AUUG Membership and General Correspondence
The AUUG Secretary
PO Box 366
Kensington NSW 2033
Telephone: 02 8824 9511
or 1800 625 655 (Toll-Free)
Facsimile: 02 8824 9522
Email: auug@auug.org.au

AUUG Management Committee
Email: auugctee@auug.org.au

President:
David Purdue
David.Purdue@auug.org.au
iPlanet e-commerce solutions
The Tea House
Level 1, 28 Clarendon Street
South Melbourne VIC 3205

Vice-President:
Malcolm Caldwell
Malcolm.Caldwell@auug.org.au
Northern Territory University
Casuarina Campus
Darwin NT 0909

Secretary:
Michael Paddon
Michael.Paddon@auug.org.au
eSec Ltd.
245 Racecourse Rd.
Flemington VIC 3031

Treasurer:
Luigi Cantoni
Luigi.Cantoni@auug.org.au
STM
PO Box 51
North Perth WA 6906

Committee Members:
Sarah Bolderoff
Sarah.Bolderoff@auug.org.au
University of South Australia
School of Computer and Information Science
Room F2-65
Mawson Lakes Campus SA 5095

Alan Cowie
Alan.Cowie@auug.org.au

Greg Lehey
Greg.Lehey@auug.org.au
Linuxcare Inc.
PO Box 460
Echunga SA 5153

Peter Gray
Peter.Gray@auug.org.au
Information Technology Services
University of Wollongong
Wollongong NSW 2522

David Newall
David.Newall@auug.org.au
Tellurian Pty Ltd.
272 Prospect Road
Prospect SA 5082

AUUG Business Manager
Elizabeth Carroll
busmgr@auug.org.au
PO Box 366
Kensington NSW 2033

Editorial

Con Zymaris
auugn@auug.org.au

Welcome to the last AUUGN for the Year 2000, and the first one with me (Con Zymaris) as editor.

I've accepted the stewardship of this long-running Journal from Günther Feuereisen, who has been editor of AUUGN for over four years. Please join me in thanking Günther for all his efforts. My putting together even one of these issues has definitely enlightened me as to the work-level required, highlighting Günther's dedication even more.

Now, onto what we have planned for you in this, and in future issues.

We've drawn upon various technical information sources for content that we think will be of benefit to Unix and open systems developers, systems administrators, managers and related professionals.

We mix big-vendor Unix platforms, as well as Open Source OSes and tools.

And finally, we hope we do what all good user-group publications are meant to do, let you know what is happening with AUUG, your user group.

If we've been edifying, entertaining and enlightening in any way, tough, that was intentional :-) If we've missed the mark by way of content or style, then it's in your hands to help rectify this; Let us know.

Cheers,

Con

Contribution Deadlines for AUUGN in 2001

Volume 22 • Number 1 – March 2001: **February 17th, 2001**

Volume 22 • Number 2 – June 2001: **May 17th, 2001**

Volume 22 • Number 3 – September 2001: **August 17th, 2001**

Volume 22 • Number 4 – December 2001: **November 17th, 2001**

AUUGN Editorial Committee

The AUUGN Editorial Committee can be reached by sending email to:
auugn@auug.org.au

Or to the following address:
AUUGN Editor
PO Box 366
Kensington NSW 2033

Editor:
Con Zymaris

Sub-Editors:
Mark Neely
Jerry Vochteloo

Public Relations and Marketing:
Elizabeth Carroll

AUUGN Submission Guidelines

Submission guidelines for AUUGN contributions can be obtained from the AUUG World Wide Web site at:

www.auug.org.au

Alternately, send email to the above correspondence address, requesting a copy.

AUUGN Back Issues

A variety of back issues of AUUGN are still available. For price and availability please contact the AUUG Secretariat, or write to:
AUUG Inc.
Back Issues Department
PO Box 366
Kensington NSW 2033

Conference Proceedings

A limited number of copies of the Conference Proceedings from previous AUUG Conferences are still available. Contact the AUUG Secretariat for details.

Mailing Lists

Enquiries regarding the purchase of the AUUGN mailing list should be directed to the AUUG Secretariat.

Disclaimer

Opinions expressed by the authors and reviewers are not necessarily those of AUUG Inc., its Journal, or its editorial committee.

Copyright Information

AUUGN is the journal of AUUG Inc., an organisation with the aim of promoting knowledge and understanding of Open Systems, including, but not restricted to, the UNIX® operating system, user interfaces, graphics, networking, programming and development environments and related standards.

President's Column

David Purdue
David.Purdue@auug.org.au

> **connect** *v.* (often **+ to**, **with**) join; be joined; associate mentally or practically; (**+ with**) (of train etc.) arrive in time for passengers to transfer to another; put into communication with by telephone; (usu. in *passive*; **+with**) associate with (others) in relationships etc.; *colloq.* hit or strike effectively.
> - *The Australian Little Oxford Dictionary*

I am a very well connected person. You can reach me at almost any time, day or night, from pretty much anywhere in the world. I have a mobile phone, but I am also mostly near one land line or another. I don't have a pager, but my phone has a memo service and SMS. I have about a dozen email addresses. I even have a camera set up on my desk for rudimentary video conferencing.

Which begs the question – why am I spending so much time on aeroplanes?

It seems that we as humans have a need to connect physically with other people. There are so many forms of communication over a distance these days, but there seems to be many elements of human communication that are not captured by these communications. Serious negotiations are still carried out face-to-face.

A similar observation applies to gathering new knowledge. We have at our disposal books, web sites, videos, computer aided instruction, but still the most effective way to learn something is to have someone explain it to you. This is reflected by the decision of some academics at a major Australian university not to place learning materials or recordings of the lectures for their courses on the web. They found that students were using this as an excuse not to attend lectures, and thus were not learning as effectively.

This is partly what is driving the AUUG Management Committee to organise more events where people will come and explain things to you.

During November AUUG organised the Security Symposium and the Australian Open Source Symposium. Both these events are in their second year, and were both very successful, if not as well attended as we would like.

We think that the one day symposium is a good way for AUUG to deliver value back to our members, and we will be increasing the number of symposia we run so that we can cover more topics and present events in more locations. We would be keen to hear from you if you have an idea for a symposium topic, and especially if you would like to speak at or help organise one.

To ensure you hear about AUUG organised events, subscribe to auug-announce. To ensure that AUUG has your correct email address and that you are on the list, please send an email to Liz Carroll at <busmgr@auug.org.au>.

Public Notices

Clunies Ross National Science & Technology Award 2001

Date: 28 March 2001

Event: Clunies Ross National Science & Technology Award 2001

Award recipients will be publicly honoured at a formal ceremony and dinner to be held at Hotel Sofitel, Melbourne on Wednesday 28 March 2001.

This annual Award has now honoured 52 special Australians who have made an outstanding contribution to the application of science and technology for the economic, social or environmental benefit of Australia.

Contact Details: Mary Bolger on (03) 9854 6266 (Email: icr@crnet.com.au) or visit our web site at http://www.cluniesross.org.au

Upcoming Conferences

December 3 - December 5, 2000.
Wireless DevCon 2000
San Jose Doubletree Hotel, San Jose, CA.

December 3 - December 8, 2000.
LISA 2000
New Orleans, LA.

January 30 - February 2, 2001.
LinuxWorld Conference & Expo
New York, NY

March 5 - March 8, 2001
The Ninth International Python Conference
Long Beach, California

My Home Network (November 2000)

Author: Frank Crawford
<frank@crawford.emu.id.au>

Its time for another AUUGN, but this time there are a number of differences. Firstly, we have a new editor for AUUGN, Con Zymaris, and I'd like to welcome him to this demanding job. Con, have fun with it. I'd also like to thank our previous editor, Gunther Feuereisen, for his efforts over the years, well done. However, changes in AUUGN are not the only thing I want to mention, I have changed jobs, moving from ANSTO, to a new company, ac3. Still doing the same type of work, managing systems, but at an organisation dedicated to high performance computing.

One reason I mention the change in jobs is that many of the items within my home network have been reimplemented at ac3 as I've helped to build up an entire computer infrastructure from scratch. Again, this shows that the requirements for an organisations network and a home network are not really that different.

One of the other similarities between such systems is how it is managed, in particular you need to follow the three rules of system administration: "automate, automate, automate". :-) A big area for this is keeping software up-to-date, in particular downloading updates for later (or immediate) installation.

Despite the prevalence of the web and the assumption that it can be used for everything, most installation programs, particularly for GUI based systems, assume that a user is available to launch the process. As well, updates are getting bigger and bigger, and more and more frequent. All of which means that you need either a lot of time to sit at your computer or need a way to make the whole process non-interactive, i.e. automate it.

The first step in this is to realise that while downloads are getting bigger, the protocol used for it is amongst the oldest available on the Internet, FTP. This has been supplemented by HTTP download capabilities, but in most cases you can access the files via either method.

Given a server running continuously, with sufficient disk space, it is simple to set up an FTP mirror. While there are a few programs around to do it, the one I use is called 'mirror'. It is written entirely in Perl and is simple to install and use. By itself it downloads files recursively from a source directory, but include a script to run multiple downloads from different sites at once.

Aside from just downloading files, it deletes files that have been removed, creates new directories, excludes files matching certain patterns and ultimately sends out mail when updates occur. In fact the comments in the documentation says:

Mirror is a package written in Perl that uses the FTP protocol to duplicate a directory hierarchy between the machine it is run on and a remote host. It avoids copying files unnecessarily by comparing the file time-stamps and file sizes before transferring. Amongst other things, it can optionally rename, compress, gzip, and split files.

Mirror was written by Lee McLoughlin <lmjm@icparc.ic.ac.uk> for use by archive maintainers but can be used by anyone wanting to transfer a lot of files via FTP. Although originally only available on Un*x with version 2.9 mirror will also run on Wind*ws 95 and Wind*ws NT.

The latest version of mirror can always be found at either:
ftp://sunsite.org.uk/packages/mirror/mirror.tar.gz
ftp://sunsite.org.uk/packages/mirror/mirror.zip

It is also distributed as part of the Red Hat Powertools, with version 2.9 on the RedHat 6.2 Powertools.

'Mirror' is basically controlled by two files, a global configuration file, and then individual ones for each package. The global file is usually '/etc/mirror.defaults', and generally contains options that relate to all downloads. What I have in mine is:

```
----
# This is the default mirror settings used by
my site:
# crawford.emu.id.au
#
# Frank Crawford <frank@crawford.emu.id.au>

# You should be able to use this at other
sites.  You should only have
# to change bits that reference my site.

package=defaults
        # The LOCAL hostname - if not the same
as 'hostname'
        # (I advertise the name
crawford.emu.id.au but the machine is
        #  really bits.crawford.emu.id.au.)
        hostname=crawford.emu.id.au
        # Keep all local_dirs relative to here
        local_dir=/home/mirror/pub/
        # The local_dir must exist FIRST
        #local_dir_check=true
        remote_password=root@crawford.emu.id.au
        mail_to=root@crawford.emu.id.au
        # Don't mirror file modes.  Set all
dirs/files to these
        dir_mode=0755
        file_mode=0444
        # By defaults files are owned by
root.zero
        user=0
        group=0
#       # Keep a log file in each updated di-
rectory
#       update_log=.mirror
        update_log=
        # Don't overwrite my mirror log with the
remote one.
        # Don't pull back any of their mirror
temporary files.
        # nor any FSP or gopher files...

exclude_patt=(^|/)(\.mirror$|\.mirror\.log|cor-
e$|\.cap|\.in\..*\.$|MIRROR\.LOG|#.*#|\.FSP|\.-
cache|\.zipped|\.notar|\.message|lost\+found/|-
Network Trash Folder)|suky.mpe?g
        # Do not to compress anything
        compress_patt=
        compress_prog=gzip
        # Don't compress information files,
files that don't benifit from
```

```
        # being compressed, files that tell
ftpd, gopher, wais... to do things,
        # the sources for compression pro-
grams...
        # (Note this is the only regexp that is
case insensitive.)
        # z matches compress/pack/gzip, gz for
gzip. (built into perl)
        # taz/tgz is compressed or gzipped tar
files
        # arc, arj, lzh, zip and zoo are pc
and/or amiga archives.
        # sea are mac archives.
        # vms used -z instead of .z.  stupid
vms.
        # shk is multimedia? used on apple2s.
        # rpm and deb are package formats used
on RedHat and Debian Linux
        compress_excl+|-
z(\d+)?$|\.tgz|_tgz|\.tar\.Z|\.tar\.gz|\.taz$|-
\.arc$|\.zip$|\.lzh$|\.zoo$|\.exe$|\.lha$|\.zo-
m$|\.gif$|\.jpeg$|\.jpg$|\.mpeg$|\.au$|\.shk$|-
rpm$|deb$|read.*me|index|info|faq|gzip|compres-
s|(^|/)\.\.?$
        # Don't delete own mirror log, .notar or
.cache files (incl in subdirs)
#
delete_excl=(^|/)\.(mirror|notar|cache)$
        # Ignore any local readme and .mirror
files

local_ignore=README.doc.ic|(^|/)\.(mirror|nota-
r)$
        # Automatically delete local copies of
files that the
        # remote site has zapped
        do_deletes=true
        max_delete_files=50%
        max_delete_dirs=50%
        timeout=300
        #failed_gets_excl=\:\ Permission de-
nied\.$
----
```

A couple of important bits, most lines are of the form "<keyword>=<value>" to set a value or else "<keyword>+<value>" to append additional items to an existing item. Values either consist of a simple entry or a regular expression used for various pattern matching.

To mirror a site you will need to set up a configuration for the download. The format of this file is similar to the default configuration file, but with item specific to that host. For example, my file to download Red Hat and Kernel updates is called "ftp.aarnet.edu.au" and contains:

```
----
package=redhat-updates-6.2
        comment=RedHat 6.2 Updates
        site=ftp.aarnet.edu.au
        # where to start pulling files back from
        remote_dir=/pub/redhat/updates/6.2
        # where to put the files on your machine
        local_dir+RedHat/updates/6.2
        # exclude some items
        exclude_patt+|sparc|mips|alpha|SRPMS

package=linux-kernel-2.2
        comment=Linux 2.2 Kernel
        site=ftp.aarnet.edu.au
        # where to start pulling files back from
        remote_dir=/pub/linux/kernel/v2.2
        # where to put the files on your machine
        local_dir+Kernel-2.2
        # exclude some items
        exclude_patt+|2\.2\.[0-9]\.|2\.2\.1[0-
3]\.
----
```

This will mirror all the RedHat 6.2 update directory, and all kernel updates after 2.2.14.

Once these files are setup, you can mirror the system by running 'mirror.pl ftp.aarnet.edu.au', or just to test what it will do, run 'mirror.pl -n ftp.aarnet.edu.au'. This will perform two separate downloads, one after the other, mailing output to the address given by "mirror_to" keyword.

To run just this mirror regularly, all you need to do is add a line to cron of the form:

```
05 23 * * * root /usr/bin/mirror.pl
/home/mirror/packages/ftp.aarnet.edu.au
```

and then pick up the updated files in the directory specified by "local_dir", which is constructed from "/home/mirror/pub" in '/etc/mirror.defaults' and either "RedHat/updates/6.2" or "Kernel-2.2" depending on which package is running.

One point is that it is better to run any mirror as a separate cron job, as it may often take a long time (which is why you are running it this way after all). Of course, you may want to run a number in parallel, but rather than running it multiple times, the package also supplies another Perl script which will attempt to do some simple load balancing.

This script is called 'mm.pl' (for "mirror-master") and takes a configuration file (called 'mmin') of the form:

```
----
# How many mirrors to run at the same time.
# Over a 56K modem - what do you expect?
max=4

home=/home/mirror

cmdin=/usr/bin/pkgs_to_mmin.pl packages/*
----
```

which runs up to four jobs at once, one for each of the separate files listed in the 'packages' directory. This is then controlled by a line in your crontab of the form:

```
05 23 * * * root /usr/bin/mm.pl
/home/mirror/mmin
```

There are a couple of little gotcha's I've found here. The biggest is that the default mirror command invoked by 'mm.pl' is './mirror', i.e. it is expected to be in "home" directory. This can be changed by a command in the 'mmin' file, but I found it simpler to put in a symbolic link to the real mirror program (due to all the extra arguments I had to add to the 'mirror' command).

A second point to note is that 'mm.pl' refuses to try a package again, if it has been run recently, usually within the last 12 hours. This is controlled by the file 'mm.status' generated by 'mm.pl'.

Finally, while 'mirror.pl' deletes files, it refuses to delete them if it is more than 50% of the files in a directory. This is controlled by the "max_delete_files" and "max_delete_dirs" variables in '/etc/mirror.defaults' file.

Of course, these days you also need to handle web sites that don't know anything about FTP. There are two different packages that can help you here, 'wget' or 'curl'. I'm not going to go into them here, rather I'll leave it to you, the reader as an exercise.

Please let me know what you discover, and I'll include it in my next column.

On that note I'll leave off, but please feel free to drop me a note about what you are doing. I am wondering if anyone finds any use in this column, or even has anything of their own to offer. Let me know.

AUUG Corporate Members

as at 1 December 2000

- Andersen Consulting
- ANSTO
- Aurema Pty Ltd
- Australian Bureau of Statistics
- Australian Industry Group
- Australian National University
- Australian Taxation Office
- Australian Water Technologies P/L
- BHP Information Technology
- British Aerospace Australia
- Bunnings Building Supplies
- Bureau of Meteorology
- C.I.S.R.A.
- Camtech
- Cape Grim B.A.P.S
- Central Queensland University
- Centrelink
- CITEC
- Commercial Dynamics
- Commonwealth Steel Company
- Computer Science, Australian Defence Force Academy
- Computing Services, Dept Premier & Cabinet
- Corinthian Industries (Holdings) Pty Ltd
- Corporate Express Australia Limited
- Crane Distribution Limited
- CSC Australia Pty. Ltd.
- CSC Financial Services Group
- CSIRO Manufacturing Science and Technology
- Curtin University of Technology
- Cyberscience Corporation Pty. Ltd.
- Cybersource Pty. Ltd.
- Daimler Chrysler Australia - Pacific
- Dawn Technologies
- Deakin University
- Department of Defence
- Department of Land & Water Conservation
- Education QLD
- Energex
- eSec Limited
- Everything Linux
- G.James Australia Pty. Ltd.
- Great Barrier Reef Marine Park Authority
- HIH Insurance
- HIH Winterthur
- IP Australia
- IT Services Centre, ADFA
- Land Information Centre
- Land Titles Office
- Macquarie University
- Mercantile Mutual Holdings
- Motorola Australia Software Centre
- Multibase WebAustralis Pty Limited
- Namadgi Systems Pty Ltd
- Nokia Australia
- NSW Agriculture
- NSW Public Works & Services, Information Services
- Peter Harding & Associates Pty. Ltd.
- Qantas Information Technology
- Rinbina Pty. Ltd.
- SCO
- Security Mailing Services Pty Ltd
- Snowy Mountains Authority
- St. John of God Health Care Inc.
- St. Vincent's Private Hospital
- Stallion Technologies Pty. Ltd.
- Standards Australia
- State Library of Victoria
- TAB Queensland Limited
- Tellurian Pty. Ltd.
- The Fulcrum Consulting Group
- The University of Western Australia
- Thiess Contractors Pty Ltd
- Tower Technology Pty. Ltd.
- University of New South Wales
- University of Sydney
- University of Technology, Sydney
- Victoria University of Technology
- Westrail
- Workcover Queensland

Systems Engineer

Biotechnology Company - Salary Neg.

