Upcoming USENIX Events

5th Conference on System Administration, Networking, and Security Symposium (SANS '96)
May 13–17, 1996, Washington, DC
Sponsored by the Open Systems Conference Board in cooperation with SAGE, FedUNIX, and the Escal Institute
Program Chairs: Rob Kolstad, BSDI and Michelle Crabb, NASA Ames Research Center

2nd Conference on Object-Oriented Technologies and Systems (COOTS)
June 17–21, 1996, Toronto, Canada
Program Chair: Douglas C. Schmidt, Washington University
Tutorial Program Chair: Doug Lea, SUNY Oswego
Camera-ready Papers Due: May 17, 1996

4th Annual Tcl/Tk Workshop '96
July 10–13, 1996, Doubletree Hotel, Monterey, California
Program Chairs: Mark Diskhans, The Santa Cruz Operation, Inc.; Mark Roseman, University of Calgary
Notification to Authors: April 16, 1996; Camera-ready Papers Due: May 28, 1996

6th UNIX Security Symposium—Focusing on Applications of Cryptography
July 22–25, 1996, Fairmont Hotel, San Jose, California
Sponsored by the USENIX Association. Co-sponsored by Uniforum in cooperation with The Computer Emergency Response Team (CERT), and IFIP WG 11.4
Program Chair: Greg Rose, Sterling Software
Papers Selected: April 15, 1996; Camera-ready Papers Due: June 10, 1996

10th Systems Administration Conference (LISA '96)
Co-sponsored by USENIX and SAGE, the System Administrators Guild
Program Chairs: Helen Harrison, SAS Institute; Amy Kreiling, University of North Carolina
Invited Talks Co-ordinators: Rik Farrow, Internet Security Consulting; Kimberly Trudei, Massachusetts Institute of Technology
Extended Abstracts Due: May 7, 1996; Notification to Authors: June 11, 1996; Final Papers Due: August 15, 1996

2nd Symposium on Operating Systems Design and Implementation (OSDI '96)
Co-sponsored by ACM SIGOPS and IEEE TCOS
Program Chairs: Karin Petersen, Xerox PARC; Willy Zwaenepoel, Rice University
Full Papers Due: May 7, 1996; Notification to Authors: July 30, 1996; Revised Papers Due for Shepherding: August 19, 1996; Camera-ready Full Papers Due: September 16, 1996

2nd USENIX Workshop on Electronic Commerce
November 18-19, 1996, Claremont Hotel, Oakland, California
Program Chair: Doug Tygar, Carnegie Mellon University
Extended Abstracts Due: July 16, 1996; Notification to Authors: August 5, 1996; Camera-ready Final Papers Due: October 7, 1996

USENIX 1997 Annual Technical Conference
January 6-10, 1997, Anaheim Marriott, Anaheim, California
Program Chair: John Kohl, Arriva Software
Invited Talks Coordinators: Mary Baker, Stanford; Berry Kercheval, Xerox PARC
Manuscripts Due: June 18, 1996; Notification to Authors: August 7, 1996; Camera-ready Papers Due: November 13, 1996

More information about USENIX and its events, access the USENIX Resource Center on the World Wide Web. The URL is http://www.usenix.org
Hype

I wonder if 1996 can top the hype of 1995. We had so many hyped products in 1995: a Windows launch that couldn't be beat, "the year of the Internet" (again), and, of course, who could forget Java, the enabler of all time. Of course, 1995 was also "the year of UNIX" for that marketplace and a number of other "year ofs," as well.

I watch TV once in a while to see the presidential candidates using all the same techniques that large companies use to promote their products. I recall a study unit in eighth grade where we learned the seven basic techniques of advertising. The level of transparency of all these basic promotions is quite amazing.

But I'm pinning my hopes on a revolutionary new way of promoting products (or people) that revolves around disseminating the features and benefits of a particular entity and educating potential consumers enough to enable them to make an informed decision. After dealing with the "fear, uncertainty, and doubt" method of sales -- in widespread use throughout the computer industry -- I'm thinking it's high time that we helped consumers of high technology understand exactly what it is they are buying. I'll let you know how it turns out.

Continuing on my quest from two months ago, I have commenced the health club regime and only disabled one body part so far (racquetball elbow). I had forgotten about the excitement of being on the court and competing. It really is a kick, and I'm having a great time with it. It also seems to be helping in all the ways they say: reduced stress, reduced blood pressure, reduced weight, increased self-esteem, all that good stuff. Yay.

RK

Letters to the Editor

Improvements for the Webmaster

by Jerry Peek

Dave Taylor's cool HTML scripts in the December 1995 ;login: have some good ideas! But they aren't written as efficiently as they could be. Because CGI scripts tend to be executed a lot, and need to give quick response, here are some ways to speed up the scripts. The techniques I'll show in this letter are good for anyone who has been using echo(1) to pipe a shell variable's value to a utility like cut(1) or sed(1). It's a lot more efficient to use the standard expr(1) utility.

Dave uses the following two lines to get the browser type:

```
x='"echo HTTP_USER_AGENT | sed 's:/s:/ :g'"
browser='"echo $x | cut -d -fl"'
```

The expr(1) program can do all that stuff in one step, without the pipe:

```
browser=`expr "$HTTP_USER_AGENT" : \(\([^"]*\)\)\.:.*`
```

USENIX BOARD OF DIRECTORS

Communicate directly with the USENIX Board of Directors by writing to <board@usenix.org>.

President:
• Stephen C. Johnson <scj@usenix.org>

Vice President:
• Eric Allman <eric@usenix.org>

Secretary:
• Lori Grob <grob@usenix.org>

Treasurer:
• Rick Adams <rick@usenix.org>

Directors:
• Tom Christiansen <tchrist@usenix.org>
• Dan Geer <geer@usenix.org>
• Andrew Hume <andrew@usenix.org>
• Greg Rose <ggr@usenix.org>

Executive Director:
• Ellie Young <ellie@usenix.org>

CONFERENCES & SYMPOSIA

• Judith F. DesHarvais
  Conference Coordinator
  USENIX Conference Office
  22672 Lambert Street, Suite 613
  Lake Forest, CA 92630
  Telephone: 714 588 8649
  FAX: 714 588 9706
  Email: <conference@usenix.org>

• Zanna Knight
  Vendor Displays, Marketing
  Email: <zanna@usenix.org>

• Daniel V. Klein
  Tutorial Coordinator
  Telephone: 412 421 2332
  Email: <dvk@usenix.org>

WWW URL: http://www.usenix.org

AUTOMATIC INFORMATION SERVER
To receive information electronically about upcoming USENIX symposia & conferences, finger <info@usenix.org> and you will be directed to the catalog which outlines all available information about USENIX services.

USENIX SUPPORTING MEMBERS

ANDATACO
Apanix Computer Services
Frame Technology Corporation
Matsushita Electric Industrial Co., Ltd.
Open Market, Inc.
Shiva Corporation
Sun Microsystems, Inc., SunSoft Network Products
Sybase, Inc.
Tandem Computers, Inc.
UUNET Technologies, Inc.

The closing dates for the next submissions to the next two issues of ;login: are April 10 and June 12, 1996.
FROM THE EDITOR

There might be a version of `expr(1)` that doesn’t understand the `[^/]` regular expression syntax. In that case, `echo(1)` and `sed(1)` can do the whole job:

```bash
browser='echo "$HTTP_USER_AGENT"
        | sed \1s:/\.*:/\1''
```

Both of my replacements also solve the problem where multi-word names (like “Spyglass Mosaic”) put only the first word (like “Spyglass”) into `$browser`. If you use my fixed versions above, quote the value of `$browser` when you use it. For example, this part of the script on page 39 should be rewritten:

```bash
case "$browser" in
  Cello)... ... ;
  "Spyglass Mosaic") ... ;
```

A single call to `expr(1)` can replace the four processes and three (!) pipes on page 38:

```bash
TOPMOST="'echo $REMOTE_HOST|rev|cut
          -d. -f1 | rev''
DOMAIN="'echo $REMOTE_HOST|rev|cut
          -d. -f1,2 | rev''
```

The replacements look like this:

```bash
TOPMOST="'expr $REMOTE_HOST:\1\(\.*\)\1''
DOMAIN="'expr $REMOTE_HOST:\1\(\.*\)\1\$\1
          \1 $REMOTE_HOST\1''
```

The second line is obscure enough to explain. If `$REMOTE_HOST` contains two or more dot (.) characters (as in `foo.bar.usenix.org`), `expr` will return the name surrounding the rightmost dot (like `usenix.org`). Otherwise, for a `$REMOTE_HOST` like `ora.com`, the first part of the expression doesn’t match, and `expr` returns the value of `$REMOTE_HOST` without editing.

[Editor’s note: Dave Taylor responds “Jerry is a scripting wizard, so my only comment is ‘thanks!’”]

Response About DCE

by Marc Wiz
<marc@frshaire.wiz.com>

[Editor’s note: Marc Wiz is not a pseudonym.]

I just finished reading Rik Farrow’s “Musings” column in the February `login`, and I must say that I take exception to some of Rik’s comments regarding DCE.

Rik starts out commenting on how Ram Sudama is biased because of his previous and current positions. It certainly seems to this reader that Rik has an axe to grind in regards to DCE.

Rik comments on having the opportunity to learn the ISO naming convention. This is necessary only if one wishes to use the ISO naming convention. From what I have seen the ISO naming convention is mainly used in intercell communications (two or more DCE cells talking to one another)

DNS is used only for cells to communicate with one another. The DCE cells I have dealt with use the X.500 services only for intercell. I have seen no cells that use X.500 for the local cell naming service.

Another comment regards centralized administration and how PC desktop managers and the companies selling them operating systems have been trying to centralize administration. I submit that this is unfair. It is possible to put the CDS (Cell Directory Service) and the security server on one system. At the same time for load balancing it is possible to have the CDS and security servers run on separate machines as well as replicating their respective databases for both load balancing and removing single points of failure.

Having the security data on one machine definitely qualifies to me as being centralized!

I also do not understand the thrust of Rik’s desire for centralization. It seems to me that we in the UNIX community have been trying to champion the distributed systems concept in one way or another and move away from the centralized mainframe paradigm. Have I missed the boat somewhere? Are we going back in time?

Much DCE administration can be done from any system that is a member of the DCE cell. I find this to be an advantage. Perhaps Rik should take a look at the OSF’s DCE 1.1 release. There are some interesting additions in functionality for performing administration of remote systems.

Rik also comments on replication. I do not know where Rik has been, but in my opinion replication is a good idea. Is Rik familiar with the phrase “single point of failure”? That is exactly what centralization gives you.

Rik also comments on going to UNIX Expo a couple of years ago and how the developers could not get the demo to run. As we all know a couple of years is a long time in our industry.

In all fairness, Rik should have checked the current status of DCE and the status of Transarc’s monitor known as “Encina.”

The next section of “Musings” dives into Java. I cannot comment on Java as I have not looked at any technical documents. All I have been subjected to is the massive hype going on in the mainstream and computer industry press.

I feel Rik is taking the opportunity to bash DCE again. Rik mentions how public key signatures are used for authentication instead of relying on a security server.
Rik does point out that there is no method for distributing public keys. DCE stores passwords in one centralized database, the security server. Rik talks about centralization of administration. Java as it stands currently does not allow this.

I see that Rik made no mention of the DCE API or what DCE tries to accomplish. It leaves me wondering how much time Rik actually spent working with DCE.

There is a learning curve associated with DCE in both the administrative and programming areas. But this learning curve is no excuse for a poor and in my opinion unfair review.

**Rik Farrow responds:**

Hi Marc:

Sorry I failed to communicate myself clearly. First, “Musings” is never a review unless I specifically state that I have a product I am looking at and provide contact information. I was, instead, editorializing, and making comments which might indeed be unfair. You have helped to educate me, but still leave the main question begging.

But let’s take your points in order. You mention that I have “an axe to grind in regards to DCE.” You’re right. DCE appears to me to be a big step away from the philosophy of UNIX. The UNIX system started out as simple and elegant, running in 128 kbytes of memory. Of course, it has grown vastly beyond that today, with a big chunk of that being networking code. But DCE adds yet another layer on top of UNIX and whatever operating system it sits over.

You contend that DCE is centralized. It is possible that I misunderstood the size of DCE cells – that is, that they are local, not widely distributed (say, across an organization). This would require many cells, each requiring administration. So now we have many cells which have to be managed so that we can support RPC services. This is in addition to managing the individual systems themselves.

How well does this compare to the UNIX philosophy? Have things become simpler and more elegant? Nope, we have added another layer of abstraction and system management to support another API. You can argue that it’s worth it just to have the DCE API. And if you want it to work reliably, and it provides an essential system, you’d better have redundant systems.

Two years is a long time in this business. DCE was over two years old at UNIX Expo, Encina was an important project, and I was simply amazed that such a critical software system as a transaction monitor couldn’t be made to work with the developers themselves (Transarc) working on one system, and OSF’s team working on the DCE server end.

And now, DCE has reached release 1.1. Same release level, next version number. This has the appearance of working rather slowly (especially when one watches Linux releases, but I know there’s no comparison here).

This is not a review. It’s my opinion. But I did ask for people to come forward with a completed, functioning DCE project which is not just a pilot program. You apparently have some familiarity with DCE (a lot more than me, I really don’t know everything). Can you provide me with some examples?

Just assume I’m from Missouri, and don’t believe things until I see what they can do. DCE has failed to receive wide industry acceptance in about a five year period. I think the “record speaks for itself,” as politicians are wont to say, but I am willing to be proved wrong.

Regards,
Rik Farrow
<rik@spirit.com>
Phil Zimmermann is Off the Hook

by Greg Rose
<Greg_Rose@sydney.sterling.com>

It was only two days after the last issue of \login\: was frozen when the US Attorney announced, out of the blue, that Phil Zimmermann would not be prosecuted. The press release said:

Michael J. Yamaguchi, United States Attorney for the Northern District of California, announced today that his office has declined prosecution of any individuals in connection with the posting to USENET in June 1991 of the encryption program known as “Pretty Good Privacy.” The investigation has been closed.

This is great news for Phil.

What does it mean for the greater scheme of things, though? Not much. The International Traffic in Arms Regulations (ITARs) are still in force. The fact that a single indictment is not happening doesn’t affect their validity one way or another. It doesn’t change the status of the other court cases I mentioned last issue.

It would seem that this lack of decisiveness may be part of the government’s goal. If the case had gone to trial, a far-reaching precedent might have been set (one way or the other). Instead, for three years, an individual has been under great pain, and others have had to fear the same treatment; but at the end of those years, little has changed. The fear is still there. Many people say that the regulation is unenforceable or unconstitutional, but few people are prepared to disobey it.

Quoting from the first of my articles in this series:

The government clearly wishes to crush Phil and send a strong message about making software available on networks, especially software they don’t like, even if the author takes significant care to discourage or prevent export. They wish to establish that the author is responsible for potentially illegal acts committed by others even without his knowledge or control.

This is the issue that USENIX believes is important and of interest to its members.

This really hasn’t changed. Phil may not have been crushed, but he’s definitely lost weight. No sane person would volunteer for the kind of treatment he’s gone through. (Sorry, Phil. I couldn’t think of a better way to phrase that.) The other efforts to overturn this regulation are still in progress, and the fear, uncertainty, and doubt surrounding the regulations remain, so even though Philip Zimmermann is off the hook, I don’t think I am. I have to keep writing these articles . . . although I think the frequency may decrease.

Congratulations, Phil.
1996 USENIX Annual Technical Conference Reports

San Diego, California

[Note: The following reports cover most but not all of the refereed papers and invited talks. Thanks to the reviewers: Peter Collinson, Doug Steel, Jerry Peek, and Bill Rigg. For summaries of the keynote address and some invited talks, see Rik Farrow’s “Musings” column on page 43.

If you wish to refer to the full papers, go to the USENIX Online Library on the World Wide Web URL: http://www.usenix.org, or order the conference proceedings by contacting office@usenix.org. If you would like to write a summary of a conference or symposium, contact login@usenix.org.]


File Systems Session

Summarized by Peter Collinson
<pc@hillside.co.uk>

Scalability in the XFS file system

Adam Sweeney, Doug Doucette, Wei Hu, Curtis Anderson, Mike Nishimoto, and Geoff Peck, Silicon Graphics Inc.

This was the first talk in an excellent file system session that immediately followed the keynote talk.

XFS is a new file system design for SGI’s IRIX that is intended to allow Silicon Graphics to fill the needs of its customers for the next ten years.

The new file system has four goals:

1. Provide access to at least a terabyte of data. SGI is seeing demands for file-systems of this size now and thinks that even larger file systems may be required in the future.
2. Provide very fast data throughput. XFS can deliver data at 500 megabytes/second.
3. Speed up crash recovery. Most people are not prepared to wait while fsck checks the disk.
4. Support large directories and large numbers of files.

Two key changes were made. First, files are no longer stored as a series of blocks linked with pointers. Files are placed contiguously on the disk. The base and extent of the file is stored. A file can consist of several (base, extent) pairs, but this is avoided if possible. To obtain immense file systems, the file system uses a full 64-bit block address.

Second, all the linear structures on the file system that store the metadata – inode tables, free space bitmaps, and the like – are replaced by B+ trees. A B+ tree is a...
balanced multiway tree that has inherent ordering. The balancing means that the tree does not get unnecessarily deep. The ordering means that the time spent searching the tree is logarithmic and not linear.

The key to a fast file system is fast space allocation procedures. Free space is stored in a B+ tree that records the base and extent of the free space. This means that it’s very fast to find an area on the disk that is the size that you want.

The semantics of the UNIX write system call provides a fundamental problem for file systems based on (base, extent) pairs because there is no preallocation of space on the disk. The kernel does not know how large a file will grow and cannot accurately allocate an appropriately sized contiguous area on the disk that is guaranteed to contain the whole file. To get around this problem, the file is written into the buffer cache in memory until no more space is left; then the whole file is written into a correctly sized area. This works well for small files; they often stay in memory and are written in one operation. If the process is still writing when memory becomes full, then the file is known to be large, and steps can be taken to allocate a large chunk of disk for it.

The B+ trees grow as they are needed, so the XFS file system allocates inodes dynamically. This is a huge improvement on the existing file systems that create a fixed number of inodes when the file system is created.

B+ trees are used to store directories. The filenames stored in the directory are hashed to a 4-byte value that acts as the key for the B+ tree. The benefit is that the system permits directories with millions of entries to be stored and serviced efficiently.

The other side to improving file system performance is to make improvements in handling the file system metadata. One of the biggest problems with the existing file systems for UNIX is the desire to write metadata synchronously providing a reliable file system that can be recovered easily in the event of a system crash. XFS avoids this need by writing a transaction log ahead of the change that is to be made to the file system. If the system fails over, then a replay of the transaction log will result in a clean working file system.

There are situations in which this transaction log could become a bottleneck, restricting file system performance. The performance is improved by exploiting parallelism, both in a multiprocessor sense and also by writing the log to a separate log device.

Adam’s talk ended with some discussion on the performance that has been achieved, which was summarized by a single slide: XFS kicks butt.

AFRAID – A Frequently Redundant Array of Independent Disks
Stefan Savage, University of Washington; John Wilkes, Hewlett-Packard Laboratories, Palo Alto.

The last talk in the file system session described a modification to the RAID5 disk management system. The paper was the co-winner of the best student paper award.

RAID level 5 disk arrays achieve high performance by striping data across several disks and accessing this data in parallel. To withstand disk failures, RAID5 arrays generate a parity block for each stripe and store it on a disk not used for the stripe’s data. If a single disk fails, the parity block can be used to reconstruct the data that was lost. Keeping the parity consistent leads to what is known as the "small-update problem." Small writes in RAID5 require three extra disk I/O operations to update parity, making these common operations much less efficient than for single disks.

The AFRAID system delays the writing of the parity information in a controllable way, meaning that a small write operation will simply place the data on the disk, and the parity update will be done later, when system demand on the disk array diminishes. How quickly the parity update is performed is configurable, so the user can control the trade-off between performance and the risk of data loss.

The AFRAID design arose because of three observations:

1. Modern disks are very reliable, much more so than the other components in disk arrays.
2. Many disk work loads are bursty, with long delays between client activity and these delays can be used to create parity values.
3. People are used to the notion that there can be a time limited exposure to risk of data loss.

When you buy a RAID5 system, you are quoted an availability number that is some hundreds of millions of hours. This is computed from the failure rate of disks and completely ignores the failure rate of the other hardware that surrounds the disks. When these other components are included, the overall availability is limited to two million hours, even when you have redundant controllers, batteries, fans, etc. A major contention of the paper is that the usual RAID5 equations don’t consider the whole picture, what the authors call the “end-to-end” availability. The paper goes through the mathematical logic supporting this view. So most of the extra I/O operations that are being done to obtain redundancy are actually worthless.

The idea was backed up by a number of simulations using some real-life work loads; for details, see the paper. Broadly, the work showed that the system could offer 42%
The Fast File System was implemented by Kirk McKusick for 4.2BSD. We have seen some papers in previous USENIX conferences that have tinkered with the algorithms that are used by FFS, and Kirk is not averse to messing with them himself. The work described in this paper analyzes a change that Kirk implemented for 4.4BSD Lite, but it was #ifdef'ed out because of a bug. The bug is now fixed, and it's possible to look and see whether the change improves things.

With the FFS, Kirk worked hard to maintain locality of reference. To minimize seek times on the disk, an attempt is made to place each directory and its files in the same cylinder group. A cylinder group is some area of the disk. The intention was that cylinder group also contained the metadata for the file: the inodes, block pointers, and the like. To improve the chances of a free block being in the desired cylinder group, the disk is intentionally run at 10% free.

When the kernel needs a block on the disk, the allocation code will supply the next contiguous block if it is available. If not, a "nearby" block is allocated. I will not go into the bunch of heuristics in the code that decides what "nearby" means. The new mechanism (the realloc algorithm) reallocates disk space when the buffer cache is flushed; the idea is that because you know how many blocks you want to write, you can achieve contiguous allocation by writing the blocks to an extent of free space that is the same size.

In order to examine the two different allocation methods, a test workload was generated. The workload can be replayed to artificially age a file system, supplying ten months worth of file updates. The test workload is initially applied to an empty disk so the results are reproducible. Some measure is needed; a "layout score" is computed for files by measuring what fraction of the files blocks are optimally located. These scores are aggregated into a layout score for the file system.

To check the validity of the workload, the simulation was run and a value of layout score compared with the old FFS system. Two benchmarks were used. First, sequential I/O performance was examined; files of varying sizes were written and read from the disks. The disk that used the realloc algorithm performed the standard FFS code in 21 out of

**A Comparison of FFS Disk Allocation Policies**  
**Keith A. Smith, Harvard University; Margo Seltzer, Harvard University**

This paper was the co-winner of the Best Student Paper award and added greatly to this interesting session on file systems.

Listening to the talk, I was reminded of a past experience. The paper was also about "why UNIX file systems perform less well with age." I was talking notes at an EUUG conference in Paris in around 1980 while a paper was presented that explained why the performance of the original UNIX file system quickly degraded with age. However, that paper was typical of its time: logical deduction based on observations of the algorithms. These days, it's great to see Science with a capital "S" being used for Computer Science. This paper is based on the same basic inspection of the algorithms, but the deductions are backed up with some observations, some considerable simulation and modeling—and we are all better for the result. I digress, I guess.

The Fast File System was implemented by Kirk McKusick for 4.2BSD. We have seen some papers in previous USENIX conferences that have tinkered with the algorithms that are used by FFS, and Kirk is not averse to messing with them himself. The work described in this paper analyzes a change that Kirk implemented for 4.4BSD Lite, but it was #ifdef'ed out because of a bug. The bug is now fixed, and it's possible to look and see whether the change improves things.

With the FFS, Kirk worked hard to maintain locality of reference. To minimize seek times on the disk, an attempt is made to place each directory and its files in the same cylinder group. A cylinder group is some area of the disk. The intention was that cylinder group also contained the metadata for the file: the inodes, block pointers, and the like. To improve the chances of a free block being in the desired cylinder group, the disk is intentionally run at 10% free.

When the kernel needs a block on the disk, the allocation code will supply the next contiguous block if it is available. If not, a "nearby" block is allocated. I will not go into the bunch of heuristics in the code that decides what "nearby" means. The new mechanism (the realloc algorithm) reallocates disk space when the buffer cache is flushed; the idea is that because you know how many blocks you want to write, you can achieve contiguous allocation by writing the blocks to an extent of free space that is the same size.

In order to examine the two different allocation methods, a test workload was generated. The workload can be replayed to artificially age a file system, supplying ten months worth of file updates. The test workload is initially applied to an empty disk so the results are reproducible. Some measure is needed; a "layout score" is computed for files by measuring what fraction of the files blocks are optimally located. These scores are aggregated into a layout score for the file system.

To check the validity of the workload, the simulation was run and a value of layout score compared with the real file system from which the workload was derived. In fact, the simulation behaved somewhat better than the real file system, but the graphs showed that the difference over time is constant. So the model seemed real enough.

The simulation tool was then used to investigate various aspects of the new realloc disk allocation code when compared with the old FFS system.

The overall results showed that the realloc code was a win; it resulted in 56% improvement in the layout of the disk. The next question was: what was this good for? Did the system perform better when better layout was achieved?

Two benchmarks were used. First, sequential I/O performance was examined; files of varying sizes were written and read from the disks. The disk that had used the realloc algorithm performed the standard FFS code in 21 out of
24 tests. The largest performance increase was 58%; the largest performance decrease was 10%.

Second, a “Hot File” benchmark was used. This tried to approximate the performance of the file system for a set of files that were in use. The files that had changed in the last 30 days of the ten-month period were selected. The files are read and written. The realloc file system read the files 32% faster than the system that used the original FFS algorithm. It wrote the files about 20% faster.

So the conclusion was that the realloc algorithm was a win. The idea of aging a file system provides an interesting tool that can be used to understand long-term file system performance.

**Web Session**

*Summarized by Peter Collinson*

*pc@hillside.co.uk*

**World Wide Web Cache Consistency**

*James Gwertzman, Microsoft Corporation; Margo Seltzer, Harvard University*

The first in a session on Web issues, this paper presented the results of a simulation study on the effect of different caching algorithms for Web pages.

The Web is known as the Internet “killer” application; it has made the world sit up and take notice of the Internet, but also hits network bandwidth because people are copying the same pages again and again over the wires. Caching seems an attractive way of reducing this traffic, assuming that the problems of cache consistency can be solved.

There are three main approaches to making sure that a reader gets the most up-to-date pages. The first method involves the use of an invalidation protocol and is the most efficient way of maintaining consistency. The other two methods will return stale data, but it was found that they would behave surprisingly well.

The first of these two weak consistency methods involves the use of a Time-to-Live (TTL) field supplied by the author on each page. The HTML header has provision for the author to supply a TTL, although few authors do, except perhaps for short-lived data like newspapers and the like. Caching seems an attractive way of reducing this traffic, assuming that the problems of cache consistency can be solved.

The second of these weaker methods is based on client polling. The cache polls the client for new copies of the pages using some time-out algorithm. The algorithm that seemed interesting was that used by the Alex FTP system; it is based on the assumption that old files are modified less frequently than young files. In fact, the older a file is, the less likely it is to be modified. If these assumptions are adopted, a cache needs to poll much less frequently for older files.

The results from the study showed that invalidation worked best for the caches directly mirroring hot pages. Here you want a zero level of stale data, and more importantly, there is total control over the server and the mirror. It’s much harder to make invalidation protocols work for the general case where you are serving both proxy and client caches. For these cases, the TTL or Alex methods seem the most practical today. They will work without modification to the server or client and so are instantly usable.

The results of a comparison between the TTL and Alex methods today are inconclusive. They perform almost identically. The analysis shows that allowing just a small amount of stale data will make these two protocols reduce network loading considerably. They provide network loads similar to the invalidation protocol if you are prepared to accept that as little as 1% of requests could return stale data.

At the end of his talk, James went over some suggestions that are “hints for designers.” Four types of information would be good to see in the HTML message. The first is to start making use of the “expire by” information supported by HTML. This is rarely used today; less than 1% of the files in the study supplied this hint to caches. The second header of interest is an “if modified since” field; this is how a cache could ensure consistency, saying give me this page if it was modified since a particular date. The final two header lines are “last modified” and the “date.” These would give direct input to the caching system.

In the answer to a question, James emphasized some points made earlier in the talk. Web data consist of several types of different objects, ranging from pages that change every time they are accessed, through regular HTML data, to large graphical images. Caching should be done differently for these types of data. Dynamic pages should not be cached at all. It was nearly always worth caching images because they represent a good percentage of the total traffic on the Web and change much less often than regular HTML pages. He feels that, as the Web develops, all HTML pages will become dynamic and hence will not be cached.

**A Hierarchical Internet Object Cache**

*Peter Danzig, Anwat Chankhunthod, Chuck Neerdaels, University of Southern California; Michael Schwartz and Kurt Worrell, University of Colorado, Boulder*

This talk discussed the Harvest object cache, a program that has been in use on the Internet for about 1.5 years. It’s designed to act in two roles: first, as part of a hierarchical
proxy cache for WWW objects; second, as a front end to a local httpd, where it acts as a Web accelerator. The program is publicly available and runs on all UNIX platforms.

There are now several top-level caches handling chunks of the US and other parts of the world. They are looking for other willing victims to join in with worldwide caching of Internet objects.

To plug into the network, you can install one or many caches on your site. Each cache in the hierarchy will independently decide whether to fetch a reference from the object’s home site or from other parent or sibling caches. The hard thing to get right is deciding what not to cache, and this is made more difficult because the underlying protocols do not help much.

The cache is designed as a single, nonblocking process using the BSD select system call to dispatch events. In the commercial version of the cache that is being developed, anything that blocks, like a directory operation, is done in a separate thread. The free version does have some blocking I/O calls. The process does not fork to avoid overloading the host system with many processes servicing events in parallel.

The Harvest cache comes with its own DNS cache, and when the DNS cache misses, it will perform its own nonblocking DNS lookups. A negative DNS cache is also implemented, so dead sites are blocked for five minutes. A negative object cache with a settable time-out also prevents fruitless looping looking for pages that are unavailable.

The Harvest cache is an order of magnitude faster than the CERN cache and presents orders of magnitude less load on the host. People using the CERN cache with load averages of 150 will go down to load averages of less than one-half.

The cache comes with routing protocol that allows local interworking. The design goal was that you should be able to easily supplement an overloaded cache by supplying an additional machine that is easily configurable to take over some of the load. Peter mentioned that a uniprocessor should be able to handle 150,000 URLs per hour with the new commercial cache, around half that with the current free version.

The second way to employ the Harvest cache is to move your local Web server to, say, port 81 and use the cache as a Web accelerator. It acts as a RAM cache for your local web server. For dynamic pages on your server, you can rewrite the URLs so that the lookup goes directly to port 81, avoiding the cache, or you can supply an expires header so that the page is not cached. The Harvest cache behaves significantly better than the CERN cache in this role. Taking an example from a figure in the paper: 60% of the time a CERN cache returns a hit in under 500 ms; 95% of the time a Harvest cache returns a hit in under 100 ms.

