The New York Linux Scene Journal

We are Do'ers Volunteers in the Public Interest

VOLUME 1 NUMBER 1 MARCH 17, 2002 www.nylxs.com

Journal Contents and Features

And MUCH MORE INSIDE

Executive Editor:

Michael Richardson

INTRODUCTION TO NYLXS

By Kevin Mark

NYLXS is an all volunteer organization dedicated to the continuation of open source, free speech protection and Linux. This is being accomplished through the development of leadership and low cost technical proficiency. We aim to convert schools, businesses, and non-profits by showing the cost, security, and organizational benefits of the open source revolution.

This will be accomplished with various initiatives: a 'free software' institute which includes low cost cradle-to-grave linux training, an in-services lecture series, a public school linux initiative, monthly installfests and online mailing lists. We have many meetings during the month were members help shape the development of our initiatives. We also have online mailing lists which include lively discussion of linux HW/SW questions, free speech issues, open source politics.

Install fests are free invitations for people new to Linux to BYOB (bring your own box) and have gurus from our membership help you get you on your way to the freedom of linux , maybe have a beer (virtual or otherwise) and mingle with others in the open source community.

Our in-service lecture series provides opportunities to learn about current open source technologies. These are informal meetings where questions are welcomed and you learn something cool! We find IT professionals who are using open source in the field and have them present solutions to diverse problems.

Afterwards, the lecture is posted online and discussions is encouraged on our mailing list. Our Free software institute provides low cost classes on open source technology. current classes starting are:

Linux 1 starting, March 30, 2002. Unix 1 starting, March 18, 2002. Perl 1 starting, March 21, 2002. Perl 2 Starting, March 26, 2002. Visit our website for upcoming class.

As part of our public school initiative, our members are speaking to officials in the New York City School system about getting open source in schools. Our current volunteer project is helping high school students coordinate an after-school linux training course and with administration of the students' Linux machines.

The New York Linux Scene Journal - The Road Ahead

by Michael L. Richardson - Executive Editor

The New York Linux Scene Journal's main goal is to Record, Educate, Inform, and Promote Linux in the New York Area (for those of you not from Brooklyn that means the rest of the World) The Journal will be published quarterly. It will be on-line and at least one issue will be printed.

We are calling for papers in the NYLXS Journal. The Journal's looking for serious papers on Free Software research or implementation, programming practices and desktop usage's. If you work with free software, Linux, Perl, Lisp, PHP, Database design, Firewall Network. Get yourself published and increase your profile.

We need the history of Linux (as seen from those who were there), Stories from those who found Linux later, how the Linux Users Group you belong to got started and the group's history. This includes Sigs, Lugs, High Schools, Colleges and Universities and business that use Linux or Free Software.

Visions of where you see Linux, Free Software, and GNU going. How has Linux made a difference to you, your school, your job, your business and your life? In short we need to explore the world of Linux and Free Software (were it's been, were it's at now, and were it seems to be going).

Be a part of the newest and hottest group to hit the Linux and Free Software world. To those who wish to help (or just see their name in print) please e-mail Michael L Richardson at miker@nylxs.com

TABLE OF CONTENTS

Committee Reports:

Lobbying Committee Report:

Ruben Safir

NYLXS was created with the intention to pave the way for fundamental changes in which computer users interact, work with, and relate to their computer systems. With an increasing amount of communications and information gathering taking place in the digital realm, the impact of politics, regulations, the market place, and the law deeply affect the mission of NYLXS and Free Software.

Free Software can play a key role in the shape of our future as a society, in protecting and enhancing our democratic values, improving our civic freedom through empowerment, regardless of economic or social standing. But to assure this future, Free Software advocates must now act to protect long standing political rights which are now under siege. Long standing assurance of property rights, freedom of association, and free speech rights are being curtailed, and threaten the existence of Linux and other Free Software initiatives. In response, NYLXS is lobbying on the State, Federal and local level to protect Fair Use, open standards, the right to read.

Our lobbying committee is spearheading New Yorkers for Fair Use, participating in demonstrations to Free Dmitry Sklarov, and in protesting the 1998 Digital Millennium Copyright act. We have aligned ourselves with librarians in our Save the Libraries campaign, and have met with Congressional members and staff in the pursuit of representing Free Software in the political arena.

With digital rights management reasonably curtailed, Free Software assures privacy in your home and office, and we have the best chance to end the digital divide which hinders our growth as a people.

Join NYLXS, and help make a difference today.

NYLXS - Political Action Report

By Ruben I Safir

When NYLXS was founded a few months ago, I released a mission statement for the proposed volunteer organization. Chief among the growing needs for the Free Software movement in NYC was the increasing need for a professional approach to the political issues which Free Software and Linux face daily.

Both the Linux community and the community at large requires a certain set of guaranteed freedoms, which anchor us as free people. As a result of this need, Linux advocates find some of their best allies among unexpected constituencies such as professional librarians, artists, engineers, academicians, educators, and computer hardware vendors. Simply, the extension of our current political rights into the digital environment, without modification, in a attempt to extend untraditional copyright control, protects both Free Software and society as a whole. The political agenda of NYLXS, and its sister organization, New Yorkers for Fair Use, is mostly about keeping the status quo.

NYLXS realizes that it's probably shocking to a large number of people that we are for the status quo. The public often fails to understand our exact position on the issues with regard to government-granted limited monopolies (copyrights) and fair use. Our organization is actually very conservative. It is mainstream in its outlook. We are demanding protection of the status quo. Clearly we have been painted by our political foes as being radicals, and as such with a radical agenda. Radicals want to change the world. Nylxs's political agenda is not to change the world, but to maintain the exceptional outstanding guarantees of freedom and universal suffrage which every decent democratic civilization today demands from its government.

NYLXS is content to allow society to evolve freely, and let technology, economics and civilization in the public commons be the sole forces for broad sociological change. We oppose radical organizations such as The American Association of Publishers who demand the end of public libraries, and the Motion Picture Association of America, who demand wiretapping in every digital device. And when we focus on these issues, we are certain that the broad swath of American opinion, as people come to understand the issues, stands firmly on the conservation of the status quo, which is under assault from many radical groups.

Understanding this, NYLXS helped move forward the ground-breaking work by NY Fairuse on these issues. Over the previous summer and fall, in 2002, NY Fairuse started a political campaign targeting local congressional delegates in the hopes getting them to recognize the growing threat to the status quo, especially the threats presented by the Digital Millennium Copyright Act of 1998. The campaign was designed around good old NYC machine politics, going door to door with petitions, targeting supermarkets and public areas with leaflets, and shaking hands. They targeted the home district of Brooklyn and Queens Representative Anthony Weiner, and Senator Charles Schumer, both of whom reside in the Park Slope section of Brooklyn. They collected nearly 500 hundred signatures, gave out over 2000 buttons with "Save the Libraries" on them, and contacted nearly every resident in that area of the district. They pushed on through the destruction of the World Trade Center into October and the local election season. When the time came to go to Washington and deliver the petitions and the message, NYLXS was pleased to help fund the trip to Washington for NY Fairuse.

Members of both NYLXS and New Yorkers for Fair Use, boarded the NYLXS bus early Tuesday morning, February 19th, 2002 and headed down to Washington DC. The bus was arranged for by NYLXS First VP Marcus Conti. On the trip were President and NY Fairuse Founder, Ruben Safir, NYLXS Secretary Richard Weinberg, NYLXS Public Relations Committee Chairman Joe Grastara, NY Fairuse Secretary Seth Johnson, NY Fairuse member Forest Mars, and the Nylxs's Free Software Institute Chairman, Paul Rodriguez. Finally getting underway at 9:00AM, the group arrived in Washington DC around 2:00PM. Our main contact on Weiner's staff was Legislative Director Lamar Robertson, who oversees the policies concerning the Intellectual Property sub-committee of the House Judiciary committee. Lamar was on emergency child care duty that afternoon, so instead they met with Kevin Ryan, Weiner's Chief of Staff. NY Fairuse members were invited into the Congressman's office in the Conrad Building just across the street from the Capital Building.

