

```

// Interface to encapsulate all client interface applet requests
public interface IRequest {
    // Downcall from the director to the request object. Perform the action
    // on 'hostname'. Throw RequestException if an error occurs
    public void action(String hostname) throws RequestException;
    // Downcall from the director to the request object upon failure. Perform
    // any necessary cleanup code. The state of the failed request consists
    // of the 'oldhostname' and the RequestException that was thrown from the
    // action method
    public void cleanup(String oldhostname, RequestException t);
}

// Generic interface for all director applets
public interface IDirector {
    // Execute the request r on a hostname of the director's choosing. If the
    // request object throws a RequestException, assume failure of the node
    // and reapply the request after calling the request's cleanup method.
    // If there are no remaining nodes, return false. Otherwise, return true.
    public boolean apply(IRequest r);
}

```

Figure 5: This Figure describes some of the interfaces in the Smart Clients API. Classes that implement the director interface have been written to provide much of the functionality necessary to simple directors, including randomized directors (picking a random machine) and directors based on choosing the least loaded server.

manual load balancing. To improve on this interface, Smart Client applets present a single download button to the user. The client interface applet delivers requests to the director to retrieve a file, while the director picks a machine at random from a static set of nodes. When the user presses the button, the applet transparently determines the best site for file retrieval.

To demonstrate the scalability available from using Smart Clients, we measure delivered bandwidth to Smart Client applets running in Netscape Navigator from a varying number of FTP servers. We emphasize that the choice of FTP site is transparent to the end user (a single button is pressed to begin file retrieval) and that our FTP application can be downloaded to run with unmodified servers and Java-compliant browsers. The tests were run on a cluster of Sun Sparcstation 10's and 20's interconnected by a 10 Mbps Ethernet switch. The Ethernet switch allows each machine in the cluster to simultaneously deliver 10 Mb of aggregate bandwidth to the rest of the cluster without the contention associated with shared Ethernet networks. Either one, two, or four of the machines are designated FTP servers, while the rest of the machines in the cluster attempt 40 consecutive retrievals of a 512 KB file. This experimental setup approximates multiple FTP mirror sites spread across the wide area.

Figure 6 summarizes the results of the FTP scalability tests. The graph shows aggregate delivered

bandwidth in megabytes per second as a function of the number of client machines making simultaneous file requests. For one FTP server, 8 clients are able to saturate the single available Ethernet link at 1.2 MB/s³. The results for two and four FTP servers demonstrate that the random selection of an FTP server used within the applet delivers reasonable scalability. Sixteen clients are able to retrieve approximately 2 MB/s from two servers, while 16 clients saturate four servers at approximately 3 MB/s.

For small number of clients, a single FTP server demonstrates the best performance because all 40 file downloads are made during a single connection. For the multi-server tests, multiple connections and disconnections take place as the clients attempt to randomly balance load across the servers. In the future, this problem can be avoided by implementing site affinity with successive file requests (if delivered bandwidth on the previous was deemed satisfactory), implementing a load daemon on the nodes to allow the clients to continuously choose lightly loaded machines, or by using services such as the Internet Weather Map [Mat 1996] to choose low-latency hosts. This information can be used to incrementally scale connections to available FTP servers (i.e. allow some machines to be recruited only when needed).

³We were unable to take measurements for more than 16 simultaneous clients making requests to a single server because the FTP server would not allow more than 16 simultaneous file retrievals. We plan to investigate this limitation further.

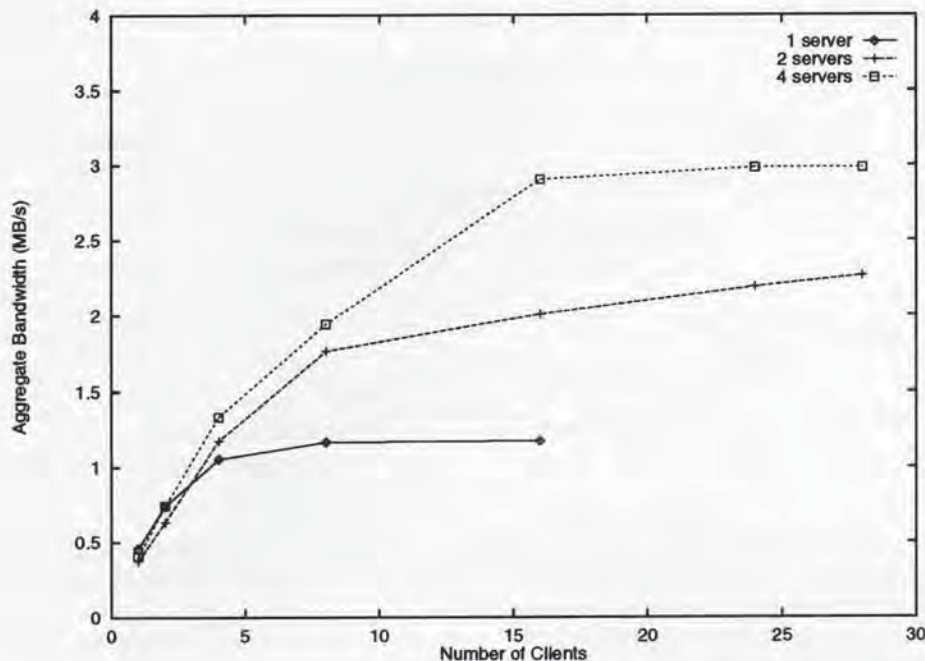


Figure 6: This figure demonstrates how a Smart Client interface to FTP delivers scalable performance. The graph shows delivered aggregate bandwidth as a function of number of clients making simultaneous requests.

4.3 Scalable Chat

The next application we implement is Internet chat. The application allows for individuals to enter and leave chat rooms to converse with others co-located in the same logical room. The chat application is implemented as Java applets run through a Web browser. Figure 7 depicts our implementation of the application. Individual chat rooms are modeled as files exported through WebFS [Vahdat et al. 1996], a file system allowing global URL read/write access. WebFS provides for negotiation of various cache consistency protocols on file open.

We extended WebFS to implement a scalable caching policy suitable to the chat application. In this model, when a user wishes to enter a chat room, the client simply opens a well-known file associated with the room. This operation registers the client with WebFS. Read and write operations on the file correspond to receiving messages from other chatters and sending a message out to the room, respectively. On receiving a file update (new message), WebFS sends the update to all clients which had opened the file for reading (i.e., all chatters in a room). In this case, the client interface applet consists of two threads, a read thread continuously polling the chat file and an event thread writing user input to the chat file. These read/write requests are sent to the chat

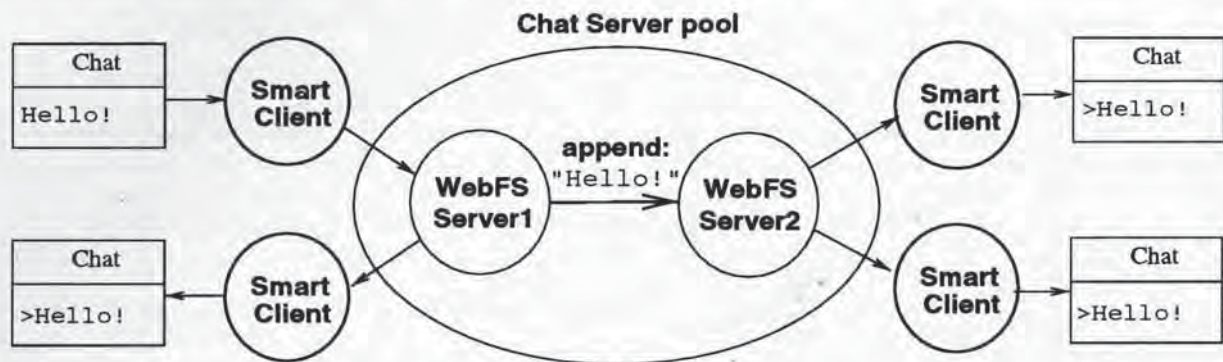
server via the director applet.

The director sends the request to the hostname that represents the best service node at the time. If the request does not complete, the request raises an exception to the director applet. The director applet then calls the service-specific cleanup routine for the request, and resends it to another service node. Note that the request takes a service specific failure event, such as chat file not found or WebFS server is down, and translates it into a general exception. Thus, the director applet can be written for a cluster of machines and reused for many different protocols: FTP, Telnet and chat.

From the above discussion, it is clear that a single WebFS server can quickly become a performance bottleneck as the number of simultaneous users is scaled. To provide system scalability, we allow multiple WebFS servers to handle client requests for a single file. Each server keeps a local copy of the chat file. Upon receiving a client update, WebFS distributes the updates to each of the chat clients connected to it. WebFS also accumulates updates, and every 300 ms propagates the updates to other servers in the WebFS group. This caching model allows for out of order message delivery, but we deemed such semantics to be acceptable for a chat application. If it is determined that such semantics are insufficient, well-known dis-

```
write(http://server1/chat, "Hello!");
```

```
read(http://server2/chat, &x);
```



```
read(http://server1/chat, &x);
```

```
read(http://server2/chat, &x);
```

Figure 7: Implementation of the chat application. Chat rooms are modeled as files with reads corresponding to receiving conversation updates and writes to sending out a message. On a write, the WebFS updates all its clients; the updates are propagated to other servers in a lazy fashion.

tribution techniques [Ladin et al. 1992, Birman 1993] can be used to provide strong ordering of updates.

Since the read requests are idempotent, and the write requests are atomic with respect to WebFS, the chat application is completely tolerant to server crashes. This fault transparency provides the illusion of a single, highly-available chat server machine to the programmer of the Chat client interface applet. Figure 8 demonstrates the behavior of the chat application in the face of a failure to the client's primary server. The graph plots response time as a function of elapsed time. The graph shows that chat delivers less than 5 ms latency to the end user. On detecting a failure, the latency jumps to 1 second before switching to a secondary WebFS server, at which point the latency returns to normal.

5 Summary

We have described a solution to the problem of scalability and high-availability which logically migrates server functionality into the client. We will now revisit the goals set forth in Section 1 and examine how Smart Clients addresses each goal:

- **Incremental Scalability** - When a machine is added to or removed from a service group, the director applet supplied by the service updates its list of peer servers. The director applet discovers such modifications through a service-specific mechanism, e.g. keep-alive messages, connecting to a well-known port, or refetching the ser-

vice certificate.

- **Load Balancing** - The director applet maintains a service-specific notion of load (such as number of processes, number of open connections, available bandwidth). Using this information, client requests are routed to the best candidate node.
- **Fault Transparency** - When a failure occurs, the director applet allows the client to clean up any stale state before resending the request to another server. Providing fault transparency requires service support when the request is non-idempotent. For example, in the chat application, the chat service provides atomic writes to the chat transcript.
- **Wide Area Service Topology** - Smart Clients does not place any restriction on topology of server machines. In fact, the director applet can choose an arbitrary site based on considerations such as proximity or predicted performance.
- **Scalable Services To Legacy Servers** - Existing servers that replicate a read-only database can be grouped together for access by Smart Clients. With knowledge of the group set, the director applet can load balance client requests among existing unmodified servers.

Finally, we believe that the architecture presented in this paper can simplify implementation of scalable services with respect to at least fault transparency and load balancing. The Smart Client director provides the illusion of a single, highly available server. This

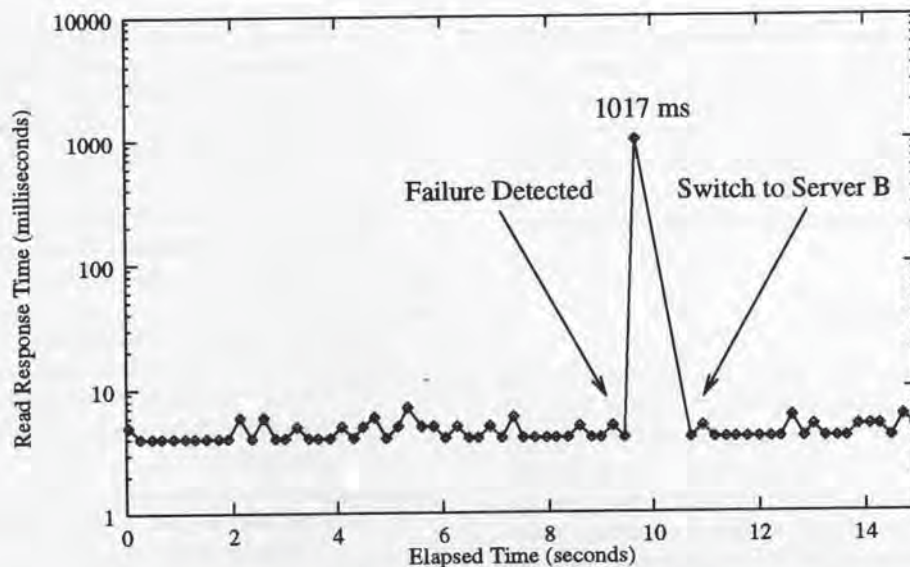


Figure 8: Chat response times in the face of server load. The chat application delivers latencies of approximately 10 ms under normal circumstances. On server failure, the applications takes one second to switch to a peer server.

model substantially decreases the complexity of the client interface applet since this applet need not be concerned with maintaining the set of server nodes. In addition, because of the public interface between the client interface and director applets, each can be written once and interchanged for a number of different services.

6 Related Work

The problem of transparently providing fault transparency and load balancing to network services has been addressed previously in a number of contexts. File systems have used server-side replication of volumes and servers to provide fault transparency in systems such as Deceit [Marzullo et al. 1990], AFS [Howard et al. 1988], and HA-NFS [Bhide et al. 1991]. More recently, systems such as xFS [Anderson et al. 1995b] and Petal [Lee & Thekkath 1996] use client-side techniques to improve overall file system performance. Many distributed clusters perform load balancing on the level of jobs (interactive or otherwise) submitted to the system [Nichols 1987, Bricker et al. 1991, Douglass & Ousterhout 1991, Zhou et al. 1992]. Once again, all these systems implement server-side solutions for load balancing and require client intervention to spread jobs among cluster machines.

Perhaps most closely related to Smart Clients are Transaction Processing monitors [Gray & Reuter

1993] (TP monitors). TP monitors provide functionality similar to Smart Clients for access to databases. The TP monitor functions as the director for transactions to *resource managers*, accounting for load on machines, the RPC program number, and any affinity between client and server. Resource managers are usually SQL databases, but can be any server that supports transactions. TP monitors differ from Smart Clients in that they deal exclusively with transactional RPCs as the communication mechanism to the servers. TP monitors are also more closely coupled with the server nodes since they are responsible for starting new server processes.

The Smart Client director can be tailored to each service, while the TP monitor is more of a general purpose director. Smart Clients also provide a bootstrapping mechanism to remove the single point of failure associated with downloading the necessary routing software. In addition, the Smart Client code is significantly more lightweight than the TP monitor which often includes many of the features of traditional operating systems: process management/creation, authentication, and linking resource manager object code with the Transaction Processing operating system (TPOS). This lightweight nature enables Smart Clients to be downloaded into existing Web browsers to customize existing Internet services.

Also related to our systems are ISIS [Birman 1993] and gossip architectures [Ladin et al. 1992] which provide a substrate for developing distributed applica-

tions. ISIS provides reliable group communication to support many of the applications we envision. Gossip architectures use front-ends analogous to Smart Clients to access replicated servers which are kept consistent through lazy updates. Both systems are orthogonal to our work in many respects and still use server-side techniques for much of their functionality.

7 Conclusions

In this paper, we have shown that existing solutions to providing transparent access to network services suffer from a lack of knowledge about the semantics of individual services. The recent advent of Java allowing distribution of portable client code presents an opportunity to migrate certain service functionality onto the client machine. We show that such migration can simplify service implementation and improve the quality of service to users. To this end, we describe our implementation of Smart Clients to show the greater flexibility available from a client-side approach to building scalable services. The Smart Clients API provides a generic interface for accessing network services. Further, the decomposition of the API into individual client interface and director applets allows interchanging of these applets for a variety of services. The Smart Clients API is specialized to provide scalable access to three sample services: telnet, FTP, and Internet chat.

In the future, we will further explore service-specific load balancing techniques for achieving scalability. We also plan to demonstrate how Smart Clients can be used to provide load balancing and fault transparency for services replicated across the wide area. We also plan to implement an interface for transparent access to HTTP servers and a fault-tolerant telnet client. Migration of other code, besides the director, from the server to the client will also be explored.

Acknowledgments

We would like to thank our shepherd Rob Gingell, Bruce Mah, Rich Martin, and Neal Cardwell for their help in substantially improving the presentation of this paper. We would also like to thank Thomas Wendt for providing the source to jfox, the Java Web browser we modified to support Smart Clients.

References

[Anderson et al. 1995a] T. E. Anderson, D. E. Culler, D. A.

Patterson, and the NOW Team. "A Case for NOW (Networks of Workstations)". *IEEE Micro*, February 1995.

[Anderson et al. 1995b] T. E. Anderson, M. D. Dahlin, J. M. Neefe, D. A. Patterson, D. S. Roselli, and R. Y. Wang. "Serverless Network File Systems". In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, pp. 109–126, December 1995.

[Anderson et al. 1996] E. Anderson, D. Patterson, and E. Brewer. "The Magicrouter, an Application of Fast Packet Interposing". May 1996. Submitted For Publication. Also see <http://HTTP.CS.Berkeley.EDU/~eanders-/magicrouter/>.

[Berners-Lee 1995] T. Berners-Lee. "Hypertext Transfer Protocol HTTP/1.0", October 1995. HTTP Working Group Internet Draft.

[Bhide et al. 1991] A. Bhide, E. N. Elnozahy, and S. P. Morgan. "A Highly Available Network File Server". In *Proceedings of the 1991 USENIX Winter Conference*, pp. 199–205, 1991.

[Birman 1993] K. P. Birman. "The Process Group Approach to Reliable Distributed Computing". *Communications of the ACM*, 36(12):36–53, 1993.

[Bricker et al. 1991] A. Bricker, M. Litzkow, and M. Livny. "Condor Technical Summary". Technical Report 1069, University of Wisconsin—Madison, Computer Science Department, October 1991.

[Brisco 1995] T. Brisco. "DNS Support for Load Balancing", April 1995. Network Working Group RFC 1794.

[Dig 1995] Digital Equipment Corporation. *Alta Vista*, 1995. <http://www.altavista.digital.com/>.

[Douglass & Ousterhout 1991] F. Douglass and J. Ousterhout. "Transparent Process Migration: Design Alternatives and the Sprite Implementation". *Software - Practice and Experience*, 21(8):757–85, August 1991.

[Ghormley et al. 1995] D. Ghormley, A. Vahdat, and T. Anderson. "GLUnix: A Global Layer UNIX for NOW". See <http://now.cs.berkeley.edu/Glunix/glunix.html>, 1995.

[Goldstein & Dale 1995] I. Goldstein and P. Dale. "A Scalable, Fault Resilient Server for the WWW". OSF ARPA Project Proposal, 1995.

[Gosling & McGilton 1995] J. Gosling and H. McGilton. "The Java(tm) Language Environment: A White Paper". <http://java.dimensionx.com/whitePaper/java-whitepaper-1.html>, 1995.

[Gray & Reuter 1993] J. Gray and A. Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, 1993.

- [Howard et al. 1988] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West. "Scale and Performance in a Distributed File System". *ACM Transactions on Computer Systems*, 6(1):51-82, February 1988.
- [Jav 1996] JavaSoft. *Java RMI Specification, Revision 1.1*, 1996. See <http://chatsubo.javasoft.com/current/doc/rmi-spec/rmiTOC.doc.html>.
- [Katz et al. 1994] E. D. Katz, M. Butler, and R. McGrath. "A Scalable HTTP Server: The NCSA Prototype". In *First International Conference on the World-Wide Web*, April 1994.
- [Ladin et al. 1992] R. Ladin, B. Liskov, L. Shirira, and S. Ghemawat. "Providing Availability Using Lazy Replication". *ACM Transactions on Computer Systems*, 10(4):360-391, 1992.
- [Leach 1996] P. Leach. Personal Communication, November 1996.
- [Lee & Thekkath 1996] E. K. Lee and C. A. Thekkath. "Petal: Distributed Virtual Disks". In *Proceedings of the 7th International Conference on Architectural Support for Programming Languages and Operating Systems*, October 1996.
- [Marzullo et al. 1990] K. Marzullo, K. Birman, and A. Siegel. "Deceit: A Flexible Distributed File System". In *Proceedings of the 1990 USENIX Summer Conference*, pp. 51-61, 1990.
- [Mat 1996] Matrix Information and Directory Services, Inc. *MIDS Internet Weather Report*, 1996. See <http://www2.mids.org/weather/index.html>.
- [Net 1994] Netscape Communications Corporation. *Netscape Navigator*, 1994. <http://www.netscape.com>.
- [Nichols 1987] D. Nichols. "Using Idle Workstations in a Shared Computing Environment". In *Proceedings of the Eleventh ACM Symposium on Operating Systems Principles*, pp. 5-12, November 1987.
- [Vahdat et al. 1996] A. Vahdat, M. Dahlin, and T. Anderson. "Turning the Web into a Computer". May 1996. See <http://now.cs.berkeley.edu/WebOS>.
- [Wendt 1996] T. Wendt. *Jfox*, 1996. <http://www.uni-kassel.de/fb16/ipm/mt/java/jfox.html>.
- [Wetherall & Tennenhouse 1995] D. Wetherall and D. L. Tennenhouse. "Active Networks: A New Substrate for Global Applications". 1995. Submitted for Publication.
- [Zhou et al. 1992] S. Zhou, J. Wang, X. Zheng, and P. Delisle. "Utopia: A Load Sharing Facility for Large, Heterogeneous Distributed Computing Systems". Technical Report CSRI-257, University of Toronto, 1992.

Libraries Home | My Library Account | Hours Rutgers Home | Search Rutgers

Research Assistance How Do I...? Site Search

RUTGERS
University Libraries Library Catalog

Basic Advanced Browse Course Reserves
Book Bag (0) My Library Account Help

Keyword
Search

Back to results » Proceedings of the USENIX 1997 annual technical conference : » Holdings

**Proceedings of the USENIX 1997 annual technical conference :
January 6-10, 1997, Anaheim, California, USA /**

Corporate Author: **USENIX Association. Winter Conference**
 Other Contributors: **USENIX Association.**
 Format(s): **Book**
 Language: **English**
 Published: **Berkeley, CA : The Association, c1997.**

Tools

- [Add to Book Bag](#)
- [Permalink](#)
- [Cite](#)
- [Text](#)
- [Email](#)
- [Export](#)
- [Report error](#)

Holdings Description Staff View

[Book Delivery](#) | [Special Request](#)

Library Annex Storage Facility (Busch) - STACKS

QA76.76.O63U84 1997	IN LIBRARY - BOOK-Y
---------------------	---------------------

More on this subject

- [UNIX \(Computer file\) > Congresses.](#)
- [Operating systems \(Computers\) > Congresses.](#)

Site Index
Contact Us
Website Feedback
Privacy Policy

New Brunswick 848-932-6000
Newark 973-353-5901
Camden 856-225-6034
Health Sciences 973-972-4580

Ask A Librarian

INTERNET ARCHIVE
Wayback Machine

36 captures
7 Feb 97 - 7 Mar 12

http://usenix.org/publications/library/proceedings/ana97/program.html Go

JAN Yoshi 1 of 1 x

1996 1997 2000 Help ?

USENIX 1997 Annual Technical Conference

January 6-10, 1997
Anaheim Marriott Hotel
Anaheim, California

The only conference by and for programmers, developers, and system administrators

[Letter from the Program Chair](#)

[What's New At Anaheim 97](#)

[Program Committee](#)

[Conference Overview](#)

[Tutorial Program](#)

[Technical Program](#)

[USELINUX Program](#)

[USELINUX Program Committee](#)

[Conference Activities](#)

[Birds-Of-A-Feather Sessions](#)

[Vendor Exhibits](#)

[Registration Form](#)

Letter from the Program Chair

Dear Conference Attendee:

1996 has been one of the most exciting years for those of us involved in advanced computing. The Web is attracting tremendous interest in the business world corporate America is rushing to embrace emerging technologies. New products are being developed and introduced at breakneck-speed. And you the programmers, developers, and system administrators are the ones being charged to make it happen and make sure it works.

How can you figure out which technologies and products are worth your time and money for 1997 and beyond? If its your job to know, plan to be in Anaheim in January!

You can choose from 20 tutorials. There will be new topics such as secure Java programming and applets, Windows NT and Windows 95, CGI and WWW programming in perl, Web security, setting up a Web server, Solaris system administration, and Linux device drivers. Or you can attend tried-and-true tutorials which have proven their worth on topics such as UNIX Network Programming, IPv6, UNIX security tools, and Perl.

You will hear talks on state-of-the-art practice in the refereed papers track on topics you can use in your daily work, such as Web tools, file system advances, distributed computing, and high-speed networking. Invited speakers share their experiences on topics such as cryptography, programming hints, and WWW server techniques. You can form a BoF ([Birds-of-a-Feather](#)) session with other attendees based on a topic of common interest. And you can talk seriously with the manufacturers at our informal trade show.