Expanding Biotechnology company based at Sydney suburb of Nth Ryde requires an experienced administrator to lead their small systems administration and support team. Both hands on and Supervisory experience necessary as you will ensure that c50 user network is adequately supported. You will also design and implement solutions for different user groups. Consequently good verbal and written communication skills are necessary. Your technical experience should comprise the following:

**systems admin. experience with NT, NT servers, Unix and Linux.*

**knowledgable re. high-end / enterprise hardware systems*

**managing & building TCP /IP networks, Firewalls, DMZ, Apache web servers, network topology, sub-networking and virtual hosting.*

**networking protocols such as LDAP, NIS+, etc.*

**Samba & NT networking*

**NFS, NIS and communications between various forms of Unices.*

**DNS servers & internet domains*

**mail routing under Linux*

**setup & management of dialin servers & communications via Frame-relay, ISDN & ADSL.*

If you are interested in establishing a Managerial career with a company committed to maintaining it's cutting edge IT environment , please E-mail your CV to Angela Kerley or Mike Steele at:
topjobs@carterandstone.com.au or Ph (02) 9955 5477

CARTER AND STONE

For more information please contact Angela Kerley or Mike Steele on
Ph: (02) 9955 5477 • Fax: (02) 9955 5898
Email: Please click the "Apply Now" button below

Installing FreeBSD for i386

Author: Jordan K. Hubbard

Quick Start

This manual documents the process of making a new installation of FreeBSD on your machine. If you are upgrading from a previous release of FreeBSD, please see the file UPGRADE.TXT for important information on upgrading. If you are not familiar with configuring PC hardware for FreeBSD, you should also read the HARDWARE.TXT file - it contains important information which may save you a lot of grief.

If you're new to FreeBSD then you should also read EVERYTHING listed in the Documentation menu of the installer. It may seem like a lot to read, but the time you spend now reading the documents will be made up many times over because you were adequately prepared. Also, you will know the types of information available should you get stuck later. Once the system is installed, you can also revisit this menu and use a WEB browser to read the installed FAQ (Frequently Asked Questions) and Handbook HTML documentation sets for FreeBSD. You can also use the browser to visit other WEB sites on the net (like http://www.freebsd.org) if you have an Internet connection. See ABOUT.TXT for more information on the resources available to you.

The best laid plans sometimes go awry, so if you run into trouble take a look at TROUBLE.TXT which contains valuable troubleshooting information. You should also read ERRATA.TXT before installing and follow the pointers there carefully since this will stop you from falling over any problems which have reported in the interim for your particular release.

DISCLAIMER: While FreeBSD does its best to safeguard against accidental loss of data, it's still more than possible to WIPE OUT YOUR ENTIRE DISK with this installation if you make a mistake! Please do not proceed to the final FreeBSD installation menu unless you've adequately backed up any important data first! We really mean it!

FreeBSD requires a 386 or better processor to run (sorry, there is no support for '286 processors) and at least 5 megs of RAM to install and 4 megs of RAM to run. You will need at least 100MB of free hard drive space for the most minimal installation. See below for ways of shrinking existing DOS partitions in order to install FreeBSD.

Installing FreeBSD from CDROM or the Internet

The easiest type of installation is from CD. If you have a supported CDROM drive and a FreeBSD installation CD from Walnut Creek CDROM, there are 2 ways of starting the installation from it:

1. If your system supports bootable CDROM media (usually an option which can be selectively enabled in the controller's setup menu or in the PC BIOS for some systems) and you have it enabled, FreeBSD supports the "El Torrito" bootable CD standard. Simply put the installation CD in your CDROM drive and boot the system to begin installation.

2. Build a set of FreeBSD boot floppies from the floppies/directory in every FreeBSD distribution. Either simply use the "makeflp.bat" script from DOS or read floppies/README.TXT for more information on creating the bootable floppies under different operating systems. Then you simply boot from the first floppy and you should soon be in the FreeBSD installation.

If you don't have a CDROM and would like to simply install over the net using PPP, slip or a dedicated connection, simply fetch the
<FreeBSD-release>/floppies/boot.flp file from:

ftp://ftp.freebsd.org/pub/FreeBSD

or one of its many mirrors
(http://www.freebsd.org/handbook/mirrors.html)
and follow step 3 above. You should also read the floppies/README.TXT file as it contains important information for downloaders.

Once you have your boot floppies made, please go to section 1.5 of this document for additional tips on installing via FTP or NFS.

Detail on various installation types

Once you've gotten yourself to the initial installation screen somehow, you should be able to follow the various menu prompts and go from there. If you've never used the FreeBSD installation before, you are also encouraged to read some of the documentation in the the Documentation submenu as well as the general "Usage" instructions on the first menu.

NOTE: If you get stuck at a screen, hit the F1 key for online documentation relevant to that specific section.

If you've never installed FreeBSD before, or even if you have, the "Standard" installation mode is the most recommended since it makes sure
that you'll visit all the various important checklist items along the way. If you're much more comfortable with the FreeBSD installation process and know _exactly_ what you want to do, use the Express or Custom installation options. If you're upgrading an existing system, use the Upgrade option.

The FreeBSD installer supports the direct use of floppy, DOS, tape, CDROM, FTP, NFS and UFS partitions as installation media, further tips on installing from each type of media listed below.

Installing from a network CDROM

If you simply wish to install from a local CDROM drive then see the Quick Start section. If you don't have a CDROM drive on your system and wish to use a FreeBSD distribution CD in the CDROM drive of another system to which you have network connectivity, there are also several ways of going about it:

1. If you would be able to FTP install FreeBSD directly from the CDROM drive in some FreeBSD machine, it's quite easy: You simply add the following line to the password file (using the vipw command):

ftp:*:99:99::0:0:FTP:/cdrom:/sbin/nologin

And anyone else on your network will now be able to choose a Media type of FTP and type in: "ftp://<machine with CDROM drive>" after picking "URL" in the ftp sites menu.

2. If you would rather use NFS to export the CDROM directly to the machine(s) you'll be installing from, you need to first add an entry to the /etc/exports file (on the machine with the CDROM drive) which looks something like this:

/cdrom -ro ziggy.foo.com

To allow the machine "ziggy.foo.com" to mount the CDROM directly via NFS during installation. The machine with the CDROM must also be configured as an NFS server, of course, and if you're not sure how to do that then an NFS installation is probably not the best choice for you unless you're willing to read up on rc.conf(5) and configure things appropriately. Assuming that this part goes smoothly, you should be able to enter: cdrom-host:/cdrom as the path for an NFS installation when the target machine is installed, e.g. wiggy:/cdrom

Installing from Floppies

If you must install from floppy disks, either due to unsupported hardware or just because you enjoy doing things the hard way, you must first prepare some floppies for the install.

First, make your boot floppies as described in floppies/README.TXT

Second, read the file LAYOUT.TXT and pay special attention to the "Distribution format" section since it describes which files you're going to need to put onto floppy and which you can safely skip.

Next you will need, at minimum, as many 1.44MB floppies as it takes to hold all files in the bin (binary distribution) directory. If you're preparing these floppies under DOS, then THESE floppies *must* be formatted using the MS-DOS FORMAT command. If you're using Windows, use the Windows File Manager format command.
Don't trust Factory Preformatted floppies! Format them again yourself, just to make sure. Many problems reported by our users in the past have resulted from the use of improperly formatted media, which is why I'm taking such special care to mention it here!

If you're creating the floppies from another FreeBSD machine, a format is still not a bad idea though you don't need to put a DOS filesystem on each floppy. You can use the 'disklabel' and 'newfs' commands to put a UFS filesystem on a floppy, as the following sequence of commands illustrates:

```
fdformat -f 1440 fd0.1440
disklabel -w -r fd0.1440 floppy3
newfs -t 2 -u 18 -l 1 -i 65536 /dev/rfd0
```

After you've formatted the floppies for DOS or UFS, you'll need to copy the files onto them. The distribution files are split into chunks conveniently sized so that 5 of them will fit on a conventional 1.44MB floppy. Go through all your floppies, packing as many files as will fit on each one, until you've got all the distributions you want packed up in this fashion. Each distribution should go into its own subdirectory on the floppy, e.g.: a:\bin\bin.inf, a:\bin\bin.aa, a:\bin\bin.ab, ...

IMPORTANT NOTE: The bin.inf file also needs to go on the first floppy of the bin set since it is read by the installation program in order to figure out how many additional pieces to look for when fetching and concatenating the distribution. When putting distributions onto floppies, the <distname>.inf file MUST occupy the first floppy of each distribution set! This is also covered in ABOUT.TXT

Once you come to the Media screen of the install, select "Floppy" and you'll be prompted for the rest.

Installing from a DOS partition

To prepare for installation from an MS-DOS partition you should simply copy the files from the distribution into a directory called "FREEBSD" on the Primary DOS partition ("Drive C:"). For example, to do a minimal installation of FreeBSD from DOS using files copied from the CDROM, you might do something like this:

```
C:\> MD C:\FREEBSD
C:\> XCOPY /S E:\BIN C:\FREEBSD\BIN
```

Assuming that 'E:' was where your CD was mounted.

For as many 'DISTS' as you wish to install from DOS (and you have free space for), install each one in a directory under 'C:\FREEBSD' - the BIN dist is only the minimal requirement.

Once you've copied the directories, you can simply launch the installation from floppies as normal and select "DOS" as your media type when the time comes.

Installing from QIC/SCSI Tape

When installing from tape, the installation program expects the files to be simply tar'ed onto it, so after

fetching all of the files for the distributions you're interested in, simply tar them onto the tape with a command something like this:

```
cd /where/you/have/your/dists
tar cvf /dev/rwt0 (or /dev/rsa0) dist1 .. dist2
```

When you go to do the installation, you should also make sure that you leave enough room in some temporary directory (which you'll be allowed to choose) to accommodate the FULL contents of the tape you've created. Due to the non-random access nature of tapes, this method of installation requires quite a bit of temporary storage! You should expect to require as much temporary storage as you have stuff written on tape.

SPECIAL NOTE: When going to do the installation, the tape must be in the drive *before* booting from the boot floppies. The installation "probe" may otherwise fail to find it.

Now create a boot floppy as described in section 0.1 and proceed with the installation.

Installing over a network using FTP or NFS

After making the boot floppies as described in the first section, you can load the rest of the installation over a network using one of 3 types of connections:

```
Serial port:    SLIP / PPP

Parallel port:  PLIP (using ''laplink'' style
cable)

Ethernet:       A standard Ethernet controller
(including certain PCCARD devices).
```

Serial Port

SLIP support is rather primitive, and is limited primarily to hard-wired links, such as a serial cable running between two computers. The link must be hard-wired because the SLIP installation doesn't currently offer a dialing capability. If you need to dial out with a modem or otherwise dialog with the link before connecting to it, then I recommend that the PPP utility be used instead.

If you're using PPP, make sure that you have your Internet Service Provider's IP address and DNS information handy as you'll need to know it fairly early in the installation process. You may also need to know your own IP address, though PPP supports dynamic address negotiation and may be able to pick up this information directly from your ISP if they support it.

You will also need to know how to use the various "AT commands" for dialing out with your particular brand of modem as the PPP dialer provides only a very simple terminal emulator.

Parallel Port

If a hard-wired connection to another FreeBSD or Linux machine is available, you might also consider installing over a "laplink" style parallel port cable. The data rate over the parallel port is much higher than what is typically possible over a serial line (up to 50k/sec), thus resulting in a quicker installation. It's not typically necessary to use "real" IP addresses when using a point-to-point parallel cable in this way and you can generally just use RFC 1918 style addresses for the ends of the link (e.g. 10.0.0.1, 10.0.0.2, etc).

IMPORTANT NOTE: If you use a Linux machine rather than a FreeBSD machine as your PLIP peer, you will also have to specify "link0" in the TCP/IP setup screen's "extra options for ifconfig" field in order to be compatible with Linux's slightly different PLIP protocol.

Ethernet

FreeBSD supports most common PC Ethernet cards, a table of supported cards (and their required settings) being provided as part of the FreeBSD Hardware Guide (see the Documentation menu on the boot floppy or the top level directory of the CDROM). If you are using one of the supported PCMCIA Ethernet cards, also be sure that it's plugged in _before_ the laptop is powered on! FreeBSD does not, unfortunately, currently support "hot insertion" of PCMCIA cards during installation.

You will also need to know your IP address on the network, the "netmask" value for your address class and the name of your machine. Your system administrator can tell you which values are appropriate to your particular network setup. If you will be referring to other hosts by name rather than IP address, you'll also need a name server and possibly the address of a gateway (if you're using PPP, it's your provider's IP address) to use in talking to it.

If you do not know the answers to these questions then you should really probably talk to your system administrator _first_ before trying this type of installation! Using a randomly chosen IP address or netmask on a live network will almost certainly get you shot at dawn.

Once you have a network connection of some sort working, the installation can continue over NFS or FTP.

NFS installation tips

NFS installation is fairly straight-forward: Simply copy the FreeBSD distribution files you want onto a server somewhere and then point the NFS media selection at it.

If this server supports only "privileged port" access (as is generally the default for Sun and Linux workstations), you will need to set this option in the Options menu before installation can proceed.

If you have a poor quality Ethernet card which suffers from very slow transfer rates, you may also wish to toggle the appropriate Options flag.

In order for NFS installation to work, the server must also support "subdir mounts", e.g. if your FreeBSD distribution directory lives on: wiggy:/usr/archive/stuff/FreeBSD Then wiggy will have to allow the direct mounting of /usr/archive/stuff/FreeBSD, not just /usr or /usr/archive/stuff.

In FreeBSD's /etc/exports file this is controlled by the "-alldirs" option. Other NFS servers may have different conventions. If you are getting 'Permission Denied' messages from the server then it's likely that you don't have this properly enabled!

FTP Installation tips

FTP installation may be done from any mirror site containing a reasonably up-to-date version of FreeBSD. A full menu of reasonable choices for almost any location in the world is provided in the FTP site menu during installation.

If you are installing from some other FTP site not listed in this menu, or you are having troubles getting your name server configured properly, you can also specify your own URL by selecting the "URL" choice in that menu. A URL can contain a hostname or an IP address, so the following would work in the absence of a name server:

ftp://192.216.191.11/pub/FreeBSD

There are two FTP installation modes you can use:

o FTP:

For all FTP transfers, use the standard "Active" mode for transfers. This will not work through most firewalls but ill often work best with older ftp servers that do not support passive mode. If your connection hangs with passive mode, try this one!

o FTP Passive:

For all FTP transfers, use "Passive" mode. This allows the user to pass through firewalls that do not allow incoming connections on random port addresses.

NOTE: ACTIVE AND PASSIVE MODES ARE NOT THE SAME AS A 'PROXY' CONNECTIONS, WHERE A PROXY FTP SERVER IS LISTENING ON A DIFFERENT PORT!

In such instances, you should specify the URL as something like:

ftp://foo.bar.com:1234/pub/FreeBSD

Where "1234" is the port number of the proxy ftp server.

Tips for Serial Console Users

If you'd like to install FreeBSD on a machine using just a serial port (e.g. you don't have or wish to use a VGA card), please follow these steps.

1. Connect some sort of ANSI (vt100) compatible terminal or terminal emulation program to the COM1 port of the PC you are installing FreeBSD onto.

2. Unplug the keyboard (yes, that's correct!) and then try to boot from floppy or the installation CDROM, depending on the type of installation media you have, with the keyboard unplugged.

3. If you don't get any output on your serial console, plug the keyboard in again and wait for some beeps. If you are booting from the CDROM, proceed to Step 5 as soon as you hear the beep.

4. For a floppy boot, the first beep means to remove the kern.flp floppy and insert the mfsroot.flp floppy, after which you should press enter and wait for another beep.

5. Hit the space bar, then enter

boot -h

and you should now definitely be seeing everything on the serial port. If that still doesn't work, check your serial abling as well as the settings on your terminal emulation program or actual terminal device. It should be set for 9600 baud, 8 bits, no parity.

DOS user's Question and Answer section

Help! I have no space! Do I need to delete everything first?

If your machine is already running DOS and has little or no free space available for FreeBSD's installation, all is not lost! You may find the "FIPS" utility, provided in the tools/ subdirectory on the FreeBSD CDROM or on the various FreeBSD ftp sites, to be quite useful.

FIPS allows you to split an existing DOS partition into two pieces, preserving the original partition and allowing you to install onto the second free piece. You first "defrag" your DOS partition, using the DOS 6.xx "DEFRAG" utility or the Norton Disk tools, then run FIPS. It will prompt you for the rest of the information it needs. Afterwards, you can reboot and install FreeBSD on the new partition. Also note that FIPS will create the second partition as a "clone" of the first, so you'll actually see that you now have two DOS Primary partitions where you formerly had one. Don't be alarmed! You can simply delete the extra DOS Primary partition (making sure it's the right one by examining its size! :)
NOTE: FIPS does NOT currently work with FAT32 or VFAT style partitions as used by newer versions of Windows 95. To split up such a partition, you will need a commercial product such as Partition Magic

3.0. Sorry, but this is just the breaks if you've got a Windows partition hogging
your whole disk and you don't want to reinstall from scratch.

Can I use compressed DOS filesystems from FreeBSD?

No. If you are using a utility such as Stacker(tm) or DoubleSpace(tm), FreeBSD will only be able to use whatever portion of the filesystem you leave uncompressed. The rest of the filesystem will show up as one large file (the stacked/dblspaced file!). DO NOT REMOVE THAT FILE as you will probably regret it greatly!

It is probably better to create another uncompressed DOS extended partition and use this for communications between DOS and FreeBSD if such is your desire.

Can I mount my DOS extended partitions?

Yes. DOS extended partitions are mapped in at the end of the other "slices" in FreeBSD, e.g. your D: drive might be /dev/da0s5, your E: drive /dev/da0s6, and so on. This example assumes, of course, that your extended partition is on SCSI drive 0. For IDE drives, substitute "ad" for "da" appropriately. You otherwise mount extended partitions exactly like you would mount any other DOS drive, e.g.:

mount -t msdos /dev/da0s5 /dos_d

Can I run DOS binaries under FreeBSD?

Ongoing work with BSDI's doscmd utility is bringing this much closer to being a reality in FreeBSD 3.0, though it still has some rough edges. If you're interested in working on this, please send mail to emulation@FreeBSD.org and indicate that you're interested in joining this ongoing effort!

There is also a neat utility called "pcemu" in the ports collection which emulates an 8088 and enough BIOS services to run DOS text mode applications. It requires the X Window System (XFree86) to operate.

Creating a Log Class in Perl

Author: The Outrider Computing Journal

One recurrent theme in my job as a database administrator/assistant systems administrator/systems analyst is the need to keep track of what happened on the systems while I wasn't watching. What did the cron job do last night. What did all those spooler daemons do while I was at lunch? In other words logging. It bothered me that there was a lack of simple tools for doing such a simple, redundant job. So, I set out to do build some myself. My systems programming tool of choice is Perl, so, that is language I chose for the project. This journey took me out of my normal routine of straight-line Perl programming and dumped me in the land of Modules and Object Oriented Perl. I'm glad to say it didn't overwhelm me and in fact I found it rather easy to write.