Binary and source distributions of the Harvest cache are available from <http://excalibur.usc.edu>.

Tracking and Viewing Changes on the Web
Fred Douglis, AT&T Research; Thomas Ball, Bell Laboratories

The Web session finished with a talk on a new set of tools from a team in AT&T Bell Laboratories. (The team has split as a result of the AT&T reorganization, and it was not yet clear into which part of AT&T the project would fall after the split.)

The tools, called AT&T Internet Difference Engine (AIDE), provide a way of monitoring a set of Web pages, detecting differences, and displaying the changes that have been made to a page. The idea is to provide a more accurate view of the changes that have been made to a page from the last time that you saw it, rather than having the author supply visual clues with “New” stickers and the like.

The front-end program, Htmldiff, displays the page differences in a visual way. The output looks like a regular HTML page, but it puts a line through the words that have been deleted and renders new text in bold italics. Actually, comparing HTML documents is nontrivial because the algorithm needs to be intelligent about HTML markup. The program views the pages that are compared as a set of sentences separated by markup. The same algorithm that is implemented in the standard UNIX diff command is used to perform the actual comparison.

The w3newer program is a rewritten version of w3new (that is available on the Web). The job of w3newer is to generate an HTML page filled with links telling users which pages from their browser hot list have changed. The program is a robot that reaches out to the URLs and checks whether the known modification date is more recent than the stored date. The known modification date is obtained from a variety of sources: the last dates of w3newer, the proxy server's cache, or the HEAD information provided by httpd for the URL. The program is also user configurable, so access to pages with known characteristics can be controlled. For example, pages known to change every day can be omitted from the search list, or pages from indexing services can be checked once a week.

The final part of the package is a program called snapshot. This program stores previous copies of pages so they may be compared with Htmldiff. The RCS version control system is used to provide a centralized database containing a history of page changes. The authors reasoned that they
could not persuade all the Web page providers in the world to store page versions, and it was inefficient to permit personal storage of page history. Storing all the history of a page means that users can see the differences in the page since they last looked at it. Also, they can see the page as it existed on a certain date. Users can compare different page versions from different points in time.

The system looked very nice; the output from HtmlDiff was easy to understand. The authors pointed to an example of a real situation where it was a win. USENIX had simply altered the date of the OSDI conference on its Web pages without flagging it. Many people could have registered the old date and would probably not have noticed that the date had changed on the page, even though they had seen the page before. AIDE flagged the change and made it visible.

There was a question from a member of the audience about whether the AIDE system fixed the wrong bug: USENIX should have flagged the change on its page. Then again, I have had a "NEW" sticker on one of my pages for three months or so; this is not new to people who have revisited my page in that period. It's a moot point. Read the paper and make up your own mind.

For more information on AIDE, see <http://www.research.att.com/orgs/ssri/people/aide>

OS Extensions Session
Summarized by Doug Steel
<doug@wg.icl.co.uk>

A Comparison of OS Extension Technologies
Christopher Small and Margo Seltzer, Harvard University

Christopher Small presented a study of several OS extension technologies and compared their performance and safety. These technologies fell into the following categories:

- those with no protection (C in the kernel)
- those which use hardware protection (C with upcalls)
- those which used interpreters (Java)
- those which used safe compilers (Omnware and Modula-3).

The criteria for judging the technologies was the performance impact of the extensions.

Small presented several examples: VM page eviction, MD5 fingerprinting, and Logical Disk. For each of these examples, each technology measured against a calculated break-even point.

His conclusions were that interpreters are too slow, and that upcalls and safe compilation were fast enough. He expects Java to become fast enough while remaining safe, and Omnware to become safer while maintaining its performance.

In the Q&A session he stressed that Java is good at what it is intended for, but is currently not fast enough. A question was asked about the cost of an upcall and he replied that it depends on what an upcall is. Another question was about sharing between extensions in different protection domains and the reply was that this would involve some data movement overhead.

An Extensible Protocol Architecture For Application-Specific Networking
Marc Fiuczynski and Brian Bershad, University of Washington

Marc Fiuczynski presented Plexus, an implementation of an extensible protocol architecture for the SPIN system (see <http://www-spin.cs.washington.edu>). The goals of Plexus are to enable protocol extensions which outperform conventional implementations without sacrificing safety.

Plexus supports extensions by defining a protocol graph where nodes and edges can be inserted by applications. Access is restricted to underlying interfaces and packets are filtered before they arrive at an extensions handler. This ensures safety.

He then presented two example of extensions using Plexus:

- an http proxy server which exhibits low latency
- a video server which avoid copy in/out overheads

He concluded that it outperforms existing implementations without sacrificing safety.

Linux Device Driver Emulation in Mach
Shanatanu Goel and Dan Duchamp, Columbia University

One problem with Mach is its lack of available device drivers. Shanatanu Goel presented a framework for allowing unmodified Linux device drivers to be used under Mach.

He described several issues that arise in emulating the Linux kernel environment within Mach. These include initialization, address space, memory allocation, synchronization, machine resources, and kernel interfaces.

They have managed to compile 34 Ethernet drivers and have tested ten; they also compiled 17 SCSI drivers. Performance was respectable when compared to the available Mach drivers. There are some remaining issues with the co-existence of Linux and Mach drivers.
Opinions on Recent Legal Decisions

Moderator: Ed Gould, Digital Equipment Corporation
Panelists: Dan Appelman, Partner, Heller, Ehrman, White, and McAuliffe; Philip R. Karn, Qualcomm Inc.; Mitch Dembin, Assistant US Attorney, Chief, Financial Fraud Section

Summarized by Bill Rigg
<bill.rigg@ptde.eds.com>

1995 was a good year for lawyers and the Internet; 1996 also promises to be a good year. With recent court cases and passed legislation, I would say it might be a great year for lawyers.

Dan commented on the civil side of law and the Communications Decency Act. This past year was a search for a standard of accountability on the Internet. Many questions have been raised; obscenity, indecency, liability, infringements of intellectual property rights are but a few. No effective enforcement standards mechanisms are in place. Current legislation is sure to be tried in the courts.

Dan reviewed a number of cases that were being tried this past year. At the heart of these cases was the question: who is liable in the chain of information dissemination? Is it Internet service providers? Bulletin boards? News servers? System administrators? Those that post to various bulletin boards or newsgroups? Those that read what might be posted? Just to confuse the issue, what about email and anonymous/pseudoanonymous remailers? The constitutional rights of all concerned also colors these outcomes. These issues alone will keep the lawyers and courts busy for years to come.

Phil jumped into the encryption debate. He got caught up in this battle by asking the State Department if it was okay to export Bruce Schneier’s excellent book Applied Cryptography and a floppy disk containing the source code listed in this book. This often humorous discussion revolved around the fact that the book complete with source code is exportable. However, the same information in electronic format is a “defense article under category XIII(b)(1) of the United States Munitions List” and is not exportable. For a thorough review of Phil’s situation, stop by his home page at: <http://lorien.qualcomm.com/people/pkarn/index.html>.

Mitch, a prosecuting attorney for the US Government, started his presentation with a hearty “greetings from the enemy.” Although he understands the humor that some government actions may bring, Mitch presented the sad truth that his inputs were from the victims of crime, and these questions must be answered: was a crime committed? Is there any evidence? Is it possible to prosecute? Recourse is needed when a crime is committed. Computers and crime create a number of concerns, from records of a crime being stored on a computer to the computer as a vehicle of crime (i.e., to transmit threats, and various types of fraud, and, of course, hacking). Due to the international scope of the Internet, prosecution can be hampered by jurisdiction disputes along with constitutional concerns such as freedom of speech and invasion of privacy.

Protocols Session

Summarized by Bill Rigg
<bill.rigg@ptde.eds.com>

FLIPC: A Low Latency Messaging System for Distributed Realtime Environments

David L. Black, Randall D. Smith, Steven J. Sears, and Randall W. Dean, Open Software Foundation Research Institute.
Presenter: David Black

FLIPC is a messaging system to support realtime applications utilizing high-performance communications hardware. The following are characteristics of FLIPC:

- support for multithreaded applications
- support for threads and message streams of varying importance
- performance optimization for medium-sized messages (50–500 bytes)
- support for explicit control of resource allocation

In order to relieve the main processor(s) from coping with communication functions, FLIPC exploits the programmable controllers that are found on many interfaces to high-speed interconnects and networks. FLIPC utilizes wait-free techniques for utilization between the messaging engine and applications. These interface controllers are programmed as operating system components, as opposed to part of the applications.

The following are the primary components of FLIPC:

- the messaging engine (the hardware and software that moves messages between nodes)
- a fixed size nonpageable communication buffer that is shared between the messaging engine and all applications that use FLIPC
- an application interface layer that provides formal interface to applications and hides the data structures in the communications buffer (This consists of both a library and header file[s].)

The communications buffer, located in shared memory and containing all the memory resources for messaging, is key. Only when synchronization actions cannot be accomplished...
in the communications buffer via state is the operating system kernel involved. FLIPC utilizes multiple endpoints per communication buffer to support application needs. Different endpoints for different threads avoids contention for communications resources.

Cache was discovered to cause two performance problems, which together presented a performance "hit" of ~2x. On a multiprocessor Paragon test machine, it was found that the caches did not implement cache residency for multiprocessor locks, and there was a number of cache invalidations due to a false sharing of variables written by both the application library and the messaging engine. Once these concerns were optimized, latency was improved by 15 μs.

The message latencies for the FLIPC implantation on the Paragon range from about 15.5 μs to 17 μs. The corresponding standard deviations range from 0.5 μs to 0.65 μs. For message sizes of 96 bytes and above, the latencies obey the following equation:

\[ \text{Latency} = 15.45 \text{ μs} + 6.25 \text{ ns/byte} \]

Conclusions:

- Logical separation of messaging engine and OS kernel allows the messaging engine to be implemented on the communications controller.
- Asynchronous messaging interfaces and wait-free synchronization decouple the communications controller from applications.
- The shared communications buffer allows the OS kernel to be bypassed.

An Analysis of Process and Memory Models to Support High-Speed Networking in a UNIX Environment

B. J. Murphy, Cambridge University; S. Zeadally and C. J. Adams, University of Buckingham
Presenter: Brendan Murphy

Two items were addressed in providing operating support for multimedia to reduce delay and jitter. These recognized that a limit on application – network throughput is the cost of copying data in relation to the cost of processing. One was the zero-copy memory model design for a network interface. The second was a model where data are streamed between device drivers without crossing the user-kernel boundary.

The customary UNIX kernel requires two data copies in both transmit and receive directions; one copy is required between application and kernel, and an additional copy is required between kernel and network controller. If the controller has memory that can be mapped into host address space, then it is possible to eliminate the kernel-controller copy. When transmitting, the network subsystem copies data from application address space directly into the controller's memory. While receiving, the network controller assembles an incoming packet in controller memory. Performance may also be improved if the application-kernel copy is eliminated. Page remapping (in which pages are unmapped from one protection domain and mapped into the other) and copy-on-write (in which pages are shared between protection domains and copying is delayed until a process in one domain attempts to write to a shared page) are among the techniques employed. Each approach (eliminating kernel-adapter copies and eliminating user-kernel copies) provides single-copy transfers between application and network. Together with the appropriate hardware, zero-copy transfers are achievable.

Context switches can also limit throughput. Overhead involves the saving of one context (register contents, memory management information, etc.) and the restoration of another. Because traditional UNIX systems have no bounds on context switch latency, this may also have an impact on the delivery of data. Kernel-level or hardware-level streaming can help avoid context switches. Because many applications transfer data between a network controller and another device without manipulation, kernel-level streaming becomes attractive. This can reduce the amount of copying involved and eliminates context-switch overhead, improving CPU availability and reducing jitter.

Zero-Copy TCP in Solaris

H. K. Jerry Chu, SunSoft, Inc.

Solaris now utilizes virtual memory remapping and checksumming from the networking hardware. This eliminates data-touching overhead. High data throughput is required because of the advancement of high-speed networks and multimedia applications. Data copying and checksum often dominate processing time, with multiple data copy and separate data checksum operations performed on each byte of a data packet being common. Per-packet cost is roughly constant for a given network protocol, regardless of packet size, whereas the per-byte cost is determined by data copying and checksumming overhead. Because Solaris is a widely used operating system, SUN decided not to alter any interface, at either the application or device driver level.

The following are the different zero-copy schemes:

- user accessible interface memory
  Requiring substantial changes in software and complicated hardware support, this scheme utilizes network interface memory that is accessible and premapped into user and (possibly) kernel address space.

- kernel-network shared memory
  The operating system manages the interface memory and uses direct memory access (DMA) or program I/O to
move data between the interface memory and application buffers. Software support may be complicated. Kernel networking buffer management code must be enhanced to support the interface memory, which is co-managed with the device driver.

- user-kernel shared memory
  This approach utilizes DMA to move data between the network interface and shared memory. However, it defines a new set of APIs (application programming interfaces) with shared semantics between the user and kernel address spaces. Application compatibility will become a major problem.

- user-kernel page remapping + COW (copy-on-write)
  This scheme uses DMA to transfer data between interface memory and kernel buffers and remaps buffers by editing the MMU table to give the appearance of data transfer. By combining it with COW technique on the transmit side, it preserves the copy semantics of the socket interface.

Efficient page remapping and COW operations relative to memory copy are what determine the performance of zero-copy versus single copy. Significant performance gains can be achieved with the least number of changes by utilizing virtual memory page remapping and COW techniques.

Forming a More Perfect Net Governance

Carey Eugene Heckman, Adjunct Professor of Law, Stanford Law School, Co-director of Stanford Law and Technology Policy Center
Summarized by Jerry Peek
<jerry@ora.com>

The Net may seem like a robust technology that can evolve to meet new needs and challenges. But, Professor Carey Heckman says, “The rapidly growing use of the Net, and the awareness of its existence, has awakened previously sleeping giants. . . The wondrous opportunities and benefits of the Net are now in grave, grave danger.” His talk had five parts.

What’s wrong?
The Net isn’t the cooperative place it used to be. But the spamming, cancelbots, impersonations, and other obvious problems aren’t the worst threat.

One major threat is government bureaucrats. They see the Net chaos and, instinctively, press for strict regulation. Major international organizations – the Group of Seven (G7), the UN, the European Union, and others – are looking into new regulations for the Net. But many of those bureaucrats clearly don’t understand the technology.

Another threat is big corporations coming onto the Net and, by their sheer size, setting de facto standards that can be very hard to reverse. For example, Heckman said, “Look how one company’s operating system design decisions in the early 1980s to some extent still cripple personal computer applications and software development.”

Right now there’s no way to make consistent “rules of the road” that are realistic technically and also take into account the needs of a broad spectrum of interests. Also, government bureaucrats in one jurisdiction may make decisions that can affect the entire Net. (I would add that we’ve seen this more and more in recent events.)

Why worry?

Maybe governments will do the right things, and technical tools will evolve to meet the challenges.

The speaker didn’t think this was very likely. The increasing chaos will drive people into subgroups, walled off from each other so they can get something accomplished without all the noise and intrusions. We’ll lose many of the incredible possibilities if we don’t avoid a “Mad Max” type of scenario in the Net community.

What do we need?

- fair rules and fair processes – for both developed and developing countries
- a wise sense of ownership: rules that are adopted, not imposed
- participation that’s inclusive but not so broad that it’s unworkable (For example, it’s easy for a meeting with hundreds of decision-makers to get bogged down.)
- technologically informed rules, not rules that bear no resemblance to how the Net really works
- intellectually grounded rules, not just convenient or pragmatic, but well thought out
- adaptive and flexible rules – the technology can change rapidly, and the rules need to be able to change with it
- the ability to work within existing traditions and the government framework – because those won’t go away
- enforceable rules and transparency (When someone isn’t following the rules, we can observe it.)
- simple rules that are easy to understand

What should we do?

There won’t be a world government or a police force for the Net. We need to form Net governance systems.
A governance system isn’t a government. A governance system is “social institutions or sets of rules guiding the behavior of those engaged in identifiable social practices.” A government is “organizations or material entities established to administer the provisions of governance systems.” We can have governance systems without an actual Net government.

There are “many examples” of governance systems that work. One is GATT, the General Agreement on Tariffs and Trade. Another is COCOM, a group of nations that discuss export controls. [GATT has been in the news. I found a lot about COCOM by searching AltaVista, <http:liwww.altavista.digital.com>.]

Governance systems aren’t easy to establish. Once established, they can fail. At least eight things make them succeed:

1. focusing on key issues, but issues that are broad enough to be able to negotiate trade-offs and make a deal
2. issues at stake lending themselves to dealing in a “contractarian mode,” an ongoing system that evolves as the issues change (For example, the system for trade on the Net and tariffs for it will have to change as Net commerce develops.)
3. no obvious perpetrators and victims, no clear winners and losers (Participants must be willing to do what officials agree upon; they probably won’t be willing if they feel they have little chance to win. For example, in the US football Super Bowl, neither team knows the outcome ahead of time; they’re willing to play because they trust that officials will help ensure that the game is fair.)
4. reaching a consensus, buying into the system, so that the Net users feel the process is fair
5. simple arrangements (Complex setups tend to fail.)
6. transparency to verify compliance with the rules (Heckman acknowledged that many people fervently believe in anonymity on the Net. He says it’s hard to enforce the rules unless there’s some ability to find out who’s been on the Net and what they did.)
7. crises to solve: immediate, concrete problems (If nothing seems urgent, people aren’t as likely to try hard enough.)
8. effective leaders who can frame the issues

What can we (as computer professionals) do?

We need more and better data. Referring to his point in the previous section, Heckman asked, “How do you know you’re in a crisis if you have no data?” Having data lets you challenge proposals that are wrong. Data also let you identify emerging customs and norms; the governance system can ratify these as a standard for conduct on the Net. (In the US and a lot of the world, there’s a history of making established practices into law.)

We can pull together ideas and principles to make a framework for the way that the Net should operate. Lately, people who are hostile to the Net (I can think of some Senators, for example) have been making that framework. If we want to see the Net survive and flourish, we need to take the initiative.

We can educate decision makers about the technology. People who understand how the system works need to communicate with policy makers (and with the general public) so they can make informed decisions.

CitySpace: Come Build It Yourself – A User-Extensible Virtual Environment for Real-time Play
Zane Vella, Coco Conn, and Chris Cederwall, CitySpace
Summarized by Jerry Peek
<jerry@ora.com>

In 1993 the “Information Superhighway” was a new idea to the public. That year, when Zane Vella and Coco Conn met at the SIGGRAPH conference, they founded a unique project to get kids involved on the Net. CitySpace has been building a virtual city on the Internet.

After SIGGRAPH, the momentum built as teams of people – kids, mentors, and educators – worked together. They took the project to computer conferences and other technical events; attendees wanted to know about the software and how people used it. More good things happened, though, when CitySpace went to science museums and other places with fewer software professionals: more kids got involved in the project. Now individuals and groups are designing the city from classrooms, labs, and museums: fitting together stills, videos, and sounds. For three months in 1995, CitySpace was set up interactively at both the Ontario Science Centre in Toronto and the Exploratorium in San Francisco; participants worked together in simultaneous workshops, across the Net, to build their Net city.

CitySpace is about more than networking and graphics tools. One of the main goals is social interaction. Young people, ages 10–16, cooperate to build the city from across the Net. The project helps them learn to communicate and solve problems. They design their own structures, but they also use building blocks contributed via anonymous FTP from many other classrooms. The kids work together on high-speed networks with 3-D modeling and graphics tools, video communication software like CU-SeeMe, and sounds. There’s an interactive theatre with a Silicon Graphics Onyx Reality Engine II supercomputer, a large-format data pro-
jector, and individual workstations for graphics and modeling.

Coco told one story that made it clear how much the kids turned their environment into real life. One worker in Boston was building a house; she had chosen some furniture from the anonymous FTP site. Another worker in Los Angeles was building a casino; he saw the same furniture and took it first. She was furious; and she let him know over interactive video, with CU-SeeMe, that she wasn’t only unhappy about the furniture. She thought his casino would bring crime to the city. The two of them discussed it. He eventually agreed to donate the proceeds from his casino to the project; they “shook hands” over the Net to seal the deal.

The speakers at the USENIX session showed a video tour of parts of the city: streets with all the features you’d see in a “real” city (and some you wouldn’t!), roads, building exteriors and interiors, trees and towers. You can see the city and get more info about the project on the Web at <http://cityspace.org> or by sending email to <info@cityspace.org>. CitySpace has been supported by the USENIX Association and Silicon Graphics Inc.

Selling Stuff that’s Free:
The Commercial Side of Free Software

Moderator: Mary Baker, Stanford University
Summarized by Jerry Peek
<jerry@ora.com>

The panelists:

- Michael Tiemann, a founder of Cygnus Support. They have provided contract support and consulting for free software since 1989.

- Bob Bruce, founder and owner of Walnut Creek CD-ROM, a company that distributes software on 80 different CD-ROM titles. They also distribute free software from ftp.cdrom.com and support many free software projects, including FreeBSD and the Linux Slackware distributions, SIMTEL, and the X Consortium.

- William H. Davidow of Mohr, Davidow Ventures, who used to head Intel’s Microprocessor Division and is now a venture capitalist. He says he’s not at all interested in subsidizing free software; he is interested in anyone who converts free software into a paid-for service. “Giving people free software is pretty much like giving them the first shot of any drug,” he quipped.

- Linus Torvalds, a student and researcher at the University of Helsinki, who created the Linux operating system and is an author of free software. “I’m the only person here who isn’t making any money at all” [from free software – JP].

The panel covered a lot of topics. Here are some of the more interesting comments, questions and answers. [The audio-tape I made of the discussion was poor. I haven’t attributed comments if I couldn’t be sure who made them, and I’ve used quotation marks only when I could hear word for word. – JP]

Panelists seemed to agree that the “free” in “free software” refers to freedom, not to the price. What free software vendors are selling is convenience, service, and/or support. For instance, Walnut Creek sells CD-ROMs that are fully-indexed and ready to use; you don’t need to search the Net or wait for the software to download through a modem.

When you buy commercial software, you buy “something that’s buggy.” To get fixes, you need to wait for (and often pay for) an upgrade. The vendor decides what bugs to fix and features to add by putting all the user requests in a pot, stirring it up, and choosing a mix that fixes serious problems and will help them beat the competition. Part of the reason vendors make upgrades is to sell them to you. Individuals have almost no hope of getting a specific problem solved.

When you buy supported free software, you’re buying support from someone who wants to help you personally.

Software is often written by programmers for their own use. Distributing their code as free software is irrelevant to them.

**Question: How do the panelists feel about shareware, crippleware, etc.?**

**Linus:** He really dislikes “guiltware.” You’ll find a useful small program that isn’t useful enough to pay for, especially because cutting a check to pay for the software (in another country’s currency) can easily cost as much as the software itself. He could see himself writing commercial software or accepting money (for free software) from people who wanted to send him money. For Linux, he used the GNU public license, partly because he felt indebted for the gcc compiler. That was a moral decision, not one that the FSF imposed on him.

**Bob:** There’s only so much room for software on Internet archives, CD-ROMs, etc. Software with lots of restrictions is less likely to be distributed.

**Michael:** “I really hate (software that’s) ‘free for noncommercial use.’ [applause] . . . It is extremely frustrating to see 13 programs that all provide some sort of VRML browsing capability and all of them are free for noncommercial use . . . The number 13 or 14 is important because there’s really only going to be one or two winners . . . That means that there’s going to be 10 losers, which means that there are ten people whose time really has been wasted.”
The success or failure of free software is more democratic than commercial software, which is controlled by the marketplace: the success of free software is controlled by the (skill of its) programmers.

**William:** There's a lot of benefit in starting with base software that's completely free. If someone chooses to use the software in that form, that's fine. If a programmer enhances that free software and charges for it, people can pay for the extra value and support they get.

**Mary:** In places like comp.sources on USENET, everyone "takes according to their need and gives according to their ability." What about putting out software just because it might be of some use to someone else?

**Bob:** What his company is doing is providing other channels to get that same free software to users who need it.

**Linus:** Having commercial services distribute and support Linux is fine. It gives users more choices.

**Michael:** "This question touches on the difference between means and ends. People who hold free software as a moral high ground are really interested in the end: people who want to live in a world which has particular properties. . . . There are also people who (see free software as) the means. . . . We see free software as an opportunity to crack into markets that are fundamentally not well served by proprietary alternatives. Virtually every single customer of ours . . . built some kind of proprietary system or product. We do not feel that we made the world a worse place by saving them money and helping them produce better products."

**Question:** What effect, if any, would paying royalties to the software author have on the software?

**Bob:** Most of the profit he makes doesn't go back to the authors. Walnut Creek's model doesn't give money to people who write one small freeware package. His company does donate to the Free Software Foundation and other organizations. Walnut Creek pays authors who have substantial distributions, like a Gigabyte, who work directly with Walnut Creek to make the package more compatible with CD-ROM media.

**Michael:** As a service business, what Cygnus is providing is fuller value, not factor value. Factor value is developing a product that's proprietary itself. Fuller value is promising that your product is great; people pay you to make that true in the future. About paying authors -- in this model, there's not compensation to give to people who've already made their contribution: the software itself. But there are ways to compensate people who will be contributing: salary, contracts, and so on. They also have a matching fund for customers who want to contribute to the Free Software Foundation. Cygnus will match their donations.

**Question:** Aren't you both scam artists because you haven't paid Linus for his efforts?

**Bob:** "I haven't paid Linus anything, but I also haven't taken anything from him. I mean, I'm using his software, but that's not taking anything; he's giving that away for free. . . . If he's willing . . . to do something particular, such as writing a better CD-ROM installation program, I would be happy to pay him."

**Michael** [tongue in cheek - JP]: "Bill is being blinded by moralistic issues and he does not see the investment opportunities." [laughter]

**Question:** Some free software authors make their software freely redistributable so that lots of people will use it. Yet there are many more copies of commercial software than free software being used. Isn't that because money from software sales is used for marketing it? Making software free has reduced its distribution, not increased it.

**Michael:** That statement is too general. . . . There are a couple of stunning free software success stories -- like TCP wins, OSI loses. It's true that TCP is sold as a commercial product. But the fact that the reference implementation is free software was a key enabling factor in the growth of the Internet.

**William:** Look from the customer's point of view. For a technically sophisticated customer, free software is cheap and convenient. An unsophisticated customer will go for convenience, seeing an ad in PC Week and buying the product. . . . for this person, it's a cost/benefit win. One reason that there are so many copies of Windows is that so many users see it as a cost/benefit win.

**Linus:** For him, the number of people using Linux was never the issue. He wanted people who were in his position -- as a student with not much money -- to be able to use Linux for free. He didn't care about people who already had access to UNIX.

**Bob:** Free software isn't always friendly to naive users. The programmers' primary motivation isn't to sell as many copies as possible; they want to build a tool they can use.

**Comment:** Here's another way to make money from free software: My work on free software has gotten me job offers from companies that know what I've done and are impressed.

**Question:** A company always has to worry about what other companies are doing to be competitive. If you're starting a
company, is there any extra threat posed by free software, written by (people) in their free time?

Bill: It always bothers you if someone's giving away what you're trying to sell. [laughter] But you also have to talk about share. If there are a million customers, and only 100 of them know about the free product, it's not hard to whip them with a product that costs a lot of money. Once the free product tries to get on the distribution channel, it's a whole new game: try getting on the shelves at CompUSA with a free product. The Internet is starting to change that model, though. Now, in the Information Age, you don't have to have a physical product to sell it. That's one thing that could make free products a real threat to commercial products.

Performance Session
Summarized by Doug Steel
<doug@wg.icl.co.uk>

A Performance Comparison of UNIX Operating Systems on the Pentium
Kevin Lai and Mary Baker, Stanford University

Kevin Lai's presentation was motivated by his need to choose a Pentium-based system to use for operating systems research. His main candidates were LINUX, FreeBSD and Solaris. The main features he was interested in were:

- performance
- reliability
- availability of kernel source
- technical support and availability of drivers
- application support
- large user base

His benchmarks were based on John Ousterhout's microbenchmarks and the Modified Andrew Benchmark. He concluded that FreeBSD performed reasonably on all benchmarks, Linux and Solaris ran well on some but not on others. Linux performed well on system calls, context switching, and pipe bandwidth, but did poorly on the networking benchmarks, although it has improved significantly since the tests were performed. Solaris performed poorly on the system calls, context switching, and pipe benchmarks, but well on the large file benchmarks. His eventual choice was Linux for reasons other than performance (kernel source availability and support).

Q: Why choose Linux when it has poor network performance?
A: Not particularly important for his research (Mosquito-Net).

Q: Why choose a Linux NFS server for the NFS client benchmarks?
A: It was better than the Sun they originally used.

Imbench: Portable Tools for Performance Analysis
Larry McVoy, Silicon Graphics; Carl Staelin, Hewlett-Packard Laboratories

Larry McVoy presented the latest implementation of his benchmarking toolkit 1mbench. He illustrated his talk with a mass of data that he has gathered using this suite of tools. 1mbench has evolved from Larry McVoy's experience and measures the performance of the total system (operating system and hardware). He mentioned that Intel used 1mbench when designing the P6, and that it can be used to predict the critical path in a system.

The main results he presented were memory system latency and context switching performance of several systems using various hardware (including P6, UltraSparc, MIPS R10000, DEC ALPHA) and software (including Linux, Solaris, and IRIX).

The memory latency benchmark was based on walking a linked list (using p = *p) and varying the stride and size of the list. He pointed out that cache size and latency were obvious from the results produced. He concluded this section by saying that Intel's P6 was catching up with the workstation vendors and that HP's memory system was very good.

The context-switching benchmark was based on using a ring of processes connected by pipes. The variables were the number of processes and the footprint size of the processes. This benchmark was very system dependent, stressing both the OS and the hardware.

He quickly commented about file system benchmarks, saying that "Most either cheat or suck."

Q: This is not a strict apples-to-apples comparison.
A: True, but it can show differences.
Q: Can this explain application performance?
A: Not completely, but it can highlight bottlenecks.

Process-Labelled Kernel Profiling: A New Facility to Profile System Activities
Shingo Nishioka, Atsou Kawaguchi, and Hiroshi Motoda, Advanced Research Laboratory, Hitachi Ltd

Shingo Nishioka presented Pkprof - a solution to the kernel profiling problem of being unable to determine which part of kernel execution time is related to which specific user process.
**USENIX NEWS**

Pkprof implements a profile buffer for each user process. This allows the profiler to know which user process to attribute kernel time to. Interrupts present a problem since it is not immediately apparent which process an interrupt relates to. Pkprof solves this by using the buf or mbuf structure as an identifier for such an asynchronous event and storing this in a temporary profile buffer. When control returns to the top half of a driver the receiver process can be identified and the profile modified accordingly.

The processing of asynchronous events involves a high overhead. The overhead of pkprof is 36% of kernel time compared with kprof which takes 30% of kernel time.

For additional comments on the San Diego Technical Conference, see Rik Farrow's "Musings" on page 43.

