AUUGN

The Journal of AUUG Inc.

Volume 21 • Number 1
March 2000

Features:

News:

Regulars:

ISSN 1035-7521

Print post approved by Australia Post - PP2391500002

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
Tattersall's
787 Dandenong Road
East Malvern VIC 3145

Vice-President:
Mark White
Mark.White@auug.org.au
Red Hat Asia-Pacific
Suite 141/45 Cribb Street
Milton QLD 4064

Secretary:
Stephen Boucher
Stephen.Boucher@auug.org.au
The Australian Industry Group
380 St. Kilda Road
Melbourne VIC 3004

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

Committee Members:
Malcolm Caldwell
Malcolm.Caldwell@auug.org.au
Northern Territory University
Casuarina Campus
Darwin NT 0909

Lucy Chubb
Lucy.Chubb@auug.org.au
Aurema Pty. Ltd.
79 Myrtle Street
Chippendale NSW 2008

Günther Feuereisen
Gunther.Feuereisen@auug.org.au
GFH Computer Services
PO Box 1350
Double Bay NSW 1360

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

Michael Paddon
Michael.Paddon@auug.org.au
Australian Business Access
33 Lincoln Square South
Carlton VIC 3053

AUUG Business Manager

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

Editorial

Günther Feuereisen
Gunther.Feuereisen@auug.org.au

Welcome to Y2K. I hope all of you who were involved in the general paranoia that gripped the world, got through unscathed. I spent my New Year watching things tick over – and for the first time at New Year's, I was glad to NOT see fireworks :-)

With the start of the year, we start to look to the Elections for the AUUG Management Committee.

The AUUG Management Committee (or AUUG Exec) is responsible for looking after your member interests. How AUUG serves you is Its' primary function. This includes such things as organising the Winter Conference, Symposia around the country, coordinating efforts with the Chapters, making sure we have enough money to do all these things, and importantly, making sure that we give you, our Members, the best possible value for your Membership dollar.

As such, this requires individuals willing to give up some of their free time, put their hand up, and say: "I want to help decide the future of AUUG."

If you feel that you want to be involved in driving AUUG into the future, then turn to page 12 for a nomination form. Deadline for nominations is 5pm, 14 April 2000.

See you next time.

günther

❖

Thanks to our Sponsors:

AUREMA

TELLURIAN

Contribution Deadlines for AUUGN in 2000

Volume 21 • Number 2 – June 2000:	**May 17th, 2000**
Volume 21 • Number 3 – August 2000:	**July 17th, 2000**
Volume 21 • Number 4 – November 2000:	**October 17th, 2000**
Volume 22 • Number 1 – March 2001:	**February 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:
Günther Feuereisen

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

> **bloat** *v.* **1** inflate, swell. **2** (as **bloated** *adj.*) inflated with pride, wealth or food. [Old Norse] - *The Pocket Oxford Dictionary*

I recently attended a Microsoft Executive Breakfast. I know, I know, but in the real world I manage the team that has to look after about 350 desktop PCs that are Windows based as well as a handful of NT servers.

The topic of this breakfast was Windows 2000, in particular the success that Microsoft has had rolling Windows 2000 out to all the desktops and servers in Redmond. And it should have been successful, if the video they showed is any indication of the level of resource thrown at it – at least 50 sys admins, about a dozen project managers, 24 x 7 access to the Windows 2000 development team (without the need to go through the Microsoft support phone line), a fully stocked "war room", radios, phones, the lot.

At the end of the seminar my Microsoft account manager told me that even though Windows 2000 was not officially released until February, that date was just for the plebs and he could get it for me earlier if I wanted to start an immediate roll out to all my desktops. I politely declined.

Of all the information passed on to me over the bacon and eggs that morning, only one piece really sticks in my mind – the minimum configuration for all desktop PCs in Microsoft is now a 200 MHz Pentium II CPU, with a 4 Gb hard disk and 128 Mb of memory. And all I could think was, "Time for another visit to Upgrade City!"

I think it is safe to say that Microsoft is the recognised master of bloat[1]. It strikes me as strange that since to run each new Operating System you need to buy a new computer anyway, why are they so hung up on backward compatibility?

However they are not the only guilty parties. I remember about 10 years ago when the first SPARCstation 2 arrived at work. It was so fast, it had a whole 32MB of RAM, and all us 1+ users were really envious. I still have a 32MB SPARCstation 2, and recently tried running Solaris 2.6 on it. Have my expectations of speed really changed that much over ten years, or does Solaris 2.6 place much more demand on the hardware than SunOS 4.1 did?

Why does bloat occur? Well, actually for good reasons. Given the rate of improvement of computer technology and the reduction in its price, as dictated by Moore's Law, it is not unreasonable to expect to have a larger faster system to deploy software on than was available 5 years ago. Add to that the customer's desire for more new features.

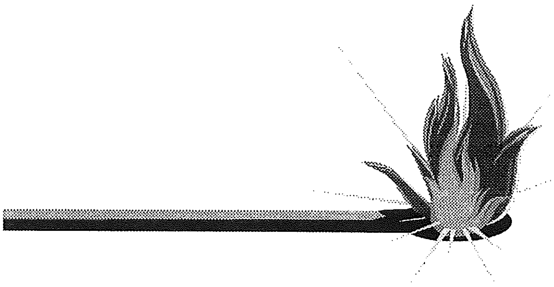
In addition, the computing companies today are all competing for the big end of town, and so they are gearing their operating systems to run popular software (e.g. SAP, PeopleSoft, Oracle) most efficiently on their big hardware. So they inflate their kernels with features and functions to aid that.

My point here, and I do have one, is that the developers of Open Source operating systems should take a lesson here. And I think the lesson is that they should not go down the same track for a while yet.

It is important to remember that one of the major applications of Linux and *BSD is extending the life of hardware that is too old to run the operating system they originally had installed. Take my SPARCstation 2 – I loaded OpenBSD on it and it is quite useable X Windows desktop.

I can see the attraction of adding more and more features to the kernel – if nothing else it means you get to list your name as a Linux developer. And I can see that it is very sexy to compete with the major vendors for who runs fastest on the biggest iron.

However, I would hope that the Open Source OS developers recognise that the majority of their users at this stage are at the low end, and are looking for an operating system that makes the most efficient use of their underpowered hardware.



Imminent Death of the Internet

Well, that's probably a bit strong, but I think there is a definite cue for Dr McCoy to come on the set and say, "It's the Internet, Jim, but not as we know it."

I have here an article from *The Age* of 16th December 1999 headlined, "How name-calling became a $360m Net bonanza." It covers the spectacular share market success of the float of Melbourne IT – the offshoot of Melbourne University that sells Internet domain names, primarily in .com.au.

[1] For the pedants out there: yes I do realise that I have given a definition of "bloat" that shows it is a verb and then proceeded to use it as a noun.

The float of Melbourne IT on the ASX is described as the second most successful in ASX history. The shares were issued at $2.20, and instantly jumped to $8.20. They peaked at $9.10 – a 313% profit. No wonder that in the photo that accompanies the story the staff all look so happy – each of them picked up a parcel of shares in an incentive scheme.

Melbourne IT charges $125 to register a .com.au name, and $110 to register in .com, .org or .net, these registrations being renewable every two years. It was forecast that Melbourne IT would have an income of $7.8 million from domain name sales in 1999. As *The Age* article states, "It's a licence to print money."

According to the article, Melbourne IT occupies a multi-million dollar King Street office replete with relaxation centre, espresso bar and prayer room.

So why do I see this as the death of the Internet as we know it? Because it is the clearest indication I have to date that the Internet is no longer driven by community, it is only driven by greed.

In the last few years some difficult decisions have been made about how the DNS name space is to be administered. The old fashioned way – a bunch of individuals and institutions who provided time for free in their spare time and equipment as part of their own Internet set up – could obviously not continue. The costs of equipment and connectivity were getting too great for the institutions to bear and the individuals could not devote the time to satisfy all the DNS requests. We had to set up organisations that could devote themselves to DNS maintenance.

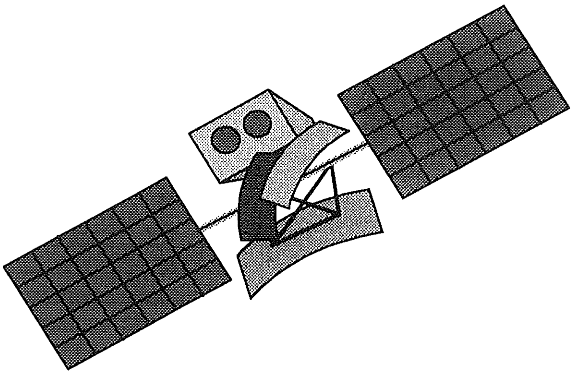
How did we, as an Internet community, allow this important infrastructure function go to a set of private companies? Surely we expect these companies to act only in the interest of their shareholders. Why did we not insist that this function go to non-profit entities with the aim of supporting and promoting an open Internet for the entire world? As Professor Gerrand himself is quoted as saying, "When I went to the University of Melbourne with the idea of setting up Melbourne IT and selling domain names, the vice-chancellor said our aim should be not to make a lot of money but rather to reflect the leadership qualities of the university."

I am ashamed to be living in a society where not only did Professor Gerrand ignore the vice-chancellor's advice, but *The Age* should congratulate him for doing so.

❖

Member Services Update:

New Online News Service



EXCLUSIVE ON-LINE NEWS SERVICE FOR AUUG MEMBERS.

AUUG in partnership with IDG Communications, the world's foremost IT publisher, are offering members a comprehensive UNIX information on-line news service. The free service is available daily or weekly depending on your preference and adds great value to your existing AUUG membership.

Run by the ComputerWorld division of IDG, the service will pull content from IDG's global resources of more than 280 publications and from all its local magazines and services.

The content is specifically tailored to your information needs. AUUG will be defining the areas of interest that the tailored email service will be based upon. As a subscriber, you will receive a daily or weekly email with headlines and summaries of the stories pointing back to the full text of the article on a dedicated web site. A fully searchable archive of news and feature stories will provide a comprehensive and continuously growing information resource for AUUG members.

The service has strict privacy policies and your email address will only be used for this specific news report. You can subscribe or unsubscribe at any time.

This service brings AUUG closer to our members and we are pleased to be able to offer this to you.

To join the service, send an email to `auugnews@email.idg.com.au` and include the phrase `join auugnews` in the body of the email.

If you would prefer to receive the news service via an AUUG maintained mailing list, then instead send the message `subscribe idgnews` to `majordomo@auug.org.au`.

❖

News from the AUUG Business Manager

Liz Carroll
busmgr@auug.org.au

Hi All,

Sydney Morning Herald

By now you are probably all aware that AUUG is running a fortnightly column in the Sydney Morning Herald. (Thank you to everyone who has contributed to date.) As such, we are always looking out for good articles. Should anyone wish to contribute please email me your article for submission. If you are unsure whether something is appropriate, an outline of the topic will be sufficient and I will let you know whether it will be okay. Following is a brief outline of the type of article the SMH is looking for:

"The Sydney Morning Herald is targeting its' IT section to business, running it after the business section and targeting readership at CEOs. The column is a UNIX column (rather than an AUUG column). SMH audience also runs 80-90% of their IT section in the Age in Melbourne. UNIX Column - 600-700 words with final credit to writer and pointer to AUUG Web site."

✧ ✧ ✧

Systems Magazine

In addition to the SMH, AUUG is also running articles in Systems Magazine (copies of which AUUG members are now receiving). As such, articles of approximately 1,700 words would also be appreciated.

Should anyone have any questions, please feel free to email me or call on 02 8824 9511. I look forward to reading your articles!

✧ ✧ ✧

Membership Cards

Many of you have asked about Membership Cards. Currently there has been a delay on these - they have been redesigned, and are currently being printed - will aim to be out by the end of March.

✧ ✧ ✧

Renewals

Don't forget to renew your AUUG Membership! Those with a December 1999 renewal date - it is now due!

For an update on your AUUG membership status, or any queries, please call:

1-800-625 655 or 02 8824 9511

or send email to:

`busmgr@auug.org.au`

Remember, if you change jobs, or move, please let us know!!

✧ ✧ ✧

AUUG2K Sponsorship Update

redhat.

I'm happy to announce that Redhat will be a major sponsor for this years conference: AUU2K - "Enterprise Security, Enterprise Linux".

❖

For the latest news on AUUG

Check out the AUUG website at:

`www.auug.org.au`

Linux Backup Offer to AUUG Members

Get a free licence for a great product whilst helping the Australian software industry...

Functional Software will shortly be launching its COSbackup product onto the Linux market. Ahead of the official launch, we are seeking feedback on the Linux version of the product from AUUG members. This feedback will provide us with valuable input into our product development plans.

In exchange for completing an evaluation questionnaire, we will provide each AUUG member with a permanent licence to use the software on Red Hat version 6.x - either for personal or commercial use.

FUNCTIONAL
SOFTWARE

The process will be:

- you register your interest
- we email you a copy of the software, documentation and a 30 day evaluation key
- you install and configure the product without any assistance from us
- you send us the evaluation questionnaire
- we send you a permanent licence key

Interested?
If so, please register by sending the following details to info@fs.com.au or call Daniel or PJ on 02 9869 0240:

Your name and email contact details
AUUG Membership Number
Company
Position
Host name (required for licence key)

About FS
Functional Software is an Australian company. We develop system management software for automating operations management functions in UNIX environments. Our integrated toolset technology, The Functional Toolset, and our COSmanager software applications are used worldwide by hundreds of high-end UNIX clients.

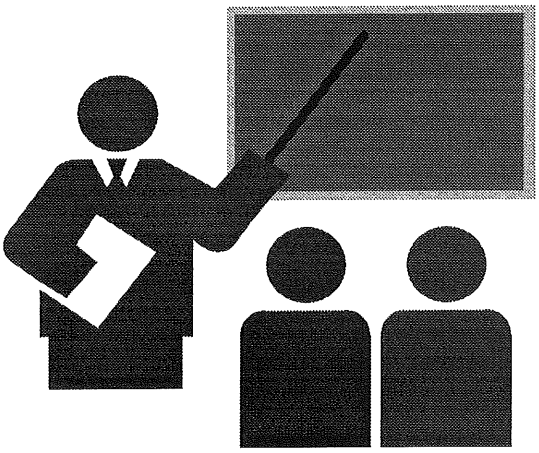
About COSbackup
COSbackup provides network-wide backup and recovery management for UNIX systems, including support for stackers, silos, Oracle databases and NT clients. It is one of the best commercial backup products available on the UNIX market.

You can check out Functional Software and COSbackup at www.fs.com.au.

FUNCTIONAL
SOFTWARE

Call for Papers AUUG2K

AUUG2K Conference
28-30 June 2000
Australian National University
Canberra ACT Australia



THEME: "ENTERPRISE SECURITY, ENTERPRISE LINUX"

The AUUG Annual Conference will be held at the Australian National University, Canberra, Australia, 28-30 June 2000.

The Conference will be preceded by three days of tutorials, to be held on 25-27 June 2000.

The program committee invites proposals for papers and tutorials relating to:

- Security in the Enterprise
- How to get Linux into your business or corporation.
- Applications made possible by Open Source.
- Technical aspects of Computing.
- Networking in the Enterprise.
- Business Experience and Case Studies

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.

IMPORTANT DATES

Abstracts/Proposal Due:
14 April 1999

Authors notified:
1 May 2000

Final copy due:
2 June 2000

Tutorials:
25-27 June 2000

Conference:
28-30 June 2000

Proposals should be sent to:

AUUG Inc.
PO Box 366
Kensington NSW 2033
AUSTRALIA

Email: auug2k@auug.org.au

Linux under Sail

D.F. Davey
daved@physiol.usyd.edu.au

The enjoyment of cruising, and its safety, is critically dependent on the weather. When doing coastal cruising, where heading into a safe haven is an option if the weather gets nasty, having access to good weather information is critical. Special marine weather forecasts and reports are broadcast on marine frequencies regularly, but there is nothing better when making important decisions than the weather charts produced by the Bureau of Meteorology. Sometimes these can be had on television, but there are two other important sources, the Bureau of Meteorology Web site, and the Naval HF Weatherfax transmissions. The former re-quires a computer, the second can be received with dedicated hardware, or more economically with a computer interface to an HF receiver. The Web solution only works when within range of a digital telephone network; the HF transmissions can be received anywhere.

