AUUGN

The Journal of AUUG Inc.

Volume 19 • Number 2

May 1998

Use the Source, Luke ... Again!

Internet Payment Systems

We meet Mark White from the Exec

Traps & Tricks, Solaris Musings and Book Reviews ...

ISSN 1035-7521

Print post approved by Australia Post - PP2391500002

AUUGN

The Journal of AUUG Inc.

Volume 19 • Number 2
May 1998

AUUG Membership and General Correspondence

The AUUG Secretary
PO Box 366
Kensington NSW 2033

Tel: 02 9361 5994
Fax: 02 9332 4066
Toll Free: 1800 625 655
Internet: *auug@auug.org.au*

AUUG Executive

President:
Michael Paddon
Michael.Paddon@auug.org.au
Australian Business Access
723 Swanson Street
Carlton VIC 3053

Vice President:
Lucy Chubb
Lucy.Chubb@auug.org.au
Softway Pty. Ltd.
79 Myrtle St
Chippendale NSW 2008

Secretary:
David Purdue
David.Purdue@auug.org.au
Sunsoft Pacific
Level 4, Sunsoft Building
44 Hampden Road
Artarmon NSW 2064

Treasurer:
Stephen Boucher
Stephen.Boucher@auug.org.au
MTIA
509 St. Kilda Road
Melbourne VIC 3004

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

Luigi Cantoni
Luigi.Cantoni@auug.org.au
STM
Suite 3, 77 Millpoint Road
South Perth WA 6153

Peter Laytham
Peter.Laytham@auug.org.au
SCO
Level 7, 157 Walker Street
North Sydney NSW 2060

Mark White
Mark.White@auug.org.au
Tandem Computers
143 Coronation Drive
Milton QLD 4064

AUUG Business Manager:

Elizabeth Egan
busmgr@auug.org.au
Level 4, 90 Mount Street
North Sydney NSW 2060

AUUGN Editor

Günther Feuereisen
gunther@ibm.net
PO Box 366
Kensington NSW 2033

AUUGN Correspondence

Please send all correspondence regarding AUUGN to:

AUUGN Editor
auugn@auug.org.au
PO Box 366
Kensington NSW 2033

Submission Guidelines

Submission guidelines for AUUGN contributions are regularly posted on the **aus.org.auug** news group.

They are also available from the AUUG World Wide Web site at:

http://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
Australia

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.

Tel: (02) 9361 5994
Fax: (02) 9332 4066

During normal business hours.

Disclaimer

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

Contribution Deadlines for AUUGN in 1998/99

Volume 19 • Number 3 • August 1998 : **July 7th 1998**

Volume 19 • Number 4 • November 1998 : **October 7th 1998**

Volume 20 • Number 1 • February 1999 : **January 7th 1999**

Volume 20 • Number 2 • May 1999 : **April 7th 1999**

Editorial

Günther Feuereisen <gunther@ibm.net

Something I have been watching for years is Unix bashing. The old "Unix is dead .." articles.

Recently however, there have been some interesting articles on Intel's upcoming IA-64 processor, code-named "Merced". The interest is because suddenly everyone is talking of an IA-64 port of their flavour of Unix.

Now Unix on Intel Architecture is fairly common these days (Linux, SCO, Solaris, BSD to name a few), but when the likes of Digital and HP start talking of IA-64 ports, you start to pay a little closer attention.

What does this mean for Unix? It means that other competing Operating Systems, such as NT, will have to start to compete on a performance, scalability and reliability issues, as opposed to the "Well, if you buy NT on Intel, it will be much cheaper than buying a name-brand Unix on that company's proprietary architecture", which unfortunately is what sways a lot of decisions.

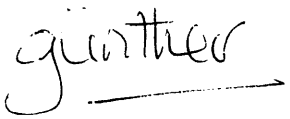
It's an exciting time for Unix – some vendors have even professed to shipping an IA-64 version of Unix in preference to an IA-64 version of NT. Now that is definitely not what the analysts have been trying to convince us would happen.

Watch this space. "Merced" is due in 1999 ..

Moving right along, don't forget, it's not too late to get your AUUG conference paper abstracts in! Also, Dan Klein is hitting the road, see page 9 for info on the roadshow.

And finally, on behalf of everyone, I wanted to say thank you to Michael Paddon for carrying the mantle of él Présidenté for the past three years. From my own point of view, it was nice to find someone who seems to sleep even less than I do :-)

Thanks Michael!



❖

President's Column

Michael Paddon <Michael.Paddon@auug.org.au>

When I first stepped forward to meddle in the running of AUUG, I was consumed by a righteous anger. The latter part of the eighties was a time of some dismay for those of us who had grown up with Unix, as we watched a unified vision and elegance torn apart for commercial convenience. Unix was a sweet young technology, still in its teens, and it was being put to brutal use by dissolute, middle aged, faceless suits with no thought but for the dollar she could turn tonight.

OK, over melodramatic I admit. But not far from the truth as the industry's giant, foundering oligarchies searched for a life preserver to save them from the tempest brought on by microprocessors, cheap computers and motivated programmers. Unix was a focal point of this revolution, and the big hardware vendors that survive to this day are the ones that were smart enough to see this, and not to be too proud to adapt.

So the rules changed. Everyone had access to computers. Portable languages and operating systems tore down the artificial barriers preventing the promulgation of software and ideas. A political battle had been won, and a massive transfer of power from the corporations to the individual programmer was there to cement the new status quo.

This was way too good to last, and in a sense this utopia was only ever approached asymptotically. The natural reaction of the corporation was to differentiate their product from their competition. Now, diversity is a key to successful evolution, and we have seen very successful use of this principle with projects such as Linux, FreeBSD and OpenBSD. Unfortunately, the differentiation here was more akin to the creation of a horrid and pathetic doublivore by a mad vivisectionist.

Not surprisingly, the worst of these experiments eventually died off, and the vendor specific versions of Unix that survive unto this day are those which received additional technical vitality from their owners rather than the poison of marketing positioning and customer locking.

What we tend to forget, in this latter part of the decade, was that there was a time when the entire idea of an open system could have died quietly. Indeed, the commercial powers of the time even succeeded in usurping the term "open". Can anyone actually rationally explain why the word "Open" is written with a capital "O" in so much marketing verbiage?

So there I was. The early nineties, and pissed off at all of the above. But even worse, the same influences had begun to infiltrate a number of user groups, including AUUG. Not only was our operating system, our code and our freedom at stake. These bastards were going to turn the one place that us shell-shocked hackers could get together for a quiet drink into some kind of marketing encounter group. If you've ever had your favourite watering hole turned into a yuppie wine bar, you'll start to appreciate just how bad this was.

That's the fire that got me involved in the AUUG committee, and eventually led me to run for President. It's driven the changes I've fought for, chosen the paths I followed and, just as importantly, the paths I chose not to take. I've now had the privilege to serve as President for three years, and I earnestly hope that the vision I espoused and the actions I took have been worthy of my predecessors, our membership and our common goals. In particular, my desire was always to bring AUUG back to the people who had founded it and supported it, the technologists to whom Unix and open systems meant so much. In the end, however, that is for you to judge, and I will not dishonour my post by attempting to sway you further.

However I have served, it is now time for a fresh viewpoint and a new fire to augment the old. Three years is time enough for any AUUG President, and I will be standing down come these next elections, though I will continue to serve on the committee for as long as the membership wills. I must admit, I am looking forward to working on several specific projects for AUUG that have been in gestation for several years.

The great thing about being an outgoing President, however, is opportunity to tell everyone how things should be run in the future, and I beg your indulgence in this instance.

As I have said, when I first joined the committee, I was full of fire about the extant influences upon Unix and AUUG. The world has changed since then, and the specific issues, so close to my heart, have long since run their course. In a larger sense however, the battle rages on, and the only thing that would ever cause me regret for the energy I have invested in AUUG would be to see us lose as this late stage.

Unix is about freedom. The freedom to write and share software. Why is this important? Simply because software is an expression of the infinitude of human thought. It is imagination, made corporeal, and yet not shackled by the tawdry bounds of time or space or the world we live in. Heavy concepts, yes, but given even a quantum of credence they start to explain why freedom of software is more important than commerce, and freedom of ideas is more important than, well, just about anything.

When you are fighting for freedom, it helps to have a banner to rally beneath. My claim is that Unix is that banner. It was the first popular expression of the ideas we are speaking of, and it still inspires innovation and evolution far beyond it's own domain. Let's recapture the word Unix from the clutches of trademark law, and put it into common use as the summary of everything that Linux, GNU, BSD and others stand for. Or if you don't like that, use your own label. It's all the same.

Now that we have named the battlefield, just what is it that we are fighting for? The recent rise of large software monopolies shows that freedom is a fragile thing indeed, and that freedom (if I may) still intends to use our work and intellect for it's own end. We need to recognise that the libertarian impulse that led us to embrace Unix as programmers is also at work in the world at large.

Software is not just C code, it encompasses the whole field of human creativity rendered into the digital continuum. This includes images, text, spreadsheets, models, networks, and literally everything else you can think of. Freedom of access to computers is now ubiquitous, but freedom of software is still an unrealisable ideal for much of humankind.

We are the enlightened ones. In most cases, this is not through hard work, but by inheriting the vision of others (certainly, I fall into that category). Nevertheless, we have the responsibility to promulgate the ideas and the freedom we take for granted. I have no doubt that these things are so basic to the human spirit that they will win out in the end, however we can contribute towards that happening sooner.

These are big ideas, driven by the concept that computing and software is not simply just a fast way of doing math, but a fundamental step in the evolution of humanity itself. Buy me a beer sometime, and I'll elucidate, if you are not already convinced. Hell, I don't care if I never convince you so long as you keep shouting.

Or a more mundane scale, what can AUUG do? What should AUUG be? The simple answer is always what it has been, even when it put on fancy airs and forgot from whence it sprang. AUUG is a bunch of technical people. Mostly software people, although those who fiddle with soldering irons are welcome and always have interesting stuff to say. Some of us write software for a living, some of us administer and run software for a living and some of us even run companies, sometimes for a living.

The thread that pulls us together is the fact that we want to learn and we want to share what we know. Some of us like to do this formally, in conferences and tutorials, and some of us prefer to mix high technology and beer. Intellectual freedom, software and alcohol are a potent cocktail.

The whole point, in the end, is to learn about neat stuff. Let's face it, AUUG became a forum for technology way beyond what any operating system encompassed years ago. Networking, windowing systems, various languages, programming environments, databases, graphics, and even domain specific applications are all in the AUUG mainstream. And rightly so.

Any software which is freely available, not only for use, but for study, modification and improvement follows the path blazed by Unix, and falls within the scope of AUUG. This is the what the AUUG membership wants to talk about over beer, and see at conferences -- the fun, new, neat stuff. The stuff that will change, and perhaps revolutionise, our industry.

Which segues nicely into AUUG's "commercial relevance". From my perspective, I am tempted to demand the relevance of commerce to the far grander chapter that computing and open systems are writing in the book of human experience. However, for the sake of the temporal, AUUG's commercial role is straightforward: we are the technical specifiers. We are the people who plan, architect and build tomorrow's systems.

We have the opportunity to work with business to build better, more open and more profitable systems. And at the same time we can continue to promulgate the Unix ideal. Commercial success and freedom are not at odds, and there are enough very successful example concerns out there to prove the point.

In closing, I'd simply like to say thank you for allowing me to play the part I have, in what has been some eventful years for AUUG. I have put a not inconsiderable chunk of my life into the organisation, but I have no doubt that I have received my investment back many-fold, not just in satisfaction but in visible, concrete change in our industry for the better. Should anyone be considering serving on an AUUG committee, I commend it to you. In the meantime, I'll be maintaining my rage, and I wish the new President the best of luck in taking AUUG towards the millennium.

❖

Upcoming USENIX Events

June 15-19, 1998, New Orleans, Louisiana
1998 USENIX TECHNICAL CONFERENCE

This year's program offers 22 tutorials, 23 refereed papers, invited talks by James "The Amazing" Randi, Eric Raymond, Kirk McKusick, Dennis Ritchie, John Quarterman, and Steve Mann (to name a few), and a track on freely redistributable software. There will also be a panel discussion on clustered computers, WIP reports, and Freenix BOFS hosted by Linus Torvalds, Richard Stallman, and others.

Time is running out...hotel savings deadline is May 22.

August 3-5, 1998, Seattle, Washington
2ND USENIX WINDOWS NT SYMPOSIUM

Forum for industrial and academic researchers and product developers basing their work in Windows NT.

Please note: The co-located Windows NT and LISA NT events share a day of six tutorials on August 5.

August 5-8, 1998, Seattle, Washington
LARGE INSTALLATION SYSTEMS ADMINISTRATION OF WINDOWS NT (LISA NT) CONFERENCE
Co-sponsored by SAGE, the System Administrators Guild

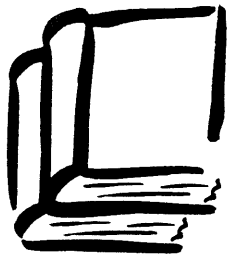
Share solutions with peers and experts to managing, scaling, and integrating large Windows NT environments.

August 31-September 3, 1998, Boston, Massachusetts
3RD USENIX WORKSHOP ON ELECTRONIC COMMERCE

Includes highly interactive sessions on Public Key Infrastructures Six tutorials: *two by Schneier on Cryptography *Smart Cards *Electronic Commerce Law *Secure Web Server *E-Payment Systems

If you need more information, send email to conference@usenix.org or call the Conference Office at 714-588-8649. Website is http://www.usenix.org/events/

❖



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.

Use the Source, Luke! . . . Again

Warren Toomey <wkt@henry.cs.adfa.oz.au>

So you call yourself a Unix hacker: you know what `bread()` is, and the various `splxx()` routines don't faze you. But are you *really* a Unix hacker? Let's have a look at a brief history of Unix and the community of Unix users and hackers that grew up around it, and some recent developments for real Unix hackers.

Unix took the academic world by storm in 1974 with the publication of Ken Thompson's paper about its design, which was published in the Communications of the ACM. While not containing many radically new ideas, Unix had an elegance, simplicity and flexibility that other contemporary operating systems did not have. Soon, lots of people were asking Bell Laboratories if they could get copies of this wondrous new system.

This was the cause of some concern within AT&T, because of the restrictions of an anti-trust decree brought against them in the '50s. This decree effectively stopped AT&T from selling or supporting software: they could only engage in telco business. Their solution to meet the Unix demand was to charge a nominal `license' fee to obtain Unix, and to distribute tapes or disks `as is'. You'd receive your disk in the mail with just a short note:

> Here's your rk05, Love, Dennis.

AT&T's stance on Unix was often seen as an OHP slide at early conferences:

> No advertising
> No support
> No bug fixes
> Payment in advance

``This slide was always greeted with wild applause and laughter'' says Andy Tanenbaum. This lack of support was tolerated for several reasons: Ken and Dennis did unofficially fix things if you sent them bug reports, and you also had the *full source code* to Unix.

At the time, having full source code access for a useful operating system was unheard of. Source code allowed Unix users to study how the code worked (John Lions' commentary on 6th Edition), fix bugs, write code for new devices, and add extra functionality (the Berkeley Software Releases, AUSAM from UNSW). The access to full source code, combined with AT&T's `no support' policy, engendered about the strong Unix community spirit which thrived in the late 70's and early 80's, and brought many Unix users groups into existence. When in doubt as to how a program (or the kernel) worked, you could always `Use the source, Luke!'.