**1996 Lifetime Achievement & STUG Awards**

At the 1996 USENIX Annual Technical Conference in San Diego, California, Steve Johnson, President of the USENIX Association, presented the following awards:

The USENIX Lifetime Achievement Award recognizes and celebrates singular contributions to the UNIX community in both intellectual achievement and unparalleled service. The 1996 Award was presented to the Software Tools Users Group. The principal recipients and "keepers of the flame" are Dennis Hall, Deborah Scherrer, and Joseph Sventek. The originators and key inspiration are Brian Kernighan and P. J. Plauger.


Before the general availability of UNIX, the Software Tools project popularized a new vision of operating system software, offering a bridge to portability and power for those limited by proprietary operating systems.

With its extraordinary focus on building clean, portable, reusable code shared amongst multiple applications and runnable on virtually any system, the Software Tools movement established the tradition of empowering users to define, develop, control, and freely distribute their computing environment. The contributions of STUG in retrospect can be seen to have been vital.

Past recipients of the USENIX Lifetime Achievement Award are the Computer Science Research Group at the University of California, Berkeley, for producing the UNIX BSD releases; Van Jacobson and Mike Lesk for their contributions to networking technology; and Tom Truscott, Steve Bellovin, and Jim Ellis for their work in creating USENET.

Simultaneously, the USENIX Association, acting on behalf of the Software Tools Users Group (STUG), presented the Software Tools Users Group Award to Michael Tiemann.

Michael Tiemann's work in C++ led to fundamental contributions to the GCC, the GNU C Compiler, which has had an unparalleled influence upon the availability of efficient and standard code on a vast number of hardware platforms. GCC has provided a development base for thousands of projects.

The Software Tools User Group Award recognizes significant contributions to the general community which reflect the spirit and character demonstrated by those who came together in the Software Tools User Group. Therefore, recipients of the Software Tools Award exhibit one or both of these traits in a conspicuous manner: a contribution to the reusable code-base available to all, or the provision directly to users in a widely-available form of a significant, enabling technology.
USENIX Allocates $300,000 to Student Outreach Programs

Recognizing the importance of reaching the student community, the USENIX Association’s Board of Directors allocated $300,000 in 1996 to expanding its existing student outreach programs and creating new ones.

“UNIX and its derivatives have been widely studied in universities for decades, and many of these students have gone on to become major contributors to USENIX and to the computer industry,” said Steve Johnson, president of USENIX’s Board of Directors. “We are happy to be able to increase significantly our level of support to students to further encourage their participation in our activities.”

Student programs fall into five categories. Here’s a summary of the programs, and details on how to get involved:

- Best Student Paper Awards. USENIX seeks to increase the number and quality of paper submissions to its conferences. A cash prize of $1,000 is awarded for the best student papers at our Annual Technical and LISA conferences. $500 prizes will now be awarded at our single-topic conferences and symposia. Single-topic conferences for the remainder of 1996 include object-oriented technologies and systems, UNIX Security, Tcl/Tk, operating systems design and implementation, and electronic commerce.

If you or your colleagues are doing work in these areas, please consider submitting a paper to the program chairs. Details of how to submit are included in each Call for Papers in this newsletter (pages 60-69), and at the USENIX Web site, <http://www.usenix.org>. The program chairs will be able to answer any questions you might have about submitting and will guide you through the process.

- Student Stipends. The USENIX student stipend program covers travel, living expenses, and registration fees to enable full-time students to attend USENIX conferences. USENIX expects to make over 100 grants totalling $90,000 in 1996.

To apply for a stipend, read comp.org.usenix about 6-8 weeks prior to a conference for detailed information. The information is also on the USENIX Web site, or contact Diane DeMartini at USENIX <diane@usenix.org>.

- University Outreach Program. In exchange for complimentary membership and free conference registration, Computer Science department faculty and staff on various campuses distribute USENIX materials to their students, maintain a library of conference proceedings, answer questions, and spread the word about USENIX’s activities. USENIX would like to have representatives on 100 campuses by the end of the year. If you or someone you know would like to represent USENIX on your campus, please contact Diane DeMartini for the details: <diane@usenix.org>

- Academic Achievement Awards. USENIX plans to allocate $100,000 to establish a scholarship fund for graduate and undergraduate students. A committee was formed to make recommendations on how the Association might launch such a program. Suggestions are encouraged and welcome. Please send them to Diane DeMartini, <diane@usenix.org>.

- Pre-college Programs. In addition to encouraging college-level students, USENIX funds several pre-college level student programs. To obtain a grant, a proposal must be submitted to and approved by the USENIX Board of Directors. Proposals should contain a detailed description of the program’s goals and objectives, how it will be executed, and a detailed budget. Send proposals to the Executive Director, Ellie Young, <ellie@usenix.org>. Programs should be broad-based, rather than local or regional. Some of the projects USENIX has supported are:

  - a workshop and network event at the Supercomputing ‘95 conference.
  - the CitySpace Project, a series of focused pre-production workshops exploring Internet communications, three-dimensional modeling, as well as fundamentals of system administration and maintenance for students between the ages of 10-16.
  - the annual USA Computing Olympiad, which creates and distributes a set of computer problems that help high school students measure their algorithmic computer programming skills. Various rounds are conducted over the summer, and approximately 15 students train and compete for one of five places on the USA Computing Olympiad team. The team is then sent to the annual International Olympiad in Informatics to compete against other teams.
USENIX Board Meeting Summary

by Ellie Young
<ellie@usenix.org>

Below is a summary of the actions taken at and just after the regular quarterly meeting of the USENIX Board Of Directors held on January 27, 1996 in San Diego, CA.


Computing Systems

Dave Presotto, the journal's editor, reported that he was having difficulty in getting enough quality submissions to continue the journal in its current focus and frequency. It was decided that the membership and operating systems community should be notified of this and encouraged to submit. A committee was formed and will make a recommendation concerning the journal's future in a year.

International Outreach/Groups

The representative from the AUUG, Chris Maltby, reported on their recent conferences and activities. He and the AUUG conference organizers were encouraged to contact the USENIX staff and program chairs to obtain address information for USENIX speakers who they may wish to invite to present at their conferences. Simon Kenyon reported on behalf of EurOpen that they were offering more network services to their members (such as offering online services/directory for various groups on a Web site). They were also sponsoring small workshops and seminars in various countries.

Legal

The Association's attorney explained that lobbying activities are narrowly defined by the IRS, and that USENIX can engage in these activities if it wishes, so long as they are not a large part of the budget. He recommended the Board exercise their fiduciary responsibility by reviewing and approving all requests for such activity by USENIX and SAGE.

PGP Key Signing Service

Rose was asked to proceed with plans to offer this service at our larger conferences, beginning with the UNIX Security Symposium in July.

Member Services

Geer and Rose will explore the issues around voting electronically. The proposal to expand the benefits offered to our Supporting members was approved. In addition to the benefits that all Institutional members receive, Supporting members would also get one free ad in ;login: (on a space available basis); a one time half-price rental of our mailing list; and the ability to send up to 10 total attendees to the tech sessions of our conferences at the member rate. A committee was formed to look into new editorial directions for ;login:

Services & Outreach to Students

It was agreed to re-allocate some of the funds in the budget for Good Works/Student Outreach as follows: $20,000 to expand the number of university liaisons; $10,000 to increase the amount of the best student paper awards at the LISA and Annual Technical conferences to $1,000 and to also offer a $500 student award at the other USENIX conferences; $90,000 to increase the number of student stipends (that help defray travel and registration expenses for full-time students to attend USENIX conferences); and $40,500 to fund the pre-college programs, CitySpace and the annual USA Computing Olympiad. (See page 21 for more regarding the student outreach program).

Terminal Room at Security Symposium

It was agreed to support Gretchen Phillips’ idea for providing a secure terminal room at this event (and to lend support and funds to make it happen).

Proposal for Equipment for MBONE Service

The proposal from Evi Nemeth that USENIX purchase two workstations to be used to broadcast the MBONE and also be the terminal room server at the IETF meetings and USENIX Technical and LISA conferences was approved.

Report on SAGE

Evans reported on the recent meeting of the newly elected SAGE board, which had decided to emphasize the following goals in 1996: produce at least 2 pamphlets; pick a set of topics for future pamphlets; foster local groups; enhance SAGE's on-line presence; and communicate to the membership on publicly stated goals including finding an alternate model for the working groups.

Standards

It was agreed to accept a proposal from Nick Stoughton to investigate the usefulness of providing reports on areas of standardization besides POSIX, such as the IETF activities.
Funds were allocated for going to one exploratory IETF meeting in March, preparing snitch reports, and producing a follow-up recommendation for the Board’s consideration.

Two New Supporting Members Join USENIX

Sybase, Inc. and Open Market, Inc. are the latest additions to the USENIX Supporting Members list.

Sybase donated the software and services for System 11, its newest release. Sybase is a leading relational database maker, with products and services designed specifically for the client/server computing model and complete interoperability with multi-vendor hardware and software. They are located in Emeryville, California.

Open Market donated its Merchant Solution software to USENIX. Open Market offers a full spectrum of high-performance business software products for the Internet, making secure electronic commerce possible, and addressing the needs of companies who must do business in the information economy. It is headquartered in Cambridge, Massachusetts. Their URL is http://www.openmarket.com.

The Communications Decency Act of 1996 – What it May Mean to You

by Dan Appelman, Heller, White, Ehrman, & McAuliffe
<dan@hewm.com>

On February 8, 1996, President Clinton signed into law the Telecommunications Act of 1996. That Act makes sweeping changes to existing telecommunications law and will dramatically effect the structure of the broadcasting, cable TV, telephone and data communications industries.

Included in the new legislation is the Communications Decency Act of 1996, in a form much as originally proposed by Senator Exon (D-NB). This new law makes it a Federal crime to make certain materials available electronically to children under 18 or to facilitate others who do so. Although this legislation was intended to impose liability primarily on the Online Service Providers, such as CompuServe, America Online and Prodigy, and certain electronic bulletin board providers, it has significant implications for systems administrators and others as well.

In addition to criminalizing the actions of those who are directly responsible for giving children access to obscene or indecent material, the new law imposes the same liability on those who “knowingly permit telecommunications facilities under [their] control” to be used for those same activities. Many, if not most, systems administrators of systems connected to the Internet may be considered as having telecommunications facilities under their control, and are therefore subject to the new law. Giving access to certain Usenet newsgroups or to other postings or Web content which are obscene or indecent, at least if you know or have reason to know that children under 18 may thereby also obtain access, is now a crime punishable by fines and prison terms. What is “obscene” or “indecent” is determined by local community standards, despite the fact that the Internet serves a national and an international constituency.

The constitutionality of the new law is being challenged. We will keep you informed of any important developments. In the meantime, those responsible for offering and administering Internet and other electronic services will have to live with considerable uncertainty. But there are several things you can do.

1. Together with your employer, you should develop and communicate a clear set of user guidelines and policies. Among those policies should be a set of prohibitions on the distribution of obscene or indecent materials. [SAGE will shortly be publishing a pamphlet, “Writing Policies for Computer Sites,” that will be of interest to many readers.]

2. You and your employer should agree upon a written, limited, and unambiguous description of your job responsibilities. The description should make clear the extent of your responsibility to monitor the use of your employer’s system by others.

3. Consider taking steps to help your activities fall within the several defenses to prosecution and liability offered by the new Act. There is no liability, for example, for those who “solely” provide “access or connection to or from a facility, system or network not under their control.” Neither is there liability for those who take “good faith, reasonable, effective, and appropriate actions” to restrict or prevent access by minors.

4. Read forthcoming articles and attend conferences and tutorials to keep abreast of the latest developments. [USENIX/SAGE will offer its next tutorial on this topic at its next LISA conference in Chicago in October, 1996. USENIX also expects to publish a longer article on “secondary” liability of Internet Service Providers, Online Service Providers, and systems administrators in a forthcoming issue of ;login:. [Later this year, SAGE expects to publish another pamphlet in its series addressing the legal liabilities of systems administrators.]

[See page 37 for an interview with Dan Appelman]
SAGE, the System Administrators Guild, is dedicated to the advancement and recognition of system administration as a profession. In three years, SAGE's membership has increased steadily, and there is growing recognition of SAGE as a representative in system administration issues. SAGE brings together system and network administrators for:

- professional and technical development,
- sharing of problems and solutions,
- communicating with users, management, and vendors on system administration topics.

SAGE News Editor
- Tina Darmohray
  <tmd@usenix.org>

SAGE Board of Directors
- Paul Evans, President
  <ple@usenix.org>
- Tim Gassaway, Secretary
  <gassaway@usenix.org>
- Barb Dijker, Treasurer
  <barb@usenix.org>
- Helen Harrison
  <helen@usenix.org>
- Bryan McDonald
  <bigmac@usenix.org>
- Hal Miller
  <halm@usenix.org>
- Kim Trudel
  <kim@usenix.org>

SAGE Working Groups

<table>
<thead>
<tr>
<th>GROUP</th>
<th>CHAIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>sage-certify</td>
<td>Paul Moriarty</td>
</tr>
<tr>
<td>sage-edu</td>
<td>Ron Hall</td>
</tr>
<tr>
<td>sage-ethics</td>
<td>Hal Miller</td>
</tr>
<tr>
<td>sage-jobs</td>
<td>Tina Darmohray</td>
</tr>
<tr>
<td>sage-locals</td>
<td>Rene Gobeyn</td>
</tr>
<tr>
<td>sage-online</td>
<td>Pat Wilson</td>
</tr>
<tr>
<td>sage-policies</td>
<td>Lee Damon</td>
</tr>
</tbody>
</table>

You can contact these groups via email at <their-name@usenix.org> for example, <sage-certify@usenix.org>.

SAGE Discussion Groups

<table>
<thead>
<tr>
<th>group</th>
</tr>
</thead>
<tbody>
<tr>
<td>sage-security</td>
</tr>
<tr>
<td>sage-managers</td>
</tr>
<tr>
<td>sage-outreach</td>
</tr>
<tr>
<td>sage-pt</td>
</tr>
<tr>
<td>sage-solo</td>
</tr>
</tbody>
</table>

SAGE BofS

<table>
<thead>
<tr>
<th>group</th>
</tr>
</thead>
<tbody>
<tr>
<td>sage.board</td>
</tr>
<tr>
<td>sage.bof.women</td>
</tr>
</tbody>
</table>

SAGE Online Services

Email server:
<majordomo@usenix.org>

FTP server:
ftp.sage.usenix.org

WWW URL:
http://www.sage.usenix.org

SAGE Supporting Members

Enterprise Systems Management Corp.
Great Circle Associates
Pencom Systems Inc.

---

Disaster Preparation

by Tina M. Darmohray
<tmd@iwi.iwi.com>

In early December 1995, Oregon and California were hit by an unusually strong storm. Wind gusts broke records and trees, power poles, and power lines. Pacific Gas and Electric appeared unprepared to deal with the resulting power outages. Hundreds of thousands were without power for days. I was one of them; my house went without power for five and a half days. Because my office is in my home, my computer went without power, too. My nerves (I have three young children) went out after about four days. On the fifth day without power, my husband and I went Christmas shopping for one another: we purchased a very romantic generator (the last one they had on the shelf several communities away).

I live in earthquake country, so I had planned for the inevitable disaster. But I quickly realized I hadn’t considered all the angles. Without any video displays working in my house, I had lots of time to think through how better to prepare for “the worst.” Here’s how I decided to analyze my plans in the future:

- Think about what could go wrong (be outrageous). I had thought about power outage, for instance, but not for five days!
- Make a list of what will be essential to “survive” and prioritize it. Then look for interdependencies. (Canned goods won’t feed you in a power outage if you don’t have a manual can opener!)
- Perform a mock disaster drill. What worked? What didn’t?
- Bring in someone from outside your organization to review your plan. Sometimes the familiar can weaken your critical eye.
- Diversify location (and lower your probabilities of negative impact).
- Consider redundant systems. Locate a full backup offsite (my generator is out in a shed, not in our basement). Arrange for emergency services/support with another organization (maybe your service provider could “hold” your email until you get back online?).

Inspired by my recent experience without my normal suite of creature comforts, I’ve asked several fellow system administrators to put into writing their computer-related disaster experiences or how they plan for disaster at their site. I hope that their articles will give you some ideas of how wrong things can go, or how planning for the worst can keep you from actually experiencing it. Make plans for your site now; there’s no time like the present!
When It Rains...

by Rick Kawin
<kawin@stat.berkeley.edu>

Over time I have become accustomed to the sporadic failures that plague computers and their system administrators. It is not uncommon for me to have to respond to electronic component failures, deterioration of the media on a disk, the computer room overheating due to an air conditioner malfunction, or to the various problems associated with brownouts and blackouts. What I did not expect to encounter was a flooded computer room, because our group is located on the fourth floor of a ten story building.

A year ago I was shocked when I discovered that all the cables, punchblocks, transceivers, and electrical outlets that supply our three servers were completely submerged. The raised subfloor in our computer room was covered in six or eight inches of water. The cause of the flooding was a water pipe valve that broke while workmen were making modifications on the floor above our computer room. The same path that brings air into the computer room apparently brought water. In our building the subfloor functions as a plenum for the return air for the air conditioner (yes, the air flow is set up backwards). Water flowed from the fifth floor, down the water pipe and into the air conditioner room, which shares a subfloor with our computer room.

To my amazement, we were able to halt the computers before we turned off the power! After bailing out the water using a pump, buckets, and sponges, we gathered every fan, heater, hair dryer, and heat gun we could find to dry the floor. Luckily, the week before, we had moved a powered multiport Ethernet box from the subfloor to a rack in the computer room. The remaining transceivers on the subfloor were the old black 3Com boxes with the components embedded in epoxy. We were able to dry all these and reinstall them without any failures (sometimes there is an advantage to not having enough money to replace old hardware!)

The crisis brought on by the flooding was not all bad. I finally had an opportunity to rearrange all the cabling under the floor, and for the first time the subfloor was really clean! In the end, we lost a day’s work, but no equipment. The rooms below us were less fortunate because water came down from their ceiling and damaged some workstations.

It’s been a year since the flooding occurred. Since then, I have lined the computer room subceiling with plastic sheeting to deflect any water away from the computer equipment. I also keep a roll of plastic sheeting in each of our computer labs (we have already used one roll!). And, I’m still trying to get a drain placed in the subfloor.

Managing Your UPS in an Emergency
or (Don’t) Wait Until Dark

by Mark K. Mellis
<mkm@mellis.com>

I spent several years serving as an acolyte in the power plant of a nuclear powered submarine, and the experience taught me the real meaning of the phrases “anal retentive,” “worst-case scenario,” and “workaholic.” (You might think, with that in my background, I was predestined to become a UNIX system administrator.) One of the things we were especially paranoid about was electricity, specifically, how to avoid losing it, what to do in its absence, and how to find it again. These subjects turn out to be of intense interest to system administrators, too.

At our site in the Silicon Valley, we once believed that the answer to all electrical problems was embodied in a magic box known as a UPS, short for “uninterruptible power supply” (that we couldn’t convince our management to buy). Eventually, through diligence and refined tantrum-throwing, we finally got one. Imagine our delight when we discovered that the UPS introduced a whole range of new power-management “opportunities.”

The main purpose of our UPS was to condition the incoming power and to let the servers “ride through” transient events. In a power outage lasting more than a few seconds, all our desktop devices would be shut down anyway, so we didn’t see a need for more batteries than were necessary to let us gracefully shut down the servers. We designed ours for 15 minutes, which is the amount of time we were without power after the 1989 Loma Prieta earthquake. (Theory was that a bigger quake would knock the building down, and we wouldn’t really care about the power failure.)

As we grew, we added more buildings, some WAN links, and an Internet connection. Now when one building went down due to a power failure, we still needed to keep critical network equipment online: routers, comm equipment, name servers, and firewalls. Just because the local site was down didn’t mean that our remote offices should be impacted. By judiciously shedding the larger loads early in the power outage, we were able to preserve the equipment needed to keep the rest of the company running much longer.

You need more than just specialized equipment to deal successfully with an emergency. You need to prepare. When the lights are out (you did remember emergency lighting, didn’t you?), it’s difficult to remember all the dependencies. So go
SAGE NEWS

sit down in that console chair now and build yourself a checklist that tells which machines to shut down and in what order. Print it out on paper. Hang it in a conspicuous spot, and review it with your sysadmin team during a staff meeting or periodic training session. It will help you keep your enterprise up even though your site is down.

Remember, an UPS is a complex device, and it needs maintenance too. If it is of the hard-wired type (ours was), hooked directly into the building wiring, you’ll find that you will have to shut down all your loads when you service the UPS. If (heaven forbid) your UPS fails, you’ll be down until an electrician can disconnect it and down again while it is reconnected – unless you have a bypass switch installed. Many UPSs have manual internal bypass switches. We found that an external bypass was the best choice for us, because that allowed us to completely disconnect our UPS with only momentary service interruptions.

Because you are still working on that load shedding checklist, try to recall the last time you performed a cold start of the UPS. Do you remember how to ask it how much time its batteries have left? How does that manual bypass switch work? Where’s the UPS instruction manual, anyway? You may want to add these items to your checklist and incorporate them in that training session so that your whole team knows how to ride out a power failure successfully.

The secret to success in emergency situations is to think the problem through ahead of time, write it down, walk through the motions, and do it often enough so that you’ll remember it when you need it. A little preparation will pay dividends when the lights go out.

The Sound of the Waters Gently Lapping at Your 30-Amp Circuits

by Paul Evans <ple@synopsys.com>

On the afternoon of Friday, November 17, I was getting ready for the Synopsys Network and Computing Services quarterly downtime. During the quarterly downtime (always scheduled over a weekend in the middle two weeks of the middle month of each financial quarter to minimize impact), we are free to carry out maintenance, upgrades, and changes in our network and computing environment with no service guarantees whatsoever to users. Because I expected to be working late into the night for the whole weekend, I decided to go to my bank and deposit a check; otherwise I wasn’t going to get to it until early the next week. Before leaving, I asked J Greely, one of our system administrators, to check under the raised floor in the machine room to see if an electrical circuit he needed to power a new system he was bringing up had in fact been installed. I left the building and went to the bank, and J went to the computer room, where he made an interesting discovery.

I was standing in front of the ATM machine, putting my check in the envelope, when my pager went off: “Water under computer room floor.” I got in my car and headed back to Synopsys as quickly as I could. While I was driving, my pager went off again. This time the message was simpler: “911 computer room emergency.” When I returned, I found several facilities people and system administrators peering under the raised floor at the pond of water, created by a malfunctioning air conditioner, and the tiny waves, formed by the force of the air, gently lapping at my 30-amp circuits.

Our facilities division set to work removing the water from under the floor. I ran out to my office and grabbed both my radio and a list that J Greely had prepared for the downtime that gave the correct order for rebooting the nearly 200 servers in the machine room. I handed the reboot list to the secretary of our group and asked her to make 40 copies and hand them out to as many people in the group as she could find. I then got on the radio and started directing people to shut down machines in the correct order. At this point, Juli Gumbiner, our lead help desk person, joined me in my cube and started crossing systems off the list as we got reports that each one was halted and then powered down. The group was able to shut down all the systems in the computer room in a few minutes.

Why were we able to handle this emergency so promptly? There were at least three areas of preparation that made a difference. First is the fact that we had the partially ordered server reboot list, updated and on my desk in hard copy. Updating this list is a normal part of our downtime preparation, and we just happened to have our computer room disaster a few hours after the most recent version had been completed. I strongly recommend that no matter how big or small your site, you create such a list and revise it on a regular basis. Second, our daily use of radios to do our job made them an effective tool to coordinate the activities of a large number of people (around 40) in a very short time. If you have more than one system administrator at your site, you should seriously consider getting radios for them. Finally, having a good working relationship with your facilities group is also important. In any serious disaster situation, you will be dependent on them for many critical services. Don’t wait until the waters are gently lapping under your raised floor to get ready for a computer room disaster.
Reflection
by Hal Miller
<halm@cse.ogi.edu>

[SAGE Editor's Note: Hal penciled this piece during the same storm-induced power outage in December '95.]

Here I sit, in an extended power outage, suddenly with "nothing to do." Much of the city is down, with the area covering both my office and house likely to be down a day or two. I've turned off all power switches, sent people home, cut off the breakers, and gone home myself.

Started thinking "Ah, a chance to finish testing some software I've been playing with." Except that SPARCStations don't run on AA batteries. OK, catch up on news and email. Nope. Well, perhaps I'll write that article I was planning for ;login: - hmmm, where's that "manual graphite display generator"?

I can't even check my telephone list (well, actually I have a printout only a couple weeks old).

Has this happened to you? What does this tell us?

The kids are complaining: the TV is out, as is their Mac. What's the matter with kids nowadays? Can't they go work on their Scout projects or something? Well, maybe I can get some other housework done. Nope, washing machine and dishwasher aren't running.

My stomach hurts. Nothing new, I've had stress problems for years.

Wait a minute! What are we doing with our lives? Are our priorities so badly screwed up that we can't find the time to do things around the house, or with the family? Are we unable to do anything "off-line"? Are we so selfish and self-centered that we can't think outside of the boxes we've created for our computing lives? Are we so shallow that we can't live without our digital "fix"?

I've just changed jobs. More stress, of course, but part of the reason was to give me more time and/or energy to see a little more of life while I'm passing through. Just had another friend incapacitated by heart troubles. If that happens to me, I intend to be able to say that I've done well along the way. What's more, I intend that other people will be able to say the same.

Sit back a moment. Is your life contained in that box in front of you? Is that really leading to your doing your best on the job? (Assuming that it means you're not doing your best off it.) Maybe it's time to change that. Live a bit.

Elementary Intrusion Detection, Part 1
by Karen Casella
<kcasella@eng.sun.com>

What is an intruder? The typical dictionary definition of the word "intrude" limits the meaning to "come in inappropriately or rudely."1 In the context of UNIX system security, there are two types of intruders. The first, which is covered by the dictionary definition, is the external penetrator: someone who attempts to gain access to a system without permission. The second is the legitimate system users who attempt to gain privilege or access to restricted resources for which they have not been expressly authorized. A system that is secured by preventing access from outside may not be safe from inside attacks accomplished by abusive use by authorized users.

In this first of two articles on the subject of elementary intrusion detection, I'll address the first of these intruders, the external penetrator attempting to gain access to a system without permission. In the second article, which will appear in the next issue of ;login:, my focus will be on detecting the intrusion of a system insider. Both articles describe my personal experiences implementing elementary intrusion detection. The methods employed can be used by sysadmins with varying levels of experience using freely available tools and operating system utilities. There are much more sophisticated intrusion detection systems, based on statistical anomaly detection, rule-based anomaly detection, the state transition approach, and the like; but they are really overkill for the situation I describe here and for many situations that system administrators are likely to encounter.

The problem that I'll use to illustrate elementary intrusion detection involved monitoring access to a server that contained highly sensitive engineering data. The engineering manager who owned the server was very concerned about who might have access to the data. When I met with the engineering group to discuss their requirements, I was able to convince them that intrusion prevention can be quite effective and can reduce the amount of intrusion detection that might be needed. The requirements upon which we agreed were:

* All attempts (successful or otherwise) at access were to be logged with as much information as possible regarding the source of the attempt.
* In the case of multiple failed connection attempts, the sysadmin was to be notified by pager.
* FTP access was to be limited to a certain group of systems in engineering (as defined by the netgroup "ftpeng"). All other attempts were to be reported to the sysadmin by email.
• Telnet access was allowed only from the engineering domain. If any attempts were made from outside of engineering, the sysadmin was to be notified by email.

• There was some suspicion that a certain engineering system was being used as a springboard for unauthorized users. There were, however, some legitimate engineering users on the system. Therefore, all telnet connections from the system ("suspect") were allowed, but the sysadmin was to be paged (at which point he could investigate).

• No login nor rsh access was to be allowed. The person attempting access would be presented with a message that stated the policy and the sysadmin was to be notified of the attempt by email.

• Several services were expressly not allowed: uucp, talk, tftp, wall, rexd. (These were simply commented out of the inetd configuration file and the inetd daemon restarted.)

• The finger command was allowed only from the system administration trusted host ("olympus").

• Only the sysadmin was allowed root access on the system (there had been between four and six people who had the root password). If it was detected that anyone else was able to gain root access, the sysadmin was to be paged.

• If system activity exceeded certain thresholds during certain times of the day, the sysadmin was to be paged.

• If system resource utilization changed drastically during certain times of the day, the sysadmin was to be paged.

• There was not enough disk space available to use C2 auditing.

• There was no money available for this project (surprise!).

Because of the requirements, I decided to use a combination of publicly available tools to complete the project. For most of the access control and connection detection, the tcp_wrappers package was the most logical choice. For anyone not familiar with tcp_wrappers, it is a package that allows you to monitor and filter incoming requests for many network services. The inetd configuration is changed so that all requests for a network server program are handled by a wrapper program that logs information about the requesting client, checks for access control specifications, and then executes the regular server program. There is a much more detailed description of the tcp_wrappers package and how it can be used in the author's paper and included with the package documentation (get it from your favorite anonymous FTP site).

The tcp_wrappers daemon (tcpd) uses two files to define the access control policy. The simple access control language is based on client (hostname/address, user name) and server (process name, hostname/address) patterns. Access is granted when a client pair (daemon, client) matches an entry in the /etc/hosts.allow file. Access is denied when a client pair matches an entry in the /etc/hosts.deny file. If no match is made in either file, access is granted. The access control files can also be used to specify actions that should be taken when a match is made and where logging information is to be written (by default, actions are logged to syslog).

The general format of an access control file entry is:

daoen_list : client_list [ : 
    shell_command ]

If host access control language extensions are enabled at compile time, there are many options that can be added using the access control language:

daoen_list : client_list : option : option

In either form, the daemon_list is a list of one or more daemon process names or wildcards. The client_list is a list of one or more hostnames, host addresses, patterns, or wildcards.

In the example I am describing, the second form of the control language is used. The following is similar to the /etc/hosts.allow and /etc/hosts.deny files used for this configuration (names have been changed to protect the guilty). (See example on facing page).

So, what does it all mean? One recurring field in both files is used for extended logging:

spawn (echo "'date' %d connect from %c" >> 
    /var/log/tcpd.log) &

By default, tcp_wrappers performs standard syslog logging, but with minimal information regarding the connection. By including this line for each service, the tcp_wrapper daemon writes more detailed information in a separate log file that can be browsed or parsed later. The echo statement includes information about the client system attempting access to the server. The "%d" is replaced by the daemon name to which a connection is being made, for example, in.telnetd. The "%c" is replaced by client information, as much as is available, for example, user@host, user@address, hostname, or just an address.