They spent about an hour with Chief of Staff Ryan, explaining the importance of Fair Use to the development of the modern computing industry. Head spokesman for the group was Reuvain Safir. He methodically laid forward the arguments for maintaining the status quo in the relationship between citizens and the information they own. The threat to this by the Digital Millennium Copyright Act was particularly emphasized. He covered the foundation of the current computer industry with the reverse engineering of the IBM PC by Compaq, paving the way for standardized hardware, fair competition, expanding markets, and the growing high tech industry.

Underlining this issue, Ruben presented the case of the copyrighted computer bios software. This software is needed for booting the computer. A DMCA protected encryption of the PC bios would end free software immediately under the current DMCA provisions. It would make running Linux a felony, just like playing a commercial DVD on Linux today is likely felony. Ruben emphasized they need to prevent us from becoming share-croppers to our own cultural heritage, and how libraries, municipal governments and minorities were all being adversely affected by the change in the status quo by the DMCA.

Seth Johnson made the point that other bills have been floated by various members of Congress which have confused the usage of the word 'security'. He reminded Chief of Staff Ryan that computer security and content protection systems are completely different issues despite the fact that such content protection systems have been termed 'content security' systems. He emphasized that national security needs of the nation and general computer security can not be obtained by methods which are designed to prevent music file sharing, or the playing of DVD's on Linux. Paul Rodriguez noticed that the Congressman was running Linux in his office with the TIVO video system which was perched above the door opposite his desk. The group was told that the Congressman loved the TIVO. He loved the fair use he makes of broadcast video through the TIVO. In fact, it became very apparent that the entire Weiner staff leverages traditional Fair Use through digital means in a way which would make most organizations do back flips. They record volumes of broadcast media and share them in whole for political reasons with allies and constituents. They archive extensive material from local newspapers in an in house database for later printing. They freely distribute this information as part of their political activities. They freely leverage a digital environment to remain informed and one step ahead.

While in the office, Ruben also showed off his Sony picture book running Linux, Ximian and playing MP3s. The purpose of the demonstration was to show that real industry in Free Software exists in NYC, and it needs protection. Kevin enjoyed the MP3s, and the group discussed momentarily the Napster issues, and how DVD's were illegal to play under Linux. They discussed the huge copying of DVD's and the copyright violation which took place in Queens this month, with hundreds of thousands of dollars of bootleg DVD's being made available by direct copying of the disks, contrary to claims by Time Warner which justified the use of software to prevent Linux from reading the DVD's. Standard Dvd's read the encrypted DVD's by use of their 'Content Security System' encryption system. This system controls access to the information on DVD. Time Warner claimed that CSS made this kind of copying impossible and that the Linux programs used to read DVD's were illegal. Yet it is obvious that a bit by bit copy is indeed possible and allows for copyright violations. Only legitimate use of the DVD's is affected by CSS. After the discussion the NYLXS group left Weiner's office.

As the group left, we were called by Lamar from his home with an invitation to have dinner with him and his family. Since Lamar Robertson is the key man for Weiner on these issues, the members took him up on his offer. They headed to his Maryland home. As soon as the group hit the door, Lamar was armed with a NY Times article on the Supreme Courts review of the Sony Bono amendment of the copyright act. Ruben pointed out, with much excitement, that not only was this one of the issues that New Yorkers for Fair Use was fighting for, but in addition, that Lamar's use of the complete NY Times article, which was a copy of the article in total, was exactly the kind of fair use which we needed to protect for all individuals if we are to have a free and open society. The fact that a politician's Legislative Director is making fair use copies of such a digitally available articles outlines, as clear as anything, exactly how serious a blow the DMCA is to the status quo, and how it jeopardizes the political freedoms of all US citizens in their ability to access information in the course of their normal lives and in their participation in government.

The group stayed for nearly 4 hours at Lamar's residence, enjoying scotch and getting to know each other. Each member of the group introduced himself, and said his piece. The conversation became very detailed, and Lamar took many notes on the issues which were raised. One of the issues brought up was the protection of public libraries, and particularly the radical position of the A.P., and Pat Schroeder. It was pointed out that the A.P. has suggested we close down libraries and replace them with Barnes and Noble stores. It was reviewed that Judith Platt compared librarians with Waco styled terrorists. And NYLXS went over its interaction with Pat Schroeder, the A.P. spokeswomen. Pat claimed that college students who said that file swapping was ethical were made more sensitive to the issue when comparing music to beer. At the 'Text Zero One' conference in Brooklyn last year, Mrs. Schroeder claimed that if everyone could duplicate beer like music, students would understand how distributing copies of beer for free would be stealing. At this point, Ruben Safir, who was in the audience, asked Mrs. Schroeder if she was seriously saying that if mankind was able to reproduce food as freely and inexpensively as we can with music, would she find it immoral to stamp out world hunger on the basis that this was stealing from farmers?

This story was outlined to Lamar, giving the room some comic relief. Lamar made the point that in the sub-committee no one really understood the issues well, and that most of the members were consistently dependent on ASCAP for testimony on these issue. NYLXS and NY Fair Use has pledged itself to help the Congressman's office gain a broader and more detailed policy sheet on these issues, and to come to Washington when asked to testify for the sub-committee authentically and professionally. We expressed that we hope that this is a start of a long term relationship. Ruben and Lamar hugged, and the group headed back to Brooklyn feeling that much has been accomplished in cementing the foundation for another voice in Congress.

We hope that Representative Weiner adds his voice to that of Representative Boucher, and will fight to retain the status quo in the relationship that individuals and groups have with the information that they own.

Committee Reports

Greetings New York Linux Scenesters and Hangout List Members

by Joe Grastara - Chairman of the Public Relations Committee

My name is Joe Grastara and I am serving as the chairman of NYLXS Public Relations Committee. Just to give you a little bit of background about myself, I am 22 years old, a recent graduate of New York University and am currently employed as a Web Developer for the NYU School of Medicine. I first became interested in Linux about two years ago when I was working as an intern at my present place of employment. I had to do a good deal of my work on a UNIX Machine and quickly came to prefer that environment. I decided that I wanted to learn more about UNIX, and given the limited budget of college student and the x86 machine I already had at home Linux was a very natural choice for me. After a couple of weeks of using Linux at home, I convinced my boss to allow me to install Yellow Dog on my Mac at work. Since then I haven't used anything except Linux at home or at work, with the exception of Mac OS X on the iBook I recently acquired (sorry, I know it's not free but I like it).

So enough about me, and on to plans for the Public Relations Committee. I see the mission of the Public Relations Committee as being to get as much public attention as possible on the activities of NYLXS. To that end, some tasks of the Public Relations Committee will be to forge relationship with people in the News Media, and notify them of NYLXS activities which might be news-worthy, to develop relationships with other Linux/Free Software groups and keep them notified of NYLXS activities, and generally anything to get the message out there.

So now here's the part where I beg for help. HELP! Seriously though, I am looking for volunteers to help with what I imagine will be the enormous task of keeping the rest of the world informed about all of the many undertakings of NYLXS. If you would like to help out send me an email at joeg@nylxs.com.

How I Got From Art to Linux

by Marco Scoffier

My name is Marco Scoffier, I come from an Arts background, though I did take quite a few physics and engineering courses before becoming a sculpture major in college. I graduated in 1995 so the internet was just coming into wide public knowledge, but I remember telnet'ing into the school servers for email and newsgroups, also finger'ing, and talk'ing with friends over the unix servers, so I had some exposure to the wonders of the Unix environment. Now it seems since everyone has a hotmail account, no undergraduate is even exposed to unix.

For the past five years I lived in France where I had studio space in France's largest art school, and taught Audio and Video (mostly mac applications). I also ran the lab which bought all the equipment (video camera's , slide projectors, photo camera's, extension cords etc.) which the student's could borrow to take home. As artists tend to want to reinvent the world, I worked on some rather more technical projects, such as programming an interface to control editing VTR's through the mac serial port, building midi interfaces (yes soldering iron) for interactive installations, and building all sorts of speakers and microphones.

I spent enormous amounts of time teaching myself all sorts of things on the net, and came across Eric Raymond's "Cathedral and the Bazaar" and "Homesteading the Noosphere. "*flash of light* I convinced a local vendor (a Vietnamese man who built his own boxes) to trade the cost of a MS license for SUSE 6.2, and double checked all the hardware to make sure it was compatible. (I'm still using this same box, just more memory, and quieter fans). *I was off* To tell you the truth I was amazed at how advanced the operating system was, and all the tools I got for free. I thought I was getting a command line and compilers which would make it easier for me to learn programming. I really wanted a lisp interpreter as I had read many books on artificial intelligence (especially Douglas Hofstader and Melanie Mitchell's Copycat project) .

