

UNIX Network Programming

Volume 1

Second Edition

Networking APIs: Sockets and XTI

UNIX 网络编程

卷 1

连网的 API: 套接字与 XTI

第 2 版

W. Richard Stevens

清华大学出版社

Prentice-Hall International, Inc.

(京)新登字 158 号

UNIX network Programming Volume I 2nd ed.
networking APIs: Sockets and XTI/W. Richard Stevens

© 1998 by Prentice Hall, Inc.

Original edition published by Prentice Hall, Inc., a Simon & Schuster Company.

Prentice Hall 公司授权清华大学出版社在中国境内(不包括中国香港特别行政区、澳门地区和台湾地区)独家出版发行本书影印本。

本书任何部分之内容,未经出版者书面同意,不得用任何方式抄袭、节录或翻印。

本书封面贴有 Prentice Hall 激光防伪标签,无标签者不得销售。

北京市版权局著作权合同登记号: 01-98-1086

图书在版编目(CIP)数据

UNIX 网络程序设计:卷 I 第 2 版:英文/史蒂文斯(Stevens, W.R.)著.

- 影印版. - 北京:清华大学出版社,1998.7

(大学计算机教育丛书)

ISBN 7-302-02942-3

I. U... II. 史... III. UNIX 操作系统-程序设计-教材-英文 IV. TP316

中国版本图书馆 CIP 数据核字(98)第 09265 号

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

<http://www.tup.tsinghua.edu.cn>

印刷者:清华大学印刷厂

发行者:新华书店总店北京科技发行所

开本:850×1168 1/32 印张:32.5

版次:1998年7月第1版 1998年12月第2次印刷

书号:ISBN 7-302-02942-3/TP·1554

印数:5001~10000

定价:43.00 元

出版前言

我们的大学生、研究生毕业后,面临的将是一个国际化的信息时代。他们将需要随时查阅大量的外文资料;会有更多的机会参加国际性学术交流活动;接待外国学者;走上国际会议的讲坛。作为科技工作者,他们不仅应有与国外同行进行口头和书面交流的能力,更为重要的是,他们必须具备极强的查阅外文资料获取信息的能力。有鉴于此,在国家教委所颁布的“大学英语教学大纲”中有一条规定:专业阅读应作为必修课程开设。同时,在大纲中还规定了这门课程的学时和教学要求。有些高校除开设“专业阅读”课之外,还在某些专业课拟进行英语授课。但教、学双方都苦于没有一定数量的合适的英文原版教材作为教学参考书。为满足这方面的需要,我们挑选了7本计算机科学方面最新版本的教材,进行影印出版。首批影印出版的6本书受到广大读者的热情欢迎,我们深受鼓舞,今后还将陆续推出新书。希望读者继续给予大力支持。Prentice Hall公司和清华大学出版社这次合作将国际先进水平的教材引入我国高等学校,为师生们提供了教学用书,相信会对高校教材改革产生积极的影响。

清华大学出版社
Prentice Hall 公司

1997.11

Preface

Introduction

Network programming involves writing programs that communicate with other programs across a computer network. One program is normally called the *client* and the other the *server*. Most operating systems provide precompiled programs that communicate across a network—common examples in the TCP/IP world are Web clients (browsers) and Web servers, and the FTP and Telnet clients and servers—but this book describes how to write our own network programs.

We write network programs using an *application program interface* or *API*. We describe two APIs for network programming:

1. sockets, sometimes called “Berkeley sockets” acknowledging their heritage from Berkeley Unix, and
2. XTI (X/Open Transport Interface), a slight modification of the Transport Layer Interface (TLI) developed by AT&T.

All the examples in the text are from the Unix operating system, although the foundation and concepts required for network programming are, to a large degree, operating system independent. The examples are also based on the TCP/IP protocol suite, both IP versions 4 and 6.

To write network programs one must understand the underlying operating system and the underlying networking protocols. This book builds on the foundation of the my other four books in these two areas, and these books are abbreviated throughout this text as follows:

- APUE: *Advanced Programming in the UNIX Environment* [Stevens 1992],
- TCPv1: *TCP/IP Illustrated, Volume 1* [Stevens 1994],
- TCPv2: *TCP/IP Illustrated, Volume 2* [Wright and Stevens 1995], and
- TCPv3: *TCP/IP Illustrated, Volume 3* [Stevens 1996].

This second edition of *UNIX Network Programming* still contains information on both Unix and the TCP/IP protocols, but many references are made to these other four texts to allow interested readers to obtain more detailed information on various topics. This is especially the case for TCPv2, which describes and presents the actual 4.4BSD implementation of the network programming functions for the sockets API (`socket`, `bind`, `connect`, and so on). If one understands the implementation of a feature, the use of that feature in an application makes more sense.