Co-located with the USENIX Technical Conference is USELINUX, the first Linux Applications Development and Deployment Conference. Co-sponsored by Linux International and USENIX, this conference will offer a technical track and a business track, as well as presentations and case studies. Anything you wanted to know about Linux, you can find out here. Your registration fee allows you to attend USENIX and USELINUX for the same price.

You need practical tools and information. We are gathering the best experts to present it. In one place. In one week.

We hope this program will help you to make the best use of your time at the Conference. I look forward to seeing you in Anaheim from January 6th to 10th.

Sincerely,

John Kohl, *Pure Atria Corporation*
Program Chair

P.S.: Remember to sign up for tutorials early. Space is limited and demand is high. You'll get your first choice and you'll save money.

Important Dates to Remember:

Early Registration Savings Deadline: November 22, 1996

Hotel Discount Deadline: December 20, 1996

NEW AT ANAHEIM:

USELINUX, the Linux Applications Development and Deployment Conference, co-sponsored by Linux International and USENIX.

If you are:

- An application developer porting or developing Linux applications,
- a system administrator having to maintain Linux systems,
- a business person who wishes to develop a Linux business.

Plan to attend USELINUX. One fee covers the registration for both conference programs, and you can go freely back and forth between them. (Tutorials carry a separate fee for both USENIX and USELINUX).

CONFERENCE ORGANIZERS

Program Chair: John T. Kohl, *Pure Atria Corporation*

Program Committee:

Matt Blaze, *AT&T Research*
Bill Bolosky, *Microsoft Research*
Nathaniel Borenstein, *First Virtual Holdings*
Charlie Briggs, *Digital Equipment Corp.*
Clem Cole, *Digital Equipment Corp.*
Fred Douglass, *AT&T Research*
Rob Gingell, *Sun Microsystems*
Mike Karels, *Berkeley Software Design, Inc.*
John Schimmel, *Silicon Graphics*
Carl Staelin, *Hewlett-Packard Labs*

Invited Talks Coordinators:

Mary Baker, *Stanford University*
Berry Kercheval, *Xerox PARC*

CONFERENCE OVERVIEW

Sunday, January 5

Registration 4:00pm - 9:00pm
Kickoff Reception 6:00pm - 9:00pm

Monday, January 6

Registration 7:30am - 5:00pm
Tutorials 9:00am - 5:00pm

Tuesday, January 7

Registration 7:30am - 5:00pm
Tutorials 9:00am - 5:00pm

Birds-of-a-Feather Sessions 6:00pm - 10:00pm

Wednesday, January 8

Registration 7:30am - 6:00pm
Keynote Address 9:00am - 10:30pm Technical Sessions 11:00am - 5:00pm
USELINUX Developers 9:00am - 5:30pm
Vendor Display Noon - 7:00pm
USELINUX Case Studies 7:30pm - 11:00pm
Birds-of-a-Feather Sessions 7:30pm - 11:00pm

Thursday, January 9

Registration 7:30am - 6:00pm
Technical Sessions 9:00am - 6:00pm
USELINUX Developers 9:00am - 5:30pm
Vendor Display 10:00am - 4:00pm
Birds-of-a-Feather Sessions 6:00pm - 10:00pm
USELINUX Case Studies 6:00pm - 10:00pm

Friday, January 10

Technical Sessions 9:00am - 5:45pm
USELINUX Business 9:00am - 4:00pm

TUTORIAL PROGRAM

Monday and Tuesday, January 6-7, 1997

Technology is changing more rapidly than ever before. Whether you are a programmer, developer or system administrator, you are expected to stay on top of the latest improvements and do your job. Sign up for tutorials and you'll get an immediate payoff from gaining command of the newest developments and putting them to work in your organization.

USENIX tutorials aim to provide the critical information you need. Delivered by experts, tutorials are practical, intensive, and essential to your professional development.

Our guarantee: If you're not happy, we're not happy. If you feel a tutorial does not meet the high standards you have come to expect from USENIX, let us know by the morning break and we will either change you to any available tutorial immediately or arrange for you to attend another tutorial at another USENIX event without paying another fee.

Register now to guarantee your first choice - seating is limited.

Tutorial fees include printed and bound tutorial materials from your sessions, lunch, CD-ROM with Tutorials, Referreed Papers, and Invited Talks. Admission to the Vendor Exhibits

Continuing Education Units

USENIX provides Continuing Education Units (CEUs) for a small additional administrative fee. The CEU is a nationally recognized standard unit of measure for continuing education and training, and is used by thousands of organizations across the United States. Each full-day USENIX tutorial qualifies for 0.6 CEUs. You can request CEU credit by completing the CEU section on the registration form. USENIX provides a certificate for each attendee taking a tutorial for CEU credit, and maintains transcripts for all CEU students.

TUTORIAL OVERVIEW

Monday, January 6

[M1: Beginning Perl Programming for UNIX Programmers \(Updated for Perl 5\)](#)
[M2: The Kerberos Approach to Network Security \(Updated\)](#)
[M3: An Introduction to Java](#)
[M4: Secure Java Programming](#)
[M5: Windows NT and Windows 95 - The Win32 API](#)
[M6: UNIX Network Programming](#)
[M7: Selected Topics in System Administration \(New\)](#)
[M8: How Networks Work - The Limits of Modern Internetworking \(Updated\)](#)
[M9: System and Network Performance Tuning \(New\)](#)
[M10: Inside the Linux 2.0 Kernel \(New\)](#)

Tuesday, January 7

- [T1: UNIX Security Tools: Use and Comparison.](#)
- [T2: CGI and WWW Programming in Perl \(New\)](#)
- [T3: Security on the World Wide Web \(New\)](#)
- [T4: Creating Effective User Interfaces \(New\)](#)
- [T5: Java Applets and the AWT \(New\)](#)
- [T6: Setting Up And Administering A Web Server \(New\)](#)
- [T7: Security for Software Developers: How to Write Code that Withstands Hostile Environments \(New\)](#)
- [T8: Solaris System Administration \(New\)](#)
- [T9: IP version 6: An Introduction](#)
- [T10: Writing Device Drivers Under Linux \(New\)](#)

M1: Beginning Perl Programming for UNIX Programmers (Updated for Perl 5)

Tom Christiansen, *Consultant*

Who should attend: System administrators and toolsmiths, database managers, software test and support engineers, GUI and World Wide Web programmers unfamiliar with Perl or those with little programming experience. You should have a background in UNIX shell programming with a good working knowledge of regular expressions. A background in sed, awk, and/or C programming will prove useful.

What you will learn: Perl syntax and semantics. More importantly, you will learn how to read Perl (and thus learn from other peoples programming experiences).

Have you been spending a lot of time trying to solve problems in the shell or C? Perl is an extremely powerful and robust scripting language that can help you solve problems in less time. Now ten years old, Perl is the tool of choice because of its power, plus, it works on nearly every conceivable platform. Because it incorporates aspects of more than a dozen well-known UNIX tools, experienced UNIX users will come up to speed on Perl rapidly, and even programmers inexperienced at UNIX will learn UNIX through learning Perl.

You will learn about these topics: detailed descriptions and numerous examples of the syntax and semantics of the language; its data types and data structures; operators and control flow; regular expressions; I/O facilities; database access; user-defined functions; writing and using library modules; and an easy intro to Perls object-oriented programming mechanisms.

You will also hear about some of the new Perl 5 modules, including examples of full applications for Tk-based graphical programming, CGI programs, and client/server programming.

NOTE: This course is based on the current release of Perl (version 5.002.) but is not intended to be a detailed discourse on all advanced programming constructs of that release. It is a jump-start course on Perl for experienced UNIX programmers, not an advanced course for previous Perl programmers.

Tom Christiansen is a consultant specializing in Perl applications, optimizations, and training. He is a frequent instructor at USENIX and other conferences. He earned an MS degree in computer science from the University of Wisconsin at Madison.

M2: The Kerberos Approach to Network Security (Updated) Daniel Geer, *Open Market, Inc.*, Jon A. Rochlis, *BBN Planet*

Who should attend:

- System administrators who are concerned about the inherent lack of security and accountability in conventional UNIX network services environments
- System developers responsible for applications for networked workstation environments, particularly those whose environments include networks which are not themselves physically secure (i.e. open networks)
- Technical managers in enterprises where the flow of electronic information is the core of that enterprise and must be protected without imposing the costs of a security culture.

What you will learn: the Kerberos network security system; how to start work on a computing environment that is both open and accountable, and be able to explain all this to your colleagues.

No matter what your position is (developer, system administrator, or manager) security is one of the most urgent issues facing you today. In this course, you will learn how the Kerberos approach might solve the practical challenges of providing security for the cooperative electronic workplace, workplaces that aspire to location and scale independence in the client-server idiom or which need to provide the kind of security services that enable external access to enterprise data.

You will learn about these topics:

- Fundamental network security: threats and possible solutions
- Common fallacies: control of external access vs. internal security mechanisms
- Kerberos network security system extensions and enhancements, commercial providers, details of administering and integrating it in your environment
- Public key techniques
- X.509 authentication model
- Pretty Good Privacy (PGP)
- Privacy Enhanced Mail (PEM)
- The difference between a firewall, a proxy server, and blind faith

Above all, we will stress nuts-and-bolts of making this work in your environment, including administration and integration of this technology with your existing environments.

Daniel E. Geer, Jr. is director of engineering at Open Market, Inc. Formerly he was chief scientist, vice president of technology, and managing director of security consulting services for OpenVision Technologies. He is a consultant to major Wall Street financial institutions, and a frequent speaker at many technology conferences. He holds a Doctor of Science from Harvard.

Jon Rochlis is an engineering manager with BBN Planet where he leads groups developing managed connectivity and security services. Previously he was with OpenVision Technologies, responsible for systems and security management products.

M3: An Introduction to Java Ken Arnold, *Sun Microsystems Labs.*

Who should attend: Experienced programmers and technical contributors familiar with the C and C++ programming languages, the basics of object-oriented programming, and the basics of how the World Wide Web operates and is organized.

What you will learn: An understanding of the structure and features of the Java language.

No matter what your programming background is, programming in Java has become an important skill. Java is one of the most talked-about programming languages because it makes the Web interactive, rather than just a passive medium. Learn about Java and how it enables live content programming. This live content consists of full programs that can be placed into HTML documents. When such a document is loaded into a Java-enabled HTML viewer such as HotJava, the program is run, creating a page with active or even interactive elements.

You will also learn about Java features that allow you to write secure, robust programs that can be relied upon to run in a finite amount of space without crashing into a pile of bits. These features include garbage collection, exceptions, strong typing, and a clean separation between classes and interfaces. You will also examine the set of class and interface libraries which are defined as part of any compliant Java implementation.

You will find out how to create applets, programs that can be included as part of a standard HTML document, including how to write applets, what resources are available to them, and what features of the language and runtime environment allow these applets to be run securely by anyone on the Internet.

If time permits, the day will end in wild speculation and discussion of the possible technological and social impacts of the kind of computing that Java allows.

Ken Arnold is part of the team developing the JavaSoft Remote Messaging Interface for communication between Java code running on different machines. He is co-author, with James Gosling, of The Java Programming Language. He is a leading expert in object-oriented design and implementation, and has written extensively on C and C++ topics for UNIX Review, and is also co-author, with John Peyton, of A C Users Guide to ANSI C.

M4: Secure Java Programming (New)

Marianne Mueller and David Brownell, *JavaSoft*

Who should attend: Java developers who want to learn more about how Java security works.

What you will learn: The basics of Java security and the default applet security policy, as well as new features such as Java code signing and Java APIs for access control lists and certificate management.

Topics will include:

- How to construct an applet, including step by step examples of a commerce-related applet (e.g., shopping cart), and an overview of the applet API.
- How to write applets that do useful things within the confines of the applet security policy, including:
 - A description of the default applet security policy.
 - Using the applets host server to store persistent information.
 - How to take advantage of HotJavas more configurable security environment (e.g. read & write file ACLs).
 - How to send a CGI request from an applet.
 - How to send a servlet request from an applet (servlet is a JavaSoft proposal for server side extension the servlet API can be thought of as a replacement for the CGI API).
 - Configuring your Web environment so that a browser behind a firewall can do DNS name resolution of machines outside the firewall.
- Overview of Secure Java Platform
 - Language features: private, protected, namespace partitioning, memory management and garbage collection, arrays, strings, lack of preprocessor.
 - Learn how to take advantage of these language features to write secure Java applets and applications.
 - Verifier features: Description of what the verifier does for you. Learn how to use the byte code verifier with your stand-alone Java application.
 - Security Manager: How you might design and implement a security manager for a Java stand-alone application.
- How to get accurate and up-to-date info on Java security bugs.
- How to create signed applets and signed servlets.
 - How to create a JAR file. The JAR file is a Java ARchive file, and it can contain class files, gif, jpeg, HTML, etc.
 - How to generate a key pair, register your public key with a public key distribution center, how to sign the JAR file, and distribute the signed JAR file.
 - How to use the Java Access Control List package (java.security.ACL).
 - How to associate limited capabilities with a signed servlet.
 - How to administer the Java Web server so that it only accepts code signed by a set of trusted signatures.
 - How to administer the Java Web server so that it grants limited capabilities to trusted code.
- X509 Certificate Management in Java.

Marianne Mueller and David Brownell are staff engineers in JavaSoft at Sun Microsystems Inc. Before working on security for the Java project, Marianne worked on floating point, compiler optimizations, and tools for multithreaded programming.

M5: Windows NT and Windows 95 The Win32 API (New)

Joseph M. Newcomer, *Consultant*

Who should attend: Programmers who expect to be involved in the implementation of a Win32-based product, or the porting of a UNIX product to Win32. You should be versed in C. Knowledge of X11 or C++ is helpful, but not required. You do not need to have worked in a windowing system.

What you will learn: A basic understanding of the issues and paradigms of Windows programming techniques, in particular, Windows 32 (Windows 95 and Windows/NT).

Topics include:

- Architecture of Win32
- Basic window structure
- Window messages and handler procedures
- Resource scripts
- Menus, dialogs, introduction to controls (built-in widgets)
- Window classes
- Painting a window
- Subclassing windows
- Handling the mouse
- Managing the clipboard
- Common dialogs (file open, etc.)
- Fonts
- The Multiple Document Interface (MDI)
- Timers, threads, and synchronization primitives

Dr. Joseph M. Newcomer is a consultant specializing in Windows application development. He is a Microsoft MVP (Most Valued Professional), an award given for expertise and public service in supporting Microsoft products on public forums. He recently co-authored the book, Win32 GUI Programming. His recent publications include several articles on Windows programming in Dr. Dobbs Journal.

M6: UNIX Network Programming

Richard Stevens, *Consultant*

Who should attend: UNIX/C programmers interested in learning how to write programs that communicate across a network. You should have a basic familiarity with networking concepts and the TCP/IP protocols.

What you will learn: The knowledge required to write network programs and to develop and examine actual examples. Although the tutorial primarily focuses on the Berkeley sockets interface, the course covers UNIX network programming concepts using TCP/IP that are applicable to both sockets and TLI.

You will hear about these topics:

- Introduction (10%): The big picture, standards, UNIX process handling, connections and associations, concurrent vs. iterative servers.
- Berkeley sockets (90%). All the socket functions, TCP and UDP client-server examples, reserved ports, stream pipes, multiplexed I/O, out-of-band data, raw sockets, broadcasting, inetd superserver, constructing Internet addresses, and socket changes with 4.4BSD.

Richard Stevens is an independent consultant and author of the books TCP/IP Illustrated, Advanced Programming in the UNIX Environment and UNIX Network Programming.

M7: Selected Topics in System Administration (New)

Trent Hein, *XOR Network Engineering*

Evi Nemeth, *University of Colorado, Boulder*

Who should attend: System and network administrators

What you will learn: Insights into real-world, common network problems from all-new, crisis case studies.

Topics include:

- IPv6 What will 128-bit IP addresses mean to your site? What features and motivations in IPv6 should you be thinking about when planning your network for the future? Well give you a good overview of the IPv6 standard and explain how it relates to your existing environment.
- Advanced Routing Protocols The days of RIP as a useful routing protocol are numbered. As internetworks scale rapidly, you have no choice but to look towards protocols such as BGP and OSPF for reliable connectivity. Well cover the basics of the protocols and explain their use in real-world environments.
- Security Auditing 101 So, you've done everything the experts recommend and more to secure you site. Now, how do you measure how secure your site really is? Well take you through the anatomy of a security audit from start to finish.
- Network Monitoring Bigger networks need bigger management tools. Until recently, automated network monitoring has been implemented as a mish-mash of home grown tools at most sites. Now there are a number of production quality tools available both commercially and from the net. Well explain what a number of these tools really do and show you a comparison between them.
- Server Performance Years ago, sinking more money into a bigger CPU was often the fix for performance problems. With CPU's outperforming many other aspects of machines today, performance problems most often appear in areas such as network bandwidth, software optimization, memory usage, and system configuration. Learn how to tune your modern UNIX box to get the most bang for your buck.

Trent Hein is chief network architect at XOR Network Engineering. He worked on the 4.4 BSD port to the MIPS architecture at Berkeley, and is co-author of the UNIX System Administration Handbook.

Evi Nemeth, a faculty member in Computer Science at the University of Colorado, has managed UNIX systems for the past 19 years, both from the front lines and from the ivory tower. She is co-author of the best-selling UNIX System Administration Handbook.

M8: How Networks Work - The Limits of Modern Internetworking (Updated)

Vincent C. Jones, *PE*

Who should attend: Technical individuals, regardless of title, who are responsible for the design and upkeep of extended LAN and LAN/WAN networks supporting multiple protocol architectures. A working knowledge of TCP/IP, Ethernet, and the OSI reference model is assumed.

What you will learn: Practical knowledge: how to benefit from emerging technologies without getting burnt and how to take advantage of mature technologies without getting locked in. You will gain an understanding of the theory behind the rules so you will know which rules must be strictly obeyed, which can be safely stretched, and under what conditions.

Technology is getting more complex every day, but how do you evaluate emerging technologies? In this course, you will gain an understanding of the theoretical underpinnings of modern network technology, probing the current limits of performance and distinguishing between those limits which are fundamental and those which are temporary. You will learn what the hype and what the reality on a wide range of critical technologies, from cell relay (ATM) and virtual LANs to IP version 6 and link state routing.

The bottom line is that every networking protocol and technology is a compromise among competing needs. The better you understand the underlying theory, the better able you are to make appropriate choices that best meet the needs of your users and their applications.

Topics include:

- Speed and distance limits of twisted-pair and fiber-optic cable.
- Cut-through switching and why every vendors switch has a latency of at least 40 microseconds.
- What is wrong with RIP and why is it so popular.
- The real differences between DLSw and RFC 1490 for IBM SNA over Frame Relay.
- Native ATM versus LAN emulation (LANE).
- Why some protocols, such as NetBIOS and NetWare are particularly challenging when moving from a LAN to a LAN/WAN enterprise network (and how to make them behave).

Vincent C. Jones is an independent consultant on network planning, design, and analysis of integrated local and wide area networks for cooperative, distributed processing in multivendor environments. Dr. Jones has over twenty five years of experience finding practical, cost-effective solutions to complex networking issues.

M9: System and Network Performance Tuning (New)

Hal Stern, *Sun Microsystems*

Who should attend: Novice and advanced UNIX system and network administrators, UNIX developers concerned about network performance impacts. You should have a basic understanding of the UNIX system facilities and network environments.

What you will learn: Procedures and techniques for tuning systems, networks, and application code.

More measurement and performance evaluation is being demanded of systems and networks than ever before. In this course you will learn about procedures and techniques for tuning systems, networks, and application code. Starting from the single system view, you will examine how the virtual memory system, the I/O system, and filesystem can be measured and optimized. The single host view will expand to include Network File System tuning and performance strategies.

A detailed treatment of networking performance problems, including network design and media choices, will lead to examples of network capacity planning. You'll learn about application issues, such as system call optimization, memory usage and monitoring, code profiling, real-time programming, and techniques for controlling response time. Many examples will be given, along with guidelines for capacity planning and customized monitoring, based on your workloads and traffic patterns.

Topics include:

- Performance Tuning Strategies Practical goals Monitoring intervals Useful statistics Tools, tools, tools
- Server Tuning Filesystem and disk tuning Memory consumption and swap space System resource monitoring
- NFS Performance Tuning NFS server constraints NFS client improvements NFS over WANs Automounter and other tricks
- Network Performance, Design and Capacity Planning Locating bottlenecks Demand management Media choices and protocols Network topologies: bridges, switches and routers Throughput and latency considerations Modeling resource usage
- Application Tuning System resource usage Memory allocation Code profiling Job scheduling and queuing Real-time issues Managing response time

Hal Stern is a Distinguished Systems Engineer with Sun Microsystems where he focuses on high-end server technology, operations management, networking, performance tuning, and information systems architecture. Hal has been a UNIX administrator for more than 10 years. He is the author of *Managing NFS & NIS* and several articles on application performance and network design.

M10: Inside the Linux 2.0 Kernel (New)

Stephen C. Tweedie, *Digital Equipment Corporation*

Who should attend: Application programmers and kernel developers. You should be reasonably familiar with C programming in the UNIX environment, but no prior experience of the UNIX or Linux kernel code will be assumed.

What you will learn: An introduction to the structure of the Linux kernel, the basic features it provides, and the most important algorithms it employs.

If you have always wanted to program in Linux or are considering switching to it because of its low cost, you will need to understand its kernel. The Linux kernel aims to achieve conformance with existing standards and compatibility with existing operating systems, however, it is not a reworking of existing UNIX kernel code. The Linux kernel was written from scratch to provide both standard and novel features, and take advantage of the best practice of existing UNIX kernel designs.

Topics will include:

- How the Linux kernel is organized: scheduler, virtual memory system, filesystem layers, device driver layers, and networking stacks.
 - The interface between each module and the rest of the kernel and the functionality provided by that interface.
 - The common kernel support functions and algorithms used by that module.
- How modules provide for multiple implementations of similar functionality (network protocols, filesystem types, device drivers, and architecture-specific machine interfaces).
- Basic ground rules of kernel programming such as races and deadlock conditions.
- Implementation of the most important kernel algorithms and their general properties (aspects of portability, performance and functionality).
- The main similarities and differences between Linux and traditional UNIX kernels, with attention to places where Linux implements significantly different algorithms.
- Details of the Linux scheduler, its VM system, and the ext2fs filesystem.
- The strict requirements for ensuring that kernel code is portable between the many architectures that Linux supports.

Stephen Tweedie works for Digital's Operating Systems Software Group and did his doctoral research on parallel multicomputer network performance at the University of Edinburgh. He has been contributing to Linux for a number of years, in particular designing some of the high-performance algorithms central to the ext2fs filesystem and the virtual memory code.

Tuesday, January 7

T1: UNIX Security Tools: Use and Comparison

Matt Bishop, *University of California at Davis*

Who should attend: UNIX security administrators and other interested users.

What you will learn: How to use publicly-available programs to improve the security of your system. You will compare the uses and drawbacks of several different programs, with an emphasis on when to use which.

Topics will include:

- Tool checking and analysis: what to look for, how to analyze a tool, checking downloaded tools for security problems.
- Tools for authentication: proactive password changers (e.g. npasswd, passwd+); password generation tools, challenge-response and one-time password techniques (e.g. S/key), password concealment and cracking tools (e.g. shadow, crack)
- Static analysis tools: Static system analysis (e.g. Tripwire, Tripwire), more general techniques (e.g. Brian CORRY)

- Static analysis tools: the system auditing (e.g. tripwire, omdaunt), more general analysis tools (e.g. ugen, COFF)
- Network analysis and security tools: monitors (e.g. tcp_wrapper), probers (e.g. strobe), NFS and NIS analysis and testing tools (e.g. nfsbug, nfswatch), ISS, SATAN, Gabriel, Courtney
- Tools for privilege: managing shells (e.g. osh, sudo), shared account management (e.g. lsu)
- Tools for logging and log analysis tools: (e.g. swatch, etc)
- Libraries: (e.g. securelib, msystem, safe_access, etc.)

Matt Bishops research and teaching areas at the University of California at Davis include computer and network security, along with operating systems and software engineering. He chaired the first two UNIX Security Workshops, and his column on computer security appears regularly in the Best Practises newsletter.

T2: CGI and WWW Programming in Perl (New)

Tom Christiansen, *Consultant*

Who should attend: Programmers with a light background in Perl and HTML. No previous CGI experience is required. If you dont have any Perl background, read the Llama book first or take the M1 tutorial on Monday. This is not a for non-programmers course nor a for guru programmers course. Its for accidental programmers, folks other than UNIX systems gurus who need to deal with CGI and WWW programming.

What you will learn: CGI and other WWW programming using Perl. Special attention is given to system security issues.

All aspects of writing and processing fill-out forms are covered using the standard CGI.pm module. Some attention is also given to parsing of HTML documents and writing spiderbots, automata that navigate the Web on their own.

