevas), "/my/path/foo.png" );
go = GTK_GEVASOBJ( gi );

int x = 100, y = 50;
gevasobj_move(      go, x, y );
gevasobj_set_layer( go, 1 );
```

I created a simple client showing how to connect Evas events from the canvas to some GTK+2 widgets outside the

gEvas widget. Look in the demo directory of the gEvas package for signalconnect.c. For a more-advanced example, take a look at the testgevas client for the use of both raw Evas callbacks and Evas-triggered callbacks that are marshaled to glib signals. Signalconnect is shown in Figure 3.

Connecting to Evas events is handled by way of the GtkgEvasEvHClass subclasses. The below fragment causes evh to marshal Evas' mouse up/down events into glib signals, which then are connected to reporting functions. Also, when the user moves the raptor image, raptor_moved() is called by way of a glib2 signal to update various GTK+2 widgets with the current coordinates of the image:

```
static gint raptor_moved(
    GtkgEvasObj* o,
    Evas_Coord* x, Evas_Coord* y,
        gpointer user_data )
{
 gtk_progress_bar_set_fraction( x_coord_tracker,
        (1.0 * (*x)) / CANVAS_WIDTH );
 gtk_range_set_value(
        GTK_RANGE(y_coord_tracker), *y );

 return GEVASOBJ_SIG_OK;
}

static gboolean
gtk_mouse_down_cb(GtkObject * object,
 GtkObject * gevasobj, gint _b, gint _x, gint _y,
 gpointer data)
{
 char buffer[1024];
 snprintf(buffer,1000,"mouse_down b:%d x:%d y:%d",
        _b, _x, _y);
 gtk_label_set_text( e_logo_label, buffer );
 return FALSE;
}

...
gi = gevasimage_new();
go = GTK_GEVASOBJ( gi );
gevasimage_set_image_name( gi, "raptor.png" );
...

/** Let the user drag the raptor around **/
GtkObject *evh = gevasevh_drag_new();
gevasobj_add_evhandler( GTK_GEVASOBJ( gi ), evh );

gtk_signal_connect( go, "move_absolute",
    GTK_SIGNAL_FUNC( raptor_moved ), go );

gi = gevasimage_new();
go = GTK_GEVASOBJ( gi );
gevasimage_set_image_name( gi, "e_logo.png" );
...

evh = gevasevh_to_gtk_signal_new();
gevasobj_add_evhandler( GTK_GEVASOBJ( gi ), evh );

gtk_signal_connect(GTK_OBJECT(evh), "mouse_down",
  GTK_SIGNAL_FUNC(gtk_mouse_down_cb), NULL);
```

```
gtk_signal_connect(GTK_OBJECT(evh), "mouse_up",
  GTK_SIGNAL_FUNC(gtk_mouse_up_cb), NULL);
```

The following are some more functional event handlers that can be attached:

```
/* Standard GTK+ popup menu creation + handling */
static gboolean
gtk_popup_activate_cb(GtkObject * object,
    GtkObject * gevasobj, gint _b, gint _x, gint _y,
    gpointer data)
{
 static GtkMenu *menu = 0;
 ...
}

GtkgEvasObj* go  = ...;
GtkObject*   evh = 0;

/* Make the object throb when mouse is over it */
GtkgEvasEvHThrob* evht = gevasevh_throb_new( go );

/* Allow the user to drag the object around  */
evh = gevasevh_drag_new();
gevasobj_add_evhandler( go, evh );

/* Make a popup menu appear on right mouse click */
evh = gevasevh_popup_new();
gevasobj_add_evhandler( go, evh );

gtk_signal_connect(GTK_OBJECT(evh),"popup_activate",
    GTK_SIGNAL_FUNC(gtk_popup_activate_cb), NULL);
```

Handling a selection in the canvas is a little trickier than the above event handlers. This is so because more than one object is involved in the selection process. You create a selector event handler object of class GtkgEvasEvHGroupSelector that attaches to the object you want as the background unselectable object. You can think of this object as where the rubber-band rectangle is drawn to indicate which objects should become selected. The rubber band always is drawn at a higher layer than the selectable objects. Each selectable object on the canvas then has a GtkgEvasEvHSelectable object attached to it that communicates with the GtkgEvasEvHGroupSelector object:

```
GtkWidget*      gevas = ...;
GtkObject*      evh_selector = 0;
GtkgEvasImage* gevas_image;

gevas_image = gevasimage_new();
gevasobj_set_gevas(gevas_image, gevas);
gevasimage_set_image_name(gevas_image,".../bg.png");

/* Make this a group_selector */
evh_selector = gevasevh_group_selector_new();
gevasevh_group_selector_set_object(
  (GtkgEvasEvHGroupSelector*)evh_selector,
  GTK_GEVASOBJ(gevas_image));

GtkgEvasObj* go  = ...;
make_selectable( gevas, go, evh_selector );
```

```
...