During this period, Unix became wildly popular at universities and in many other places. In 1982, a review of the anti-trust decree caused the break-up of AT&T into the various ``Baby Bell'' companies. This gave AT&T the freedom to start selling software. Source code licenses for Unix became very expensive, as AT&T realised that Unix was indeed a money-spinner for them. Thus the era of Unix source code hackers ended, except for notable exceptions like the 4BSD work carried out at the University of California, Berkeley.

Those organisations lucky enough to have bought a `cheap' Unix source license before 1982 were able to obtain the 4BSD releases from UCB, and continue to hack Unix. Everybody else had to be satisfied with a binary-only license, and wait for vendors to fix bugs and add extra functionality. John Lions' commentary on how the Unix kernel worked was no longer available for study, being restricted to one copy per source code license, and not to be used for educational purposes.

What were Unix hackers going to do, with no Unix source code to hack any more? The solution was to create Unix clones which didn't require source code licenses. One of the first was Minix, created by Andy Tanenbaum, and aimed squarely at teaching operating systems. Early versions of Minix were compatible with 7th Edition Unix; the most recent version is POSIX compliant, and can run on an AT with 2 Meg of memory and 30 Meg of disk space.

Many Minix users tried to convince Andy to add features such as virtual memory and networking, but Andy wanted to keep the system small for teaching purposes. Eventually, one user called Linus Torvalds got annoyed enough that he used Minix to create another Unix clone with these extra features. And so Linux was born.

While Linux was taking off like a plague of rabbits, the BSD hackers were working on removing the last vestiges of Unix source code from their system. They thought they had done so, and released BSD/386, a version of 4.3BSD which ran on Intel platforms. AT&T, however, wasn't so sure about the complete removal of Unix source code, and took them to court about it.

Now, AT&T is not a good company to be sued by: they tend to have a small army of lawyers. Eventually, the conflict was settled out of court with a few compromises, and we now have several freely-available BSDs: FreeBSD, NetBSD and OpenBSD. Of course, they all come with source code.

The Unix hacker of the late 90's surely has an abundance of source code to hack on: Linux, Minix,

OpenBSD etc. But is she really a Unix hacker, or just a Unix clone hacker? Wouldn't it be nice if we could hack on real Unix, for old time's sake.

Unix turned 25 in 1993, which makes the early versions of Unix nearly antiques. Many of the old Unix hackers (hackers of old Unix, that is) thought the time had come to get the old, completely antiquated Unix systems back out for sentimental reasons. After all, ITS, CTSS and TOPS-20 had been rescued and made publically available, why not Unix.

At the time, Unix was undergoing a crisis of ownership. Did AT&T own Unix this week, or was it Novell, Hewlett-Packard or SCO? Unix is a trademark of someone, but I'm not sure who. After the dust had settled, SCO had the rights to the source code, and X/Open had dibs on the name `UNIX', which is probably still an adjective.

During the ownership crisis, Peter Salus, Dennis Ritchie and John Lions had begun to lobby Novell: they wanted John's Commentary on Unix to be made publically available in printed form. It wasn't until the Unix source code rights had been sold to SCO that this finally was approved. It helped to have some old Unix hackers, Mike Tilson and Doug Michels, inside SCO to fight the battle. You can now buy John Lions' Commentary on 6th Edition Unix (with source code) from Peer to Peer Communications, ISBN 1-57398-013-7. As Ken Thompson says: ``After 20 years, this is still the best exposition of a `real' operating system".

One of the restrictions on the Commentary's publication is that the Unix source contained within cannot be entered into a computer. Ok, so you can read the book, but what use is source code unless you can hack at it?!

At the time that SCO bought Unix, I began to lobby SCO to make the old source available again, unaware of the efforts to release the Lions' Commentary. SCO's initial response was ``this will dilute the trade secrets we have in Unix, and it wouldn't be economically viable.'' My efforts drew a blank.

To help bring greater lobbying power to bear on SCO, the PDP Unix Preservation Society was formed. Its aims are to fight for the release of the old Unix source, to preserve information and source from these old systems, and to help those people who still own PDP-11s to get Unix up and running on them. After realising that SCO was never going to make the old Unix source code freely available, we explored the avenue of cheap, personal-use source licenses. The Society set up a Web petition on the topic, and gathered nearly 400 electronic signatures.

Inside SCO, we were very fortunate to contact Dion Johnson, who took up our cause, and fought tooth and nail with the nay-sayers and the legal eagles at SCO. The combined efforts of the PUPS petition and Dion's hard work inside SCO has finally borne fruit.

On the 10th March, 1998, SCO made cheap, personal-use Unix source code licenses available for the following versions of Unix: 1st to 7th Edition UNIX, 32V, and derived systems which also run on PDP-11s, such as 2.11BSD. The cost of the license is US$100, and the main restriction is that you cannot distribute the source code to people without licenses. Finally, we can be real Unix hackers and `Use the Source, Luke!' again.

Acknowledgments and References

I'd like to thank Dion Johnson, Steven Schultz, the members of the PDP Unix Preservation Society, and the people who signed the PUPS petition, for their help in making cheap Unix source licenses available again. Dion, in particular, deserves a medal for his efforts on our behalf.

You can find more about the PDP Unix Preservation Society at:

http://minnie.cs.adfa.oz.au/PUPS/

and details on how to obtain your own personal Unix source license at:

http://minnie.cs.adfa.oz.au/PUPS/getlicense.html

SCO won't be distributing Unix source code as part of the license. PUPS members have volunteered to write CDs and tapes to distribute old versions of Unix to license holders. We currently have 5th, 6th, 7th Edition, 32V, 1BSD, all 2BSDs, Mini UNIX and Xinu. We are looking for complete versions of PWB Unix and AUSAM. We desperately want anything before 5th Edition: hopefully these early systems haven't gone to the bit bucket. Please contact us if you have anything from this era worth preserving.

If you are licensed and want a copy of the PUPS Archive, see the PUPS web page above for more information. We expect to be deluged by requests for copies, and so if you can volunteer to write CDs or tapes for us, please let us know.

You don't need own a PDP-11 to run these old systems. The PUPS Archive has a number of excellent PDP-11 emulators. If you have bought a copy of the Lions' Commentary (and you should), now you can run real 6th Edition Unix on an emulator. And if you want, you can hack the code!

Finally, it should be noted that source code access to 4.xBSD requires a 32V source code license. Coincidentally, the new SCO license covers 32V! There are rumours that the last product to come out of CSRG will be a set of CD-ROMs containing all the 3BSD and 4.xBSD releases. I'll let you know as soon as I hear anything.

Dan Klein
Australian Tour
presented by AUUG Inc

Dan Klein will be holding the following 1 day tutorials

Tutorial A: Setting Up And Administering A Web Server

Tutorial B: Security for Software Developers:
How to design code that withstands hostile environments

About the Presenter

Daniel Klein has been teaching a wide variety of Unix-related subjects since 1984, and has been involved with Unix since 1976. His experience covers a broad range of disciplines, including the internals of almost every Unix kernel released in the past 22 years, real-time process control, compilers and interpreters, medical diagnostic systems, system security and administration, web-related systems and servers, graphical user interface management systems, the 900-year history of drawing languages, and a racetrack betting system. He contributes regularly to the proceedings of the USENIX Association, and is also their tutorial coordinator and a frequent invited speaker. He holds a Masters of Applied Mathematics from Carnegie-Mellon University in Pittsburgh, and in his free time is a member of a capella choir and an improvisational comedy troupe.

AUUG Inc
UNIX® AND OPEN SYSTEMS USERS

Dates:

Mon *Tue*	*29 June* *30 June*	*Tutorial A* *Tutorial B*	*NT University* *Darwin*
Thu *Fri*	*2 July* *3 July*	*Tutorial A* *Tutorial B*	*Victoria Leagues* *Perth*
Mon *Tue*	*6 July* *7 July*	*Tutorial A* *Tutorial B*	*University of Tasmania* *Hobart*
Thu *Fri*	*9 July* *10 July*	*Tutorial A* *Tutorial B*	*International House* *Melbourne*
Mon *Tue*	*13 July* *14 July*	*Tutorial A* *Tutorial B*	*Australian National University* *Canberra*
Wed *Thu*	*15 July* *16 July*	*Tutorial A* *Tutorial B*	*Wesley Centre* *Sydney*
Mon *Tue*	*20 July* *21 July*	*Tutorial A* *Tutorial B*	*QUT* *Brisbane*

Schedule:

8:30 - 9:00 .....Registration & Coffee

9:00 - 10:30..... TUTORIAL

10:30 -11:00.....Coffee Break

11:00- 12:30..... TUTORIAL

12:30 - 1.30.....Lunch

1:30 - 3:00 ..... TUTORIAL

3:00 - 3:30 .....Coffee Break

3:30 - 5:00 ..... TUTORIAL

FOR FURTHER INFORMATION PLEASE CONTACT:

AUUG Secretariat
Ph: 1-800-625655
email: auugbiz@auug.org.au

AUUG WEBSITE:

http://www.auug.org.au

TUTORIAL A

Setting Up And Administering A Web Server

Intended Audience:

Administrators who are interested in creating a World Wide Web service for their company, and hence become their company's "Webmaster." The course is intended for people who have some knowledge of UNIX system administration.

The World Wide Web is the most widely used Internet service. Companies are quickly discovering that they need to be on the Web to provide information to customers and to keep up with the competition. This course describes how to set up and maintain a World Wide Web server on a UNIX platform. The servers covered in the course include the popular and freely available Apache and NCSA Web servers (these servers own approximately 60% of the server market). Topics covered include:

- **The Architecture of the Web**
- **The HTTP Protocol**
  HTTP Server to Browser Conversations
- **Compiling the Server**
- **Server Configuration**
  Creating "Virtual Hosts"
  Resource Configuration
  Access Configuration
  Per User Access
- **Analyzing and Rotating Logs**
  Making sense of Agent and Referrer Logs
- **Registering and Announcing the Server**
- **Web-Related Security Issues**
- **Electronic Commerce Issues**
- **Security and the Web**
  Operating System, CGI, and Software Considerations
  Setting Up and Configuring SSL (Secure Sockets Layer)
- **Server Performance Issues**
- **Using Multiple Servers**
- **Detecting Server Problems**

Setting up the web server is only half of the battle. Understanding exactly how the protocol works, what performance issues are critical, what security implications are and other nuances are just some of the important issues that all webmasters need to thoroughly understand. After completing this course, webmasters should have an in-depth understanding of their server environment and the critical issues surrounding ongoing maintenance.

TUTORIAL B

Security for Software Developers: How to design code that withstands hostile environments

Who should attend:

System managers and software engineers who are developing client/server applications that will be used over the Internet.

Prerequisites:

A strong background in UNIX and UNIX programming is recommended. Many examples will refer to C (and occasionally Perl) programming constructs. Familiarity with C is not a prerequisite, but familiarity with programming under UNIX is strongly recommended for attendees.

Overview:

This tutorial is for UNIX programmers who want to learn to write security critical networking software. Increasingly, client/server software is being deployed in hostile environments that it may not have been designed to withstand. Attendees will learn how to spot and avoid making typical flaws in security programming, using examples and case studies from existing applications.

Section 1: Basics

* Taxonomies of software and system flaws
* The importance of security
* Putting security at the right layer
* Orange book (C2, B1, B2 systems)
* Authentication versus Authorization

Section 2: Data Protocols

* How protocols are secure or insecure
* Designing a protocol for security
* Typical weaknesses of protocols

Section 3: Using cryptography

* Basics
  - Public key
  - Secret key
  - Certificates
* Randomness
* Algorithms
* Synchronizing protocols
* What cryptography can do for you
* What cryptography can't do for you

Section 4: Authentication

* What to authenticate
* Challenge/response
* Authenticating packet streams
* Publicly available authentication systems you can use

Section 5: Writing secure network daemons

* Chroot
* Setuid
* Minimizing code
* How to avoid doing everything as "root"

Section 6: Case studies

* A simple file transfer daemon
* Using file system permissions
* Locking up a process

Registration Form

Dan Klein - Australian Tour

presented by AUUG Inc

Section A: PARTICIPANT PERSONAL DETAILS (photocopy for additional attendees)

Surname _____ First Name _____

Title: _____ Position _____

Organisation _____

Address _____

Suburb _____ State _____ Postcode _____

Telephone: Business _____ Private _____

Facsimile: _____ E-mail _____

AUUG Membership No (if applicable) _____

Section B: TOUR LOCATIONS

Please indicate, by ticking the applicable box in which city you will be attending this one day seminar.

Tutorial A: ***Setting Up And Administering A Web Server***

Tutorial B: ***Security for Software Developers:***
How to design code that withstands hostile environments

Location		*Day*	*Date*	*Tutorial*	*Please tick the Tutorial(s) you will be attending*
Darwin	*NT University, Rm 39.1.37,*	*Mon*	*29 June*	*Tutorial A*	
	Darwin NT 0909	*Tue*	*30 June*	*Tutorial B*	
Perth	*Victoria Leagues, 276 Onslow Road,*	*Thu*	*2 July*	*Tutorial A*	
	Shenton Park WA 6008	*Fri*	*3 July*	*Tutorial B*	
Hobart	*University of Tasmania, Computer Science ,*	*Mon*	*6 July*	*Tutorial A*	
	Rm C219, Centennary Building, Sandy Bay	*Tue*	*7 July*	*Tutorial B*	
Melbourne	*International House, 241 Royal Parade,*	*Thu*	*9 July*	*Tutorial A*	
	Parkville VIC 3052	*Fri*	*10 July*	*Tutorial B*	
Canberrra	*Australian National University,*	*Mon*	*13 July*	*Tutorial A*	
	Rm 101, John Dedman Building	*Tue*	*14 July*	*Tutorial B*	
Sydney	*Wesley Centre, 220 Pitt Street,*	*Wed*	*15 July*	*Tutorial A*	
	Sydney NSW 2000	*Thu*	*16 July*	*Tutorial B*	
Brisbane	*QUT, Rm 771, Library Building,*	*Mon*	*20 July*	*Tutorial A*	
	2 George Street, Brisbane	*Tue*	*21 July*	*Tutorial B*	

Section C: REGISTRATION FEES

	Charge to attend 1 Tutorial	*Charge to attend 2 Tutorials*	*Amount Payable*
AUUG Members	$300	$600	
AUUG Student Members	$150	$300	
Non Members	$450	$900	

Section D: AUUG MEMBERSHIP

I wish to become a member of AUUG and enclose payment for:

Individual Membership	@ $100	
Student Membership (full time students only)	@ $25	

TOTAL AMOUNT PAYABLE $ _____

Please mail completed form with payment to:

AUUG Inc
PO Box 366
KENSINGTON NSW 2033

Ph: 1-800 625655

Or Fax to:

AUUG Inc
(02) 9332 4066

Cancellation Policy

Cancellations received in writing prior to 19 June will receive a complete refund.

Cancellations received in writing after 19 June will forfeit 50% of registration fees. No refunds however will be allowed 3 days prior to each event, however substitutions will be accepted.

Section E: PAYMENT

Cheques to be made payable to AUUG Inc
- OR -

☐ Please debit my credit for A$_____

☐ Bankcard ☐ Visa

☐ Mastercard

Card Number _____

Name on Card _____

Expiry Date _____

Signature _____

Information Disclaimer

The speakers, topics and times are correct at the time of publishing and in the event of unforeseen circumstances, AUUG Inc reserves the right to alter or delete items from the Conference Programme.