I got a full fledged workstation with more tools that you could use in a lifetime. To get myself up and running with free software, I have bought a huge number of books (many from O'Reilly). I started with "Running Linux" which refuses to age though it was written more than ten years ago, and my favorite "Unix Power Tools" which is a treasure trove which helps you get into the mind of brilliant users of Unix. Once you have a few of these ways of working under your belt, you really understand the power of this operating system which also refuses to age. Linux has taken over almost all of my computing needs.

All of my networking, of course. I have had it running as a gateway, appletalk server, mailserver, webserver, and dns server for my small group of machines (I mean to get a timeserver running, but haven't yet). It has zapped ads for me and cached web pages when I was on an excruciatingly slow dial-up. But linux is also taking over most of my multimedia needs...

I have used the Gimp extensively, especially to clean and compress photographic jpegs for the net, as you have much more control over this than in any adobe product. I have also used perl and the Image Magick libraries to manipulate directories full of images, as well as a bit of the perl interface to the gimp (though there is much more I could do with this).

I have used Blender to make 3d animations to explain my rather complex installations -- there is no way I could have completed these animations any other way. Blender I am convinced will blow away anyone using 3d, motion graphics, or even video editing. It has a full python interpreter inside and an api which allows you to control any piece of your 3d world, and automate whatever you can dream of. Just recently, I captured DV video through a firewire card, and am working on the fine points of getting this footage into DivX for presentation on my website, Mpeg2 for burning onto CD's and eventually onto a DVD. I have already gotten the DV footage into Blender where I can use it as a background to pass titles over, mix it with 3d objects or use it as a texture (using it as a texture for a lamp makes an excellent looking video projection in a 3d world).

Free Software Institute

by Paul Rodriguez

My name is Paul Rodríguez. I am 22 years old and am taking a leave of absence from Hampshire College where I was studying Comparative Religion and Political Science. Over the past three years I have been learning about Linux and the issues revolving around Free Software and Fair Use. I have become very passionate about the potential of Free Software. As well as the potential of it's adoption and use. Central to the fulfillment of this potential is an understanding of the technology and the human factors which determine its use and development. I first heard of Linux while helping to administer funds for my university.

A group of students (our in-house LUG as it turned out) came to me asking for funding to distribute free cds with different Linux distributions on them. When I took a leave of absence from that school, I went to work with a variety of NGO's and local organizations in Brazil. These were organizations with little financial backing, all of which were planning and implementing extensive educational initiatives. I became interested in their information infrastructure.

While I didn't have the technical expertise at the time (and frankly, still don't), I began to realize the enormous potential of free software in a great variety of environments. As it so happens, Brazil has a very active and growing base of Linux users (as evidenced by 18 year old Marcelo Tosatti, the new maintainer of the Linux kernel). Ok. That's some background on me. And now to the really juicy stuff.

The Free Software Institute is an initiative of NYLXS. Its goal is to promote Linux and Free Software through education and training. This meant to be a service to the Free Software community. We are putting together a complete "cradle to grave" Linux educational program. These classes are designed to be taken together to get a well rounded base in Unix administration.

This includes use, installation, programming, and system and network administration. With the funds generated from these classes we are looking to allocate space that will be available to the various Free Software groups in the area. Securing a space for the community to meet, learn, and discuss. We currently need people who are willing to share their knowledge and are willing to teach! I am also looking for people willing to help organize this initiative.

-Paul Rodríguez sangrelinux@yahoo.com 201-864-4323 You can also IM me using AIM at SangreToro

NYLXS Inservice Tech Lecture

by Ruben I Safir

The New York Linux Scene was established to development leadership within the New York Area in Linux related technologies and development. In order to accomplish this, NYLXS is dedicated to volunteerism and education. The most important need for those of us in the computing field, either professionally or hobbyists, is the presentation of new areas of discovery and development being explored by memberships of the Linux fraternity. We are kicking off our inservice program, which hopes to bring recent research and developments to peers in a useful and productive manner. The first presentation will be my good friend and co-worker Brian Sietz.

Brian, a High School award winner in programming, is a Senior Computer Science major students at NYU. During his time in NY he worked for 2 years at the NYU Dental School where he developed his skills in Web programming, Perl, terminal programming, database development, Apache system administration, object orientated programming, and system administration. Later he moved on to a now defunct startup called Scalable Technologies where Brian extended his skills to include PHP, shell and some advanced C. Currently he is finishing up at NYU, and expects to enter the full-time job market in the very near future. Anyone who hires him is getting far more than they pay for.

IRCBOTS

by Brian Seitz

Ever since I had started using IRC, even before I was a serious programmer, I had a fascination with bots. They can be very simple programs to write, but the social aspect of interacting with your friends and a pseudo-A.I. makes it more exciting than just writing a text-based program for yourself. I had just finished working on an IRC bot using Perl's Net::IRC module. RokerBot, as it is called, queries a national weather website for forecasts and sends them to an IRC channel. I mentioned the bot to Ruben Safir in passing. He thought it would be a good in-service topic, so I wrote two more example programs and a tutorial.

While what I had written could have been a long, boring lecture, the NYLXS audience was interactive from the start. Giving out copies of the tutorial and code helped everyone follow along and ask questions as we built IRC bots from scratch. To me, it felt like a good college class in which both the students and the professor really want to be there. Much of the example code depended used other Perl modules such as LWP and HTML::Parser.

While we didn't study in depth about these sections of the code, I tried to show the power of Perl by using these modules. It's one thing to write "hello world" in some particular API, it's quite another to write a useful application. Net::IRC is also a good introduction to event-driven programming, which is used by GUI toolkits like GTK+

Notes from NYLXS's first Inservice

What is an IRC bot?

By Brian Seitz (brian@wholok.com)

Automated client

Examples

Etiquette What you will need to program an IRC bot

Perl

Net::IRC module

An IRC server (own for debugging)

Other software for functionality

A very simple bot: HelloBot

What HelloBot does

Creating $irc object

Creating $conn object

Event driven programming

on_connect

on_join

irc->start()

Handling other common events: RokerBot

What RokerBot does

on_public

on_msg

Other events in Event.pm

TriviaBot: using $irc->do_one_loop

What TriviaBot does

Handling timing

Parsing commands

Resources

What is an IRC bot?

An automated client: To an IRC server, an IRC bot is virtually indistinguishable from a regular IRC client (i.e., a person using a program such as X-Chat, mIRC, or ircii). However, there's no person typing behind an IRC bot. It only makes automated responses, based on (usually) what is happening on IRC. An IRC bot can do things based on public messages, private messages, pings, or any other IRC event. But a bot isn't limited to the world of IRC. It can talk to a database, the web, a filesystem, or anything else you may imagine.

Examples: Here are some common IRC bots that you may have seen in your travels already: File serving: This type of bot emulates an FTP program by interfacing with a filesystem. Users talk to the bot using private messages with commands like "ls" and "get". The user can send and receive files using DCC (a part of IRC that allows the initiation of peer-to-peer file transfers). Channel administration: This bot maintains a list of channel ops (people who run the channel) and makes sure they stay in control of it, even if individual people are disconnected. They may also kick people from the channel who violate its etiquette (e.g., talking in all caps, using colors, flooding, etc.) Games: Some bots will allow the people in the channel to text-based games. We'll learn later how to program a trivia bot.