/* lets make this object also selectable */
void make_selectable( GtkgEvasObj* object,
                      GtkObject* evhsel )
{
 GtkgEvasObj* ct  = 0;
 GtkObject* evh = gevasevh_selectable_new( evhsel );
 gevasevh_selectable_set_confine(
    GTK_GEVASEVH_SELECTABLE(evh), 1 );

 gevasobj_add_evhandler(object, evh);
 gevasevh_selectable_set_normal_gevasobj(
    GTK_GEVASEVH_SELECTABLE(evh),  object);

 ct = (GtkgEvasObj*)gevasgrad_new(
     gevasobj_get_gevas( GTK_OBJECT(object)));

 gevasobj_set_color( ct, 255, 200, 255, 200);

 gevasgrad_add_color(ct, 120, 150, 170, 45, 8);
 gevasgrad_add_color(ct, 200, 170, 90, 150, 16);

 gevasgrad_set_angle(ct, 150);
 gevasobj_resize( ct, 200,100);
 gevasobj_set_layer(ct, 9999);

 gevasevh_selectable_set_selected_gevasobj(evh,ct);
}
```

You then can easily test if objects are selected or get a collection object to perform operations on all objects selected:

```
GtkgEvasEvHGroupSelector* ev = ...;
GtkgEvasEvHSelectable*    o  = ...;
GtkgEvasObjCollection*    col = 0;

gboolean yn = gevasevh_group_selector_isinsel(ev,o);
col = gevasevh_group_selector_get_collection( ev );
gevas_obj_collection_move_relative( col, 100, 200 );
```

In addition, some objects, such as geTransAlphaWipe, were created to perform image transitions before Edje existed. Although Edje is the way of the future, the alphawipe code allows you to perform a common simple transition without involving Edje. This is used in gevasanim to create a sprite-like object that transitions between its frames using alpha blending.

The xxx_from_metadata() functions in gEvas allow you to set up location, image filename, visibility and other attributes for a new object using a single string. Both the from_metadata() and the transition code duplicate functionality now also available in Edje:

```
sprite = gevas_sprite_new( GTK_GEVAS(gevas) );

for( i=1; i<frame_count; ++i )
{
 gchar* md = g_strdup_printf(
   "cell%ld.png?x=120&y=120&visible=0&fill_size=1"
   ,i);
```

```
 gi = gevasimage_new_from_metadata( GTK_GEVAS(gevas), md
);
 g_free( md );
 gevas_sprite_add( sprite, GTK_GEVASOBJ( gi ) );
}

gevas_sprite_set_default_frame_delay( sprite,2000 );
gevas_sprite_play_forever( sprite );

/* frame transitions */
geTransAlphaWipe* trans = 0;
trans = gevastrans_alphawipe_new();
for( i=0; i<frame_count; ++i )
 gevas_sprite_set_transition_function(
     sprite, i, trans );
```

For the grand finale, I put an Edje object onto the gEvas canvas. The client I created to demonstrate for this article is demo/gevasedje in the gEvas distribution. The interesting parts are shown below. The Edje itself has an Enlightenment logo that spins around and one that throbs under mouse clicks:

```
/* init engines */
ecore_init();
edje_init();
gtk_init(&argc, &argv);

...
gevas = ...;
/* allow edje to update the canvas as well */
gevas_setup_ecore( (GtkgEvas*)gevas );

/* place an edje object */
GtkgEvasEdje* gedje
 = gevasedje_new_with_canvas( gevas );