Of course there are other Internet sources of information, including tides, contact with most marine equipment manufacturers, Notices to Mariners (more later) and more. And of course there is E-mail.

ECONOMICAL WEB TRANSACTIONS

That one can achieve an IP connection over a GSM phone does not mean one can afford to use it. The timed call charges are orders of magnitude higher than ISP charges, so keeping connections to a minimum duration is critical. This is where Linux comes into its own. (If you are interested in the hardware and/or software, see the Appendix.)

HTTP REQUESTS

Rather than use an interactive web browser to request files, I use wget[1], which can be called from a script. Parallel asynchronous requests turn out to be even more connect-time efficient, so a script like:

```
#! /bin/sh

URLS="url1 url2 url3 ..."

for i in $URLS
do
  wget $i &
done

wait
echo -e '\007$0 finished'
```

Of course the real thing is more complex, e.g. exploiting sed to change HTML image links from the original ones to ones pointing at local files, using time-stamped file names for wget output, and setting symbolic links to these files from generic names pointed to by a local index file that can be used with a web
browser off-line.

Some of this is easier said than done, for some web documents using Java and/or frames and/or image maps can be interesting challenges to get down to the particular URL of the desired specific piece of information sought. Naturally this is best done when a shore line is available. Using a browser's document information facility or scanning the HTML is sometimes needed. Trial runs with wget usually pay off. Using browsers that do not support frames or Java (e.g. Amaya) can help if the server responds appropriately to such browsers.

CONDITIONAL HTTP REQUESTS

An example of Web documents that are valuable on board are the *Notices to Mariners* published by the Australian Hydrographic Office. These advise of changes to nautical charts that need to be applied for safety. Published fortnightly and made available electronically[2] as PDF files they are extremely valuable to a careful navigator. Getting these efficiently means knowing what files are available compared to the ones you already have. Some issues are in the form of multiple files of text and images, so this is not done efficiently interactively. Again Linux comes into its own, with a simple mirror script using wget to obtain the current list of files, comparing this to the local list and requesting anything new. Incidentally xpdf[3] is a nice alternative to commercial PDF readers for viewing the files.

E-MAIL TRANSFERS

I often have 50 work-related E-mail messages in a day plus the ones I really want to read, so once again, efficiency is critical. Reading E-mail with a telnet session, or transferring single messages with sendmail or pop is not the way. Having access to a land-based Unix system provides a number of alternatives. The most efficient ones all involve dealing with all the E-mail off-line, and transferring incoming mail as a single file compressed with gzip, i.e. the land-based system compresses my entire mail file, moves it aside and passes it to the on-board system. Outgoing mail files in the queue directory are put into a compressed tar file before the IP connection is made, then passed to the land-based system which unpackages the file into its queue directory and triggers sendmail into action. For simplicity and to minimise connect times, this is done with sendfile, part of the TCPmsg[4] system. So in

[1] Available from the author's site ftp://gnjilux.cc.fer.hr/pub/unix/util/wget/wget.tar.gz or gnu sites like ftp://prep.ai.mit.edu/pub/gnu/

[2] http://www.hydro.navy.gov.au/prodserv/n2m/index.n2m

[3] http://www.foolabs.com/xpdf/

[4] ftp://ftp.cs.usyd.edu.au/TCPmsg/tcpmsg-1.1-tar.gz

principal the process is:

- `tar` and `gzip` the local `mqueue` directory and clear it so `sendmail` does nothing when the link is up
- dial and start the `cslip` connection process
- look for the remote host with `ping` packets
- `sendfile` the `tar`-file to the remote (where `cron` takes care of it in due course)
- log in to the remote and trigger the incoming mail transfer (scripted again for speed)
- wait for completion of the `sendfile` and `receiver` processes (easy because they both establish lock files)
- break the connection

This usually takes 1 to 1.5 minutes for a typical batch of mail that might keep me occupied for several hours dealing with it. The outgoing mail file contains a test message addressed to the remote machine with a time-stamped *Subject:* line, and a matching informative message is sent to the on-board machine. On the subsequent connection the test message comes back. Any failure of the system would become evident if this did not happen.

TIDAL INFORMATION

There are web sites that serve tidal information, but for all but the most critical requirements, software together with data files of tidal harmonic constants can deliver adequate tidal predictions for most sites and any time. The X based program `xtide`[5] is a very easy to use application that produces graphical or tabular output.

WEATHERFAX BY HF RADIO

Weather and sea state information are broadcast over HF radio frequencies from Naval transmitters in Canberra and Darwin. As many as 65 charts are broadcast daily using what is effectively old technology but one which suits one way and noisy communication lacking error correction. The system is one in which the source image is scanned at about 2 lines per second, and black and white in the source is encoded as a frequency shift in the radio signal. No timing information is contained in the signal, so the receiver must somehow sync to the source scan, which requires knowing the scan rate, though there is scan frequency information contained in the start of the transmission - effectively a black bar at the top of the image. Decoding this signal is in most ways trivial, though it is a real-time process which potentially Linux is not good at.

A simple interface can convert the modulated signal into a binary one that can be monitored with the modem control lines on an RS-232 interface, or with a parallel port. Because the scan frequency is so low, the real time requirement is not difficult, so it is possible to continuously sample the signal and the system clock (which has microsecond resolution). In fact "continuously" is far too fast, and judicious use of `usleep()` calls to create a pseudo regular sampling proved the best answer to avoid having to use MS-DOS software. So a simple X program displays the incoming chart a line at a time, and the real time problem is not really a problem at all.

One problem does remain however: the reception process is very slow and it takes 10 minutes to receive a chart (and of course if you miss a scheduled broadcast you just have to wait for the next, anywhere from 6 to 24h). But at least with Linux, while you wait the 10 minutes, you can be doing something else.

APPENDIX

Power

Power on all but the largest yachts is usually restricted to 12V DC. Sometimes people use 24V, and the odd boat has an inverter to produce mains voltage AC, but being able to work with 12V is handy. Fortunately much portable computer gear will operate on 12V DC perfectly well. That includes all the equipment mentioned here. On our yacht we use large lead-acid batteries charged with the engine alternator and with solar panels. On sunny days, the panels will drive the computer gear with power to spare.

Hardware

I use an AcerNote laptop which has a PCMCIA slot, parallel, serial and IrDA ports, 1.2Gbytes hard disk, 32Meg memory and a floppy drive. I have PCMCIA peripherals: a modem card for my Ericsson GSM mobile phone (which also works with the IrDA interface), an interface for a portable CD reader, and a SCSI card for a JAZ portable disk drive (unfortunately needing both 5 and 12V DC). A Canon BJC-80 portable BubbleJet printer works with the infrared port. Finally a weather fax interface to our HF radio receiver connects through the serial port.

Linux Version

I run a file system that traces back to something like Linux-0.81, and which I have progressively upgraded myself. Recent upgrades have been driven by the IrDA code which is under intensive development. The code now intended for Linux 2.2.15 is stable and reliable for my purposes.

System software

I run X with a 1280x1024 virtual screen which overcomes the smallish display on the AcerNote. Print spooling to the printer is with lpd using Ghostscript, so it looks like a PostScript printer. I use dip and CSLIP for networking because in my tests, CSLIP throughput is 10-30% better than ppp. Sendmail provides the primary mail handling. wvWare[6] deals with the irritating MS-Word attachments people insist on sending me.

[5] `http://www.flaterco.com/xtide/files.html`

[6] `http://www.wvWare.com`

Mark your diary!

AUUG2K

THEME "ENTERPRISE SECURITY; ENTERPRISE LINUX"

It's only 4 months away!

AUUG2K will be held in Canberra from
28th to 30th June, 2000.

Proudly sponsored by:

redhat®

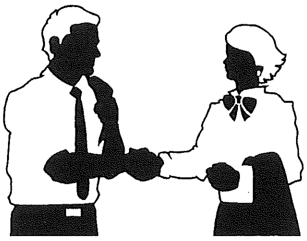
Where do you want AUUG to go tomorrow?

Help make AUUG the kind of organisation you want it to be – nominate for a position on the AUUG Management Committee! The call for nominations and a sample nomination form can be found on the next few pages. The nomination form should be returned to AUUG by the 14th of April.

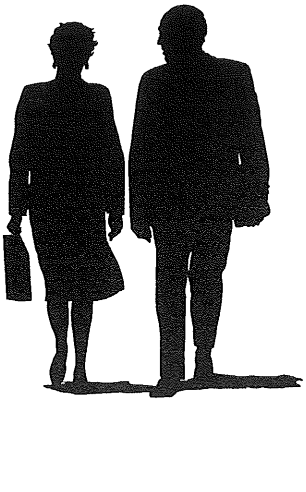
You need to be nominated by three voting members of AUUG (that is, either individual members or institutional members), and you must be an individual member yourself.

If you want to know more about serving on the Management Committee, e-mail the current committee at:

auugexec@auug.org.au



What? You can't find three members to nominate you?



Send in your nomination form anyway – we'll find someone to sign it!

And don't forget your 200 word policy statement!

AUUG Inc.
2000 Annual Elections
Nomination Form

We,

(1) Name: ______________________________ AUUG Member #:________ and

(2) Name: ______________________________ AUUG Member #:________ and

(3) Name: ______________________________ AUUG Member #:________

being current financial members of AUUG Inc. do hereby nominate:

____ for the following position(s):

(Strike out positions for which nomination is not desired. Each person may be elected to at most one position, and election shall be determined in the order shown on this nomination form.)

President

Vice President

Secretary

Treasurer

Ordinary Management Committee Member (5 positions)

Returning Officer

Assistant Returning Officer

Signed (1) ____________________________________ Date ________________

Signed (2) ____________________________________ Date ________________

Signed (3) ____________________________________ Date ________________

I, Name:____________________________________ AUUG Member #: ___________

do hereby consent to my nomination to the above position(s), and declare that I am currently a financial ordinary member of AUUG Inc.

Signed ____ Date _________________

AUUG Inc. Annual Elections Call for Nominations

Nominations are invited for the following positions within AUUG Incorporated:

- President
- Vice President
- Secretary
- Treasurer
- Ordinary Management Committee Member (5 positions)
- Returning Officer
- Assistant Returning Officer

Nominations must be made in writing and must be signed by the nominee and three (3) financial voting members of AUUG Incorporated, and must state which position(s) are sought by the nominee. The nominee must be a financial ordinary member of AUUG Incorporated, and can nominate for any or all of the above positions. While any ordinary member may be nominated to more than one position, no person may be elected to more than one position. Election to positions is determined in the order shown above.

A sample nomination form can be found on the previous page.

Nominees may include with their nomination a policy statement of up to 200 words. This word count will not include sections of the statement stating, in point form, the name of the nominee and positions held on, or by appointment of, the AUUG Management Committee or positions in AUUG Chapters.

Policy statements that exceed the word limit shall be truncated at the word limit when included in the ballot information.

Nominations must be received by the Secretary of AUUG Incorporated by the 14th of April 2000, and may be lodged by one of the following methods:

1. by post to:
   The Secretary
   AUUG Inc.
   PO Box 366
   Kensington, NSW, 2033
   the nomination must be received no later than April 16th and must be postmarked no later than 12 noon on April 14th 2000.
2. by hand to:
   The Secretary (Stephen Boucher)
   OR
   The AUUG Inc. Secretariat
   no later than 5pm on April 14th 2000.
3. by FAX to:
   The AUUG Inc. Secretariat
   on 02-8824-9522
   no later than 5pm on April 14th 2000.

Stephen Boucher
Secretary
AUUG Inc.

❖

AUUG Incorporated Election Procedures

These rules were approved by the AUUG Inc. Management Committee on 14/12/1994.

1. Notice Of Election

The Returning Officer shall cause notice of election to be sent by post to all financial members no later than March 15 each year.

2. Form Of Notice

The notice of election shall include:

(a) a list of all positions to be elected, namely:
President
Vice President
Secretary
Treasurer
Ordinary Committee Members (5)
Returning Officer
Assistant Returning Officer
(b) a nomination form;
(c) the date by which nominations must be received (in accordance with clause 21(2) of the Constitution, this date is 14 April);
(d) the means by which the nomination form may be lodged;

a description of the format for a policy statement.

4. Policy Statement

5.

A person nominated for election may include with the nomination a policy statement of up to 200 words. This word limit shall not include sections of the statement stating in point form the nominee's name, personal details and positions held on, or by appointment of, the AUUG Management Committee and chapters.

Policy statements exceeding the word limit shall be truncated at the word limit when included in the ballot information.

The Returning Officer may edit policy statements to improve readability, such edits being limited to spelling, punctuation and capitalisation corrections and spacing modifications.

Use of the UNIX wc program shall be accepted as an accurate way to count words.

4. Receipt Of Nominations

In accordance with clause 21(2) of the Constitution, nominations shall be received by the Secretary up until April 14. A nomination shall be deemed to have been received by the due date if one of the following is satisfied:

- it is delivered by post to AUUG Inc.'s Post Box, the AUUG Secretariat's Post Box or the AUUG Secretariat's street address no later than 2 business days after April 14 and is postmarked no later than 12 midday on April 14;
- it is delivered by hand to the Secretary or the AUUG Inc Secretariat no later than 5pm on April 14;
- it is transmitted by facsimile to the Secretary or the AUUG Inc Secretariat no later than 5pm on April 14.

5. Requirement For A Ballot And Due Date

In accordance with clause 21(5), no later than May 1, the Secretary

- shall advise the Returning Officer of all valid nominations received;
- and if a ballot is required, shall advise the Returning Officer of a date no later than May 15 for the ballot for all contested election.

In accordance with clause 42(3), the due date for return of ballots shall be 4 weeks after the date advised above.

6. Form Of Ballot Paper

The ballot paper shall contain:

- details of all positions for which the number of nominations exactly equals the number of positions to be filled;
- for each position for which a ballot is required, the names of all persons seeking election to that position, except those already elected to a higher position, with a square immediately to the left, for the elector to place a voting preference;
- instructions on how to complete the ballot paper;
- instructions on how to return the ballot paper;
- a brief description of how the ballot is to be counted.

The ballot paper shall not contain any identification of existing office-bearers.

The ballot paper shall be accompanied by a copy of all policy statements submitted by all persons nominated, including any persons elected unopposed. These policy statements may be truncated or modified as outlined in 3.

7. Method Of Voting

Voting for each position shall be by optional preferential vote. The number "1" must be placed against the candidate of the elector's first preference, and a number other than "1" against any or all of the other candidates. Preferences shall be determined by the numbers placed against other candidates, which must be strictly monotone ascending to count as preferences.

A vote shall be informal if:

- it does not have the number "1" against exactly one candidate.

8. Secrecy Of Ballot

The ballot paper shall be accompanied by two envelopes, which may be used by the elector to ensure secrecy. On completion of the ballot paper, the paper may be placed inside the smaller envelope. This envelope is then placed inside a second envelope. The elector must then sign and date the outer envelope, making the following declaration:

"I, ________________________________,
member number __________, declare that I am entitled to vote in this election on behalf of the voting member whose membership number is shown above, and no previous ballot has been cast on behalf of this voting member in this election."

9. Returning Ballot

To be considered to have been returned by the due date, the ballot paper together with declaration as above must be returned by one of the following means:

- it is delivered by post to AUUG Inc's Post Box, the AUUG Secretariat's Post Box or the AUUG Secretariat's street address no later than 2 business days after the due date and is postmarked no later than 12 midday on the due date;
- it is delivered by hand to the Returning Officer or the AUUG Inc Secretariat no later than 5 pm on the due date.

10. Method Of Counting

Where there is an election for a single position, the votes shall be counted by the preferential method. Where there is more than one position to be filled, the votes shall be counted by the modified preferential Hare Clark system described in Schedule 1.

11. Method Of Election

A person may be elected to only one position. Elections shall be counted in the order of positions described in 2(a). When counting ballots, any person previously elected shall be deemed withdrawn from that election, and all ballot papers shall be implicitly renumbered as though that person was not included.

12. Notification Of Result

In accordance with clause 42(7) of the Constitution, the Returning Officer shall advise the Secretary in writing of the result no later than fourteen days after the due date. The Returning Officer shall advise all candidates for election of the result no later than fourteen days after the due date. The Returning Officer shall advise the AUUGN Editor in writing of the result no later than fourteen days after the due date. The AUUGN Editor shall include the results in the first issue of AUUGN published after receiving the results from the Returning Officer.