My first order of business was to take my old standby logging routines and objectify them. I had several concise routines that I would either import into the main package through a use statement or just simply copy/paste depending on my mood and what I was doing. They consisted of four routines: start_logging, stop_logging, restart_logging and log.

```
sub start_logging{
    local($log_file) = @_[0];
    open(LOG,"$log_file");
    $log = 1 if LOG;
}

sub stop_logging{
    close(LOG) if ($log);
    $log=0;
    return 1;
}

sub log {
    local($string) = @_[0];
    if($log){
        print LOG &datetime,":
$string\n";
        return 1;
    }
    return 0;
}
```

Lot's of room for improvement there (I left restart_logging out because it only differed from start_logging by one character. Instead of open(LOG,"$log_file"); it had: open(LOG,"$log_file"); to append to the log file and not overwrite it). This was quick and dirty code that, while it did the job, was not very simple to use. For example, if I needed to redirect the output of a sub-process to the log file I would have to say: stop_logging(), then run that process and redirect its output to the log then restart_logging() again. It was Rather clumsy and difficult to document. So, I set about to rewrite the routines in an object oriented manner. I followed the 'Three little rules' as formulated by Larry Wall in the objperl(1) man page and restated by Damian Conway in his book "Object Oriented Perl":
Rule 1. To create a class, build a package
I created a new directory called File under one of the site_perl directories that our perl executables looks in (I found the directory by running "perl -V" and examining the @INC array). In the File directory I created a file called Log.pm. I set this file as its own package by declaring a package name at the top of the file:

```
package File::Log;
```

Get the connection between the package hierarchy and the directory structure? I included the standard packages I use for messages and file handles and such (good Perl programmers always use strict and -w):

```
use strict;
use FileHandle;
use Carp;
Rule 2. To create a method, write a
subroutine
```

```
First, I wrote the _start_log method.
sub _start_log {
    my ($self, @args) = @_;
    unlink $self-{'_logfile'};
    $self-print("Begin logging");
    return 1;
}
```

Pretty simple (we will look at the strange references in a moment). This looks nothing like the start_logging subroutine I started with. Where did I put the open for the log file? I had an inspiration. I put the open and the close statements in the print method so my programs wouldn't keep the log file open all the time. I can then read the log file while the program is running and always see the latest logging information. So, in fact I no longer even needed a start_log method, but I did want to put an entry at the top of the log file so I would have an accurate log start time recorded. All I needed to do from here is write the print method:

```
sub print {
   my ($self, @args) = @_;
   unless($self-{'_fh'}-open (".$self-
{'_logfile'})){
      croak "Failed to open \".$self--
{'_logfile'}."\"";
   }
   print {$self-{'_fh'}} $self--
_datetime().": @args\n";
   $self-{'_fh'}-close();
}
```

I have tried to sneak in some stuff without explaining it, but now I need to explain. Believe me this looks much more complicated than it really is. First, the way in which an attribute of a class is referenced is like so: $self-{_logfile} (I added the tic marks so that strict wouldn't complain about bare words). This is essentially a pointer to an attribute called _logfile that belongs to the instance of class Log called $self. Where did that $self come from in the method? When you say this: $self-print() Perl considers it a method call and it automagically passes the reference object as the first argument to the method. Hence the idiom:
`my ($self, @args) = @_;`
as the first line of the method to extract the object reference from the "real" arguments. If I had written print as a subroutine within main without the FileHandle methods it would have looked something like this:

```
sub print {
   local($logfile, @message) = @_;
   open FH, "$logfile" or croak "Failed
to open $logfile";
   print FH datetime(), ": ",@message,
"\n";
   close FH;
}
```

With all of those '-' out of the way it looks cleaner but it is not as flexible. Not to mention the fact that there would be confusion between the built-in print routine and this subroutine (Perl wouldn't be confused but programmers could easily be mislead).
Rule 3. To create an object, bless a referent

I created a constructor method. In object oriented programming the standard name for a constructor method is 'new'.

```
sub new{
  my ($class, @arg) = @_;
  my $self = bless { _logfile =
$arg[0], _fh = new FileHandle,
}, $class;
    $self-_start_log();
    return $self;
}
```

Bless is a built-in Perl function used to mark a variable as belonging to a particular class (or namespace). The 'new' method blesses the object handle ($self), calls _start_log to initiate a logging session (and write that initial line to the log) and then returns the object handle back to the calling routine so we can reference the object. _logfile and _fh are the attributes of this new object and hold the log file name and the file descriptor respectively.

Then I added the _datetime method to the class. I will leave that out of this discussion since it merely returns the date and time in a readable format to prepend to each line of the log.

That is all there is to it. But, there are a couple of things we ought to add. First of all in true object oriented programming there are two methods required for any class a constructor and a destructor. We already have a constructor so let's build a destructor. We'll use some more Perl built-in stuff to do this. Perl will automatically call a method called DESTROY for any object when it goes out of scope. It would most likely be able to handle this simple log object without any help but there is one thing I would like to do when I am done logging. I want the program to write a final line so I have a quick method to determine if a process stopped short or finished processing and have an ending time for the log file. So here is my destructor:

```
sub DESTROY {
   my ($self, @args) = @_;
   $self-print("End Logging");
   undef $self-{'_fh'};
   undef $self-{'_logfile'};
   return 1;
}
```

This nicely frees up what little space was being used by this object and writes a short message to the end of the log. I have left the destruction of the object itself for Perl to handle. Since this was so simple to do I decided to add a few niceties into the class while I was there. I added this to the top of the file.

```
{
my %_visible =
      (
       _logfile = undef,
      );

   sub _accessible {
         exists $_visible{$_[1]};
   }
}
```

A hash that has the names of the attributes that I want to make publicly accessible (its a simple matter to add to this list if I add attributes or want other attributes to be available) and an _accessible subroutine to return true if a particular attribute is accessible. The reason I wrote it this way was to make use of another Perl built-in: AUTOLOAD. If the Perl interpreter comes to a method call that doesn't exist, it looks to see if there is an AUTOLOAD method in the same class. It puts the non-existent method name in a variable called $AUTOLOAD and executes the AUTOLOAD method. In my case I only want it to return the value stored in the attribute so I wrote this:

```
sub AUTOLOAD {
   my ($self) = @_;
   $AUTOLOAD =~ /.*::get(_\w+)/
      or croak "No such  method: $AUTO-
LOAD";
   $self-_accessible($1)
      or croak "No such attribute: $1";
   return $self-{$1};
}
```

This calls _accessible to see if there is a publicly accessible attribute that looks like the method call without 'get' at the beginning and returns the value of the attribute if it exists. Got that? Let's look at it this way: I want programmers to have access to the _logfile attribute so that they can at any point recall the name of the log file being written to. So, I put _logfile in the %_visible hash, which I have told my AUTOLOAD subroutine to look at for possible matches. So if I use a method like this: $log_file_name = $log-get_logfile(); even though I haven't explicitly written a method called get_logfile it will return the contents of the attribute _logfile. Kinda neat, huh?
Now to setup and write to log files in my programs is simple:

```
Use File::Log;

# create a log file
$log = new File::Log("mylogfile.log");

# write to the log
$log-print("This is a log entry");

# redirect the sub-process to my log
file
system("my_sub_process", "2&1", "",
$log-{'_logfile'});

# write more stuff to the log
$log-print("This is another log en-
try");

# close up the logging session
undef $log;
```

I added a little bit more to the class than I described here. If you'd like a copy of the Log class in it's complete form it can be found here Log.pm.

This article is re-printed with permission. The original can be found at
http://www.diverge.org/outrider/200011plog.shtml

Releasing OSDA at AOSS2

Author: Andrew van der Stock
email: ajv@greebo.net

Friday

After a hectic week, I made my way from home to the airport with my frantically packed carry-on and laptop, and thus to the Qantas Club with unseemly haste. I had a couple in the Club before boarding my flight to Adelaide.

Once in Adelaide, I zoomed to the cafe where people from the conference were having dinner. I should have caught an earlier flight - I do like my food, but good company is so much more. Adelaide didn't disappoint on the cake and coffee front, and the company was fine. I met up again with my friend Skud and met Sarah, one of the organisers, and a few of the other speakers for the first time.

Saturday

Got up a little too early; dang that half-hour time zone difference! Breakfast being delivered is the nicest part about staying away from home, and this was no exception.

I was dropped off by the cab almost at the conference venue, but since I needed to make a booking for a hire car for the next day, I didn't mind too much. I was a little early, and managed to organise a car and still be the first person to register for the day. Conference attendees received these nifty packs with stuff in them, like Caldera's Linux Technology Preview. I thought I had every RedHat publication under the sun, so I avoided one of their folders, and so missed out on Red Hat 7.0 CD's. Not a great loss.

The conference kicked off well, with pretty good attendance for a smaller city like Adelaide. We had a quick pep talk from one of the local IT boosters, and then onto the main program.

Dan Shearer: Open Source, Opening Doors

A good talk aimed at increasing OSS usage in companies. The entry by stealth model is falling away as the desired mechanism and how you can make money doing open source.

Richard Sharpe: Cutting code in Qantas Club

Richard is probably best known for his Samba work, but this talk was more about Ethereal, which I use extensively. Richard didn't have time to discuss how he codes at the Qantas Club, but I imagine with the free booze and other distractions available there... :-)

Greg Lehey: Revamping the FreeBSD SMP implementation

Excellent talk given by a master of the trade. Greg detailed how the new SMP implementation differed from previous efforts, and the benefits of the new implementation.

Michael Still: Panda

Michael gave us a talk about his PDF enabled graphics library. Panda allows programs to directly output to PDF at the highest quality available to them. It's still a work in progress, but it seemed to work nicely.

Jay Schulist: Implementing Network Device Drivers in the Linux kernel

Jay knew his stuff and he gave an excellent presentation, showing us how easy it is to make a working network driver. Of course, it was one that he had prepared earlier, but he did run make. :-)

Lunch was nice, and I had a good chat with various people.

Geoffrey D. Bennett: The Katie revision control system

Katie is a clearcase filesystem revision control system. It worked very nicely and with a bit of polishing will be an excellent tool for developers sick of CVS.

Kirrily "Skud" Robert: Perl 6

A good talk, certainly one of the more interesting to me as they seem to be applying large scale software engineering to the open source model. I will be very interested to see how this turns out. Skud used Mr Laptop who runs Win2K. She still used a HTML presentation, though :-)

Presentations, AUUG and SAGE-AU (and ISOC-AU)

This one was a surprise for me as I didn't expect to have to do this one. So I winged it. ISOC-AU were probably unaware of it as well, as no one was there who was a member (unusual) or from the exec. I presented first and got the message across as to what SAGE-AU does for its members (which is quite a lot, but not everyone sees that).

Afternoon tea

I was pleasantly surprised to be hunted down by Phil Kernick. Phil is one of our SAGE-SA members, but SAGE-SA doesn't exist yet, and I'd like it to. Phil basically demanded to be let run it, so by the time you read this SAGE-SA should be off the ground. Yeehah! Who says conferences are a waste of time?

Glen Turner: Writing programs for future networks

Glen's talk was excellent and I managed to talk to him later about IPv6, a major pet project of mine. AARnet are likely to be an excellent test bunny for my subversive ideas. :-)

Conrad Parker: Sweep

About the only end user application presented at the conference, which made a pleasant change. Conrad showed off Sweep, a sound program that does for sound what Gimp does for graphics. Very nice. He gave out handouts with the Sweep plugin SDK.

Andrew van der Stock: OSDA

I did the only PowerPoint presentation of the entire conference! :-) I couldn't contact my ISP due to my modem dialling too fast for the hotel's poor excuse for a PABX, so Luke's magicpoint HTML simply didn't come through in time. OSDA details can be found at

Michael Neuling: Linux packet filtering

Michael, one of the authors of IP chains, gave an overview of the more flexible NetFilter which is due to appear in 2.4 when it finally finishes baking. As a security freak, I enjoyed the talk.

After the conference had finished, we headed off to the pub, and had a few drinkies. North Terrace is where the Hyundai Excel Rice Boy Car Club has their unofficial 20 km/h drag races, so we saw a wide range of tricked up Excels. Very amusing.

http://www.riceboypage.com

After the pub, we walked clear across town to a Japanese restaurant. They took a long time to serve us, which detracted from an otherwise excellent feed. Again, the company was excellent. I had turned into major pumpkin and decided to call it a night after that. The others pottered off with the change to another pub.

Sunday

Had a late breakfast and picked up the car and then Skud before driving out to Greg Lehey's place. Skud doesn't have a license I found out, and surprisingly enough for a SCA person, her navigational skills with a map were fairly rusty. Since I'm of the Dirk Gently school of thought when it comes to going places, we

missed our turn off and drove a little further than we expected to.

Once we arrived a little after the 11 am start time, we found that we were the first lot of people to turn up there that day, with Luigi seemingly staying at Greg's. Greg and his family live on acreage out in the country. It made a nice change. It's only 35 minutes out of Adelaide, but it really is the country.

They have a lovely rambling house, horses and whippets. Unfortunately, whilst we were waiting for the others to arrive, one of the family's two cats was found dead on the highway outside the property. Luigi and Greg gave it a proper burial. I felt so bad, and I thank Greg and his family for continuing on the BBQ. If Greebo or Meebles died, I would have sent everyone home whilst I had a good blubber.

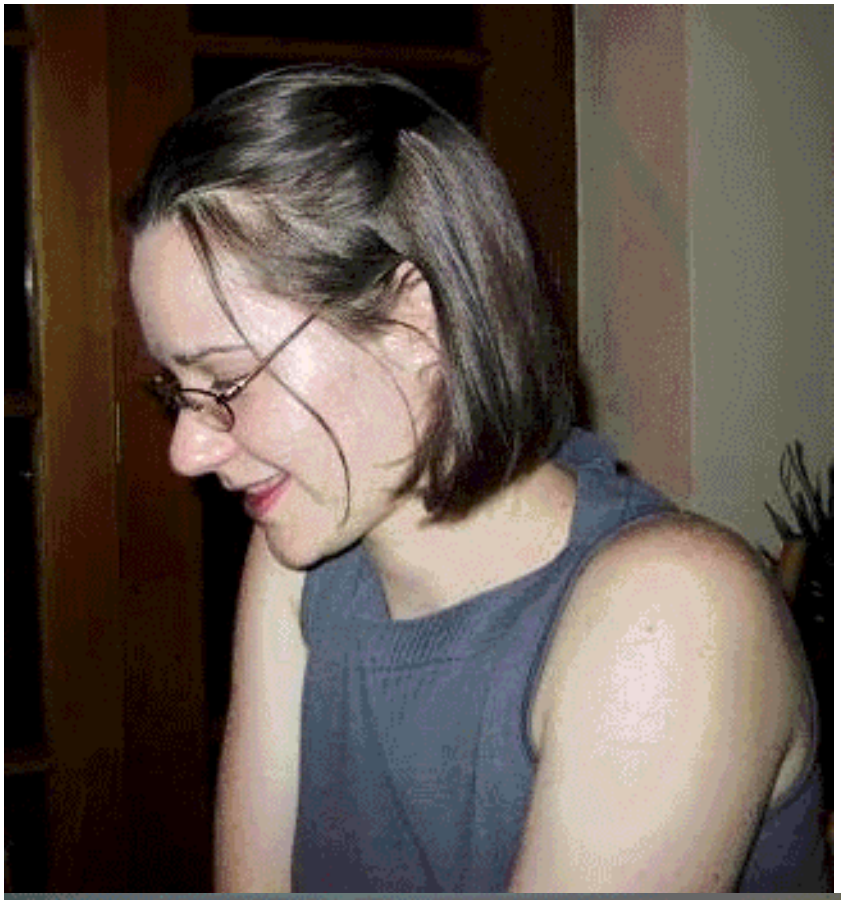
Despite this tragedy, we had a good lunch with mostly everyone turning up after getting lost in various ways. Greg gave us good directions, but unfortunately, no one had a GPS receiver and metropolitan maps do not detail every little C road, and signage in South Australia could be better.

Greg showed off his computer rooms. He has a wide variety of equipment in various stages of disrepair or working order. His guestrooms even have their own terminals.

After a long day, Skud and I departed for the airport. Skud is off to the wide white land of Canada soon. I wish her well; she'll do great at e-Smith.

Friday night

Sarah

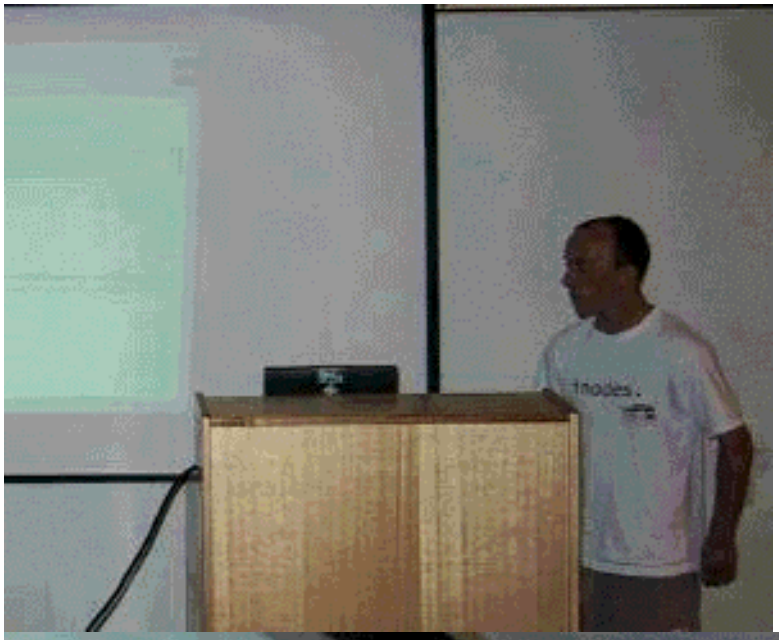


Richard Sharpe

Dan Shearer

Afternoon Tea

Conrad Parker



Glen Turner

Kirrily 'Skud' Robert

Greg Lehey

Jay Schulist

Greg Lehey and Friends

Advertisement

Red Hat Linux 7

You will find enclosed a copy of Red Hat Linux 7. This product is ideal for the experienced user, Red Hat Linux contains all the essentials to upgrade or install a workstation or server. If you would like enhanced support features and applications, then you can purchase the deluxe or professional versions of Red Hat 7. To receive a list of distributors email pacificrim@redhat.com

Red Hat 7 Deluxe

Red Hat Deluxe Workstation contains everything you need to introduce yourself to Red Hat Linux 7. In addition to the award-winning Red Hat Linux operating system, Deluxe Workstation gives you Office Suites, PowerTools, demos and full versions of numerous applications, plus services to help you get started.

Red Hat 7 Professional

Red Hat Professional Server contains everything you need to set up Red Hat Linux 7 for a serious server environment. In addition to the award-winning Red Hat Linux operating system, Professional Server gives you four bonus CDs of server-related software and two CDs of workstation applications plus the services to help you get started. Support entitlements include 30 days telephone installation support, 90 days web-based installation support, 30 days web-based apache configuration support.

Red Hat Training

Red Hat Asia-Pacific has a wide range of training courses designed for all levels of Red Hat Linux users. Red Hat offers the highest quality Linux and certification available. All Red Hat courses are taught by Red Hat instructors who have achieved their RHCE.

Skills Courses:
RH300: Red Hat Certified Engineer
Ensures that a person is ready from a technical point of view for professional responsibilities in managing a Red Hat Linux system for common uses.

Developer courses - Tracks for kernel and device driver developers and for application and GUI programmers.
RHD143: Red Hat Linux Programming Essentials
Trains you in skills for developing applications on Red Hat Linux

RHD221: Red Hat Device Driver Training
Designed to teach the experienced programmer how to develop device drivers for Linux systems

Advanced courses
RHD320: Red Hat Apache and Secure Web Server Administration
For those who desire intensive hands on training on configuration and management of an Apache web server.

For more information visit www.redhat.com/training/asiapac.
To receive a discount for AUUG members email training-au@redhat.com

The Open Source Lucky Dip

Con Zymaris
conz@cyber.com.au

Welcome back.

Hopefully you made it to either (or both of!) the AUUG Security Symposium or the Australian Open Source Symposium over the past month or so. I had the chance to make it to one of them, and I was very impressed by both the breadth and quality of the speakers. Do yourself a favour and book in the next AUUGN organised event. In this industry you need to keep your proboscis ahead of pack, and attending these highly informationally nutritious events is a good start. Righto, let's check out this edition's pickings.

###

Report is, everyone who has seen the [Windows] code is now dead. They have all laughed themselves to death.

-- From a Slashdot discussion about the Microsoft "hack"

###

Java Media Framework

A recent news item from the Blackdown Java-Linux Team announceed the availability of the JMF 2.1.1--beta2 Performance Pack for Linux/i386:

According to the notes from the Team, the Java Media Framework (JMF) is an API for incorporating audio, video and other time-based media into Java applications and applets. It is an optional package that extends the multimedia capabilities on the Java2 platform.

For general information about JMF and documentation visit *http://java.sun.com/products/java-media/jmf/index.html*

Jumpgate 0.6

Jumpgate is a TCP connection forwarder that author Patroklos Argyroudis claims that provides many enhancements and improvements over the existing programs that do the same thing. Read more here: http://zion.bsd.gr/~invisibl/projects/

Meow 1.0

From the wacky Unix tools collection, comes Meow. Apparently Meow is a text stream monitor. Like the cat command, Meow accepts lines of text from stdin and echos them to stdout. As Meow relays each line of text, it compares the line to a set of user-defined patterns, and if a match is found, it plays the corresponding sound file. The intention of all this is to make Meow is an ideal tool for monitoring system log files for specific events.