Specific topics include:

- A light introduction to using Perls object-oriented class libraries
- Comparisons with other languages
- Setting up your server for CGI and SSI
- An overview of the CGI protocol and SSI CGI-related environment variables CGI without forms Debugging your CGI programs Using UNIX-domain sockets to serialize access to daemons Non-parsed headers scripts
- Data and system security Setuid execution and taint checking Avoiding the perils of shell escapes and backquotes The Oxford Safe CGI Perl environment Backgrounding long-running CGI scripts Non-parsed headers scripts Sending mail safely
- Sample problems and solutions Remote browser and user determination Generating dynamic forms, multistage (shopping cart) forms Credit-card algorithms File uploads Database access using flat text or HTML files, DBM files, and a full SQL database HTML parsing and link analysis
- Image maps
- Writing well-behaved robots

Tom Christiansen is a consultant specializing in Perl applications, optimizations, and training. He is a frequent instructor at USENIX and other conferences. He earned an MS degree in computer science from the University of Wisconsin at Madison.

T3: Security on the World Wide Web (New)

Daniel Geer, *OpenMarket, Inc.*, and Jon Rochlis, *BBN Planet*

Who should attend: Anyone responsible for running a Web site who wants to understand the tradeoffs in making it secure and how it is likely to be secured.

What you will learn: A comparison of available methods of Web security.

The World Wide Web is perhaps the most important enabler (so far) of electronic commerce. It has grabbed the popular imagination and the engineering and marketing efforts of a generation of on-line entrepreneurs and consumers. But it was initially designed with little thought to industrial strength security. Over the past several years numerous proposals have surfaced to secure the Web. This course will survey them with the goal of understanding the strengths and weaknesses of each.

The topics will include:

- Client/server network security
- Brief overview of encryption and its role in all security
- Simple schemes (Basic Auth)
- Prevailing protocols (SSL, S-HTTP, PCT)
- IP Security
- Payment protocols (Cybercash, DigiCash, Open Market, First Virtual, Visa/Mastercard, and others)
- Secure operation (configuration, containment, interaction with firewalls, replication, proxy servers, logging)

Daniel E. Geer, Jr. is director of engineering at Open Market, Inc. Formerly he was chief scientist, vice president of technology, and managing director of security consulting services for OpenVision Technologies. He is a consultant to major Wall Street financial institutions, and a frequent speaker at many technology conferences. He holds a Doctor of Science degree from Harvard.

Jon Rochlis is an engineering manager with BBN Planet where he leads groups developing managed connectivity and security services. Previously he was with OpenVision Technologies, responsible for systems and security management products.

T4: Creating Effective User Interfaces (New)

Joseph A. Konstan, *University of Minnesota*

Who should attend: People involved in the process of designing, implementing, evaluating, and managing the development of interactive graphical user interfaces. You need not have programming experience, but experience using interactive applications is required.

What you will learn: How to design a good user interface, and also how to constructively (and authoritatively) criticize a bad user interface.

With the advent of readily available, easy-to-use packages for building GUI-based systems, GUI-based systems have moved from the realm of specialty to almost mundane. But along with the widespread accessibility of GUI-builders has arisen a glut of poorly conceived user interfaces, which are hard to use, difficult to understand, or simply hard on the eyes.

You will get a broad introduction to user interface design and evaluation, introducing the task-centered interface design method, and covering principles behind UI design and techniques for UI evaluation. These techniques are pulled together at the end with a set of examples from visualization and Web page design.

Topics include:

- Foundations of user interfaces
- Task-centered approach to UI design
- Usability goals and measures
- Interface evaluation without users
- Interface evaluation with users
- Interactive programs: basics of event-driven programming
- Current trends: visualization and the Web

You will learn why it is a very, very bad thing and how to criticize evil name-brand software effectively!

Joseph A. Konstan is an award-winning teacher and directs a research program in user interfaces and multimedia systems. He regularly teaches short courses on user interface design and evaluation at the University of Minnesota.

T5: Java Applets and the AWT (New)

Nataraj Nagaratnam, *Syracuse Univ.*

Who should attend: Developers interested in developing interactive, animated GUI Java applications by exploring the capabilities of the Java Abstract Window Toolkit (AWT). Participants should be familiar with Java. Tutorial M3 will provide a good introduction.

What you will learn: You will become familiar with applet construction using the classes in the Java Abstract Window Toolkit (AWT) package, and the techniques to build a GUI in Java.

What you will learn: You will become familiar with applet construction using the classes in the java.awt package, learn the techniques to build a GUI in Java.

Topics include:

- Applet construction
- Structure of the Abstract Window Toolkit
- Principal AWT classes and methods
- Designing and implementing Java GUIs.

This course will include in-depth treatment of:

- Developing platform-independent GUIs.
- Details of laying out windowing components using the layout managers.
- Enhancing the applets by using the graphical classes.
- Event handling.
- Event propagation through peers.
- Hierarchical propagation through containment and inheritance.
- Using classes and methods for manipulating images which form a sub-package in the AWT.

Nataraj Nagaratnam is a PhD candidate in Computer Engineering at Syracuse University. He is the lead author of the upcoming book titled Waite Groups Java Networking and Windowing API SuperBible. He is a part of the Diamonds research group working in the areas of object-oriented languages and systems.

T6: Setting Up And Administering A Web Server (New)

Bryan Buus, *XOR Network Engineering*

Who should attend: Webmasters and administrators charged with creating a World Wide Web service for their company. You should have some knowledge of UNIX system administration.

What you will learn: An in-depth understanding of your server environment and the critical issues surrounding ongoing maintenance.

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

Setting up the Web server is only half of the battle. Understanding exactly how the protocol works, what performance issues are critical, what the security implications are, and other nuances are just some of the important issues that all webmasters need to thoroughly understand.

Topics include:

- The architecture of the Web
- The HTTP protocol
- Compiling the server
- Server configuration
 - Creating virtual hosts
 - Resource configuration
 - Access configuration
 - Per user access
- Analyzing and rotating logs
- Registering and announcing the server
- Electronic commerce issues
- Security and the Web
 - Operating system, CGI, and software considerations
 - Setting up and configuring SSL (Secure Sockets Layer)
 - Server performance issues
- Using multiple servers
- Detecting server problems

Bryan Buus is the manager of XOR Network Engineering's Web services group. Prior to joining XOR, Bryan kickstarted O'Reilly's & Associates online efforts in 1992. He is a co-author of *Managing Internet Information Services*, and has given seminars on managing Web services for CERFnet, the LISA Conference, and Hewlett Packard.

T7: Security for Software Developers: How to Write Code that Withstands Hostile Environments (New)

Marcus J. Ranum, *V-ONE Corp.*

Who should attend: System managers and software engineers developing client/server applications to be used over the Internet. A strong background in UNIX and UNIX programming is recommended. Many examples will refer to C programming constructs though familiarity with C is not a prerequisite.

What you will learn: Increasingly, client/ server software is being deployed in hostile environments that it may not have been designed to withstand. You will learn how to spot and avoid making typical flaws in security programming, using examples and case studies from existing applications.

Topics will include:

- Basics
 - Taxonomies of software and system flaws
 - Putting security at the right layer
 - Orange book (C2, B1, B2 systems)
 - Authentication versus authorization
- Data Protocols
 - How protocols are secure or insecure
 - Designing a protocol for security
 - Typical weaknesses of protocols
- Using cryptography
 - Basics (public key, secret key, certificates)
 - Randomness
 - Synchronizing protocols
 - What cryptography can and cant do for you
- Authentication
 - What to authenticate
 - Challenge/response
 - Authenticating packet streams
 - Publicly available authentication systems you can use
- Writing secure network daemons
 - Change root and setuid
 - Minimizing code
 - How to avoid doing everything as root
- Case studies
 - A simple file transfer daemon
 - Using file system permissions
 - Locking up a process

Marcus J. Ranum is a network and computer security consultant. He is the principal author of several major Internet firewall products, including the DEC SEAL, the TIS Gauntlet, and the TIS Internet Firewall Toolkit. Marcus has been managing UNIX systems and network security for over 13 years, including configuring and managing whitehouse.gov. Marcus is a popular lecturer and conference speaker on computer security topics.

T8: Solaris System Administration (New)

Marc Staveley, *Consultant*

Who should attend: System administrators who need to know the differences between SunOS 4.x and Solaris 2.x administration. Portions of this course will also be useful from a BSD to SysV.4 perspective. It will be most meaningful to system administrators who have some experience setting up and maintaining a network of SunOS 4.x workstations and servers.

What you will learn: New methods in Solaris to accomplish the same task as in SunOS (for example, the new NFS filesystem administration commands) and new features in Solaris (for example, the CacheFS filesystem). The course will concentrate on the Solaris 2.5 release.

Topics will include:

- Installation (packages, jumpstart, etc.)
- Booting and halting
- Kernel enhancements (dynamic loading, multi-threaded, layout on disk, /etc/system)
- Networking (NFS, AutoFS Automounter, PPP)
- CacheFS (including Cache Only Clients)
- NIS+ vs. NIS (YP)
- Volume manager (mounting CDs and floppies without root privileges)
- Service access facility (a getty replacement and much more)
- Printing (lpdvs, lpsched, SunSoft Print Client, Print Protocol Converter)
- Sun's Migration CD (making the move from SunOS to Solaris as painless as possible)

Marc Staveley has 14 years experience in UNIX application development and administration. An independent consultant, he is working with the Sun Microsystems Developer Support Centre assisting customers in migrating from SunOS to Solaris. He is a frequent speaker on the topics of standards-based development, multi-threaded programming, and system administration.

T9: IP version 6: An Introduction

Richard Stevens, *Consultant*

Who should attend: Network programmers and system administrators who will be converting applications and networks from IPv4 to IPv6, and implementors of IPv6. You should have a basic understanding of TCP/IP.

What you will learn: How to transition from IPv4 to IPv6 from the administration and programming standpoints.

Over the past few years, proposals have been made to replace IPv4 with a new version, mainly to overcome the addressing limitations of IPv4. In July 1994 the successor was chosen and named IPv6. Since that time numerous working groups have been busy completing the specifications for all facets of IPv6, and implementations are starting to appear. It is expected that vendor-supplied implementations of IPv6 will appear in the coming years and there will be a gradual transition of the Internet to IPv6.

You will get an overview of all aspects of IPv6, from the perspectives of a system administrator who needs to transition a network from pure-IPv4 hosts and routers to a mixture of IPv4 and IPv6 nodes and a programmer who needs to convert applications from IPv4 to IPv6.

Topics include DNS support, new socket address structure, address conversion functions, transition mechanisms, automatic tunneling, header fields and extension headers, source routing, path MTU discovery, upper-layer issues, ICMPv6, multicasting, neighbor discovery, CIDR, anycasting, and mobility.

W. Richard Stevens is an independent consultant and author of the books *TCP/IP Illustrated*, *Advanced Programming in the UNIX Environment*, and *UNIX Network Programming*.

T10: Writing Device Drivers Under Linux (New)

Theodore Tso, *Massachusetts Institute of Technology*

Who should attend: System programmers who need to write, modify, or maintain device drivers for the Linux operating system. You should know the C programming language. Some knowledge of general UNIX or Linux kernel design principals is desirable but not necessary.

Linux is becoming more and more popular, particularly because of the wide variety of device drivers which are available in the Linux kernel. You will learn about the several basic classes of Linux device drivers character devices, block devices, and network devices and the abstract device driver layers which simplify the task of writing tty devices and SCSI devices.

You will also hear about general kernel-level design and programming issues and the basic Linux kernel services needed by device drivers. Examples in the tutorial will be based on actual device drivers taken from the Linux 2.0 kernel and will include a tty-based device driver, a hard-disk device driver, and an Ethernet device driver.

Theodore Tso has been a Linux kernel developer since almost the very beginning of Linux. He implemented POSIX job control in the 0.10 Linux kernel. He is the maintainer and author for the Linux COM serial port driver, and the Control Rocketport driver. He also architected and implemented Linux's tty layer. Outside of the kernel, he is the maintainer of the e2fsck filesystem consistency checker. Theodore is the project manager for the Kerberos V5 development team at MIT. He participates in the Internet Engineering Task Force where he serves on the Security Area Directorate.

TECHNICAL PROGRAM
Wednesday-Friday, January 8-10, 1997

TECHNICAL SESSIONS

WEDNESDAY, JANUARY 8

9:00-10:30

Opening Remarks: John Kohl, Pure Atria Corporation

Keynote Address: Developing on "Internet Time"

James Gosling, Sun Microsystems

The development of the WWW and its attendant phenomena have led to the notion of Internet Time in which things seem to just happen faster. To avoid running into the sorts of nonsense results expressed by people such as Fred Brooks in *The Mythical Man Month*, you can't simply work faster you have to change the way you work. One such change has been the development and deployment of the Java language and the write once, run anywhere applications it enables. Such technological developments are not usually, by themselves, sufficient to accomplish the necessary changes often the skills and knowledge of the people using the technology have to change as well. This talk will discuss the changes in practice and perspective that such technologies have required of those creating and using Java and its associated tools.

James Gosling is a vice president and Fellow at Sun Microsystems where he has been the lead engineer for the Java/HotJava system. He has built satellite data acquisition systems, a multiprocessor version of UNIX, several compilers, mail systems, and window managers. He has also built a WYSIWYG text editor, a constraint-based drawing editor, and a text editor called Emacs for UNIX systems. Early in his career at Sun, he was lead engineer of the NeWS window system. James Gosling received a PhD in Computer Science from Carnegie-Mellon University.

REFEREED PAPERS

11:00-12:30: Performance I

Session Chair: Carl Staelin, Hewlett-Packard Laboratories

Embedded Inodes and Explicit Grouping: Exploiting Disk Bandwidth for Small Files

Gregory R. Ganger and M. Frans Kaashoek, *Massachusetts Institute of Technology*

Observing the Effects of Multi-Zone Disks

Rodney Van Meter, *Information Sciences Institute, University of Southern California*

A Revisitation of Kernel Synchronization Schemes

Christopher Small and Stephen Manley, *Harvard University*

2:00-3:30: Interface Tricks

Session Chair: Rob Gingell, Sun Microsystems

Porting UNIX to Windows NT

David G. Korn, *AT&T Research*

Protected Shared Libraries - A New Approach to Modularity and Sharing

Arindam Banerji, John M. Tracey, and David L. Cohn, *University of Notre Dame*

Extending the Operating System at the User-Level: the Ufo Global File System

Albert D. Alexandrov, Maximilian Ibel, Klaus E. Schauser, and Chris J. Scheiman, *University of California, Santa Barbara*

4:00-5:00: Client Tricks

Session Chair: Fred Douglass, AT&T Research

Network-aware Mobile Programs

Mudumbai Ranganathan, Anurag Acharya, Shamik Sharma, and Joel Saltz, *University of Maryland*

Using Smart Clients to Build Scalable Services

Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler, *University of California, Berkeley*

THURSDAY, JANUARY 9

9:00-10:30: Clustering

Session Chair: Clem Cole, Digital Equipment Corporation

Building Distributed Process Management on an Object-Oriented Framework

Ken Shirriff, *Sun Microsystems Laboratories*

Adaptive and Reliable Parallel Computing on Networks of Workstations

Robert D. Blumofe, *University of Texas, Austin*
Philip A. Lissiecki, *Massachusetts Institute of Technology*

A Distributed Shared Memory Facility for FreeBSD

Pedro A. Souto and Eugene W. Stark, *State University of New York, Stony Brook*

11:00-12:30: Tools

Session Chair: Matt Blaze, AT&T Research

Libcdt: A General and Efficient Container Data Type Library

Kiem-Phong Vo, *AT&T Research*

A Simple and Extensible Graphical Debugger

David R. Hanson and Jeffrey L. Korn, *Princeton University*

Cget, Cput, and Stage Safe File Transport Tools for the Internet

Bill Cheswick, *Bell Laboratories*

2:00-3:30: Works in Progress

Short, pithy and fun. WIP reports introduce new and ongoing work. If you have interesting work to share or a cool idea not ready for publication, a Works-In-Progress Report is for you. You'll get feedback from your fellow attendees. We are especially interested in the presentation of student work. To reserve a slot, send email to the WIP coordinator at wips@usenix.org. Topics are announced on-site.

4:00-5:30: Inferno

Rob Pike, Bell Labs

As telecommunications, entertainment, and computing networks merge, a wide variety of services will be offered on a diverse array of hardware, software, and networks. Inferno provides a uniform execution environment for applications and services in this chaotic world. Inferno comprises a networked operating system that can run native or above a commercial operating system, a virtual machine, a programming language, protocols, and standard interfaces for networks, graphics, and other system services. This architecture offers unprecedented portability for applications and services.

FRIDAY, JANUARY 10

9:00-10:30: User Tools

Session Chair: Charlie Briggs Digital Equipment Corp.

WebGlimpse - Combining browsing and searching

Udi Manber, Michael Smith, and Burra Gopal, *University of Arizona*

Mailing List Archive Tools

Sam Leffler and Melange Tortuba, *Silicon Graphics*

Experience with GroupLens: Making Usenet Useful Again

Bradley N. Miller, John T. Riedl, and Joseph A. Konstan, *University of Minnesota*

11:00-12:30: Performance II

Session Chair: Mike Karels, Berkeley Software Design

Overcoming Workstation Scheduling Problems in a Real-Time Audio Tool

Isidor Kouvelas and Vicky Hardman, *University College London*

On Designing Lightweight Threads for Substrate Software

Matthew Haines, *University of Wyoming*

High-Performance Local-Area Communication With Fast Sockets

Steven H. Rodrigues, Thomas E. Anderson, and David E. Culler, *University of California, Berkeley*

2:00-3:30: Caching and Stashing

Session Chair: Bill Bolosky, Microsoft Research

An Analytical Approach to File Defragmentation

An Analytical Approach to File Fetching
Hui Lei and Dan Duchamp, *Columbia University*

Optimistic Deltas for WWW Latency Reduction
Gaurav Banga, Fred Douglass, and Michael Rabinovich, *AT&T Research*

A Toolkit Approach to Partially Connected Operation
Dan Duchamp, *Columbia University*

4:15-5:45: Joint Closing Session

Severe Tire Damage's Stupid Mbone Tricks - A Lecture/Demonstration

Severe Tire Damage is the first band on the Internet, the first band on the Mbone, and hosts the first live video worldwide interactive multimedia show on the information superhighway. Members of STD will describe how it all works and show off their chops in a live show for USENIX.

INVITED TALKS

WEDNESDAY, JANUARY 8

11:00-12:30: Nomadicity and the IETF

Charles E. Perkins, IBM T.J. Watson Research Center

Laptop computers and the growth of wireless communications are driving a push towards nomadic computing and mobile networking. This talk will cover recent protocol developments at the IETF in areas that directly affect nomadic users, including mobile-IP, service location, IPv6, dynamic host configuration protocol for IPv4 and IPv6, dynamic updates to DNS, and ad-hoc networking. I'll attempt the dangerous task of describing these protocol developments as part of a coherent whole and offer some ideas about the future evolution of mobile networking.

2:00-3:30: If Cryptography Is So Great, Why Isn't It Used More?

Matt Blaze, AT&T Research

By separating the security of information from the security of the media over which it is transmitted and stored, cryptographic techniques seem ideal for protecting data on widely decentralized networks or data stored in files on insecure PCs and workstations. That's the theory, but in practice, cryptography is almost never available where it would be most useful. This talk will focus on some of the technical (rather than political) problems in integrating secure cryptography into real applications, operating systems, and networks.

4:00-5:00: The Inktomi Web Search Engine

Eric Brewer, University of California, Berkeley

This talk will describe the technology behind the Inktomi Web search engine. The Inktomi technology exploits parallel computing technology to build a high-speed, scalable Web server using commodity workstations. The prototype technology was developed as part of the Network of Workstations (NOW) project at the University of California at Berkeley and has also been used to build a new commercial search engine, HotBot. With 54 million documents, HotBot is the most complete Web index online.

THURSDAY, JANUARY 9

9:00-10:30: The AltaVista Web Search Engine

Louis Monier, Digital Equipment Corporation

AltaVista is the result of a research project started in the summer of 1995 at Digital's Research Laboratories in Palo Alto, California, that combined a fast Web crawler with scalable indexing software. Within three weeks of launch, the AltaVista site was handling over two million HTTP requests per day. By May 1996, the index had grown to more than 30,000,000 pages, and the site was receiving twelve million daily HTTP requests. This talk will describe the software and hardware technology behind this popular and successful Web search engine.

11:00-12:30: IPv6: The New Version of the Internet Protocol

Steve Deering, Xerox Palo Alto Research Center

A new version of the Internet's core protocol, IP, has been developed by the Internet Engineering Task Force (IETF) and is now entering the IETF Standards track. The new IP, known as IPv6, is designed to meet the scaling requirements imposed by the explosive growth of the Internet, and to meet the demand for greater functionality at the Internet layer. This talk will include a brief review of the motivations and events that led to the development of IPv6, a description of how the new protocol differs from the current version, and a status report on the IPv6 specifications, implementations, and transition mechanisms.

2:00-3:30: Highlights from 1996 USENIX Conferences and Workshops

This session will present highlights from some of USENIX's 1996 events: Conference on Object-Oriented Technologies and Systems (COOTS); LISA, the 10th Systems Administration Conference; the Second Symposium on Operating Systems Design and Implementation (OSDI); Sixth USENIX Security Symposium; and the Second USENIX Workshop on Electronic Commerce.

FRIDAY, JANUARY 10

9:00-10:30: Measuring Computer Systems: How to Tell the Truth with Numbers

Margo Seltzer and Aaron Brown, Harvard University

"Benchmarks shape a field (for better or worse); they are how we determine the value of change." -David Patterson, 1994.

If benchmarks shape a field, then computer science is in bad shape. Our field is characterized by a few examples of excellent benchmarks, and a large number of poorly conceived measurements. We will examine the current practice in computer system measurement, citing both surprisingly good and embarrassingly bad real-world examples that illustrate common benchmarking pitfalls. Through this we will offer some benchmarking tips and guidelines.

11:00-12:30: Stupid Net Tricks

Bill Cheswick, Bell Laboratories

The Net (born c. 1992) is a new Thing. Is it a highway? A library? The old analogies don't work very well.

This new Place is a rich source of new experiments, new tricks, and new troubles. It is subject to clever and annoying hacking tricks, an arms race that the hackers seem to be winning at the moment. They are even putting the network infrastructure at risk, though benign administrative problems may well beat them to the punch. Netnews has evolved horizontal and vertical spamming and cancelbots. The Web offers new publishing opportunities. Marketers can game search engines by, for example salting their pages with extra keywords, causing an arms spiral with the engine designers. Now Java designers would have us execute safe programs in unsafe containers. When all the security stupid tricks are fixed, denial-of-service attacks will remain.

2:00-3:30: Finding Bugs in Concurrent Programs

Gerard J. Holzmann, Bell Laboratories

In a concurrent system, machines can maliciously conspire to find loopholes in our otherwise perfect code, causing irreproducible forms of deadlock or failure to perform the desired function. Since software now controls just about everything of significance, it is important to discover these types of bugs well before the software is used.

SPIN is a popular verification system for concurrent software. In this talk Ill show how SPIN works, and give some examples of some of the more remarkable applications.

USELINUX PROGRAM

USELINUX DEVELOPERS

WEDNESDAY, JANUARY 8

9:00-10:30: Linux: What It Is and Why It Is Significant

Mark Bolzern, *Work Group Solutions*
Tom Miller, *X Engineering Software Systems*

Bill Gates said at its introduction that Windows NT will be a better UNIX than UNIX. We say, Why wait, Linux already is! We will demonstrate that Linux is not only significant, but the next generation heir of UNIX and what Linux will be in the near future and why. There will be case histories from a user perspective on Why Linux.

11:00-12:30: The Sparc Port of Linux

David S. Miller, *Rutgers CAIP*
Miguel de Icaza, *Instituto de Ciencias Nucleares, Ciudad Universitaria, Universidad Nacional Autonoma de Mexico*

Regardless of the many barriers the port of Linux to Sparc-based workstations and servers had to overcome, it was very successful. This talk will emphasize performance, stability, and how Linux became more portable as a result of the port.

2:00-3:30: Advanced Device Drivers

Alessandro Rubini, *Universita di Pavia*

This presentation will go slightly deeper than the usual overview of the programming interface for Linux device drivers. Youll learn details of asynchronous events (select and fasync), as well as using kernel timers and task queues. Youll hear about kernel threads and how they can help you design unusual device driver features. The kmouse virtual device will be used as sample code showing how to take advantage of such software technologies.

4:00-5:00: Future of the Linux Kernel

Linus Torvalds, *Helsinki University*

The future of Linux! The lead developer of Linux talks about the future of Linux kernel development, including topics such as advanced memory management techniques, high-speed networking, and portability.