13. Publication Of These Rules

The Returning Officer shall advise the AUUGN Editor of the current rules, and the AUUGN Editor shall cause the current rules to be published in the first issue of AUUGN published on or after 1 January each year. Where no issue of AUUGN has been posted by February 28 in any calendar year, the Returning Officer shall cause the current rules to be distributed with the notice of election.

14. Occasional Variation From These Rules

Subject to the Constitution, the Management Committee may authorise occasional variations from these rules. Such variations shall be advised in writing to all members at the next stage in the election process in which information is distributed to members.

15. Execution

Where these rules require the Returning Officer to carry out an action, it shall be valid for the Returning Officer to delegate execution to the Secretariat from time to time employed by the Management Committee.

16. Retention Of Ballot Papers

The Secretary shall retain that ballot papers and member declarations (as specified in 8) until the AUUG AGM of the calendar year following the year of the election, unless a general meeting of AUUG directs the Secretary to hold them for a longer period.

Schedule 1:

1) Each ballot paper shall initially have a value of one.
2) The value of each ballot paper shall be allotted to the candidate against whose name appears the lowest number on the paper among those candidates not elected or eliminated. If there is no such candidate (i.e. the ballot paper is exhausted) the ballot paper shall be set aside.
3) A quota shall be calculated by dividing the number of formal votes by one more than the number of positions remaining to be elected, and rounding up to the next whole number.
4) If any candidate is allotted a total value greater than the quota, that candidate shall be declared elected, and the ballot papers allotted to that candidate shall be assigned a new value by multiplying their previous value by the excess of the candidate's vote above the quota divided by the candidate's total vote. This new value shall be truncated (rounded down) to 5 decimal places. Ballot papers that subsequently have a value of zero shall be set aside. Steps 2 and 3 shall then be repeated.
5) If no candidate is allotted a total value greater than the quota, the candidate who is allotted the lowest total value among those candidates not elected or eliminated shall be eliminated. Steps 2 and 3 shall then be repeated.
6) Where:

(a) two or more candidates declared elected at the same stage of counting according to Step 4 have an equality of votes, and it is necessary to determine which is deemed elected first,

or

(b) a candidate is required to be eliminated under Step 5, and two or more candidates have an equally low vote,

the Returning Officer shall return to the immediately preceding stage of counting and

(i) in the case of candidates elected, deem first elected the candidate with the highest vote at the immediately preceding stage, and

(ii) in the case where a candidate is to be eliminated, eliminate the candidate with the lowest vote at the immediately preceding stage.

Where an equality of votes still exists at the immediately preceding stage, the Returning Officer shall continue proceeding to preceding stages until a result can be determined.

In the event that candidates have maintained an equality of votes throughout the entire counting process, the Returning Officer shall determine which candidate is to be determined first elected or to be eliminated by lot in the presence of the Assistant Returning Officer.

❖

JAVA Based TERMINAL EMULATOR

ANY
UNIX HOST
PLUS
HP3000
& NT

Javelin® Terminal Emulator
Providing precise emulation of HP2392A, HP700/9x and VT320 terminals. Javelin is a powerful connectivity tool that enables you to make host connections via web browsers.
Support for Netscape and Internet Explorer. Features the industries smallest applet size. Micrsoft Interent Explorer Cab file 90,000 and Netscape Jar file 110,000 bytes.

NO LOCAL TERMINAL EMULATION NEEDED!

Our latest Javelin review:

"Integration, A look at Javelin" by Jim Alton *(Jim Alton writes for Interact and the NewsWire bulletin)*
"Minisoft has done an excellent job with Javelin. It's compact, clean and quick. If your environment provides the capability to ensure and control Web Browsers installation, then Javelin may give you an excellent alternative to traditional PC based terminal emulation"

Test drive JAVELIN TODAY

Contact Pathway Pacific
☎(02) 9956 7699 📠(02) 9929 8703
sales@pathway.com.au
Unit 8/281 Pacific Highway
North Sydney NSW 2060

Review FULL product details @ www.minisoft.com

Aegis and Distributed Development

Peter Miller
peterm@lucent.com

[Editor's Note: this was originally presented at AUUG'99. Our thanks to the author for allowing the reprint of this article in AUUGN.]

ABSTRACT

A significant hurdle for Open Source projects is getting the base source code to all the participants, and then getting each contributor's changes to the source code to everyone else. How do you manage this?

Source management is not enough if you are participating in an Open Source project. Software projects are more than the aggregation of their source files. A tool which supports the software development process for large teams is required, particularly if you are performing a team activity such as an Open Source project over the Internet.

Aegis supports large teams and large projects. Aegis is designed around change sets. Aegis is designed around repository security (availability, integrity and confidentiality). Aegis' distributed development uses this existing mature functionality to keep two or more repositories synchronised.

Aegis supports multiple repositories, multiple lines of development, multiple distributed copies of repositories, disconnected operation, and is security conscious.

Aegis is licensed under the GNU GPL.

Aegis is mature software. It is 8 years old. It has users all around the world. It is actively being maintained and enhanced.

Aegis is easy to use. It is big, it does have a lot of functionality, but the essential process can be learned in less than a day.

Aegis is available from

```
http://www.canb.auug.org.au/
      ~millerp/aegis/
```

Please download it, plus one of the template projects, to get a feel for the environment. If you would like more information, there is also a Reference Manual and User Guide available from the same place.

ROAD MAP

Aegis is not a ubiquitous system, and so it is not assumed that readers are familiar with Aegis or the facilities it provides or its terminology. This paper wants to talk about a recent extension to Aegis which supports distributed development of software by providing facilities to keep two or more software repositories synchronised.

In order to do this, the following material will be covered:

- The fundamental aspects of software development are examined, with the intention of establishing a common terminology.
- The concept of a transaction is introduced, extending the idea of a change set with which many readers will be familiar.
- A very brief overview of Aegis is given, mapping the concepts in the first two sections onto concrete commands, so the reader has some feel (however brief) for the Aegis way of doing things.
- The facilities use to replicate repositories will be described, and some examples of the many different ways of using them will be given.

SOURCE MANAGEMENT IS NOT SUFFICIENT

In looking for a better way to manage the distributed projects' sources, it is necessary to look beyond the obvious and perennial file bashing, to see if there could be a larger picture.

In writing software, there is one basic underlying activity, repeated again and again:

edit, build, test, check, commit

Different textbooks and tools will call the various steps different things, like

edit, make, Unit Test, Peer Review, check-in

and for single-person projects, some of these steps are so abbreviated as to be almost invisible, especially when you simply jump in and edit the files in the master source directly.

And the activities are rarely so pure, usually there are iterations and backtracking, which also serves to obscure the underlying commonality of software development. The review step, in particular, often moves around a great deal.

For the maintainer of an Internet project, the activities are remarkably similar:

edit
: apply an incoming patch, build it (also serves to make sure it is consistent with itself and the rest of the project),

test
make sure it works (does the thing right),

review
make sure it is appropriate (does the right thing),

commit
yes, I'll accept this.

The term "source management" carries with it a focus on the source files, but the activities outlined above only talk about files indirectly! Source management alone is not enough.

Tools like RCS and SCCS concentrate exclusively on single files. CVS also concentrates on files, but only at a slightly higher level.

Change Sets

One of the most obvious things about the software development process outlined above, is that it is about sets of files. You almost always edit several files to fix a bug or add a new feature, you then build them to stitch them together into the project, you test them as a set, if there is a review they will be reviewed as a set, and you commit them together.

A project makes progress by applying a series of these change sets, so tracking them is the only way to re-create self-consistent prior versions of the project.

Software developers, however, frequently work on several changes at once. Figuring out where one change sets stops and another starts requires a modicum of discipline. The fuzziness of the boundaries often serves to obscure the underlying presence of change sets.

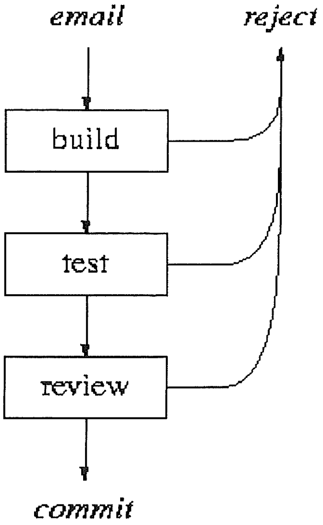
But are change sets enough? Change sets are, after all, a way of aggregating the right versions of sets of files, and the software development process above only mentions change sets indirectly.

For many developers, even those working in large companies and in large teams, change sets are the best tool they have. They work, day in and day out, with change sets. And they get the job done.

But take a look, for a moment, at what the project maintainer does:

- if the patch doesn't apply cleanly, don't accept it
- if the patch doesn't build, don't accept it
- if the patch doesn't test OK, don't accept it
- if the patch doesn't look right, don't accept it
- else commit

Stepping back a bit, you will notice that these apply equally in work within a software house.



How often have we all seen stuff which was allowed to skip one of the validations, only to get yanked and re-fixed later?

The next step in improving the development process is automating the tracking of these steps, to make sure each one has been done. Some tools merely beep at you if you skip a step, others make the validations mandatory before a commit may occur. Mandatory things usually get developers riled up, and prevent introduction of the tools.

But these validations are done for a purpose: they are there to catch stuff-ups before they reach the repository. They exist to defend the quality of the product. They are not arbitrary rules, they are just checking that we are doing the things we say we are doing already.

The pay-back for such a tool is to detect such process blunders before they introduce defects into the project. Fixing them before they are committed is less effort than fixing them after they are released (if we are to believe cumulative experience and the numerous studies).

Let's look at the maintainers role again for a moment. Those first 3 steps (patch, build, test) can be automated. I would not suggest for a moment that the commit should be unconditional! Thus, the 4th step, the code review, is the essential work of the maintainer. The pay-back of this is also clear - less mindless tedium for the maintainer.

Team Work Factors

Most folks are not convinced by any of this. It's just a crock. They can do it perfectly well manually. They have been doing it manually for a decade or more - with more flexibility, too!

Working in a team comes with a number of costs. The most obvious cost is that you need to manage the interactions between the developers. It becomes rapidly obvious that they can't all just leap into the source tree and edit on the files directly, because pretty much instantly nothing compiles for anyone. And the change sets are obfuscated beyond redemption.

That's what work areas are for - they've been re-invented thousands of times, and have been called zillions of different names (e.g. sand boxes), but they all do the same thing: Each developer gets their own work area, and they leave the master source alone. They do all their work there, and only when they are ready to commit do files get modified in the master source.

Notice the strong correlation between work areas and change sets? Different tools make this correlation weaker or stronger, depending on what they are trying to achieve. The basic concept, however, is that change sets have meaning even after the files are committed, whereas a work area is where change sets are created and reproduced.

A tool which seeks to do more than just manage files, or even change sets, needs to address work areas, too. This is particularly true when one of the validations (build, test or review) fail. You don't want the master source polluted.

Work areas are only half the story though. Teams almost immediately lead to the next problem: file conflicts. No matter how you implement file locking, at some point you have to merge the competing edits. Different tools do this at different points in the software development process, but they all do it.

The tool needs to track file versions in work areas, so you know if the file is up-to-date (if someone has committed a competing edit ahead of you). This isn't a big problem, because change sets must record file versions anyway. If the file isn't up-to-date, you need a 3-way merge to bring it up-to-date (and you have the 3 versions - the one copied, the one in the work area, and the one most recently committed). Most tools prevent commit from occurring if the file needs to be merged. (You could prevent build and test, too, but that's a bit too officious - there are often good reasons for working with outdated sources.)

TRANSACTIONS

Time to divert for a moment into another discipline. There are large parallels between what change sets are, and how they are managed, and transactions, such as database transactions.

• When a large database system implements transactions, they span multiple tables, and they quarantine transactions until you commit them. You can abort them without modifying the tables at all, or having to remember to go back and fix things.	• A software configuration management system which supports change sets and individual work areas is doing the same thing.
• Database systems frequently perform input validation, to ensure that the data being entered is correct, or at least self consistent. Different systems provide different amounts of support for this.	• Compiling programs and running tests fulfil a similar role in software configuration management systems. Different systems provide different amounts of support for this.
• Database transactions are all-or-nothing actions. You either get all of the transaction applied to the tables, or none of it. Half a transaction is considered to be a corrupted database.	• Software configuration management systems can work the same way. Again, half a transaction frequently means a project that won't compile, or wont function correctly.
• Database systems frequently maintain calculated indexes, derived from the information in the tables. In addition, views are derived from the database contents, and some systems cache them for performance.	• Think of compiling and linking programs as derived views of the source code. Cached for performance (you don't want to compile each time you execute, after all). There are other views of the code, including code metrics and generating documentation by extracting comments.
• The database tables are always present and populated, and available for querying and browsing.	• A simple *version management* system doesn't have this. But a full-blown configuration management system does. The working version of the program is always available for use.
• Databases are pre-occupied with locking and avoiding deadlocks.	• Developers are pre-occupied with locking and avoiding deadlocks.
• Database transactions are not commutative. It is true that some transactions do not interact at all, and within a given time window, these will be commutative. However, in the general case, you	• Source code transactions are not commutative. *Some* will not interact, but real world software projects find out *very* fast that the order of changes is important.

cannot assume all database transactions are commutative all of the time.	
• Database systems maintain audit trails and roll-forward (or roll-back) journals. These are essential in determining where data went, and reconstructing the system in the event of a failure.	• SCM systems maintain audit trails and roll-forward (or roll-back) journals. These are essential in determining what changed and when, and reconstructing the system to reproduce bugs.
• Database systems are designed for multiple simultaneous users, for concurrent transactions, for security (meaning: integrity, availability and confidentiality).	• Configuration management systems need to support multiple simultaneous developers, concurrent development, and security.

Now we have a simple way of defining that configuration word in SCM: think of a configuration as a database snapshot immediately after a transaction commit.

Software Configuration Management

"Nuh, uh. No way! I've tried BarfCase and it always crashed and went far too slowly and harassed me. Not going there!"

This is a common reaction to tools which attempt to do more than baby-sit files. On the whole, it's a very reasonable reaction, considering what some of them do to you and your system.

However, SCM is the correct term (in the textbooks, anyway) for looking after the process and not just the files. To look after more you need to actually track the progress of change sets as they work their way through the process. Some tools are very invasive about this, and some are more subtle.

There are things the SCM tool needs to know to do its job:

- when a change set is created (this often implies the creation of a work area)
- when a file is added to a change set, so the version can be recorded (this often implies a copy into the work area)
- when files are created or deleted or renamed as part of a change set.
- the results of building the change set (either for warnings or errors, if a commit is tried against a failed build).
- the results of testing the change set (either for warnings or errors, if a commit is tried against a failed test).
- the results of a review of the change sets (either for warnings or errors, if a commit is tried against a failed review).
- when the change set is committed or abandoned (i.e. when it is finished)

None of these things are new. All of us are doing all of them already. Sometimes, some of the steps are pretty short, but they are all there.

Distributed Development

Once you have change sets, you have the basics of distributed development. You can use their information about files and file versions to package them up and sling them across the net. But more: once you have transactions, you have something worth slinging across the net: they are complete, they build, they test, they work.

To state this around the other way: if you don't have change sets, distributed development will be difficult. How do you know what to send (in a push mode), and how do they know what to fetch (in a pull model)?

But what do you do when you are the recipient of a change set? There is no way you are going to apply the damn thing to your repository sight unseen. You are going to check it all the ways you can: you will build it, you will test it, you will review it, and maybe decide to commit it. You need the process encapsulated by the transaction concept: isolate it, verify it, and verify it again.

Even when you are working alone, when you are the only user on a single PC, participation in a distributed development project is a team activity, and you need an SCM tool which is designed for working in teams. Source management alone is not enough.

Aegis

Aegis is a software configuration management system. It does all of the above and more besides, but it delegates as much as possible, so as to give you access to the other development tools you need...

Aegis delegates as much as possible, which is why it is more than just a version control tool. Some of the delegations are shown below:

redhat.®
Asia-Pacific

Training Courses for Australia

Be recognised as a guru in just 5 days – add a feather to your hat with Red Hat's Certified Engineer Course.