Each Delegate named herein, or their later substitute, accepts that AUUG Inc, their agents, speakers, chairmen, secretariat or any other persons ("the parties") involved in the preparation of this pamphlet or the planning or presentation of the conference, do not accept any contractual, tortious or other form of liability for losses or damage suffered by the delegate or their later substitute relying on any statement representation advice or opinion (whether true or otherwise, written or oral) and whether due to the negligence of any of the said parties by this disclaimer of liability to exclude liability, if any, for such statement representation advice or opinion, and that said party may rely on the disclaimer of liability in the event of any demand claim suit or action brought against any or all of them arising out of any statement representation advice or opinion. The authors have prepared this material for Professional Development purposes. Although they trust that it will be useful for this purpose, neither the authors nor AUUG Inc can warrant that the use of this material, would be adequate to discharge the legal or professional liability of members in the conduct of their practices.

AUUG Inc reserves the right to cancellation of the event

Internet Payment Systems

Peter Billam <Peter.Billam@marine.csiro.au>

[Editor's Note: This talk by Peter Billam of P J B Computing was presented to the summer conferences of the Canberra and Tasmanian branches of AUUG in February 1998.]

ABSTRACT

This talk surveys some of the Digital Payment systems most adapted to life on the Internet. At their best they can benefit organisations large and small, offering them payment systems with the advantages of the Internet itself - global reach, high speed, low transaction cost and high automatability. Some of the privacy and security risks are mentioned. Particular emphasis is placed on the Australian context. The technologies discussed here could revolutionise the commercial and financial systems of the world.

PHYSICAL PAYMENT SYSTEMS

Cash

The most sophisticated and efficient payment system, offered by governments to avoid circulating gold. Transfer is instant and 100 per cent efficient. No transaction record is created. Cash is such a tempting target for theft that it is unsafe to keep large sums in cash, or to send cash by post.

Cheque

Banks offer safekeeping for your cash, protecting you from theft; they grant you access to your money on your signature. A cheque, a signed instruction to pay, can be sent by post, offering global range. The payee can be anybody, not just a business. The bank retains a record of the amount of the transaction, but not of what item was purchased.

For the payer, cheques are slow to write. For the payee, they can bounce or be cancelled, and take several days to clear.

Credit Card

The payee must be a business. The card carries a raised, embossed number. As originally introduced, the merchant puts the card through a roller which reads the number onto a slip of carbon paper, and the customer authorises the payment by signature. The payment cannot bounce or be cancelled, with the bank assuming the risk, and charging the merchant several per cent accordingly.

Mail-order merchants may ask their bank to be trusted to receive payments without any signed authorisation; the merchant just quotes a card number and an amount and the bank just believes them. The customer is responsible for checking their monthly account and complaining to their bank about payments they don't remember.

DIGITAL PAYMENT SYSTEMS

Credit Cards on the Internet

The Merchant gets to see thousands of live credit card numbers, and is under suspicion every time there is fraud on any of them. The Consumer's money is spent without any say-so from the consumer. The Bank guarantees the transaction and thus incurs significant risks; and charges accordingly.

First Virtual's PIN System

This is a very elegant, well conceived, low-tech system, built on top of the Credit Card infrastructure. It avoids card numbers ever being sent over the Internet or disclosed to merchants, and it allows the purchaser to confirm the payment. The purchaser must be reachable by e-mail. Amazingly, it uses no encryption, so it has no problems with the U.S. munition export laws, and can be used by customers in countries such as France and Iraq.

The customer gives their card number to the First Virtual Bank by phoning up a particular number and typing it into a touch phone. In return they are assigned a PIN password. The merchant must be registered with First Virtual, and must have a bank account able to accept payments by the ACH (Automated Clearing House) system; that is to say, U.S. bank account.

When the customer makes an order, they give the merchant their PIN password. The merchant then contacts First Virtual, quotes them the PIN and asks for the money. First Virtual send the customer an e-mail asking for their OK. The customer replies either "Yes", "No" or "Fraud", and if the reply is "Yes" the transaction goes through.

ACH

Merchants and consumers in the U.S. may gain direct access to the Automated Clearing House system used to transfer money between banks. CheckFree of Ohio interfaces with PC financial packages such as Quicken to allow consumers to make payments, and CheckFree's Gateway system allows U.S. merchants direct access to the ACH, over the Internet using PGP, for 27 cents per payment.

DigiCash

Developed by Dr David Chaum, sold by DigiCash BV in Amsterdam. The consumer downloads the

DigiCash software to run a digital wallet, opens an account with the local mint. The mint could be run by a government or a bank; DigiCash BV is in the process of signing up numerous banks to run mints (this is reminiscent of the situation in Australia last century where banks issued their own banknotes).

The user creates some "coins" and gets them signed by the mint. The wallet can exchange coins with other wallets using a custom IP protocol; coins can also be sent in text form by e-mail or other means. When desired, they can be cashed in again at the mint.

The payer knows the identity of the payee, but the payee does not find out the identity of the payer (unless the payer attempts to double-spend a coin).

CyberCash / CyberCoin

CyberCash is a system which uses public-key cryptography to leverage credit cards onto the Internet, and CyberCoin is an extension of CyberCash to allow small-value transactions.

The consumer downloads the CyberCash digital wallet software, and enrols their credit card with the wallet, and with CyberCash; they may also open a CyberCoin account and move some money into it. The wallet registers itself as a helper application for Netscape or Internet Explorer.

When the consumer approves a transaction, an encrypted payment order is sent to the merchant, who adds some payment information, signs the order, and forwards it to the CyberCash gateway. The merchant never sees the consumer's credit card number.

SET

The Secure Electronic Transaction protocol is being developed by MasterCard, Visa and various computer companies, in order to transmit payment information over the Internet. It can not be used to encrypt other messages, and so the U.S. State Department has deigned to grant export permission to some SET implementations. It is hoped that SET will eventually be built into many "commercial products". Merchants (and in Mastercard's implementation also consumers) must have digital certificates signed by their banks.

Functionally, SET works in a similar way to CyberCash, except that the acquiring bank can, at its option, inform the merchant of the card number when it sends. Thus SET does not necessarily improve the customer's security much, as compared with sending the card number in plain text.

Smart Cards

A smart card is like a credit card equipped with a CPU. It can store lots of information, can be password-protected, and can even run an RSA encryption engine.

Smart cards have been used for years in European telephones, Mondex uses a smart card, and Visa introduced their Visa Cash Card for the Atlanta Olympics.

Mondex

Modex is not an Internet payment system, but it is quite widespread; it is a closed proprietary system involving smart cards which communicate using a secret protocol.

The consumer "refills" their card at a specially equipped ATM machine, and purchases can be made by inserting the card into a "Mondex wallet" or by using a proprietary telephone.

Mondex is used in a pilot project in Swindon, England, and campus-wide at the universities of Exeter and York. There have been trials in Hong Kong, Canada, and San Francisco. In November 1996, MasterCard International purchased 51 per cent of Mondex.

Telstra's SureLink

In the Australian context, Telstra's SureLink, has been operational since October. It is not a minimal system; it has a lot of added value, and is aimed only at Internet Commerce.

Telstra have bundled a link to the EFTPOS payment infrastructure, which is well established in Australia and has low cost per transaction, together with a shopping cart application, into a package which is very convenient for the merchant. There is built-in support for hardgoods, softgoods, and subscriptions to softgoods. Customers can be anywhere, but the merchant must bank in Australia.

The merchant has to run a web site in which they make "Digital Offers" which are URL's and look like

```
Unlock Professional features in Shareware
version:
<a
href=http://payment.eps.com.au:80/bin/paymen
t.cgi
?beef3e92e313ef8ed2e4dabcc9776cd4:
kid=100086.100168&valid=8104227285&domain=mi
key
&desc=Management%20Info%20Pro%20Key&expire=2
592000
&ss=env&cc=AU&goodstype=i&amt=15.95&fmt=int
&url=http%3A%2F%2Fwww.swanhill.com.au%2Fstor
e100086%Fmikey>
<img    src="Key.gif"    border=0    width=38
height=34></a>
```

The bit in bold is crucial; it's a checksum which hashes the rest of the URL together with a secret key particular to that merchant. Digital Offers are binding offers, and the checksum is what prevents a customer

from changing some details of the offer, such as the price, prior to accepting it. The secret key is changed every month by a Keymaster, who needs superuser access to the web server.

The Digital Offer URLs are generated by Open Market Secure Link Executive, a commercial package, which must run on the web server; for example as a Server-Side Include. The Server-Side Includes are put together for the merchant by a SureLink Business Partner.

Australian Business Access

Australian merchants wishing to run their own shopping cart scripts, or selling more expensive goods, might prefer ABA's Epayment, which uses strong encryption to keep the card number away from the merchant. The merchant is charged a connection fee of $2500, and thereafter pays just a flat charge of 90 cents per transaction. ABA connect directly to the Australian clearing house system. The customer needs a Java-capable browser, preferably Java 1.1.

WHAT IT COSTS YOU

This uses SureLink as an example . . .

Cost	Item	Beneficiary
$100 for 3 years	Business Name	State Government
$100	Setup Fee	Merchant Acquirer
$20 per month	Running Fee	Merchant Acquirer
5 per cent ($4/month min)	Transaction Fee	Merchant Acquirer
3 per cent ($5/month min)	Transaction Fee	Telstra

There will always be places in every country where you can't run a business without having to pay some of your takings to some effective and established local organisation. The Internet in Australia is one of those places; I'm not sure how 8% of every transaction measures up, on a world scale . . .

There's a sense in which every country **is** such an organisation; their governments raise taxes and, in return, provide currencies, payment mechanisms, infrastructure, services and so on; so having to pay someone a percentage of your takings in order to operate is not inherently unacceptable. But governments tax only a narrow range of transaction types, (such as salary payments from employer to employee), transactions for which they can force accurate reporting. Also, governments tax profits, not takings, and they provide more services for the money than banks do.

Technology	Authentication	Reach	Speed	Inefficiency	Provider
Cash	Personal	1 metre	Instant	0 per cent !	National Governments
Cheque	Signature	National/Global	Several Days	Bank Fees	Retail Bank
Credit Card	Personal	Global	Minutes	4 per cent	Merchant Acquirer
ACH	?	USA	Minutes	27 cents	ACH
Virtual PIN	Personal	Global	Minutes	4 per cent	Merchant Acquirer, Telstra
DigiCash	Private Key	Global	Minutes	?	Government or Bank
SureLink	Personal	Global	Minutes	8 per cent	Merchant Acquirer, Telstra

Table: Comparison of Payment Systems

A machine with an 8% loss, in comparison with a machine with 0% loss, is, quite objectively, bad engineering.

CONCLUSIONS

In most contexts, the Internet offers particularly efficient mechanism. If you ftp a file of 3 Mb, you'll be disappointed if even a single byte does not arrive. It's saddening that Internet Payment Systems are much less efficient, down to 92 per cent, than their low-tech counterparts.

There is no technical reason why a 100% efficient Internet Payment System could not be provided at the national level, and one day, some government, perhaps under pressure from its own merchants, may do this. It could take the form of a giro-like system with a publicly accessible IP interface, using PGP or ssh to sign instructions. This would benefit local population and businesses.

The Europe of the future, with its large single currency, and strong national giro tradition, would be well placed to introduce efficient payment mechanisms and develop a more vigorous internal Internet trade.

Depending on the system's policy on privacy, it could also offer government very complete reporting of a much larger class of financial transactions, information which is currently given to private interests who use it for market research.

Governments could use it to widen their choice of tax base, a choice which they could then use as a lever to put policy into effect by differential taxing, rather than just by forbidding things or making them compulsory.

Internet commerces from other currency zones would find it in their interests to open local subsidiaries, on local web servers, so as to gain efficient payment which they could then repatriate later at a time of their own choosing, in larger amounts with lower overheads.

AUTHENTICATION

At basis, purchasers are known to the Financial System by their signatures, on paper, and by being able to show certain documents that no-one else is supposed to have. This means you need a shopfront to witness the purchaser sign, and to view the documents. The banks provide this shopfront, and the purchaser can choose from various schemes that allow them to leverage their signature into some other more convenient authentication mechanism, such as swipe card and pin number.

Hypothetically, purchasers could permitted to identify themselves to the financial system by some electronic means, involving strong cryptography. In this case, purchasers, indeed residents in all situations, might just as well plug strait into the Clearing House mechanism, and be able to make payments to whom they wanted, with very low overheads, perhaps even as low as cash.

It's worth noting that having an efficient digital payment system would bring us back to the situation we have with cash, where you can lose all your life savings in a simple breaking and entry job. The intruders just have to persuade you to give them your PIN number or PGP pass phrase, and a lot of ugly scenes could be caused that way.

Banks would then revert to their core business, that of keeping money safe, and undertaking to give it back to you on corporal authentification, such as iris scan, DNA, fingerprint scan, or even the old signature on paper.

CURRENTLY . . .

At current prices, Internet Payment Systems do not offer a general-purpose method of transferring money. In many cases, including Telstra's SureLink, the recipient of the payment can only be a merchant, person-to-person payments are not supported; only Internet Commerce is supported.

Currently, Internet Commerce applies primarily to niche markets:

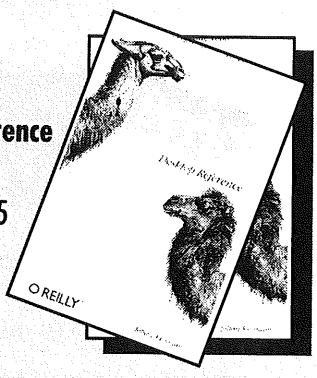
- Where very broad stock is a competitive advantage (CD's, books, musical scores) global megastores may be favoured, or search engines for co-operating chains of smaller stores that are technically able to share their databases.
- Specialised product, where the customers are spread all over the world, but there are very few of them per square kilometre, so they can't be effectively reached by a shop-front. This has the potential to open up for trade a whole range of new connaisseur products which hitherto you simply have not been able to trade in any cost-effective way.
- Electronic Product, such as newspaper searches, legal or medical searches, books distributed in Postscript form, etc.