/* eet files can contain many edje objects */
gevasedje_set_file( gedje, "e_logo.eet", "test" );
go = GTK_GEVASOBJ(gedje);
gevasobj_move(     go, 300, 300 );
gevasobj_resize(    go, 370, 350 );
gevasobj_set_layer( go, 10 );
gevasobj_show(     go );
```

Wrap-Up

I have omitted memory consumption figures from the article in order to conserve space, as those figures are less important for modern desktop machines. A future benchmark will pit QTCanvas resizing against an Edje resize program to provide a more fair comparison between those two.

If you are looking for a canvas for a new GTK+2 application, consider both GNOME Canvas and gEvas plus Edje before hacking.

Resources for this article: www.linuxjournal.com/article/8647.

Ben Martin spends most of his time working on virtual filesystems and data mining them, although he also is known to play with animating other peoples' image art in two and three dimensions.

FREE NEWSLETTER!

Wish you could get the latest from *LJ* more than once a month? You can—sign up today for *LJ's weekly e-mail newsletter*.

Each week the *LJ* newsletter features great tech tips, links to web-only articles, and news on the latest events in the Linux market.

Sign up for the *LJ* e-mail newsletter now:

http://www.linuxjournal.com/

Bringing Usability to Open Source

How usability testing is helping improve desktop Linux. **BY NAT FRIEDMAN**

During the past year, we at Novell have conducted hundreds of usability tests on different parts of the Linux desktop. We use two video cameras—one on the face, one on the hands—and a frame grabber to record everything the user does. We ask our subjects to perform five or six simple tasks with Linux, and we burn the result to a DVD.

This month, we've released our test results to the Linux community at betterdesktop.openSUSE.org. Linux developers have access to more than 200 user tests and analysis. You can watch real users encounter real design problems and get a sense of their thought process.

ister for her, however. Instead, because she wanted to "send" a mail, she clicked repeatedly on the "Send" part of the Send / Receive button just to the right. For about a minute.

This is easy to fix; we just need to change the labels to be more sensible (and then test again on 5–6 people to be statistically comfortable that we changed them appropriately). It was interesting to watch this video and instantly realize that the Send / Receive button is all about how Evolution works and not about what the user wants to do. I've been staring at that button for five years, and I never realized it was wrong until I saw that video.

recording are implemented entirely in software. At these prices, this introduces a new way for people to contribute to open-source projects: perform some usability testing and send the results to the authors.

In one test, we asked a woman to find a document she had created earlier and make some changes to it. There are a couple of ways to do this: she could use Beagle to search for the file, she could open the file manager and step through her directory structure until she found the file or she could use the recent-documents facility in the OpenOffice.org File menu to find the document.

She paused briefly with a blank desktop in front of her, looking to see if the document she had created might be there. She opened the file manager and poked around a little bit. And then, finally, she ran OpenOffice.org and went directly to the File menu. "I know I'm doing this wrong", she said, selecting the file by name from the menu, "but at least this will work."

Technically, she succeeded in achieving her goal: she opened the file, and it didn't take her very long to do it. But she tried a few different avenues before finding one that worked. And she wasn't comfortable about it.

There's a difference between software that is usable and software that is a pleasure to use. Until you watch people using your software, it's hard to know how well you're doing.■

This month, we've released our test results to the Linux community at betterdesktop.openSUSE.org. Linux developers have access to more than 200 user tests and analysis.

We've all read about the benefits of usability testing, but until you actually try to sit through two hours of these videos, you don't viscerally understand why it's so important. Watching these videos is exciting and emotionally exhausting. You squirm. And it focuses you like a laser.

For example, we asked a woman to send mail to a friend. Against all odds, she started Evolution (nothing in the menus indicates that it's a mail program; something we hadn't realized before, but it was immediately obvious after watching her stalk one by one through the menu items muttering to herself along the way).

The correct next step would have been for her to click on the New button that's in the upper left-hand corner of the window. This button didn't even reg-

Most industrial usability testing labs are impressive facilities: a one-way mirror separates the test subject from a video processing/observation room, where the video footage is recorded and where engineers and product managers can watch the test subjects in real time. The observation room usually has to be sound-proofed to muffle the anguished shouts of the engineers. These facilities are impressive, but they are also very expensive and somewhat intimidating to test subjects, who often feel like their abilities are being scrutinized in the stark light.

We built our labs for less than $1,000 US each with a couple of cheap Webcams, a video mixer, a DVD recorder and a small TV used as a monitor. It is feasible to get this cost down to $200 if the frame grabbing and video

Nat Friedman is vice president of Linux desktop engineering at Novell. Both a hacker and an entrepreneur, he co-founded Ximian, which was acquired by Novell. Nat started the Beagle, Hula and Better Desktop projects and served as chairman of the GNOME Foundation for two years.

Rackspace — Managed Hosting backed by Fanatical Support.™

Servers, data centers and bandwidth are not the key to hosting enterprise class Web sites and Web applications. At Rackspace, we believe hosting is a service, not just technology.

Fanatical Support is our philosophy, our credo. It reflects our desire to bring responsiveness and value to everything we do for our customers. You will experience Fanatical Support from the moment we answer the phone and you begin to interact with our employees.

Fanatical Support has made Rackspace the fastest-growing hosting company in the world. Call today to experience the difference with Fanatical Support at Rackspace.

rackspace™
MANAGED HOSTING

1.888.571.8976 or visit us at **www.rackspace.com**