Open-source software is revolutionising the technology industry. Red Hat is the world's #1 distributor of Linux – the world's fastest-growing OS. With 15 million users and climbing, Linux is gaining momentum and mindshare at an unprecedented rate.

Join the revolution and fast-track your career with our 5-day RH300 Red Hat Certified Engineer (RHCE) course. If you're a UNIX or Linux-experienced user, a networking specialist, a systems integrator, or a system administrator, this is the course for you. This intensive training programme will reward you with a formal Red Hat Certification for the skills you have gained. The course comprises of four days extensive hands-on followed by 1-day certification exam. If you think you have the right stuff you can even just enquire about our exam-only option.

You'll be challenged by hands-on labs, exercises and daily quizzes. You'll learn all about Red Hat Linux on your own workstation, and will be provided with materials, study aids, handouts and a pre-assessment questionnaire. A catered lunch will be provided every day, and you'll look sharp in your own Red Hat Cap.

And if you're an AUUG member, you'll automatically receive a $200 discount.

What you'll learn:

How to install and configure Red Hat Linux, plus an understanding of hardware limitations. You'll be able to configure basic networking and file systems, the X Window System and basic security, along with setting up common network (IP) services and carrying out basic diagnostics and troubleshooting. Plus you'll be able to perform essential Red Hat Linux system administration.

What others will recognise in you

You'll be recognised anyway in the world as having passed one of the toughest certification courses around. You'll be recognised as having the skills to get the job done. Join the ranks of the many Australians who have already been certified.

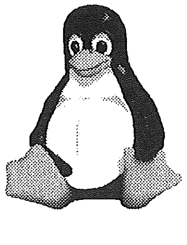
Prerequisites for RH300

Other Red Hat Training Courses or equivalent experience with UNIX, LAN/WAN fundamentals, Internetworking with TCP/IP, knowledge or experience setting up NFS, HTTP, DNS, FTP, NIS, DHCP and other networking services, along with security. For a complete list of prerequisites, see the Prerequisites for RH300.

And it doesn't hurt to have a desire to be associated with some of the brightest minds in the world. And penguins.

How to Register

Check out the form on the next page for dates and locations that suit you, fill in the form and fax/post to us today to kick off the next stage in your career.



redhat.
Asia-Pacific

Australian Training Registration Form – Special AUUG Offer

To register for a training course, please print and complete this form, and **fax** it to **(07) 3257 4800**. Send original form and/or purchase order to: **Red Hat Asia-Pacific, Suite 141, 45 Cribb St. Milton QLD 4064**
Registration in a class is only complete when we receive a signed company purchase order, credit card reference or other direct payment. Make cheques payable to "Red Hat Asia-Pacific"

Attendee Details

Name: ______________________ Title: ____________________

Company: ____

Address: ____

______________________________________ Postcode: ____________

Telephone: ____________________ Fax: ____________________

E-Mail: ____

AUUG Membership # (required for discount, otherwise add $200): ____________

□ Please include me on your mailing list for future support, training, product and service offerings

Select Preferred Course & Date – All courses are RHCE (RHCE)

□ **Brisbane** April 10 – April 14	□ **Brisbane** May 15 – May 19	
□ **Sydney** March 27 – March 31	□ **Sydney** May 8 – May 12	□ **Sydney** May 29 – June 2
□ **Canberra** May 1 – May 5	□ **Canberra** May 22 – May 26	
□ **Melbourne** April 3 – April 7	□ **Melbourne** May 1 – May 5	□ **Melbourne** May 29 – June 2
□ **Adelaide** April 10 – April 14	□ **Adelaide** May 22 – May 26	
□ **Perth** March 27 – March 31	□ **Perth** May 29 – June 2	

*Courses will be held in a near-CBD location in each city (TBA), if more than one attendee please write in number otherwise tick box.

Payment Information – RHCE (RH300) AUD$3195 (normally $3395)

□ I attach an official company purchase order □ I attach a cheque for payment in full

□ Please charge my credit card (details below)

Card Type: □ Visa □ Mastercard □ Bankcard □ American Express □ Diners Club

Card Number: ______________________________ Expiry: ____/____

Cardholder Name: ______________________________________

I confirm that I have read and understood course prerequisite details posted at http://www.redhat.com/products/training_prereq.html . Red Hat Asia-Pacific reserves the right to cancel or reschedule courses. By supplying my AUUG membership number I understand that I will receive a $200 discount off of the standard course price of $3,395.00.

Authorised Signature: ______________________________________

Name: ______________________________ Date: __________________

Build

The build step is watched, but what it does, and what tool you use to do it, is up to you. Yes, you can use make.

Aegis has support for heterogeneous development, and it keeps track of which architectures you have built and tested on, and which are yet to be done.

Merge

File merges are watched, but what it does, and what tool you use to do it, is up to you. Aegis tracks when files in a transaction need merging, to make sure do don't commit out-of-date files.

Cross branch merges are also supported, in a natural way.

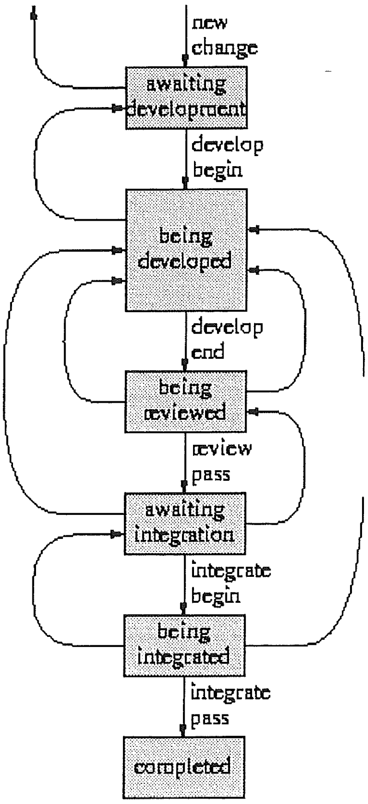
Test

The test step is watched, but what it does, and what tool you use to do it, is up to you. It's also optional. Aegis remembers all tests, and they are available for developers to run in the future. Aegis can also suggest tests which may be appropriate, using historical information about source files and tests in transactions in the past.

Review

The review step is watched, but what it does, and what tool you use to do it, is up to you.

The code review is a mandatory portion of the process. What ever style you want (singe peer, review board, etc) are all supported.



Version Control

The commit step is watched, but what it does, and what tool you use to do it, is up to you. Yes, you can use RCS. Yes, you can use SCCS.

Aegis isn't just another version control system. Version control is like foundations in a house: essential, but you rarely see them.

Aegis transactions, simply called *changes*, pass through a number of states, and different commands apply in different states. Some of them are shown on the state transition diagram on the left. Most of the action happens, and most of the commands are used, in *the being developed state.*

Most of Aegis' commands perform functions developers are already intimately familiar with, and others with obvious purpose in a process like the one described above. Some of them are described here:

aenc	New Change	This commands is used to create a transaction.
aedb	Develop Begin	This command is used to create its work area.
aecp	CoPy files	This command is analogous to RCS "co", used to copy files into the transaction, and remember the version.
aeb	Build	used to run the build tool of your choice, and wait for the exit status.
aed	Difference	This command is used to see the differences between the baseline and the transaction.

aet	test	This command is used to run tests on the change.
aede	Develop End	This command is used to say the transaction is ready for review. (It doesn't commit, yet.)
aerpass	Review Pass	This command is used to say a transaction has passed review.

The commands are different (e.g. aeb *vs* make, aecp *vs* co) but the activities are familiar. Aegis is easy to use - believe it or not, you've just seen all of the routine commands necessary for a developer to submit a change (there are only a couple more routine commands for transaction integrators, and they are often automated).

A brief example, so you can get a feel for it. Say we have simple program, rather typical of programming examples. This one has a bug...

% hello Hello, Wuld! %	Spot the bug.
% aenc -edit -p example.4.2	Create a new change (*aenc*), and begin development

Thrown into your favourite editor aegis: project "example.4.2": change 10 created % aedb 10 %	(*ae*db). This creates the work area for you.
% aecd aegis: cd ~pmiller/example.4.2.C010 %	Aegis remembers where all of the work areas are. You may work on two or more changes at once; each gets its own work area.
% aecp hello.c aegis: file "main.c" copied % vi hello.c *Fix the bug.* %	You browse the baseline to figure out what you need to copy in order to fix the bug. In this simple example, this isn't hard.
% aeb aegis: make -f *baseline/Makefile* *gcc -o hello hello.c* %	Aegis watches the result if the build. You can use whatever tool you like (there are *some* constraints). You can do partial builds (just name files on the aeb command line) but they don't count as exit criteria.
% aent aegis: new test "test/00/t0001a.sh" created % vi test/00/t0001a.sh *Write a shell script to validate output.* %	You create tests within Aegis, so it knows they are tests and not something else, and so it can remember them right along side the source files they test. Just *what* you test is up to you: unit tests, integration tests, system tests, whatever; if you can script it, Aegis can run it.
% aet aegis: sh test/00/t0001a.sh aegis: test "test/00/t0001a.sh" passed %	Positive tests establish that the bug is fixed. Aegis remembers the exit status.
% aet -baseline aegis: sh test/00/t0001a.sh aegis: test "test/00/t0001a.sh" failed (as it should) %	Negative tests establish that the bug *was reproduced.* (You can't just write "exit 0" and hope nobody checks.) Aegis remembers the exit status.
% aed aegis: fcomp -w -s *baseline*/hello.c hello.c hello.c,D aegis: fcomp -w -s *baseline*/test/00/t0001a.sh test/00/t0001a.sh test/00/t0001a.sh,D %	The difference files are available for the reviewers, so they can know they are seeing all of the differences, not just the ones they can find. (The fcomp tool is like a whole-file *diff -u*, but you can use whatever you prefer.)
% aede aegis: project: "example.4.2": change 10: development complete %	This command checks things: • It checks that you have been working on up-to-date copies of the files. • It checks that you have built the change. • It checks that you have tested the change, forwards and backwards. • It checks that you have differenced the change. • It checks that you haven't changed any of the files since the relevant command was run.
These commands are performed by a different staff member, a person on the reviewer list... % aecd example.4.2 10 aegis: cd ~pmiller/example.4.2.C010 % aedless %	The reviewer goes to the work area, and looks over your work. They *know* things about your change: • it is against the latest files • it doesn't conflict with any other change being reviewed • it builds cleanly • it tests forwards and backwards **Note:** the reviewer list and the developer list need not intersect. In addition, you can prevent a developer from reviewing his own work.

`% aerfail -edit example.4.2 10`
`thrown into text editor for reason`
`aegis: project "example.4.2": change 10: returned to developer for further work.`
`%`
	Failing a review is simple. You need to supply a comment as to why, and Aegis remembers it. It isn't the reviewers job to fix things: they have no write permission, only *veto*.

`% aerpass example.4.2 10`
`%`
	Passing a review is simple.

These commands are performed by a different staff member, a person on the integrator list...
`% aeib example.4.2 10`
`% aeb && aet && aet -bl && aed`
`%`
	The integration process parallels the development process. This is automatable, or adds a second review state.

`% aeifail -edit`
`thrown into text editor for reason`
`aegis: project "example.4.2": change 10: returned to developer for further work.`
`%`
	Failing an integration is simple. You need to supply a comment as to why, and Aegis remembers it. It isn't the integrators job to fix things: they have no write permission, only *veto*.

`% aeipass`
`aegis: ci hello.c`
`%`
	Passing an integration is simple. This is the point at which Aegis updates the history.
	Note: Even now, the integrator has *no* write permission on the repository. Only Aegis does.

It's that simple. Of course, there is more - much more. But the essentials of the process really are this simple.

Aegis and Distributed Development

One more command... The `aedist` command is used to package change sets for sending, and unpackage them on receipt. Yes, this is the part of the paper you have been waiting for.

The essential activity of keeping two databases in sync is to forward transactions. This was the inspiration for the Aegis distributed development work. Aegis does the same thing. For example, you can clone a change locally using the following command:

```
aedist -send | aedist -receive
```

This isn't especially efficient (there is an `aeclone` command which does this faster) but it gives you the general idea.

• You always send an entire change, there is no way to send half a change. You usually send a **completed** change, though it is possible to send a work-in-progress change (it will be labelled as such).
• Sending the change to the other developers in a open source project can be done automatically, using and *integrate pass* hook. There are other ways, too.
	`% aedist -send -change=`*`n`*` \`
	`    | mail example-dev`
	`%`

• Each incoming change is placed into its own work area, just like any other Aegis change.
• The change is then automatically unpacked and built and tested. The usual Aegis commands you would use to do this manually are used; there is no magic involved, no special permissions given to the `aedist` command (*i.e.* no security decrease).
	`% show | aedist -receive`
	`aedist: aegis -new-change`
	`aedist: aegis -dev-begin`
	`aedist: aegis -build`
	`aedist: aegis -test`
	`aedist: aegis -dev-end`
	`%`

• The `aedist -rec` command checks for *trojan horses*. If certain files are changed (you can nominate more) the change will not be built or tested, so that you can inspect it *before* build and test run any commands.	*(In this case, you have to `aeb` && `aet` && `aede` manually, after inspecting carefully.)*

• The `aedist` command only takes the change to be *being reviewed* state. It is up to a human to decide if the change is acceptable.	`% aerpass`
	`%`

• Commit is **not** unconditional	`% aerfail -edit`
	`give reason`
	`%`

As you can see, there is nothing magical here. All of the Aegis process, already in place to protect projects from less-than-perfect changes, works very well in validating incoming random changes in a distributed development project.

The design of Aegis, and aedist in particular, emphasize security. Computer security in the sense of availability, integrity and confidentiality. The usual Aegis process makes sure that changes don't damage the repositories' integrity and availability (i.e. don't break it).

In addition, the aedist -rec command checks for potential trojan horses, by not even building or testing changes which could carry trojan horses. You can nominate more files, as appropriate for your project. It errs on the side of caution, rather than let a random shell script loose on your system.

Distribution by EMail

It is possible to specify a command to be run at the end of the aeipass command. This could be used to automatically distribute changes to other developers (or development groups) in the project. In this case, you would use something like:

```
aedist -send -change n |
mail example-dev
```

which will take change set n and mail it somewhere. Sending every change to every remote site may be a little too much traffic, and there are simple methods of performing the send at the end of a branch, instead, and the remote sites would thus receive fewer (but larger) changes.

When receiving this kind of email, you apply it using a command (I use MH in this example) similar to

```
show | aedist -receive
```

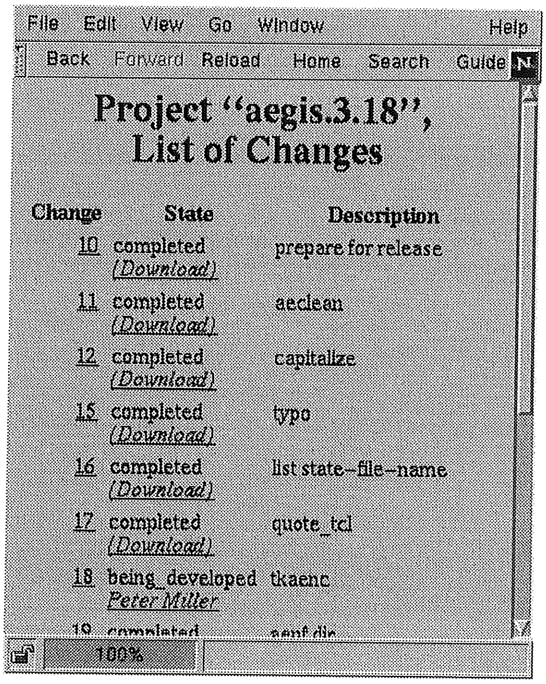
The change set will unpacked into a separate work area, be built, and be tested (if tests enabled). If the change set has no problems, it will then stop and wait for review.

Because it stops at a sensible place, you can automate all of this with, say, `procmail` or even a `sendmail` alias. This means that the burden of receiving every single change is no too great, if you have a need to keep two repositories closely synchronised.

Just as with any other form of email, you always have the option of using PGP to authenticate the send of the email. This is probably essential if you are going to automate the receipt of changes. By only accepting changes from known and authenticated senders, the risks of trojans decreases. Even when you use PGP, a human review is still necessary before the change is committed to the repository.

More importantly, commercial enterprises can keep the crown jewels secret by using PGP's encryption capabilities, and still use the Internet to connect their various campuses.