❖

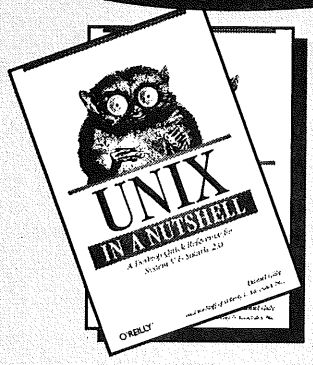
REFERENCES

- Achieving Electronic Privacy, David Chaum, Scientific American, August 1992, pp76-81
- Computer Money: a systematic overview of electronic payment systems, A Furche and G Wrightson, Dpunkt Verlag,
- 1996
- Digital Cash, Alan Tyree, Butterworths, 1997
- Web Commerce and Internet Security, Simson Garfinkel and Gene Spafford, O'Reilly, 1997
- SureLink Business Partner Manual, Telstra, 1997
- http://www.pjb.com.au/comp/auugtalk.html

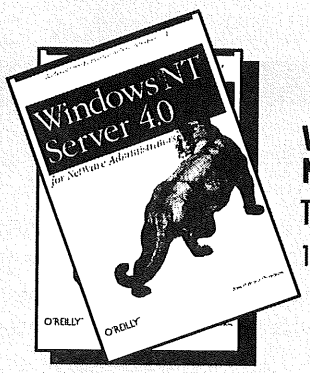
O'REILLY ANIMAL MAGNETISM



Perl 5 Desktop Reference
Vromans ORE
1565921879 14.95

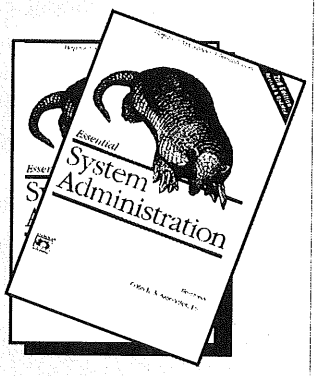


UNIX in a Nutshell System V & Solaris 2 2/e
Gilly ORE
1565920015 39.95

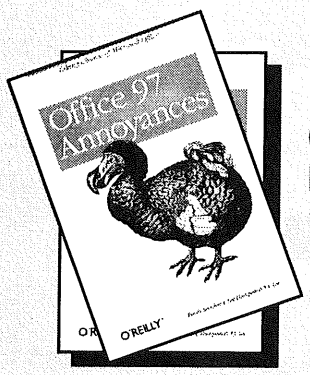


Windows NT Server 4.0 for NetWare Administrators
Thompson ORE
1565922808 79.95

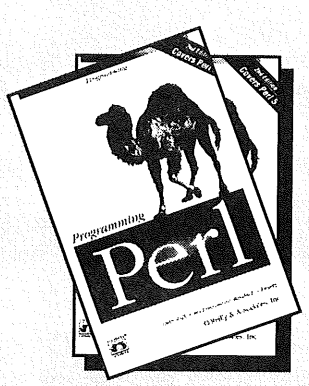
Java in a Nutshell 2/e
Flanagen ORE
156592262X 39.95



Essential System Administration 2/e
Frisch ORE
1565921275 65.00

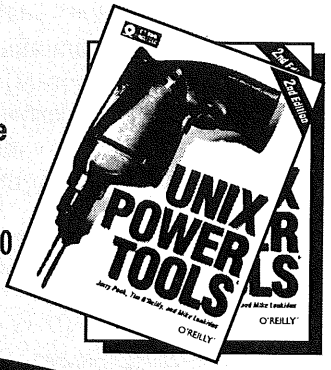


Office 97 Annoyances
Leonhard ORE
1565923103 44.95

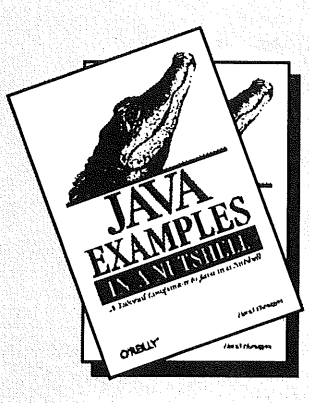


Programming Perl 2/e
Wall ORE
1565921496 79.95

TCP/IP Network Administration 2/e
Hunt ORE
1565923227 65.00



Unix Power Tools 2/e (Bk/CD)
Peek ORE
1565922603 120.00



Java Examples in a Nutshell
Flanagen ORE
1565923715 39.95

recommended retail price - note price and availability subject to change without notice

Save up to 60% off computer and business books
at http://www.bargainbooks.com.au

WOODSLANE

Available from all good bookstores
or direct from Woodslane on
Tel: (02) 9970 5111 Fax: (02) 9970 5002

O'REILLY™

The 1998 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 has played in bringing UNIX to Australia, the formation of AUUG, and the promotion of the values held by the open systems community.

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 31st July 1998

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

John Lions Award - 1997 Winner

The winner of the John Lions Student Award for Open Systems for 1997 was Jerry Vochteloo, a Phd student at the University of New South Wales, who won the award for his work related to the protection mechanism of the research operating system Mungi.

It is significant that a student from the University of NSW won the award which was established to recognise and honour Associate Professor John Lions from the same university.

"It is encouraging to see systems type research rewarded in Australia," said Jerry Vochteloo. "My involvement with John Lions and my association with the University of New South Wales have added a great personal significance to being the first recipient of the award."

The award was presented by John Lions, wife, Marianne, as Lions was unable to attend due to ill health.

[Left to Right: Marianne Lions, Jerry Vochteloo and Lucy Chubb]

Bargain Books

http://www.bargainbooks.com.au

Bargain books is where you will find a wide range of Computer, Business and Trade books that have been returned from a retail outlet, hurt in transit, or are publishers' overstocks. As they have been through the sales cycle once, we can offer them to you at incredible savings off the normal recommended retail price.

How it Works

Bargain Books operates much like a sale does. Every Friday morning, we load up the week's stock and those who log on and order first, get the best deals.

Special Offer

Every week, we have a series of great specials where you can save up to 60% off the RRP. See what specials are on this week!

Bargain Bin

If you're on a budget and only want to buy books within a certain price range, visit our unique Bargain Bin. You can search for books based on price and only see books that fit your budget. Start searching for bargains now!

Finding Books

We've tried to make it as easy as possible for you to find the books you're looking for, or might be interested in. Here's how:

- Fast Find - select a category, type in your request and click on "Find It".
- Searching - Search by title, author, imprint or series, publisher or ISBN number.
- Browsing - you can find books by subject or imprint/series.
- Categories - you can also jump to specific categories of books by using the menu on the bottom left. Keep Me Informed

Updates

If you want us to email you when books are added to particular subject categories of interest, register with Keep Me Informed.

http://www.bargainbooks.com.au/keep_informed.asp

More Information

Got a question? If so, ask us by filling in the form below. Make sure you include your details so we can respond to your query.

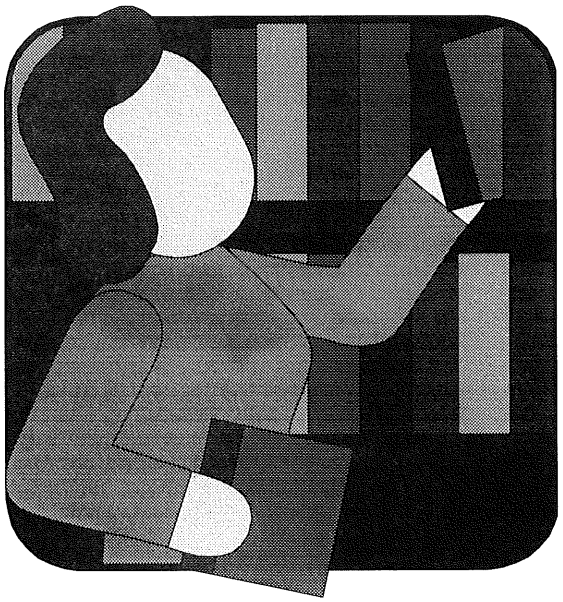
http://www.bargainbooks.com.au/contact_us.asp

Contact

Nichola Walker
Email: nicholawa@woodslane.com.au
Tel: 9970 5111
Fax: 9970 5002

Nathan Moyes
Email: nathanmo@woodslane.com.au
Tel: 9970 5111
Fax: 9970 5002

❖



Understanding IPV6

Michael Paddon <Michael.Paddon@auug.org.au>

INTRODUCTION

IPV6, or Internet Protocol Version 6, is the next generation of the fundamental networking protocol that powers the Internet. Version 6 is a significant and, in some ways radical, departure from the current state of the art, driven by a need for more capacity and higher performance than ever foreseen by the original designers.

As the basis of the Internet in the 21st century, an understanding of this technology is imperative today, especially in regards to its capabilities, limitations and design tradeoffs. Equally important is the issue of how we upgrade our own small pieces of the global network to participate in IPV6 smoothly and without significant loss of service.

The purpose of this paper is to provide the reader, who may never have heard of IPV6 before today, with such an understanding. It will equip you with an understanding of IPV6, from why it exists, through to how it works, and on to how the transition to the brave new world may be managed.

I do assume that the you understand the basics of an internetworking protocol. This is a technical paper, and it would be impossible to cover the required ground starting from first principles. In particular, some familiarity with IPV4 (the current version of IP), especially with the way addressing and routing works in the Internet today will help you get the most out of this paper.

WHY DO WE NEED A NEW PROTOCOL?

This is the most important question this paper will pose.

Changing from one protocol to another is always a difficult and expensive task, and when the affected network spans the globe and contains millions of nodes the issue is even more contentious.

New functionality, alone, is not enough to drive this change on a worldwide basis. Why should a use, content with the services provided by IPV4, undergo the pain and trouble of switching over when they stand to gain very little?

However, there are two major problems looming over the Internet as we know it today, each of which mandates a major change to the way the Internet Protocol must operate.

The Address Space Crisis

Although, the IPV4 address space can theoretically support over 4 billion hosts, by the early 1990's it was becoming clear that the effective utilisation of this space was much less. In other words, most addresses were (and still are) being wasted because of the hierarchical structure imposed by the routing system.

For example, to connect a single machine, I am usually assigned a class C address which can theoretically support 254 hosts. The result is an instant waste of 253 addresses, unless I should decide to by more machines. Even worse, many organisations have been allocated class B addresses (2^{16}) addresses, and are utilising them extremely poorly.

The degenerate case, of course, the class A addressing region which ties up a significant portion of the Internet's addresses in blocks of effectively unusable size.

Current estimates for the exhaustion of the IPV4 address space range from the year 2008 to 2018, depending on how optimistic one's outlook is [Bradner95]

The Routing Crisis

At the same time that addresses have been wasted by the hierarchical structure imposed on them, another problem has been caused by there not being enough structure.

The Internet backbone must have some idea where every network in the world is, so that packets can be forwarded to their destination. Unfortunately, since any IPV4 network (be it class A, B or C) can be located anywhere in the world, routers must maintain a record for every active network.

This has lead to a rapid explosion in the size of routing tables in the "core gateways", with the obvious concern that we were well on the way to exhausting the maximum table capacity of these devices. There was also a related explosion of routing information traffic.

There is no easy answer to this problem given the IPV4 address design, however, a scheme called CLIR (Classless Interdomain Routing has been implemented as a stop-gap measure to minimise this problem until a better solution can be rolled out. CLIR coalesces related routing records by issuing network numbers in blocks, and effectively destroys the concept of address classes.

A Solution

The solution to these problems is a much larger address space, allowing for more addressing structure, less stringent allocation polices and more efficient routing.

Since the new IP is required to support a larger address space it makes sense to provide as much new functionality and performance as possible, given that such a protocol switch will probably only happen once a generation. It is also the perfect opportunity to feed the last 20 years of operational experience back into the design process; to remove features that didn't work out and to refine those that did.

It is also important to realise that IPV4 simply does not properly support some of the things that people want to do routinely today, for example:

- Mobile computing.
- Automatic configuration.
- Real time video and audio.

Without addressing these needs, the Internet would run the risk of becoming ever less applicable and relevant to the modern computing environment.

IPV6 was born from a desire to solve today's problems, provide the networking substrate required by tommorow's applications, and guarantee the continued health and growth of the Internet.

THE STANDARDS PROCESS

IPV6 is the result of a long process, as documented in [Bradner95]. Starting in late 1991, several groups under the auspices of the IETF (Internet Engineering Taskforce) and the IAB (Internet Architecture Board) began investigating both short and long term solutions to the Internet's growth.

These various efforts did not complement each other well, with significant disagreement over the adoption of ISO/OSI protocols being a major sticking point.

By 1993, it was clear that a major re-evaluation of requirements for the next generation of IP was needed before candidate protocols could be designed. This lead to an RFC calling for general input into the requirements process [RFC-1550]. Twenty one papers were received in response, providing input from areas as diverse as combat systems to cable TV.

Based on these requirements, a list of criteria for a new Internet Protocol was published and by late 1992 there were several proposed solutions: "CNAT", "IP Encaps", "Nimrod", "Simple CLNP", "SIP", "PIP", "TP/IX".

By late 1993, several of these had merged and evolved into new proposals, ready for formal evaluation: "TUBA", "SIPP" and "CATNIP". This evaluation found SIPP to satisfy the criteria most successfully. However, some weaknesses were identified (including the need to increase address size from 64 to 128 bits!) and some features TUBA were identified as being highly desirable. Therefore, a modified SIPP proposal was produced by mid 1994.

In July 1994, IPng area directors recommended that the modified SIPP proposal be accepted as the basis of IPV6, and an IPng working group was formed.

At the time of writing, more than fifty of the key IPV6 standards have been formalised as RFC documents, or as Internet Drafts which will become RFC's over time.

THE NEW PROTOCOL

IPV6 contains significant new functionality in addition to expanded addressing and routing capabilities.

Expanded Addressing:

- IPV6 address are 128 bits long, allowing for more nodes, more routing structure and and simpler autoconfiguration (e.g., one can embed 48 bit ethernet station id's as the host part of an IPV6 address).
- Multicast addressing has been retained and refined and a new type of address, the anycast, has been defined to support new semantics.

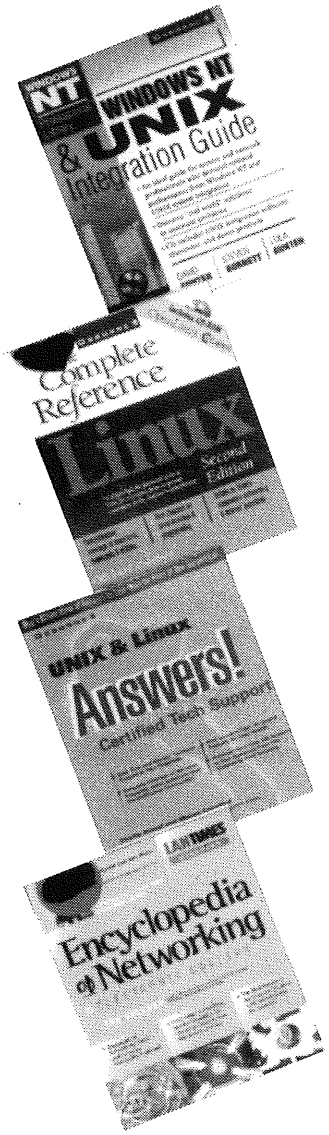
Scalable Routing:

- IPV6 addresses allow a far more flexible interpretation of what is the network part of the address. Thus, many levels of routing structure may be defined and routing tables can be far more effectively distributed.
- The IPV6 routing option provides for a mixture of "loose" and "strict" source routing in a single packet. "Loose" routing defines a path of nodes that must be traversed in order to reach the destination, with perhaps other unmentioned nodes being traversed between these points. "Strict" routing defines an exact path for the packet to follow, with unmentioned hops being illegal.
- Scope has been added to multicast addresses to provide more efficiency and scalability to the routing of multicast groups.

Better Options Support:

- IPV6 is designed to an easily expanded set of variable length options. It also provides for the efficient forwarding of packets when one or more such options are present.
- In addition, the IPV6 header is far more streamlined that that of IPV4, with many optional parts either relocated out of the header or dropped entirely.

UNIX/Linux Titles from McGraw-Hill



Windows NT and UNIX Integration Guide
David Gunter et al RRP$99.95 **AUUG$79.95**
Shows that optimal performance is possible from Windows NT and Unix system integration with the right tools. Provides real-world solutions to everyday problems from file services to remote access. The additional CD includes Unix integration software, shareware, and demo products.

SPECIAL
Save 20%
if you order
through AUUG

Linux: The Complete Reference 2/e
Richard Petersen RRP$89.95 **AUUG$71.95**
Absolutely everything you need to know about Linux!
From detailed instructions on installing Caldera's OpenLinux to troubleshooting references, *Linux: The Complete Reference* is your one-stop guide for realizing the full power and flexibility of this popular and often complicated-UNIX clone.

UNIX and Linux Answers!
Charlie Russel and Sharon Crawford RRP$44.95 **AUUG$35.95**
Providing more than 400 answers ranging from file systems to shell scripting, *UNIX and Linux Answers!* offers complete coverage of the most frequently asked questions from the UNIX/Linux community.

Encyclopedia of Networking, Electronic Edition
Tom Sheldon RRP$89.95 **AUUG$71.95**
Encyclopedia of Networking is the most complete guide to the vast world of networking available. In true encyclopedic form, this book and CD-ROM package gives you a concise introduction to every facet of networking and its technologies, followed by all the information you need to continue your research.

Also available:
Practical Performance Analyst by Gunther as featured in last months AUUG Journal

Mail or fax your order to (02) 9417 7003 today for faster processing.