For FTP access, the /etc/hosts.allow entry specifies that anyone in the netgroup “ftpeng” may have FTP access to the system. The /etc/hosts.deny entry disallows all other access. Both allowed and denied access are logged.
#/etc/hosts.allow

###ftp:

ftpd: @ftpeng:\n  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

### telnet:

telnetd: suspect:\n  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &:
  spawn (/usr/ucb/mail -s "ALERT %d connect from %c" sapage) &:
  spawn (/usr/local/bin/safe_finger -l @%h | /usr/ucb/mail -s %d-%h sysadmin)

telnetd: LOCAL, eng.abc.com:\n  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

### finger:

in.fingerd: olympus:\n  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log)

#/etc/hosts.deny

###ftp:

ftpd: ALL:\n  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

### telnet:

telnetd: ALL:\n  spawn (/usr/local/bin/safe_finger -l @%h | /usr/ucb/mail -s %d-%h sysadmin) &:
  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

### rlogin and rsh:

rlogind: ALL: banners /etc/banners:\n  spawn (/usr/ucb/mail -s "tcpwrapper alert %d from %c" sysadmin) &:
  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

rshd: ALL: banners /etc/banners:spawn (/usr/ucb/mail -s "tcpwrapper alert %d from %c" sysadmin) &:
  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &

### everything else:

ALL: spawn (/usr/ucb/mail -s "tcpwrapper alert %d from %c" sysadmin) &:
  spawn (echo "'date' %d connect from %c" >> /var/log/tcpd.log) &
SAGE NEWS

The first telnet entry in /etc/hosts.allow is used to do extended logging and alerting if access is attempted from the system “suspect.” A short message is sent to the sysadmin’s pager, and a message including more detailed information is sent to the sysadmin by email. The safe_finger command is distributed with tcp_wrappers and is used as a wrapper around the regular finger command that filters out data sent by the remote host. The second telnet entry in /etc/hosts.allow is used to indicate that anyone who is LOCAL to the server (no “.” in the hostname) or anyone in the “eng.abc.com” domain is allowed telnet access. All other telnet access is denied as specified in the /etc/hosts.deny file, and the sysadmin is notified by email with extended connection information.

The specification line for fingerd is simple — finger connections are allowed only from the system “olympus” (the system administration trusted host).

Neither rlogin nor rsh is allowed from any host as specified in the /etc/hosts.deny file. One final tcp_wrappers access control language option is introduced for these services, and that is the use of “banners.” If the “banners” option is used, the tcp_wrapper daemon looks in the specified directory, in this case /etc/banners, for a file with the same name as the daemon and echoes its contents to the client. In this case, there are two files in /etc/banners named “in.rlogin” and “in.rsh.” Whenever a user attempts an rlogin or an rsh, the contents of the file are displayed (issuing a warning that access is not allowed and that the attempt has been logged), and the connection is not completed. As before, mail is sent to the sysadmin and extended logging is done.

Finally, there is a catchall statement at the end of /etc/hosts.deny that specifies that all other connections are to be denied, logged, and a message sent to the sysadmin by email.

The alert reader may note that not all specified requirements have been met yet. In the next issue of ;login:, I’ll describe a couple of other public domain tools that were used to complete the project.

References:


Disclaimer – The information in this article is the opinion of the writer only and does not reflect any particular system monitoring or access control configuration. Sun Microsystems does not endorse this or any particular type of system monitoring or access control configuration.

The Future of IP Security

by Shawn Instenes
<shawni@celene.rain.com>

Recently I’ve wondered where security protocols for traffic on the Internet are heading. I’ve sometimes felt that “if only IP were more secure, then maybe firewalls wouldn’t be necessary,” even as I keep building them.

I don’t know how often I hear requests for support of applications that firewall technology can’t handle... yet. As an information management tool, firewalls are a bit like throwing the baby out with the bath water — you have access to this huge network of information, yet to stay secure you block off access to everything but a limited set of protocols. You assume that what you don’t know can hurt you — with good reason.

But if you knew who was communicating with your computers, and could be certain they all were authorized to do so, that’s at least as good as what your firewall does now. You could chuck the thing out the door.

But to really fix the security problem, IP itself would have to provide the means. It would be incredibly tedious (not to mention error prone) to recode application after application to provide for secure and authenticated data transmission. That’s assuming you had source.

There have been some efforts to improve IP in this area. Matt Blaze and John Ioannidis published a USENIX paper about one implementation (see the list of URLs at the end of this article) called swIPe, which I’ve used in the past. Some commercial firewall products use the swIPe protocol to provide firewall-to-firewall encryption, but it doesn’t see much use outside of that.

More recently, thanks to the work of the IETF IPSEC working group, there are Standard-track RFCs for IP security, both for IPv4 and IPv6 (IPng). IPSEC has broken down the security protocol issue into three parts: (1) authentication, provided by the Authentication Header (AH); (2) confidentiality, provided by the Encapsulating Security Payload (ESP); and (3) key management, which currently has no Standard-track RFCs, only drafts.

Why three parts? The Authentication Header is separated because the “lack of confidentiality ensures that implementations of the Authentication Header will be widely available on the Internet, even in locations where the export, import, or use of encryption to provide confidentiality is regulated” (RFC 1825). Implementation of only the Authentication Header (RFC 1826) provides protection against IP source spoofing and connection hijacking, which is much better than the situation we have now. I don’t like certain unnamed governments’ restrictions on cryptography any
more than you do, but we're stuck with it for now, and splitting the standard like this makes sense.

Key management has been separated from the rest because this "separation of mechanism is clearly wise given the long history of subtle flaws in published key management protocols" (RFC 1825). The AH and ESP standards won't be affected by changes in the Key management protocols, if any such flaws are discovered after deploying one. There are three contenders for the Key management protocol: Photuris, ISAKMP, and SKIP. All of these are still works in progress, not standards.

Photuris key exchanges start first with a Diffie-Hellman key exchange, so further Key management transactions can be private. Next is the authentication phase, which is encrypted; thus, an eavesdropper won't know the identity of either party, assuming the IP numbers being used are dynamically allocated or otherwise not associated with either party. Data encrypting keys are generated at intervals for as long as the session lasts. Photuris supports a wide variety of encryption and authentication algorithms.

Simple Key management for Internet Protocols (SKIP) has somewhat different design goals. As much as possible, SKIP attempts to be stateless, requires zero Key management traffic, and has lower overhead than Photuris. The catch, however, is that SKIP requires that cryptographically signed certificates be available for every node that wishes to communicate securely. Although the Key management traffic is zero, there is a bit of certificate management traffic that must take place before any SKIP-managed communications take place. If the certificates have not been loaded manually beforehand, the first time any two nodes communicate, certificates must be exchanged and verified.

One interesting feature I found in the SKIP drafts is an optional differentiation between nodes and communication endpoints in the Algorithm Discovery Protocol. This means different endpoints (ports) could use different encryption mechanisms or authentication. In particular, I feel that email traffic on the SMTP ports needn't be encrypted; there are methods of verifying integrity and providing confidentiality that are better suited to email already.

There's code out there you can look at, if you have an idle moment or just want to get a feel for IPSEC early, because eventually, if you're like me, you're going to have to manage this stuff. The Naval Research Laboratory has a preliminary IPv6 implementation with RFC 1825-1829 features. There are at least three sources of SKIP software.

We're not quite yet to the point where I can cart off the bastion host, but we're getting there. I have a wheelbarrow waiting.

URLs

2. <ftp://ds.internic.net/rfc>, RFCs 1825-1829.

Perl Practicum: Thanks for the Reference
by Hal Pomeranz
<hal@netmarket.com>

In the last Perl Practicum, I covered the new Perl5 construct, references. Because the syntax and usage for references are confusing in places, I promised an extended example that would cover, in a real-world example, all the material I had introduced.

The problem is to "marshall" data into format such that if we do

```perl
$string = marshall($some_ref);
eval("$other_ref = $string");
```

then the data structure pointed to by $other_ref will have the same contents as the data structure pointed to by $some_ref. If this is true, then we can save $string to a file and read it back in later in some other application. We use this kind of functionality here at NetMarket to allow Web CGI applications to share data because the format is extremely portable.

Thinking Recursively

The key to unlocking this problem is really a mode of thought, rather than a programming trick. Sometimes assuming you already know how to solve the problem enables you to develop a solution – this is the heart of "recursive thinking."
Remember from last time that the way to create a reference to an anonymous list was

$$a_{\text{ref}} = [1, 2, 3];$$

where the elements of the list could be lists (or associative arrays, etc.)

$$other_{\text{ref}} = [1, \{\text{a"}, \text{b"}\}, 3];$$

In general, then, an anonymous list reference is assigned with

$$some_{\text{ref}} = \{\text{EXPR1, . . . , EXPRn}\};$$

where EXPR1, . . . , EXPRn are either a scalar or a reference to some other complex data structure (list, associative array, list of lists, etc.).

This is where recursive thinking comes in. We assume that we already have a marshall() function which can encode EXPR1, . . . , EXPRn properly. With this simplifying assumption, we can quickly write a function that creates a string containing the righthand side of the expression above. All we have to do is put commas after successive calls to the marshall() function and put square brackets around the whole affair:

```perl
sub encode_list {
    my($list_ref) = @_; # scalar reference
    my($string);
    $string = "[ " ;
    for (@$list_ref) {
        $string .= marshall($_) ;
        $string . = ", ";
    }
    $string . = "] " ;
    return($string);
}
```

Notice the @$list in the for loop: remember that you can use a scalar reference any place you would use the name of an identifier. Yes, there is a dangling comma after the last element of the list – Perl ignores it. If you are thinking this is all handwaving and I have not even begun solving the problem, bear with me.

### Dealing with Associative Arrays

Remember that the syntax for declaring associative array references was

```perl
$mail_info = {
    "hal" => "hal@netmarket.com",
    "tina" => "tmd@iwi.com",
    "rob" => "kolstad@bsdi.com",
};
```

Here the values in the hash can actually be references to complex data structures, but the keys have to be scalars. To state things generally, we assign hash references with expressions like:

```perl
some_ref = {
    KEY1 => EXPR1, 
    ..., 
    KEYn => EXPRn
};
```

We are already assuming we have marshall() lying around to encode EXPR1, ... , EXPRn. I promise that I will write an encode_scalar() function next to handle encoding KEY1, ..., KEYn. With those two assumptions, encoding an associative array reference is now just a matter of putting commas in the right place:

```perl
sub encode_hash {
    my($hash_ref) = @_; # scalar reference
    my($string, $key);
    $string = 11  { ";
    foreach $key (keys( %$hash_ref )) {
        $string . = encode_scalar($key);
        $string .= "=> ";
        $string . = marshall
        ($$hash_ref{$key});
        $string .= ", ";
    }
    $string . = " ] " ;
    return($string);
}
```

### Encoding Scalars

You may think that encoding scalar values is trivial. After all, one of our simple examples above just used:

```perl
$mail_info = {
    "hal" => "hal@netmarket.com",
    "tina" => "tmd@iwi.com",
    "rob" => "kolstad@bsdi.com",
};
```

You would guess from this example that you just throw quote marks around the value and be done. Well, suppose your scalar value is one of these strings:

```
got"ya
$variable
```

You will end up with encodings like:

```
"got"ya  # dangling quote
"$variable"  # evaluates $variable
```
The safest thing to do is to backslash every nonalphanumeric character in the scalar:

```perl
sub encode_scalar {
  my($scalar) = @_;
  $scalar =~ s/\W/\$l/g;
  return("$scalar")
}
```

Now our strings will get encoded as:

"got\"ya"
"$variable"

Putting It All Together

All along I have been defining the encoding routines in terms of this mythical `marshall()` routine. It turns out that we can define `marshall()` in terms of the encoding routines:

```perl
sub marshall {
  my($thing) = @_;

  $type = ref($thing);
  if ($type eq "ARRAY") {
    return(encode_list($thing));
  } elsif ($type eq "HASH") {
    return(encode_hash($thing));
  } elsif (!$type) {
    return(encode_scalar($thing));
  } else { die("Can't handle $type\n"); }}
```

Remember that `ref()` is a function that returns what type of data type its argument points to, returning `undef` if its argument is not a reference.

How can this possibly work? Things will become clearer when we walk through a simple example. Suppose we do:

```perl
$simple = [1, ["a", "b"], 3];
$output = marshall($simple);
```

In the first call to `marshall()`, `$simple` is identified as a reference to a list and `marshall()` calls `encode_list()`.

The `encode_list()` function starts building up $string. First the function sets $string to be the initial opening bracket. Then `encode_list()` begins walking through the list and calling `marshall()` on each element.

The first argument of the list is a scalar, 1. When `encode_list()` calls `marshall()` on this element, `marshall()` immediately calls `encode_scalar()`, and `encode_scalar()` returns the string:

"1"

`marshall()` simply returns this value back up to the `encode_list()` function, which appends it to the value of $string. At the end of one iteration of the loop, $string looks like this:

```
[ "1",
```

Now `encode_list()` calls `marshall()` on the second argument of the list. This argument is a list reference, so `marshall()` calls `encode_list()` recursively.

This second call to `encode_list()` starts building a new $string. First it initializes this new $string with the opening square bracket. Next it calls `marshall()` on each of the list elements in turn, both of which are scalars. After the first iteration of the loop, the new $string looks like this:

```
[ "a",
```

After the second iteration, we have:

```
[ "a", "b",
```

The loop terminates and the `encode_list()` function appends a closing bracket and returns

```
[ "a", "b", ]
```

to the `marshall()` function that called it originally. In turn, `marshall()` returns this value to the original `encode_list()` call. This function appends the string above to its own $string, which now looks like this:

```
[ "1", [ "a", "b", ],
```

This `encode_list()` function now moves onto the third and final element of the list in this example. This is just another scalar, so after the third iteration we have this:

```
[ "1", [ "a", "b", ], "3",
```

We have exhausted the elements of the example list, so we fall out of the `for` loop. `encode_list()` appends the closing bracket and returns the string:

```
[ "1", [ "a", "b", ], "3", ]
```

Try working out a simple example for yourself, possibly with an associative array this time. Or just type in the code and try running some examples.
Brevity is the Soul of Wit

We can use function references to simplify the `marshall()` function. First, we build a “jump table”: an associative array that links certain keywords to various function references. In this case, we link the strings returned by `ref()` to the function used to encode each data type:

\[
%\text{encode} = ("SCALAR" => \&\text{encode}\_\text{scalar}, \\
"ARRAY" => \&\text{encode}\_\text{list}, \\
"HASH" => \&\text{encode}\_\text{hash});
\]

`marshall()` now becomes extremely terse:

```perl
sub marshall {
    my($thing) = @_; 
    my($type, $func_ref); 
    $type = ref($thing) || "SCALAR"; 
    $func_ref = $encode{$type}; 
    return(&$func_ref($thing)) if ($func_ref); 
    die("Can’t handle $type\n"); 
}
```

First we assign `$type` to be whatever gets returned by `ref()` or `SCALAR` if `ref()` returns `undef`. Next we extract the appropriate function reference from `%encode`, and call that function on the argument to `marshall()`. If we cannot find an appropriate function to call, we `die()`.

To make this even more terse, remember that you can use a block inside of curly braces in place of a scalar reference. In other words, we can use

\[
(%\text{encode}[$\text{type}])
\]

in place of `$func_ref`. Our new version of the function is:

```perl
sub marshall {
    my($thing) = @_; 
    my($type); 
    $type = ref($thing) || "SCALAR"; 
    $func_ref = %encode{$type}; 
    return(%{$func_ref}{$thing}) if ($func_ref); 
    die("Can’t handle $type\n"); 
}
```

I generally hate using extra variables, but the function above is hard for the human eye to comprehend.

Thanks for the Reference

Please type in this code and try it out on some complex examples. If nothing else, the exercise will get you using references so that you will remember how they work. Also, never forget this little demonstration of the power of recursive thinking—a useful tool for any programmer.

---

Commercial Product Design in Tcl

by John Schimmel

<jes@sgi.com>

As a systems administrator, I always thought of Tcl as a cute little scripting language to produce GUI wrappers for all my administration scripts. Now, as a programmer producing products for a systems vendor, I’ve concluded that Tcl is an ideal language for producing commercial tools.

A few years back, designing graphical interfaces was very difficult. If you wanted to produce a product that ran across a wide range of hardware and software platforms, you had to rewrite your tools for each one separately. Luckily for application developers, the windowing war has reduced the field to only a small number of choices, and the field narrows with each passing day. The three that still exist in large numbers are Motif on X Windows, Microsoft Windows, and the Apple Macintosh.

For X windows, there are many GUI builders and tool kits to make programming easier. But a small application still averages around 5,000 lines of complex code; and if the chosen fonts do not exist on the installed machine or if a vendor chooses to extend the Motif look and feel, the application can look completely different on each machine where it runs.

The Visual Languages suite from Microsoft and others on the PC have revolutionized the way people write software under Windows. But they only run under Windows, and still produce pretty complex code. The GUI builders for Visual Basic and friends are very cool and allow a programmer to pop out a program very quickly. But they are also very limiting, and the code produced usually needs to be tweaked by hand. There are now a number of tool kits for Windows that will compile some simple X programs, and some even change the widgets to look more like their Windows cousins. However, the code still has to be rewritten to look right running on Windows because of the way X applications are written.

The Macintosh is relatively easy to program because all the interfaces are fixed. But the development environment is very weak, and tools developed for the Mac never work on anything else. There are several X tool kits or Windows tool kits for the Macintosh to improve porting of other software, although none of these will produce the Macintosh look and feel.

Tcl provides the promise of being able to write a single application and have it run unchanged on any of the three prevailing platforms. At this time, Tcl and Tk have been released on both the Macintosh and Windows operating systems in beta form with the X look and feel. However, there is
a lot of effort going on to get Tk to take on the look and feel of the platform on which it is running.

Several of the recent steps in platform independence inside of Tcl include the rewriting of the I/O system, the addition of platform independent socket calls, and the addition of a network filename space. Still needed are changes to Tk to support a system-independent way to name fonts, colors, and events and a set of common widgets (e.g., file selection boxes, menu bars and dialog boxes) that conform to the underlying platform.

Writing a commercial application requires features that most system administration hacker types rarely worry about, like structured namespaces, internationalization, and code hiding. Each of these is also currently being addressed by the Tcl community.

When multiple programmers are working on different aspects of an application, they need to be protected from each other in some way. Name collisions in packages and module code have been dealt with in many ways in the past. Languages like C contain the concepts of static variables that are addressable only within the space of the current file, but little other protection is available. In C, most libraries simply prepend the name of the library or the like to the beginning of each routine or variable. This gives names like sgiTk-Whatever. C++ allows name hiding in a class hierarchy, so the programmer sees nice things like “class::function” or just “function” when inside of a member routine. But the compiler flattens this name space to produce monstrous names like NewPanel_FPvP10Tcl_InterpiPPc. Interpreted languages like Tcl can simply produce multiple symbol tables and mark them as visual only to routines within a given package.

Tcl is currently 8-bit clean so it can display ISO character codes. Work for the support of 16-bit unicode is progressing so that it can support Asian languages. Much work needs to be done in this area to support the different methods for internationalization supported by the different windowing systems. These are being addressed and are likely to exist within the next year.

Because Tcl is an interpreted language, most of the distributed code is in the form of scripts. This is not usually a problem for system administration tools or free-ware on the Net, but most commercial houses like to protect their code by compiling it into something that is harder to change. Thus programs like tcl2c, which embeds a Tcl script into a C program as a big string, were written. The concept that it is harder to support code when people can change it has been proven wrong, over and over again, by the free-ware people. For the most part, when you receive complaints about how the code works, it comes with a patch to fix it. Protecting intellectual property by compiling is also a misnomer. Because very good disassemblers exist for all the primary platforms, compiling to hide implementation is only really useful in obscuring things. A Tcl compiler is planned for the future, but the hope is more to improve performance than to hide implementation. Most likely the Tcl compiler would be a compile on the fly optimizer similar to what is used in Perl, instead of a native language compiler.

Writing large packages in Tcl is often thought to be more difficult for applications developers. However, the advent of Tcl libraries and dynamically loaded packages has actually made it much easier than writing a similar application in C or C++. Each product developer can easily work on a separate package done in Tcl and possibly shared libraries of other code, all of which can be tied together at the top using a small wrapper script that imports the needed modules.

Many very complex commercial packages developed in Tcl are now available. A list of many of them is posted on the newsgroup each month. At the Tcl conference, there were attendees with code bases as large as half a million lines of Tcl, and each was pleased with the result.

Tcl contains aspects that make it an excellent choice for use in developing commercial applications. It is available on an enormous variety of hardware and operating systems. It will soon provide native look and feel on each of the primary windowing systems. It provides elementary support for internationalization with greater support on the way. And it provides modularized programming components critical to the development of large applications. With the work currently under way, applications developed now will be marketable on a wider range of systems than using currently available methods.

A Dozen Reasons to Write a Paper for LISA '96

by Helen E. Harrison
<helh@unx.sas.com>

As the deadline for submissions for the upcoming System Administration Conference approaches, I thought, as program co-chair, that I would pass along a few exciting and highly motivational reasons that you might want to write a paper:

1. It's easier to convince your management to send you to the conference if you have written a paper.

2. Conference papers look really good on your resume.
3. Instant fame: this could be your 15 minutes. You wouldn’t want to miss it!

4. This is a good excuse to visit Chicago and its many museums, shops, restaurants and night clubs.

5. This is an opportunity to return the favor for all the good ideas and solutions you have found through past LISA conferences.

6. This is a chance to improve your writing skills.

7. You get to write your own biography.

8. This is an opportunity to show your boss just how good your work really is.

9. You get to wear a ribbon on your conference badge.

10. It’s easier than doing systems administration.

11. Systems administrators do have something to write about.

12. Technical people really can write papers!

Don’t forget: extended abstracts are due May 7.

[See pages 65-66 for the LISA Call for Papers.]

SysAdmin At The Movies Strikes Again . . .

by Nick Cuccia
<cuccia@Talamasca.COM>

. . . and her name is Babe.

It’s a Friday night, so our intrepid sysadmin movie critic decided to see what this porker was about. Well, what can I tell you, this pig is no ham. This cut is 100% pure loin.

Okay, enough of the pig crack(lin)s.

Our story starts, much the same way as our critic’s story does, at a major institution. Finding herself out in the cold, cruel world far too soon, Babe finds herself up for a couple of interesting positions, only to wind up in the same constraining situation she started her new life in. Finally – purely by luck, really – Babe winds up in an interesting shop.

Of course, our hero isn’t sure what her role in the organization is supposed to be, and ultimately finds herself in – you guessed it – the IS organization. While one of the group’s two managers has his doubts, the other, perhaps seeing something special, welcomes Babe to the best of her abilities. Not long after that, the rest of the junior staff pretty much treat Babe like one of the gang.

Babe starts out okay, but finds herself being sweet-talked by this big-nosed dude from Marketing – a real go-getter who insists that everybody jump when he says so. Having done this, Babe ultimately gets caught blue-handed, and gets sent to operations.

Alas, as we’ve all seen during our careers, the COO sees an opportunity for downsizing, and the junior staff winds up finding new opportunities elsewhere. Babe is kept around, however, and manages to get back in the good graces of at least one of the IS staff.

While the COO had different plans for our hero, he notices that Babe has certain skills that he could use, particularly after a hacking incident in which the company lost much of its intellectual property; Babe was able to stop the attackers from causing more damage. Soon, it’s discovered that Babe has a special way of getting data where it needs to be. Her nonconfrontational manner for dealing with customers has also drawn the attention – and jealousy – of the senior IS manager; while he lashes out at a customer, Babe initially gets the blame, and is almost terminated. The CEO ultimately figures out what really happened, however, and with some tough love and a little therapy, brings the senior IS manager back in line.

Most touching for me was the last scene, where the COO (without bothering to inform the CEO) helps Babe prepare a demo for a trade show. Babe faced multiple obstacles along her way – first the CEO’s executive assistant does her damndest to slash Babe’s self-esteem; the show’s board questions the right for Babe to give the demo, and Babe winds up not being able to figure out how to manage the particular protocol she’s facing. But, upon learning the secret, Babe ultimately gets the data to their destination without fragmentation or packet loss. In the end, Babe earns the enduring praise of the thronged media, of her management and COO, and of the CEO (who saw the whole scene in satellite linkup).

Kids, I never thought that I’d say this, but this G-Rated wonder was, bar none, the best movie of 1995. I give it Four Root Prompts, plus a bonus copy of The Bat Book for including “Internet Bandits” in the closing credits.

Until next time, the link is down.
Interview with Dan Appelman
by Rob Kolstad
<kolstad@BSD.COM>

[Editor's Note: ;login: conducts occasional interviews with industry personalities. This interview was conducted electronically on February 5 with Dan Appelman, a partner at the law firm of Heller, Ehrman, White, and McAuliffe. He agreed to answer some questions about the Telecommunications Reform Act and its concomitant Communications Decency Act. See page 23 for more information on this Act.]

Rob: President Clinton signed the Telecommunications Reform Act into law recently. Many cybemauts are protesting its censorship provisions. What's going on?

Dan: What's going on is a reaction in this country against the free-wheeling exchange of ideas that the Internet makes possible. It has officially started in a big way with enactment of the Communications Decency Act of 1996 (CDA), which is part of the Telecommunications Reform Act you referred to. The CDA imposes fines and jail terms on those who make "obscene" or "indecent" materials available to others over the Internet. There's even a special provision which makes it a crime to provide these materials to minors. There are so many problems with this new legislation. But I can give you a taste of what I think are its biggest problems.

The biggest problem for readers of ;login: is that the new act not only penalizes those who are responsible for digitizing and posting obscene and indecent material; it also imposes liability on anyone who "knowingly permits a telecommunications facility under his control to be used for [such purposes]." Now the trouble with this is that if we are honest, all Internet service providers and many systems administrators know that facilities under their "control" are sometimes used for purposes of distributing obscene or indecent material. Operating a USENET server would do it. Permitting your users to have access to a USENET server would also do it, so long as the telecommunications facility you are responsible for is connected to that server and is under your control.

The law doesn't say that you don't have to worry if you really can't do much about it. So the biggest problem I see is that this new law would make criminals out of everyone in the USENET's chain of distribution, so long as USENET includes newsgroups that may themselves contain obscene or indecent materials. Of course, prosecutors also have to prove that the person in control intended that it be used for such activity. But if you intend to feed all newsgroups to your users knowing that some of them are likely to contain obscene or indecent material, that may be all the intent that's required.

The second problem is that the new law makes no distinction between whether these obscene and indecent materials are distributed by posting on the USENET, by being accessed in home pages on the World Wide Web, by being made available by anonymous login via FTP, or by email. So even if you deleted access to all USENET alt.sex newsgroups, that wouldn't take care of your problem.

The third problem I see is with the definition of "obscene" and "indecent" material. Actually, the exact phrase used in the legislation is "obscene, lewd,
lascivious, filthy, or indecent.” Now the definition of “obscene” was settled by the Supreme Court in the 1973 case of Miller v. California. Before Miller, only materials that were “utterly without redeeming social value” were obscene under the law. But Miller established a new test. Under Miller, material is obscene when, taken as a whole, it does not have serious literary, artistic, political, or scientific value. You can see that this is a much broader definition of obscenity.

What’s worse is that there is no national standard. The work must be tested by the “average person” applying “contemporary community standards.” This means local standards. To say the least, this does not work well with materials distributed on the Internet. Thus, the Thomas case, where a California couple was convicted and put in jail in western Tennessee for accepting a subscription to its adult-only electronic bulletin board when a postal inspector there felt that the materials he accessed under cover would be obscene when tested by the average person in his community.

But the real innovation in the CDA is that one can violate its provisions by distributing indecent material as well.

Rob: What do they mean by “indecent”?

Dan: Indecent material doesn’t have to be obscene, just “patently offensive.” The Supreme Court established a special restriction on the dissemination of indecent materials in the FCC v. Pacifica case in 1978, but until now it has been applied only to radio and television broadcasting. The essence of the word “indecent” is that the materials it refers to are patently offensive descriptions of sexual and excretionary activities. Again, the assumption is that this offensiveness is determined by local community standards.

Well, Congress has now latched onto this indecency standard and has applied it to the transmission of materials in cyberspace. If you want to see what indecent is, I highly recommend reading the Pacifica case, and especially the appendix to the Court’s opinion, which contains George Carlin’s famous “Seven Dirty Words” monolog. It’s at 438 U.S. 661 for those who know how to navigate there. Who knows, it’s probably also on the Internet somewhere. At least temporarily.

In any event, the significance of the CDA is that it adopts this lower standard from the radio and television broadcasting industry and imposes it upon those who use, and help to operate, parts of the Internet.

Rob: But isn’t the real problem the one of responsibility for material that appears on, say, an Internet service provider’s computer? How come they will be liable when the telephone company isn’t liable for telephone conversations that plan conspiracies against our president?

Dan: Well, that’s an interesting question. The telephone company is considered a common carrier and is only subject to common carrier-type liability. That’s a very high threshold. In practice, it means that telephone companies have no liability whatsoever for the messages they carry unless they know of particular illegal content and have a practical, reasonable way of doing something about it. This comes from a public policy that we benefit most by freeing the telephone company from the responsibility of monitoring and censoring the messages it carries, even though there may be some people who are hurt as a result.

But the CDA makes clear that the common carrier standard of liability is not available for Internet service providers. Quite the contrary, they will be held liable if they “knowingly” permit any telecommunications facility under their control to be used for a prohibited activity with the intent that it be used for such activity. The courts will have to work out what “knowingly” means. This is a much lower standard than the common carrier standard. And by the way, although your question asks about the liability of Internet service providers, the act would also apply to many other ;login: readers, such as systems administrators, because presumably they have some control over the use of the telecommunications facilities of their employers.

I should also mention that the act provides several defenses to prosecution. One of those defenses says that if you only provide access and don’t exercise any control, you will not be held liable for violating the CDA. Unfortunately, I’ve never seen an instance where an Internet service provider or a systems administrator did not exercise some degree of control over its own facilities. The CDA also says that if a person makes a good faith effort to institute measures which prevent minors from getting access to the objectionable material, they may not be prosecuted for violating the act, or at least some sections of it.

To take advantage of this defense, however, Internet service providers and systems administrators would have to affirmatively immerse themselves in content-controlling methodologies. If we have learned anything from previous cases, it is the principle that the more active you are in monitoring and controlling access to certain kinds of content, the more likely the law is to require you to monitor and control for other kinds of content. Thus, if you were to control for obscene or indecent content so as to comply with the CDA, you would more than likely be held liable for not controlling for such things as libelous statements and the unauthorized transmission of copyrighted materials by means of your system. Both the Prodigy and the Netcom/Church of Scientology cases say that. It’s really a terrible Catch-22.

The bottom line, however, is that these efforts to push Internet providers to exercise increasing vigilance over the con-
tent of the materials transmitted in cyberspace will not only subject those providers to ever-increasing levels of liability, but will also radically change the way we communicate with one another.

Rob: How would our government go about enforcing these new rules?

Dan: As you know, there is a big question about whether the CDA is constitutional. The American Civil Liberties Union and the Electronic Frontier Foundation, among others, have committed to testing the constitutionality of the new legislation, and as early as possible. Given the present membership of our Supreme Court, I'm not placing any bets.