Distribution by Web



Project "aegis.3.18", List of Changes

Change	State	Description
10	completed (Download)	prepare for release
11	completed (Download)	aeclean
12	completed (Download)	capitalize
15	completed (Download)	typo
16	completed (Download)	list state-file-name
17	completed (Download)	quote_tcl
18	being_developed Peter Miller	tkaenc
19	completed	[illegible]

When distributing changes via the Web, you need to have a CGI script invoke the `aedist -send` command. An example of how to do this is distributed with Aegis, and you can see an example of it to the left.

It is also possible (in the page above the one shown) to obtain an entire branch as an `aedist` change, and apply it in a single step.

Even though changes are not necessarily commutative, you may still pick-and-choose. Because they are isolated into their own work areas on the receiving end, if they don't work, you will find out pretty soon, and without breaking your repository.

It is possible to set a *application/aegis-change-set* `mailcap` entry for your browser to invoke `aedist -receive`. This means you can unpack and apply in a single step, without saving to a temporary file first.

Other Methods...

There are other distribution methods, including sneaker-net and FTP.

One of the unexpected side-effects of adding the distributed development functionality to Aegis was the ability to package fully-configured example projects. This addressed a longstanding issue with Aegis: how do you get something so big started? The example projects weren't a design goal, but they are definitely a benefit.

This also provides a fast and effective way of getting a fully configured project image to a new contributor or remote site. After which, you can push email or they can pull downloads, or whatever.

The `aedist` command can also be used for other purposes, such as making complete archival copies or backups of the repository.

Where to from Here?

Can Aegis do the job? I believe that it can, but you should not take my word for it! Download a copy and start playing. Get a feel for it. You can get Aegis from:

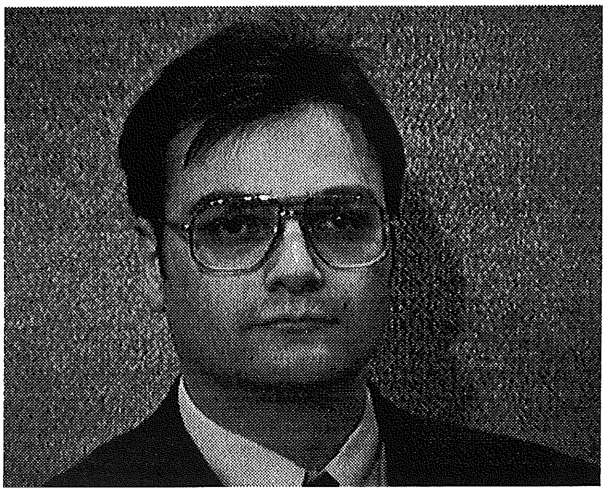
```
        http://www.canb.auug.org.au
              /~millerp/aegis/
```

If you would like to read some manuals, there are PostScript copies of the User Guide and Reference Manual available for download from the same place.

Once you have Aegis installed, download one of the template projects, available from the same place. These template projects get you up and running very quickly. (They also exercise the distributed development functionality to do so: your first taste.)

❖

Meet the Exec



DAVID NEWALL

Do you remember when Ken Thompson presented a keynote paper at an AUUG winter conference, and it wasn't about Unix? I don't remember him even mentioning the name. He spoke totally about not Unix. His talk was so not Unix that it had him driving his car down the middle of a river! It had him driving through the side of an airplane hangar. Or at least, that's what the map that his car mounted GPS receiver showed. That's what he spoke about: the GPS connected to the laptop in his car. Only Ken Thompson could get away with talking about not Unix at a Unix conference.

I think the man came from another time, another age, one in which computers were new and marvellous. A time when programs weren't things to be hidden from the world, doled out in exchange for large licence fees and signed non-disclosure agreements. Programs and ideas were shared amongst your community, and the highest compliment that could be paid was when people loved what you did.

I am lucky because I come from that time.

When I was still in high school the education department in South Australia had a world leading computing program. In typical Aussie style, a small suburban high school, in an economically depressed area, had a marvellous facility housed in a wooden shack. The outside was like any other school "transportable"; but inside was an air-conditioned room containing not one, but two IBM 1130 computers. One was fully expanded. It had 24K words of core memory! A whole 24K! The other was just a baby, having only 8K. School students from all over South Australia could use those computers. They sent in their programs on cards, that they marked with a texta or a 2B lead pencil. Printouts would come back after about a week and, as you might imagine, this gave you time to think about your program before you tried to run it.

A group of students used to attend this facility, once a month on Sunday afternoons. The Director of the Center, and his operations staff, generously donated their time, opened the place and supervised us. Some of the local Hewlett-Packard office staff used to bring HP computers for us to play with, too. We were predominantly a bunch of high school students, who met to play with the computers, to learn, and to share our knowledge and our enthusiasm. If you haven't experienced computing on the batch cycle, then you probably can't imagine exciting it would be to be able stand by the machine, feed in your cards and have the output within minutes, not days. And of course, there were two computers. Timesharing was a big thing for us ... but I don't mean that we all sat down in a room full of terminals, I mean that while one machine was reading in cards, processing and printing, the other was available for us to use interactively. We would all hover around the console, which was just like an IBM Selectric typewriter (funny that) and we'd all share the experience of whoever was lucky enough to be in the driving seat. Timesharing? We got 20 minutes at a go. "Save and sign off," was the cry that signalled the end of our time, and woe betide anyone who was slow to sign off!

Even better than the meetings once a month, were the annual summer schools in computing. People used to come from all over the world to attend these events. We had students and teachers from schools in South Australia, and from all of the other states, and from New Zealand and other parts of Asia, and from US and Canada, too! We had great people lecturing. I remember a researcher from IBM's Watson Labs taught us how to calculate the value of Pi to any arbitrary precision and, even though I didn't understand the maths I thought it was pretty cool to have such distinguished people come all the way to little Adelaide, to teach computing to us secondary students.

My first language was APL. My second was BASIC, but I didn't like the interpreter. My third language was BASIC, for which I wrote an interpretor in APL. It worked. This is not what you might expect from such young children, yet for us it was not unusual.

Put children with a love for computing in an environment where they can explore, encourage them to learn and to share, and you have a recipe for success that benefits the whole country. That's a lesson that, sadly, our leaders didn't learn. A "manager", scuttled the place while the Director of the Centre was in recovering from a life-threatening illness. It was a poor decision. Lucky for me, I had already got my start.

❖

The 2000 annual John Lions' student award for work in the area of open systems.

The John Lions award has been instituted to recognise the leading role that John Lions played in bringing UNIX to Australia, the formation of AUUG, and the promotion of the values held by the open systems community.

After Ken Thompson and Dennis Ritchie published a paper "The Unix Time-Sharing System" in May 1974, John Lions decided to base his Operating Systems course around understanding the source code. In addition to that, he founded AUUG as a group of computer scientists who had a common interest in the UNIX Operating System.

Today AUUG has members throughout Australia from industry, commerce, and education and works to promote the benefits of open architectures and standards compliance in languages, operating systems, networks, and applications. AUUG focuses on the latest developments in open systems by the exchange of ideas and solutions through local chapters, the annual conference, local chapter conferences, and its journal.

Requirements:

- The award is for a full time student at an Australian University.
- The award is for an in-progress or recently completed honours or postgraduate thesis in the area of UNIX and open systems. The judges will be looking for things like interesting uses of open systems technology, contribution to understanding of open systems, programs, tools or knowledge about UNIX and open systems.
- The award is judged on the basis of an approximately one page or 500-word description of the work. The evaluation committee may wish to interview students on the short list for the prize and possibly see a demonstration of the work so far completed.
- The evaluation committee will consist of at least 3 AUUG members, at least one of whom belongs to the AUUG national executive, and optionally a representative from another organisation.
- The decision of the evaluation committee is final and the committee reserves the right to not award the prize if a suitable entry has not been submitted.

Final date for receipt of entries is 5pm Friday 28th July 2000

The prize consists of:

- A cash prize of $1000
- One year's membership of AUUG
- Announcement of the prize at the main AUUG conference and in AUUGN (the AUUG Journal)
- A certificate
- The winner's name inscribed on a permanent awards board, displayed in the AUUG office and at the main conference

What sort of work might qualify?

The work will be focussed on software which relates to computer communications, networks, operating systems, or similar. If you are not sure whether your work may qualify, mail:

Lions_Award@auug.org.au

Entries may be submitted by email to **Lions_Award@auug.org.au** or by post to:

John Lions Student Award
AUUG
PO Box 366
KENSINGTON NSW 2033

My Home Network

Frank Crawford
frank@crawford.emu.id.au

A new column, a new year, a new millennium (or maybe not) all rolled up into one. I hope your Y2K issues were all as simple as mine, I spent New Year's Eve sitting watching computers not do anything unusual. On the other hand Y2K issues weren't non-existent, I have a list of about half a dozen things that did break later. Nothing major, just annoying. Interestingly, I haven't noticed any within my home network, which doesn't mean none exist, just they aren't obvious.

One thing that I know that doesn't have any Y2K issues, and one of the biggest advantages of a home network, is Squid, my web caching proxy. Squid is the world's most widely used web caches but works just as well in a small network, as in a large ISP. A cache is one of the most important items in any network, and these days, is essential in a home network given the high reliance on the web for most access, after all, isn't the whole of the net just the web? :-)

Squid is simple to install, reasonable simple to configure and a snap to use. In my network, I'm currently running Squid 2.2.STABLE4, which is the RPM distributed with RedHat 6.1. The latest stable version is 2.3.STABLE1, while there is further development going on in the 2.3 stream.

So, why is a cache an essential item within a home network. Well, stepping back, the function of a web cache is to provide temporary storage for any web traffic between the original web site and the end-users web browser. Most web browsers provide some form of local disk cache on the machine they are running on. What Squid provides is a shared cache between all users on your network. Consider what happens if all your machines have the same home page, rather than having it downloaded from a remote site like "ninemsn.com.au", once for each machine, it is downloaded only once and then shared between all machines in your house.

Even better than just the saving of multiple downloads, is the difference in network speeds between a modem and an Ethernet network around your house. As I've previously mentioned, I have a 56K modem, while I have a 10Mb Ethernet network around the house. There is a lot more behind it than including checks to see the page is hasn't been superseded, a non-cacheable page, CGI scripts, etc, but you quickly get the idea.

What I have found far more interesting is what uses HTTP for transport these days, and hence is cacheable in your Web cache. For example, the Microsoft Update Site is all web based and the downloads are prime targets for caching. It is really thrilling to update one host, which may take 20-30 minutes, and have the next machine perform an update in 20-30 seconds! Even better, FTP transfers through browsers are also cacheable.

So what needs to be done to set this up on your network. Well, the first thing is to obtain and install it on a server on your network. Obviously, this server needs to be available when you are using the web, otherwise you will find that you have no access rather than fast access. However, the server doesn't need to be the same as your modem server, although it is usually more efficient if it is (of course you can be sure that your cache will be available when your modem is up).

One resource that you will need is disk space, because that is where the system stores the copies of the pages. The recommended minimum is 100Mb, although the bigger the better, up to a point. In my case I have 200Mb, of which I'm only using about 40% at present. Aside from disk space, Squid also loves lots of RAM, as for performance it saves lots in RAM while it is running, again in my case 12Mb at present, just for Squid.

Once you have a suitable system, you need to configure Squid for optimum use. Now there are two possible ways to set up Squid, firstly, for it to access any web site that you require, and secondly for it to feed through a second cache, say at your ISP. Because of the tremendous savings that web caching gives you, most ISP would like all their clients to use the ISP's cache. In fact, some of them require you to use it. This means that your cache must go through the ISP's cache.

When you have worked out how you are going to configure your Squid, then you can get down to the job of setting it up. I will not go through all the details, if you need to look something up, go to the Squid web site, http://squid.nlanr.net/. As well, the default configuration file includes a lot of explanations about variables.

I now will go through some of the changes I have for my Squid. The configuration file for Squid is called (unsurprisingly) "squid.conf", and in the case of the standard RedHat distribution is in the directory "/etc/squid".

The first thing I set was the address of any other caches I was peering with. This is done with the directive "cache_peer", which gives the name and port, along with other flags. For example, for a connection to OzEmail (which has a proxy on port 8080) I have the line:

```
cache_peer proxy.ozemail.com.au parent \
8080 3130 no-query default
```

You can specify multiple peers, but in a home situation, you would only have a single one, and it would be a parent (i.e. it feeds you).

The other important thing to tune is sizes, both of the objects cached, and space to use. Squid will only cache objects that are below a set size, any item bigger than that is passed through without caching. This size is set by the "maximum_object_size" parameter. The default is 4Mb, although I have increased it to 16Mb in my installation.

The other size option to configure is how much disk space to use in the online cache. You can configure multiple disk cache areas, although again, home networks probably only need a single cache. This is set with the "cache_dir" which sets the size and configuration parameters for the disk cache. I'm currently using:

```
cache_dir /var/cache 150 16 256
```

which specifies a disk cache size of 150Mb, with 16 top level directories and 256 second level directories. The 150Mb is smaller than the disk size of 200Mb, and is due to a problem with older versions of Squid. Because squid needs to react quickly to each request, the deletion of old cache items is done as a separate thread. Unfortunately, in some versions, this deletion thread runs at a lower priority than the caching process, and hence you can run short of disk space in the short term if you have a large number of lookups. This will correct itself over the longer time period, but is still a problem. Two other parameters, "cache_swap_low" and "cache_swap_high", which are the low and high water marks for the LRU algorithm. These are normally set to 90% and 95% respectively.

The final item that needs some though is access control lists, or ACL, which are essential to configuring Squid in many ways. For example, it is necessary to define new ACLs to force Squid to only work though parent caches. The default is to try for remote web sites directly, and then go to any other caches defined. This is controlled by the "always_direct" and "never_direct" options. What I have defined is:

```
acl local-servers dstdomain
crawford.emu.id.au
always_direct allow local-servers
never_direct allow all
```

This defines an acl, which is my local web servers (or at least anything that ends in "crawford.emu.id.au"), allows direct access by Squid to these local servers, and does not allow direct access to anything else.

While there are many more options that can be set, the ones I've described should give you a start. I would suggest that you read all the configuration file, and go from there.

Okay, now that squid is configured, it should be started when the system is booted. Most distributions will include something to handle this, so I will not expand on it here. However, reconfiguration of Squid on the run is a bit different to traditional daemons. Rather than sending the daemon a SIGHUP signal (i.e. "killall -1 squid") you reinvoke squid with the argument "reconfigure", i.e. "squid reconfigure". This will communicate with the running daemon and reload any changes.

Finally, you will need to add the proxy to all your web browsers. There are a number of methods for doing this, in fact, Internet Explorer 5 implements an automatic method for finding a server, based on DNS and/or DHCP entries. Unfortunately, this is probably an overkill for most home networks.

The simplest way I've found is just to configure the proxy into each browser as a manual proxy, who's hostname is your server name and port is 3128 (the default Squid proxy port).

There is one interesting side-effect that I haven't mentioned so far. Aside from caching, Squid logs all activity, so these logs can be used to generate statistics, debug problems, or astound your family by your knowledge of what they have been viewing. In fact my children are aware that I do occasionally check on what they have been viewing, and hence tend to self regulate the browsing. They also have been known to come and ask me to track down a site they were viewing, but have forgotten the URL, so it does work both ways.

For those even more paranoid, you can get additional packages which link with Squid to parse each URL on the fly and rewrite it, if necessary. This can be used to block unacceptable sites, or redirect well-known sites to a closer mirror.

So, given the advantages of it, if you're not using Squid now, why are you causing yourself so much trouble, and wasting so much money?

On a different note, AUUG's annual conference is earlier this year, it will be held late June (see details elsewhere in AUUGN). I certainly hope to see you there, but more importantly, I hope to see papers about things that are of interest to readers of this column. Even better, why not write a paper yourself? The theme of the conference is "Enterprise Linux, Enterprise Security", but I'm sure room can be found for papers on general Linux or security issues.

Finally, in line with the conference theme, my next column will be on security issues and secure networking for home networks. Until then have fun, and let me know what you are doing.