Please print clearly:

Name:_____ Phone:Bus/Home: ( )_____

Address:_____

_____Postcode:_____

I enclose payment now by: ☐ Cheque ☐ Bankcard ☐ American Express ☐ Mastercard ☐ Visa ☐ Diners Club

Card number: Expiry Date: Signature:

Expiry Date: | | | | |

Signature: _____

(Please note: If you pay by credit card, your credit card will not be debited until 21 days after your invoice date).

Qty	ISBN	TITLE	RRP$	AUUG $
	0.07.882395.1	Windows NT and UNIX Intregration Guide - s/c Gunter	99.95	79.95
	0.07.882461.3	Linux: The Complete Reference - s/c Petersen	89.95	71.95
	0.07.882446.X	UNIX and Linux Answers! - s/c Russel	44.95	35.95
	0.07.882333.1	Encyclopedia of Networking - s/c Sheldon	89.95	71.95
	0.07.912946.3	Practical Performance Analyst - s/c Gunther	168.95	90.00
			Subtotal	
			Add $6.00 Postage & Handling	$6.00
			TOTAL	

RRP=Recommended Retail Price. Prices and availability are subject to change without notice

Please make sure you have enclosed payment with your order. If you fax your order please do not post your order also.

McGraw-Hill Book Company Australia Pty Ltd A.C.N. 000 473 674
4 Barcoo Street, Roseville NSW 2069. Tel: (02) 9417 4288 Fax: (02) 9417 7003
e-mail: profref@mcgraw-hill.com.au

AUUG398

Better Quality of Service Support:

- IPV6 defines the concept of a "flow" which identifies a packet as part of a ongoing stream of data. This allows a single set of quality-of-service parameters to be applied to an entire session.

Authentication and Security:

- IPV6 supports both authentication and encryption at the internetworking layer.

ADDRESSING ARCHITECTURE

The greatest challenge in the IPV6 design was to get the addressing scheme right, building on the experiences garnered by the Internet community to date mixed with a little bit of crystal ball gazing.

IPV6 defines addresses to be 128 bits in length, compared to the 32 bits available in IPV4. This is 2^{96} times the current address space and, as [Bradner95] points out, theoretically contains 6.7×10^{22} unique addresses for every square metre on the surface of our planet.

Of course, adding hierarchical structure to the address space significantly reduces its effective maximum utilisation. Nevertheless, even pessimistic assumptions suggest that 1.5×10^{3} usable addresses for every square metre is an achievable outcome.

It is my opinion that 128 bits will amply address the needs of a global network for the foreseeable future. A 64 bit space would have been cheaper in terms of bandwidth overhead, but would almost certainly have required continuing address rationing policies and would have reduced the opportunity for additional layers to be added to the routing hierarchy.

Looking forward, 128 bits may not be large enough for an interplanetary Internet, and it certainly won't meet the needs of an interstellar one. Similar problems arise if micro- or nano- machines require universal Internet addressability. These scenarios, thankfully, are well outside the scope of this paper.

Address Types

There are three types of address defined by IPV6:

- Unicast
- Anycast
- Multicast

The broadcast addresses of IPV4 have been completely superceded by IPV6's multicast functionality. A corollary of this is that there are no longer any magic "all one's" and "all zero's" host numbers to indicate MAC-layer broadcasting.

Unicast Addresses

At least one unique unicast address is assigned to every network interface that requires addressability. Interfaces may remain completely anonymous if they never originate or act as packet destinations, e.g. a point to point link between two routers.

No concept of class A, B, C etc. addresses has been carried over into IPV6. The structural components of a unicast address are specified by contiguous bitmasks, not unlike the CLIR mechanism except allowing multiple levels of hierarchy. Routers at different levels in the routing hierarchy may apply different masks, allowing organisations to define arbitrary structure upon their networks.

Several forms of unicast address are currently defined:

Provider-based Addresses:

- Provider based addresses provide an addressing structure based on who is providing connectivity to the interface in question. It is currently proposed that a hierarchy of:

  < registry, provider, subscriber, subnet, interface >

  be conventionally defined within these addresses, with each field being of variable length.

Geographic-based Addresses:

- Geographic based addresses work on a similar principle, excepting that they reflect the physical location of an interface rather than the location relative to a provider.

NSAP and IPX Addresses:

- These addresses support several marginally used protocols.

Site Local and Link Local Addresses:

- The site local and link local schemes provide addressability that is limited in scope, and not required to be globally unique. Routers do not pass packets containing these addresses outside the appropriate domain of validity. Link local addresses are useful for auto-address configuration and neighbour discovery, while site local addresses are often applied to an unconnected (or firewalled) network.

One area of the unicast space is reserved for special addresses, that deserve special mention:

IPV4 Addresses:

- One very important addressing form that is supported by IPV6 is that of embedded IPV4 addresses. In fact there are two forms of IPV4 address representation:

  - IPV4 compatible address.

- IPV4 mapped address.

- The compatible address type is used for nodes that retain an IPV4 address but that support the IPV6 protocol. IPV4 mapped addresses indicate that the node does not support IPV6, and hence the packet must be delivered by an intermediary agent which can perform the appropriate protocol translation.
- This feature is obviously a key element of the IPV4 to IPV6 transition mechanism.

Unspecified Address:

- The address 0:0:0:0:0:0:0:0 is defined to be the unspecified address and should never be assigned to an interface. IPV6 addresses are conventionally written as 8 16-bit hexadecimal numbers, separated by colons. For example, the IPV6 address of lynx.aba.net.au is 0:0:0:0:0:FFFF:CB15:5401. There are many variations on this scheme, as specified in [RFC-1884].

Loopback Address:

- The address 0:0:0:0:0:0:0:1 is the loopback address.

Anycast Addresses

An anycast is an address that is assigned simultaneously to multiple interfaces. Packets sent to an anycast destination will be delivered to the nearest one (in the routing sense).

Anycast addresses are assigned out of the unicast address space, making them syntactically indistinguishable from the latter. An address becomes anycast the moment it is assigned to more than one interface, and it is necessary for interfaces to be told that they are part of an anycast group.

The routing of anycast addresses is complex, requiring a separate host route to be advertised for each member of the group. When all the host routes share addresses with a common prefix, the propagation of this routing information may be circumscribed. However, when there is little or no locality shared by members of the group, the propagation of these routes must be advertised globally which leads to scaling problems.

Until substantial field experience has been gathered, anycast addresses may not be used as the source address of a packet and may only be assigned to router interfaces. They are expected to be useful for such scenarios as addressing a service provider's routers (you don't care which one you get to, so long as the packet goes to that cloud).

Multicast Addresses

A multicast address identifies a group of nodes. When a packet is sent to a multicast destination, it is routed to each of the members of the multicast group.

Depending on which part of the multicast address space is used, a multicast group will have a specific scope:

- Node local.
- Link local.
- Site local.
- Organisation local.
- Global.

There is room in the addressing scheme for further scopes to be defined.

Some predefined multicast addresses have been defined within most scopes, to address common groups such as "all nodes", "all IPV6 nodes", "all routers", "all NTP servers", etc.

Multiple Homing

Most IPV6 network interfaces will be multiply homed, having several unicast addresses (typcially a loopback address, a link local address and a provider based address), and several multicast addresses.

In general, there is no limitation to the number of addresses an interface can represent.

Address Allocation

The IPV6 address space is divided into regions set aside for each of the above address types. By examing the leading binary digits of any address, it is straightforward to identify its type, as shown in the table below.

Approximately 15% of the address space has currently been allocated. IP Address allocation is as follows:

Allocation	Prefix (base 2)	Fraction of Address Space
Reserved (including IPV4)	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP	0000 001	1/128
Reserved for IPX	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Unassigned	001	1/8
Provider-Based Unicast Addresses	010	1/8
Unassigned	011	1/8
Geographic-	100	1/8

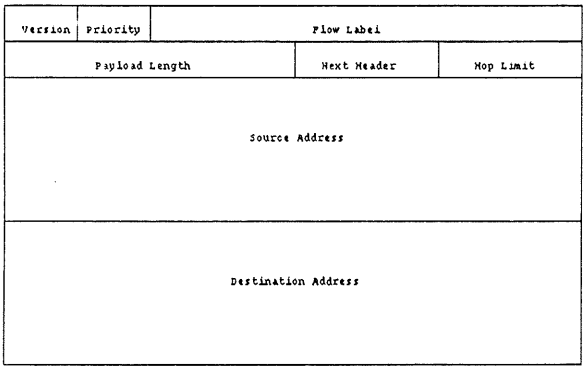
Based Unicast Addresses		
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link Local Use Addresses	1111 1110 10	1/1024
Site Local Use Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

PACKET FORMAT

An IPV6 header is significantly simpler than the ones found on IPV4 datagrams. The rationale for this is twofold:

- A simplified header makes it cheaper and faster to process a packet through an intermediate gateway.
- The bandwidth overhead of 128 bit addresses is mitigated by reducing the rest of the header payload.

The IPV6 header format, depicted below, is fully specified in [RFC-1883]. The header consists of several fields:



Version (4 bits):

- The version number of the protocol, 6 in this instance.

Priority (4 bits):

- The priority of the packet. There are two bands of priority: congestion-controlled and non-congestion-controlled. The latter is used for real time services such as video and audio.
- The congestion-controlled priorities range through filler traffic, unattended data transfer, attended data transfer, interactive traffic to control traffic. The non-congestion-controlled priorities provide an 8 value scale from least willing to discard to most willing to discard.

Flow Label (24 bit):

- IPV6 introduces the concept of a flow identifier which marks packets as belonging to a stream of data requiring special handling by routers. For example a stream of video data requiring a real time type of service would be assigned a flow identifier by the source node.
- Information about the special requirements of a flow may be communicated to routers via extension headers, or by completely separate protocols (as might be required for operations such as resource reservation). These mechanisms have not yet been well defined. Neither is the lifetime of a flow label well defined, however the minimum time between flow label reuse is 6 seconds.
- Finally, it is useful to note that flows can be used by routers to cache the results of processing extension headers such as hop-by-hop, to speed up the processing of subsequent packets.
- A flow label of zero implies no defined flow, which is expected to be the common case.

Payload Length (16 bits):

- The length of the payload (ie.the packet length, excluding the header). If the payload length is zero, the length is carried as a jumbo payload option in the hop-by-hop extension header.
- All IPV6 implementations are required to handle packets of at least 576 bytes, and additionally must be able to reassemble fragmented packets of at least 1500 bytes. Unlike IPV4, it is illegal to send datagrams exceeding 576 bytes unless an MTU discovery mechanism has reported a larger maximum size.
- Larger packets, of course, may be fragmented into multiple datagrams.

Next Header (8 bits):

- Defines the type of the next header. This is either an extension header or the header of the payload protocol (such as a TCP or UDP header). In the latter case, the values from [RFC-1700] are used.

Hop Limit (8 bits):

- The count of hops until the packet is routed to the great bit bucket in the sky. This is similar in nature to the TTL field in IPV4.

Source Address (128 bits):

- Address of originating interface.

Destination Address (128 bits):

- Address of destination interface(s).

One area of great concern to me is the hop count field which limits the diameter of a network to 256. I remain unconvinced that there is enough headroom in this value, especially when packets traverse several global, overlapping networks.

Extension Headers

Extension headers are generally not examined by routers as the packet is forwarded. The one exception to this rule is the hop-by-hop header which contains information to be processed at each point along the delivery path. If the hop-by-hop header is present it must be the first extension header in the packet.

When the packet reaches it's destination, any extension headers are processed strictly in the order in which they appear in the payload. There is no fixed limit to the number of extension headers that a packet may contain.

The currently defined extensions are:

Hop-by-Hop Options:

- A variable length header designed to carry options relevant to each hop. The only hop-by-hop option currently defined is the jumbo payload option, which allows a payload size of up to 232 octets. That's one hell of a large datagram.

Routing:

- A variable length header listing intermediate hosts that must be traversed before the packet is delivered. Multiple types of routing header are allowed for, but only type 0 is currently defined.
- A type 0 routing header allows each intermediate host to be treated as either a "loose" or a "strict" route in the IPV4 sense. Unlike IPV4, the routing instructions are only examined at the destination, so the entire packet is addressed to the next host, unpacked there, and then resent if there is anything left on the routing header list.

Fragment:

- A fixed length header specifying which fragment of a packet is contained in the payload of the current datagram. All fragments of a packet must arrive at the destination within 60 seconds or reassembly is abandoned.
- Fragmentation in IPV6 occurs at the originating node, and not in intermediate routers. This obviates the need for the IPV4 "don't fragment" flag.

Destination Options:

- A variable length header containing options that are relevant to the destination node. No destination options have been defined yet.

Authentication:

- A variable length header containing authentication data. This header is specified in [RFC-1826]. The overall security architecture of IPV6 is a complex subject, falling outside the scope of this paper. A very detailed expose is available in [RFC-1825].
- The authentication data acts as an electronic "fingerprint" to prove that the packet has not been tampered with.
- Different authentication algorithms may be applied (although these are not defined by the RFC) to different effect. For instance, MD5 is a cryptographically strong message digest algorithm that requires no key distribution. Alternatively, DES authentication can only be created by a party holding a secret key. Finally, public key encryption makes it possible to create an authentication header that acts as a digital signature, proving that a particular party originated the packet.

Encapsulating Security Payload:

- A variable length header that encapsulates data that has been encrypted. This header is specified in [RFC-1827].
- This mechanism is used to protect transported data from unauthorised interception. All payload that does not need to be visible to intermediate systems is cryptographically transformed and placed within an encapsulating security header. Like the authentication header, specific algorithms are not specified by the RFC, but may be chosen to meet the specific needs of the user.

The interested reader is referred to the RFC's (particularly [RFC-1883], [RFC-1826], [RFC-1827]) for more information on the exact layout of extension headers.

Checksums

Unlike IPV4, there is no header checksum included in the IPV6 packet. Damaged packets will either be delivered incorrectly or be discarded, and the hop count ensures that the worst case effects are severely limited.

The advantage gained by dropping checksums is the higher switching rates achieved by not calculating a sum every hop. Even mall savings on the processing resources for a packet yield enormous network performance gains when multiplied by billions of packets.

Higher level protocols, of course, protect the actual payload with their own checksum algorithms which are only applied at the endpoints.

AFFECTED PROTOCOLS

In order to utilise IPV6 effectively, many of the layers that it provides service to, as well as many supporting protocols, will have to be modified. In most cases, these modifications are straight forward since IP is usually utilised for unreliable packet delivery without making too many other assumptions.

One protocol that changes radically, however, is ICMP (Internet Control Message Protocol), since it interacts with IP at a fundamental level. ICMPV6 is defined in [RFC-1885].

Address discovery protocols such as ARP and RARP will need to change. RARP is no longer needed, being superceded by the ability of IPV6 to automatically allocate link level addresses, at which point a higher level address assignment protocol can kick in (such as DHCP). ARP can be easily extended to support 128 bit addresses, and can take advantage of multicasting rather than relying on the prescence of a MAC-layer broadcast facility.

UDP and TCP must change in order to accommodate the 128 bit addresses. In addition, the pseudo header used to calculate their checksum must be constructed differently. Obviously, any upper layer protocol utilising the services of UDP and TCP must be modified to support 128 bit addresses as well.

The DNS must change, again in order to accommodate the larger addresses. In addition, it is desirable for the DNS to go on supporting IPV4 addresses since the mapping to the IPV6 space is both easy and automatable. Recommended DNS extensions are documented in [RFC-1886].

Routing protocols (both internal and external) need change little in the way they measure routes and propogate information. Of course the address length issue is visible at this level, as well. There is significant opportunity for extending the flexibility of routing systems in the future to take advantage of the new addressing structure.