New version of AMANDA released

AMANDA (Advanced Maryland Automatic Network Disk Archiver,) for those that don't know, is a backup system that allows the administrator of a LAN to establish a single master backup server to perform backups of multiple hosts to a single tape drive. AMANDA uses native dump and/or GNU tar facilities and can back up a large number of workstations running various versions of Unix. SAMBA is also supported so that AMANDE can be used to back up Microsoft Windows 95/NT workstations and servers. More information available here: http://www.amanda.org/

Relive past ZORK glories.

Feel like blowing away many hours in the mindless pursuit of *adventure?* Just in time for the summer holidays is Frotz. Frotz is an interpreter for playing all of Infocom's text adventures and other Z-Machine games. Written by David Griffith, Frotz complies with Graham Nelson's Z-Machine standard v1.0, and runs well on most flavors of Unix. Download it from http://www.cs.csubak.edu/~dgriffi/frotz/

GUI for Embedded Linux

Linux seems to have opened up the world of embedded systems development for many developers. Palm-like devices running Linux are popping up all over. If you are at all interested in working with one of these, you'll obviously need a GUI, and since X is way too big for this form factor, you'll need another GUI. According to the people at the MiniGUI consortium, MiniGUI is a mini graphical user interface (GUI) support system for Linux which provides an event-driven API for applications. MiniGUI provides an application the ability to create multiple windows in multiple threads, and can draw in these windows without interfering with each other. MiniGUI 0.3.xx has a Graphics Abstract Layer (GAL) and an Input Abstract Layer (IAL), so MiniGUI can run on many Graphics Engines, including SVGALib and LibGGI. By using GAL and IAL, MiniGUI applications can run on X Windows as well. The MiniGUI consortium also states that this makes debugging of applications and porting to other hardware easier. MiniGUI provides native support for many image types including GIF, JPG, PCX, LBM/PBM, and BMP, support for TrueType and Adobe Type1 fonts, and support for GB2312 and BIG5 charsets. MiniGUI is available as LPGL software, so it's for all the family. More information is here: http://www.minigui.org/

NetBSD: Port it here, Port it there...

Ok, it's been around 6 months since the last release, but the NetBSD team have been busy beavers. For the minority reading this who don't know, NetBSD is a free, open source, highly portable, UNIX-like operating system available for many platforms, from 64-bit AlphaServers to handheld devices. Its followers are people who enjoy clean design and advanced features, and NetBSD is known for its reliability in production environments and its interest for research groups. NetBSD runs on thirty different system architectures featuring eleven distinct families of CPUs, and is being ported to more. But then, you knew that ;-) http://www.netbsd.org/ is, of course, home.

If you have any experiences using Linux that you would like to share with other AUUGN readers, drop us a line at:

auugn@auug.org.au

We'd love to hear from you!

linux.conf.au

Con Zymaris
conz@cyber.com.au

The forthcoming Linux technical conference, linux.conf.au, is shaping up to be a most interesting event. The number of interesting talks scheduled makes for compelling reason to attend. Even for those not using Linux, there are a plethora of worthwhile subjects covered, including Perl, Gnome, Security, DNS, MTAs, clusters, 2D/3D graphics, H.323 comms, rsync, docbook and documentation of Open Source projects. I've included the list here, as it was released by the conference organisers at the time of writing.

Keynote Speakers

Alan Cox, Dave Miller, Tridge

Papers

Cluster System Administration and GFS
- John Goebel, Ken Preslan

The Debian packaging system
- Wichert Akkerman

This talk will explain the workings and design of the Debian packagemanagement system. This system consists of a lot of compononts that work together to provide an comprehensive system for managing a Linux or other Unix-like system.It will cover everything from the lowlevel tools to manipulate packages to friendly user interfaces and utilities to make packaging easier.

e-smith server and gateway
- Kirrily 'Skud' Robert

The e-smith server and gateway is a GPL'd linux distribution aimed at small to medium sized organisations who require Internet services (mail, web server, proxies, collaboration tools, etc) without the sysadmin requirements of a general purpose Linux system. It is quick and easy to install, and is administered primarily via a web interface.This presentation takes a look at the technology underlying the e-smith server and gateway system, and discusses some of the challenges faced by e-smith in building an Open Source community of developers and users.Kirrily "Skud" Robert (will have, by the time of the conference) recently started working for e-smith in Ottawa, Canada.

GCTP and OpenFlock
- David L. Sifry

Gimp 1.2 & 2.0
- Tuomas Kuosmanen

.
Globally Distributed Content
- Horms (Simon Horman)

Electronic content made available over the Internet is becoming increasingly important for providers and users alike. To provide the best possible service to end users it is desirable for content to be network--wise as close to client hosts as possible. Static mirrors of sites are one means of distributing traffic between sites and giving users the opportunity to connect to a site that will give them a good response.Instead of users manually selecting a mirror, it makes sense for the service provider to automatically direct clients to a site that will offer them good performance, that is to have a load balancing algorithm in place. Once such algorithm is to select clients based using BGP to select which site has the least cost path to a given client. This paper will examine the implementation of such an load balancing scheme.

Hardware accelerated image blending, rendering, scaling, anti-aliased text rendering - a reality on your desktop
- Rasterman (Carsten Haitzler)

Quietly in the bowels of some CVS repository in a galaxy far far away some code has been brewing... The results? Anti-aliased text in X with full hardware acceleration AND optimized software paths included, hardware accelerated image scaling and blending, to make stunning user interfaces on the Linux desktop a reality. This paper will cover what was needed to get this far, the pitfalls of working on such a project and the great benefits and how to take advantage of the work that has gone into this.

How To Remotely Build and Manage a Linux Solution (or How Open Source Software Keeps Me From Driving to the CoLo at 3 AM)
- Gregory J. Pryzby

Linux is growing in popularity. Reason differ, but commodity hardware and open source software (OSS) are two reasons. Both drive the total cost of ownership (TCO) down. If I can manage the systems remotely and more efficiently, the TCO drops again.Using two OSS projects, VACM (http://vacm.sourceforge.net/) and SystemImager (http://systemimager.sourceforge.net/), I can remotely install, update, rebuild and power cycle systems. Once the systems are initially installed, there is no requirement for physically access the systems. All the changes and management required can be done over the internet, securely-- even BIOS changes.

Is 2D graphics the next killer app for Linux?
- Raph Levien

The compelling technical strengths of Linux and other free software systems in multitasking and networking have brought it considerable success in the area of Web servers. In this presentation, I demonstrate that the similar technical strengths free software is gaining in 2D graphics, and argue that this area could be the next "killer app" for Linux.

IA64 Linux
- Stephane Eranian

The Linux Device Filesystem
- Richard Gooch

The Device File-System (devfs) provides a powerful new device management mechanism for Linux. Unlike other existing and proposed device management schemes, it is powerful, flexible, scalable and efficient.It is an alternative to conventional disc--based character and block special devices. Kernel device drivers can register devices by name rather than device numbers, and these device entries will appear in the file-system automatically.

Linux Standard Base
Christopher Yeoh

I'll talk about the goals and reasons behind the development of the Linux Standard Base. The presentation will cover the current status of the specification, test suites and sample implementation(s). Also discussed will be a summary of the current compliance of the different distributions, and POSIX compliance of the latest kernels and glibc versions. It will describe porting and development of the test suites on Linux as well as their use as a general regression testing tool for kernel development.

Leases & Directory Notification
- Matthew Wilcox

This paper describes how the Linux kernel was extended to supply Leases & Directory notifications to applications. Leases allow an application to be notified when a file is modified, allowing that application the opportunity to cache changes until such time as they must become visible to others. Directory notification allows an application to be notified when the contents of a directory change.

Memtest: Finding holes in the VM system
- Juan J. Quintela

This paper describes the development of a test suite for the VM subsystem and several of the resulting programs in detail. A proposal for dealing with the shown bottlenecks are made. This suite of programs is called memtest. The suite is composed of several programs that generate different kind of IO and memory loads, such as writing big files (mmap*), using a lot of shared memory (ipc*), programs that do a lot of memory allocations/frees (misc*). This test suite is not usable for benchmarking, it is used to find bottlenecks.

OpenH323
- Craig Southeren

Craig is the co-founder of the Openfont23 project, an Open Source project that has been in operation for over two years. The code is currently in use by major vendors such as Nortel and provides the only Open Source H.323 protocol implementation available.H.323 is the protocol used for video and voice conferencing by programs such as NetMeeting. We can, and have, been using Linux as our primary developmnent platform and have been making voice and video calls over the planet via the Internet for over a year.

Perl 6
- Kirrily 'Skud' Robert

In July 2000, at The Perl Conference 4.0, Larry Wall announced that development would begin on Perl 6. The primary intention was to improve the internal code of the Perl interpreter and make it easier to extend and improve. At the same time, the Perl development process would be restructured to (hopefully) encourage more input from the Perl community and to otherwise improve the way in which Perl itself was developed.This presentation reviews the first few months of Perl 6 development, including the changes to the Perl community and development process, the Perl RFCs submitted as part of the pre-design brainstorming, the current state of Perl 6's design, and the future of the project and of Perl itself.Kirrily "Skud" Robert is the chair of the Perl 6 language design working group, and is actively involved in the Perl community.

Porting device drivers to the Linux 2.4 kernel
- Jonathan Corbet

Rproxy
- Martin Pool

Caches are used to good effect on today's web to improve response times and reduce network usage. For any given resource, such as an HTML page or an image, the client remembers the last instance it retrieved, and it may use it to satisfy future requests. However, the current-system is all-or-nothing: the resource must either be exactly the same as the cached instance, or it is downloaded from scratch.A far better approach would be for the server to download a description of the changes from the old instance to the new one: a 'diff' or 'delta'. rproxy adds backwards--compatible extensions to HTTP that come into operation when two parties to a web request understand the 'hsync' encoding. rproxy can be inserted as a stand-alone proxy so that neither the server nor client need be changed. We plan to integrate the rproxy into popular web software including Squid and Mozilla in the near future.

Rsync, TDB, Gzip and Apt-Proxy: A Hacker's Tale
- Rusty Russell

This meandering talk will discuss one humble coder's attempt to reduce bandwidth consumption of constantly-upgrading Debian users in the Linuxcare OzLabs office. It will follow this hacker's journey from one quick hack (apt-proxy) to a more significant hack (gzip --rsyncable), climaxing in a series of modifications to rsync itself, including a minor tour into Andrew Tridgell's Tiny DataBase. It shows how a series of small, persistant hacks have the power to change the world as we know it. Or not.

Scratching An Itch, With and Without Help
-Hugh Blemings

This paper examines open source projects with particular reference to the different challenges that come from working with and without hardware vendor

support. To provide context a discussion of the "Scratching An Itch" principle of open source development will be presented. The principle "without" case examined is gnokii, an open source project that provides tools and drivers for Nokia(TM) mobile phones under Linux, BSD and other operating systems. The primary "with" case presented is the process of writing Linux kernel drivers for the Keyspan(TM) range of USB to serial adapters.

Too Little, Too Slow: Linux 2.5 Memory Management
- Rik van Riel

In Linux 2.5 virtual memory management will see some considerable changes. One of the main problems with the current Linux memory management is that sometimes we cannot make a proper distinction between pages which are in use and pages which can be evicted from memory to make room for new data.In order to improve that situation and make the VM subsystem more resilient against wildly variable VM loads, we will use ideas from various other operating systems to improve Linux memory management. The main page replacement routine will use the active, inactive and scavenge (cache) lists as found in FreeBSD. This mechanism maintains a balance between used and old memory pages so there will always be "proper" pages around to swap. In addition to this there will probably be things like dynamic and administrator settable RSS limits, anti hog code to prevent one user or process from hogging the machine and slowing down the rest of the machine and per-user memory accounting.

Userspace: The Final Frontier, or "Fear and Loathing in sysdeps/hppa/"
David Huggins-DainesSo, your port of Linux to a new architecture works. The kernel cross-compiles, links, and boots on your hardware. You've implemented system calls and memory management, and managed to boot it into a shell prompt. You've ported Ethernet and SCSI drivers and built a few simple programs so you could test them. Wow! Isn't that cool? So, what do you do next?This talk is about bootstrapping the GNU system on a new port of Linux. This includes a discussion of kernel support for userspace, interactions between GNU libc and the kernel, and, last but not least, ELF dynamic linking. It draws on my experiences bootstrapping Debian GNU/Linux on the port of Linux to the PA-RISC architecture.

YAMA (Yet Another Mail Architecture)
- Mikolaj J. Habryn

Mikolaj presents YAMA (Yet Another Mail Architecture), featuring 350k users, geographic redundancy, no license fees and a case study in large clustered linux systems running open source software doing real work.Also touching on how to deploy and manage distributed clusters counting hundreds of individual member servers without an operations or administration team, and some of the trials and tribulations (mostly technical, some political) involved in getting such an "alternative" system accepted.

Work in Progress Presentations

bewdy, Maaate!
Silvia Pfeiffer

MPEG Maaate is an audio analysis toolkit for MPEG--encoded audio files. bewdy is a graphical user interface to play around with the analysis modules and results of the MPEG Maaate libraries. MPEG Maaate has has been published under the GNU GPL and can be found under http://www.cmis.csiro.au/dmis/Maaate/.With the vast amount of multimedia data online, content--based access to multimedia files becomes more and more interesting to users. One type of multimedia files widely used nowadays are MPEG-encoded audio files (MPEG-1 layers 1, 2, 3 (MP3)). MPEG Maaate supports the extraction of structure and content of such audio files.

GEGL: An Advanced, Flexible 2D-Imaging Library
- Manish Singh

In the field of digital imaging, there are myriad ways to store and represent image data. Many of the algorithms used for processing image data are the same, regardless of the data format. However, to efficiently and accurately process such data, the code should handle that data format natively. Creating and maintaining a library of algorithms by hand for each format you want use is time consuming and error prone.Enter GEGL. GEGL defines a generic way of describing an image processing algorithm, so you only write the code for it once. It will then autogenerate the code for the specific image format cases you want. To handle a new format, one just needs to write a backend specification, instead of reimplementing all the algorithms by hand. The algorithm descriptions themselves are simple and managable, but you get fast, optimal code autogenerated. This is similar in vein to the approach GCC takes as a retargetable compiler.

irc++
- Liam Quin

Internet Relay Chat (IRC) provides an international textual chat facility used by hundreds of thousands of people world wide. As IRC has grown, problems have become apparant both in the scalability of the implementaiton and in the network protocl it uses.This paper describes these problems in more detail and also introduces new software (irc++) intended to address these problems. The irc++ system is compatible with existing IRC clients, and also provides some MUSH/MUD/MOO-like facilities.

Outdoor Augmented Reality With Wearable Computers Running Linux
- Wayne Piekarski

This paper describes in detail the research work being performed at the Wearable Computer Lab in the University of South Australia. Our primary research work is being performed in the area of Augmented Reality (AR). AR is the process of overlaying computer--generated graphics over the real visible world in real time. It is similar to current virtual reality (VR) techniques, except the displays used are transparent. Using portable wearable computers, it is possible to build an AR system to take outdoors, allowing us to

visualise structures that only exist in a computer, along with the real world, at the same time.

State of SparcLinux on high-end servers
- Anton Blanchard

From Anton's submission: "We got SparcLinux to boot on a sun E10k a few weeks ago and I am looking into supporting such large machines better. This includes scaling better with large numbers of cpus but also supporting hot swapping of cpus, RAM and devices."Anton has now been poked for a more detailed summary. Thanks to all avid pokers out there, you may stop now.

TLB Sharing In IA-64 Linux
Alan Au, Gernot Heiser

It is well known that TLB miss handling and hence TLB coverage is often a crucial bottleneck in overall operating system performance. This fact is becoming compounded by TLB sizes that have not scaled along with a trend towards increasingly larger memory systems. The traditional approach to increasing TLB coverage is to use larger page sizes. However, this solution is non-general and, worse still, leads to possible adverse paging effects. Intel's newly released IA-64 architecture provides unique system features which allow memory translation and protection to be done orthogonally. By using this support and drawing on aspects of single address space operating system (SASOS) technology, a novel memory managment scheme for improving TLB coverage in Linux is presented here.

Tutorial Sessions

Bonobo, the GNOME component model
- George Lebl, Maciej Stachowiak

This will be a larger-format tutorial style workshop on Bonobo, the GNOME component model. Both the theory behind Bonobo and CORBA and detailed examples will be discussed. It is intended for people faimiar with GNOME programming wishing to use Bonobo in their applications. Here is an brief outline of the workshop. Time for Q & A will be given at the end of each section.

How to maintain DNS, DHCP and YP tables from a unique host file
- Marc Merlin

Introduction to using DocBook for application documentation
- Malcolm Tredinnick

Learning to use the DocBook DTD, and derivatives thereof, presents a reasonably steep initial learning curve. For many people who use DocBook for writing articles and books, the solution is often to read one of the many introductory tutorials available on the internet or to leverage some previous (open source) document and use it as a template.

Using CVS
- Malcolm Tredinnick

CVS is used throughout Open Source community as a means of providing up-to-the-minute sources for users and coordination amongst developers. Many organisations also use it internally for their developers, since, once setup correctly, it requires very little work on a day-to-day basis.

Birds of a Feather Sessions

Debian
Wichert Akkerman

Documentation
Malcolm Tredinnick

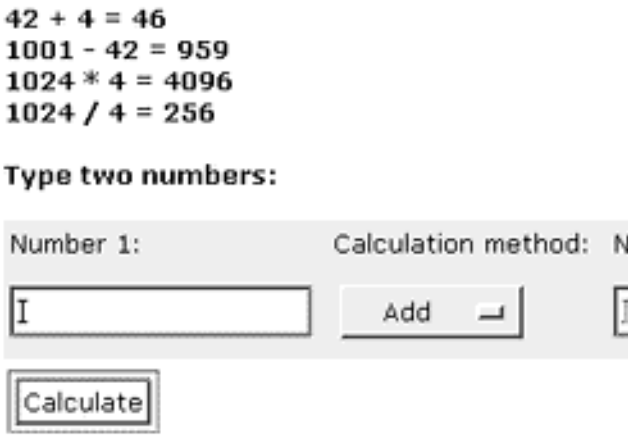
Security+
Peter Nixon

Universal Serial Bus
Brad Hards

Interfacing with Java from PHP

Author: Bård Farstad

PHP4 ext/java provides a simple and effective means for creating and invoking methods on Java objects from PHP. This article will give you a quick tutorial on how to use the PHP Java extension.



Screenshot of Java and PHP in action.

As you probably know PHP lacks some object oriented functionality. It has the basics, but stuff like: virtual functions, destructors, function overloading and private functions/variables are missing. Still PHP has many benefits, and it is a great language to use for website building. This article will demonstrate the ability PHP has to interface with a java class, using the java extension to PHP.

So what is ext/java? The README file that comes with ext/java has the following definition:

PHP4 ext/java provides a simple and effective means for creating and invoking methods on Java objects from PHP. The JVM is created using JNI, and everything runs in-process.

The Java extension is written and maintained by Sam Ruby. For details about how to install ext/java see the README file which comes with PHP. The readme file is also listed in full at the last page of this tutorial.

Java and PHP

In this small tutorial I will show you how to make a simple calculator using a java class to calculate the results and PHP to present the results and interface with the user. The example code can be tested here.