THURSDAY, JANUARY 9

9:00-10:30: Real Time

Victor Yodaiken and Michael Barabanov, *New Mexico Institute of Technology*

Real-time Linux is a preemptable version of the Linux kernel that can run tasks with hard real-time deadlines. Our primary goal is to make it convenient to use a PC to control lab equipment, robotics, or other devices that need precise timing and limited response times. This talk will focus on how to use real-time Linux, how to write tasks as loadable kernel modules, and how to adjust the scheduler to different problems.

11:00-12:30 (Shared Session): /proc

Stephen Tweedie, *Digital Equipment Corporation*

Linux provides several interfaces for programs to access system data. This talk explains how system administrators and integrators can use the two main interfaces the /proc filesystem and the "sysctl" interface to tune system performance.

11:00-12:30 (Shared Session): The Pluggable Authentication Modules (PAM) Framework

Ted Tso, *Massachusetts Institute of Technology*

Authentication systems are constantly evolving, and it is important to shield the system administrator and application programmers from such changes. The Pluggable Authentication Module (PAM) framework allows a system administrator to dynamically change the authentication schemes used by programs such as login, ftpd and passwd. This talk will discuss how the PAM system uses dynamically linked shared libraries to provide pluggability for the functions of system authentication and account, session, and password management.

2:00-3:30: Standards

Heiko Eissfeldt, *Unifix Software*

An introduction to POSIX.1-related standards, starting with a survey of the evolution of the most relevant related standards ANSI C, XPG4, and the POSIX.1* extensions followed by a closer look at POSIX.1. A standard can be thought of as a protocol defining an interface between a system and applications. I will primarily focus on the perspective of application programs, but also discuss some POSIX.1 implementation issues.

4:00-5:30: Connecting Legacy and Open Systems

Michael Callahan, *Stelias Computing, Inc.*

While Linux's abilities as an Internet platform are well-known, it also makes a capable server for microcomputer LANs running legacy protocols. This talk will describe how to use Linux in Microsoft, Apple, and Novell networks in a way which complements its capabilities as a TCP/IP server.

USELINUX BUSINESS

FRIDAY, JANUARY 10

9:00-9:30: Linux: What It Is and Why It Is Significant

Mark Bolzern, *Work Group Solutions*
Tom Miller, *X Engineering Software Systems*

Linux is a freely-distributable operating system being implemented by an international team of highly-skilled programmers over the Internet. The history and philosophy of on-going development used in Linux guarantee a platform on which you can grow your business.

This talk will demonstrate the philosophical and concrete reasons why Linux is becoming a force in the commercial marketplace.

9:30-10:30: Linux and Distribution Channels: Ways to Enter the Commercial Market

Don Rosenberg, *Stromian Technologies*

While the Linux operating system and improvements to it will remain free, a commercial marketplace is growing up around the distribution of the system with added-value installation aids, utilities, and help and maintenance. With the arrival of full-featured applications, Linux developers and vendors are entering a growing market with developing channels: direct sales, catalogs, OEM, integrators, resellers, distributors, and VARs. Learn the benefits and drawbacks of these channels, and how to enter them.

11:00-12:30: Using Linux in Your Business: A Business Justification

Presented by Linux International

How and why to justify to your or your customers management the use of Linux as an operating system. Pitfalls to avoid, arguments to use.

After this session, you will know where to go to get software and hardware support and applications. You will also see a price comparison between systems using Linux and systems using proprietary

operating system code.

This talk is for the end user who is trying to convince their management to use Linux, or for the OEM, VAR, or reseller trying to convince their customer that Linux is the operating system to use.

2:00-4:00: The Linux Market: Who, What, Where, When and Why?

Presented by Linux International

Everything that a marketing manager needs to put together a marketing plan utilizing Linux as a base operating system. Whether you are the marketing manager for an ISV, a VAR, or a reseller, you will walk away with a skeleton marketing plan that can easily be adapted to fit your product or service. NOTE: This talk is not directed at the end user for Linux, but may contain useful information for them also.

Included in the plan will be: current status, market size and growth, market demographics and customer profiles; status of ports to different architectures; different distributions and their market focus; vertical vs. horizontal markets; differentiations between Linux and UNIX systems; applications currently available on Linux; pricing your Linux application or service for maximum revenue, and much, much more.

USELINUX PRESENTATIONS AND CASE STUDIES DESCRIPTIONS

FIRST attend these evening sessions, THEN attend the business conferences on Friday to develop your business plan. Nothing gives you a better sense of how things really work than to hear about real-life experiences. Attend these evening sessions to learn about specific applications and uses.

WEDNESDAY, JANUARY 8, 7:30pm - 11:00pm

The Use of Linux for Dedicated Systems

Chel van Gennip, *HISCOM BV*

Using a distributed environment with dedicated servers for well-defined tasks enables us to reduce the costs of administration and maintenance if the task is well isolated. Find out how using an inexpensive OS such as Linux or inexpensive hardware will enable you to isolate tasks in a cost-effective manner.

Perceptions: A Strategic Deployment of Linux in the Health Care Environment

Greg Wettstein, *Velocity LLC*

Is your company one of the many seeking to leverage the power of internetworking tools and protocols through the deployment of corporate Intranets? The ability of Linux to effectively platform applications in this marketplace is the focus of this talk. You will hear about the issues involved when Linux is selected as a deployment base. You will learn about the advantages of selecting Linux and the corporate challenges experienced when freely distributable tool sets are chosen.

The Future of the Linux Desktop

Ken Apa, *Governors State University*; Jim Fetters, *Chicago Mercantile Exchange*; Joe Sloan, *Toyota Motor Sales USA*

An overview of present desktop and user-interface requirements for the 90s and beyond. Join a discussion of file managers, Motif, Tk and ease-of-use issues, as well as a comparison with other directions for desktops and how Linux can meet the challenge. Integrating Linux with other systems such as NT, the Web, and the 3D graphics capabilities of Linux will also be explored.

The Classroom of the Future

Karl Jeacle, *Broadcom Eireann Research Ltd.*

The Linux operating system has been used as the basis for a low cost computing solution connecting primary (K-12) and secondary (junior/senior high) schools in Ireland to the Internet. A pilot network has been set up between six schools in the Dublin area, three of which have been interconnected using ISDN. Each of these three schools have used Linux to provide services such as Email, Usenet, IRC and the World Wide Web to both students and staff.

Learn how this was done, and explore the possibilities of doing this in your own community.

THURSDAY, JANUARY 9, 6:00pm - 10:30pm

Using GNUstep to Deploy User Applications

Scott Christley, *NET-Community*

GNUstep is a set of free software libraries and applications that can be used to create portable, graphical user applications. GNUstep is known to run on many versions of UNIX and Windows NT/95. You will hear historical information, a technical overview, the current state of development, future development, and available third party software and support organizations.

Embedded, Turnkey and Real Time

Phil Hughes, *Linux Journal*

Linux is well established as a good development platform and is now coming into its own as a base system for business applications. Linux can also excel as an environment for embedded systems from small communications controllers and routers through Raster Image Processors and workgroup servers.

You will have a chance to hear how Linux is being used in these areas, possible problems and its potential.

Developing Linux-based Electronic Markets for Internet Trading Experiments

Paul J. Brewer, *Georgia State University*

Linux is currently playing a very important role in the development of Internet-accessible, world-wide, real-time electronic markets for use in laboratory research experiments. These markets are full-featured electronic markets without the high-tech security and encryption functions required for handling high value transactions. Learn how using Linux as a central server enabled us to make architectural decisions not possible under other operating systems.

Linux in the Radio Amateur and Electronic Engineering Market

Bruce Perens, *Pixar*

Linux is currently enjoying growing popularity in the electronic and radio amateur market. Technical organizations are centering on using Linux as well as the large market for electronic hobbyists. Learn the size of these markets, what their needs are, how to reach them.

USELINUX Program Committee

Conference Chair: Michael Johnson, *Red Hat Software*

Technical Track Committee:

Michael K. Johnson, *Chair, Red Hat Software*

Mark Bolzern, *WorkGroup Solutions*

Alan Cox, *3Com, Remote Access Products*

Jon maddog Hall, Esq., *Digital Equipment Corporation*

Lorrie LeJeune, *O'Reilly and Associates*

Dr. Tom Miller, *North Carolina State University*

Erik Troan, *Red Hat Software*

Business Track Committee:

Jon maddog Hall, Esq., *Chair, Digital Equipment Corporation*

Jonathan Eumice, *President, Founder, Research Director, Illuminata, Inc.*

Michael K. Johnson, *Red Hat Software*

Lorrie LeJeune, *Product Manager of Internet and Linux, O'Reilly and Associates*

Bryan Sparks, *President, Caldera, Inc.*

Paul Wimbauer, *Director of Technical Programs, Avnet Computing*

Bob Young, *President, Red Hat Software*

Don't bring, I presume, your own software

ABOUT LINUX INTERNATIONAL

Linux International exists to promote the use of Linux. The volunteers who created Linux are now joined by volunteers who specialize in promoting it. We know how good Linux is, and want it to become an accepted alternative to products from even the largest computer companies. We are made up of individuals, member companies, and sponsoring member companies who all wish to help promote the use of Linux.

We are focused on these areas at the moment:

- The Linux Development Fund Linux International has set up a worldwide Development Fund, publicly and accountably controlled by three prominent members of the Linux community. The goal of the Development Fund is simply to collect and distribute money to developers to help them in their work on Linux. The money is used to pay for things such as manuals, hardware, journal subscriptions and so on. Full details (and a donation form!) are available in the Development Fund document.
Email: donations-info@li.org.
- Promoting Linux To Companies and Individuals.
It is vital that more software and hardware developers be brought into Linux. We are campaigning to alert them to the maturity of Linux and the money-making potential it has for them. The media coverage we strive to get appeals not only to commercial enterprises but also to individuals.

So now you know about Linux International. The rest is up to you. To volunteer some skill you have, make a cash donation, or even pass on some still-useful piece of hardware, please contact us at our website:

[Linux International](#)

VENDOR EXHIBITION

Wednesday, January 8, Noon - 7:00pm
Thursday, January 9 10:00am - 4:00pm

"Two days of exposure to the cream of the UNIX User Community."
-Neil Groundwater, Enterprise Management Group, SunSoft, Inc.

The emphasis is on serious questions and feedback at the USENIX 97 Exhibition. In the relaxed environment, attendees have time for in-depth discussions with technical representatives. You will find the products and services of some 60 vendors on display.

If you cannot make it to the conference but would like to visit the exhibition, please contact Cynthia Deno, Exhibit Coordinator, at 408.335.9445 or cynthia@usenix.org.

CONFERENCE ACTIVITIES

Schedule a BoF! Talk to an expert! Present new work!! Don't miss these special activities, designed to maximize your time at the conference.

BoFs

Birds-of-a-Feather Sessions (BoFs)

Tuesday, Wednesday, and Thursday evenings

Do you have a topic that you'd like to discuss with others? Birds-of-a-Feather sessions may be perfect for you. BoFs are interactive, informal gatherings for attendees interested in a particular topic. Schedule your BoF in advance. Call the Conference Office at 714.588.8649 or send email to conference@usenix.org. Topics are announced at the conference. BoFs may also be scheduled on-site.

The Guru is IN

Have a question that's been bothering you? Try asking a USENIX guru! Noted experts from the USENIX community will be available to spark controversy and answer questions. Please contact the Invited Talks Coordinators via email to IT@usenix@usenix.org if you would like to volunteer your expertise.

Works-in-Progress Reports

Short, pithy, and fun, Works-in-Progress Reports (WIPs) introduce interesting new or ongoing work. If you have work to share or a cool idea not quite ready to be published, a WIP Report is for you! You will receive insightful feedback. We are particularly interested in presenting student work. WIPs are scheduled within the technical sessions program. To reserve a slot, send email to wips@usenix.org. Topics are announced on-site.

Social Activities

Welcome Reception and Kickoff, Sunday, January 5, 6:00pm - 9:00pm

The Welcome Reception offers you a chance to say hello over soft drinks and snacks. The Kickoff introduces attendees to conference events and to Anaheim. The Kickoff follows the Welcome Reception at 8:00pm.

Soft Drinks and Snacks

Enjoy light food and beverages Tuesday-Thursday evenings in the Foyer.

CONFERENCE SERVICES

Terminal Room

Internet and dial-out access are provided in the Terminal Room. The Terminal Room will be open throughout the conference week. Look for details posted to comp.org.usenix.

Attendee Message Service

Electronic message service will be available Monday, January 6 through Friday, January 10. Electronic messages to conference attendees should be addressed:

first_lastname@conference.usenix.org. Telephone messages may be left by telephoning the Marriott Hotel at 714.750.8000 and asking for the USENIX Message Desk extension. The Message Desk will be open Sunday, January 5, 4:00 pm - 9:00 pm, and during conference hours until Friday, January 10 at 3:00 pm.

USENIX Membership Information

About USENIX

USENIX is the UNIX and Advanced Computing Systems Technical and Professional Association.

Since 1975 the USENIX Association has brought together the community of engineers, system administrators, scientists, and technicians working on the cutting edge of the computing world.

The USENIX Conferences have become the essential meeting grounds for the presentation and discussion of the most advanced information on new developments in all aspects of advanced computing systems.

The USENIX Association and its members are dedicated to:

- problem-solving with a practical bias,
- fostering innovation and research that works,
- communicating rapidly the results of both research and innovation,
- providing a neutral forum for the exercise of critical thought and the airing of technical issues.

SAGE, a Special Technical Group within the USENIX Association, is dedicated to the recognition and advancement of system administration as a profession. To join SAGE, you must be a member of USENIX.

Join USENIX

You can join USENIX or renew your membership, and attend the conference for the same low price. Just check the box on the registration form. \$70 of your fees will be allocated to membership, and you can take advantage of all our member benefits.

If you are a system administrator and would like to join SAGE or renew your membership, just check the box on the registration form. For only \$25, you can be a part of the only organization devoted to system administrators.

For more information about USENIX and SAGE, please contact:

The USENIX Association
2560 Ninth St, Ste. 215
Berkeley, CA 94710
Phone: 510.528.8649
Fax: 510.548.5738
Email: office@usenix.org

HOTEL AND TRAVEL INFORMATION:

Hotel Discount Reservation Deadline: Friday, December 20, 1996

USENIX has negotiated special rates for conference attendees at the Anaheim Marriott. Contact the hotel directly to make your reservation. You must mention USENIX to get the special rate. A one-night room deposit must be guaranteed to a major credit card. To cancel your reservation, you must notify the hotel at least 24 hours before your planned arrival date.

Anaheim Marriott
700 West Convention Way
Anaheim, CA 92802
Toll Free: 800.228.9290
Phone: 714.750.8000
Reservation Fax: 714.750.9100

Room Rates: \$107/Single, \$117/Double (plus local taxes, currently at 15%)

Special Note: This conference places a heavy demand on meeting space. To get meeting space and other services free and keep your conference fees low, USENIX guarantees to use a number of sleeping rooms. Contracts are signed long in advance. The penalty for not meeting the guarantee may exceed \$100,000. You must mention USENIX when reserving your room to ensure that it counts against our room guarantee. If you use a corporate rate, it will not count against our commitment.

Need a Roommate?

Usenet facilitates room sharing. If you wish to share a room, post to and check comp.org.usenix.roomshare.

Discount Airfares and Car Rentals

Special discounted air fares and car rentals are available only through JNR, Inc., a full service travel agency. All restrictions apply. Please call JNR for details. Call toll free 800.343.4546 in the USA and Canada or telephone 714.476.2788.

Transportation to the Hotel

The Anaheim Marriott is located 31 miles, about 50 minutes, from the Los Angeles International Airport and 16 miles or 25 minutes from the John Wayne Airport-Orange County.

Shuttle Service Super

Shuttle offers transportation to and from both LAX and John Wayne Airports. Advanced reservations are required for John Wayne pick ups and to avoid delays at LAX. Call 714.517.6600. One way fare from LAX is \$13 and \$10 from John Wayne.

Taxis are available, and cost approximately \$65 one way from LAX and \$28 from John Wayne.

What To Do in Orange County

Anaheim may be best known as the home of Disneyland (just steps from the hotel), but it is also well located for exploring some of the hot spots of Southern California. Most of these are within an hours drive of Anaheim. Southern California is car country. You may want to rent a car since attractions and restaurants are spread out and public transportation is not convenient. When in Rome

Disneyland: The original Disneyland is just steps from the hotel. Besides Disneyland, there's Knotts Berry Farm and Universal Studios.

Beaches: Orange County has a 42 mile coastline, comprised of public and state beaches.

Museums/Galleries: Orange County has over 300 museums and galleries. LA offers the County Museum and the Getty.

REGISTRATION INFORMATION

Cancellation Policy If you must cancel, all refund requests must be in writing, and postmarked no later than December 27, 1996. Telephone cancellations cannot be accepted. You may substitute another in your place. **For more information, 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 Hours: M-F, 8:30am-5:00pm Pacific Time

REGISTRATION FORM

Note: For Your Convenience--the registration form is provided separately for easier downloading and printing. Simply double click on "REGISTRATION FORM" above to access a PostScript form of the document.

Statewide Illinois Library Catalog

UNIV OF ILLINOIS
[Ask A Librarian](#)

Libraries that Own Item

• This screen shows libraries that own the item you selected.

Home

Databases

Searching

Results

[Staff View](#) | [My Account](#) | [Options](#) | [Comments](#) | [Exit](#) | [Hide tips](#)

List of Records | Detailed Record | Marked Records | Saved Records | Go to page ▼

E-mail
 Print
 Return
 Help

Current database: **WorldCat** Total Libraries: 15

Title: Proceedings of the USENIX 1997 annual technical conference : January 6-10, 1997, Anaheim, California, USA **Author:** USENIX Association **Accession Number:** 37971838

Libraries with Item: "Proceedings of the USENIX..." ([Record for Item](#) | [Get This Item](#))

Location	Library	Local Holdings	Code
US,MA	MASSACHUSETTS INST OF TECH		MYG
US,MD	JOHNS HOPKINS UNIV		JHE
US,NJ	RUTGERS UNIV		NJR
US,NY	UNIV OF ROCHESTER		RRR
US,OH	BAKER & TAYLOR		BAKER
US,OH	UNIV OF CINCINNATI		CIN
US,WA	MICROSOFT CORP, INFO SERV		MSF
China	SHANGHAI LIBR		SLY
Germany	MPIF INFORMATIK		DEPIN
Hong Kong	CHINESE UNIV OF HONG KONG		CVU
Kenya	US INT UNIV-AFRICA		V7O
Netherlands	BIBLIOTHEEK UNIVERSITEIT VAN AMSTERDAM		Q GK
Netherlands	CENTRUM VOOR WISKUNDE EN INFORMATICA		NLCVW
Netherlands	DELFT UNIV OF TECHNOL	Local Holdings Availa...	NLTUD
United Kingdom	BRITISH LIBR		BRI

Record for Item: "Proceedings of the USENIX..." ([Libraries with Item](#))

[GET THIS ITEM](#)

Availability: **Check the catalogs in your library.**

- [Libraries worldwide that own item:](#) 15
- [Search the catalog at the Library of University of Illinois at Urbana-Champaign](#)

External Resources:

- [Discover full text](#) [Discover UIUC Full Text](#)
- [Interlibrary Loan Request](#)
- [Cite This Item](#)

[FIND RELATED](#)

More Like This: [Advanced options...](#)

Find Items About: [USENIX Association](#), (max: 2)

Title: Proceedings of the **USENIX 1997 annual technical conference : January 6-10, 1997, Anaheim, California, USA /**

Corp Author(s): [USENIX Association](#), [Technical Conference](#) (1997 : Anaheim, Calif.); [USENIX Association](#).

Publication: Berkeley, CA : The Association,

Year: 1997

Description: vi, 318 pages : illustrations ; 28 cm

Language: English

Standard No: **ISBN:** 1880446847; 9781880446843

SUBJECT(S)

Descriptor: [Operating systems \(Computers\) -- Congresses](#),
[Operating systems \(Computers\)](#),
[Besturingssystemen](#).

Genre/Form: [Conference papers and proceedings](#).

Title Subject: [UNIX \(Computer file\) -- Congresses](#),
[UNIX \(Computer file\)](#)

Note(s): Includes bibliographical references and index.

Class Descriptors: **LC:** [QA76.76.O63](#)

Other Titles: [Conference proceedings](#), [USENIX 1997 annual technical conference](#), Anaheim, California, January 6-10, 1997; [USENIX 1997 annual technical conference proceedings](#); 1997 [USENIX technical conference](#), January 6-10, 1997, Anaheim, California

Responsibility: the [USENIX Association](#).

Vendor Info: Baker & Taylor (BKTY) 40.00 **Status:** active **Note:** B&T Title: [Usenix 1997 Annual Technical Conference](#)

Material Type: Conference publication (cnp)

Document Type: Book

Entry: 19960301

Update: 20150801

Accession No: **OCLC:** 37971838

Database: WorldCat

E-mail
 Print
 Return
 Help

Current database: **WorldCat** Total Libraries: 15

Using smart clients to build scalable services

Authors: [Chad Yoshikawa](#) [Computer Science Division, University of California, Berkeley, CA](#)
[Brent Chun](#) [Computer Science Division, University of California, Berkeley, CA](#)
[Paul Eastham](#) [Computer Science Division, University of California, Berkeley, CA](#)
[Amin Vahdat](#) [Computer Science Division, University of California, Berkeley, CA](#)
[Thomas Anderson](#) [Computer Science Division, University of California, Berkeley, CA](#)
[David Culler](#) [Computer Science Division, University of California, Berkeley, CA](#)



1997 Article

Bibliometrics

Downloads (6 Weeks): 0
 Downloads (12 Months): 0
 Downloads (cumulative): 0
 Citation Count: 51

Published in:

· Proceeding
 ATEC '97 Proceedings of the annual conference on USENIX Annual Technical Conference
 Pages 8-8
 USENIX Association Berkeley, CA, USA ©1997
[Table of contents](#)

Tools and Resources

Save to Binder

Export Formats:

[BibTeX](#) [EndNote](#) [ACM Ref](#)[Publisher Site](#)

Share:

Author Tags ▾



Recent authors with related interests ▾

Concepts in this article ▾

powered by
IBM Watson™
 Contact Us | Switch to [single page view](#) (no tabs)

[Abstract](#) [Authors](#) [References](#) [Cited By](#) [Index Terms](#) [Publication](#) [Reviews](#) [Comments](#) [Table of Contents](#)

Individual machines are no longer sufficient to handle the offered load to many Internet sites. To use multiple machines for scalable performance, load balancing, fault transparency, and backward compatibility with URL naming must be addressed. A number of approaches have been developed to provide transparent access to multi-server Internet services including HTTP redirect, DNS aliasing, Magic Routers, and Active Networks. Recently however, portable Java code and lightly loaded client machines allow the migration of certain service functionality onto the client. In this paper, we argue that in many instances, a client-side approach to providing transparent access to Internet services provides increased flexibility and performance over the existing solutions. We describe the design and implementation of Smart Clients and show how our system can be used to provide transparent access to scalable and/or highly available network services, including prototypes for: telnet, FTP, and an Internet chat application.

Powered by **THE ACM GUIDE TO COMPUTING LITERATURE**

The ACM Digital Library is published by the Association for Computing Machinery. Copyright © 2016 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads: [Adobe Reader](#) [QuickTime](#) [Windows Media Player](#) [Real Player](#)

Using smart clients to build scalable services

Authors: Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, David Culler



1997 Article

Bibliometrics: Downloads (6 Weeks): 0, Downloads (12 Months): 0, Downloads (cumulative): 0, Citation Count: 51

Published in: ATEC '97 Proceedings of the annual conference on USENIX Annual Technical Conference, Pages 8-8, USENIX Association Berkeley, CA, USA ©1997

Tools and Resources

- Save to Binder, Export Formats: BibTeX, EndNote, ACM Ref, Publisher Site, Share: Facebook, Twitter, LinkedIn, etc.