A Proposed Programming Interface

The Berkeley socket layer is easily modified to support an IPV6 based protocol stack, as described in [API]. The required changes include:

- Creating a new data structure to carry 128 bit addresses (struct sockaddr_in6).
- Providing new name to address mapping functions.
- Providing new address conversion functions.
- Providing new setsockopt(2) calls to access IPV6 functionality.

These modifications may be made to sockets libraries in such as fashion as to provide both source and binary backwards compatibility; a tribute to the original design.

In addition an advanced sockets interface is proposed in [ADVANCED] that allows full access to IPV6 extension headers, hop-by-hop options and the like. This is provided for "advanced" applications such as ping, traceroute and routing daemons.

Transitioning Strategies

Given the size of the Internet, it is impossible for there to ever be a "flag day", where IPV4 is turned off and IPV6 turned on. Even within a single organisation, some hosts and routers will be IPV6 capable before others, and a slow controlled migration has the attraction of both minimising risk and maximising the immediate benefits of the new technology.

In order to support a network with a growing number of IPV6 nodes coexisting with the current (and still growing) IPV4 installed base, there are several transition elements:

Support a Dual Universe:

- It is highly desirable that IPV6 nodes also support the old protocol. This makes interoperability with IPV4 hosts and routers straightforward and cheap. It also means that as IPV6 becomes prevalent on the backbone, IPV4 connectivity won't suffer.

Retain Addresses:

- Upgraded hosts can continue to use their old address both with their IPV4 stack and, as a mapped address, with IPV6. Additional IPV6-only addresses can be assigned to nodes as necessary without compromising their IPV4 adressability.

Tunnel IPV6:

- While there is still a significant IPV4 backbone, IPV6 traffic may be tunnelled to its destination by encapsulating it in IPV4 packets. This allows the early roll out of IPV6 hosts, without requiring a supporting router infrastructure.

Header Translation:

- When (and if) the backbone becomes IPV6 only, IPV4 traffic may be injected into the cloud by a header translation agent. A symmetric translator will recieve the packet and forward it to the destination node. This scheme also takes advantage of the IPV4 mapped addresses in the IPV6 space.

A detailed examination of transition issues and scenarios is provided in [RFC-1933].

Implementations

There are an impressive number of IPV6 implementations being developed for various platforms, including:

- 4.4BSD (US Naval Research Laboratory (NRL))
- AIX (IBM, Bull)
- BSDI/OS (WIDE)
- Digital UNIX (Digital)

- HP-UX (Swedish Institute of Computer Science)
- Linux (DRET, and others)
- NetBSD (Inria)
- SCO (SCO)
- Solaris (Sun Microsystems)
- System V Streams (Mentat)
- BS2000 Mainframe (Siemens Nixdorf)
- MacOS (Apple and Mentat)
- Netware (Novell)
- Windows 95 (FTP Software)
- VMS (Process Software)
- Embedded Systems (Pacific Softworks)
- Generic (University of New Hampshire)
- Routers (3Com, Bay Networks, Cisco Systems, Digital, Ipsilon Networks, Penril Datability Networks, Sumitomo Electric, Telebit Communications A/S)

Where Do We Go From Here?

The IPV6 story is just beginning. Never before has a protocol been designed with such an open process, and with such feedback from practical field experience. Never before has a protocol been subjected to such a degree of competition and scrutiny before being accepted as a standard.

And never before have we attempted to upgrade a global network the size of the Internet.

There is no doubt, however, that regardless of the difficulties in moving to a new Internet that IPV6 will survive, prosper and change the fundamental way that we achieve global networking.

With all the resources and commitment to IPV6, it will be the dominant Internet substrate protocol within the decade, with implementations coming into common use within 2 years. It may well become the standard protocol on the majority of LAN's within the same timeframe.

Personally, I'm hoping to look back in ten years and say, "Wow! That was easy. And look at the cool things we can do now that we'd never even thought of..."

Acknowledgements

I would like to extend my thanks to Jon Eaves, without whom this paper would have have been a lot less informative and readable. Thanks also to Enno Davids who made sure that this paper actually got written.

Finally, I'd like to acknowledge Linda, my wife, for the steady stream of beer, munchies and humour that kept my body and soul together during the long typing sessions.

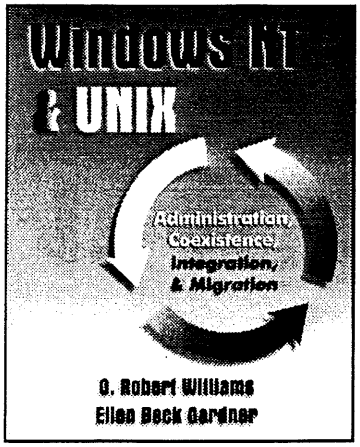
Bibliography

1. **[RFC-1550]** S. Bradner, A. Mankin. *IP: Next Generation (IPng) White Paper Solicitation.* RFC 1550, December 1993.
2. **[RFC-1700]** J. Reynolds, J. Postel. *Assigned Numbers.* RFC 1700, October 1994.
3. **[RFC-1825]** R. Atkinson *Security Architecture for the Internet Protocol* RFC 1825 , August 1995.
4. **[RFC-1826]** R. Atkinson *IP Authentication Header.* RFC 1826 , August 1995.
5. **[RFC-1827]** R. Atkinson *IP Encapsulating Security Payload (ESP).* RFC 1827 , August 1995.
6. **[RFC-1883]** S. Deering, R. Hinden. *Internet Protocol, Version 6 (IPv6) Specification.* RFC 1883, December 1995.
7. **[RFC-1884]** R. Hinden, S. Deering. *IP Version 6 Addressing Architecture.* RFC 1884, December 1995.
8. **[RFC-1885]** A. Conta, S. Deering. *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6).* RFC 1885, December 1995.
9. **[RFC-1886]** S. Thomson, C. Huitema *DNS Extensions to support IP version 6.* RFC 1886, December 1995.
10. **[RFC-1933]** R. Gilligan, E. Nordmark. *Transition Mechanisms for IPv6 Hosts and Routers.* RFC 1933, April 1996.
11. **[API]** R. Gilligan, S. Thomson, J. Bound. *IPv6 Program Interfaces for BSD Systems.* Internet Draft, draft-ietf-ipngwg-bsd-api-06.txt, November 1996.
12. **[ADVANCED]** W. Stevens, M. Thomas *Advanced Sockets API for IPv6.* Internet Draft, draft-stevens-advanced-api-00.txt, October 1996.
13. **[Bradner 95]** Scott O. Bradner, Allison Mankin (editors). *IPng Internet Protocol Next Generation.* Addison-Wesley, September 1995, ISBN 0-201-63395-7

❖

New from Addison Wesly Longman

Addison Wesley Longman



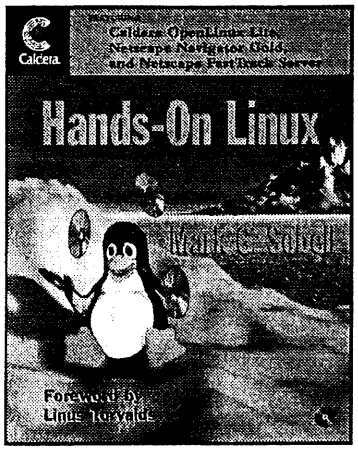
WINDOWS NT & UNIX

G ROBERT WILLIAMS AND ELLEN BECK GARDNER

This book is designed to bring administrators of each operating system up to speed with the administration of the alternative system, including cross-referencing the important utilities of both. The core of this book focuses on the three specific areas of interaction administrators must understand to be successful: coexistence, integration and migration

The book includes an in-depth look at how to plan for and implement the introduction of Windows NT into a UNIX environment and an examination of available tools for porting UNIX applications

ISBN : 0-201-18536-9 RRP $89.95



HANDS-ON LINUX

MARK G SOBELL

This book is the perfect combination for Linux enthusiasts who want both a complete tutorial and reference on Linux as well as a comprehensive Linux software starter kit. Beginning with an extensive Linux tutorial and then quickly progressing to detailed chapters on GUIs, networking, the **vi** and **emacs** editors, three popular shells, programming tools and system administration. Part II is a comprehensive reference containing descriptions and examples of eighty-seven utility programs

ISBN : 0-201-32569-1 RRP $89.95

AUUG Members receive a 20% Discount off the RRP

order form

To order a copy of these texts, please fill in the details below.
This form should be faxed or mailed (freepost) to Lisa Russell, Trade Marketing Co-ordinator at Addison Wesley Longman, Level 1, 2 Lincoln Street. Lane Cove NSW 2066. Ph: (02) 9428 8086 or Fax: (02) 9427 9922

Book Details

Author/Title	ISBN	Price
_____	_____	_____
_____	_____	_____
_____	_____	_____
	Total less 20%	_____

Order details

If you are **purchasing** the books detailed above, please give the following information. (Please note that books can also be ordered directly from bookstores.)

Book(s) price:_____ Note: Prices are subject to change without notice

Postage: 1 book = $4.00, 2 or more = $7.00 Total $ _____

❒ I enclose a cheque made out to Addison Wesley Longman

❒ Please charge my credit card:

❒ Bankcard ❒ Visa ❒ Mastercard ❒ American Express ID No:_____

Card Number: _____ Expiry Date _____

Signature:_____

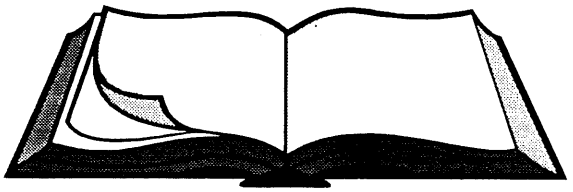
Your Name: Mr/Mrs/Ms _____

Address: _____

_____

Daytime Ph: _____

Book Reviews



THE JAVA TUTORIAL: OBJECT-ORIENTED PROGRAMMING FOR THE INTERNET

by Mary Campione and Kathy Walrath
Reading, MA: Addison-Wesley
2nd Edition, 1998, ISBN 0-201-31007-4

Reviewed by
Paul A. Watters
Department of Computing
Macquarie University NSW 2109
<pwatters@mpce.mq.edu.au>

It's not often the case that you can get something for free, and then decide it's good enough to pay for. The "Java Tutorial, 2nd Edition" by Campione and Walrath is a book that falls squarely into this category: it is an imprint of the very successful Java tutorial available (free) at the Java web site:

http://java.sun.com/docs/books/tutorial/index.html

The book is written in a very readable, friendly and often humourous style that I found enjoyable (this style is maintained in both the online and paper versions of the book).

The volume is divided into seven main "trails". Each trail is quite self-contained, but can also be read sequentially for a complete coverage of all aspects of java programming. Topics range from how to run the java compiler to discussions of advanced object-oriented concepts and their implementation.

Beginners would gain the most from reading the early trails and referring to the online examples, while more experienced users would benefit from the discussion of networking and an elaboration of the differences between JDK 1.0.2 and JDK 1.1. I found the strategies for graphics and animation the most useful section in later trails.

Although I wouldn't normally recommend the print version of a book that is available electronically - as the use of hypertext and online searching is preferable in most cases - this book has a number of innovative design features that contribute to the largely seamless interface with the online version. The authors went so far as to include a "toolbar" on every page for those who feel at home with a screen.

The URL for each online chapter is cross-referenced in the paper version above the title of each chapter. In addition, each chapter contains "links" to relevant material in other trails which can be followed for further information. This is as close to hypertext in a book that I've ever seen.

The second edition fully integrates the many changes from the JDK 1.0.2 to the JDK 1.1 (and includes a copy of the JDK for Sun Solaris and Microsoft Windows on a CD at the back of the book). These changes are discussed in-depth in the text, from improvements in garbage collection to the new JAR platform independent file format.

If this book is available electronically for free, why would you decide to pay for it? Well, I can think of three good reasons:

- You're working away from the Internet on a machine that doesn't have a CD-ROM (but you still need to write that applet by Monday...).
- You want to absorb the 964 pages of introductory Java by learning everything about it before going near a keyboard.
- You are a highly moral person who wants to reward the authors and publishers for providing such an excellent resource for free on the Internet in an era of mindless commercialism.

None of these are trivial reasons, and there are potentially more general issues about the ergonomics of the printed page versus the high-frequency computer monitor which might also be considered by book-lovers. I would not hesitate in recommending this book as a reference work for the bookshelf and CD-rack of beginners and experienced java programmers alike.

❖

Meet the AUUG-Exec

MARK WHITE

"Mum, I've decided - I want to be a biochemist. So I'm going to go to Uni and do a science degree. Mostly chemistry stuff. And some biology too, I guess."

Now my mother has a better long-term memory than almost everyone, and I was reminded of this comment during last year's annual family Christmas gathering. The previous week had seen me in Beijing and Shanghai (speaking neither Mandarin or Shanghainese, I might add – but enjoying the food!) convincing people that UNIX clusters - real clusters, not that failover stuff - was the way to build modern information systems. Go figure.

Anyway as this is AUUGN, and not "Biochemistry Australia" you can probably surmise that I didn't really follow my once-intended career part. Not too well, anyway.

There was something about CS100; maybe it was the goofy-looking lecturer who used to occasionally quote Bruce Springsteen while teaching Pascal (I still listen to Springsteen occasionally, but I never, *ever* write code in Pascal). Maybe because I forgot to notice the absurdly high geek:girl ratio. And around that period most of my spare time was spent playing bass in various cover bands around South-East Queensland, so I guess my interest in things biochemical completely waned, in favour of a spirit of basic survival in the Computer Science department.

So I somehow found myself entering my final year of said science degree having completed minimal chemistry and biology and instead having – quite amazingly – *enjoyed* some of the computer subjects. But at this stage not really knowing quite what computer science graduates actually *did* for a living (several of my friends would walk around talking about binary trees and their artificial intelligence honours projects) I followed the queue into a small software shop in Brisbane.

At UQ, you understand, UNIX was something called "Tahoe" on a series of machines called "madvax". Ha Ha - Get it? (I never saw the movie until years later, so the joke was always something of a mystery to me). I'd worked on the basis that these mysterious, invisible machines (all we ever saw was a lab full of crapped-out VT100's) were to be minimally used for assignments only and otherwise forgotten. It was only at the aforementioned programming shop did I start to learn that UNIX was a whole lot more fun than most other systems out there – you could easily network it, for starters. And, well, if you needed to there was a way to do almost *anything*. And, well – it was fun.

During these formative years I also learned an important lesson – lots of people didn't understand computers. So being able to write, translate and importantly present "geek-speak" to the pointy-haired folks was something of an asset, and something that I also came to enjoy.

This set the stage for gradually working into team- and project-management roles. Still a lot of hands-on work, but moving towards the business side as well. I migrated out of the general information systems arena into telecommunications and networking roles and decided that telecommunications was it. However, an eventual promotion to a role where I was spending more time analyzing other people's timesheets than doing "meaningful" work led to yet another change of course: consulting and pre-sales for Tandem Computers, who design and manufacture the most universally resilient computers on the planet. And besides, they also develop UNIX platforms for the telecommunications industry. I was hooked.

Three years (and a successful merger with Compaq) later I run Tandem/Compaq's UNIX & Telecommunications Marketing efforts in the Asia-Pacific region. I work mostly with our sales teams in Asia, as well as our telecommunications-focused solution partners and the odd switch vendor. I get involved in a variety of projects, from glossy product-launches to porting and benchmarking software to making China's telephone network more efficient. We're also preparing to introduce commercial, Intel-based UNIX clusters – single system image, dynamic process migration from one server to another, good scalability. Fortunately, the days are never dull. I'm also a case study in telecommuting, as I (nominally) work in Singapore, but live in Brisbane.