We will first start with the calculator class. I've called the class eZCalc. The code is shown below:

```
public class eZCalc
{
    float NumberA, NumberB;

    public eZCalc( )
    {
        NumberA = 1;
        NumberB = 1;
    }

    public void setA( float a )
    {
        NumberA = a;
    }

    public void setB( float b )
    {
        NumberB = b;
    }

    public float multiply()
    {
        return NumberA * NumberB;
    }

    public float divide()
    {
        return NumberA / NumberB;
    }

    public float add()
    {
        return NumberA + NumberB;
    }

    public float subtract()
    {
        return NumberA - NumberB;
    }
}
```

The magic

I've decided not to use templates in the .php file due to simplicity. If you want to know more about how to separate the html tags from the PHP code you can read the article about block templates here

The contents of the PHP file is shown at the end of the article as a whole. Study this code and I will try to explain the basic behavior of the code.

First off I want to mention the fact that you can instantiate Java classes and use them as PHP objects directly in the PHP code. The code snippet below shows how you can instantiate the eZCalc class into a PHP object called $calc. You can call member functions on the $calc object directly, as shown in the example. Note that you should manually set the type of the PHP variables sent as arguments to the Java objects. This is due to the large difference in types between PHP and Java. The example below demonstrates how you set the values used for calculation in the Java object.

```
// create a new java object, $calc
     $calc = new Java( "eZCalc" );

// force the type to be compatible with
Java types
```

```
    setType( $number1, "double" );
    setType( $number2, "double" );
    $calc->setA( $number1 );
    $calc->setB( $number2 );
```

Now lets see how the calculation works. First off we have four calculation types in our calculator; add, sub, mul and div. We switch on the different types and call up the $calc object with the correct calculation type, e.g. $result = $calc->subtract();. Here the Java object returns the value calculated inside the Java object.

In this example I have only used simple types to communicate between Java and PHP. You can also exchange arrays between Java and PHP. Objects cannot be exchanged between PHP and Java.

```
$operator = "";
    switch( $calctype )
    {
        case( "add" ):
            $result = $calc->add();
            $operator = "+";
            break;
        case( "sub" ):
            $result = $calc->subtract();
            $operator = "-";
            break;
        case( "mul" ):
            $result = $calc->multiply();
            $operator = "*";
            break;
        case( "div" ):
            $result = $calc->divide();
            $operator = "/";
            break;
    }
```

Source listing
The full source code for the example is listed below.

```
<html>
 <head>
  <title>The fantastic java calculator</title>
 </head>

 <body>

<?php

$result = 0;
if( isset( $number1 ) && isset( $number2 ) )
{
   $calc = new Java( "eZCalc" );
   setType( $number1, "double" );
   setType( $number2, "double" );
   $calc->setA( $number1 );
   $calc->setB( $number2 );
   $operator = "";
   switch( $calctype )
   {
      case( "add" ):
         $result = $calc->add();
         $operator = "+";
         break;
          case( "sub" ):
              $result = $calc->subtract();
              $operator = "-";
              break;
          case( "mul" ):
              $result = $calc->multiply();
              $operator = "*";
              break;
          case( "div" ):
              $result = $calc->divide();
              $operator = "/";
              break;
    }

    $stack .= "$number1 $operator $number2 =
$result<br>";
    print( "<h2>$stack</h2>" );
}


?>
        <h1>Type two numbers:</h1>
  <form action="ezcalc.php" method="post">
  <table bgcolor="#eeeeee" cellspacing="0"
cellpadding="3" border="0">
            <tr>
                <td>
                    Number 1:
                </td>
                <td>
                    Calculation method:
                </td>
                <td>
                    Number 2:
                </td>
            </tr>
            <tr>
                <td>
                    <input type="text"
name="number1">
                </td>
                <td>
                    <select name="calctype">
                        <option
value="add">Add</option>
                        <option
value="sub">Subtract</option>
                        <option
value="mul">Multiply</option>
                        <option
value="div">Divide</option>
                        </select>
                </td>
                <td>
                    <input type="text"
name="number2">
                </td>
            </tr>
        </table>
        <input type="Submit" value="Calculate">
        <input type="hidden" value="<?php print(
$stack ) ?>" name="stack">
        </form>
    </body>
</html>
```

This article is re-printed with permission. The original can be found at www.zez.org

---

Fun with Regular Expressions

by Adrian J. Chung
ajchung@email.com

Many of the text processing GNU tools include a powerful pattern matching mechanism called Regular Expressions. A more or less complete implementation is supported by a utility called "egrep". Other text utilities such as "gawk" also support regex's. Some tools like "sed" and "grep" support an older less powerful regex syntax. Perl implements a regex variant that has proven so popular that the same syntax is used in Python regex's.

For the rest of this article we'll be using the POSIX standard regular expressions as supported by "egrep".

"egrep" is a tool that searches for substrings. The following command will output the lines of the /etc/inittab file that contain the string "ini". Note that "ini" does not have to appear as a word by itself.

```
$ egrep ini /etc/inittab
```

We can search for digit strings also:

```
$ egrep 321 /etc/termcap
```

One may be interested only in the lines that start with the given target string. We use the special character ^ which will match the beginning of a line. To search a dictionary for words that start with "rege":

```
$ egrep ^rege /usr/dict/words
```

Similarly, special character $ can be used to match the end of a line. We can now answer the age old question -- what words end in "gry"?

```
$ egrep 'gry$' /usr/dict/words
```

The single forward quotes are needed to prevent the command shell (bash in this case) from interpreting the special characters before passing them to egrep. If you want to match any of these special characters so that they no longer have any special interpretation within egrep, precede them with a backslash:

```
$ egrep '\^Q' /etc/termcap
```

This matches a two character substring of ^ followed by "Q".

A single period matches any single character. To find all three letter words starting with "p" and ending with "n":

```
$ egrep '^p.n$' /usr/dict/words
```

Note that this pattern matches both the start and end of the line in order to force an exact match rather than just a substring. "egrep" has an option to enable this behavior so that the ^ and $ become unnecessary:

```
$ egrep -x 'p.n' /usr/dict/words
```

Sometimes the period is too general and one needs to match a more restricted range of characters. This command:

```
$ egrep 't[aeiou]p$' /usr/dict/words
```

outputs all words containing "tap", "tep", "tip", "top", or "tup". Instead of enumerating all matching characters, a range can be specified:

```
$ egrep ':[3-5][0-9]:' /etc/termcap
```

Date stamps where the minutes field is in the latter half of the hour are extracted. Ranges also work with letters:

```
$ egrep '^[n-t].[aeiou]$'
/usr/dict/words
```

This finds all three letter words beginning with the letter "n", "o", "p", ..., or "t", and ending with a vowel. Ranges and enumeration may be mixed:

```
$ egrep -x '.[aeit-z]' /usr/dict/words
```

lists all two letter words ending in "a", "e", "i", or any letter from "t" through to "z". If the leftmost character between the [] is a ^ then the match is negated:

```
$ egrep -ix '[^n-t][^aeiou].'
/usr/dict/words
```

finds all three letter words beginning in any character other than the letters "n" through "t", with a consonant for the second letter. The "-i" option makes the match case insensitive. Words like "Sri" and "DEC" are omitted.

The '\<' and '\>' combinations match the beginning and end of complete words respectively:

```
$ egrep '\< r' /etc/passwd
```

matches lines that contain words beginning with "r".

```
$ egrep 't\>' /etc/inittab
```

matches lines with words ending in "t".

The * is a repetition operator. Any pattern that immediately precedes it, can match zero or more times:

```
$ egrep 'ho*t$' /usr/dict/words
```

matches word such as "fight", "hot", and "hoot". It will even match "hooooot" if it were in the dictionary.

```
$ egrep '^s.*ho*t$' /usr/dict/words
```

shows the effect of * on special patterns like the single period. It is the equivalent of having zero or more periods in the regular expression. Words like "sleight", "snapshot", and "sharpshoot" match.

```
$ egrep -ix '[^e]*e[^e]*'
/usr/dict/words
```

finds all words that use exactly one "e"

```
$ egrep -ix '[a-ep]*' /usr/dict/words
```

finds all words spelt using the letters "a","b", "c", "d", "e", and "p" only. Similary, + makes the preceding pattern match one or more times. Expression 'ot+o$' is equivalent to 'ott*o$'.

```
$ egrep '^s.*ho+t$' /usr/dict/words
```

matches "shot" and "shoot" but not "sleight".

```
$ egrep -i 'f.+f' /usr/dict/words
```

lists words that contain two non-adjacent f's.

Here are some more repetition operators. List all three letter words:

```
$ egrep -x '.{3}' /usr/dict/words
```

All words at least 19 letters long:

```
$ egrep -x '.{19,}' /usr/dict/words
```

And words between 11 and 14 letters in length, inclusive:

```
$ egrep -x '.{11,14}' /usr/dict/words
```

Find words with at least 4 consecutive vowels:

$ egrep '[aeiuo]{4}' /usr/dict/words

Find a word with six consecutive consonants, excluding "y":

$ egrep -i '[^aeiouy]{6}' /usr/dict/words

(People that frequent central London will know this.)

The ? is equivalent to {0,1}:

$ egrep -x 'po?l.' /usr/dict/words

matches "ply", "pole", "poll" and "polo", but not "pools".

A | in the regular expression acts like a boolean OR:

$ egrep -ix 'p.n|b.+ght' /usr/dict/words

outputs words matching either 'p.n' ("pin", "pan", etc) or 'b.+ght' ("brought", "blight", etc.)

$ egrep -x 'b(ea|oo).' /usr/dict/words

matches "bead" and also "book". Patterns joined with | need not be the same length:

$ egrep '^s(ha|o)p' /usr/dict/words

matches both "shape" and "soprano". Note the use of rounded brackets to delimit the reach of the | operator. The ( ) can also define the scope of the repetition operators:

```
$ egrep -i '([aeiou][^aeiou]){7}'
/usr/dict/words
```

lists words with 7 alternations of vowel and consonant.

```
$ egrep -ix '[^s]*s([^s]+s){2,}[^s]*'
/usr/dict/words
```

finds all words containing at least 3 occurances of the letter "s", none of which are adjacent to each other.

Parenthesis have another important use. Any text that matches the pattern enclosed in the () is stored temporarily. This text can then be referred to later in the same expression. In the following command the parenthesis encloses a pattern matching any single vowel:

```
$ egrep '([aeiou])\1' /usr/dict/words
```

The \1 now refers to whatever vowel that was matched, hence this regex matches words containing "aa", "ee", "ii", "oo", or "uu". When there is more than one pattern in parenthesis, the matched text is referenced by \1, \2, etc.

```
$ egrep -x '(.)(.)\2\1' /usr/dict/words
```

matches words like "deed", "noon", etc. Each (.) matches a single letter, and the \2\1 must match these same letters but in reverse order in which they previously appeared.

```
$ egrep -ix '(.)(.)(.).*\3\2\1'
/usr/dict/words
```

lists words whose last three letters are the same as the first three letters reversed.

A more complicated example:

```
$ egrep '^(.).+\1\1.+\1$'
/usr/dict/words
```

What does it do? It returns words like "enfeeble", "gagging", and "sicknesses". The parenthesized pattern matches the first letter in the word. Any matching text must also end in this letter, and must also contain this same letter doubled somewhere in the middle.

Patterns within parenthesis can be of any length:

```
$ egrep -i '(.{5}).+\1' /usr/dict/words
```

lists words that contain a subsequence of 5 letters more than once.

And finally a really advanced example:

```
$ egrep -ix '(.).*(\1.+).*\2'
/usr/dict/words
```

find words whose last few letters are also found, adjacent and in the same order, somewhere in the middle

of the word; the initial letter of this group also being identical to the first letter of the word.

For more details see the "regex" info page, regex(7) man page (by typing "man 7 regex"). Also the "awk" and "egrep" documentation is worth checking out.

This article is re-printed with permission. The original can be found at:
http://thelinuxgurus.org/regexp.html

Becoming More Advanced in the Bash Shell

Author: Sam Rowe
deadman@deadman.org

If you've ever used GNU/Linux, chances are good that you've used bash. Some people hold the belief that using a GUI is faster than using a CLI. These people have obviously never seen someone who uses a shell proficiently. In this tutorial, I hope to show you just a few of the amazing features bash provides that will increase your productivity in the shell. Bang Bang and history

Everyone knows about bash history, right? You'd be surprised. Most modern distributions come with bash history enabled and working. If you've never done so before, try using the up and down arrow keys to scroll through your command history. The up arrow will cycle through your command history from newest to oldest, and the down arrow does, well, the opposite.

As luck would have it, different terminals handle arrow keys differently, so the brilliant minds behind bash came up with additional methods for accessing and making use of the command history. We'll start with history. This command simply gives you a numbered list of the commands you've entered with the oldest command having the smallest number. Simple right?

Here's an example of history output:

```
190  ps -axu | grep htt
191  /www/bin/apachectl start
192  vi /usr/local/lib/php.ini
193  cat /www/logs/error_log
194  ps -auxw | grep http
195  pwd
```

This brings us to bang-bang or !!. !! tells bash "repeat the last command I entered." But the magic doesn't stop there, if you order now, you'll also receive !xyz. !xyz will allow you to run the last command beginning with xyz that you typed. Be sure to add enough to the abbreviation to make it unique or you could run into problems, for instance: If you ran trn then tar xvzf mozilla.tar.gz and then tail ransom-note and typed !t you'd be looking at the ransom note again when you actually wanted to be reading news again. !tr is just enough to be unique and give you a much better chance of hitting your targeted command. :p isn't just an emoticon

If you need to be very sure of the command you're targeting, :p can be a huge help. !xyz:p will print the command that would be executed rather than executing it. :p is also clever enough to add the printed command to your history list as the last command executed (even though it didn't execute it) so that, if you decide that you like what was printed, a !! is all you need to make it happen cap'n.

Bash provides a couple of methods for searching the command history. Both are useful in different situations. The first method is to simply type history, find the number of the command you want and then type !N where "N" is the number of the command you'd like to execute. (:p works here too.) The other method is a tad more complex but also adds flexibilty. ctrl-r followed by whatever you type will search the command history for that string. The bonus here is that you're able to edit the command line you've searched for before you send it down the line. While the second method is more powerful, when doing some redundant task, it's much easier to remember !22 than it is to muck with ctrl-r type searches or even the arrow keys. Bang dollar-sign !$ is the "end" of the previous command. Consider the following example: We start by looking for a word in a file

```
grep -i joe
/some/long/directory/structure/user-
lists/list-15
```

if joe is in that userlist, we want to remove him from it. We can either fire up vi with that long directory tree as the argument, or as simply as

```
vi !$
```

A word of caution: !$ expands to the end word of the previous command. What's a word? The bash man page calls a word "A sequence of characters considered as a single unit by the shell." If you haven't changed anything, chances are good that a word is a quoted string or a white-space delimited group of characters. What is a white-space delimited group of characters ? It's a group of characters that are separated from other characters by some form of white--space (which could be a tab, space, etc.) If you're in doubt, :p works here too.

Another thing to keep in mind when using !$ is that if the previous command had no agruments, !$ will expand to the previous command rather than the most recent argument. This can be handy if, for example, you forget to type vi and you just type the filename. A simple vi !$ and you're in. Circumflex hats

Have you ever typed a command, hit return and a micro-second later realized that you made a typo? Seems like I'm always typing mroe filename. Luckily, the folks who wrote bash weren't the greatest typists either. In bash, you can fix typos in the previous command with a circumflex (^) or "hat." Consider the following:

```
vi /etc/X11/XF86config
```

```
oops!
^6c^6C
```

What happened there? The name of the file that I was trying to edit was /etc/X11/XF86Config (note the capital "C.") I typed a lower-case "c" and vi saw my error as a request for a new file. Once I closed out of vi I was able to fix my mistake with the following formula: ^error^correction.

Hats needn't be only used for errors... Let's say you have a few redundant commands that can't be handled with a wildcard, hats will work great for you. For example:

```
dd if=kern.flp of=/dev/fd0
^kern^mfsroot
```

A few handy movement commands

Sometimes a mistake is noticed before the enter key is pressed. We've already talked about terminals that don't translate cursor-keys properly, so how do you fix a mistake? To make matters worse, sometimes the backspace key gets mapped to ^H or even worse something like ^[[~. Now how do you fix your mistake before hitting the enter key?

Once again, bash comes through for us. Here are some of the movement keystrokes that I use most often: ^w erase word ^u erase from here to beginning of the line (I use this ALL the time.) ^a move the cursor to the beginning of the line ^e move the curor to the end of the line

There are more of course, but those are the ones you simply can't live without. For those who don't know the ^N notation means ctrl+N. tab-tab

One of my favorite features of bash is tab-completion. Tab-completion works in a couple of ways, it can complete filenames in the current directory or in your $PATH. Like the !commands above, you just need to give bash enough of the filename to make it unique and hit the tab key -- bash will do the rest for you. Let's say you have a file in your home directory called ransom.note, consider the following:

```
mor[tab] ran[tab]
```

Will expand to

```
more ransom.note
```

Let's say you also have a file named random in your home directory. ran above is no longer enough to be unique, but you're in luck. If you hit tab twice, bash will print the list of matching files to the screen so that you can see what you need to add to make your shortcut unique. Aliases

Using aliases is sort of like creating your own commands. You decide what you want to type and what happens when you type that. Aliases can live in a few of different places, ~/.bashrc ~/.bash_profile ~/.profile and ~/.aliases are some, but not all. In fact, you're not really limited to keeping them all in once place. Those different files behave differently based upon what kind of shell you're running, but that's beyond the scope of this document. For the purposes of this discussion, we'll settle on ~/.bash_profile (used for login shells.)

In that file, usually at the bottom, I assemble my aliases. Here's some examples:

```
alias startx='startx 2>&1 | tee ~/.Xlog &'
alias ls='ls --color=auto'
alias mroe='more'
alias H='kill -HUP'
alias getxcvs='CVS_RSH=ssh; export CVS_RSH; cvs -
d anoncvs@anoncvs.xfree86.org:/cvs checkout xc'
```

The bottom one will probably wrap, but it provides a great example of why aliases are great. A whole string of commands has been reduced to something short and easy to remember.

I hope this tutorial has been useful to you. The most difficult hurdle here is not the learning curve, but simply becoming accustomed to using these built-ins. Just like learning vi, once you get good with these, you'll be amazed you ever lived without them.

This is just the tip of the bash iceberg. If you enjoyed this, you might want to look around the Net for more bash information, or even buy a book!

This article is re-printed with permission. The original can be found at www.deadman.org

Selling in the Bazaar : How Open Source Manages Code

Author: Niramiai Rikishi
niramiai@myself.com

The second in the series, this article discusses the various models of open source software development and the reasons why open source projects succeed.

In the first part of this series [1] we looked at some of the background that surrounds open source and software development. In this part, we will discuss and focus on successful open source projects and try to understand what makes them so successful.

There are two well known open source development models. The first model, is one most famously followed by Linux [2] and in some sense Perl [3]. It is often known as the benevolent dictator (BD) model. In this model, a well known authority or established figure is the overall controller of the entire project. Overall design and long term decisions are taken by this person. All bug fixes, coding issues and implementation details could be handled by associates or maintainers for the individual parts of the code. The name for this model stems from the fact that while the leader of the project chooses to remain neutral and evaluate proposals on their merit, once in a way, a particular design decision could be made by the dictator right against the code maintainers" choice. This privilege of pulling rank in an open source project is used rather infrequently. In the case of Linux, Linus Torvalds [4], guides the code and in many ways, the overall direction that the Kernel takes (note the Kernel != Linux). In the meantime, he has been content to let companies battle to position Linux in various positions, as they see fit. By being neutral to the various niches, he can avoid seeming to lean towards any particular company or distribution. Larry Wall [5] has also performed a similar role for Perl, although with recent versions of Perl notably Perl6 [6] this BD known as a language designer, albeit that term is used as wine would be in a different bottle. This model is highly adaptable and mobile. In the truest sense of the word, this model can allow developers to modify code quickly and easily to suit requirements. This model depends highly on the BD and his ability to foresee the eventual direction of the project, manage design issues and if the need arises apply bug fixes and write quality code.