Etiquette: Many IRC networks restrict the use of bots. When you log into an IRC server, check the MOTD (message of the day) to see what the policy is on bots. If you violate this policy, you may be banned from the server. In practice, however, bots aren't automatically detectable. As long as your bot doesn't join an established channel and annoy people, you probably won't get in trouble. But IRC channels are free; make your own, invite your friends, and have a ball. If you want to be really safe, you can run your own IRC server. If you're planning on extensive debugging, running an IRC on localhost will save you lots of time anyway. Check out generic IRCd and DALnet's Bahamut IRCd. These are very stable programs and should compile easily on a UNIX system, however the specifics of configuration are beyond the scope of this document. What you will need to program an IRC bot Perl You can program in IRC bot in any language. Some IRC clients offer their own scripting languages that add automated functionality to your normal IRC session. However, we'll be using Perl. Net::IRC Module Net::IRC is Perl module that talks the IRC protocol, abstracting most of the low-level details of programming an IRC bot. It primarily works as event-driven programming (we'll get to that later). You can get Net::IRC from CPAN. IRC server You'll need to find an IRC server to connect to. A fairly comprehensive list can be found at irchelp.org. However, for debugging purposes, you may want to run your own IRC server. Other software for functionality No IRC bot is an island. While some IRC bots, such as a channel administration bot, can be useful without any interface to the outside world, we'll be using some non-IRC-specific software to make our bot more useful. To program use the bots in this tutorial, you'll need the Perl modules LWP (Library for WWW access in Perl), HTML::Parser, and Text::Wrapper. For TriviaBot, we'll be using the MySQL database and the Perl DBI module. While learning about IRC bots, you may also find the Data::Dumper module useful. Go to CPAN to get these if you don't have them already.

A very simple bot: HelloBot

What HelloBot does HelloBot is a greeting bot. When a user, let's give him the nick "Joe," joins a channels, HelloBot will say "Hello, Joe!" When Joe leaves, HelloBot will say "Goodbye, Joe!" (Of course, since Joe has already left, he won't see the message, but other users in the channel will.) (02:03:53) HelloBot (brian@L36-72.DATANET.NYU.EDU) has joined ##

(02:03:53) <HelloBot> Hello everyone!

(02:03:53) <HelloBot> Hello, HelloBot! (02:04:20) Harry (brian@turtle.wholok.com) has joined ##

(02:04:20) <HelloBot> Hello, Harry! (02:04:25) Sally (brian@turtle.wholok.com) has left ##

(02:04:25) <HelloBot> Goodbye, Sally! (02:04:38) Harry (brian@turtle.wholok.com) has left ##

(02:04:38) <HelloBot> Goodbye, Harry!

Creating $irc object The first thing we need to do is create the $irc object. The $irc object can handle more than one IRC connection, but for this tutorial we'll just be using one. So, let's do it:

use Net::IRC;

create the IRC object

my $irc = new Net::IRC;

Creating $conn object Now we need to create a connection object. This requires quite a few arguments, which should make sense if you're familiar with IRC. Our interactions with the IRC server will be through the $conn object.

Create a connection object. You can have more than one "connection" per # IRC object, but we'll just be working with one.

my $conn = $irc->newconn(Server => shift || 'turtle.wholok.com', # Note: IRC port is normally 6667, but my firewall won't allow it

Port => shift || '80',

Nick => 'HelloBot',

Ircname => 'I like to greet!',

Username => 'hello'

);

We're going to add this to the conn hash so we know what channel we

want to operate in.

$conn->{channel} = shift || '##';

The "shift ||" parts allow us to change these parameters on the command line.

For example, I could run: ./hello_nick.pl irc.slashnet.org 6667 #mychannel

Event driven programming So how does HelloBot know when someone joins the channel? In an iterative program, you would poll the IRC filehandle, parse the input, and call a subroutine if you have a join or leave event. Using Net::IRC removes the first two steps. You merely have to decide which events you want to handle, and define subroutines to handle them.

on_connect The first event we have to handle is connecting. We know that we're connected when we get the end of MOTD event, which is normally sent by an IRC server. This event is given the value "376". When we get the event, what we want to do join our channel and greet the channel. Here's how: sub on_connect {

shift in our connection object that is passed automatically my $conn = shift;

when we connect, join our channel and greet it

$conn->join($conn->{channel}); $conn->privmsg($conn->{channel}, 'Hello everyone!');

$conn->{connected} = 1;

}

The end of MOTD (message of the day), numbered 376 signifies we've connect $conn->add_handler('376', \&on_connect);

In the on_connect subroutine, we introduce two methods of the $conn object, join and privmsg. join simply takes a channel as an argument. prvimsg is a bit more complicated. It is used to send both public and private messages (despite its name). If you put a channel name as the first argument, you will send publically to a channel. If you put a nick, you will send privately to the person with that nick. The second argument is the string you want to send. After we've defined our subroutine, we use the add_handler method to link it to the event we're handling. The first argument is the event identifier, which is the string "376," not the number 376. It will not work if you send it as a number. Most of the popular events are identified by alphabetic strings, as we'll see later. The next argument is a reference to a function.

on_join

Next, we'll set up our join handler:

sub on_join {

get our connection object and the event object, which is passed

with this event automatically

my ($conn, $event) = @_;

this is the nick that just joined

my $nick = $event->{nick};

say hello to the nick in public $conn->privmsg($conn->{channel}, "Hello, $nick!"); }

$conn->add_handler('join', \&on_join); The new thing here is the $event hash that is passed to the function. Most event handlers will take this as an argument (and it's the only argument event handlers will take, not counting the object, of course). Depending on the event, the contents of this hash will vary. Later, we'll see how to print out the contents of $event to learn how to program for different events. The only part of $event we're interested in here is the nick that initiated the event. This is held in an obvious place, as we can see. Next, we say "Hello" to the nick.

$irc->start() Now that we've set up our handlers, we're ready to start IRC. This is done with

the simple command:

$irc->start(); Now we should be connected and ready to roll. Notice that we're using the first object we created, $irc, and not the $conn object. This is because Net::IRC can handle more than one IRC connection in a single instance.

Handling other common events: RokerBot

What RokerBot does RokerBot (you'll also need WeatherGet.pm and WeatherParse.pm) retrieves weather information from http://www.weatherunderground.com/auto/roker/ and sends it to IRC. It does this whenever a user says "!roker (city), (state)" (02:03:40) RokerBot (roker@turtle.wholok.com) has joined ## (02:03:40) <RokerBot> booya! it's Roker time!

(02:05:58) <wholok> !roker New York, NY (02:05:59) <RokerBot> wholok, it's 50 F in New York, NY right now. Wind is NNW at 8 mph. Humidity is 61%. Roker would say it's Clear. Tonight Mostly clear. Lows in the lower 40s. North wind 10 to 15 mph becoming light. Thursday Mostly sunny. Highs in the upper 50s. Light and variable wind becoming south and increasing to 10 mph. (02:08:55) <wholok> !roker adsfoi34 234lkndf (02:08:57) <RokerBot> Unable to retrieve weather. Sorry, wholok.

on_public This is a new event for us to handle, the public message event. Here's how

RokerBot handles it:

sub on_public {

on an event, we get connection object and event hash my ($conn, $event) = @_;

this is what was said in the event

my $text = $event->{args}(0);

regex the text to see if it begins with !roker if ($text =~ /^\!roker (.+)/) {

if so, pass the text and the nick off to the weather method my $weather_text = weather($1, $event->{nick});

wrap text at 400 chars (about as much as you should pt

into a single IRC message my $wrapped_text = $wrapper->wrap($weather_text); my @texts = split("\n", $wrapped_text);

$event->{to}(0) is the channel where this was said

foreach (@texts) {

$conn->privmsg($event->{to}(0), $_);

}

}

} $conn->add_handler('public', \&on_public);

Here there are two new parts of the $event hash that we are dealing with. The first one is $event->{args}. This is a reference to an array. The first value in this array is the text of the public message. We regex this text to check if it begins with !roker. If so, we pass off the rest of the message to the weather subroutine, which will return our weather text. The other new $event hash part we are looking at is {to}. This is an array reference of each entity the event was sent to. In our case, we simply want our channel name, so we take the first part of that array. From there, we wrap our text to a reasonable limit (sometimes the weather reports can be long), and send each line to the channel.

on_msg Our next event to handle is msg. An msg is a private message sent to the IRC bot directly. It is best to use msg events to interact with your bot if there are going to be a lot of interactions that could annoy other users. There's no practical difference in using msg as opposed to public messages, so here's a

quick example:

sub on_msg {

my ($conn, $event) = @_;

Under normal circumstances, simply reply to the nick that there's

no reason to /msg RokerBot. $conn->privmsg($event->{nick}, "Don't nobody /msg RokerBot.");

}

$conn->add_handler('msg', \&on_msg); Obviously, you could make this interaction more useful, but the example shows

the concept.