In any event, this will all take a lot of time. We won't know tomorrow or next week, or even next year, whether the Supreme Court will uphold the CDA or not. In the meantime, I expect to see several prosecutions. This is a federal law, so it will be enforced by federal law enforcement agencies. In the Thomas case, a federal grand jury in Tennessee brought indictments against the defendants. We could see a race to federal courthouses all over the country to be first to prosecute under this new legislation. The CDA also invites the Federal Communications Commission to describe measures which would constitute defenses under the CDA, but by law, it would have no enforcement power.

Rob: Is there any impact on the global Internet and communications with other nations?

Dan: There certainly will be. The main impact I see is that unless the CDA is overturned, people are going to have to be much more careful about the kinds of information they permit to be communicated over the Internet. The first consequence is that nobody will be willing to include a large number of USENET newsgroups in the alt.sex area.

But it won’t stop there. The less Internet service providers and systems administrators are treated like common carriers or mere distributors of information, the more they will be forced to intrude. They will have to look at and make decisions about messages and postings which have nothing to do with obscene or indecent sexual content, just to make sure there is no violation of law for which they could be held responsible. This would, of course, fundamentally change the nature of the Internet and the extent to which it is a vehicle for free expression.

In the famous case of New York Times v. Sullivan, the Supreme Court said straightforwardly that the media must be permitted to provide “uninhibited, robust and wide-open debate.” They said it was a prerequisite to the healthy functioning of a democratic society. Let’s hope that the present Supreme Court manages to study that decision before they write their next opinion.

The Webmaster: Traffic Jam on Iway-80
by Dave Taylor
				< taylor@intuitive.com>

Traffic Patterns
I’m going to take a break from the ongoing discussion of CGI programming (and just when we were getting to the good stuff) to talk about Web log files, how to analyze them, and what kind of information you can glean.

To explore log files, we delved into a small subset of the millions of hits of Web traffic recorded for The Internet Mall, a Web site that I run and maintain. Visit <http://www.internet-mall.com/> to add your own visit to the log file.

The Various Log Files
The typical Web server keeps a variety of different log files. I'll be working with the NCSA server, which builds and maintains the following:

- access_log - tracks all users that have accessed your Web site, what they requested, and whether it was fulfilled
- agent_log - logs the specific type of browser they were using
- error_log - records any errors encountered by the Web server during any sort of transactions
- referer_log - logs the Web page from which users arrived at your own Web pages

Other servers keep track of different information, and pre-1.4 versions of the NCSA server don't store the agent_log or the referer_log, but the essential nuts and bolts are the same. If you recall the list of different environment variables sent to the CGI program by the server, you can also see where some of this information comes from. To wit, the CGI program includes the following interesting variables:

HTTP_USER_AGENT=Mozilla/1.1N (Macintosh; I; 68K)
QUERY_STRING=
REMOTE_ADDR=205.149.165.109
REMOTE_HOST=dtaylor.vip.best.com
REQUEST_METHOD=GET
SERVER_NAME=<www.2sprint.net>
The value of \texttt{HTTP\_USER\_AGENT} is stored in the \texttt{agent\_log} file and the value of \texttt{HTTP\_REFERER} in \texttt{referer\_log}. Let's have a look at each.

\textbf{What's in the Agent Log?}

The first file we can dig into keeps track of what browsers people are using to connect to the site:

\begin{verbatim}
% head -5 agent_log
Mozilla/1.0N (Windows) via proxy gateway CERN-HTTPD/3.0 libwww/2.17
Mozilla/1.22 (Windows; U; 32bit)
Mozilla/1.22 (Windows; U; 32bit)
Mozilla/1.22 (Windows; U; 32bit)
Mozilla/1.1N (Windows; I; 16bit)
\end{verbatim}

You can see that the first few entries, at least, tell us that Netscape Navigator (a.k.a. Mozilla) is popular, and particularly on a Windows system. So, as you might expect, we can jump in and use `sort` to get a bit of a better idea of what's going on:

\begin{verbatim}
% sort -u agent_log | head -10
AIR_Mosaic(16bit)/v3.10.06.07
ArchitextSpider
Enhanced_Mosaic/2.00 Win32 Digital/3
Enhanced_Mosaic/2.00 Win32 FTP Software/
Spyglass/3
Enhanced_Mosaic/2.10.10H Win16 FTP
Software/Spyglass/10H
IBM WebExplorer /v1.01
IBM WebExplorer DLI /v1.03
IWENG/1.2.000 via proxy gateway CERN-HTTPD/
3.0 libwww/2.17
Lynx/2-4 libwww/2.14
Lynx/2.3.7 libwww/2.14
\end{verbatim}

You can see that there must be quite a variety of different browsers in use. To find out, try: `sort -u agent_log | wc -l`. The result: 62 different browsers are identified. Want to find out what the two or three most popular browsers are?

\begin{verbatim}
% sort agent_log | uniq -c | sort -rn | head -10
1157 Mozilla/1.1N (Windows; I; 16bit)
932 Mozilla/1.22 (Windows; I; 16bit)
819 Mozilla/1.1N (Macintosh; I; 68K)
538 Mozilla/2.0b3 (Win16; I)
393 Mozilla/1.2N (Windows; I; 16bit)
339 Enhanced_Mosaic/2.00 Win32 FTP
Software/Spyglass/3
313 Mozilla/1.0N (Windows)
231 Mozilla/1.1N (Macintosh; I; PPC)
198 Mozilla/2.0b3 (Win95; I)
180 IBM WebExplorer /v1.01
\end{verbatim}

Now I can extract my first interesting statistic: by using `cut` to slice off everything after the `'/', I can do this count again and find out that:

\begin{verbatim}
7985 Mozilla
467 Enhanced_Mosaic
244 NetCruiser
244 IBM WebExplorer
144 IBM WebExplorer DLL
136 IWENG
114 NCSA Mosaic(tm) for Windows
93 Microsoft Internet Explorer
80 PRODIGY-WB
80 NCSA Mosaic for the X Window System
\end{verbatim}

You can see that Netscape Navigator is by far the most popular browser, a substantial 79% (7,985 of 10,000 entries) on this top-ten list. The second most popular browser, Enhanced Mosaic, is barely on the map at all, with only 5% of the remaining traffic; and the rest are barely of note at all. Or are they? Microsoft introduced their Internet Explorer browser for UNIX and Macintosh, and its configuration options include the ability to custom pick how the browser identifies itself, with the default being, you guessed it, Mozilla!

What's also interesting in this log file is that you can make an assessment of what actual hardware platforms people are using. Notice that in Mozilla entries, the first word in the parenthesis indicates the operating system platform. We can extract that from the log file with a simple, if weird looking, UNIX command:

\begin{verbatim}
% grep Mozilla agent_log | cut -d( -f2 |
\end{verbatim}

You can see that there must be quite a variety of different browsers in use. To find out, try: `sort -u agent_log | wc -l`. The result: 62 different browsers are identified. Want to find out what the two or three most popular browsers are?

\begin{verbatim}
% sort agent_log | uniq -c | sort -rn | head -10
1157 Mozilla/1.1N (Windows; I; 16bit)
932 Mozilla/1.22 (Windows; I; 16bit)
819 Mozilla/1.1N (Macintosh; I; 68K)
538 Mozilla/2.0b3 (Win16; I)
393 Mozilla/1.2N (Windows; I; 16bit)
339 Enhanced_Mosaic/2.00 Win32 FTP
Software/Spyglass/3
313 Mozilla/1.0N (Windows)
231 Mozilla/1.1N (Macintosh; I; PPC)
198 Mozilla/2.0b3 (Win95; I)
180 IBM WebExplorer /v1.01
\end{verbatim}

An interesting result, but I can break it down even further by having a second split at the first semicolon after the `'/', I can do this count again and find out that:

\begin{verbatim}
7985 Mozilla
467 Enhanced_Mosaic
244 NetCruiser
244 IBM WebExplorer
144 IBM WebExplorer DLL
136 IWENG
114 NCSA Mosaic(tm) for Windows
93 Microsoft Internet Explorer
80 PRODIGY-WB
80 NCSA Mosaic for the X Window System
\end{verbatim}

You can see that Netscape Navigator is by far the most popular browser, a substantial 79% (7,985 of 10,000 entries) on this top-ten list. The second most popular browser, Enhanced Mosaic, is barely on the map at all, with only 5% of the remaining traffic; and the rest are barely of note at all. Or are they? Microsoft introduced their Internet Explorer browser for UNIX and Macintosh, and its configuration options include the ability to custom pick how the browser identifies itself, with the default being, you guessed it, Mozilla!

What's also interesting in this log file is that you can make an assessment of what actual hardware platforms people are using. Notice that in Mozilla entries, the first word in the parenthesis indicates the operating system platform. We can extract that from the log file with a simple, if weird looking, UNIX command:
Now you can see the kind of systems people are using. The agent_log I’ve been working with has a total of 10,000 entries, of which 7,985 are shown in the summary above. I can get other interesting statistics:

- 43% of the people who visited were running Windows 3.x
- 16% had a Macintosh
- 10% were using a UNIX machine with X11
- 5% were running an older version of Windows (Win16)
- 2% had Windows 95 running on their machine

All that, just from the simplest of the log files!

**Where Are They Coming From?**

With dozens of different search engines and countless hot links on different people’s home pages, it’s very useful to know where people are coming from when they get to our site. That’s what the referer_log file is for. Here’s a snippet of its contents:

```bash
% head -5 referer_log
-> /Graphics/imall-head.gif
-> /lcmcbks.htm
-> /sponsors/netscape.gif
-> /Graphics/new.gif
-> /Graphics/books-button2.gif
```

The log tracks the links from referring documents to the documents found on this server, so what I’ll need to do is omit all the internal references so I can focus on those that aren’t on our own site.

```bash
% grep -v internet-mall referer_log | cut -d\ -f1 | sort | uniq -c | sort -rn | head
```

The counts are surprisingly low, but the fields themselves are quite interesting. You can see that InfoSeek is where the majority of our references come from (though because all these counts are so low, we can draw the assumption that the majority of visitors are either coming to us via a bookmark/hotlist entry or typing in our URL directly).

**What’s Gone Wrong?**

One more log file that’s worth keeping an eye on is error_log, which looks like this:

```bash
% head error_log
[Tue Feb 6 18:36:03 1996] httpd: access to /cgi-bin/imall/quantum.gif failed for piweba6y-ext.prodigy.com, reason: script does not exist from -
[Tue Feb 6 18:36:06 1996] httpd: access to /cgi-bin/imall/thuridion.gif failed for piweba6y-ext.prodigy.com, reason: script does not exist from -
[Tue Feb 6 18:36:13 1996] httpd: access to /cgi-bin/imall/imall.gif failed for piweba6y-ext.prodigy.com, reason: script does not exist from -
```

What’s interesting is that some of these sites aren’t even pointing directly to the home page, but rather to other specific spots within our organization: a good reminder of why you need to assume that people can get to your site through any page, and that all pages need to clearly identify the site and offer useful navigational tools.
FEATURES

send aborted for 205.242.230.117
[Tue Feb 6 18:38:02 1996] httpd:
send aborted for 205.242.230.117
[Tue Feb 6 18:38:14 1996] httpd:
send aborted for 151.110.11.188

It’s cryptic, no question, but you can see that there are some mysterious references to quantum.gif and thuridion.gif in the cgi-bin directory, references that can’t be resolved. You can also see the culprit: someone on Prodigy! This is a good file to keep an eye on, but usually reveals nothing too scintillating.

The Real Log File

The file that contains the most interesting information, the one that gives you hit counts and more, is the access_log file. Here’s what it looks like:

```
% tail -5 access_log
lictor.acsu.buffalo.edu - -
[06/Feb/1996:18:43:33 -0600] "GET /4-servcs.htm HTTP/1.0" 200 4368
lictor.acsu.buffalo.edu - -
[06/Feb/1996:18:43:33 -0600] "GET /sponsors/mastcard.gif HTTP/1.0"
198.83.18.80 - -
[06/Feb/1996:18:43:36 -0600] "GET /Graphics/search-button.gif HTTP/1.0" 200 2350
```

This Common Log Format is confusing for a human to read, but the general form of each entry is:

```
<from> <ident> <uid> <date/time> <request> <status> <bytes-sent>
```

The ident and uid fields aren’t logged as part of an httpd transaction (because it’s all from non-logged-in users), so they’re blank. The status field is really the only other field that isn’t self-explanatory; it can have a couple of different values: 200 (document ready) is the usual status code. A 3xx status indicates some kind of redirection; 4xx indicates some kind of request error, and 5xx indicates some kind of server error.

You can extract whatever you want by picking the right field or fields. First off, total hits is simple: `wc -l`. Rather than work with the entire file here, I’ll slice off a 20,000 subset of the log file using the `head` command. This represents less than a day of traffic on this site, the time between 00:00:06 and 14:34:30 on December 1, 1995 (this means this site saw about 35,000 hits that day alone).

The actual file from which I extract this information shows the total number of hits for December and January combined to be 2,892,389.

Let’s look at the very first field first. It indicates the hostname of each person who connected to the site and requested a file. The ever-handly `cut` command can pull out just that first field, then we’ll want to strip the leftmost portion of the hostname because that’s almost always some tty or temp IP identifier. Again, we can use `cut`:

```
% cut -d\ -f1 access_log | cut -d. -f2- | sort\ / uniq -c / sort -rn / head
581 ix.netcom.com
461 proxy.aol.com
206 compuserve.com
191 netvision.net.il
181 ny.us.ibm.net
150 direct.ca
138 pgh.wec.com
136 prodigy.com
128 singnet.com.sg
126 sciatl.com
124 emea.ibm.net
122 vip.best.com
```

You can see that the most common place for people to visit from is Netcom, with America Online and CompuServe rounding out the top three groups. The .il is Israel, the .ca is Canada, and the .sg is Singapore.

What Was Most Popular?

That answers the “where?” question, but what about a quick look at “what?” Again, a well-placed `cut` can do the trick:

```
% cut -d\ -f7 access_log | sort / uniq / sort -rn / head
827 /Graphics/imall-head.gif
771 /sponsors/necxmweb.gif
727 /Graphics/index-button.gif
727 /Graphics/add-shop-button.gif
721 /Graphics/rand-o-mall-button.gif
720 /Graphics/new.gif
720 /Graphics/about-button.gif
718 /Graphics/searchfor-splash.gif
705 /Graphics/search-button.gif
698 /Graphics/rand-o-mall-splash.gif
```

That’s interesting, but it isn’t quite what I seek. Instead of a count of shared graphical elements, the non-GIF files are
really what would be of most interest. So I need to add a
grep -v gif to the sequence, and:

779 /index.html
541 / 
228 /3-comptr.htm
224 /searchfor.html
194 /5-cloths.htm
169 /2-personl.htm
141 /1-books.htm
133 /5clths.htm
113 /2dltrntdprdcts.htm
96 /whatsnew.htm
93 /food.htm

This is interesting and shows one of the limitations of the
server: the first two entries are synonymous, because a
request for "/" defaults to opening the index.html file on
the majority of server configurations, so in reality, this first
entry should indicate 1,320 hits. Because I know that the file
itself contains only 20,000 hits, this means that the most
popular file request represented a mere 6% of the traffic,
which shows the varied nature of this particular site (which
has almost 250 different Web pages available).

Another calculation I can do is the ratio of graphic file
requests to text requests.

% cut -d\ -f7 access_log | grep 'gif' | sort | 
  uniq -c | awk '{sum+=51}END {print sum}'
16619

% cut -d\ -f7 access_log | grep -v 'gif' | sort | 
  uniq -c | awk '{sum+=51}END {print sum}'
 3381

When the 20,000 hits total is taken into account, this indi¬
cates that 83% of the file requests were for graphics. And
this is a very text-intensive Web site! Go to a photographic
archive, and this percentage will be dramatically higher, I’d
bet.

Finally, let’s end this by considering the total number of
bytes transferred by the Web server in this period of time.
The command itself should be easy to build by now; it’s a
simple variant on the above cut command with the
extracted field being f10 rather than f7. Add up the transfers
for the 14 hours in the clipped log file, and it’s 75,271,042
bytes, or over 75 megabytes. Extrapolating from there, I can
assume that the server and network connection see a transfer
load of 129 MB/day, or, looking at it the other way around,
89 kilobytes/minute!

All these numbers aren’t 100% true and accurate, however,
because proxy and client caching (particularly on AOL) can
dramatically affect the counts: the server might request and
store a single copy of each file; then any other requests dur¬
out hitting your server and, of course, without logging the
hit and the traffic.

Wrapping Up

I hope this has shown you some of the basics of how to get
around the Web log files and what you can extract from this
seeming mishmash of different fields and puzzling snippets.
You might not have a full set of all these values, but even
some of the lower-budget Web servers you can rent space
on, like Best Communications, in Mountain View, Califor¬
nia, drop a nightly log file into your home directory and
leave the analysis up to you. It’s straightforward, and you
can have lots of fun producing interesting and informative
statistics.

Next issue, I promise to return to the topic of CGI program¬
ing by looking at what’s needed inside a program to actu¬
ally receive, unwrap, and process information sent by the
user.

Many thanks to Tai Jin and James Armstrong for their erudite
feedback during the analysis of these log files.

Dave Taylor clearly has UNIX on the brain when he could
just pop over to a site like NCSA and find a large list of
existing Web log file analysis tools, all easily found at:

<http://union.ncsa.uiuc.edu/HyperNews/wwwwlog-analyzers.html>

You can point this out to him, if you must, at
taylor@intuitive.com.

Musings

by Rik Farrow
<rik@spirit.com>

My, but how fast the world moves! The last issue of ;login:
had an opinion piece (Scott Hazen Mueller) entitled “Can
UNIX Survive?” This week’s (February 12) ComputerWorld
has the HP/SCO axis aligning itself against Sun and IBM. HP,
SCO, and, not coincidentally, Intel would like to see one ver¬
sion of UNIX, one controlled by HP/SCO, unifying the mar¬
ket while being used to smash the competition. Hello, does
this make sense to you?

Vendors stand to gain a competitive advantage by not hav¬
ing their operating system interoperate well. This was a
tried and true tactic of IBM and DEC, but is (supposedly)
anathema in the “Open Systems” market. And here they go
again. It would have made much more sense for HP to
embrace their largest competitors, cooperate on the operat¬
system, and then slug it out in price-performance and
quality of service rather than seeking an edge via incompatibility. This is the type of thinking that has led to the partial success of Windows NT today.

And does it really matter which version of the UNIX operating system you boot? What matters is that you can build the programs you want to build, and run the software you need, efficiently. What also matters is that you can easily administer to a heterogeneous network of computers. If UNIX vendors would simply provide us with a single administrative and programming interface, they could do anything else they want to differentiate themselves in the marketplace.

There are moves afoot to do this. The Common Desktop Environment and the Common Management Environment would take us in this direction, and at least CDE is moving along okay. But it certainly upsets me that HP would restart the UNIX wars. I expect that HP plans to swallow SCO whole once SCO's market has been taken over by BSD/OS and Linux whose UNIX-like systems are superior in many ways to SCO.

What is UNIX anyway? We have seen UNIX on top of microkernels (MACH, CHORUS), and now we will be seeing UNIX on top of Windows NT. Softway Systems Inc. (San Francisco) is working, under contract to Microsoft, to create OpenNT so that UNIX applications can run unchanged on top of Windows NT. OpenNT may result in resolving Heinz Lycklama's complaint that Windows NT should not have been declared open by the judge in the Coast Guard case (see “Open Systems, POSIX, and Windows NT — Another Point of View” by Heinz Lycklama in February's ;login:).

**The San Diego USENIX Conference**

I have my own crystal ball, of course, and I’ll share a peek into it with you at the end of this column. I wanted to talk about the latest brushfire in the UNIX wars because it leads in nicely to the keynote speech given by Van Jacobson. Van Jacobson drew a nice analogy between the design of speech in humans and the design of today’s networking. Humans, it appears, come with some basic, hard-wired capabilities for understanding and using language.

Van Jacobson provided the fascinating analogy of split-brained epileptics, whose speech-centered left brain had been surgically severed from the gestalt-oriented right brain. If you allowed many of these patients to see an object with their left eye (which is connected to the right brain), they could not describe it. But some could. This mystery was solved when someone noticed the right brain-controlled left hand tracing the object onto the right palm, bypassing the cut in the brain.

Just as speech appears to rely on deep, unconscious structures, networking also relied upon the UNIX system. The UNIX system provided the structure for TCP/IP to build upon after 1982, and the UNIX philosophy affected the design of networking. Van Jacobson’s most memorable quote was “Microsoft gives you sentences, and UNIX gives you words.” Anyone who has struggled with Visual Basic clearly understands this. And anyone who appreciates the flexibility of the UNIX system knows that UNIX, or something like it, will never go away.

Speaking of UNIX-like operating systems, Linus Torvalds gave an invited talk discussing the past and future of Linux. Torvalds said that networking was still Linux’s weakest part, but it was being worked on now. His real interest is the kernel itself, and he is looking forward to adding symmetric multiprocessing features (good luck!), improved file system caching, and a clone() call that will be like Plan 9’s rfork().

Mike Karels, of BSDI, talked about the new release of BSD/OS, 2.1, which arrived about a week after I returned from San Diego. The new release includes support for many new devices, improved virtual memory, and user classes.

**More Talks**

As usual, it was difficult to choose between the invited talks track and the refereed papers track. I always rationalize that I can read the papers out of the proceedings anytime and even send the authors email if I have any questions. I found the implementing IPv6 paper fascinating (Atkinson, et al.) and was also interested in the Web papers presented on Thursday afternoon. The lambench, a paper describing portable tools for performance monitoring, won Best Paper. I don’t mean to slight any of the papers chosen (or even any of the many passed over in the wealth of submissions), but these are the ones that caught my fancy.

John Ousterhout, father of Tcl and now employee of Sun Labs, gave a fairly balanced presentation entitled “Why Threads Are a Bad Idea (for Most Purposes).” It was not lost on most of the large audience that Tcl when used with Tk relies on events, rather than threads. But what Ousterhout had to say was quite relevant. Threads destroy determinism in programming, which is important for writing device drivers, but makes life difficult for everyday programmers.

In a threaded application, one of the big advantages is that the threads can share data. But this is also a big problem because you must coordinate modifications to shared data. The general solution is to use locks to protect data, which leads to a second problem — deadlocks. If two threads are interdependent, and one waits on a lock that the other thread
won't release until the first thread responds, you have circular deadlock. Threads also break abstraction, that is, all objects sharing data or locks must be aware of the internal behavior of other objects (to avoid deadlocks).

Ousterhout instead suggested the use of event-driven programming. Although event-driven programming is also a new paradigm (for some), GUIs are familiar and event driven. Events also avoid concurrency, the overhead of locks, and make debugging easier (timing dependencies are related to events, not thread scheduling). Ousterhout concluded by saying that threads are a necessary evil for certain type of programming (operating systems, high-end servers), but should be avoided in favor of event-driven programming.

Bruce Schneier presented his "Cryptography in the 21st Century" talk, which amounted to a very good introduction to cryptography with a forward-directed viewpoint. Besides the obvious problems with governments, we face other serious impediments to the use of strong encryption. Bad implementations (Netscape and key generation), key management (a killer problem), key escrow by governments, and lack of standards all stand in the way of our having strong encryption for commerce and privacy today.

Tom Killian, of AT&T Research, gave a technical talk about asymmetric broadband to the home. I thought at first that this would simply be a repeat of a talk given at LISA IX, but it was different. Killian focused on the human and technical difficulties of getting a cable vendor to participate in a fashion that makes getting a 10-megabit per second feed via your cable possible for the ten participants. Each connection has three times as many places to fail (a modem used for packets returning from the homes, the cable box, the terminal server, the routers at the cable plant, and the cable plant itself). The cable plant, which includes miles of fiber and coax, is itself a big problem. Although you can watch a snowy picture (you often don't have much choice), a noisy or weak signal will greatly degrade network performance. Killian showed diagrams of weak or noisy signals that could get by for TV but won't work well for networking.

Cable operators cannot provide the level of quality needed to support one-way networking today for small groups of people. It's hard for me to imagine symmetric networking to the home except in extremely competitive test markets anytime soon. More hype bites the dust.

**Cyberspace**

A small group calling itself Digital Circus Productions reported on their experiences with children in CitySpace. CitySpace allows young children, interacting over the Internet, to construct and explore a virtual city environment. The children can use a variety of tools to generate three-dimensional objects that become part of the CityScape and then can move through the CityScape with a virtual presence, an avatar that appears as a car or a jet. The San Francisco-based group, working with donated time and equipment, hopes to move into virtual reality when the technology becomes available.

The last invited talk was certainly a lot of fun. Keith Bostic created a technical executive summary, which Peter Honeyman got off to a roaring start. Honeyman addressed Mobile Communications glibly, tossing each slide in the air during his fast-paced summary. We don't need to talk about wireless communications, that's for electrical engineers, said Honeyman. He went on to discuss different mobile techniques, including a mobile base station (current technology), teleport for partially or weakly connected (slow, radio-link) operation, and deferred writes (when the network is down). He also mentioned Odyssey, a file system that adapts to network conditions.

Bill Cheswick followed with his talk on commercial munitions. Cryptography is munitions if you look at it in the right frame of mind. He included a disclaimer (his talk was not sponsored by Captain Zeiss or the Air Force Information Warfare Center). A good offensive weapon would be TCP/IP packets with unusual options, length, and flag bit combinations. Cheswick suggested getting permission to use strong encryption for key exchange and living with "weaker" encryption for the actual data (as a way of placating governments that limit export of crypto munitions).

Cheswick listed methods for breaking into UNIX systems, starting with the easiest: sendmail, guessing passwords, FTP, Telnet, cracking passwords, social engineering, NFS configuration, the portmapper, DNS attacks, IP spoofing R commands, park in Manhattan, and break secure RPC. Cheswick also mentioned that he hopes the "new" company 'B' (result of the AT&T voluntary breakup) will make it easier to release in-house software such as stelnet (secure Telnet, Bellovin and Blaze), cget/cput (simplified FTP, Ranum and Cheswick), and stage, a tool for created mirrored archives, great for updating the Web server on the other side of the firewall. Encryption was high on Cheswick's list of preventive security measures.

Dan Geer came next, speaking about marketing trends. Geer joked about how paranoid Americans are about using credit cards on the Internet. (Many Europeans find this funny, too, considering whom we share our credit cards with, like telemarketers, gas station attendants, underpaid mall employees, you name it.) Geer pointed out that it is too late to create the infrastructure we need today, such as public key distribution. He also stated that 80% of Web servers purchased are used internally. Geer mentioned the miracle
that would change client/server programming – when all business applications include a Web browser interface.

Dan Appelman talked about the law, beginning by saying that the law doesn’t lead, it follows. The Communications Decency Act makes it easier to be convicted for “indecent” content, with the definition being reduced to “knowingly permitting an activity” that violates the law (up to two years in prison or a $100,000 fine). Prodigy apparently won their case (where they “permitted” a posting that maligns a company) – Prodigy had to apologize. The Church of Scientology won their case against some of the defendants. Intellectual property, secret teaching of the church, was published on a BBS, and the individual and the operator of the BBS were held at fault for a copyright violation, but not Netcom, which provided an Internet connection for the BBS.

Jim Waldo of Sun provided a sip of Java, and began with some simple assertions. First, there’s nothing new in Java, and he challenged anyone to find something that was. Second, people don’t care about operating systems: just look at what most people buy. C++ started out okay, but it just kept growing. As far as Java goes, let the Web check it out until it’s absolutely right. Sun will change it if it’s not right. Waldo also mentioned that he asked someone to do a performance comparison of Java and C and found the Java was 40 times slower.

Operating Systems

Jim Waldo’s right, I think. Who really cares about operating systems? The people who write them, and so do people selling them. But people who use computers just want to get the job done quickly and efficiently. (Here’s where I gaze into my crystal ball.)

I love the way the UNIX system gives me words. I’m a tool user, and between the standard UNIX tools and the “free” tools, there’s little I need to buy (you might be amazed). But what about all those desktops out there? Do they really need UNIX? The market said no, and I agree.

Most of the desktops don’t need Windows either, and they certainly don’t need NT, Microsoft’s planned replacement for Windows. The future desktop will look more like the Internet appliances we’ve been hearing about, and their operating systems will look more like Plan 9. You might remember that in Plan 9, there are display servers, CPU servers, and file servers, each containing its part of the operating system, all unified under a shareable namespace. The key concept is specialization in a cooperative, networked environment.

In the desktop of the future, there won’t be a local disk (or at least not one storing anything other than paged-out memory), and there will be just enough of a microkernel to upload the application-of-the-day; client/server with no desktop maintenance. You turn on the desktop, it gets what it needs from centralized servers, and off it runs. No games, no viruses, no software distribution, no local disk backups, no device driver headaches, and true centralized administration and security.

Sun plans to be a big part of this picture with Java, which is designed just for this task. Sun has announced chips designed to run the Java Virtual Machine, so Java’s wimpy performance will be a thing of the past. So will Intel and Microsoft, if Sun has its way.

The Plan 9-style approach makes me very happy because companies will require fast, flexible, networked servers to make all this work properly. UNIX servers, perhaps, keeping us all happily in the UNIX business, and not in the business of learning and supporting Windows on thousands of PCs, all configured differently. I’ll put away the crystal ball after letting it cool down for a few hours.

A Major Contributor

by Greg Rose

<Greg_Rose@sydney.sterling.com>

A few months ago, an article about the 20 most influential people in the computer industry appeared in Byte. Now, a lot of articles like that are pretty subjective, so I didn’t think about it straight away. Later, though, an interesting thought occurred to me. Dennis Ritchie was mentioned, but Ken Thompson wasn’t. Strange.

I’m lucky in a way. I was walking along a corridor at the University of New South Wales 20 years ago when a couple of mad computer scientists kidnapped me and told me I had to write programs for this new operating system called Unix. Its name hadn’t been uppercased at that time. This was one of the first places in the world to be running UNIX, and the staff at the university was very excited by it. I wasn’t excited – I was just a second-year programming student. For the researchers, though, UNIX represented their first real chance to get close to the machine; up until then, serious computers lived in shielded rooms on hilltops and were run by data processing units whose members had no desire at all to let academics touch them. But I digress.

After I had been involved with UNIX a few years, the Australian UNIX Users Group, AUUG, was formed, named, and eventually incorporated. I became first secretary, then for a few years president. Clearly I was now a mad computer scientist in my own right, since no one else would be silly enough to donate so much time to such an effort. There was a payback for this, though. When we had our conferences,
was usually in a position to meet interesting people. Among them were Ken Thompson and Dennis Ritchie.

The thrust of the Byte snippet regarding Dennis Ritchie was that he was the creator of UNIX, C, and Plan 9 (Bell Labs' new operating system). I think there was a “with Ken Thompson” thrown in there somewhere. This viewpoint is just slightly revisionist. Part of the problem is that both Ken and Dennis are basically quiet, humble, and shy people. During an interview when Dennis was last in Australia, in 1994, he was asked something about his involvement in Plan 9. “I really don’t have much to do with its development; it’s mostly the other guys,” Dennis said, and as I personally know, completely truthfully. “The Humble Creator of Plan 9” was one of the titles he was addressed by during the next few days.