❖

Sponsorship Opportunities:

AUUG2K

AUUG2K Conference
28-30 June 2000
Australian National University
Canberra ACT Australia

THEME: "ENTERPRISE SECURITY, ENTERPRISE LINUX"

The AUUG Annual Conference will be held at the Australian National University, Canberra, Australia, 28-30 June 2000.

The Conference will be preceded by three days of tutorials, to be held on 25-27 June 2000.

Information on papers can be found at the AUUG website :

www.auug.org.au

There are also many advertising and sponsorship opportunities available for your Organisations. Information is available from the AUUG business manager.

To take advantage of these, please contact:

Liz Carroll
AUUG Business Manager
Phone: 02-9858-4542
Email: busmgr@auug.org.au

❖

DIAMOND SPONSORSHIP

A$10,000

Includes:

- 2 complimentary registrations for the Conference
- 2 complimentary invitations for the Cocktail Reception
- 2 complimentary invitations for the Conference Dinner
- logo displayed in conference Plenary Hall
- acknowledged on all appropriate occasions in both print and verbally
- small display area

Listed and identified as a sponsor in:

- the conference brochure
- the conference final programme with company description

CHOICE OF:

- **Conference Brochure**
  - ➢ wide distribution to key decision makers
  - ➢ areas of exclusive advertising
  - ➢ immediate impact prior to the conference
- **Welcome Reception**
  - ➢ prestigious event allowing sponsor to make first impression on the delegates
  - ➢ reception identified as being sponsored by the XYZ company on all printed material
  - ➢ signage on the evening
  - ➢ opportunity to address delegates
- **Conference Dinner**
  - ➢ dinner identified as being sponsored by the XYZ company
  - ➢ name printed on dinner menu
  - ➢ opportunity to distribute mementos and address to the audience
  - ➢ banner identifying the sponsoring company
  - ➢ name of sponsoring company on entry tickets

✧ ✧ ✧

PLATINUM SPONSORSHIP

A$7,500

Includes:

- 1 complimentary registration for the conference
- 2 complimentary invitations for the cocktail reception
- 2 complimentary invitations for the conference dinner
- logo displayed in conference plenary hall
- acknowledged on all appropriate occasions in both print and verbally
- display space

Listed and identified as a sponsor in:

- the conference brochure
- the conference final programme

CHOICE OF:

- **Conference Proceedings**
  - ➢ 2 A4 pages of exclusive advertising
  - ➢ long term usage and shelf life as it is a reference material
- **Tee-Shirts**

- offering long term usage and company message to recipient

- **Conference Satchel**
  - Satchel offering long term usage and company message to recipient

✧ ✧ ✧

GOLD SPONSORSHIP

A$5,000

Includes:

- 1 complimentary invitation for the cocktail reception
- 1 complimentary invitation for the conference dinner
- logo displayed in conference plenary hall
- acknowledged on all appropriate occasions in both print and verbally
- display space available for one day at the conference

Listed and identified as a sponsor in:

- the conference brochure
- the conference final programme

CHOICE OF:

- **Keynote Sessions**
  - opportunity to introduce the keynote session

- **Lapel Badges**

- **Speakers Reception**
  - event allowing sponsor to make first impression with speakers
  - reception identified as being sponsored by the XYZ company on all printed material
  - signage on the evening
  - opportunity to address speakers

✧ ✧ ✧

SILVER SPONSORSHIP

A$2,500

Includes:

- 1 complimentary invitation for the cocktail reception
- logo displayed in conference plenary hall
- acknowledged on all appropriate occasions in both print and verbally
- rack space for promotional material

Listed and identified as a sponsor in:

- the conference brochure
- the conference final programme

CHOICE OF:

- **Advertisement in conference programme**
  - reaching an audience of key decision makers

- **Conference folder insert**
  - individual inserts in conference satchels

- **Advertisement in conference proceedings**
  - A4 size

✧ ✧ ✧

ADDITIONAL OPPORTUNITIES

- **Audio Visual**

- **Conference Network**

Further information is available from the AUUG Business Manager, Liz Carroll.

Contact details:

email:	busmgr@auug.org.au
Phone:	02-8824-9511 1800-625-655

✧ ✧ ✧

ADVERTISING

Various advertising opportunities are available from $1,500, including inserts on the brochure racks, handing out promotional material etc.

Further information is available from the AUUG Business Manager, Liz Carroll.

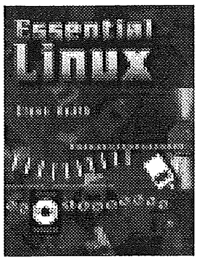
Contact details:

email:	busmgr@auug.org.au
Phone:	02-9858-4542 1800-625-655

new from digital press

an imprint of Butterworth-Heinemann

dp



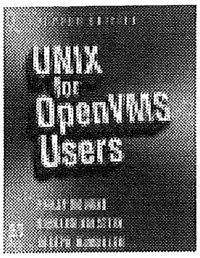
ESSENTIAL LINUX *with CD Rom included* Steve Heath

Essential Linux addresses and provides the essential information for getting a Linux system up and running, looking after it, and using it. It includes many screen shots and examples of essential commands and utilities. This book focuses on practical installations, such as converting IBM PCs, transferring data from Windows and MS-DOS and running Linux in conjunction with Windows. CD Rom included.

Contents: Inside Linux * Installing Linux * Linux commands * bash shell scripts * tcsh shell scripts * Editors * System Administration * Linux Networking * Installing XFree86 * If the command does not work..

ISBN 1 55558 177 3 : 250pp : PB : 1997 : ARRP ~~$80.00~~ : *MEMBER PRICE $64.00*

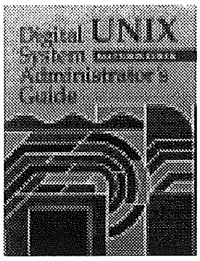
UNIX FOR OPEN VMS USERS 2ed Richard Holstein, Joseph McMullen & Philip Bourne



This book is for users who are making the transition from OpenVMS to UNIX and provides a comprehensive comparison of commands and utilities. Working into practical examples, the book shows simple daily tasks that map one-for-one from OpenVMS to UNIX.

Contents include: Introductory File Management * Editing * Communicating with Other Readers * Monitoring & Utilizing System Resources * Devices, Queues, & Background Processing * Processor-to-Processor Communications

ISBN 1 55558 1552 : 400pp : PB : 1999 : ARRP ~~$80.00~~ *MEMBER PRICE $64.00*



DIGITAL UNIX SYSTEM ADMINISTRATOR'S GUIDE Matthew Cheek

Digital UNIX System Administrator's Guide is the first book to address Digital UNIX system administration from an experienced administrator's point of view. Every topic covered is specific to Digital UNIX, serving as a valuable reference guide for experienced UNIX system administrators who are new to Digital UNIX.

Contents: Introduction * Installation * System Configuration * User Accounts and Security * Services and Resources * Networking* Performance, Tuning and Recovery * Troubleshooting * Appendices

ISBN 1 55558 1994 : 350pp : PB : 1998 : ARRP ~~$79.95~~ : *MEMBER PRICE $63.95*

TRU64 UNIX FILE SYSTEMS ADMINISTRATION HANDBOOK Steve Hancock

"Tru64 UNIX" is the new name Compaq has given the Digital UNIX operating system. This unique and authoritative book helps systems administrators and other technical professionals understand and master perhaps the most critical part of Tru64 UNIX - the file system by which the operating system stores and manipulates all of the information that enables both it and other applications to function. *Tru64 UNIX File Systems Administration Handbook* covers all of the newest and most advanced Tru64 UNIX features, including Compaq's TruCluster technology, which enables organizations to grow their systems by integrating many individual computers.

Contents include: Storage and Device Management * UNIX File Systems Architecture * Logical Storage Manager * UNIX File System * Advanced File System * Network File System * Appendices * Glossary

ISBN 1 55558 227 3 : 352pp : PB : December 1999 : ARRP ~~$75.00~~ : *MEMBER PRICE $60.00*

AUUG1

Special Offer for AUUG Members! Receive 20% discount on this order form.

ORDER FORM: Please supply the following

Qty	ISBN	Title	Retail Price	Member Price
___	1 55558 177 3	Essential Linux	@ ~~$80.00~~	**$64.00**
___	1 55558 155 2	UNIX for Open VMS User's	@ ~~$80.00~~	**$64.00**
___	1 55558 199 4	Digital UNIX System Admin Guide	@ ~~$79.95~~	**$63.95**
___	1 55558 227 3	Tru64 UNIX	@ ~~$75.00~~	**$60.00**

Please add $10.00 Postage and Handling to order total

PAYMENT OPTIONS:

Please find attached ❐ cheque for $ .... OR ❐ charge my:

❐ Bankcard ❐ Visa ❐ Mastercard ❐ AMEX ❐ Diners

Card No: _____/_____/_____/ _____ Expiry: ___/___

Name on card: .... Signature ....

NB: All credit card payments will be processed immediately on receipt of order

CUSTOMER DETAILS:

Name: ....

Address: ....

Suburb/Town: .... **State:** .... **P/Code** ....

Tel: .... **Fax:** ....

Email: ....

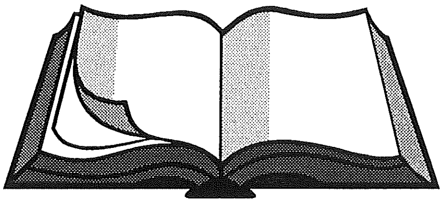
BUTTERWORTH HEINEMANN AUSTRALIA

To order, please return this form to

Freepost: Butterworth-Heinemann Australia
Reply Paid 1160, PO Box 251, Port Melbourne Vic 3207
Telephone: 03 9245 7188 **Fax**: 03 9245 7577
Email: bhau.marketing@reededucation.com.au

Book Reviews

Sub-editor:
Mark Neely
mpn@infolution.com.au



DEBIAN GNU/LINUX GUIDE TO INSTALLATION AND USAGE

Ossama Othman
New Riders Publishing; August 1999
ISBN: 0735709149

Reviewer:
Craig Macbride
craig@bjs.com.au

The aims of this book are pretty obvious by the title. The audience is intended to include those with no knowledge of Linux or Unix-like systems.

While most AUUGN readers would know about Unix systems already, this could be both an advantage and a disadvantage in the case of this book.

It is a disadvantage since about two thirds of the book is devoted to Linux/Unix fundamentals, but an advantage at those times when the author forgets his target audience.

Which his does, although you may or may not notice it, depending on your familiarity with Unix.

When I read the book, I didn't notice some of the assumptions. However, I handed it to someone who has used many non-Unix systems, and she picked some flaws almost immediately.

For example, she had never heard of NIS. Indeed, plenty of Unix systems aren't running NIS either. However, NIS is mentioned at one point, but not explained or even defined.

The failure to acknowledge that the readers may not know some of the technical terms used is probably the worst flaw.

One other aspect that grates with me is the haphazard nature of the index. NIS, for example, is mentioned in the text but not in the index!

I found a number of topics like that which, when mentioned in passing, didn't make it into the index. Nothing to do with hardware makes it into the index - not even "Intel".

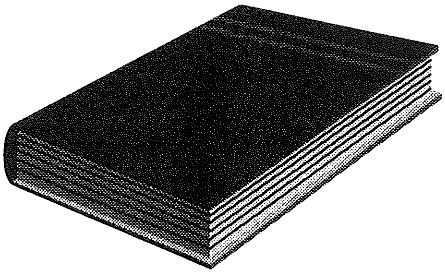
If you are reading a book cover to cover, it may be acceptable to have a small index. However, if a book is supposed to be used as a reference, not being able to find terms which you know you saw at some point in the book, or know should have been mentioned at some point becomes incredibly annoying.

Isn't Linux supposed to run on DEC Alphas? How about Amigas? PowerPCs? SGIs? There is no mention of Linux being run on non-Intel-based machines in this book, nor is there a comment to say that the authors are deliberately ignoring other architectures.

The only other criticism I have of this somewhat chatty book is that there are some lazy pieces of editing. Page 5 starts by saying "As we mentioned in section 1.1, the design of Debian GNU/Linux comes from the Unix operating system." Unfortunately, they did not mention that at all in section 1.1.

This book looks quite good as a brief installation guide for Debian Linux, on Intel (or clone) hardware. Unfortunately, it is next to useless if you want to use Linux on any other platform and it doesn't discuss this fact anywhere.

Despite the claim to the contrary, some prior Unix knowledge would be useful when reading this guide. However, those who know Unix already will find the larger section concerning usage of the operating system is worthless.



INTEGRATING UNIX AND NT TECHNOLOGY: THE DEFINITIVE GUIDE

Dulaney, E., Sankar, V., and Sankar, S.
29th Street Press, 1999
361pp.
ISBN: 1882419847

Reviewer:
David M. Williams
dave@qed.newcastle.edu.au

In this Internet age of heterogeneous networks, I'm pleased to report that this book looks to be a worthwhile resource for modern network administrators.

It is a modern title, too, with coverage of Windows 2000 and Active Directory - in addition to name resolution, printing, network browsing and the like.

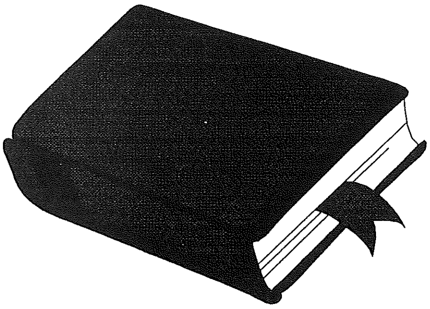
The book begins with an overview of 'the Unix world' and 'the Windows NT world' as well as a table comparing many features of the two different environments. I feel this table is balanced and covers many deep, technical aspects of the two technologies. This happily sets the pace for the rest of the book, where one can confidently see that the authors do have a strong understanding of both environments.

That said, though, I do think there was a stronger Windows NT emphasis. At times, it seemed as though the book may have been written for the hapless NT administrator who has been lumbered with an existing Unix system that they must cooperate with. Screenshots abound of NT tools and screens and performance graphs. There are chapters devoted specifically to NT technologies - like Active Directory. Also, there is a chapter on routing in NT but no corresponding coverage of routing in Unix.

I do not necessarily feel this detracts from the book – but potential buyers should perhaps be aware that there is a stronger emphasis on the NT side of the network.

I particularly felt, though, that the chapters on Samba and printing were very helpful to Unix administrators and gave strong Unix coverage.

All in all, I do recommend this book to network administrators who have Unix and NT systems in place, which must cooperate. It is not a book about basic system administration, nor is it a guide to either operating system. What you get is an up-to-date and hassle-free guide to sharing resources among these two diverse environments.



UNIX NETWORK PROGRAMMING, VOL. 2., INTERPROCESS COMMUNICATIONS, 2D. ED.

Stevens, W. Richard
Prentice Hall, 1999
558pp.
ISBN: 0130810819

Reviewer:
David M. Williams
dave@qed.newcastle.edu.au

This book should be on the shelf - or desk - of any person involved with programming in Unix.

W. Richard Stevens is a Unix expert, best known for his titles on socket programming. In fact, this volume is part of a series on Unix network programming with the first volume being sockets and XTI. The back cover advertises a forthcoming third volume on applications. However, Stevens died at the end of 1999.

Any large Unix programming will require well-implemented interprocess communications (IPC). This large work provides a comprehensive guide to every form of IPC, such as message passing, synchronisation, shared memory, pipes, locking and remote procedure calls (RPC).

There is a wealth of source code provided. The examples are not trivial or contrived; rather, the reader gets involved with creating real implementations of message queues and semaphores and the like. I do feel confident that this title is a definitive resource on Unix IPC programming, and that a depth of understanding will be gained that may not be available elsewhere.

There are questions posed, with solutions available at the back of the book, for those who might like to test themselves (or use the book in some teaching fashion).

The only disappointment I have is that there will not be another W. Richard Stevens book to follow.

❖

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!

AMERICAN
BOOK STORE

10% DISCOUNT
TO AUUG MEMBERS
ON OUR COMPLETE RANGE
OF COMPUTER, BUSINESS
AND GENERAL BOOKS

173 Elizabeth St, Brisbane Queensland 4000
Ph: (07) 3229 4677 Fax: (07) 3221 2171 Qld Country Freecall: 1800 177 395
american_bookstore@compuserve.com

Name: Date:

Address:

Post Code:

Phone Number:

Payment Method: □ Cheque □ Money Order □ Amex □ Bankcard
□ Diners □ Mastercard □ Visa

Card Number:

Expiry Date: Signature:

This is a: □ Special Order □ Mail Order □ Book on Hold

QUANTITY TITLE PRICE

SUBTOTAL $

LESS 10% DISCOUNT $

POST & PACK $

TOTAL $

POSTAGE AND HANDLING FEES: 1 BOOK $6.00 2-4 BOOKS $7.00
BOOKS OVER $70.00 WE WILL SEND CERTIFIED - PLEASE ADD ANOTHER $1.50 OR WAIVE CERTIFIED DELIVERY.