Other events and methods Check out the source code of Events.pm (or do a perldoc Net::IRC::Events) to see what other events an IRC bot can handle. When experimenting with these events, you may want to use the module Data::Dumper to see what the event is sending you:

use Data::Dumper;

sub default {

This is helpful to see what an event returns. Data::Dumper will # recursively reveal the structure of any value

```
my ($conn, $event) = @_;

print Dumper($event);

}

# experiment with the cping event, printing out to standard output $conn->add_handler('cping', \&default);
```

Pretty much any IRC client command is a method of IRC::Connection. If you look in the source of Connection.pm, you'll get fairly good documentation on how to call them. For example, here's how you can use the mode method to make someone an op:

```
$conn-$gt;mode('#linux', '+o', 'tux');
```

TriviaBot: using $irc->do_one_loop

What TriviaBot does TriviaBot (you'll also need Trivia.pm) takes questions from a database, asks them, and checks users for answers to the question. There's on tricky thing, though, it has to give a 10 second interval between each question. With our event driven model, we can't quite do this.

(01:52:30) TriviaBot (brian@L36-72.DATANET.NYU.EDU) has joined ## (01:52:30) <TriviaBot> booya! it's Trivia time!

(01:53:10) <Joe> !trivon (01:53:10) <TriviaBot> Trivia is on! First question in 10 seconds.

(01:53:22) <TriviaBot> Pick a hand.

(01:53:22) <TriviaBot> *****

(01:53:25) <Joe> le

(01:53:30) <Joe> left

(01:53:34) <wholok> ri

(01:53:34) <TriviaBot> ri***

(01:53:35) <wholok> rig

(01:53:35) <TriviaBot> rig**

(01:53:39) <Sally> right (01:53:39) <TriviaBot> Correct answer by Sally!

(01:53:39) <TriviaBot> right (01:53:39) <TriviaBot> Next question in 10 seconds (01:53:51) <TriviaBot> How large is an IPv6 address?

(01:53:51) <TriviaBot> *** ****

(01:53:56) <wholok> really big

(01:54:01) <Joe> 128 bits (01:54:01) <TriviaBot> Correct answer by Joe!

(01:54:01) <TriviaBot> 128 bits (01:54:01) <TriviaBot> Next question in 10 seconds

(01:54:12) <Joe> !score

(01:54:12) <TriviaBot> Score:

(01:54:12) <TriviaBot> Sally: 1

(01:54:12) <TriviaBot> Joe: 1 (01:54:13) <TriviaBot> Who invented Perl?

(01:54:13) <TriviaBot> ***** ****

(01:54:28) <Sally> Larry rall

(01:54:28) <TriviaBot> Larry *all

(01:54:32) <wholok> larry wall (01:54:32) <TriviaBot> Correct answer by wholok!

(01:54:32) <TriviaBot> Larry Wall (01:54:32) <TriviaBot> Next question in 10 seconds

(01:54:36) <wholok> ha ha Sally! (01:54:44) <TriviaBot> Who is the Linux mascot?

(01:54:44) <TriviaBot> ***

(01:54:50) <Joe> penguin

(01:54:56) <wholok> TUX! (01:54:56) <TriviaBot> Correct answer by wholok!

(01:54:56) <TriviaBot> Tux (01:54:56) <TriviaBot> Next question in 10 seconds (01:55:08) <TriviaBot> What TV network does Mickey Mouse own?

(01:55:08) <TriviaBot> ***

(01:55:14) <Sally> NBC

(01:55:14) <TriviaBot> *BC

(01:55:19) <wholok> ABC (01:55:19) <TriviaBot> Correct answer by wholok!

(01:55:19) <TriviaBot> ABC (01:55:19) <TriviaBot> Next question in 10 seconds

(01:55:23) <wholok> !trivoff

(01:55:23) <TriviaBot> Trivia is off!

(01:55:25) <wholok> !score

(01:55:25) <TriviaBot> Score:

(01:55:25) <TriviaBot> Sally: 1

(01:55:25) <TriviaBot> Joe: 1

(01:55:25) <TriviaBot> wholok: 3

Handling timing Our problem is fixed with the method do_one_loop. This takes the place of the start method. After establish our handlers, we call it like this: sub handle_trivia_loop {

my $conn = shift;

$_ = $conn->{trivia}{status};

STATUS: {

if (/^getquestion$/) { # get question from trivia object $conn->{trivia}->get_question(); # say the question, and show the hint repeat_question($conn); repeat_hint($conn);

last STATUS;

}

if (/^waiting$/) { handle_waiting($conn);

last STATUS;

}

}

}

while forever, handle_trivia, then do an IRC loop while (1) {

handle_trivia_loop($conn);

$irc->do_one_loop();

} Inside of handle_trivia_loop, we check timestamps if we need a 10 second interval. We also get a new question if our status tell us so. However, checking trivia answers is still done with the event driven model.

Parsing commands One of the hallmarks of an IRC bot that takes public command is the syntax "!command". This allows the bot to easily parse the traffic in the channel, since it´s very unlikely that someone would want to say a real sentence starting with "!".

```
sub on_public {

# two args are passed on events, the connection object and a

# hash that describes the event

my ($conn, $event) = @_;

# this is the text of the event, eg, what the person said my $text = $event->{args}(0);

# we first check to see if it's a command (if it starts with !)

if ($text =~ /^\!(.+)$/) { on_public_command($conn, $1, $event->{nick});

} # otherwise, we assume it's a guess to the current question

else {

# check the answers my $res = $conn->{trivia}->check_answer($event->{nick}, $text); # if we've got the right answer, then celebrate

if ($res) { on_answer($conn, $res, $event->{nick});

} # otherwise, repeat the current hint

else { repeat_hint($conn); }

}
```

}

sub on_public_command {

here, we get the connection object, the text of the command, and # the nick of the person who issued the command my ($conn, $command, $nick) = @_;

branch off given the nature of the command

$_ = $command;

COM: {

this turns trivia on

if (/^trivon$/) {

don't turn on trivia unless it's off if ($conn->{trivia}{status} ne '') { last COM; }

set status to wait 10 seconds $conn->{trivia}{status} = 'waiting'; # alert users that trivia is starting $conn->privmsg($conn->{channel}, "Trivia is on! First question in 10 seconds."); # save the current timestamp $conn->{newtime} = time; # set the amount we've waited to 0 $conn->{waittime} = 0; last COM;

}

if (/^trivoff$/) { # set status to nothing $conn->{trivia}{status} = ''; # tell everyone that trivia over $conn->privmsg($conn->{channel}, "Trivia is off!"); last COM;

}

if (/^score$/) { # we want to see the scores, so let’s have ’em handle_score($conn);

last COM;

}

}

}

Resources Get modules: Comprehensive Perl Archive Network (CPAN) Find a server: irchelp.org’s server list

HelloBot: hello_nick.pl RokerBot: roker.pl, WeatherGet.pm, WeatherParse.pm TriviaBot: triviabot.pl, Trivia.pm

Milestones in Linux

TUX AND THE VIRGIN

I was a Linux Virgin! The Strange and Sordid Tale of an Innocent Who Was Led Astray... and Loved It!

by Marcia A Baczynski

Like most people in this day and age, I use computers. And like most people, I'm frustrated by them. The illogic of menus, "Help" that isn't, applications that don't work and won't tell you why. I long suspected my computer was a tool in some sort of experiment being conducted by aliens or Bill Gates and I was the unwitting subject. I couldn't figure out why someone who is perfectly capable of checking her car's oil or changing a tire couldn't understand her computer. For years I wondered, "Must it be this way?" And for years, the answer always appeared to be yes.

That is, until I found Linux. It started out innocently enough. A conversation in the school cafeteria with a boy. A conversation about Operating Systems. He was a Linux user. Had been since '95. "What's an operating system?" I asked naively."What's Linux?" As a PR major, I had no knowledge of such things. "Well," he said, putting his arm around me (and winking to his friends), "let me tell you all about it..."

Over the course of the next few months, he lured me in with nice words and big ideas, never for a moment suggesting that I should ever actually get involved.

"I bought this book today, 'Cathedral and the Bazaar,'" he'd say. "You might be interested." Links to Slashdot articles started showing up in my inbox. Invitations to campus discussions on the Microsoft anti-trust case became opportunities for him to ask me out.

Shamelessly preying on my innocent affections for the First Amendment, he'd spend hours explaining hackers' free speech/free beer arguments. Before long, he'd insinuated Linux into my every thought about computers (mostly occurring when AOL froze because I had too many windows open on my Windows 95 machine). But in the end, there was just one thing that was my undoing.