The other model is the one adopted in various degrees by OpenOffice, Apache, Mozilla, Debian [7] and the *BSD teams. In this scheme, there is no clear figurehead leader or controller. A team of various people involved in the project by their contributions, their skill and management abilities become co-ordinators. In a common scenario, it would imply that a core group of programmers actually are allowed to make design changes and set goals. A set of people at a lower level are allowed to commit code into the CVS (Concurrent Versioning System) Repository that is maintained for the code. Thus, bug fixes, security issues and other low level details can be maintained at a different level while the core group can concentrate on more important issues. In the case of each of the examples, a core group exists which is usually composed of members of the community that are well respected. From time to time, the groups may recompose themselves [8] and add or delete members. From the evidence of the BSD lineage [9], this model is prone to forking. A fork in an open source project occurs when key members are unable to agree on critical design or implementation issues. Such members are then free to stop working with the current setup, and start on a new system altogether, using the current setup as a starting point. Forking, while seemingly negative and dangerous [10], actually leads to better focus and from the evidence that is present [11], better code and better software. This model is scalable and less prone to individual idiosyncrasies. Decisions on controversial issues are mostly resolved by voting of some sort and thereby a fairer picture usually results. This scheme also allows for a much larger volunteer group to exist as the levels at which people interact are much clearly structured, leading it to be easier to manage.

In both the models, some common elements exist. A program is developed in two clear phases. The first stage is when the code is considered "experimental", "bleeding edge" or alpha. At this point, various features that are being considered are publicly debated upon and argued passionately till some form of agreement is reached. These discussions usually occur over mailing lists in which the eventual users and developers are present. Once features are fixed, various developers across the world could either in isolation or as part of a team write code that satisfies the requirements [12]. This code is then downloaded, complied, testing and sometimes even deployed by users, volunteers and developers worldwide. These development versions are often updated on a frequent basis and are usually distinguishable by a particular versioning or naming scheme. After sufficient testing and bug fixes are in place, the code is deemed to be a stable or release version. This is usually given a different version number. This version also has fewer updates and often the recommended version for average users to obtain. This phenomenon is most noted in the Linux distribution Debian which maintains a highly stable and relatively old setup for users while maintaining a highly upto date and bleeding edge version for developers to test. Most users in the open source world, when not unduly concerned about stability rarely ever install stable versions and prefer to be as bleeding edge as possible.

The essential reason why open source works lies in the manner in which the eventual users of the project guide the progress of the project right from the start. By being active parts of the process, they facilitate early prototyping and proof by demonstration about what is possible and what is not. Code is released early and released often. The source code being freely accessible allows more developers to examine the code and suggest potential improvements, bug fixes and other constructive criticism that can improve the code. Open source projects are often the clearest example of a meritocracy where you are often judged by

the merit of your code and your suggestions. In that sense, as you argue either for or against some code, there is a large amount of ego gratification that can be achieved while still being constructive. In the words of Gerald Weinberg [13], this critical element is best termed as egoless programming.

Another vital and perhaps underplayed element of the open source model is that a large amount of chaos pervades the entire architecture. Rigidity and formalism are rarely any part of an open source project. From elementary chaos theory and evolutionary biology, we know that the most adaptive, dynamic and successful systems operate at the edge of chaos. Too much rigidity and the system dies as it cannot evolve. Too much chaos and the system is destroyed. The balance is what open source projects seemingly achieve the balance and operate at maximum efficiency. Consider this: from the depths of chaos theory and the rigidity of formal complex mathematics arise the strikingly beautiful and amazing fractals [14] that like open source will remain a marvel of this information age.

References

Selling in the Bazaar : The Software Development Process De--Mystified
http://www.kernel.orghttp://www.perl.orghttp://www.cs.helsinki.fi/~torvalds/
http://www.wall.org/~larry/http://www.perl.org/perl6/
http://www.debian.org
http://www.bsdtoday.com/2000/October/News306.html
http://www.cs.ruu.nl/wais/html/na-dir/386bsd-faq/part1.html
http://users.andara.com/~sdinn/halloween.html
http://www.openresources.com/documents/halloween1/node4.html
http://www.stsc.hill.af.mil/crosstalk/2000/jul/henderson.asp
http://www.geraldmweinberg.com/Bookstuff/Each_Book/Psychology.html
http://www.unizh.ch/~chaos/mand/

This article is re-printed with permission. The original can be found at:
http://www.ciol.com/content/services/forum/artdisplay.asp?secid=4&thid=1&id=93 *g*

Version Control management with CVS - Part 1 & 2

Author: Jan Borsodi

CVS is a Version Control System which helps multiple developers manage software projects. I'll not discuss whether or not CVS is the best choice over other free and commercial Version Control Systems, I'll instead show how CVS is used and give some small tips. This first part deals with setting up CVS locally, checking out a project and getting updates.

Setup Before you start using CVS you'd better setup a couple of environment variables. All examples uses bash syntax.

First the CVS root needs to be set, this tells the CVS program were to look for projects unless another root is set explicitly. The CVSROOT variable consists of four items:

1. The protocol type, this can either be the use of pserver, remote shell execution (RSH and SSH) or locally.
2. The user name which has CVS access.
3. The server on which the repository resides.
4. The path on the server to the repository.

For instance using pserver, which is often used in local LANs, you could do

```
export
CVSROOT=:pserver:user@server:/path/to/cvsroot
```

where user is your user on the remote server, server is the remote server and the /path/to/cvsroot is the path on the remote server.

You can also have the repository on your local machine in which case you write

```
export CVSROOT=:local:/path/to/cvsroot
```

where /path/to/cvsroot is the same as the one above but now locally. If you want to you can skip the :local: prefix since it is only needed on the MS-Windows platform where the path often contains a :.

The last and most used mode on the Internet is by using so called remote shell execution, this is done with either RSH (very insecure and not recommended) or with SSH. Before you use this you need to set another shell variable, namely CVS_RSH. To use SSH you would write

```
export CVS_RSH=ssh
```

you would then change the CVSROOT to

```
export CVSROOT=user@server:/path/to/cvsroot
```

where the user, server and /path/to/cvsroot is exactly the same as in the pserver example above.

One last thing you should do is to set the default editor for committing changes. If this variable is not set CVS will use vi for editing, so unless you're a vi fanatic you probably want to change it to something more sensible for you. Good editors for this kind of use are pico, nano or nedit. You can also use Emacs for this but I would recommend using the emacsclient, it saves a lot of loading time.

So to change it to use pico you simply perform this shell command:

```
export CVSEDITOR=pico
```

If you want these variables to be set each time you log in you can add them to your ~/.bashrc or ~/.bash_profile file.

Logging in If the repository you're working is using the pserver protol, you might want to login to it to avoid typing your password on every CVS action.

However if you're very concerned with security you might want to skip this, the reason is that when you login the password is stored, although encrypted, on your local account in a file called .cvspass, but that's entirely up to you.

Logging in is done with the CVS command login. What we're going to do is to try to log in to a remote repository with an anonymous user, this gives us read-only access to the repository. To have something concrete to try it out on I'll use my own project on Sourceforge as example.

```
cvs -
d:pserver:anonymous@cvs.RegExplorer.sourceforge.-
net:/cvsroot/RegExplorer
```

login You will then be prompted for a password. This is normally were you write in your password for the remote user, but in this case we simply press Enter or Return). We are now logged in to the remote repository and no longer need to pass a password for all CVS operations.

Time to move on to the real commands.

Checking out If you already have a project that you wish to "check out" there a couple of simple steps to it:

What you first need to figure out is where the repository is located, if it's in the same place as the CVSROOT you set earlier you're set to go, if not you must supply a parameter to the CVS command.

The next thing you need to decide is which project you want to check out, this is either done with a path in the repository or with a CVS module name.

Checking out is done with the CVS command checkout

```
cvs checkout module
```

or the short version

```
cvs co module
```

where the module is either the name of the defined module on the CVS server or the relative path to the project.

To use another repository than the default you must specify this with the -d option, this is called a global option and is always put before the CVS command. So if we wanted to check out the RegExplorer project we would do

```
cvs -z3 -
d:pserver:anonymous@cvs.RegExplorer.sourceforge.-
net:/cvsroot/RegExplorer co regexplorer
```

You should now get a local copy of the CVS project. You might have noticed the use of the global -z3 option. This simply tells the CVS program to use the gzip compression when sending data over the network, this helps the transfer speed when using the Internet, the number 3 is the compression-level for the gzip program.

The local copy will present in the subdirectory regexplorer on your local machine, the module name is always used for creating a local subdirectory. You can now enter this directory, look around and make changes.

Any changes you make to this project will not have any impact on the repository version, there are two reasons for this.

The first reason is that you accessed the CVS server with an anonymous user which only gives you read--only access. To get write access you will need a user on the repository server and checkout the project with that user.

The second reason is that any changes by you on your local machine will not be incorporated to the remote server before you do a commit. How to do a commit is explained in a future part.

Updating

Updating, one of two widely used CVS commands, is vital to any cvs project. It makes sure your local copy is up to date with the remote repository.

First lets picture a scenario:

You successfully checked out the project and have used it successfully for a couple of days. You then get a message from your friend telling you that a new super cool feature has been committed to the project. And of course you're interested in getting it.

Well the solution is simple, what you need to do is to perform a CVS action called updating, this is done with the CVS command update. To do the update you first need to change the current directory to wherever you have the project locally, then you do a

```
cvs -z3 update
```

or using the short form

```
cvs -z3 up
```

there is no longer any need to specify the location of the repository this is because when you checked out the project the location was stored locally.

When you execute the command you will notice that it lists a couple of files with some letters in front of them. The letter in front gives you a clue on the status of the file. For instance

U means the file was brought up to date.
P means the file has been patched with the repository version.
A the file was added by you but not committed yet.
R the file was removed by you but not committed yet.
M the file has been modified by you but not committed yet.
C the file has been committed by you locally and in the repository so a conflict has occurred, more information on conflicts follows in the next part.
? the file is not part of the CVS project.

One thing to notice when you update is that you only get the changes in the files you already have locally. Any new files or directories will not be downloaded. To make sure you get those too you have to add an update specific option -d.

```
cvs -z3 up -d
```

The reason for not updating all files by default is that you sometimes don't want all files in a project and has explicitly checked out only a small portion of them, it can then be very annoying if the number of files just keeps increasing on each update.

Another option, which can be useful, is to stop the CVS program from recursing trough subdirectories. Sometimes you just want to perform an action in the local directory. To perform a non-recursive update you do

```
cvs -z3 up -l
```

which performs the update but only in the current directory.

One other handy trick you can do with the update command is to get a simple status report on the repository. Sometimes you're just curious if the CVS repository has changed in any way but don't want to do an actual update to find out. To do this you have to specify the global -n option. This option tells the CVS program not to perform any changes to the local files. We can then do

```
cvs -z3 -n up -d
```

we then get the same list of files as we do with a normal update but no files are changed. You can then easily pick out which files need patching, which ones are new and which ones are removed.

Conclusion

You should now have to knowledge to login, check out and update projects from a CVS repository. These are the only things you need to know if you're only interested in using other peoples projects. For instance it allows you to download bleeding edge versions of your favorite programs, no need to wait for the next binary release. You can also help a developer find bug reports in the very latest version, this is much appreciated by most, if not all, developers

Part 2.

Comitting

Committing, the second of the two widely used CVS commands (the first is updating), takes care of sending all your local changes to the CVS repository. To be able to commit your changes you need write access to the CVS repository, so any anonymous logins won't work.

Before I start explaining the commit command I'll tell you a little secret that will make your committed work so much more appreciated. The scenario is this.

You're working on a project and have successfully enhanced it by adding your own super code locally. You've tried out your personal changes several times and everything seems to work OK. Now you probably want to send these changes to the repository so you perform the commit command. However while you were busy creating your changes somebody else has done some other changes to the project which directly affects your work. If you were to commit you changes as it is now one of three things could happen:

1. No conflicts or compile problems occur for other co-developers.
2. No conflicts occur but the code will not compile with the latest CVS version.
3. Conflicts occur because you and one or more co--developers have modified the same code lines as you.

If number 3 happens you won't be able to commit because you don't have the latest version, and you cannot commit a lower version over a newer version, this means you have to update your local CVS repository. Number 1 is harmless since it works for all parts, but number 2 is something you want to avoid since it might break code. So how do you avoid number 2 you ask?

As a rule of thumb you should always, and I mean always, update before you do a commit. Why so? Well if you have the latest version locally and your new code still works, you can rest assure it will work for your co-developers as well(unless you consider conflicts).

With that said I'll continue with explaining how to commit.

Committing is done with the CVS command commit. You do this with:

```
cvs commit
```

or short

```
cvs com
```

another short form

```
cvs ci
```

Remember that you can use the global options you learned in part 1, for instance the -z3 might be useful.

When the command is run CVS will start looking trough your directories, starting in the current and recursing trough subdirectories, finding files which are modified locally. When this is done CVS will ask you for a message regarding the changes you have made, this is done with the editor set in the CVSEDITOR variable (this is explained in part 1), when you save this message and quit the editor the changes will be sent to the repository with the change message.

If you want to abort the commit you can do this by quitting the editor without saving.

However committing this way is not what will make you popular with your co-developers, this is because your change message will be appended to all modified files. For instance your bug fix in one of the code files has no use being in the README file. This means that you should commit each file separately (or at least groups of files which belong together). To do this you supply the file(s) after the commit command, for instance:

```
cvs commit README
```

which will only commit changes for the README file.

Or using several files

```
cvs commit qaregexp.cpp qaregexp.hpp
```

which will commit changes for both the C++ body and header files (C++ body and header files are closely related).

The message itself can also be specified directly to the CVS program by using the -m option. For instance

```
cvs commit -m "Changed the mail addresses to my
new one" README
```

The last word on committing is that you write sensible messages. Messages saying, fixes, stuff or other meaningless sentences are useless, spend some time writing the messages and make sure you get all changes in it. And remember the spellchecker is not your enemy.

Adding files

Being able to do changes and committing them is now possible, but how do you add new files to the repository? Thanks to marvelous new technology this is possible with a few simple steps.

The command for adding new files is add. For instance to add the installation file you do:

```
cvs add INSTALL
```

You can also add sub-directories, for instance if src is a sub-directory do:

```
cvs add src
```

But remember only the sub-directory will be added not the content, you need to add these yourself afterwards.

If you have more than one file to add you can specify them all to the command line, to add all .cpp files, the FAQ and the installation file do:

```
cvs add *.cpp FAQ INSTALL
```

After the file(s) are added you need to commit the changes, this allows you to write an nice initial message explaining why you added it.

There is one important thing you should know about files in the CVS repository, and that is that they never disappear (they never get smaller either). It will disappear from the local copy and the local files may look smaller, but in the real CVS repository all changes you, and your co-developers, have ever committed will be present.

At first this might seem strange, but consider what would happen if you wanted an earlier version of the project and the files didn't exist anymore? Well it wouldn't work. That is why CVS keeps all information sent to it.

With this said you probably wonder how you remove files from CVS. As always you do this with a CVS command and this time it's called remove.

For instance to remove the installation file we added earlier do:

```
cvs remove INSTALL
```

But remember CVS will not remove the file in the repository unless it is removed locally first(it can be renamed/moved too if you want to keep it), the file will now be marked as removed.
To perform the removal you have to do a commit (with a nice comment). The file will then be moved, in the repository, from the current directory to a sub-directory called Attic. All removed files will be put in the Attic.

Creating patches

The two previous pages explained how you can commit your changes and add files to a CVS repository if you have the proper write access. But what if you only have anonymous access and want the changes you have made locally to appear in the CVS repository?

Luckily for you there is one option and that is to create a patch and send it to someone with write access. But don't count on the changes to appear at all, it's entirely up to each developer to use the patches submitted by you. To help on this you should always send a detailed description on what you have changed and ask politely.

To create a patch you have to find out what the difference between your local copy and the CVS repository is. This is done with the CVS command diff, the usual option to use for diff is -u which outputs the difference in a unified format. You can also use the -c option which uses the context output format. So we want to do:

```
cvs diff -u
```

After a while it will start spitting out lots of text explaining the differences, to put this into a file we use redirection.

```
cvs diff -u >my.patch
```

This creates a file called my.patch which can be emailed to the developer, and hopefully will be incorporated into the CVS tree.

Version numbers

You might have noticed when working with a CVS repository that each file has a given number. For instance it might have 1.0, 1.2 or 1.8, the number is what's commonly called a revision number, which gives the user a clue on how many times the file has been modified. Files which are frequently modified will have high numbers, for instance 1.50.

Revision numbers are not be confused with program version. A program with version 1.4.2 might have files with varying revision numbers.

A version number will always have an even number of period-separated decimal integers. So a revision number of 1.3.2.2.4.5 is allowed. Revision 1.1 is by default the first version of a file. When revision numbers have more than one period it is part of a branch, branches will be explained in a later part. Normally there's no reason to care about revision numbers as CVS will automatically increase them internally. However sometimes you might want to change the revision on all your files when you have released major version of your software.
Note that using tags, which will be explained in a later part, is a much better way to handle releases.

So if you wanted all files to have a new revision number you would use the commit command with an option. For instance

```
cvs commit -r 3.0
```

will bring all your files up to revision 3.0. Revision numbers can only be increased, so trying to set a revision number of 1.3 when other files have revision 1.5 will fail.

Conclusion You should now have the knowledge to perform changes to projects, whether good or bad, with the use of CVS. You should also know how to create patches in case you just have anonymous read access. So why not help out your favorite Open Source project and make the world a better place.

The next part in this series will go into the details of CVS repository management, that is creating a new CVS repository, importing sources and handling CVS modules. For those of you who simply cannot wait I recommend you either check out the man pages for CVS, read it's PostScript version (usually located in /usr/doc/), the online manual or the PDF document.

This article is re-printed with permission. The original can be found at www.zez.org

AusCERT: Steps for Recovering from a UNIX or NT System Compromise

Author: The AusCERT Team
auscert@auscert.org.au

This document is being published jointly by the CERT Coordination Center and AusCERT (Australian Computer Emergency Response Team). It describes suggested steps for responding to a UNIX or NT system compromise. Your response should be carried out in several stages:

Introduction

This document sets out suggested steps for responding to a UNIX or NT system compromise.

Note that all actions taken during your recovery from a system compromise should be in accordance with your organization's policies and procedures.

A.Before you get started

1. Consult your security policy

2. If you do not have a security policy

i.Consult with management

Depending on how your organization is structured, it may be important to notify management in order to facilitate internal coordination of your recovery effort. Also be aware that intrusions may get the attention of the media.

ii.Consult with your legal counsel

Before you get started in your recovery, your organization needs to decide if pursuing a legal investigation is an option.

Note that the CERT Coordination Center and AusCERT (Australian Computer Emergency Response Team) are involved in providing technical assistance and facilitating communications in response to computer security incidents involving hosts on the Internet. We do not have legal expertise and cannot offer legal advice or opinions. For legal advice, we recommend that you consult with your legal counsel. Your legal counsel can provide you with legal options (both civil and criminal) and courses of action based on you or your organization's needs.

It is up to you how you wish to pursue this incident. You may wish to secure your systems or to contact law enforcement to investigate the case.

If you are interested in determining the identity of or pursuing action against the intruder, we suggest that you consult your management and legal counsel to see if any local, state, or federal laws have been violated. Based on that, you could then choose to contact a law enforcement agency and see if they wish to pursue an investigation.

We encourage you to discuss the root compromise activity with your management and legal counsel to answer the following questions:

- What is your legal status in terms of your ability to trap intruders or trace connections (i.e., do you have a login banner stating that connections can be tracked or traced? See CERT Advisory CA-92:19, "Keystroke Logging Banner").
- What are your legal responsibilities if your site is aware of the activity and does not take steps to prevent it?
- Have any local, state, or federal laws been violated?
- Should an investigation be pursued?
- Should you report the activity to local, state, or national) law enforcement?

iii.Contact law enforcement agencies

In general, if you are interested in pursuing any type of investigation or legal prosecution, we'd encourage you to first discuss the activity with your organization's management and legal counsel and to notify any appropriate law enforcement agencies (in accordance with any policies or guidelines at your site).

Keep in mind that unless one of the parties involved contacts law enforcement, any efforts to trap or trace the intruder may be to no avail. We suggest you contact law enforcement before attempting to set a trap or tracing an intruder.