Author Tags

Recent authors with related interests, Concepts in this article, powered by IBM Watson

Contact Us | Switch to single page view (no tabs)

- Abstract, Authors, References, Cited By, Index Terms, Publication, Reviews, Comments, Table of Contents

51 Citations

List of 51 citations including: Valeria Cardellini, Michele Colajanni, Philip S. Yu, Dynamic Load Balancing on Web-Server Systems, IEEE Internet Computing, v.3 n.3, p.28-39, May 1999; Mon-Yen Luo, Chu-Sing Yang, System support for scalable, reliable and highly manageable web hosting service, Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems, p.18-18, March 26-28, 2001, San Francisco, California; Eric A. Brewer, Lessons from Giant-Scale Services, IEEE Internet Computing, v.5 n.4, p.46-55, July 2001; Byoung-Dai Lee, Jon B. Weissman, An Adaptive Service Grid Architecture Using Dynamic Replica Management, Proceedings of the Second International Workshop on Grid Computing, p.63-74, November 12, 2001; Geunyoung Park, Boncheol Gu, Junyoung Heo, Sangho Yi, Jungkyu Han, Jaemin Park, Hong Min, Xuefeng Piao, Yookun Cho, Chang Won Park, Ha Joong Chung, Bongkyu Lee, Sangjun Lee, Adaptive load balancing mechanism for server cluster, Proceedings of the 2006 international conference on Computational Science and Its Applications, May 08-11, 2006, Glasgow, UK; Chu-Sing Yang, Mon-Yen Luo, Building an Adaptable, Fault Tolerant, and Highly Manageable Web Server on Clusters of Non-Dedicated Workstations, Proceedings of the Proceedings of the 2000 International Conference on Parallel Processing, p.413, August 21-24, 2000; Chu-Sing Yang, Mon-Yen Luo, Realizing fault resilience in Web-server cluster, Proceedings of the 2000 ACM/IEEE conference on Supercomputing, p.21-es, November 04-10, 2000, Dallas, Texas, USA; Mon-Yen Luo, Chu-Sing Yang, Enabling fault resilience for web services, Computer Communications, v.25 n.3, p.198-209, February, 2002; Sofiane Mounine Hemam, Ouassila Hioual, Load balancing by requests redistribution in failure nodes context, Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication, p.1-5, November 23-25, 2015, Batna, Algeria; Chu-Sing Yang, Mon-Yen Luo, Efficient support for content-based routing in web server clusters, Proceedings of the 2nd conference on USENIX Symposium on Internet Technologies and Systems, p.20-20, October 11-14, 1999, Boulder, Colorado; Nisha Talagala, Satoshi Asami, David Patterson, Bob Futernick, Dakin Hart, The Art of Massive Storage: A Web Image Archive, Computer, v.33 n.11, p.22-28, November 2000; Jongbae Moon, Yongyoon Cho, A high-availability webserver cluster using multiple front-ends, Proceedings of the First international conference on Computational and Information Science, December 16-18, 2004, Shanghai, China; Katia Obraczka, Grig Gheorghiu, The performance of a service for network-aware applications, Proceedings of the SIGMETRICS symposium on Parallel and distributed tools, p.81-91, August 03-04, 1998, Welches, Oregon, USA; Dheeraj Sanghi, Pankaj Jalote, Puneet Agarwal, Using Proximity Information for Load Balancing in Geographically Distributed Web Server Systems, Proceedings of the First EurAsian Conference on Information and Communication Technology, p.659-666, October 29-31, 2002; Kimmo Kaario, Timo Hämäläinen, Jian Zhang, Tuning of QoS Aware Load Balancing Algorithm (QoS-LB) for Highly Loaded Server Clusters, Proceedings of the First International Conference on Networking-Part 2, p.250-258, July 09-13, 2001; Sewook Wee, Huan Liu, Client-side load balancer using cloud, Proceedings of the 2010 ACM Symposium on Applied Computing, March 22-26, 2010, Sierre, Switzerland; Li Xiao, Xiaodong Zhang, Zhichen Xu, On Reliable and Scalable Peer-to-Peer Web Document Sharing, Proceedings of the 16th International Parallel and Distributed Processing Symposium, p.228, April 15-19, 2002; Colin Allison, Martin Bramley, Jose Serrano, David McKechn, Replicating the R in URL, Proceedings of the 8th Euromicro conference on Parallel and distributed processing, January 19-21, 2000, Rhodes, Greece; Jiannong Cao, Yudong Sun, Xianbin Wang, Sajal K. Das, Scalable load balancing on distributed web servers using mobile agents, Journal of Parallel and Distributed Computing, v.63 n.10, p.996-1005, October 2003; Mehmet Karaul, Yannis A. Korlis, Ariel Orda, A market-based architecture for management of geographically dispersed, replicated Web servers, Proceedings of the first international conference on Information and computation economies, p.158-165, October 25-28, 1998, Charleston, South Carolina, USA; Girma Berhe, Lionel Brunie, Jean-Marc Pierson, Modeling service-based multimedia content adaptation in pervasive computing, Proceedings of the 1st conference on Computing frontiers, April 14-16, 2004, Ischia, Italy; Rainer Koster, Thorsten Kramp, Structuring QoS-supporting services with smart proxies, IFIP/ACM International Conference on Distributed systems platforms, p.273-288, April 03-07, 2000, New York, New York, USA




-  [Vivek S. Pai, Mohit Aron, Gaurov Banga, Michael Svendsen, Peter Druschel, Willy Zwaenepoel, Erich Nahum, Locality-aware request distribution in cluster-based network servers, ACM SIGPLAN Notices, v.33 n.11, p.205-216, Nov. 1998](#)
- [Xiaonan Ma, A. L. Narasimha Reddy, SPIRAL: A Client-Transparent Third-Party Transfer Scheme for Network Attached Disks, Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies \(MSS'03\), p.254, April 07-10, 2003](#)
- [Jiannong Cao, Alvin T. S. Chan, Weidong Cao, Cassidy Yeung, Virtual Programming Lab for Online Distance Learning, Proceedings of the First International Conference on Advances in Web-Based Learning, p.216-227, August 17-19, 2002](#)
-  [Mohit Aron, Peter Druschel, Willy Zwaenepoel, Cluster reserves: a mechanism for resource management in cluster-based network servers, ACM SIGMETRICS Performance Evaluation Review, v.28 n.1, p.90-101, June 2000](#)
- [Stephen S. Yau, Gaurav Goyal, Yisheng Yao, Replication for Adaptive Responsiveness in Service-Oriented Systems, Proceedings of the Fifth International Conference on Quality Software, p.161-168, September 19-20, 2005](#)
- [Nabor C. Mendonça, José Ailton F. Silva, Ricardo O. Anido, Client-side selection of replicated web services: An empirical assessment, Journal of Systems and Software, v.81 n.8, p.1346-1363, August, 2008](#)
- [Radek Vingralek, Mehmet Sayal, Yuri Breitbart, Peter Scheuermann, Web++ architecture, design and performance, World Wide Web, v.3 n.2, p.65-77, 2000](#)
- [Mohit Aron, Peter Druschel, Willy Zwaenepoel, Efficient support for P-HTTP in cluster-based web servers, Proceedings of the annual conference on USENIX Annual Technical Conference, p.14-14, June 06-11, 1999, Monterey, California](#)
- [Lei Gao, Mike Dahlin, Jiandan Zheng, Lorenzo Alvisi, Arun Iyengar, Dual-quorum replication for edge services, Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware, p.184-204, November 01-01, 2005, Grenoble, France](#)
- [Li Xiao, Xin Chen, Xiaodong Zhang, Yunhao Liu, On scalable and locality-aware web document sharing, Journal of Parallel and Distributed Computing, v.63 n.10, p.945-962, October 2003](#)
- [Lei Gao, Mike Dahlin, Jiandan Zheng, Lorenzo Alvisi, Arun Iyengar, Dual-Quorum replication for edge services, Proceedings of the ACM/IFIP/USENIX 6th international conference on Middleware, p.184-204, November 28-December 02, 2005, Grenoble, France](#)
- [Steven D. Gribble, Matt Welsh, Eric A. Brewer, David Culler, The multispace: an evolutionary platform for infrastructural services, Proceedings of the annual conference on USENIX Annual Technical Conference, p.12-12, June 06-11, 1999, Monterey, California](#)
- [Bharat Chandra, Mike Dahlin, Lei Gao, Amol Nayate, End-to-end WAN service availability, Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems, p.9-9, March 26-28, 2001, San Francisco, California](#)
-  [Wai Yip Lum, Francis C.M. Lau, On balancing between transcoding overhead and spatial consumption in content adaptation, Proceedings of the 8th annual international conference on Mobile computing and networking, September 23-28, 2002, Atlanta, Georgia, USA](#)
- [An-Cheng Huang, Peter Steenkiste, Network-sensitive service discovery, Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems, p.18-18, March 26-28, 2003, Seattle, WA](#)
- [Michael Dahlin, Interpreting Stale Load Information, IEEE Transactions on Parallel and Distributed Systems, v.11 n.10, p.1033-1047, October 2000](#)
- [David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, Rohit N. Rao, Improving web availability for clients with MONET, Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, p.115-128, May 02-04, 2005](#)
-  [Lei Gao, Mike Dahlin, Amol Nayate, Jiandan Zheng, Arun Iyengar, Application specific data replication for edge services, Proceedings of the 12th international conference on World Wide Web, May 20-24, 2003, Budapest, Hungary](#)
- [Radek Vingralek, Yuri Breitbart, Mehmet Sayal, Peter Scheuermann, Web++: a system for fast and reliable web service, Proceedings of the annual conference on USENIX Annual Technical Conference, p.13-13, June 06-11, 1999, Monterey, California](#)
- [Kenneth G. Yocum, Darrell C. Anderson, Jeffrey S. Chase, Amin M. Vahdat, Anypoint: extensible transport switching on the edge, Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems, p.15-15, March 26-28, 2003, Seattle, WA](#)
-  [Paola Salomoni, Silvia Miri, Stefano Ferretti, Marco Rocchetti, A multimedia broker to support accessible and mobile learning through learning objects adaptation, ACM Transactions on Internet Technology \(TOIT\), v.8 n.2, p.1-23, February 2008](#)
- [Yatin Chawathe, Eric A. Brewer, System support for scalable and fault tolerant internet services, Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing, September 01-01, 1998, The Lake District, United Kingdom](#)
- [Michael Dahlin, Bharat Baddepudi V. Chandra, Lei Gao, Amol Nayate, End-to-end WAN service availability, IEEE/ACM Transactions on Networking \(TON\), v.11 n.2, p.300-313, April 2003](#)
- [Robert Grimm, Brian N. Bershad, Providing policy-neutral and transparent access control in extensible systems, Secure Internet programming: security issues for mobile and distributed objects, Springer-Verlag, London, UK, 2001](#)
- [Lei Gao, Mike Dahlin, Amol Nayate, Jiandan Zheng, Arun Iyengar, Improving Availability and Performance with Application-Specific Data Replication, IEEE Transactions on Knowledge and Data Engineering, v.17 n.1, p.106-120, January 2005](#)
- [Amin Vahdat, Michael Dahlin, Thomas Anderson, Amit Aggarwal, Active names: flexible location and transport of wide-area resources, Proceedings of the 2nd conference on USENIX Symposium on Internet Technologies and Systems, p.14-14, October 11-14, 1999, Boulder, Colorado](#)
-  [Armando Fox, Steven D. Gribble, Yatin Chawathe, Eric A. Brewer, Paul Gauthier, Cluster-based scalable network services, ACM SIGOPS Operating Systems Review, v.31 n.5, p.78-91, Dec. 1997](#)
-  [Robert Grimm, Brian N. Bershad, Separating access control policy, enforcement, and functionality in extensible systems, ACM Transactions on Computer Systems \(TOCS\), v.19 n.1, p.36-70, Feb. 2001](#)
-  [Valeria Cardellini, Emiliano Casalicchio, Michele Colajanni, Philip S. Yu, The state of the art in locally distributed Web-server systems, ACM Computing Surveys \(CSUR\), v.34 n.2, p.263-311, June 2002](#)

Powered by **THE ACM GUIDE TO COMPUTING LITERATURE**

The ACM Digital Library is published by the Association for Computing Machinery. Copyright © 2016 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Reader](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

System support for scalable and fault tolerant internet services

Full Text:  PdfAuthors: [Yatin Chawathe](#) *University of California at Berkeley*
[Eric A. Brewer](#) *University of California at Berkeley*

2009 Article

Published in:






Proceeding
Middleware '98 Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing
Pages 71-88
Springer-Verlag London, UK ©1998
[table of contents](#) ISBN:1-85233-088-0

 Bibliometrics

Downloads (6 Weeks): 0
Downloads (12 Months): 2
Downloads (cumulative): 49
Citation Count: 7

Tools and Resources




TOC Service:
 Email  RSS

 Save to Binder

Export Formats:
[BibTeX](#) [EndNote](#) [ACM Ref](#)

Upcoming Conference:
[Middleware'16](#)

Share:

Author Tags Recent authors with related interests Concepts in this article powered by
IBM Watson™ Contact Us | Switch to [single page view](#) (no tabs)

[Abstract](#) [Authors](#) [References](#) [Cited By](#) [Index Terms](#) [Publication](#) [Reviews](#) [Comments](#) [Table of Contents](#)

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

- 1 3COM Corporation (1996), '3COM PalmPilot', <http://www.palmpilot.com/>.
- 2 [Kevin C. Almeroth](#), [Mostafa H. Ammar](#), [The interactive multimedia jukebox \(IMJ\): a new paradigm for the on-demand delivery of audio/video](#), Proceedings of the seventh international conference on World Wide Web 7, p.431-441, April 1998, Brisbane, Australia
- 3 [Elan Amir](#), [Steven McCanne](#), [Randy Katz](#), [An active service framework and its application to real-time multimedia transcoding](#), Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication, p.178-189, August 31-September 04, 1998, Vancouver, British Columbia, Canada [doi>[10.1145/285237.285281](#)]
- 4 [Elan Amir](#), [Steven McCanne](#), [Hui Zhang](#), [An application level video gateway](#), Proceedings of the third ACM international conference on Multimedia, p.255-265, November 05-09, 1995, San Francisco, California, USA [doi>[10.1145/217279.215277](#)]
- 5 Anderson, E. (1995), The Magicrouter: An Application of Fast Packet Interposing. Class report, UC Berkeley.
- 6 Anderson, T. E., Culler, D. E., Patterson, D. A.&the NOW team (1994), A Case for Networks of Workstations: NOW, in 'Principles of Distributed Computing'.
- 7 Anonymizer Inc. (1996), 'The Web Anonymizer'. <http://www.anonymizer.com/>.
- 8 [Hari Balakrishnan](#), [Srinivasan Seshan](#), [Elan Amir](#), [Randy H. Katz](#), [Improving TCP/IP performance over wireless networks](#), Proceedings of the 1st annual international conference on Mobile computing and networking, p.2-11, November 13-15, 1995, Berkeley, California, USA [doi>[10.1145/215530.215544](#)]
- 9 [Joel F. Bartlett](#), [A NonStop kernel](#), Proceedings of the eighth ACM symposium on Operating systems principles, p.22-29, December 14-16, 1981, Pacific Grove, California, USA [doi>[10.1145/800216.806587](#)]
- 10 [William J. Bolosky](#), [Robert P. Fitzgerald](#), [John R. Douceur](#), [Distributed schedule management in the Tiger video fileserver](#), Proceedings of the sixteenth ACM symposium on Operating systems principles, p.212-223, October 05-08, 1997, Saint Malo, France [doi>[10.1145/268998.266692](#)]
- 11 Borenstein, N.&Freed, N. (1993), *MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies*. RFC-1521.
- 12 Brisco, T. (1995), *DNS Support for Load Balancing*. RFC-1764.
- 13 Chawathe, Y., Fink, S., McCanne, S.&Brewer, E. (1998), 'A Proxy Architecture for Reliable Multicast in Heterogeneous Environments'.
- 14 Chen, M.&Hong, J. (1997), Cha-Cha: Contextualizing Hypertext Searches. Class report, UC Berkeley.
- 15 Cisco Systems (1996), 'Local Director'. <http://www.cisco.com/warp/public/751/lodir/>.
- 16 CitySearch Inc. (1997). <http://www.citysearch.com/>.
- 17 [Mark Crovella](#), [Azer Bestavros](#), [Explaining World Wide Web Traffic Self-Similarity](#), Boston University, Boston, MA, 1995
- 18 [Stephen Edward Deering](#), [Multicast routing in a datagram internetwork](#), Stanford University, Stanford, CA, 1992
- 19 [Stephen Deering](#), [Deborah L. Estrin](#), [Dino Farinacci](#), [Van Jacobson](#), [Ching-Gung Liu](#), [Liming Wei](#), [The PIM architecture for wide-area multicast routing](#), IEEE/ACM Transactions on Networking (TON), v.4 n.2, p.153-162, April 1996 [doi>[10.1109/90.490743](#)]
- 20 Demers, A., Peterson, K., Spreitzer, M., Terry, D., Theimer, M.&Welch, B. (n.d.), 'The Bayou Architecture: Support for Data Sharing Among Mobile Users'.
- 21 Digital Corporation (1996), 'The AltaVista Search Engine', <http://altavista.digital.com/>.
- 22 [Sally Floyd](#), [Van Jacobson](#), [The synchronization of periodic routing messages](#), IEEE/ACM Transactions on Networking (TON), v.2 n.2, p.122-136, April 1994 [doi>[10.1109/90.298431](#)]
- 23 Fox, A. (1997), The Case For TACC: Scalable Servers for Transformation, Aggregation, Caching and Customization. *Qualifying Exam Proposal, UC Berkeley Computer Science Division*.
- 24 [Armando Fox](#), [Ian Goldberg](#), [Steven D. Gribble](#), [David C. Lee](#), [Anthony Polito](#), [Eric A. Brewer](#), [Experience with Top Gun Wingman: a proxy-based graphical web browser for the 3Com PalmPilot](#), Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing, September 01-01, 1998, The Lake District, United Kingdom
- 25 [Armando Fox](#), [Steven D. Gribble](#), [Eric A. Brewer](#), [Elan Amir](#), [Adapting to network and client variability via on-demand dynamic distillation](#), Proceedings of the seventh international conference on Architectural support for programming languages and operating systems, p.160-170, October 01-04, 1996, Cambridge, Massachusetts, USA [doi>[10.1145/237090.237177](#)]
- 26 [Armando Fox](#), [Steven D. Gribble](#), [Yatin Chawathe](#), [Eric A. Brewer](#), [Paul Gauthier](#), [Cluster-based scalable network services](#), Proceedings of the sixteenth ACM symposium on Operating systems principles, p.78-91, October 05-08, 1997, Saint Malo, France [doi>[10.1145/268998.266662](#)]
- 27 Fox, A., Gribble, S. D., Chawathe, Y.&Brewer, F. A. (1998). Adapting to Network and Client Variation Using Active Proxies:

Lessons and Perspectives, in 'A special issue of IEEE Personal Communications on Adaption'.

- 28 [Jim Gray, The transaction concept: virtues and limitations \(invited paper\), Proceedings of the seventh international conference on Very Large Data Bases, p.144-154, September 09-11, 1981, Cannes, France](#)
- 29 [Steven D. Gribble, Eric A. Brewer, System design issues for internet middleware services: deductions from a large client trace, Proceedings of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems, p.19-19, December 08-11, 1997, Monterey, California](#)
- 30 [Steven D. Gribble, Gurmeet Singh Manku, Drew Roselli, Eric A. Brewer, Timothy J. Gibson, Ethan L. Miller, Self-similarity in file systems, Proceedings of the 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems, p.141-150, June 22-26, 1998, Madison, Wisconsin, USA. \[doi>10.1145/277851.277894\]](#)
- 31 Handley, M. (1996), *SAP: Session Announcement Protocol*. Internet Draft.
- 32 Inktomi Corp. (1996), 'The HotBot Search Engine', <http://www.hotbot.com/>.
- 33 [Will E. Leland, Murad S. Taqqu, Walter Willinger, Daniel V. Wilson, On the self-similar nature of Ethernet traffic \(extended version\), IEEE/ACM Transactions on Networking \(TON\), v.2 n.1, p.1-15, Feb. 1994. \[doi>10.1109/90.282603\]](#)
- 34 MapQuest (1996). <http://www.mapquest.com/>.
- 35 McQuillan, J., Richer, I.&Rosen, E. (1980), 'The New Routing Algorithm for the ARPANET', *IEEE Transactions on Communications* 28(5), 711--719.
- 36 Mitzenmacher, M (1997), How useful is old information? Extended abstract: Using stale information for load balancing in distributed systems.
- 37 [C. Mohan, Don Haderle, Bruce Lindsay, Hamid Pirahesh, Peter Schwarz, ARIES: a transaction recovery method supporting fine-granularity locking and partial rollbacks using write-ahead logging, ACM Transactions on Database Systems \(TODS\), v.17 n.1, p.94-162, March 1992. \[doi>10.1145/128765.128770\]](#)
- 38 National Aeronautics and Space Administration (1997), 'The Mars Pathfinder Mission Home Page', <http://mpfwww.jpl.nasa.gov/default1.html>.
- 39 Nonnenmacher, J.&Biersack, E. W. (1998), Optimal Multicast Feedback, in 'Proceedings of IEEE INFOCOM '98', San Francisco, CA.
- 40 [Eric S. Raymond, The New Hacker's Dictionary, MIT Press, Cambridge, MA, 1996](#)
- 41 Schuett, A.&Raman, S. (1997), MARS: A Media Archive Server for On-demand Remote Playback. Class report, UC Berkeley.
- 42 Schulzrinne, H., Casner, S., Frederick, R.&Jacobson, V. (1996), 'RTP: A Transport Protocol for Real-Time Applications', Internet Engineering Task Force, Audio-Video Transport Working Group. RFC-1889.
- 43 [John F. Shoch, Jon A. Hupp, The "worm" programs—early experience with a distributed computation, Communications of the ACM, v.25 n.3, p.172-180, March 1982. \[doi>10.1145/358453.358455\]](#)
- 44 [Carl A. Waldspurger, William E. Weihl, Lottery scheduling: flexible proportional-share resource management, Proceedings of the 1st USENIX conference on Operating Systems Design and Implementation, p.1-es, November 14-17, 1994, Monterey, California](#)
- 45 Yahoo! Inc (1995). <http://www.yahoo.com/>.
- 46 [Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, David Culler, Using smart clients to build scalable services, Proceedings of the annual conference on USENIX Annual Technical Conference, p.8-8, January 06-10, 1997, Anaheim, California](#)

Powered by **THE ACM GUIDE TO COMPUTING LITERATURE**

The ACM Digital Library is published by the Association for Computing Machinery. Copyright © 2016 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Reader](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Network Protocol Handbook

- TCP/IP
- IEEE 802.2
- DECnet/LAT
- IPX/SPX
- OSI
- GOSIP
- XNS
- AppleTalk

Matthew Naugle

Uyless Black,
Series Advisor

McGraw-Hill Series on Computer Communications

NOTICE: Return or renew all Library Materials! The *Minimum Fee* for each Lost Book is \$50.00.

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University. To renew call Telephone Center, 333-8400

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

	NOV 04 1997	
	MAY 11 1999	
DEC 01 1997	MAY 06 1999	
	DEC 21 1995	
FEB 23 1998	JUN 14 2000	
MAY 10 1999	NOV 21 2000	
	JUL 11 2003	
		JUN 24 2016
MAY 11 1998		
JUN 11 1998		
JUN 08 1997		
JUL 27 1998		
JUL 20 1998		
DEC 15 1998		

L161—O-1096

Network Protocol Handbook

Matthew G. Naugle

McGraw-Hill, Inc.

New York San Francisco Washington, D.C. Auckland Bogotá
Caracas Lisbon London Madrid Mexico City Milan
Montreal New Delhi San Juan Singapore
Sydney Tokyo Toronto

Library of Congress Cataloging-in-Publication Data

Naugle, Matthew G.

Network protocol handbook / Matthew G. Naugle.

p. cm. — (McGraw-Hill series on computer communications)

Includes index.

ISBN 0-07-046461-8

1. Computer network protocols. I. Title. II. Series.

TK5105.55.N38 1994

004.6'2—dc20

93-2013

CIP

Copyright © 1994 by McGraw-Hill, Inc. All rights reserved. Printed in the United States of America. Except as permitted under the United States Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a data base or retrieval system, without the prior written permission of the publisher.

1 2 3 4 5 6 7 8 9 0 DOC/DOC 9 9 8 7 6 5 4 3

ISBN 0-07-046461-8

The sponsoring editors for this book were Neil Levine and Jeanne Glasser, and the production supervisor was Pamela Pelton. It was set in Century Schoolbook by North Market Street Graphics.

Printed and bound by R. R. Donnelley & Sons Company.

NetWare, IPX, SPX, SAP, VAP, and NLM are registered trademarks of Novell, Inc. IBM, IBM PC, and IBM AT are registered trademarks, and PC/ST, PC-DOS, and DOS are registered trademarks of International Business Machines.

AppleTalk, LocalTalk, EtherTalk, TokenTalk, Appleshare, AFP, Macintosh,

LaserWriter, LaserWriter Plus, LaserWriter NTX, MAC 128, MAC 256, MAC 512, and 512 Enhanced are registered trademarks of Apple Computer.

XNS and Ethernet are registered trademarks of Xerox Corporation.

DELNI, DEMPR, VAX, microVAX, LAT, and DECnet are registered trademarks of Digital Equipment Corporation.

UNIX is a registered trademark of AT&T Bell Laboratories.

SUN Workstation is a registered trademark of Sun Microsystems, Inc.

MS-DOS is a registered trademark of Microsoft Corporation.

Information contained in this work has been obtained by McGraw-Hill, Inc. from sources believed to be reliable. However, neither McGraw-Hill nor its authors guarantee the accuracy or completeness of any information published herein and neither McGraw-Hill nor its authors shall be responsible for any errors, omissions, or damages arising out of use of this information. This work is published with the understanding that McGraw-Hill and its authors are supplying information but are not attempting to render engineering or other professional services. If such services are required, the assistance of an appropriate professional should be sought.