FOR SPECIAL ORDERS, PLEASE ENCLOSE $10.00 PER BOOK AS A DEPOSIT.

The Open Source Lucky Dip

Con Zymaris
conz@cyber.com.au

> *"He's a Unix Wizard,*
> *I just can't get the gist*
> *A Unix wizard's*
> *Got such a mental twist."*
> *-- Unix Wizard (to the tune of Pinball Wizard)*

✧ ✧ ✧

Hello again.

Let's delve into a few more useful, and some perhaps not so useful apps this issue, along with any pertinent news stories (with long half-lives ;-)

As always with the projects and applications presented here, caveat emptor! Be sure to check the applications out in some detail before progressing too far with production-level deployment.

✧ ✧ ✧

> *"I can neither deny nor confirm the rumor that we're working with the Alsatians doing a laser gun"*
> *-- Linus Torvalds, in response to rumors that Transmeta was working with alien technology*

✧ ✧ ✧

Onto our apps for this issue:

✧ ✧ ✧

APACHE JSERV

Java based servlets have become a compelling vehicle for deploying Java technology through an increasing number of organisations. And what better way to deploy a Java servlet engine than through the World's favourite web server? Apache JServ is now on version 1.1b1. It is a Java servlet engine; an external process written in Java which runs separate from the web server and handles requests to Java servlets. mod_jserv is an Apache HTTPD module written in C which provides an interface to the Apache JServ engine.

Homepage: `http://java.apache.org/`
Author: The Java Apache Project
License: BSD type

✧ ✧ ✧

ANTEATER: A SENDMAIL LOG ANALYZER

Anteater seems to be something that may be of use to all Unix/FreeBSD/Linux sysadmins. It is a log analyser for Sendmail MTA logfiles. It's written in C++ and the author claims it to be very easy to customise. It handles input, output, and the analysis are done by modular class objects with a clear interface. Currently there are modules for reading the `syslog` format that do up to seven useful analyses and write the result in plain ASCII to `stdout`.

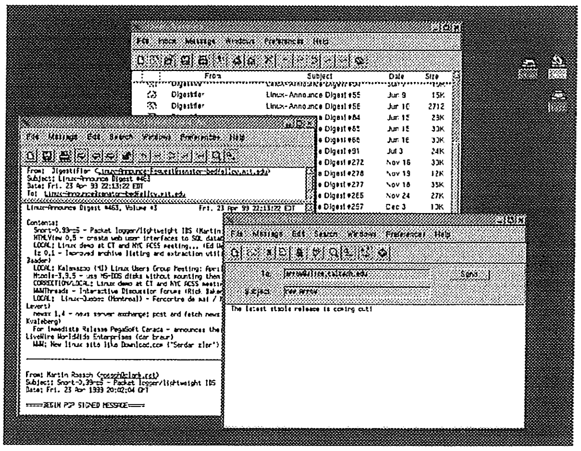
Homepage: `http://www.profzone.ch/anteater/`
Author: Tobias Erbsland
License: GPL

✧ ✧ ✧

ARROW: GUI MAILER

For those of you who have yet to settle upon an emailer, perhaps you should consider Arrow. Its creator Glenn Bach claims arrow to be an elegant, powerful, graphical interface to electronic mail. Arrow does seem to be fairly featureful, but then it would need to compete with the console-based competition like mutt ;-) In Arrow, each mailbox and message are shown in a separate window, thereby allowing one to simultaneously open as many mailboxes and view and compose as many messages as one wishes. This visual paradigm also allows one to drag-and-drop text between messages and organise ones mail by dragging messages between mailboxes.

Homepage:
`http://www.newplanetsoftware.com/arrow/`
Author: Glenn W. Bach
License: OpenSource


File Mess Message Windows Preferences Help
File Message Edit Search Windows Preferences Help
File Message Edit Search Windows Preferences Help
Send

	From	Subject	Date	Size
	Digestifier	Linux-Announce Digest #55	Jun 9	15K
	Digestifier	Linux-Announce Digest #56	Jun 10	2712
		e Digest #64	Jun 15	23K
		e Digest #65	Jun 15	33K
		e Digest #66	Jun 16	33K
		e Digest #91	Jul 3	24K
		e Digest #272	Nov 16	33K
		e Digest #276	Nov 19	12K
		e Digest #277	Nov 18	35K
		e Digest #265	Nov 24	27K
		e Digest #257	Dec 3	13K

```
From: Digestifier <Linux-Announce-Request@senator-bedfellow.mit.edu>
Subject: Linux-Announce Digest #463
Date: Fri, 23 Apr 99 22:13:22 EDT
To: Linux-Announce@senator-bedfellow.mit.edu

Linux-Announce Digest #463, Volume #3          Fri, 23 Apr 99 22:13:22 EDT

Contents:
  Snort-0.99-b5 - Packet logger/lightweight IDS (Martin
  HTMLView 0.5 - create web user interfaces to SQL datab
  LOCAL: Linux demo at CT and NYC ACSS meeting... (Ed W
  lz 0.1 - Improved archive listing and extraction util
Baader)
  LOCAL: Kalamazoo (MI) Linux Users Group Meeting: Apri
  Mtools-3.9.5 - use MS-DOS disks without mounting them
  CORRECTION/LOCAL: Linux demo at CT and NYC ACSS meetin
  WWWThreads - Interactive Discussion Forums (Rick Baker
  LOCAL: Linux-Quebec (Montreal) - Rencontre de mai / M
Levert)
  newsx 1.4 - news server exchange: post and fetch news
Kvaleberg)
  For Immediate Release PegaSoft Canada - announces the
LiveWire WorldWide Enterprises (der braur)
  WWW: New linux site like Download.com ("Serdar Yegulalp")

From: Martin Roesch <roesch@clark.net>
Subject: Snort-0.99-b5 - Packet logger/lightweight IDS
Date: Fri, 23 Apr 1999 20:02:04 GMT

-----BEGIN PGP SIGNED MESSAGE-----
```

To: arrow@its.caltech.edu
Subject: Re: Arrow

The latest stable release is coming out!

✧ ✧ ✧

AUTORPM: AUTOMATED RPM INSTALLER

If you've ever wanted to mirror RPMs from an FTP site, keep installed RPMs consistent with an FTP site or local directory and keep installed RPMs in a cluster or network of systems consistent. then AutoRPM is the program for you! It is apparently highly flexible and even contains a very nice, menu-driven Interactive-Install mode.

Homepage:
`http://www.kaybee.org/~kirk/html/linux.html`
Author: Kirk Bauer
License: GPL

✧ ✧ ✧

"A surprising development in the linux-kernel mailing list surfaced when Alan Cox announced the release of a 2.2 Linux kernel existing both as an official stable kernel and as a prepatch kernel."
-- Alan Cox Releases Quantum Kernel

✧ ✧ ✧

ASP2PHP: MOVE AWAY FROM ASP TO NICE, PORTABLE PHP WEB-SCRIPTING

Michael Kohn has recently released this new version of his ASP to PHP converter. Unlike Microsoft's ASP, PHP an embedded web-scripting technology is both multi-platform and open source. asp2php converts WWW Active Server Pages (ASP, essentially VBScript) files that run on the Microsoft IIS Web Server into PHP pages to run on Apache, and will allow you to move most of your ASP project automatically, leaving some residual clean-up by hand.

Homepage: `http://home.i1.net/~naken/asp2php/`
Author: Michael Kohn
License: free to use but restricted

✧ ✧ ✧

DYNDNS: DYNAMIC DNS SERVER

DNS servers like DynDNS may be the shape of things to come. DynDNS is a dynamic DNS server written in Python, which supposedly has a simple but powerful CGI frontend and command-line clients. It allows users to do things not possible with many other similar services, such as controlling one's own domain and creating a tree of hostnames.

Homepage: `http://www.ahv.cx/`
Author: Brian Shirk
License: GPL

✧ ✧ ✧

ETHEREAL 0.8.2

For all you wire-tappers out there, Ethereal is a network protocol analyser, or sniffer, that lets you capture and interactively browse the contents of network frames. According to project author Gerald Combs, the goal of the project is to create a commercial-quality analyser for Unix and to give Ethereal features that are missing from closed-source sniffers. Ethereal supports a large number of protocols; a full list is given at the homepage.

Homepage: `http://ethereal.zing.org/`
Author: Gerald Combs
License: GPL

✧ ✧ ✧

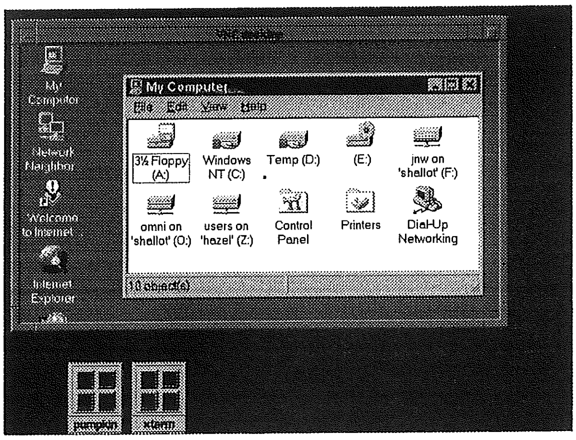
"The box said "Requires Windows 95 or better." I can't understand why it won't work on my Linux computer."

✧ ✧ ✧

VNC: VIRTUAL NETWORK COMPUTING

This has to be one of the coolest tools available from the open source world. If you have the intermittent need to interoperate with Windows applications, VNC is for you. If you haven't done so already, download and trial it, you will not be disappointed. VNC stands for Virtual Network Computing. It is, in essence, a remote display system which allows you to view a computing 'desktop' environment not only on the machine where it is running, but from anywhere on the Internet and from a wide variety of machine architectures. VNC was written by AT&T's Cambridge Labs and is distributed under the GPL.

Homepage: `http://www.uk.research.att.com/vnc/`
Author: AT&T Labs, Cambridge, UK
License: GPL



✧ ✧ ✧

BOA: A DIFFERENT TYPE OF HTTP SERVER

The author of Boa claims it to be the king-of-the hill for web-serving speed. Boa is described as a single-tasking HTTP server. That means that unlike traditional web servers, it does not fork for each incoming connection, nor does it fork many copies of itself to handle multiple connections. It internally multiplexes all of the ongoing HTTP connections, and forks only for CGI programs (which must be separate processes), automatic directory generation, and automatic file gunzipping. This seems to be what the other web-serving speed 'daemon' (;-), Zeus, does. Paul Phillips, the program's author states that preliminary tests show boa is capable of handling several hundred hits per second on a 100 MHz Pentium, and dozens of hits per second on a lowly 20 MHz 386/SX.

Homepage: `http://www.boa.org/`
Author: Paul Phillips
License: GPL

✧ ✧ ✧

GANYMEDE: DIRECTORY SERVICES FOR THE REST OF US!

The blurb from the Ganymede project web-page states: Ganymede is a portable and customisable network directory management system, released under the GNU General Public License. It is free software. It is similar in concept (if not in scale) to network directory systems like Microsoft's ActiveDirectory and Novell's Novell Directory Services. Ganymede differs in that it is written entirely in Java (making everything very portable), and in that it is designed to provide management for existing NIS, DNS, LDAP, and other network directory servers, not to replace them. So if you feel that open, interoperable directory services (and not closed, expensive proprietary ones!) give Ganymede some consideration. There's a lot of code here, and the project developers have put a lot of work into this one.

Homepage: `http://www.arlut.utexas.edu/gash2/`
Author: Jonathan Abbey
License: GPL

✧ ✧ ✧

GNU PARTED: FREE PARTITION MAGIC(TM)-LIKE TOOL

My sources tell me this is written by a Melbournian, and while this tool is still in its infancy, it looks very promising. GNU parted allows you to create, destroy, resize and copy partitions. It currently supports ext2 and FAT (FAT16 and FAT32) filesystems and Linux swap devices, and MS-DOS disk labels.

Homepage: `http://www.gnu.org/software/parted/`
Author: Andrew Clausen, Lennert Buytenhek, Matt Wilson
License: GPL

✧ ✧ ✧

OPENMERCHANT: FULL-BLOWN E-COMMERCE PACKAGE

OpenMerchant is another contender for the ultimate e-commerce package. Written by OpenSales in the US, it's their attempt at garnering a bigger potential market by releasing this major application into the open source space. This approach has worked well for the guys who released Zope, so big things may be in store for Open Merchant. The application is based on Perl and vanilla SQL. Targeted at Linux, MySQL, and Apache, it will run on a wide range of OS, DBMS, and webserver options. In addition to a shopping cart, it provides tools for content, merchandise, statistics, vendor, order, inventory, search term, and customer service management. The authors claim that its modular design allows it to be used as a complete or partial solution. Currently in pre-beta, the distribution includes source, documentation, and a sample store site.

Homepage: `http://www.opensales.org/`
Author: OpenSales, Inc.
License: GPL

✧ ✧ ✧

QUICK PASSWORD: PASSWORD GENERATOR

This app would rate a 'must-have' status on many sites that I know ;-) Quick Password is a program that generates passwords of user specified lengths which are easy to remember and pronounce. The passwords are reasonably secure and can easily be remembered because they are composed of dictionary words that are pronounceable. The program generates passwords of between 2 and 10 words each between 3 and 10 characters in length separated by non-alpha characters. It is usable as a web application from the home page if you do not wish to download it. So, while it may not generate passwords which are secure enough for my security-focused colleague Mike Ciavarella (;-) chances are that they would be fine for the rest of us.

Homepage: `http://www.zaph.com/Password/`
Author: Dan Grassi
License: GPL

✧ ✧ ✧

"Carpe Daemon -- Seize the background process!"

"Type cat vmlinuz > /dev/audio to hear the Voice of God."

❖

OpenBSD Merchandise For Sale

AUUG Inc has a limited amount of CD-ROMs and T-shirts from the Open BSD project available for sale to members. To order, please fill in the form below, then send it, with payment, to:

AUUG Inc.
PO Box 366
Kensington, NSW, 2033

or fax your order (credit card payment only) to (02) 8824 9522. Queries: call Liz Carroll on 1800 625 655.

Orders will be dispatched by express post after payment is confirmed. Orders will be processed in order of receipt, so it is important to include a contact number so that if we have run out of what you want, we can work out a substitute or refund.