U.S. sites interested in an investigation can contact their local Federal Bureau of Investigation (FBI) field office. To find contact information for your local FBI field office, please consult your local telephone directory or see the FBI's field offices web page available at:

http://www.fbi.gov/contact/fo/fo.htm

For more information, please see the web page of the FBI Washington Field Office Infrastructure Protection and Computer Intrusion Squad (WFO IPCIS):

http://www.fbi.gov/programs/ipcis-/ipcis.htm

You may wish to contact the U.S. Secret Service for incidents involving the following:

- theft or abuse of credit card information (e.g., credit card fraud, the exchange of credit card information)
- threats to the President of the United States (e.g., threatening email messages)

- impersonation of the President of the United States (e.g., the creation of forged email appearing to come from the President)

To contact the Secret Service:

> Secret Service main phone number:
> +1 202 435-7700
>
> Financial Crimes Division - Electronic Crimes Section
> Phone: +1 202 435-5850
> Fax: +1 202 435 7607

Non-U.S. sites may want to discuss the activity with their local law enforcement agency to determine the appropriate steps that should be taken with regard to pursuing an investigation.

To contact the Australian Federal Police:

Canberra	+61 2 6256 7777	Ask for the Co-ordination centre
Brisbane	+61 7 3222 1222	Ask for Operations
Sydney	+61 2 9286 4000	Ask for the Co-ordination centre
Melbourne	+61 3 9607 7777	Ask for the Co-ordination centre
Adelaide	+61 8 8419 1811	Ask for the Co-ordination centre
Perth	+61 8 9320 3444	Ask for the Co-ordination centre
Darwin	+61 8 8981 1044	Ask for the Co-ordinator

iv.Notify others within your organization

In addition to notifying management and legal counsel at your site, you may also need to notify others within your organization who may be directly affected by your recovery process (e.g., other administrators or users).

3. Document all of the steps you take in recovering

The importance of documenting every step you take in recovery can not be overstated. Recovering from a system compromise can be a hectic and time--consuming process and hasty decisions are often made. Documenting the steps you take in recovery will help prevent hasty decisions and give you a record of all the steps you took to recover, which you can reference in the future. Documenting the steps you take in recovery also may be useful if there is a legal investigation.

B.Regain control

1.Disconnect compromised system(s) from the network

To regain control, you will need to disconnect all compromised machines from your network including dial in connections. After that you may wish to operate in single user mode in UNIX or as the local administrator in NT to ensure that you have complete control of the machine; however, by rebooting or changing to single user/local administrator mode, you may lose some useful information because all processes executing at the time of discovery will be killed.

Therefore, you may wish to work through steps in section C.5. Look for signs of a network sniffer to determine if the compromised system is currently running a network sniffer.

Operating in single user mode on UNIX systems will prevent users, intruders, and intruder processes from accessing or changing state on the compromised machine while you are going through the recovery process.

If you do not disconnect the compromised machine from the network, you run the risk that the intruder may be connected to your machine and may be undoing your steps as you try to recover the machine.

2. Copy an image of the compromised system(s)

Before analyzing the intrusion we encourage you to create a backup of your system. This will provide a "snapshot" of the file system at the time that the root compromise was first discovered. You may need to refer back to this backup in the future.

If you have an available disk which is the same size and model as the disk in the compromised system, you can use the ***dd*** command in UNIX to make an exact copy of the compromised system.

For example, on a Linux system with two SCSI disks, the following command would make an exact replica of the compromised system (/dev/sda) to the disk of the same size and model (/dev/sdb).

```
# dd if=/dev/sda of=/dev/sdb
```

Please read the ***dd*** man page for more information.
There are many other ways to create a backup of your system. On NT systems there is no built in command like *dd*, but there are a number of third party applications that will make an image copy of an entire hard drive.

Creating a low level backup is important in case you ever need to restore the state of the compromised machine when it was first discovered. Also, files may be needed for a legal investigation. Label, sign, and date the backup and keep the backup in a secure location to maintain integrity of the data.

C. Analyze the intrusion

With your system disconnected from the network, you can now thoroughly review log files and configuration files for signs of intrusion, intruder modifications, and configuration weaknesses.

1. Look for modifications made to system software and configuration files

Verify all system binaries and configuration files.
When looking for modifications of system software and configuration files, keep in mind that any tool you are using on the compromised system to verify the integrity of binaries and configuration files could itself be modified. Also keep in mind that the kernel (operating system) itself could be modified. Because

of this, we encourage you to boot from a trusted kernel and obtain a known clean copy of any tool you intend to use in analyzing the intrusion. On UNIX systems you can create a boot disk and make it write protected to obtain a trustworth kernel.

We urge you to check all of your system binaries thoroughly against distribution media. We have seen an extensive range of Trojan horse binaries that have been installed by intruders.

Some of the binaries which are commonly replaced by Trojan horses on UNIX systems are: telnet, in.telnetd, login, su, ftp, ls, ps, netstat, ifconfig, find, du, df, libc, sync, inetd, and syslogd. Also check any binaries referenced in /etc/inetd.conf, critical network and system programs, and shared object libraries.

On NT systems, Trojan horses commonly introduce computer viruses or "remote administration" programs such as Back Orifice and NetBus. There have been cases where the system file that handles internet connectivity was replaced with a Trojan horse.

Because some Trojan horse programs could have the same timestamps as the original binaries and give the correct ***sum*** values, we recommend you use ***cmp*** on UNIX systems to make a direct comparison of the binaries and the original distribution media.

Alternatively, you can check the MD5 results for either UNIX or NT on suspect binaries against a list of MD5 checksums from known good binaries. Ask your vendor if they make MD5 checksums available for their distribution binaries.

Additionally, verify your configuration files against copies that you know to be unchanged.

When inspecting your configuration files on UNIX systems, you may want to:

- check your /etc/passwd file for entries that do not belong.
- check to see if /etc/inetd.conf has been modified.
- if you allow the "r-commands" (rlogin, rsh, rexec), ensure that there is nothing that does not belong in /etc/hosts.equiv or in any .rhosts files.
- check for new SUID and SGID files. The following command will print out all SUID and SGID files within your filesystem.

```
    # find / \( -perm -004000 -o -perm
-002000 \) -type f -print
```

When inspecting NT systems, you may want to

- check for odd users or group memberships.
- check for changes to registry entries that start programs at logon or services. (see LISTING 1)
- check for unauthorized hidden shares with the 'net share' command or Server Manager tool.
- check for processes that you do not identify using the pulist.exe tool from the NT resource kit or the NT Task Manager.

2. Look for modifications to data

Data on compromised systems is often modified by intruders. We encourage you to verify the integrity of web pages, ftp archives, files in users' home directories, and any other data files on your system.

3. Look for tools and data left behind by the intruder

Intruders will commonly install custom-made tools for continued monitoring or for access to a compromised system.

The common classes of files left behind by intruders are

- **Network Sniffers**

A network sniffer is a utility which will monitor and log network activity to a file. Intruders commonly use network sniffers to capture username and password data that is passed in cleartext over the network. (see section C.5 below)

Sniffers are more common on UNIX systems, but on NT systems check for key logging programs.

- **Trojan Horse Programs**

Trojan horse programs are programs that appear to perform one function while actually performing a different function. Intruders use Trojan horse programs to hide their activity, capture username and password data, and create backdoors for future access to a compromised system. (see section C.1 above)

- **Backdoors**

Backdoor programs are designed to hide itself inside a target host. The backdoor allows the user that installed it to access the system without using normal authorization or vulnerability exploitation.

- **Vulnerability Exploits**

A majority of compromises are a result of machines running vulnerable versions of software. Intruders often use tools to exploit known vulnerabilities and gain unauthorized access. These tools are often left behind on the system in "hidden" directories.

- **Other Intruder Tools**

The intruder tools listed above are not intended to be a conclusive or comprehensive list. There may be other tools left behind by an intruder. Some of the other types of tools you may find are tools to

  - ☆ probe systems for vulnerabilities
  - ☆ launch widespread probes of many other sites
  - ☆ launch denial of service attacks
  - ☆ use your computing and networking resources

- **Intruder Tool Output**

You may find log files from any number of intruder tools. These log files may contain information about other sites involved, vulnerabilities of your compromised machine(s), and vulnerabilities at other sites.

We encourage you to search thoroughly for such tools and output files. Be sure to use a known clean copy of any tool that you use to search for intruder tools.
When searching for intruder tools on a compromised system

- Look for unexpected ASCII files in the /dev directory on UNIX systems. Some of the Trojan binaries rely on configuration files which are often found in /dev.
- Look very carefully for hidden files or directories. If an intruder has created a new account and home directory then there may be hidden files or directories.
- Look for files or directories with strange names such as "..." (three dots) or ".. " (two dots and some whitespace) [UNIX]. Intruders often try and hide files within such directories. On NT systems, look for files and directories that closely match what may appear as a system file (EXPLORE.EXE, UMGR32.EXE, etc).

4. Review log files

Reviewing your log files will help you get a better idea of how your machine was compromised, what happened during the compromise, and what remote hosts accessed your machine.

Keep in mind when reviewing any log files from a compromised machine that any of the logs could have been modified by the intruder.

On UNIX systems, you may need to look in your /etc/syslog.conf file to find where syslog is logging messages. NT systems generally log everything to one of three logs for NT events, all of which are viewed through the Event Viewer. Other NT applications such as IIS server may log to other locations. IIS by default writes logs to the c:\winnt\system32\logfiles directory.

Below is a list of some of the more common UNIX log file names, their function, and what to look for in those files. Depending on how your system is configured, you may or may not have the following log files.

- **messages**

The **messages** log will contain a wide variety of information. Look for anomalies in this file. Anything out of the ordinary should be inspected. Also, look for events that occurred around the known time of the intrusion.

- **xferlog**

If the compromised system has a functioning ftp server, **xferlog** will contain log files for all of the ftp transfers. This may help you discover what intruder tools have been uploaded to your system, as well as what information has been downloaded from your system.

- **utmp**

This file contains binary information for every user currently logged in. This file is only useful to determine who is currently logged in. One way to access this data is the ***who*** command.

- **wtmp**

Every time a user successfully logs in, logs out, or your machine reboots, the wtmp file is modified. This is a binary file; thus, you need to use a tool to obtain useful information from this file. One such tool is ***last***. The output from ***last*** will contain a table which associates user names with login times and the host name where the connection originated. Checking this file for suspicious connections (e.g., from unauthorized hosts) may be useful in determining other hosts that may have been involved and finding what accounts on your system may have been compromised.

- **secure**

Some versions of UNIX (RedHat Linux for example) log tcp wrapper messages to the **secure** log file. Every time a connection is established with one of the services running out of inetd that uses tcp wrappers, a log message is appended to this log file. When looking through this log file, look for anomalies such as services that were accessed that are not commonly used, or for connections from unfamiliar hosts.

The common item to look for when reviewing log files is anything that appears out of the ordinary.

5. Look for signs of a network sniffer

When a system compromise occurs, intruders could potentially install a network monitoring program on UNIX systems, commonly called a sniffer (or packet sniffer), to capture user account and password information. For NT systems, remote administration programs would be more commonly used for the same purpose.

The first step to take in determining if a sniffer is installed on your system is to see if any process currently has any of your network interfaces in promiscuous mode. If any interface is in promiscuous mode, then a sniffer could be installed on your system. Note that detecting promiscuous interfaces will not be possible if you have rebooted your machine or are operating in single user mode since your discovery of this intrusion.

There are a couple of tools designed for this purpose.

- **cpm** - UNIX

available for download from:

ftp://coast.cs.purdue.edu/pub/tools/unix/cpm/

- **ifstatus** - UNIX

available for download from:

ftp://coast.cs.purdue.edu/pub/tools/unix/ifstatus/

Keep in mind that some legitimate network monitors and protocol analyzers will set a network interface in promiscuous mode. Detecting an interface in promiscuous mode does not necessarily mean that an intruder's sniffer is running on a system.

Another issue to consider is that sniffer log files tend to grow quickly in size. You may want to use utilities such as ***df*** to determine if part of the filesystem is larger than expected. Remember that ***df*** is often replaced by a Trojan horse program when sniffers are installed; therefore, be sure to obtain a known clean copy of that utility if you do use it.

If you find that a packet sniffer has been installed on your systems, we strongly urge you to examine the output file from the sniffer to determine what other machines are at risk. Machines at risk are those that appear in the destination field of a captured packet, but if passwords across systems are common or if the source and destination machines trust each other the source machine will also be at further risk.

Many common sniffers will log each connection as follows:

```
     -- TCP/IP LOG -- TM: Tue Nov 15 15:12:29 --
     PATH: not_at_risk.domain.com(1567) =>
at_risk.domain.com(telnet)
```

For sniffer logs of this particular format, you can obtain a list of affected machines by executing the following command:

```
     % grep PATH: $sniffer_log_file |
awk '{print $4}' | \
       awk -F\( '{print $1}'| sort -u
```

You may need to adjust the command for your particular case. Also, some sniffers encrypt their logs so they may not be obvious. Because of this check for files that grow quickly.

You should be aware that there may be other machines at risk in addition to the ones that appear in the sniffer log. This may be because the intruder has obtained previous sniffer logs from your systems or through other attack methods.

For more information, we encourage you to review CERT Advisory CA-94:01, available from:

http://www.cert.org/advisories/CA-94.01.ongoing.network.monitoring.attacks.html

The advisory describes of sniffer activity and suggests approaches for addressing this problem.

Please send us a list of all hosts you know to be affected. This will help us determine the scope of the problem.

If Australian or New Zealand hosts have been involved, please inform auscert@auscert.org.au.

6. Check other systems on your network

We encourage you to check all of your systems, not just those that you know to be compromised. In your check include any systems associated with the compromised system through shared network-based services (such as NIS and NFS) or through any method of trust (such as systems in hosts.equiv or .rhosts files, or a Kerberos server).

In examining other systems on your network, we encourage you to use our Intruder Detection Checklists:

http://www.cert.org/tech_tips/intruder_detection_checklist.html
http://www.auscert.org.au/Information/Auscert_info/Papers/win_intruder_detection_checklist.html

7. Check for systems involved or affected at remote sites

While examining log files, intruder output files, and any files modified or created during and since the time of the intrusion, look for information that leads you to suspect that another site may be linked with the compromise. We often find that other sites linked to a compromise (whether upstream or downstream of the compromise) have often themselves been victims of a compromise. Therefore it is important that any other potential victim sites are identified and notified as soon as possible.

D. Contact the relevant CSIRT and other sites involved

1. Incident Reporting

Intruders will frequently use compromised accounts or hosts to launch attacks against other sites. If you find evidence of compromise or intruder activity at any other sites, we encourage you to contact those sites. Tell them what you have found, explain that this may be a sign of compromise or intruder activity at their site, and suggest that they may wish to take steps to determine if/how the compromise occurred and prevent a recurrence. When contacting other sites, please give them as much detail as possible including date/timestamps, timezone, and what to do if they have follow-up information.

We would appreciate a "cc" to cert@cert.org or auscert@auscert.org.au as appropriate on any correspondence. If you like, you can let the site know that you are working with us on this incident (please include the assigned CERT or AusCERT tracking number in the subject line of your messages). Also let them know that we can offer assistance on how to recover from the compromise.

2. Contact AusCERT - Australian Computer Emergency Response Team

We would appreciate it to be informed of any incidents involving Australian and New Zealand sites as it helps us to gauge the extent and nature of intruder activity.

Our contact information is as follows:

> Internet: auscert@auscert.org.au *monitored during business hours (GMT+10:00)*
> Telephone: +61 7 3365 4417 *monitored during business hours (GMT+10:00)*
> Hotline: +61 7 3365 4417 *monitored 24 hours, 7 days for emergencies (GMT+10:00)*
> Facsimile: +61 7 3365 7031

Australian Computer Emergency Response Team
The University of Queensland
Brisbane
Qld 4072
AUSTRALIA

3. Contact the CERT Coordination Center

We would appreciate it if you would complete and return an Incident Reporting Form as this will help us better assist you, and allow us to relate ongoing intruder activities. This also provides us a better overview of trends in attack profiles and provides input for other CERT documents such as Advisories and Summaries. We prefer that Incident Reporting Forms are sent to us via email. The Incident Reporting Forms are available from:

> http://www.cert.org/ftp/incident_reporting_form

Our contact information is as follows:

> Email: cert@cert.org
> (monitored during business hours)

Telephone: +1-412-268-7090 24-hour hotline
Fax: +1-412-268-6989

CERT Coordination Center personnel answer business days (Monday-Friday) 08:30-17:00 EST/EDT (GMT-5)/(GMT-4), on call for emergencies during other hours.
CERT Coordination Center
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA USA 15213-3890

4. Obtain contact information for other sites involved

If you need contact information for a .COM, .EDU, .NET, or .ORG top-level domain, we encourage you to use the InterNIC's whois database.

> http://rs.internic.net/tools/whois.html

To find contact information from the appropriate registrar, we encourage you to use the InterNIC's Registrar Directory:

> http://rs.internic.net/origin.html

To find contact information for the Asia-pacific region and Australia respectively:

> http://www.apnic.net/apnic-bin/whois.pl
>
> http://www.aunic.net/cgi-bin/whois.aunic

To find contact information for other incident response teams, you may also want to check the contact list of the Forum of Incident Response and Security Teams (FIRST), available in:

> http://www.first.org/team-info/

More information about finding site contacts is available from:

> http://www.cert.org/tech_tips/finding_site_contacts.html

We do not recommend sending email to "root" or "postmaster" of a machine that is suspected of being involved in intruder activity. If that machine is the source of an intruder attack, it is possible that that machine itself may be compromised and the intruder may have root access and/or be reading or intercepting email sent to that host.

If you are still unsure of a site or contact details, please get in touch with us.

E. Recover from the intrusion

1. Install a clean version of your operating system

Keep in mind that if a machine is compromised, anything on that system could have been modified, including the kernel, binaries, datafiles, running processes, and memory. In general, the only way to trust that a machine is free from backdoors and intruder modifications is to reinstall the operating system from the distribution media and install all of the security patches before connecting back to the network. Merely determining and fixing the vulnerability that was used to initially compromise this machine may not be enough.
We encourage you to restore your system using known clean binaries. In order to put the machine into a known state, you should re-install the operating system using the original distribution media.

2. Disable unnecessary services

Configure your system to offer only the services that the system is intended to offer and no others. Check to ensure that there are no weaknesses in the configuration files for those services and that those services are available only to the intended set of other systems. In general, the most conservative policy is to start by disabling everything and only enabling services as they are needed.

3. Install all vendor security patches

We strongly encourage you to ensure that the full set of security patches for each of your systems is

applied. This is a major step in defending your systems from attack and its importance cannot be overstated.
We encourage you to check with your vendor regularly for any updates or new patches that relate to your systems.

4. Consult AusCERT advisories and external security bulletins

We encourage you to consult past AusCERT advisories and external security bulletins and to follow the instructions that are relevant to your particular configuration. Be sure that you have installed all applicable patches or workarounds described in the AusCERT publications.

Remember to check the advisories periodically to ensure that you have the most current information.
Past AusCERT advisories are available from:

http://www.auscert.org.au/Information/Advisories/aus_advisories.html
ftp://ftp.auscert.org.au/pub/auscert/advisory/

External Security Bulletins are available from:

http://www.auscert.org.au/Information/Advisories/esb_advisories.html
ftp://ftp.auscert.org.au/pub/auscert/ESB/

5. Consult CERT advisories, summaries, and vendor--initiated bulletins

We encourage you to consult past CERT advisories, summaries, and vendor-initiated bulletins and to follow the instructions that are relevant to your particular configuration. Be sure that you have installed all applicable patches or workarounds described in the CERT publications.