004.62

N224n

Engineering

This book is dedicated to my wife, Regina. Her constant reassurance and most of all her unrelenting patience made it possible to write this book. Also many thanks to my three children: Bryan, Courtney, and Lauren, who have proven that life is most enjoyable when looking at it through simplistic, innocent, and trusting eyes.

OP

Contents

Preface	xi
Chapter 1. Introduction and Wiring Concepts	1
Overview	4
The Open Systems Interconnection (OSI) Model	6
Topologies	8
Star Topology	8
Ring Topology	10
Bus Topology	10
Wiring Systems for Ethernet and Token Ring—Physical Layer	11
Ethernet Layer Physical Components	12
Thick Coaxial Cable	13
Thin Coaxial Cable	16
Unshielded Twisted Pair	19
Repeaters	22
Token Ring Physical Layer	27
Data Connectors	31
Multistation Access Unit (MAU or MSAU)	32
Chapter 2. Ethernet and Token Ring	37
The Data-Link Layer	37
Ethernet	37
Token Ring	43
Addressing the Packet	52
Bridges, Routers, and Basic Network Information	62
Bridges (Ethernet or IEEE 802.3)	62
Bridges (Token Ring)	71
Routers	78
Conclusion	82
Chapter 3. IEEE 802.2	89
LLC Type 2 Operation	92
Connection-Oriented LLC2—Asynchronous Balance Mode (ABM)	92
Frame Formats	93
SAP Addressing	94
	vii

viii Contents

Sequencing of Data (LLC2)	98
Timer Functions	101
Connection-Oriented Services of the IEEE 802.2 Protocol	101
Details of LLC Type 2 Operation	102
LLC2 Frame Reception	104
A Live Connection	105
LLC Type 1 Operation	108
Information Transfer	109
SNAP	110

Chapter 4. Xerox Network System (XNS) 115

Network Definition	116
Terminology and Definitions	121
XNS at Level 0	124
Nonbroadcast or Point to Point	124
Internet Transport Protocols-Level 1	127
Host Numbers	128
Network Numbers	129
Socket Numbers	129
Identifying the Internet Datagram Fields	130
Level 2	133
Routing Information Protocol	133
Level 2 Error Protocol	154
Level 2 Echo Protocol	157
Level 2 Sequence Packet Protocol (SPP)	157
Level 2 Packet Exchange Protocol	169

Chapter 5. Novell NetWare 179

Introduction	181
Version History	181
Concepts	182
Internet Packet Exchange (IPX)	183
IPX Routing Architecture	185
IPX Routing Functions	192
IPX Routing Tables	194
IPX Routing Information Protocol (RIP)	198
Router Operation	200
The Service Advertising Protocol (SAP)	208
SAP Operation	212
An Introduction to Workgroup Client-Server Computing	216
Physical and Virtual Drives	216
Printing Using a Server	218
The NetWare Interface	219
The Workstation	221
Connection Establishment	222
File Server Concepts	226
Server System Calls	233
NetWare Supplementals	236
Packet Burst Mode Technology	236
Multiple NetWare Protocols	238

Chapter 6. Transmission Control Protocol/Internet Protocol (TCP/IP)	241
Introduction	242
Fundamentals	243
Request for Comments (RFCs)	246
The Protocol Suite	247
Overview	247
Section 1: The Network Layer	249
Internet Protocol (IP)	249
IP Addresses, Subnetting, and the Address Resolution Protocol (ARP)	255
IP Routing	278
Routing Information Protocol (RIP)	287
Internet Control Message Protocol (ICMP)	301
Section 2: Transport Layer Protocols	304
User Datagram Protocol (UDP)	304
Transport Control Protocol (TCP)	309
Section 3: Selected TCP/IP Applications	325
TELNET	326
File Transfer Protocol (FTP)	330
Trivial File Transfer Program (TFTP)	334
Domain Name Service (DNS)	335
Simple Mail Transfer Protocol (SMTP)	337
Chapter 7. AppleTalk	341
The Physical Layer—AppleTalk Hardware	341
LocalTalk	341
LAP Manager for EtherTalk and TokenTalk	360
The AppleTalk Network Layer: End-to-End Data Flow	366
Datagram Delivery Protocol (DDP)	366
Routers, Routing Tables, and Maintenance	371
AppleTalk Echo Protocol (AEP)	385
Names on AppleTalk	387
Transport-Layer Services	394
AppleTalk Transaction Protocol (ATP)	395
Printer Access Protocol (PAP)	397
AppleTalk Session Protocol (ASP)	398
AppleShare and the AppleTalk Filing Protocol (AFP)	405
Chapter 8. Digital Network Architecture (DNA)	411
History	411
The Routing Layer	414
DECnet Phase IV Routing	414
End Communication Layer: The DNA Transport Layer	441
The Session Control Layer	451
Network Application Layer	453
Data Access Protocol (DAP)	455
Network Virtual Terminal (NVT)	455

x Contents

Chapter 9. Local Area Transport (LAT)	465
Node Types on a LAT LAN	465
LAT Topology	470
Service Class Layer	472
Slot Layer	473
Virtual Circuit Layer	474
LAT Components	475
Services	477
Service Ratings	478
Multicasting Service Announcement Messages	479
Maintaining Service and Service Node Directories	479
Groups	480
Session Establishment	480
Session Establishment	482
Data Transfer	485
Session Flow Control	486
Slot Flow Control	487
Session Termination	489
Other LAT Services	489
Host-Initiated Requests	489
Session Management	491
Virtual Circuit Maintenance	491
Connection Queue and Maintenance	492
Chapter 10. Open Systems Integration (OSI) Protocol	493
OSI Routing	495
OSI Addressing	499
Transport Layer	502
Session Layer	507
Applications	508
File Transfer, Access and Management, and X.400	508
X.400	509
Index	513

Preface

Today, there are an incredible amount of protocols that make up an internet. This includes local area networks, wide area networks, and the software that runs them. Although most of them go unnoticed, protocols range from the electrical specifications of the connectors to the protocols that run the internetwork. Just to mention a few:

Local Area Networks

Ethernet

Token Ring

Fiber Data Distributed Interface (FDDI)

Asynchronous Transfer Mode (ATM)

WAN Protocols

Proprietary (usually a derivative of HDLC)

Point-to-Point Protocol (PPP)

Switched Multimegabit Data Service (SMDS)

Frame relay

Synchronous Optical Network (SONET)

High-Speed Serial Interface (HSSI)

X.25

High-level Data Link Control (HDLC)

Telecommunications (phone systems, clocking, CSU/DSU)

LAN/WAN Protocols

Transport Control Protocol/Internet Protocol (TCP/IP)

Xerox Network Standard (XNS) and any derivative thereof

xii Preface

- Internet Packet eXchange (IPX; i.e., Novell NetWare)
- DECnet
- IEEE 802.2
- Local Area Transport (LAT)
- AppleTalk
- SNA and SNA Gateways
- Open System Interconnect (OSI)
- Bridging (Source Route, Transparent with Spanning Tree, Translation, Source Route Transparent)

Miscellaneous

- Network Management Systems
 - Simple Network Management Protocol (SNMP)
 - Common Management Interface Protocol (CMIP)
 - Common Management Interface Specification (CMIS)
 - CMIP over TCP/IP (CMOT)
 - Proprietary
- Protocol analyzers
- NetView
- Operating systems (too many to mention)
- Wiring hubs
- Hardware platforms (personal computers, workstations, mini's and mainframes)

Network Applications

- Novell NetWare
- IBM LAN Server
- TCP applications (and the applications for other protocols)
 - Telnet
 - File Transfer Protocol (FTP)
 - Simple Mail Transfer Protocol (SMTP)
 - Domain Name Server (DNS)

There are more that should be on the list, but from the foregoing sampling, I think the reader should have an understanding of the problems associated with the many intricacies of networking. All the preceding protocols are less than 25 years old and most are not more than 12 years old.

The real problem is that network specialists are expected to know everything there is to know about the protocols (hardware or software) that make up an internet. Besides that, a network administrator, network user, network planner, network buyer, etc., expects any vendor of any network communication equipment to understand all of the preceding protocols. It is impossible to understand all the protocols that have evolved since 1981 (the beginning of the network in the commercial workplace). Let me explain right here that it is impossible to know all the intricacies about each and every network protocol in use today.

The intent of this book is to introduce the readers to the majority of the basic functions of the aforementioned protocols. It introduces and explains the protocols so that if the reader needs to understand any protocol that is not in this book, he or she should have an easier time in comprehending that protocol.

To understand networks, it is best to *believe what you experience, not what you read.*

It is my belief that studying XNS is the fastest way to introduce any reader into the world of networks and internetworks. It is the fastest way to understand them. It was a protocol derived and meant to run on top of a LAN (Ethernet). The technical portion of the protocol is easy to understand and, by completely understanding this protocol, the reader will be able to read any of the remaining chapters with a greater comprehension at the end of the reading of that chapter. It is therefore suggested that the reader read Chaps. 1, 2, and 3 of this book before reading any other chapter. There are terms and definitions in these chapters that will be used throughout the rest of the chapters. (Chapter 3 on IEEE 802.2, although not mandatory, will provide the reader valuable information that will make the rest of the book easier to read.)

Otherwise, the reader should feel free to skip through the chapters and read them as needed. For example, if you are most interested in the operation of DECnet, it is suggested that you read the first three chapters and then move on to the DECnet chapter. There is no reason to read any other protocol chapter.

The book does not try to make the reader understand all the aspects of each and every protocol. It merely tries to introduce the reader to some fairly common protocols found on most of today's networks. The book is not meant to be used to write code or to troubleshoot a network. It is meant to introduce a reader to the protocols and to give a fairly detailed comprehension of network protocols without the theory.

Some topics are complex; some are not. The book is fairly balanced in this regard. It is not a "fluff" book that introduces only simple concepts into networks. It starts out with introduction material and then digs

into the protocols fairly heavily. The reader of this book can be technical or nontechnical and the outcome will be the same: a full understanding of network protocols.

Acknowledgments

Many thanks go to four individuals:

Uyless Black who is the consulting editor for this book. He definitely fed me good information.

My editor, Neil Levine, who never seems to get into a bad mood, is incredibly patient, and has always offered genuinely good advice.

Brad Black, who is manager of the internet of the Duke Power Company in Charlotte, North Carolina. Brad accepted the invitation to review the original manuscript of this book and offered great advice to improve the book. Many of his ideas and suggestions are in this book.

Jack Maxfield, a systems engineer for Wellfleet Communications in Atlanta, Georgia, improved the book with his vast knowledge of protocols and many years of experience in the networking arena. Jack offered great advice and corrected the manuscript where needed (even if it is dry in some places!).

Also, I would like to thank all the individuals I have met throughout the years who have not held back in conveying their knowledge and ideas to me. These individuals are true tributes to their fields of expertise.

Matthew G. Naugle

Transmission Control Protocol/Internet Protocol (TCP/IP)

Transmission Control Protocol/Internet Protocol (TCP/IP) is one of today's most widely used networking protocols. A TCP/IP network is generally a heterogeneous network, meaning there are many different types of network computing devices attached.

Before TCP/IP, network protocols were proprietary and known to only a few individuals. Users and network administrators were held to proprietary network environments and proprietary network applications, which deterred network development and enhancement in all corporate environments. TCP/IP allows public access to network protocols and allows seamless integration between all computing environments that wish to operate in a network environment. Commercial success of this protocol happened unexpectedly. TCP/IP was never envisioned as a commercial network system and was not envisioned to gain the widespread use it has today. Had the open protocol of TCP/IP not flourished, the network environment would possibly be in the same situation as the operating system world was in the 1970s. Everyone had their own proprietary operating system, and users were stuck with one operating system. TCP/IP not only gave the network world the groundwork for future protocols (OSI), it allowed open access so that users may choose this network operating system without having to choose a single vendor along with it. We should be thankful for this. There are many more advantages to this protocol that will be pointed out throughout the text.

TCP/IP allowed for open communications to exist and also allowed for the proliferation of LAN to LAN to LAN to WAN connectivity between multiple operating environments. The only people hurt by

this open protocol are the network companies that do not support it. All other companies have flourished based on it.

From this, one would tend to think that this operating system was developed by a large-scale R&D center like that of IBM or DEC. It wasn't. It was developed by a team of research-type people, but these individuals were college professors, graduate students, and undergraduate students from major universities. This should not be hard to believe. These individuals are the type who enjoy not only R&D work, but who also believe that, when problems occur, the fun starts.

Many years from now we will look back on the TCP/IP protocol as the protocol that provided the building blocks of future data communications.

The following text will show the inner workings of the TCP/IP protocol. The text will not enable the reader to write code or perform protocol analyzer traces on protocol packets, but will merely provide an alternative to the current theory books.

Introduction

The suite of protocols that encompass TCP/IP were originally designed to allow different types of computer systems to interact. It was developed by a project underwritten by an agency of the Department of Defense known as the Advanced Research Projects Agency (DARPA).

There are many reasons why TCP/IP became popular, two of which are paramount. First, DARPA provided a grant to allow the protocol suite to become part of Berkeley's UNIX system. When TCP/IP was introduced to the commercial marketplace, UNIX was always mentioned in every story about it. Berkeley UNIX and TCP/IP became the standard operating system and protocol of choice for a lot of major universities, where it was used with workstations in engineering and research environments. In 1983, all U.S. government proposals that included networks mandated the TCP/IP protocol in all the government proposals.

Second was the capability of the protocol to allow dissimilar systems to communicate through the network. At the time of the TCP/IP influx, other protocols were in use and very popular with the LAN vendors. Variations of Xerox's XNS and Digital's proprietary DECnet/LAT were the most popular. One drawback for users of these protocols is that the protocols were *vendor dependent*. Running XNS on one system did not guarantee compatibility of communication to any other system except for the same vendor. This was good for the vendor, but it tended to lock users into one vendor.

TCP/IP eliminated this. TCP's beginnings were rough (interoperability issues), but the protocol stabilized and the interoperability between different computer and operating systems became a reality. For example, a DEC system running the VMS operating system combined with

TCP/IP running with a Sun Micro systems could communicate the specific applications able to log on to the two across a network. When interconnecting TCP/IP, it does not require a new operating system of hardware, implementing it with the same methods used to

Fundamentals

TCP/IP originated as a response to a difficult problem with one another. A difficult task considered in the early 1970s was to develop a system that would not disclose its details to anyone. This is known as the "secret sauce" problem.

The architecture of TCP/IP developed to communicate across different hardware architectures on those systems.

The original network (NCP). The protocol was primarily used for process-to-process communication such as file transfer.

The first few years had some serious problems overcome these problems with a recommendation to rework the protocol with another call to action. The person responsible for the Internet Protocol

* ARPAnet was the network of universities and scientific institutions that included many different computers on the Internet.

TCP/IP running as the network operating system can communicate with a Sun Microsystems UNIX workstation running TCP/IP. The two systems could communicate by taking advantage of the protocol and the specific applications written for the protocol, primarily by being able to log on to one another and by being able to transfer files between the two across a network.

When interconnecting computers and their operating systems with TCP/IP, it does not matter what the hardware architecture or the operating system of the computer is. The protocol will allow any computer implementing it to communicate with another. What follows are the methods used to accomplish this.

Fundamentals

TCP/IP originated when DARPA was tasked to bring about a solution to a difficult problem: allowing different computers to communicate with one another as if they were the same computer. This was a difficult task considering that all computer architectures in those days (the early 1970s) were highly guarded secrets. Computer manufacturers would not disclose either their hardware or software architectures to anyone. This is known as a *closed* or *proprietary* system.

The architecture behind TCP/IP takes an alternative approach. TCP/IP developed into an architecture that would allow the computers to communicate without grossly modifying the operating system or the hardware architecture of the machine. TCP/IP runs as an application on those systems.

The original result was known as the Network Control Program (NCP). The protocol was developed to run on multiple hosts in geographically dispersed areas through a packet switching internet known as the Advanced Research Project Agency network—ARPAnet.* This protocol was primarily used to support application-oriented functions and process-to-process communications between two hosts. Specific applications, such as file transfer, were written to this network operating system.

The first few years of this design proved to be an effective test, but had some serious design flaws. A research project was developed to overcome these problems. The outcome of this project was a recommendation to replace the original program known as NCP program with another called Transmission Control Program (TCP). The protocol responsible for routing the packets through an internet was termed the Internet Protocol. Today, the common term for this standard is TCP/IP.

* ARPAnet was the term used to describe the public TCP/IP network that links universities and scientific centers. Due to the large growth of this network (which now includes many different private, government, and public agencies), it is now known as the Internet.

With TCP/IP replacing NCP, the NCP application-specific programs were converted to run over the new protocol. The protocol became mandated in 1983, when ARPA demanded that all computers attached to the ARPAnet use the TCP/IP protocol.

In order to perpetuate the task of allowing dissimilar government computers to communicate, DARPA gave research grants to UCLA, University of California at San Bernadino, the Stanford Research Institute (SRI), and the University of Utah. A company called BBN provided the Honeywell 316 Interface Message Processors (IMPs) which provided the internet communications links. In 1971, the ARPAnet Networking Group dissolved and DARPA took over all the research work. Between the years of 1975–1979, DARPA had begun the work on the Internet technology which resulted in the TCP/IP protocols as we know them today.

Around 1980, the Internet was started when DARPA began to move their machines and research networks over to the TCP/IP protocols.

Around 1983, the Office of the Secretary of Defense mandated that all government computers be switched over to the new TCP/IP protocols. Also during this year, the ARPAnet was split into two networks:

1. The Defense Data Network (DDN)—also known as the MILNET (military network)
2. The DARPA Internet—a new name for the old ARPAnet network.

In 1985, the National Science Foundation expanded the use of the Internet by using it as the vehicle to network as many scientists as possible. This program was started as network access to their six super computer sites. The NSFNET was formally established in 1986.

There are two prominent items in the architecture of this protocol that need to be understood: the architecture itself and the elements that control the architecture.

First, the protocol is well defined. The TCP/IP protocol is known as an open protocol. The architecture is defined in public documents for anyone who would like to build a TCP/IP operating system. What governs the protocol? Refer to Fig. 6.1a. The TCP/IP protocol suite is governed by an organization known as the Internet Activities Board (IAB). This group was originally set up by DARPA to allow information to be exchanged between the major individuals involved in the ARPAnet. Each member of the IAB heads a group known as the Internet Engineering Task Force (IETF) that is responsible for investigating a problem or a set of issues. Each IETF has a chairperson known as the Internet Architect whose responsibilities include future directions, coordination of the activities, and technical directions.

Finally, there are documents written known as Request for Comments (RFCs) that define the processing functions of this protocol.

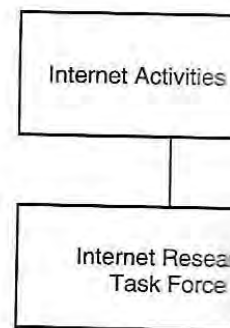
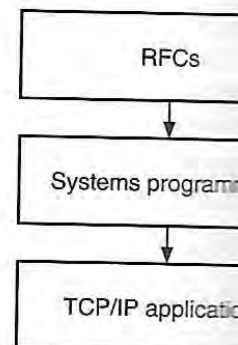


Figure 6.1 (a) Intern



These documents
 ested to read. The

Network Information
 14200 Park Meadow
 Suite 200
 Chantilly, VA 22021
 (703) 802-4535

The ARPAnet
 What is signific
 ARPAnet), it will
 erally about a col
 spelled with a low

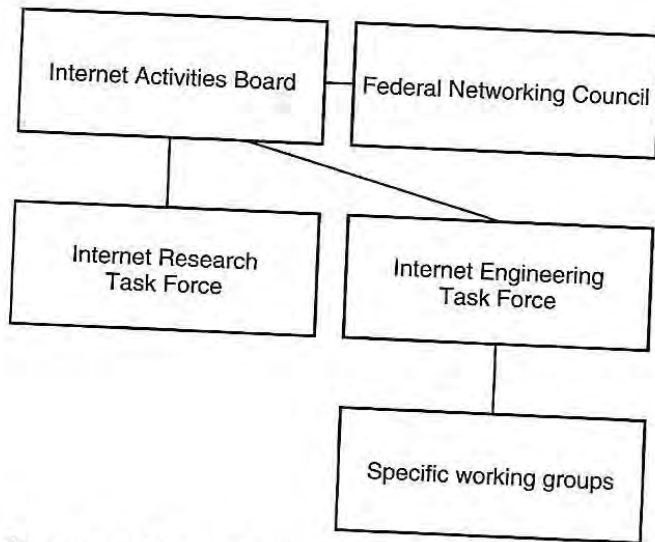


Figure 6.1 (a) Internet structure.

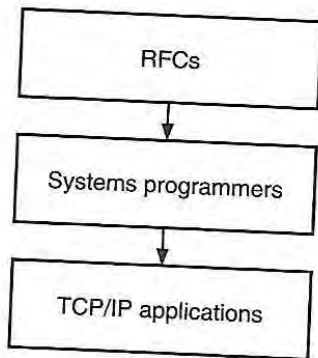


Figure 6.1 (b) development of TCP from the RFCs.

These documents are placed in the public domain for all who are interested to read. The address and phone number to purchase the RFCs is:

Network Information Center
14200 Park Meadow Drive
Suite 200
Chantilly, VA 22021
(703) 802-4535

The ARPANet changed names and is now known as the Internet. What is significant here is when talking about the Internet (the ARPANet), it will always be spelled with a capital I. When talking generally about a collection of networks that form an internet it will be spelled with a lowercase i.

One last distinction of TCP/IP. To run the protocol on any network does not require a connection to the Internet. TCP/IP may be installed on as few as two network stations or as many as can be addressed (possibly millions). A TCP/IP network may be built privately. This means that there is no access to the Internet. When a network requires access to the Internet, the network administrator must call the NIC to place a request for access and to be assigned an official IP address.

Request For Comments (RFCs)

The Request for Comments are papers (documents) that define the TCP/IP protocol suite. These papers may be submitted to the editor in chief of NIC by anyone, and are usually about reports for work, proposals for protocols, and the actual protocol standards. As a matter of fact, there is an RFC that defines the procedure for submitting an RFC. The following text will be mentioning the RFCs periodically, for any modification to the TCP/IP protocol is defined in these RFCs.

Each RFC is assigned a number in ascending sequence (Newer RFCs have higher numbers and they are never reassigned). Newer RFCs may make older RFCs obsolete. For example, the SNMP network management was originally proposed in RFC 1065, 1066, and 1067. The latest adopted SNMP RFC is 1155, 1156, and 1157.

The RFCs are continuing to evolve as the technology demands. For example, the wide area network connection facility known as the Frame Relay specification is becoming very popular, and there are RFCs to define how to interface TCP to the frame relay protocol. RFCs also allow refinements to enhance better interoperability. As long as the technology is changing, the RFCs must be updated to allow connection to the protocol suite.

The second part of this protocol is the actual protocol itself. What is TCP/IP?

Let's start with understanding the functions and protocols by studying their placement in the OSI model. In looking at Fig. 6.2, we can see that there are distinct protocols that run at each layer of the OSI model, starting from the network layer to the application layer. The heart of the TCP/IP network protocol is at layers 3 and 4. The applications for this protocol (file transfer, mail, and terminal emulation) run at the application layer.

As you can see, this protocol runs independently of the data-link and physical layer. At these layers the TCP/IP protocol can run on Ethernet, Token Ring, FDDI, serial lines, X.25, etc. It has been adapted to run over these protocols. TCP/IP was first used to interconnect computer systems through synchronous lines and not high-speed local area networks. Today, it is used on any type of media. This includes serial lines (asynchronous and synchronous) and high-speed networks such as FDDI, Ethernet, and Token Ring.

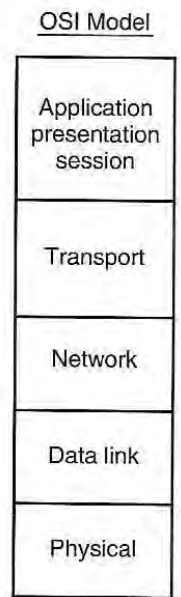


Figure 6.2 OSI model

The protocol suite

TCP/IP is actually a path that allows

In this chapter, we will discuss the following phases:

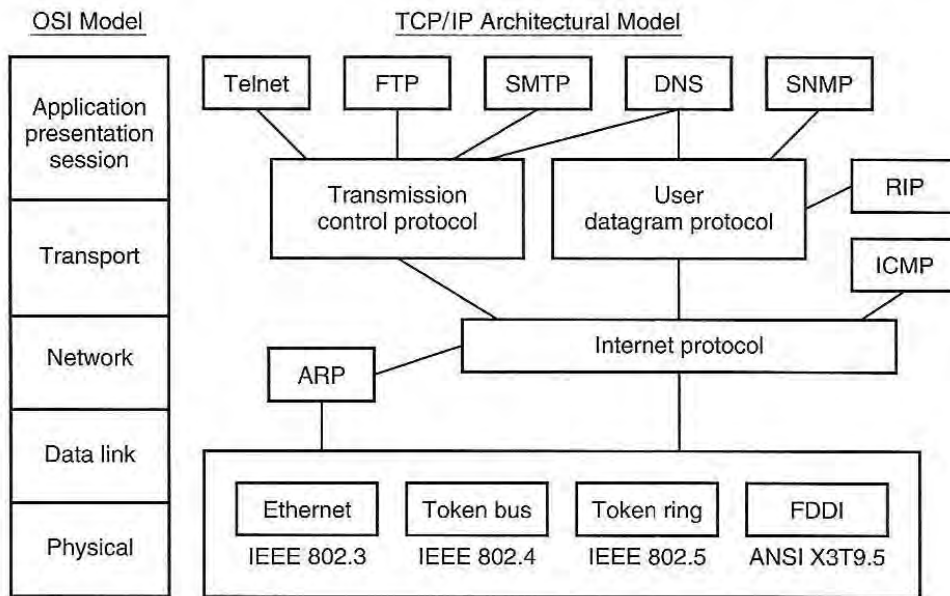
1. The Internet Protocol (IP)
2. The Transport Layer (TCP: TELNET, FTP, and SMTP) and the Application Layer (DNS), and Simple Mail Transfer Protocol (SMTP)

A brief summary of the

Overview

Section 1: the network layer

Internet Protocol (IP) is the protocol that allows data to be sent over a switched communication network.