I have the privilege of counting Dennis Ritchie among my friends, and I hope it will stay that way. But Ken Thompson is a friend, too. I can’t let the Byte article pass unchallenged. Dennis is a “mover and shaker” of the computer industry, but Ken has done more, with less publicity.

If you look at the source code of UNIX at the time I spoke of, the programs that made it up are divided into two directories, called “ken” and “dmr” (Dennis’ initials). The amount of code splits about 2:1. Go further, and you discover that the scheduler, the memory management, the assembler machine support, the file system, and so on are in “ken.”

Dennis Ritchie is the designer of the C language and implemented the first compiler for it. This implementation was required to rewrite UNIX in a high-level language. Ken Thompson collaborated closely with Dennis Ritchie on the architecture of the language, to get it to the point where it could be used for that purpose. There might have been a UNIX without C (in fact, there was, for the first few versions), but there would never have been a C without UNIX.

Ken Thompson has been a developer of Plan 9 since its inception, along with a number of others, notably Rob Pike, Dave Presotto, and Phil Winterbottom. Dennis Ritchie is their manager.

Ken Thompson has a large number of other achievements that are not well known. He led the world in computer chess for a number of years. This research has had a number of spin-offs, and his chess computer was prevented from being sent to Russia for a tournament because of its advanced technology. Recently, he has been working on compression algorithms for digital radio broadcasts. This compression allows nearly a full day of CD quality music to be compressed onto a standard CD-ROM. (He also interfaced a prototype and a PC to a mini jukebox so that his departmental secretary can play ‘50s and ‘60s music “with an appropriate user interface.”)

There is little more to be said. I just want to go on record saying that Ken Thompson should have been on that Byte list.

Highlights from the Fourth International WWW Conference

by Yonah D. Karp
<yonah@u.washington.edu>

[Editor's note: The following article provides summaries of some of the talks given in Boston in December 1995.]

Tutorials

“Security, Authentication, and Privacy on the Web,” given by Adam Cain of NCSA, won Best Tutorial, and deservedly so. It was a superb overview of the basic issues of security on the Web and covered system security/integrity, authentication mechanisms, access control and authorization, and privacy technologies. Included were discussions of cryptography, secure-HTTP, Netscape’s secure socket layer (SSL), Kerberos, and digital payment. Cain is entertaining, knowledgeable, and well organized and combines earthy wisdom with technical information. Catch Cain when you can, and you can see his tutorial slides at <http://www.ncsa.uiuc.edu/InformationServers/adamlw4/>.

Keynote

The keynote address was given by a technowiz, Bran Ferren, VP of Imagineering at Disney. His talk, “There’s No Bits Like Show Bits,” addressed the hype, reminding us of what we already have and also what we may not want: we need to take off the rose-colored glasses and look critically at what new technologies can, and cannot, do for us. “Have you ever sent a fax from the beach?” the ad asks. Ferren pointed out that he had really never had any desire to send a fax from the beach, but that we should ask the question, “Have you ever received a $175,000 phone bill?” The answer, of course, is, “You will.”

People, “the referenced standard for interactivity,” are technically complex, Ferren said. He admonished us to remember this as we design our own labyrinthine software, considering not only whether what we create will function well, but what effects it may have on the world around us. Technology will never replace human interaction. Although
the Web could be "power steering" for teachers, we need to remember how children will respond to, and learn from, what we create.

Ferren encouraged us not to get so caught up in creating technical gee-whiz that we forget to include emotional quality for the humans who will be using that technology. (If the Web starts to look a lot like TV, we need to figure out how to enhance its quality so it doesn't have the content problems TV has.) We need to remember how to tell stories. And we should look critically at technologies that provide solutions to no corresponding problems. Finally, he exhorted us to learn about design and to aim high. And most important, we need to remember that "the killer app is life."

Papers, Panels and Hot Air

Because there were six tracks, it was impossible to see very much. However, much of the conference is on the Web. To browse on your own, follow links from <http://www.w3.org/pub/Conferences/WWW4/Program.html> to find individual papers.

Online vs. Internet - Prodigy CEO

In a lunchtime talk, Edward A. Bennett, CEO of Prodigy, spoke about "Converging Online and Internet Worlds." It was interesting to see how the proprietor of a large online service justifies its continued existence. Bennett spoke about the value of online service providers in the world of the Net. He believes aggregators have real value for their delivery of bundled services. Examples are shopping malls, CNN, online brokerage services, and online service providers.

He used VCR+ as an analogy for online services. "No one gives away content for free," he states. Well, maybe. USENET, however (where loads of useful information is given away for free) is where I've learned much of what I've needed. Bennett also spoke about secure transactions and secure commerce as draws for users of online services, as well as the packaging/marketing aspect - comparing online services to the aggregators HBO and Nickelodeon. He sees combining the best of online with the best of Web.

Intelligent Searching

A panel called "Knowledge Representation and the Web" addressed the question, "How can we turn information on the Web into knowledge we can use?" Good question. Anyone who has waded through masses of URLs looking for a code fragment using a function call knows what kind of a problem we have.

The panel looked at how knowledge representation can guide evolution of World Wide Web applications and protocols to make networked information more accessible, flexible, and useful. Bob MacGregor of the University of Southern California conjectured that Web query tools using indexing schemes based on category hierarchies (taxonomies) will proliferate, resulting in very large taxonomies and large numbers of independently constructed and managed taxonomies. He talked about searching, engines, composing a search, semantics of composing, and evaluating Web query tools.

Bill Woods of Sun talked about the paraphrase problem, demonstrating some interesting ways of finding of what you want on the Web. Synonymy is not the issue, he said; it's the issue of generality. The trade-off is granularity. He discussed taxonomic, lexical, and synonymy issues, as well as subsumption algorithms. A search on "large disk," using his rules, will find "Gb SCSI," which may be what you want. He posits that better lexical, logical, and taxonomic rules are needed to make searches find the right thing. Find out more at <http://www.w3.org/pub/Conferences/WWW4/Papers/krp/>.

Realtime Video on WWW - Vosaic

By far the most gee-whiz paper I saw was "Realtime Video and Audio in the World Wide Web," by Zhigang Chen, See-Mong Tan, Roy H. Campbell, and Yongcheng Li. This presentation won Best Paper, which it deserved. The authors presented Vosaic, a modification of Mosaic that does continuous media on WWW. Music and movies run within a Web page with embedded clickable links (within the video stream). It integrates realtime video and audio into HTML. The transmission protocol is called VDP: on-demand video with audio. The demo clip presented would run for half an hour. The embedded links within video streams are really exciting. Right now they are available only for SGI, but we should look at them anyway - this is the future.

For more information, consult the following web sites:

<http://indy1.cs.uiuc.edu:8080/vosaic_home/>
<<http://www.w3.org/pub/Conferences/WWW4/Papers/2111/>

Imaging on the Web - Kodak

Christopher Dobbs of Eastman Kodak talked about the future of imaging on the Web. We've focused on text on the Web due to low bandwidth, he says, and are nearly at a place where we can incorporate more and more imaging. If we have enough imaging, the Web starts to look like TV, and
digital TV is just waiting for the infrastructure to make it happen. Dobbs mentioned HTTV, poor progressive display, progressive JPEG, PNG format, NSC/A/Kodak collaboration. Photo CD on the Web is available, free.

Dobbs showed a demo where he enlarged, then cropped, digital images with little or no degradation of the image. An example was a group of rafters sampled to the detail level to show a single rafter’s face. The technology involves scanning, then decimating the image, then going out to the world not as RGB, but as YCC, a well-defined starting point for color management. YCC guarantees colors will be true.

Dobbs has worked with the Louvre, Metropolitan, and Ford museums to get their images available on the Net. Kodak has a Java applet with the same functionality.

As far as size, a photograph will be digitized into a set of image packs at various detail levels. The original set, 24 Mb, can be condensed to a 4–5 Mb image pack, with visually lossless compression. Other compression technology (truncated image packs?) is used to get the size down: 4–5 Mb → 1.5 Mb → .75 Mb.

He sees the Web delivering image objects, not images, which will flow outside your “browser.”

He also sees some of the hype about the Web and related technologies as the emperor’s new clothes. Virtual reality is more like a virtual nightmare, he says. You’re bumping into things trying to find something, and eventually you find that there really is nothing to find.

Finally, he stated that Java extends functionality. But more important, it extends evolution. He sees the Web as an evolving organism.
The following reports are published in this column:

- POSIX and IETF
- PASC
- ANSI IISPI
- POSIX.1k: Removable Media
- POSIX.1h: SRASS
- POSIX.1a: System Services API
- APR-01: Whisky

Our Standards Report Editor, Nick Stoughton, welcomes dialogue between this column and you, the readers. Please send any comments you might have to <nick@usenix.org>.

An Update on Standards Relevant to USENIX Members

by Nick Stoughton
USENIX Standards Report Editor
<nick@usenix.org>

Snitch Reports, POSIX, and IETF

by Nick Stoughton

USENIX has been funding work in the POSIX standards activities for some years now. At its peak, POSIX attracted around 350 people, actively building vitally needed standards.

That peak has now passed. Much of the work is done. There will always be a need for standards development in this area, but at a lower rate than before. And with 28 published standards, the Portable Applications Standards Committee (PASC, who are responsible for POSIX) can never just “go away.”

So where is the current peak of standards activity? Is there one? As I write this, I am about to attend the thirty-sixth Internet Engineering Task Force (IETF) general meeting in Los Angeles (look out for some snitch reports in the next issue). Here is a group with a model very similar to POSIX, although less formal, of building on solid existing industry practice. The Internet is right now where UNIX was a few years back with respect to standards.

The IETF provides a forum for working groups to coordinate technical developments of new protocols. Its most important function is the development and selection of standards within the Internet protocol suite.

The IETF began in January 1986 as a forum for technical coordination by contractors for the then US Defense Advanced Projects Agency (DARPA), working on the ARPANET, US Defense Data Network (DDN), and the Internet core gateway system. Since that time, the IETF has grown into a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

USENIX has asked me to investigate how the IETF activity should be reported here. The POSIX standards are limited to Application Program Interfaces and produce standards and guides for application portability. An increasing proportion of USENIX members’ work is now focused on distributed applications, and the influence of Internet Protocols is an area of standardization that badly calls for a similar level of coverage.

A brief poll during and subsequent to the San Diego Technical Conference suggests that there is considerable interest among the members on the activities of a number of the IETF working groups. For instance, in just one area, that of operational requirements, SAGE would be very interested in the Guidelines and Recommendations for Security Incident Processing (GRIP) working group, which is developing guidelines for incident response teams. Likewise, Procedures for Internet Enterprise Renumbering (PIER) is looking at network renumbering technology issues. The Remote Authentication Dial In User Service (RADIUS) group is doing work that impacts directly on those using terminal servers, and the Real
The general IPv6 development work should be of interest to anyone with responsibilities for making network environments work. This list just goes on and on. So watch this space!

Standards at Work

During the recent UniForum conference in San Francisco, a small company called Softway Systems announced that it would be shipping a product called “OpenNT,” that provides, at least in the first release, POSIX.2 support for Microsoft’s Windows NT. Is this the end for UNIX? Have standards so damaged our credibility that we have to accept Microsoft as a player in the UNIX game? Well, if so, we’ve done our job, and done it well. This is what standards are all about: opening the field to competition.

As an application developer, I’m very happy to have another platform on which my applications will be able to operate. I’m even happier that I have another platform on which I can work and feel at home. So what if my UNIX system has a Microsoft badge on it too?

As an end user, I’m even happier to be able to pick and choose between the vendors and be able to buy what I want for the price I am prepared to pay for it.

I’ve worked for many major vendors as a consultant over the years; all this will do is make me have my system work better and cheaper than theirs. So everyone is happy.

This is a major triumph for standards and open systems.

Report on PASC: A Message From The Chair

by Lowell Johnson
<3lgj@rsvl.unisys.com>

We in the IEEE Portable Applications Standards Committee (PASC) are trying to be very proactive in solving the problems of creating timely standards: from the point of view of industry, the IEEE, and PASC in particular. Such things as electronic balloting and electronic monthly mailings are in the midst of trials, which I may report on later (if there is interest).

However, these are process improvements using the current working group model. We must also explore the possibilities of basic (and perhaps drastic) changes to the model itself. The following proposal for a new model tries to incorporate the best aspects of one of the consortia models with the current IEEE working group model.

The major problem with the IEEE standards development process, and often the only perceived problem, is the length of time it takes to move from the start of a new project to the final approval of the completed standard. The process is basically the same now as it has been for decades, although the economy and business models have drastically changed during that time. It typically takes three or four years to create a standard, and sometimes even longer, which is simply inadequate in the modern era of high technology and high rate of change.

The problem stems from what otherwise is the real strength of the IEEE: complete openness and individual membership. Any individual who meets some fairly basic requirements may work on and/or vote on a standard. These individuals are considered volunteers, but they are often supported by their employers to spend a small part of their time on projects that their employers think may be beneficial to their companies or to the industry as a whole.

PASC is fairly typical: in terms both of our process and the problems we now face because of it. Our working groups meet once a quarter and hold lengthy discussions to determine content and resolve issues. Then individuals are assigned to write material, review ballots, etc, before the next meeting. In years past this model worked well because most people actually had time to do significant work between meetings.

However, in the modern environment of “downsizing” and insecure employment, few people have any time at all to spend on standards when they go back to their “real jobs.” The current economic reality also causes companies to send fewer people than ever to work on standards, and thus the standards take ever longer to develop and ballot.

We need a new meeting and work model, at least for some projects, if we want IEEE standards to continue to be a viable method of de jure standardization. I propose that in some circumstances we adopt a model similar to that used by some of the industry consortia: dedicated personnel for short-term project completion.

Many of the officers in IEEE projects, such as working group chair and sponsor chair, currently require some sort of letter of support or commitment from their companies stating that adequate time and travel expenses will be available for the individual to fulfill the requirement of the position. I propose that for some projects we extend this to all personnel working on that project.

The intent of this would be that the group would not only meet more frequently (either electronically or in person), but that all individuals would be able to spend a majority of their time between meetings working on that standard. This would mean that major writing assignments, reviews, and balloting could be accomplished in monthly time frames;
therefore, an entire standard could be completed in about one year.

Assuming that the project was well defined at the time the PAR was approved and that meetings were held monthly (either in person or electronically), the following timetable may be a model we strive to achieve:

<table>
<thead>
<tr>
<th>Meeting # (month)</th>
<th>Expected input or meeting topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>Well-defined project, base inputs</td>
</tr>
<tr>
<td>Two</td>
<td>Draft 1 (partially complete)</td>
</tr>
<tr>
<td>Three</td>
<td>Draft 2 (100% complete)</td>
</tr>
<tr>
<td>Four</td>
<td>Draft 3 (revised &amp; ready for mock ballot)</td>
</tr>
<tr>
<td>Five</td>
<td>Draft 4 (revised &amp; ready for IEEE ballot)</td>
</tr>
<tr>
<td>No meeting</td>
<td>In ballot at IEEE</td>
</tr>
<tr>
<td>Seven</td>
<td>Ballot resolution and draft revisions</td>
</tr>
<tr>
<td>Eight</td>
<td>Draft 5 (ready for recirculation)</td>
</tr>
<tr>
<td>No meeting</td>
<td>In ballot at IEEE</td>
</tr>
<tr>
<td>Ten</td>
<td>Draft 6 result of ballot resolution</td>
</tr>
<tr>
<td>No meeting</td>
<td>Recirculation of changes/unresolved objections</td>
</tr>
<tr>
<td>Twelve</td>
<td>Draft 7 final discussion and input to RevCom</td>
</tr>
</tbody>
</table>

As you can see, the heaviest workload is in the first five months, when most of these meetings may need to be in person. Most of the writing and technical editing would occur during this time. However, once balloting begins, the meetings may be shorter or be mostly held electronically. IEEE ballot group formation would be done concurrently with months three and four.

There were naturally several assumptions made for this timeline, and the schedule may change based on the size of the document and the amount of contentious material. But it is an achievable model.

The most difficult task will be to find a new project that is both well enough defined and has adequate industry support to make this model work. We are exploring several possibilities, but would appreciate hearing from anyone who has a viable suggestion to prototype this process.

Report on the Information Infrastructure Standards Panel
Chuck Severance <crs@msu.edu> reports on the January 14–19, 1996 meeting in Albuquerque, NM

During the January Portable Applications Standards Committee (PASC) meeting, Kevin Lewis of Digital gave an excellent overview of the ANSI Information Infrastructure Standards Panel (IISP). The ANSI IISP is attempting to coordinate the identification and development of information technology standards that relate to, in US terminology, the National Information Infrastructure (NII).

The ANSI IISP has developed a set of 35 requirement areas that will need standardization for the NII. The area which is most applicable to POSIX is “Application Program Interface.”

After the presentation, there was some wide ranging discussion as to what the ANSI IISP should do next.

One of the tenets of the “POSIX Religion” is that a standards work will happen only if someone wants it to happen badly enough to show up and do the work. This is why the Language Independent Specifications died. You can’t really “direct” a voluntary standards organization to develop a standard. This conflicts with the ANSI IISP view that they can order standards development. As such, the POSIX group was not really very surprised that the ANSI IISP did not get much response when it went looking for standards to be developed.

There was some concern that the ANSI IISP did not truly have a “broad” perspective and was being driven by the folks who happened to show up at meetings. Another observation is that the Internet has experienced explosive growth without the ANSI IISP. The free market was providing adequate solutions, and so the IISP was not really needed.

There was a significant debate as to whether a top-down approach to standards has ever worked. MAP/TOP and OSI were cited as examples of failed top-down approaches even though a great deal of effort (and money) went into making these standards “stick.”

There was also some discussion that something as large as the NII needs someone to step in and simply make a decision. The problem is that the consensus process does not work well when there are already battle lines drawn along business and economic boundaries. The example of Lincoln setting the width of the railroad tracks by decree was given as a situation that probably could not be resolved any other way.

Another opinion was that the IISP had done good work. Their job is finished for a while, and they should wait until something happens. Some folks felt that the IISP has too optimistic an expectation in terms of time frames. Just because they have produced a “wish list” does not mean that standards will appear overnight.

Another problem that was raised is that the ANSI IISP was originally formed to solve a problem identified by the government but that a clear relationship between the IISP and the government does not seem to exist. It almost seems like
IISP created itself, appointed itself, and seems to be talking primarily to itself.

Some crazy ideas that appeared in the debate included:

- Have Congress budget some money to support standards that will have a positive impact on the NII (some positive reinforcement).
- Have the IISP focus on PR issues such as promoting standards that do apply to the NII. They could also give awards for NII innovations, develop a newsletter, sponsor NII research, and work with Congress on NII issues.

Although the discussion was wide ranging, with an equally wide ranging set of opinions, the broad overall feeling was that the IISP should figure out what its goals are and, if they have enough participation, pursue their goals. If their only goal is to have other organizations do work based on their requirements document, they should be very patient unless they have some way of bringing resources to bear on the problem.

**Report on POSIX.1k: Removable Media**

Charles DeBaun <debaun@fnal.fnal.gov> reports on the January 14-19, 1996, meeting in Albuquerque, NM

The removable media group was formed to create an optional standard for a removable media resource management command line interface. However, in the search through existing standards, it was noted that such a standard could not be implemented in a strictly compliant POSIX environment. Indeed, serial media, otherwise known as tape, cannot be supported in such an environment.

Thus, as a first step, the removable group submitted a Project Authorization Request (PAR) to provide the missing mtio semantics in POSIX.1, so that serial media, a primary type of removable media, can be supported in a POSIX environment. This PAR was approved in June 1995. A proposal is currently on the table and is being used as a working document. Draft 4 of this proposal is expected following the January 1996 meeting.

At the same time, a second PAR (POSIX.2e) was accepted to provide the mt utility definition for POSIX.2. This will provide a command line interface to the mtio API. A draft proposal is expected before the April 1996 meeting.

A third PAR is being prepared for the actual removable media resource management command line interface specification. This work is being delayed by the need to create support for serial media before it can be started. The need to backtrack to create the mtio semantics has caused a marked decrease in interest and attendance, further delaying action in this area.

Despite the low attendance at the January 1996 meeting, significant work was done to tighten up the verbiage in the P1003.1k proposal. This proposal might make a useful working paper after all.

Also on a positive note, a volunteer has come forward to fill the technical editor position.

**POSIX.1h: Services for Reliable, Available, and Serviceable Systems**

Helmut Roth <hroth@relay.nswc.navy.mil> reports on the January 14-19, 1996, meeting in Albuquerque, NM

Are you a vendor or user spending too much money and too many resources tracking down errors? Do you find yourself rewriting the same fault management or serviceability applications over and over again? Do you find yourself dealing with changing interfaces in operating systems that are breaking your applications and making your customers angry? There is hope. Vendors, users, and general interest groups can get together to reach consensus on common practices in fault management and serviceability and standardize common practices. The POSIX.1h Services for Reliable, Available, and Serviceable Systems (SRASS) working group is doing this right now and will next be meeting in Jackson Hole, Wyoming, April 15-19. The SRASS working group is in the process of developing a useful set of APIs for fault management and serviceability applications. The goal of the SRASS working group is to support fault-tolerant systems, serviceable systems, reliable systems, and available systems in a portable way. Where feasible, POSIX.1h needs to be useful for general applications, too, such as distributed, parallel, database, and transaction systems.

Right now the SRASS working group is in the process of refining the current draft of standard interfaces for logging and notification, core dump control, and configuration space management.

The logging interfaces are aimed at allowing an application to log application-specific events and system events to a system log and for the subsequent processing of those events. Fault management applications can use this API to register for the notification of events that enter the system log. Events of interest may be those that exceed some limit; a notification can have a severity associated with it, etc. A notification can provide a way to react positively and initiate steps to prevent a system failure later. Functions that are being refined are: write to the system log, open a view to the system log, read from the system log file, provide notification of entries into the system log, and search the system log.

There is a single core dump control API to enable an application to specify the location of a core dump file if a process terminates abnormally. The SRASS working group felt that
an analyst should be able to find the core dump file. In the case your system really crashes, this will be the first file you will be looking for, so it should be found easily.

Many times a corrective action such as reconfiguration is needed to keep your system alive. The configuration space management API is intended to provide a portable method of traversing the configuration space and manipulating the data content of nodes in that configuration space. This API will provide a fault management application access to underlying system configuration information and the means to direct reconfiguration of the system. In particular, the proposed set of APIs will allow a fault management application to keep track of the system configuration view and dynamically change the system’s configuration. This is achieved by means of mount and unmount operations, linking and unlinking operations, operations to add nodes to the configuration description, and several functions to allow an application to access any part of the current description of the configuration picture. Implementing and testing these APIs via ADA stubbing has been proposed by Texas Instruments to determine their applicability to an avionics real-time environment.

The realtime contingent of the SRASS working group would like to have detection as part of the SRASS API. Existing practice was presented for the IBM Phoenix Event Management, which has APIs to provide detection, but the window of opportunity to get them into the current draft before the planned mock ballot is closing rapidly. It is hoped that we will have an opportunity to look at the APIs or a proposed set to standardize on at the next meeting. It is also anticipated that these and the associated model will soon be made available for standardization.

Joint work with the realtime working group on event tracing is still taking place. So far, 35 requirements for tracing have been identified. An initial proposal for the trace APIs was expected to be ready in time for the January POSIX meeting. The proposed draft did go out over the Net but due to East Coast snow and government shutdowns, the trace subgroup did not have enough members to meet. These activities should be back on track at the next POSIX meeting. Email discussions are ongoing. Get in touch with Jim Shaffer <jls@austin.ibm.com> or Francois Riche <rich@ibm.fr> for information on trace or request to get on the trace email reflector.

Also be on the lookout for a presentation on the SRASS working group if you will be attending the Workshop on Parallel and Distributed Realtime Systems. The paper entitled "POSIX Standards for Fault Management in a Real-Time Environment" will be presented by Dr. Arun Chandra and will be published in the proceeding of the Fourth International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS) April 15–16 in Honolulu, Hawaii.

If you are interested in helping to produce standard APIs that support fault management (including serviceability and fault tolerance aspects of systems), just get in touch with me, Helmut Roth <hroth@relay.nswc.navy.mil> or Dr. Arun Chandra <achandra@vnet.ibm.com>.

Report on POSIX.1a: System Services API
J Jay Meyer <jjml@PO10.RV.unisys.com> reports on the January 14–19, 1996, meeting in Albuquerque, NM

The POSIX.1a group met in January in Albuquerque with the other Portable Applications Standards Committee (PASC) working groups. The author of the past few snitch reports has moved on to a new company and is no longer attending. Thanks for your contributions, Shravan! I’m the chair of System Services and the chair of the POSIX.1a sub-working group.

The “hardest working group in PASC” met for four days, and though now only a small group (peaking at about six people in the room), managed to make substantial progress toward getting the next draft out.

There was also a plenary meeting of the collective system services groups on Monday morning. We are not very organized in this forum yet. Minutes from the previous meeting were not distributed at all, which made them difficult to approve. We announced the passage of the POSIX.1-1990 reaffirmation ballot and discussed the fact that we need to begin the process of revising the document. ISO rules will require a revision in 1998.

We can have a Project Authorization Request (PAR) approved, but we’ll need a ballot coordinator, technical editor, and the rest of the volunteer organization to pull this off. The revision will afford an opportunity to smooth out any wrinkles between the various amendments. We should also be considering any issues that are global to the entire suite. Likely problems include considering one header per function, and doing “something” to enhance the returned error information. The “read” function does not return enough information to even be able to say that an application read to the end of file, for example. We could also provide alternate interfaces of some kind that don’t return an error status in a static variable.

The POSIX.1a group went through the long list of issues brought to the meeting by our technical editor.

One rather embarrassing uncovered problem was that we were removing common-usage C from the document, but did not get it all. Equally bad, or worse, is that this important point did not show up in our PAR. We worked out another revision to the PAR. The roots of this omission go back to Language Independent days, where we were going to have a binding to only ANSI C. By default, we’d then remove common C.
One issue still being resolved is what `flush` of a seekable input device `ought` to mean.

I think that the recent abolition of formalized ballot slots has been a positive influence on the group. There was some initial concern that the removal of deadlines would push the production of the document into the indeterminate future, but the group was much more focused and deliberate, and less “jittery” over deadline concerns.

Since I’m writing this report so long after the fact (it’s been a month already!), I have the chance to add a postscript. There have been some questions asked about the POSIX.1d Realtime Extensions and POSIX.1j Advanced Realtime Extensions documents. It turned out that POSIX.1j was being held up because of misunderstandings about ballot slots, and who had to get it sent in. This was resolved, and the document has been distributed by the IEEE with a ballot slot ending March 28. POSIX.1d is in the works, and we will make sure that its ballot does not overlap with POSIX.1j.

**New Standards Relevant to USENIX Members**

*Nick Stoughton* <nick@usenix.org> reports on a meeting held on a date he forgets in some hotel bar somewhere in the Northern Hemisphere.

A recent concept in a standards organization has been that of Authorized Project Requests (APRs), and APR-01 is now under way to examine the future of standards in human interfaces, expanding the previous Application Portability Only interfaces.

APR-01 is looking into producing a standard for that most vital of all human interfaces (at least as far as developers go) – the interface between the developer and his malt whisky.

SAGE has, of course, been working in this area for some time, and has assisted APR-01 in preparing the project review criteria for consideration at the next group meeting:

- People have been drinking Scotch for many hundreds of years. There is an amazing wealth of industry experience in the drinking of whisky from Scotland, but there are several variants in accepted behavior when imbibing.
  - Michael Jackson’s “Whisky Companion.”

- It is anticipated that a standard for the drinking of Scotch whisky can be achieved within the four year period allowed by the IEEE.

- People around the world drink whisky. A standard for acceptable practice in such consumption would fill a much needed void.

- The standard will be coordinated with any working group that attends the bar at a POSIX meeting.

- There is a clear and large commitment from the community. A significant proportion of the membership enjoys drinking single-malt whisky and will commit to endorsing any standard that is developed.

- A user interface to drinking malt whisky and aiding application developers is clearly within PASC scope.

- The Scotch BOF held at SAGE LISA is now attracting as many people as attended the entire LISA I conference in 1987. Some 40 people have attended the BOF, and these are all dedicated to developing this standard.

- No input for 1003.0 is believed necessary.

- No test plan for the standard is planned. All users of the standard who are sufficiently sober to conduct a test plan have clearly failed to read the standard.

- No steering committee is required.
Books reviewed in this column:


**The Bookworm**

*by Peter H. Salus*

<peter@uunet.uu.net>

Let me start out with an apology: in my last column, I wrote that *Programming Language Essentials* by Bal and Grune isn’t available in the US. One of the intrepid editors at Addison Wesley has informed me that not only is it available, but it's $29. That’s a bargain. If you’re interested in computer languages, get it.

**Plan 9**

In 1990, I was at the summer UKUUG Conference, where a whole bevy of guys from Murray Hill, NJ, delivered papers on “Plan 9.” I loved the papers. Last year, Rob Pike and his colleagues sent me a paper on Plan 9 for *Computing Systems*. It appeared in volume 8, number 3. And, last autumn, AT&T finally announced Plan 9 as a product. (This was a true announcement, not the sort of vapid hand waving we’ve become familiar with in the industry.) To the best of my knowledge, they’re all still in Murray Hill; God knows what part of AT&T they’re in – I wasn’t able to get a scorecard.

Recently, I was given a two-volume set labelled Plan 9 – The Manuals and Plan 9 – The Documents. I love them. Elderly folks like me recall when UNIX Programmer’s manuals were 8.5x11 in “Bell” binders. Then (with System III) we got plastic-comb-bound 6x9 manuals, a tradition that was continued in the 4.1, 4.2, and 4.3BSD manuals (which were available from your friendly USENIX Association). O’Reilly did away with the plastic combs in their 4.4BSD set. The Plan 9 volumes are 10x7, each about an inch thick. They don’t lie flat, but they are sturdy enough that they don’t start shedding pages when you put a stapler across to hold a page open. They are well typeset and legible. In The Manuals, the “BUGS” entry has been retained (unlike SVR4), as have the programmer credits (unlike SunOS or SVR4).

The entries are still in the terse and beautiful style that Dennis and Ken employed a quarter-century ago under the supervision of Doug McIlroy. I still love them, even though it takes concentration to decipher them on occasion. But then, I’ve never understood the stuff in my MS Word manual. The entries are arranged into the customary (9) sections. *time* still returns the number of seconds since the epoch: 00:00:00GMT, Jan. 1, 1970.

Volume 2, The Documents, is a gem. I suggest that anyone interested in operating systems read it from cover to cover. First of all, the material is interesting; secondly, the material was written largely by folks who try to write English and succeed at it. There are 30 papers in all, ranging from a version of the Plan 9 article that appeared in *Computing Systems* to the “Troff User’s Manual,” which bears the epigraph: “the old warhorse, updated for Unicode characters” (Pike and Thompson’s article on Unicode characters from the San Diego USENIX Conference [1993] is there, too). Other contributions are by such otherwise invisible characters as Duff, Flandrena, Glick, Holzmann, Hume, Presotto, Trickey, and Winterbottom. Collect ‘em all!