Remember to check the advisories periodically to ensure that you have the most current information.
Past CERT advisories are available from:

http://www.cert.org/advisories/

Past CERT summaries are available from:

http://www.cert.org/summaries/

Vendor-initiated bulletins are available from:

ftp://ftp.cert.org/pub/cert_bulletins-/

6. Caution use of data from backups

When restoring data from a backup, ensure that the backup itself is from an uncompromised machine. Keep in mind that you could re-introduce a vulnerability that would allow an intruder to gain unauthorized access. Also, if you are only restoring users' home directories and data files, keep in mind that any of those files could contain Trojan horse programs. You may want to pay close attention to .rhosts files in users' home directories.

7. Change passwords

After all security holes or configuration problems have been patched or corrected, we suggest that you change the passwords of **ALL** accounts on the affected system(s). Ensure that passwords for all accounts are not easy to guess. You may want to consider using vendor-supplied or third-party tools to enforce your password policies.

AusCERT has published the Choosing good passwords article which contains information to educate users to choose good passwords.

F. Improve the security of your system and network

1. Review security using the UNIX or NT Configuration Guidelines document

To help you assess the security of your system(s), please refer to our UNIX or NT Configuration Guidelines documents. These documents may be useful when checking your system for common configuration problems that are often exploited by intruders.

http://www.cert.org/tech_tips/unix_configuration_guidelines.html

http://www.auscert.org.au/Information/Auscert_info/Papers/win_configuration_guidelines.html

2. Review the security tools document

Consider using some of the software security tools that are available, such as Tripwire®, COPS, and the TCP wrapper package.

A description of some tools that can be used to help secure a system and deter break-ins is available from:

http://www.cert.org/tech_tips/security_tools.html

3. Install security tools

Install all security tools before you connect your machine back to the network. Also, this is a good time to take an MD5 checksum snapshot of the newly restored system using a tool such as Tripwire®.

4. Enable maximal logging

Make sure that logging/auditing/accounting programs are enabled (for example, process accounting) and that they are set to an appropriate level (for example, sendmail logging should be level 9 or higher). Backup your logs and/or consider writing your logs to a different machine, to an append-only file system, or to a secure logging host.

5. Configure firewalls to defend networks

Consider filtering certain TCP/IP services at your firewall server, router or at the hosts. For some suggestions, please refer to "Packet Filtering for Firewall Systems," available from

http://www.cert.org/tech_tips/packet_filtering.html

G. Reconnect to the Internet

If you disconnected from the Internet, the best time to reconnect is after you have completed all the steps listed above.

1. Update your security policy

The CERT Coordination Center recommends that every site develop their own computer security policy. Each organization may have a specialized culture and security requirements that are specific to their own organization. Please refer to RFC 2196 "Site Security Handbook" for information about developing computer security policies and procedures for sites that have systems on the Internet. This document is available from

ftp://ftp.isi.edu/in-notes/rfc2196.txt

2. Document lessons learned from being compromised

Document and review the lessons you learned from going through the process of recovering from a compromise. This will help you decide exactly how to revise for your security policy.

3. Calculate the cost of this incident
For many organizations, changes simply are not made in security policy until they understand the cost of security, or lack thereof. Calculating the cost of an incident will help measure the importance of security for your organization. You may find that calculating the cost of this incident is useful for explaining to management that security is important to your organization.
4. Incorporate necessary changes (if any) in your security policy

Making changes to your security policy is the last step to take in this process. Be sure to inform members of your organization about the changes that have been made and how that may affect them.

Questions or comments regarding this article? auscert@auscert.org.au

Summrary of Minutes from AUUG Exec Meeting

By: Liz Carroll

5 August 2000
10:00am – 4:00pm
UTS, Sydney NSW

Notetaker:	Elizabeth Carroll	EC
Attendees:	Elizabeth Carroll	EC
	David Purdue	DP
	Luigi Cantoni	LC
	Michael Paddon	MP
	Malcolm Caldwell	MC
	Peter Gray	PG
	Alan Cowie	AC
	Sarah Bolderoff	SB
	Greg Lehey	GL
Guest:	Frank Crawford	FC
Apologies:	David Newall	DN

NEW COMMITTEE

Motion to co-opt Greg Lehey and Sarah Bolderoff onto committee. They were subsequently thanked for volunteering.

PRESIDENT'S REPORT

Some procedural changes were discussed, before the President's report was given. There will be 4 meetings this year. All reports must be submitted one week prior to the meeting, with an archive of reports kept.
Procedures will be codified this year - HTML is the most logical way to do this and then store it on the AUUG website.

AUUG2K

We held a successful conference, although numbers were a little disappointing.

Planning

The main focus for this meeting is planning for the next year. AUUG2001 is more than a year off, and while we cannot ignore it we will have to focus on other activities. In particular, we need to increase our membership, and we need to codify our procedures and get our corporate memory in order.

SECRETARY'S REPORT

Current membership statistics (as at 2000/08/03).

* 664 members:

Individual Member	432	65%
Corporate Member	187	28%
Student Member	21	3%
Freebies	17	3%
Subscription	5	
Life Member	2	
NSW	194	29%
VIC	151	22%
ACT	116	17%
QLD	91	14%
WA	44	7%
SA	25	4%
OTHER	19	3%
TAS	17	3%
NT	7	1%

Correspondance of note.

Following an email complaining bitterly of our association with the ORBS anti spam service, discussion ensued on the exec list with divided opinion regarding the nature and merit of the ORBS service. MP's suggestion would be to use MAPs – in accordance to a response from Jeremy Bishop.

WA Society for Computers and the Law, would like to form an alliance with AUUG.

Directory of Australian Associations requires updated contact details for AUUG.

DP requested MP to submit annual reports to the companies register in VIC. We need to have our details up to date with them.

TREASURER'S REPORT

The Treasurer presented the AUUG Budget for 1 July 2000-30 June 2001

AUUG is solvent
A) Even if our memberships were nil, we are solvent for the year
B) Monies required by 1st July 2001, as we will need funds for the AUUG 2001 conference

Budget shows that we need the conference to happen, as the membership alone does not sustain AUUG.

BUSINESS MANAGER'S REPORT

The past few weeks, since AUUG2K, had been spent with the general post conference clean up of issues, as well as ensuring that all the 30 June 2000 membership renewals were mailed out.

Accounts

The majority of bills have been received from AUUG2K, just one or two small ones left, so most of this is up to date.
All other accounts are up to date.

AUUG2K

Some outstanding payments are yet to come in, with invoices being mailed to those outstanding

AUUG 2001

Venues are currently being sourced.

AUUGN

Gunther Feuereisen stepping down as Editor.

Systems Magazine

An article has been written for the August edition by Con Zymaris, basically a summary of the events at AUUG2K

MINUTES OF PREVIOUS MEETING

Last meeting in Canberra was not quorate, therefore notes have been produced, as opposed to minutes.

AUUG2K

FC stated that given the fact that there was short preparation time, and out of time frame, things went well. Some problems with speakers pulling out at short notice. Conference and tutorials went very well. Negative side was the short preparation time.

LC stated that FC's availability at the front desk was excellent, as any problems were rectified almost immediately. In this respect, the Programme Chair should probably have an assistant.

PG stated that looking at the Evaluation Results, that Security is still a very popular topic, and should always be included.

Comment was made that there was a tutorial that finished in about half the time that it should have. At future events, we need to look at using known presenters to try and ensure this does not happen. An idea would be to use a questionnaire on application, ie how many tutorials have you presented, where etc. PG will produce a Guide for Tutors .

Sunday was a good idea and worked well, with no complaints.

Venue – DP was very pleased with this – the rooms were actually designed for presentations. All the correct seating and equipment was there.
Looking at the response to the venue, meeting rooms was good, meals and refreshments – not so good.
Some mixed comments were received in regards to the venue, i.e. some delegates stated that it was too far from the accommodation.

DP moved a vote of thanks to FC.

EVENTS

- **AUUG 2001**

Received topics from the AUUG2K Evaluations, may be premature to use these right now.
Most important thing is the date and venue.
AUUG2001 will be run in Sydney 23-28 September 2001.

- **AOSS**

Date to be held , Saturday 25 November 2000, Adelaide.

· Security Symposium

Idea this year is to present this on Friday 3 November 2000, followed by the Exec Meeting on Saturday 4 November.

· Installfest

GL stated that this was a very successful event. Their expectation was 200, but they estimate they had about 500.

· Other Events

Two more events should be held in the beginning half of next year.

Ideas:

A) Unix Systems Administration
B) Heterogeneous Networking
C) High Level Scripting Languages
D) eCommerce and WebServing

MEMBERSHIPS AND RENEWALS

Need to increase our membership numbers so that AUUG is more self-sustaining. It is currently sustained by events, would like it sustained by membership, so that profit from events can be ploughed back into running bigger/better events and increase and improve membership benefits. We need to look at retaining members, as well as recruiting.

As we increase membership, it increases both income and expenditure. We need to look at our fixed and variable costs in relation to this.

Looking at the budget, if we were to double our membership figures it would result in a very healthy organisation.

AUUGN

Discussed at the last AUUG Exec get together – suggestion payment of $200 for appropriately refereed articles.
Motion to pursue this action.Title for this is the Reviewed Article Track .

Editor

Gunther will be standing down.

NEW COMMITTEE

President:		David Purdue
DP		
Vice President:		Malcolm Caldwell
MC		
Treasurer:		Luigi Cantoni
LC		
Secretary:		Michael Paddon
MP		
General Committee Members:		Peter Gray
PG		
		Alan Cowie
	AC	
		David Newall
	DN	
		Sarah Bolderoff
	SB	
		Greg Lehey
GL		

EC to confirm the names of the Returning and Assistant Returning Offier.

OTHER BUSINESS

John Lions account – monies to be put into a high interest fund.

Meeting Closed: 5.10pm
Next Meeting: 4 November 2000

Unix Traps and Tricks

Jerry Vochteloo
jerry@vochteloo.org

Welcome again to Unix Traps and Tricks. Keeping to the primer theme (and since I have not had time to do anything tricky in a while or received any other submissions), this quarters installment is on UNIX directory permissions.

I know this is the bad end of the year, but we have a new year coming up in which we will all have more time :) so if anyone would be willing to write something on any UNIX related topic that would be great. Submissions to jerry@vochteloo.org please.

Short primer on UNIX directory permissions

Last time I talked about permissions on files in UNIX, this time I will talk about what those permissions mean on directories.

In UNIX a directory is a file. The main difference between files and directories is that users cannot write to directories directly. All operations on directories are done by the operating system, some indirectly; ie creating a file will add an entry to the directory, deleting a file will remove an entry from the directory; and some more directly ie mkdir, and rmdir.

To understand how permissions on a directory work we have to know what is in a directory: Simply a file which contains a list of directory entries (dirents). Each directory entry contains:

An inode number (file information data structure), and the name of the file.

If we are searching for a file, the operating systems will search for the names in the directory, until it finds a match. Once a match is found, the corresponding inode number is returned and we can start operating on the file (Note this is basically what namei does).

Now to what the permissions mean. First of all we will take read permissions. If you have read permissions on a directory you can list the contents of the directory, ie the names of the files, and you can search the directory. ie without read permissions you could not do an ls on the directory.

```
unix> mkdir directory
unix> touch directory/file
unix> ls -l directory
total 0
-rw-------   1 jerry    jerry           0 Nov 22
09:51 file
unix> chmod a-r directory/
unix> ls -ld directory
d-wx------   2 jerry    jerry        4096 Nov 22
09:51 directory
unix> ls directory
/bin/ls: directory: Permission denied
```

Write permissions on a directory allow you to modify directory entries (through the operating system of course). This allows you to delete files, create files and rename files. This means that even if you have write permissions on the file itself, you cannot delete it from the directory if you do not have write permissions on the directory (Note that you can set the length of the file to zero, effectively deleting the contents of the file, but you cannot remove the file itself)

```
unix> chmod u-w directory/
unix> ls -ld directory/
dr-x------   2 jerry    jerry        4096 Nov 22
09:56 directory/
unix> ls -l directory/
total 0
-rw-------   1 jerry    jerry           0 Nov 22
09:56 file
unix> rm directory/file
rm: cannot unlink ‘directory/file’: Permission
denied
```

Finally, the hardest permission to understand (and to explain) is the execute permission on a directory. I basically think of it as permission to access the inodes that are pointed to by dirents. This means that, if the execute bit is not set, no matter what permissions you have on the files themselves, you cannot operate on the files. Further, without the execute permissions on a directory you cannot do an ls -l, as this would need to get the information from the inode.

```
unix> mkdir dirn
unix> touch dirn/file1
unix> touch dirn/file2
unix> ls -l dirn
total 0
-rw-------   1 jerry    jerry           0 Nov 27
10:58 file1
-rw-------   1 jerry    jerry           0 Nov 27
10:58 file2
unix> chmod a-x dirn
unix> ls -l dirn
/bin/ls: dirn/file1: Permission denied
/bin/ls: dirn/file2: Permission denied
total 0
unix> ls dirn
file1  file2
unix> cat dirn/file1
cat: dirn/file1: Permission denied
```

Tricks

As well as the three permissions read, write and execute, there are a couple of other permission bits that can be set on a directory; these are the set group id (set gid) and the sticky bit.

First of all, the set gid bit, indicates that all files that are created in that directory get the same group as the directory, not the group of the user. This is useful if you have sets of group work projects.

```
unix> mkdir tmp
unix> ls -ld tmp
drwx------   2 jerry    jerry        4096 Nov 27
09:00 tmp
unix> chgrp src tmp
unix> ls -ld tmp
drwx------   2 jerry    src          4096 Nov 27
09:00 tmp
unix> id
uid=666(jerry) gid=666(jerry)
unix> groups jerry
jerry : jerry src
unix> touch tmp/file1
unix> ls -l tmp/file1
```

```
-rw-------   1 jerry    jerry           0 Nov 27
09:01 tmp/file1
unix> chmod g+s tmp
unix> ls -ld tmp
drwx--S---   2 jerry    src          4096 Nov 27
09:01 tmp
unix> touch tmp/file2
unix> ls -l tmp
total 0
-rw-------   1 jerry    jerry           0 Nov 27
09:01 file1
-rw-------   1 jerry    src             0 Nov 27
09:01 file2
```

From the above you can see that once we set the set gid bit on the directory, the file that we create in that directory has the same group as the directory, not as the user that created the file.

The second extra bit on a directory is the sticky bit. This is useful in world writable directories that lots of people are supposed to use (such as /tmp or /var/tmp). Normally if a directory is world writeable (and executable) any user can remove any files from that directory. With the sticky bit set; only the owner of the directory, the owner of the file, anyone with write permission on the file (and of course root), can delete the file. Hence users can create files in this directory, without other being able to remove those files.

```
unix> mkdir test2
unix> chmod +t test2
unix> ls -ld test2
drwx-----T   2 jerry    jerry        4096 Nov 27
16:17 test2
unix> chmod a+rwx test2
unix> ls -ld test2
drwxrwxrwt   2 jerry    jerry        4096 Nov 27
16:17 test2
```

Finally, the applied tricks section: There is a hack that we can use give us a little (very little) security when we are sharing files. If we set permissions on a directory to rwx--x--x, we deny others the ability to list the directory (with ls or anything else), so there is no way that other users can find out the names of the files in that directory. With the x bit set, however, if people know the name of the file they can access it (much like a password). This gives us rudimentary passwords for files. For example:

```
unix> mkdir AUUGN
unix> chmod 711 AUUGN
unix> ls -ld AUUGN/
drwx--x--x   2 jerry    jerry        4096 Aug  9
16:14 AUUGN/
unix> echo "hello" > AUUGN/adlkfjasdlfkjasdlf
unix> chmod 644 AUUGN/adlkfjasdlfkjasdlf
unix> ls -l AUUGN
total 0
-rw-r--r--   1 jerry    jerry           6 Aug  9
16:14 AUUGN/adlkfjasdlfkjasdlf

unix> su otheruser
unix$ ls AUUGN
ls: AUUGN: Permission denied
unix$ cat AUUGN/adlkfjasdlfkjasdlf
hello
```

This shows that only those people who know that the file is called adlkfjasdlfkjasdlf can actually show the contents of the file. The only problems with this is 1) root can see all the files and 2) filenames can often be found from process information ie ps or even worse /proc if the file is mapped into the address space of a process.

I hope that this has given some insight to UNIX directory permissions,

'till next time.

AUUG Chapter Meetings and Contact Details

CITY	LOCATION	OTHER
BRISBANE	Inn on the Park 507 Coronation Drive Toowong	For further information, contact the QAUUG Executive Committee via email (qauug--exec@auug.org.au). The techno-logically deprived can contact Rick Stevenson on (07) 5578-8933. To subscribe to the QAUUG announcements mailing list, please send an e-mail message to: <majordomo@auug.org.au> containing the message "subscribe qauug <e-mail address>" in the e-mail body.
CANBERRA	Australian National University	
HOBART	University of Tasmania	
MELBOURNE	Various. For updated information See: http://www.vic.auug.org.au/auugvic/av_meetings.html	The meetings alternate between Technical presentations in the odd numbered months and purely social occasions in the even numbered months. Some attempt is made to fit other AUUG activities into the schedule with minimum disruption.
PERTH	The Victoria League 276 Onslow Road Shenton Park	Meeting commences at 6.15pm
SYDNEY	TBA	

For up-to-date details on meetings, including those in all other Australian cities, please check the AUUG website at http://www.auug.org.au or call the AUUG office on 1-800-625655.

AUUG 2001

Theme: "Always On and Everywhere"

The AUUG Annual Conference will be held in Sydney, Australia, 26, 27 and 28 September 2001.

The Conference will be preceded by three days of tutorials, to be held on 23, 24 and 25 September 2001.

The Programme Committee invites proposals for papers and tutorials relating to:

- Security in the Enterprise
- Applications made possible by Open Source
- Technical aspects of Computing.
- Networking in the Enterprise.
- Business Experience and Case Studies
- Open Source projects
- Business cases for Open Source
- Technical aspects of Unix, Linux, and BSD variants
- Open Systems or other operating systems
- Computer Security
- Performance Management and Measurement
- Networking, Internet (including the World Wide Web)

Presentations may be given as tutorials, technical papers, or management studies. Technical papers are designed for those who need in-depth knowledge, whereas management studies present case studies of real-life experiences in the conference's fields of interest.

A written paper, for inclusion in the conference proceedings must accompany all presentations.

Speakers may select one of two presentation formats:

Technical presentation:

- A 25-minute talk, with 5 minutes for questions.

Management presentation:

- A 20-25 minute talk, with 5-10 minutes for questions (i.e. a total 30 minutes).

Panel sessions will also be timetabled in the conference and speakers should indicate their willingness to participate, and may like to suggest panel topics.

Tutorials, which may be of either a technical or management orientation, provide a more thorough presentation, of either a half-day or full-day duration.

Representing the largest Technical Computing event held in Australia, this conference offers an unparalleled opportunity to present your ideas and experiences to an audience with a major influence on the direction of Computing in Australia.

Submission Guidelines:

Those proposing to submit papers should submit an extended abstract (1-3 pages) and a brief biography, and clearly indicate their preferred presentation format.

Those submitting tutorial proposals should submit an outline of the tutorial and a brief biography, and clearly indicate whether the tutorial is of half-day or full-day duration.

Speaker Incentives

Presenters of papers are afforded complimentary conference registration.

Tutorial presenters may select 25% of the profit of their session OR complimentary conference registration. Past experience suggests that a successful tutorial session of either duration can generate a reasonable return to the presenter.

Please note that with the GST changes to tax legislation we will be requiring the presentation of a tax invoice (which we will assist in producing) containing an ABN for your payment. If that is not provided then tax will have to be withheld from your payment.

Important Dates

Abstracts/Proposals Due	- 13 July 2001
Authors notified	- 27 July 2001
Final copy due	- 24 August 2001
Tutorials	- 23-25 September 2001
Conference	- 26-28 September 2001

Proposals should be sent to:

AUUG Inc.
PO Box 366
Kensington NSW 2033
AUSTRALIA

Email: **auug2001prog@auug.org.au**

Phone: 1800 625 655 or +61 2 8824 9511

Fax: +61 2 8824 9522

Membership Application

FRONT

Membership Application

BACK