FTP — File Transfer Protocol
 SMTP — Simple Mail Transfer Protocol
 DNS — Domain Name Server
 SNMP — Simple Network Management Protocol
 ICMP — Internet Control Message Protocol
 ARP — Address Resolution Protocol
 FDDI — Fiber Distributed Data Interface
 RIP — Routing Information Protocol

Figure 6.2 OSI model and TCP/IP architectural model.

The protocol suite

TCP/IP is actually a family of protocols working together to provide a path that allows internet data communication.

In this chapter, the TCP/IP protocol suite will be discussed in three phases:

1. The Internet Protocol (IP)
2. The Transport Control Protocol (TCP)
3. The suite of applications that were specifically developed on top of TCP: TELNET, File Transfer Protocol (FTP), Domain Name Service (DNS), and Simple Mail Transfer Program (SMTP), to name a few

A brief summary of these three section is given below:

Overview

Section 1: the network layer

Internet Protocol (IP). This protocol is designed to interconnect packet-switched communication networks to form an internet. It transmits

blocks of data called datagrams received from its upper-layer software to and from source and destination hosts. It provides a best effort or connectionless delivery service between the source and destination—connectionless in that it does not establish a session with the destination before it transmits its data. This is the layer that is also responsible for the protocol addressing.

Internet Control Message Protocol (ICMP). Not really part of the IP layer, this works directly with the IP layer. Reports certain error conditions on the network. Basically, it allows internet routers to transmit error or test messages. These error messages may be that a network destination cannot be reached or they may generate/reply to an echo request packet (PING, explained later).

Address Resolution Protocol (ARP). ARP is not really part of the network layer, it resides in-between the IP and data-link layers. It is protocol that translates between the 32-bit IP address and a 48-bit Local Area Network address. Since TCP was not originated to run over a LAN, an address scheme was implemented to allow each host and network on the internet to identify itself. When TCP/IP was adapted to run over the LAN, the IP address had to be mapped to the 48-bit data-link or physical address that LANs use, and this is the protocol that accomplishes it.

Section 2: transport layer protocols

Transmission control protocol. Since IP provides for a connectionless delivery service of TCP data, TCP provides application programs access to the network, using a reliable connection-oriented transport-layer service. This protocol is responsible for establishing sessions between user processes on the internet, and also ensures reliable communications between two or more processes. The functions that it provides are:

1. to listen for incoming session establishment requests
2. to request a session to another network station
3. to send and receive data reliably
4. gracefully to close a session

User datagram protocol. UDP provides application programs access to the network using an unreliable connectionless transport-layer service. It allows the transfer of data between source and destination stations without having to establish a session before data is transferred. This protocol also does not use the end-to-end error checking that TCP

uses. With UDP, overhead is low. It is used for applications that require the robustness of connectionless cast messages, not connection-oriented.

Section 3: application layer protocols

TELNET. For network-to-network packet-switching, TELNET is a protocol that emulates a terminal workstation to appear as if the terminal were connected to the network.

File Transfer Protocol (FTP). In terms of control, FTP is a protocol on the internet. FTP is a file transfer protocol (based on TCP) primarily used for transferring files between hosts.

Simple Mail Transfer Protocol (SMTP). that is robust enough to handle mail. SMTP allows for the transfer of mail between systems on an internet.

Domain Name Service (DNS). allows users to use humanly readable names to identify hosts. It provides a name-to-number mapping.

The aforementioned protocols are the TCP/IP protocols. If you use the TCP/IP protocol, it will be the same.

Section 1: The Network Layer

Internet protocol (IP)

The main goal of IP is to form an internet of hosts.

The IP protocol has four main functions:

1. Basic unit for communication
2. Addressing
3. Routing
4. Fragmentation and reassembly

uses. With UDP, transport-layer functionality is there, but the overhead is low. It is primarily used for those applications that do not require the robustness of the TCP protocol. For example, mail, broadcast messages, naming service, and network management.

Section 3: applications

TELNET. For new users to the TCP/IP protocol, this is not Telenet, a packet-switching technology using the CCITT standard X.25. It is pronounced TELNET. This is an application-level protocol that allows terminal emulation to pass through a network to a remote network station. TELNET runs on top of the TCP protocol and allows a network workstation to appear to a remote device (i.e., a host) as a terminal as if the terminal were a local device.

File Transfer Protocols (FTP, Trivial FTP). FTP is similar to TELNET in terms of control, but this protocol allows for data files to be transferred on the internet. FTP resides on top of TCP. TFTP is a simplex file transfer protocol (based on an unreliable transport layer called UDP). It is primarily used for boot loading of configuration files across an internet.

Simple Mail Transfer Protocol (SMTP). This is an electronic mail system that is robust enough to run on the entire Internet system. This protocol allows for the exchange of electronic mail between two or more systems on an internet.

Domain Name Service (DNS). This is a centralized name service that allows users to establish connections to network stations using humanly readable names instead of cryptic network addresses. It provides a name-to-network address translation service.

The aforementioned protocols are the topics for this chapter. Since the TCP/IP protocol suite basically begins at layer 3, the Internet Protocol, it will be the protocol that we will study first.

Section 1: The Network Layer

Internet protocol (IP)

The main goal of IP is to provide interconnection of subnetworks to form an internet in order to pass data.

The IP protocol provides four main functions:

1. Basic unit for data transfer
2. Addressing
3. Routing
4. Fragmentation of datagrams

Basic unit for data transfer—connectionless, best-effort delivery service. The IP layers provide the entry into the delivery system used to transport data across the internet.

Usually, when anyone hears of the name IP, they automatically think of the devices commonly known as *routers* which connect multiple subnetworks together. It is true the IP performs these tasks, but the IP protocol performs many other tasks, as mentioned previously. The IP protocol runs in all the participating network stations that are attached to subnetworks so that they may submit their packets to routers or directly to other devices on the same network. It resides between the data-link layer and the transport layer.

Declarations. The primary goal of IP is to provide the basic algorithm for transfer of data to and from a network. It provides a connectionless delivery service for the upper-layer protocols. This means that IP does not set up a session (a virtual link) between the transmitting station and the receiving station prior to submitting the data to the receiving station. It encapsulates the data and delivers it on a *best-effort basis*. IP does not inform the sender or receiver of the status of the packet. It merely attempts to deliver the packet and will not make up for the faults encountered in this attempt. This means that if the data link fails or incurs a recoverable error, the IP layer will not inform anyone. It tried to deliver a message and failed. It is up to the upper-layer protocols (TCP) to perform error recovery. In other words, TCP will time-out for that transmission and will resend the data.

IP submits a properly formatted data packet to the destination station and does not expect a status response. Because IP is a connectionless protocol, IP may receive and deliver the data (data sent to the transport layer in the receiving station) in the wrong order from which it was sent, or it may duplicate the data. It is up to the higher-layer protocols (layer 4 and above) to provide error recovery procedures. IP is part of the network delivery system. It accepts data and formats it for transmission to the data-link layer. (Remember, the data-link layer provides the access methods to transmit and receive data from the cable plant.) IP also retrieves data from the data link and presents it to the requesting upper layer.

IP will add its control information, specific to the IP layer only, to the data received by the upper layer (transport layer). Once this is accomplished, it will inform the data link (layer 2) that it has a message to send to the network. The unit of information that IP transfers is known as a *datagram*. This datagram may be transferred over high-speed networks (Ethernet, Token Ring, FDDI). When it is transmitted over these networks, it will be called a *packet*. For simplicity, consid-

ering the primary high-speed networks, it is most important to remember that IP resides in (the data-link layer).

The IP protocol knows is that it must get to the data receiver (TCP or UDP) and try to deliver it.

The IP protocol uses mechanisms on both sides to receive packets, and an IP packet may be broken. Look at the control information to be shown. Refer to the protocol related in an Ethernet II header in the packet.

The other part of the packet header for IP is the functions of the IP layer. Information in the header is:

The fields are:

VERSION. Defines the network station. Version 4.

HLLEN. The length of the header. Not all the fields are measured in the same way. The header is 20 bytes. There are 160 bits/32 bits per byte. Variable in length.

Service Type.

Precedence

Precedence. The priority of the and up to 7 (normal) priority's application. The datagram. T (through) and R (reliability) take. This field

D bit—request

T bit—request

R bit—request

ering the primary focus of the book is network protocols over high-speed networks, datagrams and packets will be synonymous. It is important to remember that IP presents datagrams to its lower layer (the data-link layer).

The IP protocol does not care what kind of data is in the packet. All it knows is that it must apply some control information, called an IP header, to the data received from the upper-layer protocol (presumably TCP or UDP) and try to deliver it to some station on the network or internet.

The IP protocol is not completely without merit. It does provide mechanisms on how hosts and routers should process transmitted or received packets, or when an error should be generated and when an IP packet may be discarded. To understand the IP functionality, a brief look at the control information it adds (the IP header) to the packet will be shown. Refer to Fig. 6.3. This figure shows the IP header encapsulated in an Ethernet frame. This is shown to indicate the position of the header in the packet.

The other part of Fig. 6.3 is the IP header. This shows the standard packet header for an IP datagram. The following text will study the functions of the IP datagram delivery through a look into the header information in the IP datagram.

The fields are defined as follows:

VERS. Defines the current version of IP implemented by the network station. Version 4 is the latest version.

HLEN. The length of the IP header (all fields but the IP data field). Not all the fields in the IP header need to be used. The field is measured in the amount of 32-bit words. The shortest IP header will be 20 bytes. Therefore, this field would contain a 5 (20 bytes = 160 bits; 160 bits/32 bits = 5). This field is necessary, for the header can be variable in length depending on the field called options.

Service Type. Which is further divided:

Precedence	D	T	R	Unused
------------	---	---	---	--------

Precedence. This field may have an entry of 0 (normal precedence) and up to 7 (network control), which allows the transmitting station's application to indicate to the IP layer the priority of sending the datagram. This is combined with the D (delay), T (throughput), and R (reliability) bits. These bits indicate to a router which route to take. This field is known as a Type of Service (TOS) identifier.

D bit—request low delay when set to a 1

T bit—request high throughput

R bit—request high reliability

ering the primary focus of the book is network protocols over high-speed networks, datagrams and packets will be synonymous. It is important to remember that IP presents datagrams to its lower layer (the data-link layer).

The IP protocol does not care what kind of data is in the packet. All it knows is that it must apply some control information, called an IP header, to the data received from the upper-layer protocol (presumably TCP or UDP) and try to deliver it to some station on the network or internet.

The IP protocol is not completely without merit. It does provide mechanisms on how hosts and routers should process transmitted or received packets, or when an error should be generated and when an IP packet may be discarded. To understand the IP functionality, a brief look at the control information it adds (the IP header) to the packet will be shown. Refer to Fig. 6.3. This figure shows the IP header encapsulated in an Ethernet frame. This is shown to indicate the position of the header in the packet.

The other part of Fig. 6.3 is the IP header. This shows the standard packet header for an IP datagram. The following text will study the functions of the IP datagram delivery through a look into the header information in the IP datagram.

The fields are defined as follows:

VERS. Defines the current version of IP implemented by the network station. Version 4 is the latest version.

HLEN. The length of the IP header (all fields but the IP data field). Not all the fields in the IP header need to be used. The field is measured in the amount of 32-bit words. The shortest IP header will be 20 bytes. Therefore, this field would contain a 5 (20 bytes = 160 bits; 160 bits/32 bits = 5). This field is necessary, for the header can be variable in length depending on the field called options.

Service Type. Which is further divided:

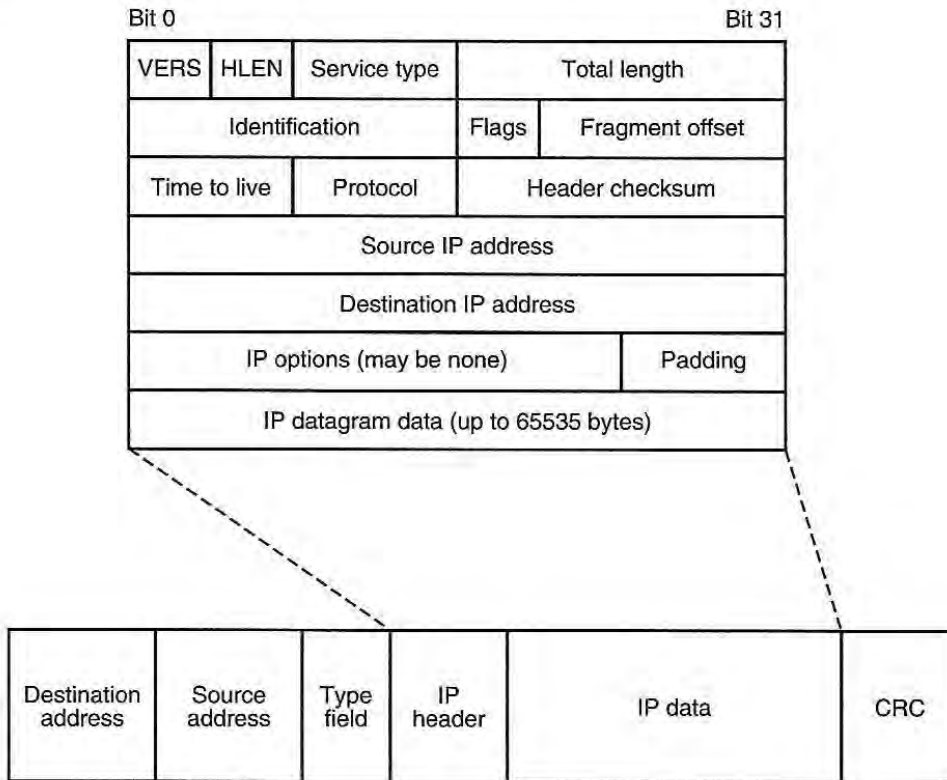
Precedence	D	T	R	Unused
------------	---	---	---	--------

Precedence. This field may have an entry of 0 (normal precedence) and up to 7 (network control), which allows the transmitting station's application to indicate to the IP layer the priority of sending the datagram. This is combined with the D (delay), T (throughput), and R (reliability) bits. These bits indicate to a router which route to take. This field is known as a Type of Service (TOS) identifier.

D bit—request low delay when set to a 1

T bit—request high throughput

R bit—request high reliability



Ethernet frame

Figure 6.3 IP header encapsulated in an Ethernet frame.

For example, if there is more than one route to a destination, the router could read this field to pick a route. This becomes important in the OSPF* routing protocol, which is the first IP routing protocol to take advantage of this. If the transaction to take place is a file transfer, you may want to set the bits to 0 0 1 to indicate that you do not need low delay or high throughput, but you would like high reliability. TOS fields are set by applications (i.e., TELNET or FTP) and not routers. Routers only read this field. Routers do not set this field. Based on the information read, routers will select the optimal path for the datagram. It is up to the TCP/IP application running on a host to set these bits before transmitting the packet on the network. It does require a router to maintain multiple routing tables—one for each type of service.

* Open Shortest Path First (OSPF) is a routing protocol that allows devices known as routers to determine the topology of the TCP/IP internet. With this, routers will be able to forward packets correctly to their destination. OSPF will not be covered in this chapter. Another routing protocol known as Routing Information Protocol (RIP) will be covered later.

Total length. This (this field allots for program may be 65535)

Fragmentation. There one network may to sider transmitting a supports 4472 bytes net LAN (which sup sion size). A TCP/D packet into smaller p tion, but what if the tiple types of media. Fragmenting a pack sion or heterogeneo the IP layer. The fol

Identification, flags, forwarded datagram can run on top of al ferent networks, the time may vary on th total of 1518 bytes around a 17800-byte FDDI allows for a 4 the largest frame p between all these ne

Each IP header f identical. The identi belong together. It datagrams do not g field and the source together.

The flags field is u

1. Whether more fra be sent for that d
2. Whether or not to

Fragmenting is by f works to be traver

* The maximum num 16/4 Token Ring).

† Taken from IBM Tok

Total length. This is the length of the datagram measured in bytes (this field allots for 16 bits, meaning the data area of the IP datagram may be 65535 bytes in length).

Fragmentation. There may be times when a packet transmitted from one network may be too large to transmit on another network. Consider transmitting a frame from a Token Ring network (which typically supports 4472 bytes as the maximum transmission size) to an Ethernet LAN (which supports only 1518 bytes as the maximum transmission size). A TCP/IP router must be able to "fragment" the larger packet into smaller packets. TCP will set up packet sizes for a connection, but what if the two communicating stations are separated by multiple types of media, each supporting different transmission sizes? Fragmenting a packet into smaller packets suitable for LAN transmission or heterogeneous LAN routing is another task accomplished by the IP layer. The following fields are used to accomplish this.

Identification, flags, fragment offset. These indicate how to fragment a forwarded datagram that is too large for the attached network. TCP/IP can run on top of almost any data link. When trying to send data to different networks, the maximum size of the data that may be sent at one time may vary on those networks. Ethernet holds its packet size to a total of 1518 bytes (including all headers). Token Ring allows for around a 17800-byte* data size (16 Mbps, 4472 bytes for 4 Mbps) and FDDI allows for a 4472-byte data size.† Any of the networks may have the largest frame passing on it. IP allows for the exchange of data between all these networks via its ability to fragment packets.

Each IP header from each of the fragmented datagrams is almost identical. The identification field indicates which datagram fragments belong together. It identifies a group to which datagrams belong so datagrams do not get mismatched. The receiving IP layer uses this field and the source IP address to identify which fragments belong together.

The flags field is used to indicate the following:

1. Whether more fragments are to arrive or whether no more data is to be sent for that datagram (no more fragments)
2. Whether or not to fragment a datagram (a don't-fragment bit)

Fragmenting is by far the most important feature to notice if the networks to be traversed employ different frame sizes. Readers who

* The maximum number of bytes is typically 4472 for ring-type circuits (FDDI and 16/4 Token Ring).

† Taken from *IBM Token Ring Architecture Reference Manual*, 1989.

understand bridges know that bridges do not have this capability. If a bridge receives a packet that is too large for the forwarded network, as mandated by the IEEE 802.1d, it will drop the packet and notify no one that it has done so. Higher-level protocols will time-out the packet and will respond accordingly. Once a session is established, most protocols have the capability to negotiate the maximum packet size that each of the stations may handle and, therefore, it will not effect bridge operation.

The total length and the fragment offset fields IP can reconstruct a datagram and deliver it to the upper-layer software. The total length field indicates the total length of the original packet, and the offset field indicates to the node that is reassembling the packet the offset from the beginning of the packet. It is at this point that the data will be placed in the data segment to reconstruct the packet.

Time to live (TTL). There are error conditions that may occur that would cause a packet to endlessly loop between routers on the internet. The initial entry is set by the originator of the packet. Time to live is a field that is used by routers to ensure that a packet does not endlessly loop around the network. This field (currently defined as the number of seconds) is set at the transmitting station and then, as the datagram passes through each router, it will be decremented. With the speed of today's routers, the usual decrement is 1. One algorithm is that the receiving router will notice the time when a packet arrived, and then, when it is forwarded, the router will decrement the field by the number of seconds the datagram sat in a queue waiting for forwarding. Not all algorithms work this way. A minimum decrement will always be 1. The router that decrements this field to 0 will discard the packet and inform the originator of the datagram that it cannot be forwarded.

The time-to-live field may also be set to a certain time (i.e., initialized to a low number like 64) to ensure that a packet stays on the network for only a set time. Some routers allow the network administrator to set a manual entry to decrement. This field may contain any number from 0 to 255.

Protocol field. This field is used to indicate which higher-level protocol sent the frame and which receiving protocol should get the frame (i.e., TCP or UDP). There are many protocols that may reside on top of IP. IP is not specific as to the protocol that runs on top of it. Currently, the most common transport implementations are TCP and UDP (explained in Section 2). In order for IP to know how to correctly deliver the packet to the correct entity above it is the purpose of this field. It will be explained in more detail in Section 2. If the protocol field is set to TCP, the packet will be handed to the TCP process for further frame processing. The same is true if the frame is set to UDP.

Checksum. This is beyond the scope of integrity of the header the IP data field and When the receiving number. If the two CRC header and the pack of this as a fancy p router, each router TTL field is changed

The IP options field. cally, it contains in Token Ring), tracing routers, and military entries, please refer These fields may on variable length head

The fields of IP sou important, for user workstation or trying domain name server final destination IP the IP address of the All hosts on an IP addressing is extren

IP addresses, subnet Resolution Protocol (

The ideas and con devised before any were developed. Hos (like Ethernet or T started in the 1970s being developed), a low-speed point-to-p Token Ring control which identify the c to Chap. 2, "Ether scheme to identify implemented. The a called the 32-bit IP

Checksum. This is a Cyclic Redundancy Check of 16 bits. This is beyond the scope of this book, but the idea behind it is to ensure the integrity of the header. A CRC number is generated from the data in the IP data field and placed into this field by the transmitting station. When the receiving station reads the data, it will compute a CRC number. If the two CRC numbers do not match, there is an error in the header and the packet will be discarded. Stretching it, you may think of this as a fancy parity check. As the datagram is received by each router, each router will recompute the checksum. This is because the TTL field is changed by each router the datagram traverses.

The IP options field. This is also beyond the scope of this book. Basically, it contains information on source routing (nothing to do with Token Ring), tracing a route, time stamping the packet as it traverses routers, and military security entries. For more information on these entries, please refer to the TCP/IP references at the end of this book. These fields may or may not be in the header (which allows for the variable length header).

The fields of IP source and destination address. These are particularly important, for users will be most aware of this when starting their workstation or trying to access other stations without the use of a domain name server or an up-to-date host file. These fields indicate the final destination IP address that the packet should be delivered to and the IP address of the station that originally transmitted the packet.

All hosts on an IP internet will be identified by these addresses. IP addressing is extremely important and a full discussion follows.

IP addresses, subnetting, and the Address Resolution Protocol (ARP)

The ideas and concepts that evolved the protocol of TCP/IP were devised before any data-link protocols of Ethernet and Token Ring were developed. Hosts were not attached to a local high-speed network (like Ethernet or Token Ring). Initial work on the TCP/IP protocol started in the 1970s (about the same time the Ethernet protocol was being developed), and hosts communicated with each other through low-speed point-to-point serial lines (telephone lines). Ethernet and Token Ring controllers have addresses known as MAC addresses, which identify the controller on the LAN. (For more information refer to Chap. 2, "Ethernet and Token Ring.") Therefore, an addressing scheme to identify TCP/IP hosts and where they were located was implemented. The addressing scheme used to identify these hosts is called the 32-bit IP address. This is also known as a protocol address.

Addressing's purpose was to allow IP to communicate between hosts on a network or on an internet. IP addresses identify both a particular node and a network number where the particular node resides on an internet. IP addresses are 32 bits long, separated into four fields of 1 byte each. This address can be expressed in decimal, octal, hexadecimal, and binary. The most common IP address form is written in decimal as is known as the *dotted decimal notation* system. This format will be shown in a moment.

There are two ways that an IP address is assigned. It all depends on your connection. If you have a connection to the Internet, the network portion of the address is assigned through a central authority known as the Network Information Center (NIC). Their address and phone number are located on p. 245. If your network will not have a connection to the Internet, the IP addresses for your internet can be locally assigned by the network administrator of your network.*

Individual host IDs are not assigned by the NIC and will always be assigned by the local network administrator of the network whether the network attachment has access to the Internet or not. When the NIC assigns your network address, it will be the network number only—the host portion of the address is locally assigned.

Whereas XNS uses the 48-bit MAC address as its host address, IP was developed before high-speed local LANs and, therefore, it has its own numbering scheme.

IP addressing was later adapted to the physical-layer addressing of Ethernet and Token Ring and will be discussed in a moment.

IP address format. Each host on a TCP/IP network is uniquely identified at the IP layer with an address that takes the form of <netid, hostid>. The address is not really separated and is read as a whole. The whole address is always used to identify a host. There is not a separation between the fields. In fact, when an IP address is written, it is hard to tell the distinction between the two fields without knowing how to separate them.