Even if you’re not running Plan 9, you’ll learn from the docs.

**More good stuff**

Plauger and Brodie have turned out yet another useful volume. In barely over 200 pages, they introduce the *Standard C Language* (= ANSI X3.159-1989 or...
ISO/IEC 9899:1990). There’s also a floppy with everything in hypertext on it. Since we’re basically talking UNIX, there’s a Chapter 0 (Introduction). The remainder of the book is made up of two parts (“The Standard C Language,” chapters 1–7, and “The Standard C Library,” chapters 8–26) and three appendices. Within each library chapter, the various macros are listed in alphabetical order. For example, Chapter 12 float.h begins with DBL_DIG and ends with LDBL_MIN_EXP.

I’m sure you all know that the Software Tools User Group received an award from USENIX this past January. Because Bill Plauger is half of the duo that brought us Software Tools in 1976, I’d like personally to thank him and Brian Kernighan for the insight into programming they gave me nearly 20 years ago.

And Some Trash

Over the past few months, my shelves have overflowed with books on the software process. These include volumes on software requirements and specifications, ISO 9000, software testing in the real world, a quantitative approach to software management, guidelines for improving the software process, and managing a programming project. When these books arrive, I immediately turn to the indices and references. I look for Fred Brooks, Wayne Babich, Watts Humphrey, and David Tilbrook. Several of the books have mentioned Brooks and/or Humphrey. None has mentioned the new edition of Brooks, Babich, or Tilbrook. The numerous papers of David Tilbrook point at the solution to version drift and inconsistent compiling, and I’d have hoped that someone might have read them.

The books are largely useless. Buy Brooks, Babich, or Humphrey, and look up Tilbrook in EUUG and USENIX proceedings.

Richard III

I admit that I’ve never heard Rich Stevens cry out for a horse (though I understand he has had run-ins with bovines), but his TCP/IP Illustrated has now hit a third volume. He must be getting tired, for although II was 1,174 pages (whew!), this one’s a mere 328. And it’s about a lot of stuff I just don’t know much about: TCP for transactions, HTTP, NNTP, and the UNIX Domain Protocols. One of the neat things about this is that all the source code Stevens cites is from the BSD-Lite release. It is thus accessible to and comprehensible by mere mortals. My guess is that he’ll now turn around and redo TCP/IP I in the light of IPv6. I’m looking forward to it.

Kernel Secrets

Jolitz and Jolitz have produced a massive volume of “source code secrets.” Parts of the book are truly interesting, but it is hard to see these as “secrets” in any way. The cover tells me that the volume is The Basic Kernel Source Code Secrets, but it should be called something like 386BSD Kernel Source Code. It is, indeed, about 386BSD, something with which the Jolitzes have been involved for some years. In fact, parts of the book at hand made up 17 articles in Dr. Dobbs Journal five years ago.

Large sections of the volume are in catechistic format: “What are processor exceptions?” “How is cansignal() implemented?” “What is groupmember()?” I guess this is helpful to some folks, I found it irritating. I also found the lack of references or credit to others irksome. Keith Bostic’s work on dynamic makefiles is credited – but this is on page 511. Earlier on (page 219), I noted a reference to Knuth. But Dennis, Ken, and their cohorts in New Jersey are gone; the entire CSRG has vanished. I know that there is bad blood, but pretending that although 386BSD was derived from BSD Networking Release 2, Net-2 had no creators or ancestors is absurd. The Jolitzes promise a second volume; perhaps references there will fill the current gap.

Advertising

Do you have an eighth-generation copy of John Lions’s v6 code and commentary? Wanna trade it in for a hardbound, legal copy? After over two years of struggling with lawyers and with the assistance of Dennis Ritchie, Mike Tilson (ex-board member of USENIX and vice-president of SCO, the current owners of UNIX) has given permission for the orange and red books to appear. Dennis has written a Foreword and Mike O’Dell, Peter Reintjes, and several other luminaries have written reminiscences and appreciations. It should be out during the summer. I’ll keep you posted.
TCP/IP Illustrated Volume I
(The Protocols)

by W. Richard Stevens, Addison Wesley, 1994,

TCP/IP Illustrated Volume II
(The Implementation)


Reviewed by George V. Neville-Neil
<gnn@netcom.com>

When Douglas Comer wrote his book on the technologies that would come to be called "the Internet" there were very few, if any, other books like it. In 1994 and 1995, when these books were written, the books by Comer (as well as several others) had already been published. What was needed was a complete reference to the Internet Protocols that would bridge the gap between the different多年 and a full textbook on networking. This is what these books provide.

Coming up with a reference on a collection of technologies that changes from day to day is a tall order. It would be very difficult to write a book that covered just the currently accepted protocol stack, as defined in the Request for Comments (RFCs) documents. To cover emerging protocols would have required much more than 1,650 pages. Having carried these books to a local cafe to write this review, I can say that I'm glad they are as concise as they are.

I have had these books now for over six months, and this review was long in coming due to the nature of the books. Unlike some other books that I have reviewed, these are not suitable for a straight-through read. These are reference books and deserve to be used as such. My review consisted of putting the books on my desk as a reference, which means they took their place in a small collection of five or six books that actually sit on my desk next to my workstation. Every time I had a question about how something was working, or why it wasn't, I opened one of the books to the index to try to find out what it had to say about that particular problem or question. Thus far, they have not failed me.

The final pages of the book wrap up with Chapter 30 (Other TCP/IP Applications), six appendices, an extensive bibliography, and the index.

Volume II is a source read though of most of the networking code from the Net/3 release of BSD. I happen to have a 4.4BSD Lite CDROM, so I can always look at it in a window while referring to this volume. The source isn't strictly necessary because all its salient portions are reproduced right in the book. The book's source code is also somewhat less confusing for a first read because a lot of the unnecessary stuff is elided.

This book is structured pretty much in the same way as volume I. After an introduction that tells how the book is laid out and other background info, we get right to the heart of the BSD networking code, Mbufs. Chapter 2 is dedicated to Memory Buffers (Mbufs) used to implement all of the networking code in BSD UNIX kernel.
From the memory system, we move up to the Interface Layer in Chapter 3. Three interfaces are then presented: Ethernet in Chapter 4, then SLIP and Loopback in Chapter 5.

The next several chapters deal with IP-level issues such as IP Addressing (Chapter 6), the Internet Protocol itself (Chapter 8), Fragmentation and Reassembly (Chapter 10), and Multicasting (Chapter 12).

Three entire chapters are dedicated to the socket interface. Because all user/network activity takes place through the socket calls, this makes sense. The next three chapters deal with routing issues. There are several books on just the issue of routing in an internetwork, but these chapters serve as an excellent and practical introduction to the subject.

After three short chapters (one on the Address Resolution Protocol, one on Protocol Control Blocks, and one on the User Datagram Protocol), we begin an in-depth study of the Transmission Control Protocol (TCP). Seven chapters, totaling 231 pages, cover the TCP layer. Because TCP is the most complex protocol built using IP, this level of depth is necessary to give the subject its proper treatment.

The final two chapters of the book cover the BSD Packet Filter (Chapter 31), and Raw IP (Chapter 32). There are three appendices, an extensive bibliography, and an excellent index.

Perhaps the best test of these books is their indices. Because they really will be a pure reference for most people, the index is the key to the book. I found that the index always led me where I wanted to go, given a key word to look up, on the first try. In short, these books are a must have for anyone implementing or debugging an IP protocol stack.
ANNOUNCEMENTS & CALLS

Second Conference on Object-Oriented Technologies and Systems (COOTS)

June 17-21, 1996
Marriott Eaton Centre – Toronto, Canada

The refereed papers for this conference are still being selected as we go to press, but we're providing early notice of the tutorial slate.

Dave Thomas of Object Technology International will give the keynote address, and there will be an Advanced Topics Workshop.

Tutorial Program – June 17-18, 1996

Java: A Language for Providing Content on the World Wide Web
Jim Waldo, Sun Microsystems Labs and JavaSoft

New ANSI C++ Features
José Lajoie, IBM Canada Laboratory

Introduction to CORBA and CORBA Services
Bruce Martin, SunSoft, Inc.

STL In Action
Graham Glass, ObjectSpace, Inc.

Programming Distributed Components Using Network OLE and C++
Don Box, DevelopMentor

Introduction to the Python Programming Language
Jim Fulton, Consultant

OO Design Patterns for Concurrent, Parallel, and Distributed Systems
Douglas C. Schmidt, Washington University, Missouri

Building Distributed Applications With CORBA and C++
Steve Vinoski, Hewlett-Packard

Pattern-Oriented Software Architecture
Hans Rohnert, Siemens AG

Inter-Domain Management: CORBA, OSI, SNMP
Subrata Mazumdar, Bell Laboratories, Lucent Technologies

Java Applets and the AWT
Nataraj Nagaratnam, Syracuse University

Advanced Modeling and Design for Java Systems
Desmond F. D’ Souza and Petter Graff, Icon Computing, Inc.

Advanced Topics Workshop

This year’s USENIX COOTS conference will conclude with an Advanced Topics Workshop. The goal of this workshop is to provide an informal setting in which to exchange in-depth technical information with your peers. This workshop will be open to authors of papers in the conference, as well as participants who submit position papers related to the workshop’s topic. This topic will be determined several months before the conference and a Call for Position papers will be announced. Past USENIX C++ conferences have held Advanced Topics Workshops on a variety of topics including distributed object computing and implementation issues related to C++ language features.

Full Program and Registration Information

Look for the full program on comp.org.usenix. and our WWW site, http://www.usenix.org. The registration brochure will be mailed in early April. If you have any questions, send email to conference@usenix.org or call the Conference Office at 714 588 8649.
Second USENIX Workshop on Electronic Commerce

November 18-20, 1996
Claremont Hotel & Resort, Oakland, CA

Sponsored by the USENIX Association

Important Dates
Extended abstracts due: July 16, 1996
Notification to authors: August 5, 1996
Camera-ready final papers due: October 7, 1996

Preliminary Program Committee
Ross Anderson, Cambridge University
Nathaniel Borenstein, First Virtual Holdings, Inc.
Stefan Brands, CWI
Dan Geer, Open Market, Inc.
Mark Manasse, DEC Systems Research Center
Clifford Neuman, University of Southern California
Doug Tygar, Carnegie Mellon University, Program Chair
Hal Varian, University of California, Berkeley
Bennet Yee, University of California, San Diego
Others to be determined

Overview
The Second USENIX Workshop on Electronic Commerce will provide a major opportunity for researchers, experimenters, and practitioners in this rapidly self-defining field to exchange ideas and present results of their work. This meeting will set the technical agenda for work in the area of Electronic Commerce by examining urgent questions, discovering directions in which answers might be pursued, and revealing cross-connections that otherwise might go unnoticed.

Tutorials
The Workshop will begin with a day of tutorials. The tutorial program will offer a selection of tutorials from among several tracks on such topics as cryptography and security.

Workshop Topics
Two days of technical sessions will follow the tutorials. Submissions are welcome for technical and position paper presentations, reports of work-in-progress, technology debates, and identification of new open problems. Birds-of-a-Feather sessions in the evenings and a keynote speaker will round out the program.

We seek papers that will address a wide range of issues and ongoing developments, including, but not limited to:

- Advertising
- Anonymous transactions
- Audibility
- Business issues
- Copy protection
- Credit/Debit/Cash models
- Cryptographic security
- Customer service
- Digital money
- E-mail enabled business
- EDI
- Electronic libraries
- Electronic wallets
- Exception handling
- Hardware-enabled commerce
- Identity verification
- Internet/WWW integration
- Key management
- Legal and policy issues
- Micro-transactions
- Negotiations
- Privacy
- Proposed systems
- Protocols
- Reliability
- Reports on existing systems
- Rights management
- Service guarantees
- Services vs digital goods
- Settlement
- Smart-cards
Questions regarding a topic's relevance to the workshop may be addressed to the program chair via electronic mail to tygar@cs.cmu.edu. Proceedings of the workshop will be published by USENIX and will be provided free to technical session attendees; additional copies will be available for purchase from USENIX.

What To Submit

Technical paper submissions and proposals for panels must be received by July 16, 1996. We welcome submissions of the following type:

Refereed Papers—Full papers or extended abstracts should be 5 to 20 pages, not counting references and figures.

Panel proposals—Proposals should be 3 to 7 pages, together with a list of names of potential panelists. If accepted, the proposer must secure the participation of panelists, and the proposer will be asked to prepare a 3 to 7 page summary of panel issues for inclusion in the Proceedings. This summary can include position statements by panel participants.

Please accompany each submission by a cover letter stating the paper title and authors along with the name of the person who will act as the contact to the program committee. Please include a surface mail address, daytime and evening phone number, and, if available, an email address and fax number for the contact person. If all of the authors are students, please indicate that in the cover letter for award consideration (see “Awards” below).

USENIX workshops, like most conferences and journals, require that papers not be submitted simultaneously to more than one conference or publication and that submitted papers not be previously or subsequently published elsewhere. Submissions accompanied by “non-disclosure agreement” forms are not acceptable and will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

Awards

The program committee will offer awards of $500 for the best paper and the best student paper.

Where To Submit

Please send submissions to the program committee via one of the following methods. All submissions will be acknowledged.

Preferred Method: email (Postscript)

Authors should ensure that their papers will print on a broad range of postscript printers and submit in sufficient time to allow us to contact the author about alternative delivery mechanisms in the event of network failure.

Alternate Method: 10 copies, via postal delivery to

Doug Tygar (program chair)
Computer Science Dept, CMU
5000 Forbes Ave
Pittsburgh, PA 15213-3891
Email: tygar@cs.cmu.edu
Fax: 412.268.5576

If you have questions on the format of submissions or about the workshop, please telephone the USENIX Association office at 510.528.8649, or email to ecauthors@usenix.org or the program chair tygar@cs.cmu.edu.

An electronic version of this Call for Papers is available at WWW URL:

http://www.usenix.org

Registration Information

Materials containing all details of the technical and tutorial programs, registration fees and forms and hotel information will be available in September 1996. If you wish to receive the registration materials, please contact USENIX at:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA 92630
Phone: 714.588.8649
Fax: 714.588.9706
Email: conference@usenix.org
URL: http://www.usenix.org

Or you can send email to our mailserver at info@usenix.org. Your message should contain the line: send catalog. A catalog will be returned to you.
Announcement and Preliminary Call for Papers

2nd Symposium on Operating Systems Design and Implementation (OSDI'96)

October 28–October 31, 1996
Seattle, Washington, USA

Sponsored by the USENIX Association
Co-sponsored by ACM SIGOPS and IEEE TCOS

After a successful first OSDI symposium, the next OSDI will continue to focus on practical issues related to modern operating systems. OSDI brings together professionals from academic and industrial backgrounds, and has become the perfect forum for issues concerning the design and implementation of operating systems for modern computing platforms such as workstations, parallel architectures, mobile computers, and high speed networks.

The OSDI symposium emphasizes both innovative research and quantified experience in operating systems. We seek papers describing original work concerning the design, implementation, and use of modern operating systems. Besides mature work, we encourage submissions describing exceptionally promising, well-grounded speculative work, or enlightening negative results. Topics of interest include, but are not limited to:

- OS structure and organization
- OS kernel internals, servers and applications
- Distributed and mobile computing
- Multiprocessor and parallel systems
- Communications
- Storage Management and I/O systems
- Security in distributed systems
- Scalability and availability
- Heterogeneous systems
- Performance and optimizations
- Language support for OS
- OS interaction with HW architecture
- OS support for real time and multimedia
- Interaction of OS and applications

Symposium Overview

The symposium will consist of one day of tutorials, followed by 2.5 days of single-track technical sessions with presentations of the refereed papers, and a half day workshop on a topic yet to be determined. One of the technical sessions will be dedicated to work-in-progress presentations and will be described in later announcements. The refereed papers will be published in the Proceedings, provided free to technical session attendees and available for purchase from USENIX. The Proceedings may also be distributed to ACM SIGOPS members. Papers of particular merit will be selected to receive an award and will be published in the IEEE TCOS Bulletin.

Program Committee

Karin Petersen, Xerox PARC (co-chair)
Willy Zwaenepoel, Rice University (co-chair)
Peter Chen, University of Michigan
Richard Draves, Microsoft Research
Carla Ellis, Duke University
Ed Felten, Princeton University
Jim Gray, Microsoft Bay Area Laboratory
Kevin Jeffay, University of North Carolina
David Johnson, Carnegie Mellon University
Jay Lepreau, University of Utah
Jeff Mogul, DEC WRL
Marc Shapiro, INRIA
John Wilkes, Hewlett-Packard Labs
John Zahorjan, University of Washington
Important Dates
Full papers due: May 7, 1996
Notification to authors: July 30, 1996
Revised papers due for shepherding: August 19, 1996
Camera-ready full papers due: September 16, 1996

Submission Process
Authors are required to submit full papers by May 7, 1996. Submitted papers should be no longer than 14 pages, spaced no closer than standard 10 point font on 12 point baseline, single- or double-column format. Longer submissions will be discarded without review. Very similar papers must not have been published or submitted for publication elsewhere. All submissions will be held in the highest confidentiality prior to publication. Papers accompanied by so-called "non-disclosure agreement" forms are not acceptable and will be returned unread.

The papers will be judged on significance, originality, clarity, relevance, and correctness. The committee will favor papers with reproducible results, especially those supplying detailed data and explanations, or offering to make data sets or source code available.

Accepted papers will be shepherded through an editorial review process by a member of the program committee.

Authors of accepted papers will be expected to provide an HTML page containing the abstract and links to their paper, slides, and software, if available. This will be collected after the event for inclusion in an electronic version of the symposium (for an example, see http://www.cs.utah.edu/~lepreau/osdi94/).

Where to submit
Submission of all papers must be made in both paper and electronic form. Fifteen (15) paper copies (double sided if possible) of the paper must be sent to:

Willy Zwaenepoel
Department of Computer Science,
Rice University
6100 S. Main St.
Houston, TX 77005, USA

and one electronic copy in Postscript (not ASCII) must be submitted by electronic mail to: osdi-papers@cs.rice.edu

For administrative reasons (not blind reviewing), every submission (in both its paper and electronic form) should include one additional page containing: (i) paper title and authors, indicating any who are full time students, and (ii) for the author who will act as the contact to the program committee, his or her name, paper mail address, daytime and evening phone numbers, email address and fax number, if available. The cover sheet mailed with the electronic paper submission should be in ASCII to facilitate accurate on-line bookkeeping, and should be included in the same electronic mail message as the PostScript file containing the paper.

For more details on the submission process authors are encouraged to consult http://sandbox.xerox.com/osdi-96.

All submissions will be acknowledged by May 21, 1996. If your submission is not acknowledged by this date, please contact the program chairs promptly at osdi@cs.rice.edu.

Cash Prizes
Cash prizes will be awarded for the best paper at the conference and the best student paper.

Registration Materials
Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be mailed in August 1996. If you wish to receive the registration materials, please contact:

USENIX Conference Office
22672 Lambert St., Suite 613
Lake Forest, CA USA 92630
Phone: 714 588 8649
Fax: 714 588 9706
Email: conference@usenix.org
WWW URL: http://www.usenix.org.
Announcement and Call for Participation

10th Systems Administration Conference (LISA '96)

September 30–October 4, 1996
Chicago Marriott, Chicago, Illinois

Co-sponsored by USENIX, the Advanced Computing Systems Professional and Technical Association and SAGE, the System Administrators Guild

Important Dates
Referred paper submissions:
Extended abstracts due: May 7, 1996
Notification to authors by: June 11, 1996
Final papers due: August 15, 1996
Registration materials available: July, 1996

Overview
LISA, the USENIX Systems Administration Conference, is the leading conference for and by system administrators. LISA originally stood for "Large Installation Systems Administration" when a large installation meant over 100 users, 100 systems, or a gigabyte of disk storage. Today, the LISA conference offers the most comprehensive program for system administrators from sites of all sizes and at all levels of experience.

LISA '96 will mark the tenth anniversary of the LISA conference. While there will be events at the conference to mark this occasion, the focus will continue to bring system administrators the latest tools, techniques, and information needed to keep pace with the rapid technology advancements, changes in public and legal policy, and changes in the ways that their employers do business.

Tutorial Program
Monday and Tuesday, September 30–October 1, 1996
The conference will offer up to 20 tutorials on two days. Tutorials are offered on all levels of system administration from novice to senior system administrator.

Technical Sessions
Wednesday through Friday, October 2–4, 1996
The three days of technical sessions consist of two parallel tracks. The first track is dedicated to presentations of referred technical papers. The second track is intended to accommodate invited talks, panels and Works-in-Progress (WIP) sessions.

Conference Topics
Papers addressing the following topics are particularly timely; papers addressing other technical areas of general interest are equally welcome.
- Innovative system administration tools and techniques
- Integrating new networking technologies
- Problem tracking
- Remote site administration
- Experiences supporting large sites (>1000 users or machines)
- Experiences supporting nomadic and wireless computing
- Integration of heterogeneous platforms—multiple UNIX platforms, PC/Mac integration, interfacing with legacy systems

Refereed Paper Submissions
An extended abstract is required for the paper selection process. Full papers are not acceptable at this stage; if you send a full paper, you must also include an extended abstract. "Extended" means 2–5 pages.

Include references to establish that you are familiar with related work, and, where possible, provide detailed performance data to establish that you have a working implementation or measurement tool.

Submissions will be judged on the quality of the written submission, and whether or not the work advances the state of the art of system administration. For more detailed author instructions and a sample extended abstract, send email to lisal0author@usenix.org or call USENIX at 510.528.8649.

Note that LISA, like most conferences and journals, requires that papers not be submitted simultaneously to more than one conference or publication and that submitted papers not be previously or subsequently published elsewhere. Papers accompanied by...
Invited Talks

If you have a topic of general interest to system administrators, but that is not suited for a traditional technical paper submission, please submit a proposal for a second track presentation to the invited talk (IT) coordinators at itlisa@usenix.org.

Workshop: Advanced Topics in System Administration

Tuesday, Oct 1, 1996

A one-day, pre-LISA conference workshop organized by John Schimmel of Silicon Graphics will focus on a discussion of the latest-breaking technical issues in the systems administration arena as introduced by those in attendance. Attendance is limited and based on acceptance of a position paper. A representative subset of positions will be discussed in an open forum.

How to Submit: Potential workshop attendees are invited to submit a proposal of at most 3 pages (ASCII) via electronic mail to fei@sgi.com no later than August 1. (More substantive reports of completed works should instead be submitted as papers to the technical sessions.) These proposals should briefly contain a topic for discussion, a description of the subject, an explanation of what makes this topic controversial or interesting, and a personal position. Acceptance notices to all participants will be issued by September 9, 1996.

Participants must be pre-registered for the LISA conference. No additional fee will be charged to attend this workshop, and lunch will be provided.

Program Committee

Program Co-chairs:
Helen E. Harrison, SAS Institute Inc.
Amy K. Kreiling, University of North Carolina

Program Committee:
Paul Evans, Synopsys, Inc.
David L. Kensing, MCI Telecommunications
Bill LeFebvre, Argonne National Labs
E. Scott Menter, Enterprise Systems Management
Pat Parseghian, AT&T Bell Laboratories
Pat Wilson, Dartmouth College
Elizabeth Zwicky, Silicon Graphics, Inc.

Invited Talks Co-ordinators:
Rik Farrow, Internet Security Consulting
Kimberly Trudel, Massachusetts Institute of Technology

Vendor Displays

Wednesday and Thursday, October 2–3, 1996

LISA attendees have an enormous interest in industrial strength, state of the art solutions to system administration problems. If your company’s products provide solutions, LISA will provide attendees with the technical expertise to understand and appreciate it. Please contact:

Zanna Knight
Tel: 510.528.8649
Fax: 510.548.5738
Email: display@usenix.org

Birds-Of-A-Feather Sessions

Birds-of-a-Feather sessions (BoFs) are very informal gatherings of attendees interested in a particular topic. BoFs are held Tuesday, Wednesday, and Thursday evenings of the conference. BoFs may be scheduled in advance by telephoning the USENIX Conference Office at 714.588.8649 or via email to conference@usenix.org. They may also be scheduled at the conference.

For Registration Information

The complete program and registration information will be available in July 1996. If you would like to receive registration materials, please contact:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA 92630
Phone: 714.588.8649
Fax: 714.588.9706.
Email: conference@usenix.org
URL: http://www.usenix.org

Or you can send email to our mailserver at info@usenix.org. Your message should contain the line: send catalog. A catalog will be returned to you.
Fifth International Workshop on Object-Orientation in Operating Systems: IWOOOS '96

October 27-28, 1996 – Seattle, WA

Sponsored by the IEEE Technical Committee on Operating Systems and Application Environments (TCOS) (pending) and USENIX Association

The fifth International Workshop on Object-Orientation in Operating Systems will bring together researchers and practitioners who are interested in object-oriented approaches to operating systems design, development, and application support. The purpose of the workshop is to provide an informal format and atmosphere in which ideas and current work can be presented and discussed at length. The workshop is designed to encourage the full participation of each attendee: both presenters and participants will be active contributors throughout the workshop.

This year's workshop will be held in Seattle, Washington, just prior to the Second Symposium on Operating Systems Design and Implementation which will be held in Seattle, Washington from October 28-31. We hope that the conjunction of the two events will foster cross-fertilization between related research communities.

The focus of the workshop is on how to effectively use objects inside operating systems and how to provide system support for object-oriented applications in a variety of application domains.

Subjects of particular interest include:

- Using objects to make operating systems customizable, extensible and adaptable
- Design patterns in operating systems
- Objects on WWW and their OS support
- Persistent objects and their OS support
- Mobile Objects and their OS support

The workshop is structured to encourage the submission of explorative work in the form of position papers. Position papers should be a maximum of 2500 words and should present initial work, new ideas, or a strong position statement.

Attendance will be by invitation only. To be invited, an attendee must submit a position paper. All submissions will be reviewed. All accepted papers will be published in a proceedings. The official language for the conference will be English.

Organizing Committee

Workshop Chair: Andrew Black, Oregon Graduate Institute
Program Chair: Nayeem Islam, IBM T. J. Watson Research Center
Local Arrangement Co-Chairs: Michael Jone, Microsoft and Crispin Cowan, Oregon Graduate Institute
Publicity Chair: Douglas Schmidt, Washington University, St. Louis
Publication Chair: Luis-Felipe Cabrera, IBM
Finance Chair: David Cohn, University of Notre Dame

Program Committee

Mustaque Ahamad, Georgia Institute of Technology
Henri Bal, Vrije University
Gary Lindstrom, University of Utah
Eric Manning, University of Victoria
Satoshi Matsuoka, University of Tokyo
Gregor Kiczales, Xerox Parc
Sacha Krakowiak, IMAG, France
Jim Purtilo, University of Maryland
John Rosenberg, University of Sydney
Margo Seltzer, Harvard University
Santosh Shrivastava, University of Newcastle-upon-Tyne
Mario Tokoro, Keio University

Important Dates

Position Papers: July 1, 1996
Invitations issued: July 30, 1996
Camera-ready copy due: September 1, 1996

Submission Instructions

Each submission should have a principal author, to whom all messages will be sent; please provide email and postal addresses as well as telephone and fax numbers. A notice will be sent to the principal author upon receipt of every paper.

Electronic submission via email is strongly encouraged. Please send your paper to the program chair at nayeem@watson.ibm.com

Submissions are required to be in HTML, ASCII, or PostScript (uuencoded). Electronic submissions will be ACK’ed within a day or so of receipt. If the submission could not be successfully printed out on paper then the program chair will attempt to confer with the sender via email about what to do as an alternative submission means. Note: if you do not receive any acknowledgment message at all within a period of several days then the submission message may have gotten lost in transit and you should send a short email message to the program chair to alert him to the difficulty.
Announcement and Call for Participation

USENIX Annual Technical Conference
Pre-Announcement and Call for Papers and Presenters

January 6-10, 1997
Anaheim Marriott Hotel
Anaheim, California

Program Committee
John T. Kohl, Program Chair, Ariad Software
Marc Blaze, AT&T Research
Bill Bolosky, Microsoft Research
Nahaniel Borenstein, First Virtual Holdings
Charlie Briggs, Digital Equipment Corporation
Clem Cole, Lotus Computing
Fred Douglass, AT&T Research
Rob Gingell, Berkeley Software Design
Dave Schaefer, Harvard University
John Schimmel, Silicon Graphics
Carl Staelin, Hewlett-Packard Laboratories

Invited Talks Co-ordinators
Mary Baker, Stanford University
Berry Ketcheval, Xerox PARC

Due Dates for Refereed Paper Submissions
Manuscripts Due: June 18, 1996
Notification to Authors: August 7, 1996
Camera-ready Final Papers Due: November 13, 1996

Conference Schedule Overview
Tutorials: January 6-7, 1997
Technical Sessions and Invited Talks: January 8-10, 1997
Birds-of-a-Father Sessions: January 7-9, 1997
Vendor Display: January 8-9, 1997
USENIX Reception: January 8, 1997

Conference Overview
The conference technical sessions include one track of refereed papers selected by the Program Committee. The refereed papers are published in the Conference Proceedings which are provided to all registered technical session attendees. There is also a parallel track of Invited Talks. These survey-style sessions given by experts range over a variety of interesting and timely topics. Submitted Notes from the Invited Talks are published and distributed to registered technical session attendees.

Two full days of tutorials precede the technical sessions with practical tutorials on timely topics.

Other highlights of the conference include a work-in-progress session, which provides a forum for short informal technical presentations; the evening birds-of-a-feather sessions which provide very informal gatherings on particular topics; and a Guru is IN sessions, informal discussions where noted experts from the USENIX community will answer technical questions; vendor exhibits, which provide the opportunity for no-nonsense evaluation of products and services.

Refereed Papers
The emphasis for this year’s USENIX Technical Conference is on advanced systems’ uses in the global computing environment. How do we build computing systems which fulfill current needs yet grow to handle the future demands? What techniques and technologies can we use to satisfy a large, growing, and changing computing appetite? How do we support new computing styles with advanced computing systems? How do we protect the systems we build from failures or abuses?

The Program Committee seeks original and innovative full papers about the applications, architecture, implementation, and performance of modern computing systems. Some particularly interesting related topics are:

- Scaling the advanced system: down to laptops, palmtops, embedded systems; up to large file systems and memories, mass storage, faster networks, new protocols
- Mobile systems: network connectivity, system support, application design
- Tasks/roles where advanced systems shine or fall short
- Practical network security, privacy, and cryptography
- Electronic commerce, internetworking
- Multi-media challenges, solutions, and innovations
- Interoperability/standards: tools, techniques, and experience connecting with other computing systems

This list is not exhaustive; you are encouraged to submit papers on other advanced system related topics. As at all USENIX conferences, papers that analyze problem areas and draw important conclusions from practical experience are especially welcome.