THAT PENGUIN!!! Yeah, so okay. I switched to Linux because of Tux. I wanted to be able to wear Tux t-shirts to work without feeling like a moron because I didn't know anything about Linux. I wanted to get a stuffed Tux without feeling too much like a wannabe poseur who's so lame I can't even pretend to be a l33t h4x0r. I realize that this information becoming common knowledge means I'll never be taken seriously as a Linux geek, but at least I didn't do it to impress some boy.

Ahem ... Back to the story. A few months later... My first try with Linux involved a dual boot Windows98 and RedHat7.2 with KDE on my brand spankin' new 1.1 gHz Athlon. I had built the computer myself in an effort to understand what various pieces of hardware did and how they worked. Fortunately for me, my roommate Patrick is a Linux system administrator, and he was willing to help me pick out hardware and do the installations for me. Not-so-fortunately, my first attempt at using Linux was not much of a success. Windows98 was my default OS and I really didn't know what I was doing, so I pretty much stayed in Windows all the time. This lasted a few months, and in October 2001, I decided to wipe the slate clean and reinstall both Linux and Windows fresh.

Once again, I had Patrick do the installation for me, and I watched as he muttered about how frustrating dealing with Windows was. Then he came to installing Mandrake 8.1. I had heard that Mandrake was a terrific distribution for newbies, and sure enough, I was able to follow what was going on throughout most of the install process. Realizing this gave me renewed confidence that I could handle this Linux thing, and when Patrick asked me what I wanted as my default OS, I said what the hell... make it Linux. This was October. It was December before I booted into Windows again.

In this time, I completely fell in love with Linux and my experiences using it radically changed my perceptions of how computers work. The things I didn't know shocked my Linux-using friends, but I was a Windows end-user in every sense of the word. Until this point, my computer was a black box which as far as I knew was controlled by demons. Little did I know I could control them!

Here are some other key things I learned:

There's always more than one way to do something, and there are often many different applications you can use to do any one thing.

What you see in the graphical user interface is not necessarily what is going on in the operating system or in the program. The GUI is separate from the operating system. Before I thought the GUI and the OS were one and the same.

Command lines aren't as hard as they look. Really. And you don't *have* to learn them.

General "help" resources on computers don't have to be vague, uninformative and counterintuitive. Under Windows if I wanted to know how to do something, I was better off searching through menus than using the help. If I couldn't find it in the menus, I assumed it couldn't be done. "Help" rarely gave me enough information if my problem was at all complex.

Linux's help resources generally err on the side of too much information, and as they're often linked to active message boards, it's much easier to find an actual person who is willing to help you. It can be overwhelming, but after a short while, patterns start to appear and it's not that difficult to figure out what you need.

On a related note, I've discovered it's better to get an error message that gives you lots of information, rather than one that gives you very little information in order to avoid scaring you.

When a program crashes, it doesn't mean you have to reboot. Bugs and crashes aren't inherent and inevitable. (This doesn't mean that Linux programs are super-stable; they aren't. But they're not much different from Microsoft programs, and unlike with Windows, I've only once had Linux itself crash on me. When I ran Windows, it would crash on me about once per week, minimum.)

The most important change in my perception is that when something goes wrong with the computer, I believe there *is* a solution.

There are still some things that I miss from Windows. I've yet to find a word processor or spreadsheet program under Linux that I like as well as Word or Excel. As far as the Internet goes, there are still quite a few web designers that design for IE only, and Flash and Javascript are sometimes flaky under Konqueror. But I love the extra features Konqueror has.

I love the massive amount of configurability my desktop has. I love that I can just highlight something to copy it and paste with just a click of the mouse (a boon to someone who does loads of editing, like I do).

I like being able to try things on the command line and being able to choose from a plethora of programs. I like that there are programs like Evolution that mimic popular Microsoft applications, and that there are programs that are completely different.

I love that when something breaks, it's not just broken... that there are always a few more things I can try before giving up.

Perhaps most importantly, I love that when I need help, I'm not treated as if I'm a moron, and I'm learning that when it comes to computers, I'm really not the idiot I've been trained to think of myself as being.

Computers still frustrate me. There are days when I just want to beat my head against the wall because I can't figure out how to rip a CD or why my word processor isn't working the way I want it to be. But the frustration doesn't run as deep anymore. I no longer believe it's hopeless to try to fix a problem.

I no longer believe that people who know stuff about computers are actually part of some sort of secret cabal that is determined to destroy the world by employing these dread machines to make our lives difficult. There actually *is* a logic to it all, and it's not *that* hard to figure out. And if I, the AOL-and-Windows95-using public relations major can figure it out, anyone can.

Can I get my giant stuffed Tux now?

How to avoid carpal tunnel syndrome (or what has the command line done for you lately?)

by Marco Scoffier

First, this is not an introduction to basic unix command line commands. (Feeling jipped? --sorry) There are many such introductions online (Unix is a Four letter word which is quite good), but quite frankly, knowing how to copy a file from one directory to another using cp rather than drag-n-drop isn't ever going to get you to prefer one method over the other.

In this article, I have tried to compile (no this isn't' about C either) a collection of shortcuts which should accelerate your daily usage of the command line. Why? because unless something is a pleasure to use, and better than other alternatives... you're never going to use it!

Unless you have a little bag of shortcuts and tricks, you will never be happy using the command line and you will never understand how someone could prefer it or even love it. (If you are already convinced of the power of the command line skip down here . If you are skeptical, read on).

Now, I was a long time mac user and administrator before converting to unix. You might think that this would make me drag-n-drop happy, but as anyone who has used macs in any professional capacity knows, you memorize every key-combo that exists to reduce your mouse-mileage and accelerate your work. (Speaking of mouse-mileage on the Gnome desktop there is an Applet under Amusements-> Odometer? It does just that...)

Now knowing the key combos on Mac power applications can take you so far, but you have not seen anything like the command line. The unix shells (the programs which run the command line) represent probably the longest running programming effort yet, built into these shells are years of short cuts developed by the world's best programmers. (Now I did say shells, yes there are different ones and they do treat some things differently. In this article I'm basically talking about bash. It is one of the more full featured shells, and usually the default for Linux. if you're running linux, and you haven't tinkered too much with your system, there's a huge chance you're running bash.)

Now if the command line is so powerful and so great, why aren't more people using it? Well, there are many answers to this question, many very long answers running the gamut from questioning the media empires and how we get our information, to our education system, but to make this a bit shorter, I want to focus on one issue I find less discussed than the rest: the difference between computer people and other people.

The command line was written by computer scientists for computer scientists right? Often these "techies" are made out to be another breed all together, but are they really a different type of person than the general population? I think not, but bear with me.

In general, it is safe to say that computer scientists are a bottom->up bunch. By bottom->up I mean these are the types of people who get excited by the idea of speaking machine-code (or klingon), and to the outsider, don't seem to get too far from obscure ruinations about bits, and clock cycles, and big-O's. Top->down people like writers or even top->float_around_in_the_nanoshpere people like artists, get exasperated seeing these guys who are hopelessly confusing, and can't get past tiny little joys that only they can understand anyway.

I think it is safe to say that top->down people get exasperated when they are faced with a bottom->up person's interface. And bottom->up people can't understand why a top->down person who can't understand the first step of some process such as allocating memory, wants to know the big picture, such as why does my mac keep crashing. What we have here is a profound failure to communicate.

What neither side realizes is that they are actually much closer than they imagine. Why do top downers use the mouse? Because they are lazy. And the command line seems difficult to learn. Why do the bottom uppers use the command line? Because they are lazy. They know about the mouse, but know that the command line is the lazy man's dream come true. Perl people will try to convince you that they use the language with the most laziness built in, but laziness has a long history in computer science, there is a story about how Richard Stallman started the whole Free Software movement because HP refused to give him access to the source code for a printer driver update, which meant that he would be forced to walk down the hall to check if his printouts had come out of the printer, rather than query the printer form his desk... but that is another story, and you want to know how you can get lazy with your command line now. So here goes.

Laziness No.1: I'm sick of typing the same stuff all the time. And typos... I hate typos....