Outside of work I'm part of the successful Queensland AUUG Chapter, and have been a member of the AUUG Executive Committee since 1996. In 1997 I was honored to be the Conference Chair for AUUG'97, and was pleased to be a part of the effort to finally stage that great event in a great city!

Outside of work **and** AUUG I share a renovated house on Brisbane's north side with my partner Susan and our cat Hobbes. We both enjoy travel, wine, good books, our MG Midget and loads of other non-computer stuff.

And fortunately, Mum's not too disappointed about the biochemistry thing not working out!

❖

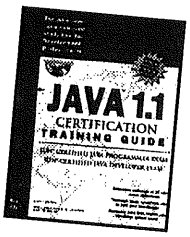
AUUG BOOK CLUB & PRENTICE HALL AUSTRALIA

☐ Essential JNI - Java Native Interface
Rob Gordon
0136798950
~~$44.95~~ $35.95

- *A complete task oriented reference to the entire Java Native Interface API*
- *Tools and strategies for integrating legacy code with new Java applications*
- *Extensive code examples and detailed debugging tips*
- *Clear explanation of JNI native types, signatures, references, and object and class functions*
- *Specific techniques for managing C++ code and converting C structures*

☐ Interprocess Communications in UNIX
John Shapley Gray
0138995923
~~$59.95~~ $47.95

- *Extensive coverage of IPC techniques*
- *Named and unnamed pipes, message queries, semaphores and shared memory*
- *RPC and sockets*
- *Multireading applications with POSIX threads*

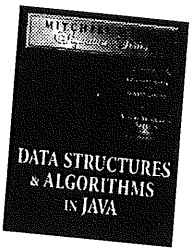


☐ Java 1.1 Certification Training Guide
Cary A Jardin
078791390x
~~$79.95~~ $63.95

- *Java test engine simulates the actual Sun Java 1.1 Programmer and Developer exams*
- *test questions delivered in random order to create thousands of unique exams and test your true knowledge*
- *Questions, objectives and scoring methods are similar to the actual exam*
- *test engine constantly pulls new questions from our updated Web site*

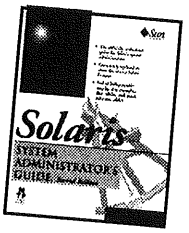
☐ Data Structures & Algorithms in Java
Robert Lafore
1571690956
~~$79.95~~ $63.95

- *Designed to be the most easily understood book ever written on data structures and algorithms*
- *Taught with Workshop Applets - animated Java programs that introduce complex topics in an intuitively obvious way*
- *text is clear, non academic, and supported by data*
- *simple programming examples are written in Java which is easier to understand than C++*



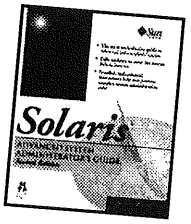
☐ Solaris System Administrators Guide 2E
J Winsor
157870040
~~$69.95~~ $55.95

- *learn to administer devices, file systems, and network services*
- *understand the process of printing and user or group accounts*
- *recognise file and access problems more easily*
- *master the use of shell and basic operating systems*
- *use the redesigned print packages effectively*



☐ Solaris Advance Systems Administrators Guide 2E
J Winsor
157870396
~~$69.95~~ $55.95

- *Connect to mail services for quicker more efficient mailing*
- *Use NIS+ naming and Automounter services*
- *Set up access to printers, modems and terminals using the Service Access Facility*
- *Take advantage of software installation and sharing*



AUUG members receive 20% discount below recommended retail price

Name: _____ Position: _____

Company: _____

Address: _____

_____ Telephone: _____

ORDER NOW

☐ Enclosed cheque for $ _____ (Payable to 'Prentice Hall Australia')

☐ Charge to me **OR** ☐ Company purchase Order No. _____

Please charge my: ☐ Bankcard ☐ Visa ☐ MasterCard ☐ AMEX

Expiry Date: _____ Credit Card No: ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

Signature: _____

Mail or fax completed order form to Prentice Hall Australia,
14 Aquatic Drive, Frenchs Forest NSW 2086, Fax (02) 9453 0117

OR ✆ Use our PHONE SERVICE by calling Jan Blenkinsop

SYDNEY (02) 9454 2211

PRENTICE HALL AUSTRALIA

AUUG498

Call For Papers

AUUG98 Conference
September 3-5, 1997
Sydney Hilton Hotel,
Sydney, New South Wales,
Australia

Open Systems:
The Common Thread

THEME: "OPEN SYSTEMS: THE COMMON THREAD"

The 1998 AUUG winter conference will be held at the Sydney Hilton Hotel, New South Wales, Australia, between September 16th and 18th.

The conference will be preceded by two days of tutorials, on September 14th and 15th.

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

Technical aspects of Unix and Open Systems

New developments in open software systems, languages and applications

Networking, Internet (including the World Wide Web) and Security

Business and Management Experience and Case Studies

The theme of this years conference is "Open Systems: The Common Thread". The program committee will interpret the theme very broadly with the aim of highlighting the breadth of applicability for Open Systems. As always, papers and tutorials with a strong technical flavour are particularly welcome.

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. Tutorials may be either 1/2 day or full day and have a strong practical focus.

All presentations must be accompanied by a written paper for the conference proceedings.

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 Unix and Open Systems event held in Australia this conference offers an unparallelled 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:	May 29, 1998
Authors notified:	June 12, 1998
Final copy due:	August 7, 1998
Tutorials:	September 14-15, 1998
Conference:	September 16-18, 1998

Proposals should be sent to:

AUUG Inc.
PO Box 366
Kensington NSW 2033
AUSTRALIA

Email: auug98@auug.org.au

❖

SAGE-AU Sixth Annual Conference and General Meeting

Tuesday 7/7/1998 to Friday 10/7/1998
Old Parliament House
Canberra, ACT,
Australia

CALL FOR PAPERS AND TUTORIALS

The System Administrators Guild of Australia (SAGE-AU) will be hosting its sixth annual conference in conjunction with its 1998 annual general meeting.

The annual SAGE-AU Conference, Tutorials and AGM provides a forum for Systems Administrators, Systems Managers, Network Administrators, Developers of Systems Administration Software and Managers of such groups to meet and share their knowledge and experiences.

SAGE-AU'98 is hereby calling for papers and tutorial presentations on any and all topics related to system administration.

DEADLINES

Applications to present tutorials and papers must reach the organisers by April 3, 1998.

To be included in the conference proceedings, papers must reach the organisers by June 19, 1998.

CONFERENCE DETAILS

SAGE-AU'98 will be a 4 day conference running from Tuesday July 7, 1998 to Friday July 10, 1998.

The first two days (Tuesday & Wednesday) will be dedicated to tutorials on tools and techniques to aid system administration.

The AGM will be held at the end of the third day (Thursday). All other times will be allocated to presentations or discussions.

A conference dinner will be held on the Thursday evening.

The conference will feature a small trade show on the third and fourth days, focusing on system administration tools and information.

PAPERS

Timeslots are available for 15, 30, 45 and 60 minute presentations. 5-10 minutes should be reserved for questions from the audience.

15 minute timeslots are less formal and are to allow people to talk briefly about some topic of interest or problem without having to prepare a formal paper (Work in Progress).

People presenting a 30+ minute talk will receive free conference registration.

People presenting a 15 minute talk will receive a 50% discount on the conference registration fees.

If you wish to present a paper, send an abstract to the address below by the due date. Please indicate whether you are asking for a 15, 30, 45 or 60 minute timeslot.

Abstracts should be 100 -- 200 words in length. Papers should have a technical orientation and should not contain advertising.

People giving 30+ minute presentations will be expected to provide a paper for inclusion in the conference proceedings.

TUTORIALS

Tutorial sessions will be either half day or full day in duration. People wishing to present tutorials should submit an abstract of the material they wish to present and an indication of whether they require a half day or a full day timeslot. Tutorials should be run in lecture format. Suggested topics include:

- Computer and Network Security/Network Authentication
- PC/Apple/Unix/Mainframe Interoperability
- NFS/Automount/AMD Configuration and Operation
- Perl/Java/Tcl/Python
- Sendmail/Qmail/smapd/Anti-SPAM
- WWW Cache/Router Config/Firewall Setup/Squid
- NT/Win95 Administration

Tutorial presenters will be paid $500 for a half day tutorial and $1000 for a full day tutorial and will receive free conference registration. SAGE-AU will reimburse tutorial presenters for reasonable costs of handout materials or will print them on your behalf.

As with papers, tutorials should have a technical orientation and should not contain advertising.

Exhibition/Trade Show

On the third and fourth days of the conference (Thursday and Friday) SAGE-AU'98 will host a small, technically orientated trade show focusing on system administration tools and information.

If you or your company are interested in participating in the trade show please contact the organisers for details.

Registration

Conference registration includes one ticket to the Conference Dinner and Conference and Tutorial registration includes Lunch and Refreshments. Additional tickets to the Conference Dinner may be purchased.

Non-members who register for SAGE-AU'98 at the non-member rate and successfully apply for membership of SAGE-AU will have their first year's membership fee waived.

Conference registration forms will be available in mid May 1998.

Registration forms for tutorials will be available approximately six weeks before the conference date.

Early Registration is considered when registration form and payment has reached SAGE-AU by COB on 19th of June 1998.

Travel

To encourage interstate attendees, SAGE-AU offers members a travel discount off registration for interstate travellers (Qld/Vic/Tas/WA/SA/NT).

Addresses

Send all enquiries regarding the conference as well as tutorial and paper abstracts to:

SAGE-AU'98
GPO Box 2984
Sydney NSW 2001
Australia

E-Mail: conference@sage-au.org.au

Requests for general information about SAGE-AU and membership applications should be addressed to:

`WWW: http://www.sage-au.org.au/`

Email: secretary@sage-au.org.au

Fax: 0500 544 488 (Attn: David Conran)

Or alternatively,

Secretary
SAGE-AU
GPO Box 2984
Sydney NSW 2001
Australia

WWW Page

A web page for the conference is

`http://www.sage-au.org.au/conf.html`

❖

Solaris Musings

David Purdue <David.Purdue@Aus.Sun.COM>

It is amazing what a trawl through the Solaris man pages will turn up. This issue I will briefly discuss filesync.

This little utility was introduced in Solaris 2.6, and is used, as the name suggests, to synchronise files that reside on different file systems. Actually, it will keep any two directory trees in synch, but its primary rationale is for synchronising a working set of files between a file server and a portable computer.

The modus operandi is as follows. Typically you will store your files on a file server and access them from there, mounting them on to your system with NFS. This makes file management easier (it is easier to provide backup for one server than for twenty desktops), but does not do you a lot of good when your computer is on your lap at 35,000 feet.

So, while you are away, you want to be able to access local copies of your files. In the meantime, you do not want to preclude other people from accessing shared files.

Enter filesync. When you run filesync, it will synchronise the mounted file system with the local file system on your portable computer. filesync's operation is controlled by a file: $HOME/.packingrulles, which I will describe later.

filesync synchronises pairs of directory trees by examining the files in these trees and seeing which have changed since the last time filesync was run. If the local copy has changed, it is copied to the server. If the server copy has changed, it is copied to the local machine. If both have been changed, then filesync has no real way of telling which is the correct copy, and so warns you and asks you to reconcile the changes manually. Alternately, you can tell filesync whether to give preference to local or remote copies of files.

filesync has a number of options, but almost all of them are for making modifications to the $HOME/.packingrulles file, and I find it far easier to edit it with my favourite editor. .packingrules contains a number of directives:

PACKINGRULES 1.1	Not required, but identifies this as a packing rules file - the "magic number".
BASE dir1 dir2	Nominates the pairs of directory trees to keep in sync. These need to be fully qualified paths. BASE defines the start of a block; LIST and IGNORE directives given in the file refer to the previous BASE.
LIST name ...	specify a list of files and directories to synchronise within the BASE directories. Any directories named will include their subdirectories and so on recursively. Regular expressions are permitted. LIST directives can contain slashes, in which case they are interpreted as relative to BASE.
IGNORE name ...	Specifies file names that are not to be synchronised. Regular expressions are permitted. If IGNORE appears before any BASE directives, then those files are ignored for all pairs. IGNORE will not override LIST.

If the argument to LIST or IGNORE starts with an exclamation mark (!), then the line is interpreted as a command that is executed, and the output of that command will be the list of files to synchronise or ignore.

Here is an example .packingrules file:

```
PACKINGRULES 1.1
# idenify the file.

IGNORE core *.o *.bak
# in all cases ignore core files, object
files and old edits.

# Most of what I want is in my home
directory.
BASE       /net/fileserver/export/home/davidp
$HOME

# Get my work files and my mailboxes.
LIST work Mailfiles

# Now get the project I am working on.
BASE       /net/projectone/export/bigproject
$HOME/project
LIST !cat .projectlist
# Don't give me the Postscript doco - too
big.
IGNORE *.ps *.ps.Z *.ps.gz
```

Once you have your packing rules, filesync is invoked without arguments to synchronise the file systems.

filesync is a simple utility - but knowing it is there saves you from writing your own.

❖

Chapter News:

Queensland

According to Mark White, Unix Marketing Manager for Tandem (a Compaq company!), the meaning of the word "cluster" has been diluted in the last few years. Most specifically by the folks from Redmond! In his presentation to the QAUUG February meeting, Mark gave an overview of Tandem's "NonStop Clusters for UnixWare", a bundled product with Intel-based servers from Tandem's new parent.

Failover "clusters" - according to Mark - are a somewhat cumbersome and expensive solution to a company's availability requirements ("buy two, get one"). An ideal solution would be a farm of smaller servers, loosely-bound together into a "single system image". As a collection of servers the scalability benefits are enormous, yet each node is an isolated fault-zone - improving availability.

Tandem, like Digital, have been producing clustered products for their proprietary platforms for years. However this product represents the first time a clustered UNIX solution has been commercially released, and a great leap forward for the UNIX community. There was a lot of interest in the topic and a great evening was had by all.

The AUUG Queensland Chapter holds meetings on the last Tuesday of every month - except in April, where the monthly meeting will be replaced by the annual one-day Chapter Technical Conference. The CTC will be held on April the 18th, for more information contact the AUUG Secretariat on 1800 625 655.

To receive announcements about the AUUG Queensland Chapter meetings and other events, send an e-mail message to majordomo@auug.org.au with the line "subscribe qauug <your e-mail address>".

❖

AUUG Local Chapter Meetings 1998

CITY	DATES	LOCATION	OTHER
BRISBANE	26 May 30 June 28 July 25 August 29 September 27 October 24 November	Inn on the Park 507 Coronation Drive Toowong	For further information, contact the QAUUG Executive Committee via email (qauug-exec@auug.org.au). The technologically 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	9 June 14 July 11 August 8 September 13 October 10 November 8 December	Australian National University	
HOBART	Each month, although dates can vary. Often will fit in with the schedule of a speaker should one be available.	University of Tasmania	
MELBOURNE	20 May 17 June 15 July 19 August 21 October 18 November 16 December	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	20 May 17 June 15 July 19 August 21 October 18 November 16 December	The Victoria League 276 Onslow Road Shenton Park	Meeting commences at 6.15pm
SYDNEY	21 May 18 June 16 July 20 August 15 October 19 November 17 December	The Wesley Centre Pitt Street Sydney 2000	The February meeting will be replaced by the summer conference on 21 February.

*** All dates are subject to change.**
Up-to-date information is available by calling AUUG on 1-800-625-655.

UNIX Traps & Tricks

Sub-Editor: Matthew Dawson
<dawson.matthew.ms@bhp.com.au>

Hi Everyone! Welcome to another edition of UNIX Tricks & Traps - the column designed to provide insights into how your fellow AUUG members make their day-to-day usage of UNIX easier.