AUUG Inc. OpenBSD Merchandise Order Form

Name: ______________________________

Address: ______________________________

____

City: ____________ State: ________ Postcode ______

Phone Number: ______________________________

Email: ______________________________

ITEM	SIZE	QTY	PRICE	TOTAL
OpenBSD 2.5 CD-ROMS	n/a		$45.00	$
& OpenBSD 2.5 T-shirt	L		$25.00	$
	XL		$25.00	$
	XXL		$25.00	$
Blowfish T-shirt (C code on back)	L		$25.00	$
	XL		$25.00	$
	XXL		$25.00	$
Wire Frame design – black background	L		$25.00	$
	XL		$25.00	$
	XXL		$25.00	$
Wire Frame design – navy blue background	L		$25.00	$
	XL		$25.00	$
	XXL		$25.00	$
Wire Frame design – dark green background	L		$25.00	$
	XL		$25.00	$
Postage and handling (per order)	N/a	1	$10.00	$10.00

GRAND TOTAL: $

Payment by: □ Cheque (enclosed)
□ Bankcard □ Mastercard □ Visa

Card Number: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Name on card: ______________________________

Expiry: ___/___ Signature: ______________________________

AUUG Corporate Members

as at 22 February 2000

- Andersen Consulting
- ANI Manufacturing Group
- ANSTO
- Australian Bureau of Statistics
- Australian Centre for Remote Sensing (ACRES)
- Australian Geological Survey Organisation
- Australian Industry Group
- Australian National University
- Australian Taxation Office
- Barwon Water
- BHP Information Technology
- British Aerospace Australia
- Bunnings Building Supplies
- Burdett Buckeridge Young Ltd.
- Bureau of Meteorology
- Bureau of Rural Sciences
- C.I.S.R.A.
- Camtech
- Cape Grim B.A.P.S
- Central Queensland University
- Centrelink
- CITEC
- Comcare Australia
- Commercial Dynamics
- Computer Science, Australian Defence Force Academy
- Computing Services
- Corinthian Engineering 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
- Defence Housing Authority
- Department of Communications and the Arts
- Department of Defence
- Department of Defence (T2 Section)
- Department of Defence (TC Section)
- Department of Environment & Natural Resources
- Department of Environment, Heritage & Aboriginal Affairs
- Department of Land & Water Conservation
- Deutsche Bank
- Energex
- Environmental Resources Information Network (ERIN)
- Financial Network Services Pty. Ltd.
- Fremantle Port Authority
- G.James Australia Pty. Ltd.
- Great Barrier Reef Marine Park Authority
- HIH Insurance
- HIH Winterthur
- Information Technology Consultants
- 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
- NRMA Information Ltd.
- NSW Agriculture
- NSW Public Works & Services, Information Services
- Peter Harding & Associates Pty. Ltd.
- Philips Broadband Network
- Platniium Technology Solutions
- Powerhouse Museum
- Primary Industries & Energy
- Qantas Information Technology
- QLD Department of Education
- QLD Department of Education, Information Systems Services Branch
- Queensland University of Technology
- Rinbina Pty. Ltd.
- SCO
- Security Mailing Services Pty Ltd
- Snowy Mountains Authority
- Softway Pty. Ltd.
- Southern Cross University
- St. John of God Health Care Inc.
- St. Vincent's Private Hospital
- Stallion Technologies Pty. Ltd.
- Standards Australia
- State Library of Victoria
- Storage Technology of Australia
- Sun Microsystems (Australia) Pty Ltd
- TAB Queensland Limited
- Technix Consulting Group Pty Ltd
- The Fulcrum Consulting Group
- The University of Western Australia
- The Walter & Eliza Hall Institute
- Tower Technology Pty. Ltd.
- Universities Admissions Centre Pty Ltd
- University Computing Services, Stirling Highway
- University of Adelaide
- University of Melbourne
- University of New South Wales
- University of Queensland
- University of Sydney
- University of Technology, Sydney
- Victoria University of Technology
- Westrail
- Workcover Queensland

Summary

AUUG Inc. Executive Meeting

NOTE: this is a summary of the meeting, not the minutes. The aim is to produce timely feedback on AUUG executive activity to the membership. Only major items are listed. Interested members should consult the full minutes when they become available for additional information.

Date:	30/1/2000
Time:	10:00 AM to 4.00 PM
Location:	Hungry Jacks Parramatta

Thanks to Luigi Cantoni for organising the venue.

Attendees:

David Purdue	DP (Meeting Chair)
Elizabeth Carroll	EC (Minutes Secretary)
Luigi Cantoni	Lca
Malcolm Caldwell	MC
Peter Gray	PG
David Newall	DN
Michael Paddon	MP
Frank Crawford	FC

(Guest, Programme Chair for AUUG2K)

Apologies were received from

Gunther Feuereisen	GF
Mark White	MW
Stephen Boucher	SB
Lucy Chubb	LC

The president indicated there appeared to be some increased awareness of AUUG amongst the media and pointed the recent request for comment on the Microsoft monopoly decision by the Australian and radio 2BL.

Discussion followed on the organisations finances. Previous problems with budgeting and reporting have mostly been resolved. While not flush with funds, the financial position of the organisation has improved substantially over the last 12 months due to cost control measures introduced. The executive agreed to continue to improve the budgeting process. Significant improvements have been made to the membership database. Liz Carroll is ensuring all information is kept accurate.

Changes to the organisations logo and tagline were then discussed. As usual, opinions were mixed. It was agreed that AUUGN will continue to use the current logo without a tagline for the next 12 months.

As proposed at the last meeting, action towards the formation of a BSD users Special Interest Group (SIG) was discussed. Michael Paddon will be coordinating efforts in this direction, and a bsd@auug mailing list will be established.

The major item on the agenda was the 2000 conference. A conference programme committee has been formed. Several speakers have been invited but there are no definite acceptances yet. The registration cost was discussed. It was agreed that the cheaper venue should allow for a reduction in the registration cost. Budgets based on various registration fees will be prepared for later consideration.

A small 1 day symposium on "Open Source" is being organised for Adelaide in April or May.

A 1 day symposium on system/network security will also be organised for November. Brisbane was agreed upon as the most likely venue.

The WA chapter requested that AUUG underwrite the cost of a speaker for an event jointly organised with SAGE-AU. This was approved.

The production of membership cards was discussed. It was agreed that the current cards were overly expensive. It was agreed to investigate a new card format similar to the USENIX membership cards. The membership application form will also be redesigned. Membership fees need to increase to take into account the GST.

Interaction with the press was discussed and it was agreed to form a group of members who could comment on current IT issues and offer comment in the press.

Changes to the constitution were discussed. Several changes need to be made. Changes discussed included new membership classes and moving the financial year. Changes will be drafted and proposed in the upcoming Management Committee elections.

Unix Traps and Tricks

Jerry Vochteloo
jerry@socs.uts.edu.au

Welcome to perhaps the last time that we print Unix Traps and Tricks, the contributions from AUUG members has long ago dried up. I guess if anyone wants to column to continue, they should contribute, otherwise drink a quiet beer to its demise.

Contributions to `jerry@socs.uts.edu.au`,

[Editor's Note: for several years now, we have kept UT&T going on a pacemaker; it's very hard to keep the column going if no-one submits Traps and Tricks.

I believe in tradition. For as long as I've been in AUUG, UT&T has been there and I would not like to be the one who pulls the plug. However, if we don't have content, there is not much we can do.

As I see it, we have some choices:

a) retire UT&T
b) receive a flood of contributions from you out there, or
c) we can look at going through back issues of AUUG, and looking for Traps and Tricks from ages past. The issues of AUUGN might be out of date, but some of the Traps and Tricks are as real today, as they were in times past.

So, what do we do? This is your journal. What would you like to see?

Email me and let me know. If you have any other ideas – I'd be glad to hear them

Gunther]

✧ ✧ ✧

TESTSYS

Jerry Vochteloo
jerry@socs.uts.edu.au

I have always been interested in how well protection mechanisms are applied by users. As such I have a little script which I run every now and then (when my system administrator is not looking), which checks to see if people have made any permission errors in their home directories. Note that I generate the list of home directories using `ypcat passwd` you might need to change this to suit.

The script is pretty configurable, allowing you to specify which files to check in each directory, but has defaults built in so it can be used by script kiddies. I have included the two config files that I use.

This is the shell script `testsys`:

```
#!/bin/sh
#
#   Checks to see if other people have their directory
#   permissions correct.
#
#   It basically checks every home directory that it can
#   find and looks for files that have the wrong permission
#
#   Files are checked for readability or writability
#   eg .Xauthority is useful if it is readable.
#
#   The script can read two files:
#       testread  -> those files to check for readability
#       testwrite -> files to check for writability
#   if you don't have these file, it will check some defaults
#
#   testsys -h will give help
#
```

```
#   Copyright J.Vochteloo 2000, all rights reserved.
#

configdir=$HOME/scripts
logfile=$HOME/log2.file

# arglist processing, sigh...

while [ $# -ne 0 ];
do
    case $1 in
    "-v"|--verbose)
             verb=1
                shift ;;
    "-h"|"--help")
             cat <<_EOF
testsys: checks a system for unsecure user directories.
usage: testsys [-v] [-h] [-l file] [-c dirn]
       -v:           verbose, prints each directory name that is checked.
       -h:           help, print this help message.
       -c configdir: where to look for config files (testread, testwrite)
       -l logfile:   set the logfile [default ~/log2.file]
_EOF
       exit ;;
    "-c")       shift
             if [ -d $1 ]; then
                 configdir=$1
             else
                 echo "Error: no such directory $1"
                 exit
             fi
             shift ;;
    "-l")       shift
             logfile=$1
             shift ;;
    esac
done

# generate these lists once

if [ -f $configdir/testwrite -a -r $configdir/testwrite ]; then
    writetargets=`cat $configdir/testwrite`
else
    if [ "x$verb" = "x1" ]; then
       echo "Warning: No file testwrite found, using defaults"
    fi
    writetargets=".profile
.zshrc
.cshrc
.kshrc
.xsession
.xinitrc"
fi

if [ -f $configdir/testread -a -r $configdir/testread ]; then
    readtargets=`cat $configdir/testread`
else
   if [ "x$verb" = "x1" ]; then
       echo "Warning: No file testread found, using defaults"
   fi
   readtargets=".Xauthority"
fi

# perform the testing

if [ "x$USER" = "x" ]; then
    myid=`whoami`
fi
for i in `ypcat passwd | grep -v $myid | cut -d':' -f6 ` # generate all home dirns
do
    if [ -d $i ]; then
    if [ "x$verb" = "x1" ]; then
      echo "Testing $i ---------------------"
    fi
    if [ -x $i ]; then
      if [ "x$verb" = "x1" ]; then
        echo "."
      fi

     # check writables

      for j in $writetargets
      do
      if [ -f $i/$j -a -w $i/$j ]; then
          echo "=====> Writable $i/$j"
      fi
      done

     # check readables

      for j in $readtargets
      do
      if [ -f $i/$j -a -r $i/$j ]; then
          echo "=====> Readable $i/$j"
      fi
      done
    fi
  fi
done  | tee $logfile
```

This is the test file `testwrite`:

```
.zshrc
.xsession
.Xauthority
.chsrc
.ctwmrc
.profile
.login
.plan
.fvwmrc
.xinitrc
.slirprc
.incoming-mail
.netscape/preferences
.cshrc
.tchrc
.kshrc
.emacs
.mailcap
.rhosts
```

This is the test file `testread`:

```
.zshrc
.xsession
.Xauthority
.chsrc
.ctwmrc
.profile
.login
.plan
.fvwmrc
.xinitrc
.slirprc
.incoming-mail
.netscape/preferences
.cshrc
.tchrc
.kshrc
```

```
.emacs
.mailcap
.rhosts.Xauthority
.incoming-mail
.pgp/pubring.pgp
.pgp/secring.pgp
```

❖

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	The Wesley Centre Pitt Street Sydney 2000	

Up-to-date information is available by calling AUUG on 1800 625 655.

Application for Institutional Membership

Section A: MEMBER DETAILS

The primary contact holds the full member voting rights and two designated representatives will be given membership rates to AUUG activities including chapter activities. In addition to the primary and two representatives, additional representatives can be included at a rate of $88 each. **Please attach a separate sheet with details of all representatives to be included with your membership.**

NAME OF ORGANISATION: ____________________

Primary Contact

Surname ____________________ First Name ____________________
Title: ____________________ Position ____________________
Address ____
Suburb ____________________ State ____________ Postcode ____________
Telephone: Business ____________________ Facsimile ____________________
Email ____________________ Local Chapter Preference ____________________

Section B: MEMBERSHIP INFORMATION.

Renewal/New Institutional Membership of AUUG *(including Primary and Two Representatives)*	□	$429.00
Surcharge for International Air Mail	□	$132.00
Additional Representatives Number	□	@ $88.00

Rates valid as at 1 March 2000. Memberships valid through to 30 June 2001 and include 10% GST.

Section C: PAYMENT

Cheques to be made payable to **AUUG Inc** *(Payment in Australian Dollars only)*

For all overseas applications, a bank draft drawn on an Australian bank is required. Please do not send purchase orders.

-OR-

□ Please debit my credit card for A$____________

□ Bankcard □ Visa □ Mastercard

Name on Card ____________________
Card Number ____________________
Expiry Date ____________________
Signature ____________________

Please mail completed form with payment to:

Reply Paid 66
AUUG Membership Secretary
PO Box 366
KENSINGTON NSW 2033

Or Fax to:

AUUG Inc
(02) 8824 9522

Section D: MAILING LISTS

AUUG mailing lists are sometimes made available to vendors. Please indicate whether you wish your name to be included on these lists:

□ Yes □ No

Section E: AGREEMENT

I/We agree that this membership will be subject to rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

I/We understand that I/we will receive two copies of the AUUG newsletter, and may send two representatives to AUUG sponsored events at member rates, though I/we will have only one vote in AUUG elections, and other ballots as required.

Signed: ____________________
Title: ____________________
Date: ____________________

AUUG Secretariat Use

Chq: bank ____________ *bsb* ____________
A/C: ____________ *#* ____________
Date: ____________ *$* ____________
Initial: ____________ *Date Processed:* ____________
Membership#: ____________

AUUG Inc

UNIX® AND OPEN SYSTEMS USERS

Membership Application

AUUG Inc
PO Box 366, Kensington NSW 2033, Australia

Tel: (02) 8824 9511
Free Call: 1 800 625 655
Fax: (02) 8824 9522
email: auug@auug.org.au

ACN A00 166 36N (incorporated in Victoria)

http://www.auug.org.au

AUUG Inc is the Australian UNIX and Open Systems User Group, providing users with relevant and practical information, services and education through co-operation among users.

AUUG OFFERS SOMETHING FOR YOU!!

AUUGN

Technical Newsletter
AUUG's quarterly publication, keeping you up to date with the world of UNIX and open systems.

Education

Tutorials
Workshops

Events.....Events......Events

- Annual Conference & Exhibition
- Overseas Speakers
- Local Conferences
- Roadshows
- Monthly Meetings

DISCOUNTS

*to all AUUG events and **education**.*

Reciprocal arrangements with overseas affiliates.

*Discounts with various **internet service providers, software, publications** and more...!!*

Connections

- Newsgroup aus.org.auug

Application for Individual or Student Membership

Section A: PERSONAL DETAILS

Surname ______________________ First Name ______________________
Title: ______________________ Position ______________________
Organisation ______________________
Address ______________________
Suburb ______________________ State ______________ Postcode ________
Telephone: Business ______________________ Private ______________________
Facsimile: ______________________ E-mail ______________________

Section B: MEMBERSHIP INFORMATION

Please indicate whether you require Student or Individual Membership by ticking the appropriate box.

RENEWAL/NEW INDIVIDUAL MEMBERSHIP		
Renewal/New Membership of AUUG	□	$110.00
RENEWAL/NEW STUDENT MEMBERSHIP		
Renewal/New Membership of AUUG *(Please complete Section C)*	□	$27.50
SURCHARGE FOR INTERNATIONAL AIR MAIL	□	$66.00

Rates valid as at 1 March 2000. Memberships valid through to 30 June 2001 and include 10% GST.

Section C: STUDENT MEMBER CERTIFICATION

For those applying for Student Membership, this section is required to be completed by a member of the academic staff.

I hereby certify that the applicant on this form is a full time student and that the following details are correct.

NAME OF STUDENT: ______________________
INSTITUTION: ______________________
STUDENT NUMBER: ______________________
SIGNED: ______________________
NAME: ______________________
TITLE: ______________________
DATE: ______________________

Section D: LOCAL CHAPTER PREFERENCE

By default your closest local chapter will receive a percentage of your membership fee in support of local activities. Should you choose to elect another chapter to be the recipient please specify here:

____

Section E: MAILING LISTS

AUUG mailing lists are sometimes made available to vendors. Please indicate whether you wish your name to be included on these lists:

□ Yes □ No

Section F: PAYMENT

Cheques to be made payable to **AUUG Inc**
(Payment in Australian Dollars only)

For all overseas applications, a bank draft drawn on an Australian bank is required. Please do not send purchase orders.

-OR-

□ Please debit my credit card for A$__________

□ Bankcard □ Visa □ Mastercard

Name on Card ______________________
Card Number ______________________
Expiry Date ______________________
Signature ______________________

Please mail completed form with payment to:

Reply Paid 66
AUUG Membership Secretary
PO Box 366
KENSINGTON NSW 2033
AUSTRALIA

Or Fax to:

AUUG Inc
(02) 8824 9522

Section G: AGREEMENT

I agree that this membership will be subject to rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

Signed: ______________________
Date: ______________________

AUUG Secretariat Use

Chq: bank __________ *bsb* __________
A/C: __________ *#* __________
Date: __________ *$* __________
Initial: __________ *Date Processed:* __________
Membership#: __________