Solution No.1 : (TAB) (also known as tab completion)

Ahhh tab completion.... the single greatest input accelerator invented in human computer interaction. You're going to get so in the habit of using tab completion, you'll want to tab-complete everything (some of you already know what I'm talkn' about right?) Example: how does it work?Let me start with something which may already be familiar to you. Many browsers will try to complete a URL for you when you start to type in an new address. For instance you type in www.goo... the browser will fill in www.google.com because you visited the site yesterday. The browser will not however compare what you are typing to every web-site that exists. The command line does. It completes what you are typing with every command in your path, or every file in your system which could possibly match what you have already typed. Examples:

You want to open the web-page you have been writing but don't remember exactly what you named it. You type: vi (TAB)this will list all the files which vi could open, and any subdirectories of the directory you are in. You remember you named the directory unix_notes, so you continue typing : u+(TAB) will print out vi unix_notes/ if there are no files or directories starting with u in your working directory. If there are other files which match (TAB)(TAB) will print the list of possible completions. So if I have the directories unix and unicode I just have to type unix_(tab) to get unix_notes/ and then k(TAB) to get key_combos.html as that is the only file I have starting with a k in my unix_notes directory. You just have to type out as many characters as will uniquely identify the file. (I've been known to rename files and directories so that I can get to them using only one character + (TAB) combos. For example: vi /v(TAB)w(TAB)h(TAB)i(TAB)p(TAB)s(TAB)b-(TAB) Will bring open the file /var/www/html/info/programming/shel-l/bash.html for editing in vi. Trust me the tab-key makes this much faster. You'll be (TAB)(TAB)(TAB)ing in no timeNow

Expert tip:If you have started tab-completing everything you type you might be happy to know that there is a mode for tab-completion in emacs, but even easier in that while writing code and in vim you just type CTRL-P! this will compete with any word already in the file you are typing and even search through include files if you are writing code!

Laziness No.2: I just typed that, or I hate typing the same stuff over and over

Solution No.2 : (ESC)^. (escape+period) or (ALT)^. (alt+period)

Depending on your setup the (ESC) key could either be one marked esc in the upper-left most corner or mapped to the alt key. (On many modern setups both will work). (ESC)+. reprints the last argument of the last command you typed. This is really handy.

Example: how does it work? You are moving many files from one directory to your web server directory. mv *.html /var/www/html/programming/shell/ mv

*.gif /var/www/html/programming/shell/ mv *.jpg /var/www/html/programming/shell/

NOTE: if you don't know what the star does you really should learn about globbing. Globbing is a huge accelerator, and is also quite well documented (one good site is here. I chose in this article to focus on some handy lesser know key combos.

Using the one character + (TAB) we just talked about this would be quite fast, but you still have to type /v(TAB)/w(TAB)/h(TAB)/p(TAB)/s(TAB)/ over and over to get /var/www/html/programming/shell/. When human beings do things over and over, they make mistakes. Machines however are wonderfull at doing things over and over so get that machine to work for you. Using the same example as above you type the first line: mv *.html /var/www/html/programming/shell/Using (TAB)'s of course. Then for the next line you type: mv *.gif (ESC) .That is either the escape-key (upper right) or the alt key (next to the space bar on your key board and a period. This will get the last argument from the line you just typed hence in this case /var/www/html/programming/shell/. Fast hunh...

Laziness No.3: I just typed that, or I hate typing the same stuff over and over (recurring laziness)

Solution No.3: History the (up-arrow) and (CTRL)^R

I'm going to go over this first part quickly, because most of you may know about the (up-arrow) already. To get the last command you typed you hit the (up-arrow), and to go back to previous commands you hit the up arrow again and again. Sometimes if you are trying to get back that complicated command you typed yesterday or last week, this could mean a lot of (up-arrow)'s. Wouldn't it be great if you could search your history? Enter (CTRL)^r. The little hat means hold down the (CTRL) key and hit "r" otherwise you will just type an r. Now when you have done this your prompt (the part of the command line before what you type) should change to this: (reverse-i-search)`': Now type the first couple of letters of the text you want to match. For example the other day I was trying to use a command line program called transcode which takes many arguments and I don't remember what exact list of arguments I used to get it to work. So on a blank command line I type :

(CTRL)^r I get the new prompt : (reverse-i-search)`': I type t : and get the line (reverse-i-search)`t': startx because startx is the most recent command I used which has a "t" in it. I then type "r" and in my case I get :

(reverse-i-search)`tr': transcode -i capture001.avi -x dv,avi -I 1 -C 1 -z -k -o ak-divxmp3.avi -y divx4 because this is the last command in my history which has a "tr" in it. An (Enter) executes this line as is. A (right-arrow) snaps me out of reverse-search mode and allows me to edit the line (say I need to touch up the command somehow). An (esc) quits reverse search mode and gets me back to my blank command line. If this was not exactly the version of the transcode command I wanted I could just continue hitting (CTRL)^r and this would keep searching back through my history for commands matching "tr". Try it out you'll get the hang of it.

NOTE: The unix jockeys would probably tell you that you could also type history | less. Search the listing using "/(search term)" get the number of the command in your history, quit less using "q", and type !(the number you have just found) to execute the command you have just found. But this seems to me more confusing, and more steps-- hence less laziness and that is a bad thing TM

Laziness No.4: this command line is sooooo long, and I have to change one letter, or one word way at the begining

Solution No.4: (CTRL)^a, (CTRL)^e, (CTRL)^b, (CTRL)^f, (CTRL)^h, (CTRL)^d, (ESC)^d, (CTRL)^w

Now in the last example we looked at some really long command lines. This can happen, and editing lots of long lines will get you tired and prone to error, unless you know how to get around them. Now if your first impulse was to use the mouse, you know what a mess that can make, even using the nifty cut and paste stuff with the middle mouse button under X. Being forced to (right-) and (left-arrow) all the time is surely one of the biggest time killers on the command line. When you finally learn to use one of the Unix text editors, either vi or emacs, you quickly realize how fast editing becomes, because you can zip around your text going to the ends of lines, to particular words without lifting your hands from the keyboard. Very quickly mousing loses its appeal. The bash shell gives you a lot of emacs combinations built right in. All you vi guys and gals didn't even notice that the (CTRL)^r from the last section is the standard reverse search in emacs. And you newer users who have been afraid of touching one of these editors have already learned an important emacs command. Now on to some more:

Command

Moves cursor to...

mnemonic

(CTRL)^e

the end of the line

"e" is for end

(CTRL)^a

the beginning of the line "a" is the furthest left key on it's row

(CTRL)^b, (CTRL)^f

back a character, forward a character (but wait isn't that just what the arrow keys do?Yes, but if one day you don't have arrow keys or one is broken you'll be happy to remember this, and see the next entry...) "b" is back, "f" is forward

ALT)^b, (ALT)^f

back a word, forward a word

look ma same mnemonic!

(DEL), (CTRL)^h

delete last character

(Most people probably know where the (DEL) key is. It's the most used key on most key boards. What you may not know is that there were once keyboards without delete keys, and that some unices would map it to something else. Now you can understand techie-talk (CTRL)^h(CTRL)^h(CTRL)^h) none I can think of...

(CTRL)^d

delete forward

This is probably the combination you would expect to be mapped to the same as (DEL) but it's not. It actually makes it easier to remember "d" as in delete (forward)

(ALT)^d

delete forward word

"d" as in delete (forward)

remember how above (CTRL)^b mapped to a letter, whereas (ALT)^b mapped to a word. You can use the same memory space to remember this one)

(CTRL)^w

delete word

"w" as in word

for the rest you are on your own. This one doesn't make a nice mnemonic like the others, but it works and is very handy

(CTRL)^u clear from cursor to the beginning of the line

? I'm not going to go through examples using these. Somehow you have to internalize them, and you can get really fast on the command line. I would suggest to pick a few and use them all the time until they are second nature. It's really not that hard. I mean memorized (CTRL)^x, (CTRL)^c, and (CTRL)^v to copy and paste on many windowing systems right? This is the same thing. Good luck. let me know if I have been confusing.

Editors note: Macro will be teaching the upcoming Unix 1 class starting March 18, 2002 .Which will cover "commands".

Point of View

Mail Server Anti-Virus Protection for Linux?

by Dennis Bella

And so began the breakout of Linux The impact of Linux reflect the stern emotions of the IT world: Enough of the one-OS-show! Break the Microsoft mold!