This issue consists mainly of alternate ways to implement David Purdue's tidypath tip published in the previous issue. David (unintentionally ???) issued a challenge in his commentary, and a few of you just couldn't resist...

The other contribution is a 'back to basics' file management tip. While it is simple to implement, it has the potential the save much heart-ache if rigorously practised.

Thanks to those who contributed to this issue, and please - keep those tips coming. We appreciate them all!

❖

SON OF TIDYPATH

In the previous issue of UT&T, David Purdue provided a C program to tidy out-of-control PATH variables. It achieved this by pruning all bar the first instance of directories which had been including multiple times.
He ended his commentary with this:

> "I tried to write tidypath as a shell script, but after ten minutes could not get the right combination of commands and so gave in and wrote it in C. I imagine there is a 5 line equivalent in perl, but I don't have a week to find, install and learn perl."

and a few of you decided to show how these different implementations of tidypath could be done.

Each of the implementations has its own pros and cons. The pure Bourne shell script is the most portable, and has the added bonus of removing any non-existent directories which have found their way into your PATH. The Perl versions are the shortest and/or simplest scripts of the bunch but not all systems have Perl installed. And the C program is the longest and requires a C compiler, but to quote David on its competitors "They are all slower and use more memory".

Once again we see that there are as many ways to solve a problem (under UNIX) as there are UNIX users...

❖

BOURNE SHELL/AWK

From: Peter Chubb <peterc@softway.com.au>

I was challenged to produce a shell (well, awk really) version of tidypath. It wasn't hard.

The only tricky bit is coping with the first element: it mustn't be preceded by a : (colon) because that means putting the current directory in the path; and you can't test path for emptiness because it springs into existence empty; and if there was a real current directory at the head of $PATH you need to cope.

```
#!/bin/sh

if [ $# -ne 1 ]
then
        echo >&2 "Usage: $0 \"$PATH\""
        exit 1;
fi
echo "$1" | awk '
        {
                n = split($0, a, ":")
                for (i = 1; i <= n; i++)
                {
                        if (!(a[i] in rev)) {
                                rev[a[i]] = i;
                                if (i == 1)
                                        path = a[1];
                                else
                                        path=path ":" a[i];
                        }
                }
                print path;
        }
'
```

BOURNE SHELL

From: Peter Chubb <peterc@softway.com.au>

This version of tidypath doesn't need awk. It also copes correctly with directories with strange names (including spaces, tabs, newlines, backslashes, and egrep special characters). I may have missed one or two egrep characters, actually -- just add them to the sed script inside egquot() if so. The script also illustrates an important shell programming technique: use of set and IFS to use the shell's parser to split a string into components.

A problem with echo is that System V echo interprets backslashes to mean things (\b means backspace, etc) and there's no standard way to turn it off! (Under bash, -E works; under Solaris, use /usr/ucb/echo, etc.) So I wrote an Echo shell function to overcome this limitation.

I thought about resolving symlinks and deleting symlinks that pointed to a place already in the PATH, but that could be bad in a dual-universe setup (like Sequent's older systems) where symlinks point to different things depending on which universe one's operating in.

To ensure that any space/tabs in the path are preserved, you should invoke tidypath like this:

```
PATH="`tidypath \"$PATH\"`"
```

```
#!/bin/sh
#
# tidypath -- tidy up a set of directories specified as a string
# with components separated with colons.
# Remove duplicates and non-directories.
# Do NOT resolve symlinks, to preserve dual-universe semantics
# where it's important.
# An empty component means current directory.
#

# Too many versions of echo out there -- we want NO interpretation at all
# (Otherwise backslashes in path directories may not do what you want)
Echo() {
        printf "%s\n" "$@"
}
#
#
# Quote a string for passing to egrep as a regexp.
egquot() {
        Echo "$@" | sed 's/\([\\\[\].*?+^$]\)/\\\1/g'
}

#
# Check usage.
if [ $# -ne 1 ]
then
        Echo 'Usage: PATH="`'"$0"' \"$PATH\"`"' >&2
        exit 1;
fi

#
# Get the bits of $PATH into $1, $2, $3, etc
OIFS="$IFS"
IFS=:
set - $1
IFS="$OIFS"

while [ ! -d "${1:-.}" ]
do
    shift
done
p="$1"
shift

for i
do
        [ ! -d "${i:-.}" ] && continue
        Echo "$p" | egrep '(^|:)'"`egquot \"$i\"`"'($|:)' >/dev/null ||
                p="$p:$i"
done
echo "$p"
```

PERL (SHORT)

From: Miles Goodhew <milesg@defcen.gov.au>

Here's another version of tidypath. I paid special attention to making the thing a brief as possible, so it reads like the transcript of a dockside pub brawl.

Regardless of whether you care or not at this stage, I'm going to tell you how it works anyway:

- Firstly the "foreach (split..." part splits the input string at colons, and iterates over each substring (element).
- The "$x .=" means append the following string to the $x variable
- The "$x =~ /..." part tests $x to see if it contains the current element. That is, it matches the pattern "(^|:)$_(:|$)" meaning: "start of string or a colon, followed by the current element, followed by a colon or end of string".
- The '? "" ' part is a conditional (just like in C), basically if the pattern matched (the element is already in $x), then append the null string, otherwise...
- The "($x ?..." is, of course, another conditional. It effectively means: "If $x is empty then set it to the current element (which is by definition not already in $x), otherwise append a colon and the current element to $x".
- The last line 'print "$x\n";' should be self-explanatory.

```
#!/usr/bin/perl
foreach (split /:/, $ARGV[0]) {$x .= (($x =~ /(^|:)$_(:|$)/) ? "" : ($x ? ":$_" : $_));}
print "$x\n";
```

Just to be a complete sod, you could trim-off some extra characters to get the single command-line equivalent:

```
perl -e 'foreach(split/:/,$ARGV[0]){$x.=(($x=~/(^|:)$_(:|$)/)?"":($x?":$_": $_))}print"$x\n"'
```

PERL (SIMPLE)

From: Phil Kernick <philk@rotfl.com.au>

In the February AUUGN, David Purdue said "I imagine there is a 5 line equivalent in Perl...". That sounded way too much like a challenge to me, so here is my version of tidypath that has exactly the same semantics, but takes only 4 lines of Perl.

It has the added advantage that if you don't explicitly specify which PATH you want to tidy, it assumes $PATH.

```
#!/bin/perl
foreach $dir (split(/:/, ($ARGV[0] or $ENV{"PATH"}))) {
  push @path, $dir if not exists $found{$dir};
  $found{$dir}++;
}
print join(':', @path);
```

CAREFUL COPY

From: David Bell <dbell@canb.auug.org.au>

Have you ever copied or linked a file from one directory to another, but you either misspelled the directory name or else the destination directory wasn't yet created, and so the command created an unwanted file in the next directory up instead? As an example:

```
cp prog /usr/local/bim
```

silently creates a file "bim" in "/usr/local" instead of creating "prog" in "/usr/local/bin" as intended.

Well, there is a simple solution to this problem. Just append a trailing slash to the destination path. This will guarantee that the final directory component is valid. Using the above example:

```
cp prog /usr/local/bim/
```

will give an error message about the non-existent directory "bim' instead of silently creating an unwanted file in the wrong place.

This is such a simple trick, but it has saved me lots of pain over the years.

❖

Notification of Change

AUUG Inc
UNIX® AND OPEN SYSTEMS USERS

You can help us! If you have changed your mailing address, phone, title, or any other contact information, please keep us updated. Complete the following information and either fax it to the AUUG Membership Secretary on (02) 9332-4066 or post it to

AUUG Membership Secretary
P.O. Box 366
Kensington, NSW 2033
Australia

(Please allow at least 4 weeks for the change of address to take effect..)

❒ The following changes are for my personal details, member #:_____

❒ The following changes are for our Institutional Member, primary contact.

❒ The following changes are for our Institutional Member, representative 1.

❒ The following changes are for our Institutional Member, representative 2.

PLEASE PRINT YOUR OLD CONTACT INFORMATION (OR ATTACH A MAILING LABEL):	PLEASE PRINT YOUR NEW CONTACT INFORMATION:
Name/Contact: _____	Name/Contact: _____
Position/Title: _____	Position/Title: _____
Company: _____	Company: _____
Address: _____	Address: _____
_____ Postcode _____	_____ Postcode _____
Tel: BH _____ AH _____	Tel: BH _____ AH _____
Fax: BH _____ AH _____	Fax: BH _____ AH _____
email address: _____	email address: _____

AUUG Secretariat Use

Date: _____
Initial: _____
Date processed: _____
Membership # _____

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 $70 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	☐	$350.00
Surcharge for International Air Mail	☐	$120.00
Additional Representatives	Number ☐	@ $80.00

Rates valid as at 07/96

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) 9332-4066

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 Secretariat
PO Box 366, Kensington NSW 2033, Australia

Tel: (02) 9361 5994
Free Call: 1 800 625 655
Fax: (02) 9332 4066

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 bi-monthly 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 ☐ $100.00

RENEWAL/NEW STUDENT MEMBERSHIP

Renewal/New Membership of AUUG *(Please complete Section C)* ☐ $25.00

SURCHARGE FOR INTERNATIONAL AIR MAIL ☐ $60.00

Rates valid as at 07/96

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) 9332-4066

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

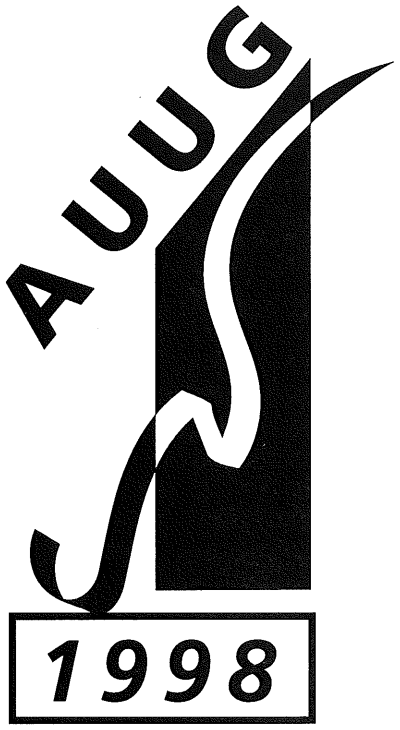
Chq: bank _____ *bsb* _____

A/C: _____ *#* _____

Date: _____ *$* _____

Initial: _____ *Date Processed:* _____

Membership#: _____



AUUG Inc. The UNIX and Open Systems Users Group, invites you to participate in this regions only UNIX and Open Systems Conference and Exhibition, to be held over 3 days, 16th to 18th September, 1998 at the Sydney Hilton Hotel.

As Unix and Open Systems push ahead, this year's exhibition will be complemented by a very strong Technical Programme.

This premier event has always attracted the participation of leading companies and organisation, be among them by securing your sponsorship and stand now!

PLEASE RETURN YOUR FORM TO:

ACMS - Conventions & Exhibitions Pty. Ltd. PO Box 468, Paddington, NSW 2021. 70 Glenmore Road, Paddington, NSW 2021
Telephone (02) 9332 4622 • Facsimile (02) 9332 4066 • E-mail: auug98@acms.com.au • URL: http://www.auug.org.au

APPLICATION FORM

16 - 18 September 1998 Sydney Hilton Hotel

Exhibit areas and sponsorships will be allocated and confirmed upon receipt of this application and the relevant amount.

EXHIBITION I/We wish to book the following location(s):

1st Preference _____ 2nd Preference _____ 3rd Preference _____

ADVERTISING & SPONSORSHIP I/We wish to be involved by:

SPONSORING THE: 1. _____ 2. _____

ADVERTISING IN: 1. _____ 2. _____

Company Name _____

Address _____

Suburb _____ State _____ Postcode _____

Telephone ( ) _____ Facsimile ( ) _____

E-mail Address _____ URL _____

Contact Name _____ Position _____

Enclosed is our cheque for A$ _____ OR Please debit my credit card for A$ _____

Tick the appropriate box:

☐ Bankcard ☐ Mastercard ☐ American Express ☐ Visa Expiry Date:__/__/__

☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

In the name of _____ Signature: _____

SPONSORSHIP

Benefits of being a sponsor at AUUG 98

A: MAJOR SPONSORS A$7,500

- One complimentary registration for the Conference.
- Two complimentary invitations for the Cocktail Reception.
- Two complimentary invitations for the Conference Dinner.
- Logo displayed in Conference Plenary Hall.
- Acknowledged on all appropriate occasions both in print and verbally

Listed and identified as a Sponsor in:

- The Conference Brochure.
- The Conference Final Programme and to the specific benefits emanating from the items they are sponsoring.

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.
- The Cocktail Reception identified as being sponsored by the XYZ Co. on all printed material and on the evening by way of a sign.

Conference Satchel

- Folder/Satchel offering long term usage and long term company message to recipient.

Conference Proceedings

- Two A4 pages of exclusive advertising.
- Long term usage and shelf life as it is a reference material.

Conference Dinner

- The Dinner identified as being sponsored by the XYZ Co.
- Printed menu identifying the sponsoring company.
- Opportunity to distribute mementos to the audience.
- Banner identifying the sponsoring company.
- Name of sponsoring company on the entree tickets

B: SPONSORS - A$2,500

- Morning or afternoon breaks (6)
- Lapel Badges

C: ADVERTISING -

A$1,500 each

Conference Programme

- 5 individual advertisements available, reaching an audience of key decision makers.

Conference Folder Insert

- Individual inserts each offering immediate enticement to the delegates to visit your stand or enquire about your product.

Conference Proceedings

- Advertisements A4 size available, a lasting reference guide.

Exhibition Show Bag

- Your logo on one side, the conference logo on the other side and your company "insert" included.

Light Box

- At entrance of Exhibition, first and last impression on entire audience.

EXHIBITING

Your participation in this Exhibition, entitles one person from your company to the following benefits:

1. Attendance at the Cocktail Reception
2. Provision of morning and afternoon coffee
3. Attendance at the Conference Dinner

And in addition your company will be given a:

4. Listing in the Conference Brochure
5. Free of charge listing in the final programme
6. Free of charge listing in the Exhibitor's Guide
7. 200 Trade passes to invite your existing and potential clients to visit the Exhibition.

THE COST FOR EXHIBITING

- A$290 per sq.m. - SHELL SCHEME

Stands are available of various sizes and configuration (please refer to the floor plan). Each shell scheme stand offers the following features:

- Velcro material surface
- Fascia headerboard company name sign
- Two 75 watt electric spot lights

Please complete the Application Form and return it to the Organisers together with 50% of the amount for the stand/area of your choice.

SETTING UP AND DISMANTLING

Setting up of the stands will take place on Tuesday, 15 September, 1998 and dismantling of the stands will take place on Friday, 18 September, 1998 after 4.00pm.
All stands and equipment will need to be cleared from the centre by 11.00 pm

CONDITIONS

- Please complete the enclosed application form and return it to the Organisers together with 50% of the amount for the stand/area of your choice.
- For withdrawals and/or cancellations of exhibition space and/or sponsored items, received in writing prior to or on the following dates, the following cancellation fees will apply:

  30th June, 1998, 50% of total amount contracted for.

  After 30th June, 1998, 100% of total amount contracted for.
- Your investment in sponsoring is dependent upon the category of your choice. Please complete the enclosed application form and return it to the Organisers together with 50% of the amount for the item you wish to sponsor.
- On the 1st July, 1998 you will be invoiced for the remainder 50% balance of the amount owed on the item you are sponsoring and/or your exhibition space and your payment will need to reach our office by no later than 31st July, 1998 in order to ensure continuity of your involvement.
- Floor plan is subject to change without notice.