The following shows the generalized format of an IP address:

<Network Number, Host Number>

IP classes. 128.4.70.9 is an example of an IP address. When looking at this address, it is hard to tell which is the network number and which is the host number. In order to understand how this is accom-

* IP addresses may be local to your network. That is, commercial businesses may assign their own addresses on their network. But, in order to connect to the Internet, the address connecting to the Internet must be assigned by the NIC.

plished, let's look at the structure of an IP address.

Byte	Bit
1	
xxx	x
<Network	Number

As shown in the diagram, the first byte sent an IP address is the network portion of the address (the reason for this is that the network portion of the address is sent first).

IP addresses are divided into three classes: A, B, and C are unicast addresses. Class D is a special type of address used for multicast routing updates.

For those trying to understand IP addresses, also know the difference between decimal and hexadecimal. The most common form is decimal.

Classes A, B, and C are unicast addresses. 6.4a and b, we can see that a host or internet address. Since the length of the address is a simple method of identifying a class of address.

IP class identification. The network ID by using the first field (the first three bytes). Refer to Figure 6.4a for an equivalent. If the first bit of the address is a 0, the first field is the next bit. If the next bit is a 1 and the third bit is a 0, the address is in the second field. If the third bit is a 1, the address is in the third field. This is how IP addresses are classified.

Class A. Class A addresses have the network number in the first byte of the address. If the first bit of the address is a 0, the last three bytes

plished, let's look first at the how IP addresses are divided. The structure of an IP address takes the following form:

Byte	Byte	Byte	Byte	
1	2	3	4	
xxx	xxx	xxx	xxx	Decimal
<Network	Number,	Host	Number>	

As shown in the this table, there are four bytes that are used to represent an IP address. The network number can shift from the first byte to the second byte to the third byte. The same can happen to the host portion of the address. xxx represents a decimal number from 0 to 255 (the reason for three x's), in decimal.

IP addresses are divided into five classes: A, B, C, D, and E. Classes A, B, and C are used to represent host and network addresses. Class D is a special type of address used for multicasting (for example, OSPF routing updates use this type of address). Class E is reserved.

For those trying to figure out this addressing scheme, it is best if you also know the binary numbering system and are able to convert between decimal and binary. Finally, IP addresses are sometimes expressed in hexadecimal and it is helpful to know. The most common form is decimal. This book shows most addresses in binary and decimal.

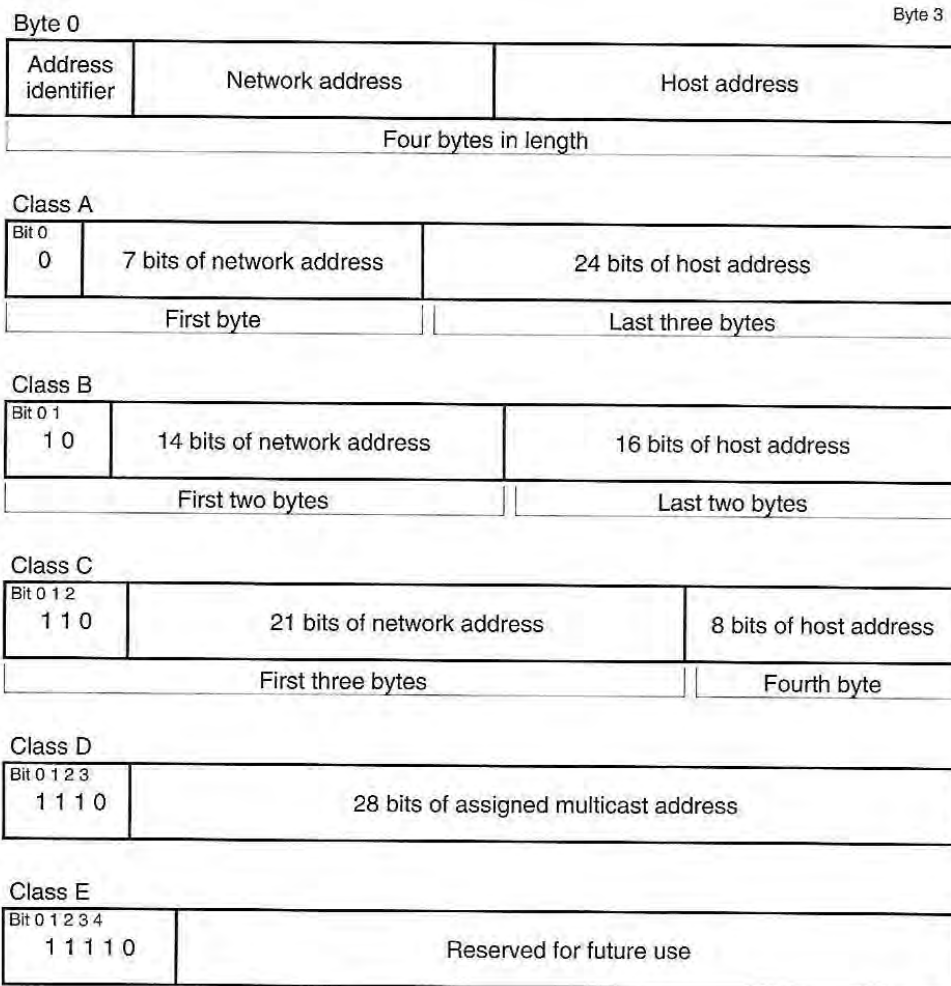
Classes A, B, and C are the most commonly used. Referring to Fig. 6.4a and b, we can see how the classes are actually defined. How does a host or internet device determine which address is of which class? Since the length of the network ID is variable (dependent on the class), a simple method was devised to allow the software to determine the class of address and, therefore, the length of the network number.

IP class identification. The IP software will determine the class of the network ID by using a simple method of reading the first bit(s) in the first field (the first byte) of every packet. IP addresses contain four (4) bytes. Refer to Fig. 6.4a. The IP address is broken down into its binary equivalent. If the first bit is a 0 of byte 0, it is automatically a class A address. If the first bit is a 1, then the protocol mandates reading the next bit. If the next bit is a 0, then it is a class B address. If the next bit is a 1 and the third bit is a 0, it is a class C address. If the third bit is a 1, the address is a class D address and is reserved for multicast addresses.

Class A. Class A addresses use only the first of the four bytes for the network number. It is identified by the first bit in the first byte of the address. If this first bit is a 0, then it identifies a class A address. The last three bytes are used for the host portion of the address. Class

An addressing scheme allows for 126 networks (using only the first byte) with up to 16 million hosts on each of the 126 network numbers (the last three bytes, 24 bits in binary) and are used for networks with a large number of hosts attached to each logical network. Why only 126 networks when there are eight bits? First, 127.x (01111111 binary) is reserved for a loop-back function. It cannot be assigned to identify a network number.

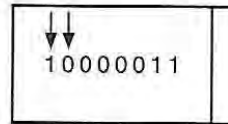
Second, since the first bit is reserved and will always be set to a 0, it leaves seven bits to assign to a network number (01111111). Seven 1s in binary is 127 in decimal. The last three fields we do not care about, for



Identifier is included as part of the address

(a)

Figure 6.4 (a) 32-bit Internet addressing scheme.



131

In binary, bit 0 and
In decimal, the first

Figure 6.4 (b) Dotted

they are assigned the range of 1 to 127 in the range of 1 to 127.

Class A addresses are in the form of <net>.host.host>, bytes

Class B. Class B addresses are in the form of <net>.net>.host>, bytes

for the network number. The network number is identified by the first two bytes, then the algorithm will identify a class.

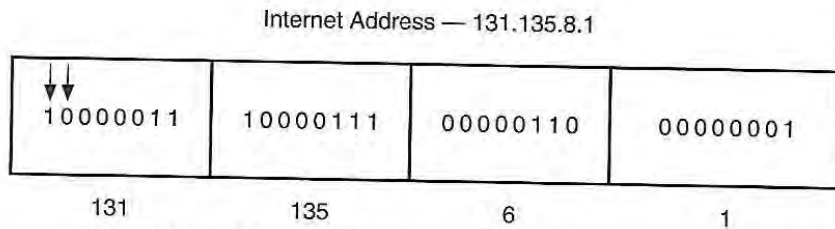
This allows for a large number of addresses (host.host), with a large number of addresses (net.net.11111111).

to identify the class type. The limited address range relates to 128 to 127 in the first field. Since the first field is free to use all addresses for network numbers (1 to 255 in the first field), 1 to 255 (in the first field), 1 to 255 (in the first field), 1 to 255 (in the first field), 1 to 255 (in the first field).

addresses. Class B addresses are in the form of <net>.net>.host>, bytes

Class B addresses are in the form of <net>.net>.host>, bytes

Class B addresses are in the form of <net>.net>.host>, bytes



In binary, bit 0 and 1 are 1 and 0, indicating Class B
 In decimal, the first byte is in the range of 128–191, indicating Class B

First field indicates the class

Class A
 1-126.host.host.host

Class B
 128-191.network number.host.host

Class A
 192-223.network.network.host
 (b)

Figure 6.4 (b) Dotted decimal notation.

they are assigned to hosts. Class A network addresses will always be in the range of 1 to 126 (127 is reserved), with each of the last three bytes in the range of 1 to 254. The last three bytes are assigned locally.

Class A addresses take the 4-byte form <network number.host.host.host>, bytes 0, 1, 2, and 3.

Class B. Class B addresses use the first two bytes of the four bytes for the network number and the last two fields for the host number. It is identified by the first two bits of the first byte. If the first bit is a 1, then the algorithm checks the second bit. If the second bit is a 0, this will identify a class B address.

This allows for 16,384 network numbers (10111111.11111111.host.host), with each network capable of supporting 65,354 hosts (net.net.11111111.11111110). Since it reserves the first two bits to identify the class type (in binary, a 10xxxxxx in the first field), there are limited address numbers that may be used in the first field. This translates to 128 to 191 (in decimal) as the allowable network numbers in the first field. Since the first field identifies the class, the second field is free to use all eight bits, and can range from 1 to 255. The total range for network numbers for class B addresses is 128 to 191 (in the first field), 1 to 255 (in the second field) and xxx.xxx (x represents the host ID) in the third and fourth fields. This is the most popular class of addresses.

Class B addresses take the form <network number.network number.host.host>, bytes 0, 1, 2, and 3.

Class C. Class C addresses use the first three out of four fields of the address for the network number and the last field for the host number. A class C address is identified by the first three bits of the first field. If the first and second bits are a 1 and the third bit is a 0, this will identify a class C address (110xxxxx). Since the first three bits in the first field will always be a 110xxxxx, the allowable network range is 192–223 in the first field. All of the bits in the second and third fields are allowed to be used. Therefore, the whole allowable range for class C network addresses is 192 to 223 (in the first field), 1 to 255 (in the second field), and 1 to 254 (in the third field). The last field will range from 1 to 255. This allows 2 million network numbers, each capable of supporting 254 hosts (all 0s and all 1s are reserved). Class C addresses allow only 254 hosts per network number. Notice that the largest number in the first field may go up to 223. Any number over 223 in the first field will indicate a class D address. Class D addresses are reserved as multicast addresses and are not used as individual addresses.

Class C addresses are the most commonly assigned by the NIC. If you can prove a need for a class B address, they will assign one to you. Usually, they assign a class C.

Class C takes the form of <network number.network number.network number.host>, bytes 0, 1, 2, and 3.

For anyone new to this protocol, the easiest way to remember IP addresses and their associated class is this: the *first byte* will always identify the *class address*. A is the *first letter* in the alphabet and therefore a class A network address is only the *first byte* leaving the last three fields for host addressing. B is the *second letter* in the alphabet and therefore the network portion of the address is the first *two bytes* of the address leaving the last two fields for host address. C is the *third letter* in the alphabet and the network portion takes up the first *three bytes* of the address and leaves one field for host addresses. As for remembering which number is associated to which class, the only field that is important is the first field. Memorize the starting network number for each class.

Let's review. Refer to Fig. 6.4a and b. All IP addresses are actually the grouping of four bytes that represents both a network number and host number. This number is usually represented in decimal. With the first bit reserved (set to 0xxxxxxx) in a class A address, the network numbers can range from 1 to 126. Number 127 is reserved as a local loop-back IP address and must not be assigned to a network number and broadcast on the network. With the first two bits reserved in a class B (10xxxxxx) or three bits in a class C (110xxxxx) address, the network numbers for class B range from 128.1.0.0 to 191.255.0.0 and for class C, they range from 192.1.1.0 to 223.255.255.0.

Examples

192.1.1.1

200.6.5.4

150.150.5.6

9.6.7.8

128.1.0.1

Notice that to number is written number display v

For those not f ing and stopping

Class A 1 to

Class B 128

Class C 192

Restrictions

1. Addresses can set to 1111. Th network classif
2. The class A ad loop-back funct communicate t send packets ou be set to 1. Rou discard the pac
3. The bits that de not be set to all cial address to particular netw the network (no pret the datagr: decimal), this i received by all n not forward a l

* The exception to th

Examples

192.1.1.1	node assigned with a host ID of 1, located on a class C network of 192.1.1.0
200.6.5.4	node assigned with a host ID of 4, located on a class C network of 200.6.5.0
150.150.5.6	node assigned with a host ID of 5.6, located on a class B network of 150.150.0.0
9.6.7.8	node assigned with a host ID of 6.7.8, located on a class A network of 9.0.0.0
128.1.0.1	node assigned with a host ID of 0.1, located on a class B network of 128.1.0.0

Notice that to represent a network number only, only the network number is written. The host field will be set to 0. This type of network number display will become apparent when looking at routing tables.

For those not familiar with binary, you need to memorize the starting and stopping points of the first byte of an IP address:

Class A	1 to 126 in the first field
Class B	128 to 191 in the first field
Class C	192 to 223 in the first field

Restrictions

1. Addresses cannot have the first four highest bits (in the first field) set to 1111. This is reserved for class E networks only (a reserved network classification).
2. The class A address of 127.x is for a special function known as the *loop-back function*. This is provided so that processes which need to communicate through TCP that reside on the same host will not send packets out to the network. x is usually set to 0, although it can be set to 1. Routers that receive a datagram addressed this way will discard the packet.
3. The bits that define the network and host portion of the address cannot be set to all 1s to indicate an individual address.* This is a special address to represent a broadcast packet to all hosts on that particular network. Broadcast addresses indicate that every host on the network (not necessarily the internet) should receive and interpret the datagram. If each byte of the IP address is all 1s (all 255 in decimal), this is known as a *limited broadcast* (the packet will be received by all network stations on the local network). Routers will not forward a limited broadcast datagram. This takes the form of

* The exception to this rule is the subnet mask. This concept is explained later.

255.255.255.255. Routers, explained later, use this type of address. They use this to update other routers with network number and hop-count updates. Routers are explained later.

4. Another form of broadcast is when the network portion of the address is set to a specified network address but the host portion of the address is set to all 1s. This is known as *directed broadcast*. Routers will forward this type of datagram. An example of this is 128.1.255.255. This signifies a broadcast address to all stations on network number 128.1.0.0.

Other types of broadcast formats will be shown in the subnet section of this chapter.

5. Any address with all 0s in the network portion of the address is meant to represent "this" network. For example, 0.0.0.120 is meant as host number 120 on "this" network (the network from which it originated).
6. There is an old form of broadcasting known as the *all-0s broadcast*. This will take the form of 0.0.0.0. This form is discouraged from being used. 0.0.0.0 is used to indicate a default router (explained later).

Class D or multicast addresses are used to send an IP datagram to a group of hosts on a network. This will prove to be very beneficial when routers need to update their neighbor routers (in some routing algorithms). It is a more efficient way of broadcasting to use a multicast address rather than a broadcast address, for the upper-layer software will not always be interrupted every time a broadcast packet arrives. With a multicast address, each individual IP station must be willing to accept the multicast IP address before the transport-layer software will be interrupted. With a broadcast address, all stations will interrupt their upper-layer software no matter what type the packet is.

Remember that the Network Information Center (NIC) assigns all IP addresses for those stations with an attachment to the Internet. No one is allowed to attach to the Internet with their own assigned address (the network number portion of the address). The NIC is the only source for an IP address. You can assign your own IP network numbers if you will *never* have access to the Internet.*

Addresses cannot be out of the 255 (decimal) range for any for the four bytes. Therefore, an address of 128.6.200.655 is not a valid address. Likewise, an address of 420.6.7.900 is not a valid address. Each of the fields must be less than 255 for network and host assignments and the host fields should be 255, or all fields set to 255, if it is a broadcast address.

* There is an exception to this rule, as noted in a previous footnote.

Subnetting. Now
confuse the issue
is an easy way of
portion of the ad
tion of the address

When a site is
administrator de
assigned only on
Why use only on
network into ma
running short of
sparingly.

There is a desir
network into mar
address and have
this many netwo
of the IP address
using a method k

Figure 6.5 sho
address. This tim
mask. In this exa
field following the
a subnet. The cla
tion of the address
any of the host bi
a network) may b
may use all of the
netting. This wo
would indicate a
hosts (63 would i

When doing thi
to identify a subn
subnet is a real
there is one class
take any amount
through 8 bits; th
IP address and m
the network num
<network number
if the address ass
portion would be
netting (assuming
subnet address),
128.1 and subnet

Subnetting. Now that IP address assignment has been shown, let's confuse the issue some more by looking at subnet masks. In short, this is an easy way of assigning some of the bits normally used by the host portion of the address and reassigning these bits to the network portion of the address. This is accomplished for the reasons that follow.

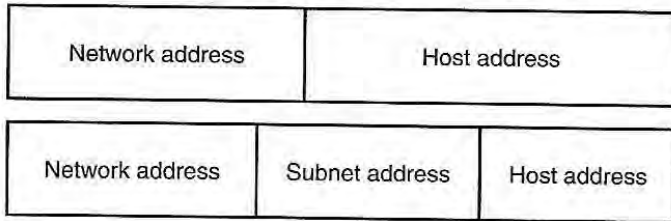
When a site is assigned (by the NIC), a network number or a LAN administrator devises an address scheme; the individual site may be assigned only one network number. This can create many problems. Why use only one network number when the desire is to segment the network into many more manageable network numbers? The NIC is running short of IP network number assignments and assigns them sparingly.

There is a desire among most companies using TCP/IP to break their network into many logical networks. Why use a single class B network address and have 65534 hosts assigned to it? Most sites will not have this many network attachments. Why not use part of the host portion of the IP address and reassign it to a network address? This is possible using a method known as *subnet masking*.

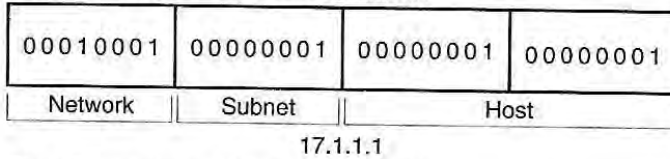
Figure 6.5 shows the three classes of networks, each with an address. This time, each of the addresses has been assigned a subnet mask. In this example of the classes A and B addresses, only the first field following the network ID portion of the address is used to indicate a subnet. The class C address uses the first three bits of the host portion of the address for the subnet. In reality, with any of the addresses, any of the host bits (except for 2 bits, there must be at least one host on a network) may be used for subnetting. For example, a class B address may use all of the third octet and two bits of the fourth octet for subnetting. This would give 1022 possible subnetwork numbers (1023 would indicate a broadcast address), with each network holding 62 hosts (63 would indicate a broadcast).

When doing this, host numbers are being taken away and given back to identify a subnet of a network address. Figure 6.5 depicts this. The subnet is a real network. In looking at the address, it appears that there is one class B address. With subnetting a class B address, we can take any amount of the bits in the third or 6 bits of the fourth byte (1 through 8 bits; they should be contiguous, starting from the left) of the IP address and make it part of the network number (a subnet under the network number). The format of the IP address would now be: <network number | subnetwork number | host number>. For example, if the address assigned to a particular host is 128.1.1.1, the network portion would be 128.1 and the host portion would be 1.1. With subnetting (assuming all eight bits of the third field were consumed for a subnet address), the address would be defined as network number 128.1 and subnet 1, with a host ID of 1.

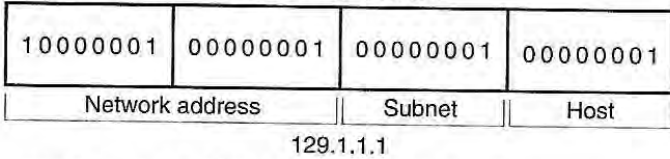
Subnet using 8 bits of the host field (according to the Class)



Class A—network 17, subnet 1, host 1.1



Class B—network 129.1, subnet 1, host 1



Class C—using a 4-bit subnet: subnet 32, host 1

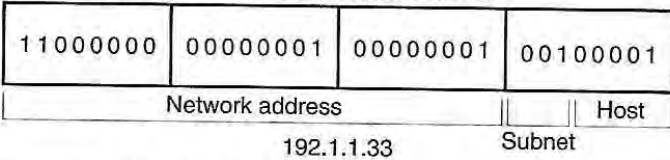
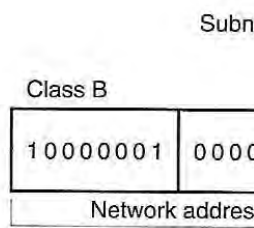


Figure 6.5 IP network addressing.

To illustrate further: If the class B network address of 130.1.0.0 is assigned to a site, it may be subnetted into the following: 130.1.xxx.0. x here indicates the decimal-formatted field that may be consumed for subnetting. The field may be subnetted using any of the bits in the field (translating binary into decimal, you may have one subnet using the most significant bit of the third field or use all the bits in the third field, which would give you 254 subnets with 254 hosts per subnet).

Refer to Fig. 6.6a. Suppose the first five bits (starting from the left; they should start from the left and remain contiguous going to the right) are reserved in the third field for assigning subnet numbers. Convert those first five bits of that octet to binary. All five of those bits are now assigned to the subnet number and may not be used for host IDs. To assign a unique subnet number, one rule must be adhered to. This is that a subnet must not have all 0s or all 1s (they are used for



- 1) All four bytes are v
not subnetted.
- 2) Write out the addr
- 3) Write out the subn
- 4) Logically AND the
- 5) The bits of the out
the network and s

Operation:
10000001 00000001

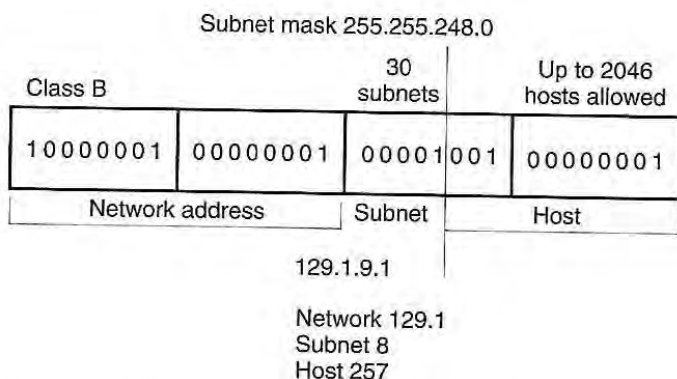
11111111 11111111

10000001 00000001

Figure 6.6 (a) Subnetting

broadcasting). App
any combination of

To identify the s
carefully. As show
and subnet portio
the subnet portio
a 1. In calculating t
into consideration.
binary 8 (the fourth
8. Each subsequent
numbers are 192, 1
is still read as if s
129.1.9.1. But, as s
This is a little trick
works installed. E
using subnetting, y



- 1) All four bytes are written the same as if you are not subnetted.
- 2) Write out the address in binary.
- 3) Write out the subnet mask in binary.
- 4) Logically AND the two.
- 5) The bits of the outcome of the AND operation show the network and subnet.

Operation:

10000001 00000001 00001001	00000001 - Address: 129.1.9.1
11111111 11111111 11111000	00000000 - MASK: 225.225.248.0
10000001 00000001 00001000	00000000 - Network 128.1 Subnet 8

(a) True network: 128.1.8.0

Figure 6.6 (a) Subnetting a class B with 5-bit subnet mask.

broadcasting). Applying this rule, we could have subnet numbers of any combination of the first five bits.

To identify the subnets is a little tricky. Please read this paragraph carefully. As shown in Fig. 6.6a, the vertical line separating the host and subnet portions of the address is the dividing line. The first bit in the subnet portion of the address is set to a 1. The subnet would not be a 1. In calculating the value of the subnet, the whole third field is taken into consideration. Therefore, since that bit is set, it is actually a binary 8 (the fourth bit). Therefore, the first subnet number will be an 8. Each subsequent subnet will be a multiple of 8. Some of these subnet numbers are 192, 128, 96, 64, 56, 32, 24, 16, 8, and so on. The address is still read as if subnetting has not been turned on. The address is 129.1.9.1. But, as shown, this is network 129.1, subnet 8, and host 257. This is a little tricky, but subnetting is found on almost all TCP/IP networks installed. Each field will still never read beyond 255, but by using subnetting, you can have a host number higher than 255.