How to Submit a Refereed Paper
It is imperative that you contact the USENIX Association office to receive detailed guidelines and suggestions for submitting a quality paper to the refereed track of the technical sessions. Please send email to

USENIX®, the Advanced Computing Systems Professional and Technical Association.
If you have a MIME-capable mail system, you are encouraged to include your PostScript manuscript as Content-Type: application/postscript, Content-Transfer-Encoding: base64. Important: For PostScript submissions, use only standard PostScript fonts, and format your paper for US Letter (8.5 x 11 inches) paper. If your paper will not print properly, your submission will be returned. You should attach the cover letter (see below) as a separate MIME enclosure.

Alternate method: Postal Delivery to:
John Kohl
Atria Software
20 Maguire Road
Lexington, MA USA 02173
Phone: 617 . 676 . 2641

The authors must also submit the following information (for administrative handling) via email to wcnix97paper@usenix.org
1. The title of the manuscript and the names of the authors. (Note: the program committee does not review papers blindly; the authors' names and affiliations will be known to the reviewers).
2. The name of one author who will serve as a contact, an email address, day and evening phone numbers, postal mail address, and fax.
3. An indication of which, if any, of the listed authors are full-time students.
4. A short abstract of the paper (100-200 words) (this can be the same as the paper's abstract).

Cash Prizes
Cash prizes will be awarded for the best paper at the conference and the best student paper.

Invited Talks
An Invited Talks track complements the Refereed Paper track. These talks by invited experts provide introductory and advanced information about a variety of topics such as using standard UNIX tools, tackling system administration difficulties, or employing specialized applications. Submitted Notes from the Invited Talks are published and distributed free to technical sessions attendees. This track also includes panel presentations and selections from the best presentations offered at 1996 USENIX conferences and symposia.

Suggestions/Proposals Wanted
The Invited Talks coordinators welcome suggestions for topics and request proposals for particular talks. In your proposal, state the main focus, include a brief outline, and be sure to emphasize why your topic is of general interest to our community. Please submit via email to wcnix97paper@usenix.org.

Tutorials
On Monday and Tuesday, you may attend intensive, immediately practical tutorials on topics essential to the use, development, and administration of UNIX and UNIX-like operating systems, networking systems, networks, advanced programming languages and related technologies. The USENIX Association's well-respected program offers introductory and advanced tutorials, presented by skilled instructors who are hands-on experts in their topic areas. USENIX will offer two full days of tutorials covering topics such as:
- System and network administration
- System and network security
- Java
- Distributed computing
- Kernel internals: SVR4, BSD, Windows NT
- Systems programming tools and program development
- Portability and interoperability
- Client-server application design and development
- Sendmail, DNS, and other networking issues
- GUI technologies and builders
- WWW technologies

Proposals Wanted
To provide the best possible tutorial slate, USENIX constantly solicits proposals for new tutorials. If you are interested in presenting a tutorial, contact the Tutorial Coordinator:
Daniel V. Klein
Phone: 412 . 421 . 2332
Email: dsk@usenix.org

Work-in-Progress Reports (Whips)
Bob Gray, U S WEST Advanced Technologies
WIPS Co-ordinator
Do you have interesting work you would like to share, or a cool idea that is not yet ready to be published? The Work-in-Progress reports, scheduled during the technical sessions, introduce interesting new or ongoing work. The USENIX audience provides valuable discussion and feedback. We are particularly interested in presenting student work. To schedule your report, send email to wips97@usenix.org.

Birds-of-a-Feather Sessions (BOFs)
The popular evening Birds-of-a-Feather sessions are very informal attendee-organized gatherings of persons interested in a particular topic. BOFs often feature presentations or demonstrations followed by discussion, announcements, and the sharing of strategies. BOFs are offered Tuesday, Wednesday, and Thursday evenings of the conference. They may be scheduled on-site at the conference or in advance by contacting the USENIX Conference Office by phone at 714 . 588 . 8649 or via email to conference@usenix.org.

Vendor Exhibits
Vendors will demonstrate the technical innovations which distinguish their products. In this relaxed environment, attendees can discuss the product features and services on display.

Vendors: This is an exceptional opportunity to receive feedback from our technically astute attendees. If your company would like to display its products and services, please contact:
Zanna Knight
USENIX Association
Telephone: 510 . 528 . 8649
Fax 510 . 548 . 5738
Email: display@usenix.org

Conference Program and Registration Information
Special Hotel Rates
The Anaheim Marriott Hotel, adjacent to Disneyland, is headquarters for the USENIX 1997 Technical Conference and will be the location for all conference activities. The Anaheim Marriott will be offering special room rates to attendees.

Registration Materials
Materials containing all details of the technical sessions, tutorial program, conference registration, hotel and airfare discounts, and reservation information will be available mid-September, 1996.

If you wish to receive the registration materials, please contact:
USENIX Conference Office
22672 Lambert St., Suite 613
Lake Forest, CA USA 92630
Phone: 714 . 588 . 8649
Fax: 714 . 588 . 9706
Email: conference@usenix.org

About The USENIX Association
Since 1975, the USENIX Association has provided a forum where the community of engineers, scientists, and technicians working on the cutting edge of the computing world come together to communicate the results of innovation and research in UNIX and modern open systems. USENIX is well known for its technical conferences, tutorial programs, and the wide variety of publications it has sponsored over the years. USENIX is the original, not-for-profit membership organization for individuals and institutions interested in UNIX and related technologies. Evolving with technology, USENIX has broadened its activities to include open systems and the globally interconnected and interoperable computing environment.

The USENIX Association and its members are dedicated to:
- problem-solving with a practical bias,
- fostering innovation and research that works,
- rapidly communicating the results of both research and innovation, and
- providing a neutral forum for the exercise of critical thought and the airing of technical issues.

SAGE, the System Administrators Guild, a Special Technical Group within the USENIX Association, is dedicated to the recognition and advancement of system administration as a profession.

For more information about USENIX and SAGE, our publications, or events, please visit our Web site. The URL is http://www.usenix.org. Or, send email to our mailserver at info@usenix.org. Your message should contain the line: send catalog.

A catalog will be returned to you.
"Such is the life of a teenager with the root password." —Ryan Tucker, loyal O'Reilly reader

O'REILLY READER PROFILE:
Ryan Tucker  AGE: 15
RESIDENCE: Des Moines, Iowa
FAVORITE TV SHOW: Star Trek ("the true sign of a geek!")
WORDS HE USES TO DESCRIBE HIMSELF: "Youthful Evil Genius"

Imagine, if you can, what being both a sysadmin and a 15-year-old must be like. "You have," Ryan Tucker informs us, "to balance school, friends, learning to drive without taking out the mailbox, and managing a couple of heavily networked and overused machines."

Benefit of his copy of his favorite O'Reilly text, young Ryan tells us, he'd really be in Stress City. "Without it, I probably would not be SLIPped and sending this e-mail via UUCP. If I'm still SLIPped when I send this, I'll send it via TCP/IP." Not until reading the Linux Network Administrator's Guide did he realize that such a thing was possible! Whenever something between my machine and the Net breaks," he says, "it's the first thing I reach for." Ryan also owns Managing Internet Information Services, with which he's set up httpd once and majordomo twice. He looks forward to similar projects and anticipates them going off without a hitch. Ah, the optimism of youth!

Since his school is only just beginning to "get into the Internet thing," he uses his ORA books almost entirely at home, where he's presently working on World Wide Web page design as a project for the Talented and Gifted Program. "My home page," he admits glumly, "has been known to put people to sleep."

He wants to buy more O'Reilly books—as what teen doesn't?—but is too busy attending school on the one hand, and watching movies and munching popcorn on the other, to work.

"Such," he sighs, "is the life of a teenager with the root password."

Check out these new O'Reilly Internet titles

Exploring Java
By Pat Niemeyer & Josh Peck
1st Edition April 1996 (est.)
ISBN 1-56592-184-4, $24.95 (est.)

CGI Programming
By Shishir Gundavaram
1st Edition March 1996 (est.)
375 pages (est.)
ISBN 1-56592-168-2, $29.95 (est.)

Getting Connected
By Kevin Dowd
1st Edition May 1996 (est.)
450 pages (est.)
ISBN 1-56592-154-2 $29.95 (est.)

Practical UNIX & Internet Security
By Sisson Garfinkel & Gene Spafford
2nd Edition April 1996 (est.), 930 pages (est.)
ISBN 1-56592-148-5, $39.95 (est.)
USENIX members receive a 15% discount

The Internet Navigator, 2ed
Paul Gilster
1-05260-4 $24.95 member price: $21.20
# of copies: __________

Advanced Topics in UNIX
Ronald Leach
1-03663-3 $24.95 member price: $21.20
# of copies: __________

Introduction to Client Server Systems
Paul Renaud
1-57774-X $34.95 member price: $29.70
# of copies: __________

Portable UNIX
Douglas Topham
1-57926-2 $14.95 member price: $12.71
# of copies: __________

UNIX, Self-Teaching Guide
George Leach
1-57924-6 $19.95 member price: $16.95
# of copies: __________

Object Oriented Programming with Turbo C++
Keith Weiskamp
1-52466-2 $24.95 member price: $21.20
# of copies: __________

Obfuscated C and Other Mysteries
Don Libes
1-57805-3 $39.95 member price: $33.96
# of copies: __________

Finding It On the Internet
Paul Gilster
1-03857-1 $19.95 member price $16.95
# of copies: __________

Internationalization: Developing Software for Global Markets 1-07661-9 (pub. date: 1/95)
Tuoc Luong $29.95 member price: $25.45
# of copies: __________

Adventures in UNIX Network Applications Programming
Bill Rieken
1-52858-7 $39.95 member price: $33.96
# of copies: __________

UNIX Shell Programming, 3e
Lowell Jay Arthur
1-59941-7 $29.95 member price: $25.45
# of copies: __________

The UNIX Command Reference Guide
Kaare Christian
1-85580-4 $32.95 member price: $28.01
# of copies: __________

Berkeley UNIX: A Simple & Comprehensive Guide
James Wilson
1-61582-X $40.95 member price: $34.80
# of copies: __________

Payment enclosed, plus sales tax

VISA

Mastercard

American Express

Card No. ________________________________

Name __________________________________________________________________________

Firm ___________________________________________________________________________

Address __________________________________________________________________________

City/State/Zip. ____________________________

Signature ________________________________

(order invalid unless signed)

Please send all orders to:

John Wiley & Sons, Inc.
Attn: Karen Cooper, Special Sales
605 Third Avenue
New York, NY 10158
Phone: (212) 850-6789
Fax: (212) 850-6142
USENIX members receive a 15% discount on the following MIT Press publications:

GLOBAL NETWORKS
Computers and International Communication
edited by Linda M. Harasim
Global Networks takes up the host of issues raised by the new networking technology that now links individuals, groups, and organizations in different countries and on different continents. The twenty-one contributions focus on the implementation, application and impact of computer-mediated communication in a global context.
340 pp. $29.95 hardcover HARNH

TECHNOLOGY 2001
The Future of Computing and Communications
edited by Derek Leebaert
Researchers, executives, and strategic planners from inside the companies and laboratories that have shaped today's information age forecast the merging technologies that could well define the computing and communications environment that lies ahead.
392 pp. $14.95 paperback LEEEP

THE DIGITAL WORD
Text-Based Computing in the Humanities
edited by George P. Landow and Paul Delany
This book explores the larger realm of the knowledge infrastructure where texts are received, reconstructed, and sent over global networks.
Technical Communication and Information Systems series 384 pp. $39.95 hardcover LANDH

SOCIO MEDIA
Multimedia, Hypermedia, and the Social Construction of Knowledge
edited by Edward Barrett
Sociomedia continues the assessment of hypertext and hypermedia systems begun in Text, Context, and HyperText and The Society of Text. It examines the use of integrated multimedia to support social or collaborative research, learning, and instruction in the university, one of the best environments for developing and analyzing the effects of computing technologies on our understanding of complex sets of information.
Technical Communications and Information Systems Series 360 pp. $50.00 hardcover BARRH

CONNECTIONS
New Ways of Working in the Networked Organization
Lee Sproull and Sara Kiesler
"...Sproull and Kiesler raise crucial questions about our technical and particularly our human strategies as a producing society."
— Howard Webber, Sloan Management Review
228 pp. $21.95 paperback SPRCP

TECHNOBABBLE
John A. Barry
"A serious study of the language of the new technocracy."
288 pp. $12.50 paperback BARCP
Prentice Hall PTR is pleased to recommend the following titles to USENIX members...

USENIX members receive a 15% discount on orders


**Object-Oriented Modeling and Design**, James Rumbaugh, 0-13-629841-9 (62984-0) List: $54.00 Members: $45.90


The Magic Garden Explained, Bernard Goodheart/James Cox, 0-13-098138-9 (09813-7) List: $38.00 Members: $32.30


**SCO Performance Tuning Handbook**, Gina Miscovich/David Simons, 0-13-102690-9 (10269-9) List: $42.00 Members: $35.70

**Object-Oriented Programming**, Peter Coad/Jill Nicola, 0-13-032616-X (03261-5) List: $48.00 Members: $40.80

Internetworking with TCP/IP, Vol. III Client Server Programming and Applications for the AT&T TLI Version, Douglas E. Comer and David L. Stevens, 0-13-474222-2 (47422-1) List: $53.00 Members: $45.05

Internetworking with TCP/IP, Vol. III Client Server Programming and Applications for the AT&T TLI Version, Douglas E. Comer and David L. Stevens, 0-13-474230-3 (47423-9) List: $53.00 Members: $45.05

The Internet Message: Closing the Book with Electronic Mail, Marshall T. Rose, 0-13-092941-7 (09294-0) List: $50.00 Members: $42.50


All About Administering the NIS+, Second Edition, Rick Ramsey, 0-13-309576-2 (30957-5) List: $42.00 Members: $35.70

**The Simple Book**: An Introduction to Internet Management, Marshall T. Rose, 0-13-177254-6 (17725-3) List: $55.00 Members: $46.75

Networking Operations on UNIX SVR4, Mike Padavano, 0-13-613555-2 (61355-4) List: $50.00 Members: $42.50

Solaris Porting Guide, Sunsoft ISV Engineering 0-13-030396-8 (03039-5) List: $52.00 Members: $44.20

Multiprocessor System Architectures, Ben Catanzaro 0-13-089137-1 (08913-6) List: $44.00 Members: $37.40

**The HP-UX System Administrator's "How To" Book**, Marty Poniatowski, 0-13-099821-4 (09982-0) List: $32.00 Members: $27.20

**UNIX System V Performance Management**, Edited by Phyllis Eve Bregman and Sally A. Browning 0-13-016429-1 (01642-8) List: $29.95 Members: $25.45

**SCO UNIX® Operating System System Administrator’s Guide**, Santa Cruz Operation, 0-13-012568-7 (01256-7) List: $39.00 Members: $33.15

HERE’S HOW TO ORDER:

CALL
800-880-6818

OR WRITE:
CompuBooks
Route 1, Box 271-D,
Cedar Creek, TX 78612

OR INTERNET:
70007.1333@CompuServe.com
(500 CBK on CompuServe)

WE SHIP ANYWHERE!

FOR MORE INFORMATION, OR QUANTITY ORDERS, PLEASE CALL 201-592-2657
A UNIQUE OFFER
ON THE BEST IN UNIX
FOR USENIX MEMBERS

□ THE INTERNET GUIDE FOR NEW USERS
D. Dern
hardcover, 016510-6, $40.00, MEMBER PRICE $32.00
paperback, 016511-4, $27.95, MEMBER PRICE $22.36

□ INTERNET FOR EVERYONE
R. Wiggins
hardcover, 067018-8, $29.95, MEMBER PRICE $23.96
paperback, 067019-6, $45.00, MEMBER PRICE $36.00

□ THE ESSENTIAL INTERNET INFORMATION GUIDE
J. Manger
707905-1, paperback, $27.95, MEMBER PRICE $22.36

□ THE INFORMATION BROKERS HANDBOOK, Second Edition
S. Rugge
911878-x, paperback, $34.95, MEMBER PRICE $27.96
Available December 1994

□ SAA AND UNIX: IBM's Open System Strategy
M. Killen
034607-0, $40.00, MEMBER PRICE $32.00

□ A STUDENT'S GUIDE TO UNIX
H. Hahn
025511-3, paperback, $28.00, MEMBER PRICE $22.40

□ UNIX DEVELOPER'S TOOL KIT
K. Leininger
911836-4, $65.00, MEMBER PRICE $52.00

□ UNIX SECURITY: A Practical Tutorial
N. Arnold
002560-6, $24.95, MEMBER PRICE $19.96

□ THE UNIX AUDIT: Using UNIX to Audit UNIX
M. Grottola
025127-4, $32.95, MEMBER PRICE $26.36
Available November 1994

□ UNIX: A Database Approach
S. Das
015745-6, $29.95, MEMBER PRICE $23.96
Available November 1994

I am a member of USENIX Association. Please send me the books I have indicated at the member special rate. I have added $3.00 postage and handling for the first book ordered, $1.00 for each additional book, plus my local sales tax.

Check or money order is enclosed—payable to McGraw-Hill, Inc.
Charge my □ Visa □ Mastercard □ Discover □ Amex

Account #__________________________
Expiration Date__________________________
Signature________________________________

USENIX Membership #__________________________

Bill & Ship To:
Name__________________________
Street__________________________
City, State, Zip__________________________
Daytime Phone #__________________________

Send or Fax Orders to:
 McGraw-Hill, Inc.
Attn: Rosa Perez
11 West 19th Street—4th Floor
New York, New York 10011
Fax: 212-337-4092

If I am not completely satisfied, I will return the book(s) within 15 days for a full refund or credit. Satisfaction unconditionally guaranteed. Prices subject to change without notice. We can only accept orders within the continental USA.
The value of UniForum general membership is about to be enhanced with the publication of the 1996 Open Systems Products Directory. You'll receive your personal copy with your paid membership (or renewal) to UniForum. Priced to sell to non-members at $150 per copy, the 1996 Open Systems Products Directory is a real value. But, why spend an extra penny when it's free with paid membership?

Your $125 UniForum membership dues entitle you to a wealth of benefits and services, including: IT Solutions Magazine, UniNews Newsletter, UniForum Technical Publications, Member Discounts and the 1996 Open Systems Products Directory. This entire package is available for one low fee. Call toll free (800) 255-5620 or (408) 986-8840 outside the U.S. Have your major credit card ready and we'll start your membership at once.

Now On-line!

The Open Systems Products Directory has long been recognized as the most comprehensive guide to open systems products and services. It carries more than 8,700 detailed product and services listings — more than ever before. With over 2,200 vendors, the Directory offers more than 1,900 pages of: Horizontal and Vertical Software, Network and Communications Software, Application Development Tools, System Hardware and Peripheral Devices, Operating Systems and Systems Software, and services such as Publications, Consultants, Conferences, Training, and much, much more.

A collection of four detailed indexes: Vendors, Products, Keywords, and Operating Systems make the Directory indispensable. No wonder UniForum members renew year after year just to receive their latest edition. Users tell us they take it with them wherever they go on the job. The 1996 Open Systems Products Directory quite simply is a resource tool you cannot afford to be without. It is yours free with UniForum membership.
LOCAL USER GROUPS

The Association will support local user groups by doing a mailing to assist in the formation of a new group and publishing information on local groups in \login:. At least one member of the group must be a current member of the Association. Send additions and corrections to: <login@usenix.org>.

California

Fresno: The Central California UNIX Users Group has a WWW contact page to which members may post questions or information. For connection information:

• Steve Mitchell
  209 278 5675
  <http://warpig.cat.ucsfresno.edu/ccuug/ccuug.html>

Orange County: Meets the 2nd Monday of each month

• UNIX Users Association of Southern California
  Dave Close
  714 434 7359
  <dhcloset@alumni.caltech.edu>
  New Horizons Computer Learning Center
  1231 E. Dyer Rd., Suite 140
  Santa Ana, CA 92705
  714 438 9440

Colorado

Boulder: Meets monthly at different sites; for membership information and meeting schedule, send email to <fruug-info@fruug.org>.

• Front Range UNIX Users Group
  Lone Eagle Systems Inc.
  636 Arapahoe #10
  Boulder, CO 80302
  Steve Gaede
  303 444 9114
  <gaede@fruug.org>
  <http://www.fruug.org/~fruug>

Washington, D.C.

Meets 2nd Tuesday of each month.

• Washington Area UNIX Users Group
  10440 Shaker Drive, Suite 103
  Columbia, MD 21046
  Alan Fedder
  301 621 5500
  <afedder@mcsw.com>

Florida

Coral Springs:
• S. Shaw McQuinn
  305 344 8686
  8557 W. Sample Road
  Coral Springs, FL 33065

Melbourne: Meets the 3rd Monday of every month.

• Space Coast UNIX Users Group
  Steve Lindsey
  407 242 4766
  <lindsey@vnet.ibm.com>

Orlando: Meets the 3rd Thursday of each month.

• Central Florida UNIX Users Group
  Mikel Manitius
  407 384 4644
  <mikel.manitius@east.sun.com>

Western: Meets 1st Thursday of each month.

• Florida West Coast UNIX Users Group
  Mike Delucia
  813 882 0770
  <pfl@cfinet.com>

Georgia

Atlanta: Meets on the 1st Monday of each month in White Hall, Emory University.

• Atlanta UNIX Users Group
  P.O. Box 12241
  Atlanta, GA 30355-2241
  Mark Landry 404 365 8108

Kansas and Missouri

Meets on 2nd Tuesday of each month.

• Kansas City UNIX Users Group (KCUUG)
  P.O. Box 412622
  Kansas City, MO 64141
  816 891 1093
  <richj@northcs.cps.com>

Michigan

Detroit/Ann Arbor: Meets on the 2nd Thursday of each month in Ann Arbor.

• Southeastern Michigan Sun Local Users Group and Nameless UNIX Users Group
  Steve Simmons
  office: 313 769 4086
  home: 313 426 8981
  <scs@lokkur.dexter.mi.us>

Minnesota

Minneapolis/St. Paul: Meets the 1st Wednesday of each month.

• UNIX Users of Minnesota
  17130 Jordan Court
  Lakeville, MN 55044
  Robert A. Monio
  612 220 2472
  <pnessutt@dmshq.mn.org>

Missouri

St. Louis:

• St. Louis UNIX Users Group
  P.O. Box 2182 St. Louis, MO 63158
  Terry Linhardt
  314 772 4762
  <uunet!jgalstl!terry>

Nebraska

Omaha: Meets monthly.

• /usr/group/nebraska
  P.O. Box 31012
  Omaha, NE 68132
  Phillip Allendorfer
  402 423 1400

New England

Northern: Meets monthly at different sites.

• Peter Schmitt 603 646 2085
  Kiewit Computation Center
  Dartmouth College Hanover, NH
  03755
  <peter.schmitt@dartmouth.edu>

New Mexico

Albuquerque: ASIGUNIX meets every 3rd Wednesday of each month.

• Phil Hertz 505 275 0466.

New York

New York City: Meets every other month in Manhattan.

• Unigroup of New York City
  G.P.O. Box 1931
  New York, NY 10116
  <uniboard@unigroup.org>
  J. P. Radley 212 877 0440

76;login: VOL. 21 - NO. 2
Oklahoma

Tulsa: Meets 2nd Wednesday of each month.

- Tulsa UNIX Users Group, USR Bill Hunt 918 494 4848
  <bhunt@tulsix.utulsa.edu>
- Mark Lawrence 918 749 7498
  <lawrence@tulsix.utulsa.edu>

Texas

- Capital Area Central Texas UNIX Society (CACTUS)
P.O. Box 9786
Austin, TX 78766-9786
Ronald S. Woan
512 838 1254
<president@caactus.org>
<http://caactus.org>

- Dallas/Fort Worth UNIX Users Group
  P.O. Box 867405
  Plano, TX 75086
  Evan Brown 214 519 3577
  <evbrown@dssc.com>

- Houston UNIX Users Group (Hounix)
  answering machine
  713 684 6590
  Jack Gilbert, President
  713 862 3637
  <jack@hounix.org>

Washington

- Seattle: Meets monthly.

- Seattle UNIX Group Membership
  Bill Campbell 206 947 5591
  6641 East Mercer
  Mercer Island, WA 98040-0820
  <bill@celestial.com>

Canada

- Manitoba: Meets 2nd Tuesday of each month.

- Manitoba UNIX User Group (MUUG)
P.O. Box 130
St. Boniface Winnipeg,
MB R2H 3B4
Barry Finch, President
204 934 1690 <info@muug.mb.ca>

- Ottawa:
  - The Ottawa Carleton UNIX Users Group
    David J. Blackwood
    613 957 9305 <dave@revcan.ca>

- Toronto:
  - 143 Baronwood Court
    Brampton, Ontario
    Canada L6V 3H8
    Evan Leibovitch
    416 452 0504 <evan@telly.on.ca>

- Quebec: Meets first Wednesday every 3rd month.

- Administrateurs de Système UNIX du Quebec (ASUQ)
  Université de Montreal,
  Dept. IRO
  C.P. 6128, Succ. Centre-Ville
  Montreal, Quebec, Canada,
  H3C 3J7
  514 343 7480
  <asuq@iro.umontreal.ca>

System Administration Groups

Back Bay LISA (BBLISA)

New England forum covering all aspects of system and network administration, for large and small installations. Meets monthly, at MIT in Cambridge, MA.

- For information, contact:
  - J. R. Oldroyd 617 227 5635
  <jr@opal.com>
  - Mailing list subscription:
    <bblisa-request@bblisa.org>
  - Mailing list postings:
    <bblisa@bblisa.org>
  - For current calendar of events:
    <bblisa@header.bblisa.org>

Bay LISA (Bay Area, California)

Meets 3rd Thursday of each month at Synopsys in Mountain View, CA.

- For information, contact: <blw@baylisa.org> or <http://www.baylisa.org>

Beach LISA (San Diego, CA)

Meets 2nd Tuesday of each month at the San Diego Super Computer Center, UCSD. For more information, contact:

- Nancy Milligan
  <npm@nmcs.com>
  (619)260-1442
  <http://nmcs.com/beachlisa>

dc.sage (Metropolitan Washington, D.C.)

"Users can be a friend of the system administrator, but they will never be able to be a peer." We're here to meet, interact, support, leverage, and otherwise make your vocation a more fruitful one. For more information, send "info dc-sage" to <majordomo@mrj.com>.

Ken Mayer <kmayer@mrj.com>

$GROUPNAME (New Jersey)

$GROUPNAME is an organization in New Jersey formed to facilitate information exchange pertaining to the field of UNIX system administration. For more information, send "infogroupname" to <majordomo@plts.org>.

Tom Limoncelli <tal@big.att.com>

New York Systems Administrators (NYSAs)

Meets 2nd Monday of each month.

<nyas-request@esm.com>
914 472 3635 or 472 3635

North Carolina System Administrators Group

The North Carolina System Administrators Group meets on the 2nd Monday each month around the Research Triangle Park area.

- Amy Kreiling 919 962 1843
  <kreiling@cs.unc.edu>
- William E. Howell 919 941 4868
  <william_howell@glaxo.com>

APRIL 1996;login: 77
This is a combined calendar of conferences, symposia, and standards meetings. If you have an event that you wish to publicize, please contact <login@usenix.org>. For complete USENIX conference and symposia listings see URL http://www.usenix.org/events/general.html.

* = events sponsored by the USENIX Association.

### 1996

#### April
- 3 - 4 NetWorld+Interop '96, Las Vegas, NV
- 14 - 19 IEEE POSIX, Jackson Hole, WY

#### May
- 6 - 8 IEEE Symposium on Security and Privacy, Oakland, CA
- 13 - 17 SANS V, Washington, DC
- 18 - 24 DECUS, Orlando, FL
- 21 - 24 SIGPLAN '96, Philadelphia, PA

#### June
- 1 - 6 DECUS '96, St. Louis, MO
- 10 - 14 NetWorld+Interop '96, Frankfurt, Germany
- 17 - 21 COOTS II, Toronto, Canada
- 24 - 28 INET '96, Montreal, Canada
- 24 - 28 IETF, Montreal, Canada

#### July
- 10 - 13 *Tcl/Tk, Monterey, CA
- 14 - 19 IEEE POSIX, Nashua, NH
- 15 - 19 NetWorld+Interop '96, Tokyo, Japan
- 22 - 25 *6th UNIX Security, San Jose, CA
- 28 - 31 FIRST, Santa Clara, CA

#### August
- 4 - 9 SIGGRAPH, New Orleans, LA
- 5 - 9 Interex '96, San Diego, CA
- 26 - 30 SIGGCOMM, Stanford, CA

#### September
- 3 - 5 GUUG, Leipzig, Germany
- 16 - 20 NetWorld+Interop '96, Atlanta, GA
- Oct 4 * LISA '96, Chicago, IL
- AUUG, Melbourne, Australia

#### October
- 1 - 4 ASPLOS VII, Cambridge, MA
- 7 - 11 NetWorld+Interop '96, Paris, France
- 8 - 10 UNIX Expo, New York City
- 10 - 16 OOPSLA '96, San Jose, CA
- 23 - 25 IEEE Symposium on Reliable Distributed Systems, Niagara, Canada
- 27 - 28 *IWOOOS '96, Seattle, WA
- 28 - 31 *OSDI II, Seattle, WA
- Nov 1 NetWorld+Interop '96, London, England

### 1997

#### January
- 6 - 9 *Linux Conference, Anaheim, CA
- 6-10 *USENIX, Anaheim, CA
- 20-24 POPL '97

#### March
- 1 - 5 ACM '97, San Jose, CA
- 12-14 UniForum, San Francisco, CA

#### April
- 7 - 11 IETF, Memphis, TN
- 21 - 26 SANS, Baltimore, MD

#### May
- 5 - 7 HotOS-VI

#### June
- 5 - 7 WITI, Santa Clara, CA
- 16 - 20 LISA '97

#### October
- 12 - 17 OOPSLA '97
- 27 - 31 LISA '97, San Diego, CA

### 1998

#### June
- 15 - 19 *USENIX, New Orleans, LA

#### December
- 7 - 11 LISA '98, Boston, MA
- JUS UNIX Fair
USENIX
1996 Conferences, Symposia, and Workshops for UNIX and Advanced Computing Systems Professionals

If these topics are important to you:

- Tcl/Tk
- UNIX Security and Applications of Cryptography
- Object-Oriented Technology
- Systems Administration
- Operating Systems Design and Implementation
- Electronic Commerce

then save these dates!

Plan to attend these USENIX events:

- 10th Systems Administration Conference (LISA). September 30-October 4, 1996. Chicago
- 2nd Electronic Commerce Workshop.

For more information:

USENIX, 22672 Lambert St., Suite 613, Lake Forest, CA 92630
Phone: 714.588.8649
Fax: 714.588.9706
Email: conference@usenix.org
WWW: http://www.usenix.org

USENIX, the UNIX and Advanced Computing Systems Technical and Professional Association, offers technical conferences for and by technical professionals. SAGE, the System Administrators Guild, a special technical group within USENIX, is dedicated to the advancement and recognition of system administration as a profession.
Wanted:
Papers to publish
and authors
to present

(Turn to pages 60-69)