Linux is said to be invulnerable to virus attacks - so far. Alternatively, a growing number of fellow penguin followers agree: Linux viruses will soon haunt the Internet alongside their Windows brethren . Linux began to be used at an institutionalized level worldwide. Enterprises and organizations at every level show increased interest in Linux on their servers and desktops.

The pace has grown worldwide. Recently, the top three IT market leaders in China agreed to Linux. South Korea has ordered 120,000 open source office solutions and projects a savings of up to 80% from previous costs. Great Britain plans the mandatory introduction of open source software in the public sector. Even Big Blue itself, IBM, started to provide Linux technical support. The introduction of open-source software in the German Parliament (Bundestag) has generated a heated debate: Linux means economic competition in the open market, democracy principles, and the local advantages of open source such as cost efficiency and flexibility.

The development of secure software with openly accessible source codes plays a central role within the EU commissions IT initiative "e-europe". (www.bundestux.de ; www.ibm.com ; www.turbolinux.com) The Linux marketing advantage resides on two fronts: while being a very stable platform, it is practically free. Linux is about to surpass the open-source personality as it leaves the limited environment of individual/informal community use. This new century will see Linux as the OS of enterprise organizations - running on high performance servers and getting the broad attention it deserves.

Protect the Penguin? While some argue that there are only a handful of Linux viruses, only several of them were actually seen in-the-wild. Others say any system can become infected with a virus if the owner is careless enough to install unchecked binaries as root. The truth: Linux can also act as a "carrier" despite the fact that it can’t become infected with Windows viruses. Linux can store viruses and infect a Windows partition or a Windows platform. In this case, you need an anti-virus with a unique engine, such as RAV Anti-Virus, which uses a multi-platform technology, thus detecting all viruses, irrespective of the OS they’re made for. With such an anti-virus program, you can manage the security for both systems: scan a Windows partition and share from a Linux platform, or conversely.

Things become even more complicated when it comes to mail servers managing a mail flow that point to different OS's. In the case of unprotected mail servers - regardless of OS, a clients' vulnerability is a fact. The administrator does not know which types of files are going through the server.

An unprotected Linux server may be safe for Linux users, at least for the short term, but may also store and transmit infected files to Windows users. A mass-mailed worm is a threat for all servers, since it doesn't damage a particular system, but fills the server's bandwidth, causing serious system damage. Hence, you have anti-virus protection for the Linux OS. There are only a few solutions that address this, and they're not at all cheap.The optimal solution would reside in combining the following features in a single anti-virus program:

Optimal pricing policy, perhaps a "per domain" or "per machine" and not "per mail box" licensing system would be best

Multi-platform engine

Reliable, certified product

Especially designed for protecting mail servers

Group configuration is a plus

Content filtering and spam filtering as a value-add

The program we took for example above provides per domain licensing, a powerful multi-platform engine, content filtering, and group configuration, which is a very important feature for ISPs who want to create specific client groups and perceive a fee from them, depending on clients' security needs. You should visit www.ravantivirus.com.

With the growing need for anti-virus on Linux, (even for all open source environments) the enormous exchange of information on mail servers gives rise to the need for such protection. Protection still comes down to the operator. The OS may be great, even perfect, but mistakes are part of human nature. The more people who use Linux, the higher the probability for increased virus attacks..

While there remains little threat for Linux itself on an "autistic" desktop, viruses are a great threat for Linux based mail servers. As the future of network communication grows and requires intricate cross OS support, an anti-virus program is needed to protect the rising star of the open source.

Free Software around New York City

by Jon Babor

The New York Linux Scene is a new organization in New York and it comes into a crowded city. NYLXS is tied to no one Free Software group in New York City, but it has ties to all of them. At many Free Software events members of many different groups appear, and many member of NYLXS are members of other Free Software groups as well.

Thus, it does not matter for example, whether a NYLUG or DeLUGE is sponsoring an installfest, or NYLXS is throwing an instalfeast. Members of any organization may be found at any event.

More importantly, there are oftentimes when no one group is big enough to handle an issue, and more are always better. In August of 2001, when Dmitry Sklyarov was imprisoned, members of NYLUG, LXNY, and NYFairuse were quickly mobilized. Personally, I became involved through subscribing to the Free Dmitry mailing list, and soon moved over to NYLUG and NYFairuse for New York specific information. The size of NYLUG made it easy to find and an obvious starting point for many who did not know where to turn, and it allowed a smaller, more local organization such as NYFairuse to mobilize better than it otherwise would have. Now, much DMCA-specific (along with other political issues, of course) discussion and action takes place as a result of collaborative efforts among LXNY, NYLXS, NYFairuse, and others.

Some of the Free Software groups in New York City include:

NYLUG is New York's traditional LUG. It holds monthly meetings, always followed by Stammtisch. From www.nylug.org:

?Linuxstammtische - are social gatherings of local Linux, FREEBSD, and other open-source software users at a local brewpub, founded in connection with the first annual Linuxbierwanderung (Linux Beer Hike, August 1999 in Bavaria, August 2000 in England's Lake District). What Is It: Beer (naturlich), good fellowship, and an occasional lesson in the German language. Translated to English, Stammtisch means 'table for regulars.' Consequently, you can generally find nyluggers near the back of The Old Stand (914 3rd Ave. on the corner of 55th Street) after every monthly meeting.

NYLUG meetings generally place at 6:30 PM on the 3rd Wednesday of every month at the IBM headquarters in

Midtown and feature guest speakers focusing on Linux. Recent topics/speakers have included Terry Schmit on Wireless Public Networking, Simon Lok on email privacy and key signing (NYLUG also holds GPG/PGP keysignings quite often,) and, in the wake of last year's terrorist attacks, a talk on ?Decentralism Against Terrorism?, by Eric S. Raymond. Also, NYLUG recently held their annual holiday party sponsored by and with a special presentation by AMD.

LXNY is ?New York's Free Software Organization.? Less traditional than NYLUG, and less Linux-centric, LXNY is for all who have an interest in Free Software, regardless of interest in Linux. Formal meetings often feature speakers, but not necessarily. Recent important topics of interest at meetings have been Dmitry Sklyarov and the DMCA and LXNY's planned upcoming Microsoft Refund Day. Also, LXNY supports GNUbies: The LXNY GNU/Linux/Free OS Beginners Group.

GNUbies is an group for people who are new to Free Software. Mostly Linux based, monthly meetings are address topics in Free Software. Unlike NYLUG or some other groups, which often assume a certain amount of technical knowledge, GNUbies assumes none. Topics are generally geared towards the use and installation of Free Software, and are presented for the newbie who knows nothing. Additionally, meetings can be of interest to more experienced users as well. Meetings are generally on the second Wednesday of every month at the IBM building in midtown.

DeLUGE is the Desktop Linux Users Group for Endusers. The group is meant to focus on the use of desktop Linux and to help people to use Linux for all of the same desktop uses of other Operating Systems. Focus is on applications and enduser software ? from office suites to games. The last meeting was on March 6th and featured a presentation on various email clients running under Linux, including Kmail, Pine, Mutt, and Evolution. DeLUGE works for Linux not for the technical guru but for Everyman.

There are many other groups in New York and a good description of them all cannot be contained here. Future journal articles will feature more descriptions of other Free Software groups and will let the reader know what is going on in the Free Software world in New York City. f you wish to have your group listed here send a outline of your group to miker@nylxs.com

NYLXS Committee Heads

Free Software for Schools: (Marcus Conti) http://www2.mrbrklyn.com/appl/nylinux/mail/bysubject.html?idno=1534

Journal Committee : (Michael Richardson) http://www2.mrbrklyn.com/appl/nylinux/mail/bysubject.html?idno=1381

Public Relations : (Joe Grastrata) http://www2.mrbrklyn.com/appl/nylinux/mail/bysubject.html?idno=1416

FreeSoftware Institute: (Paul Rodriguez) http://www2.mrbrklyn.com/appl/nylinux/mail/bysubject.html?idno=1429

Finance Committee: (Joe Maffia) http://www2.mrbrklyn.com/appl/nylinux/mail/bysubject.html?idno=1429

And our fearless leader Ruben Safir who needs no introduction as you can just:
http://www.google.com/search?hl=en&q=Ruben+I+Safir&btnG=Google+Search