Changes from the First Edition

This second edition is a complete rewrite of the first edition. These changes have been driven by the feedback I have received teaching this material about once a month during 1990–1996, and by following certain Usenet newsgroups during this same time, which lets one see the topics that are continually misunderstood. The following are the major changes with this new edition:

- This new edition uses ANSI C for all examples.
- The old Chapters 6 (“Berkeley Sockets”) and 8 (“Library Routines”) have been expanded into 25 chapters. Indeed this sevenfold expansion (based on a word count) of this material is probably the most significant change from the first to the second edition. Most of the individual sections in the old Chapter 6 have been expanded into an entire chapter with more examples added.
- The TCP and UDP portions from the old Chapter 6 have been separated and we now cover the TCP functions and a complete TCP client–server, followed by the UDP functions and a complete UDP client–server. This is easier for newcomers to understand than describing all the details of the `connect` function, for example, with its different semantics for TCP versus UDP.
- The old Chapter 7 (“System V Transport Layer Interface”) has been expanded into seven chapters. We also cover the newer XTI instead of the TLI that it replaces.
- The old Chapter 2 (“The Unix Model”) is gone. This chapter provided an overview of the Unix system in about 75 pages. In 1990 this chapter was needed because few books existed that adequately described the basic Unix programming interface, especially the differences between the Berkeley and System V implementations that existed in 1990. Today, however, more readers have a fundamental understanding of Unix, so concepts such as a process ID, password files, directories, and group IDs, need not be repeated. (My APUE book is a 700-page expansion of this material for readers desiring additional Unix programming details.)

Some of the advanced topics from the old Chapter 2 are covered in this new edition, but their coverage is moved to where the feature is used. For example, when showing our first concurrent server (Section 4.8) we cover the `fork` function. When we describe how to handle the `SIGCHLD` signal with our concurrent server (Section 5.9), we describe many additional features of Posix signal handling (zombies, interrupted system calls, etc.).

- Whenever possible this text describes the Posix interface. (We say more about the Posix family of standards in Section 1.10.) This includes not only the Posix.1 standard for the basic Unix functions (process control, signals, etc.); but also the forthcoming Posix.1g standard for the sockets and XTI networking APIs, and the 1996 Posix.1 standard for threads.

The term “system call” has been changed to “function” when describing functions such as `socket` and `connect`. This follows the Posix convention that the distinction between a system call and a library function is an implementation detail that is often irrelevant for a programmer.

- The old Chapters 4 (“A Network Primer”) and 5 (“Communication Protocols”) have been replaced with Appendix A covering IP versions 4 (IPv4) and 6 (IPv6), and Chapter 2 covering TCP and UDP. This new material focuses on the protocol issues that network programmers are certain to encounter. The coverage of IPv6 was included, even though IPv6 implementations are just starting to appear, since during the lifetime of this text IPv6 will probably become the predominant networking protocol.

I have found when teaching network programming that about 80% of all network programming problems have nothing to do with network programming, per se. That is, the problems are not with the API functions such as `accept` and `select`, but the problems arise from a lack of understanding of the underlying network protocols. For example, I have found that once a student understands TCP’s three-way handshake and four-packet connection termination, many network programming problems are immediately understood.

The old sections on XNS, SNA, NetBIOS, the OSI protocols, and UUCP have been removed, since it has become obvious during the early 1990s that these proprietary protocols have been eclipsed by the TCP/IP protocols. (UUCP is still popular and is not proprietary, but there is little we can show from a network programming perspective using UUCP.)

- The following new topics are covered in this second edition:
 - IPv4/IPv6 interoperability (Chapter 10),
 - protocol-independent name translation (Chapter 11),
 - routing sockets (Chapter 17),
 - multicasting (Chapter 19),
 - threads (Chapter 23),
 - IP options (Chapter 24),
 - datalink access (Chapter 26),

- client-server design alternatives (Chapter 27),
- virtual networks and tunneling (Appendix B), and
- network program debugging techniques (Appendix C).

Unfortunately, the coverage of the material from the first edition has been expanded so much that it no longer fits into a single book. Therefore at least two additional volumes are planned in the *UNIX Network Programming* series.

- Volume 2 will probably be subtitled *IPC: Interprocess Communication* and will be an expansion of the old Chapter 3, along with coverage of the 1996 Posix.1 real-time IPC mechanisms.
- Volume 3 will probably be subtitled *Applications* and will be an expansion of Chapters 9–18 of the first edition.

Even though most of the networking applications will be covered in Volume 3, a few special applications are covered in this volume: Ping, Traceroute, and `inetd`.

Readers

This text can be used as either a tutorial on network programming, or as a reference for experienced programmers. When used as a tutorial or for an introductory class on network programming, the emphasis should be on Part 2 (“Elementary Sockets,” Chapters 3 through 9) followed by whatever additional topics are of interest. Part 2 covers the basic socket functions, for both TCP and UDP, along with I/O multiplexing, socket options, and basic name and address conversions. Chapter 1 should be read by all readers, especially Section 1.4, which describes some wrapper functions used throughout the text. Chapter 2 and perhaps Appendix A should be referred to as necessary, depending on the reader’s background. Most of the chapters in Part 3 (“Advanced Sockets”) can be read independently of the others in that part.

To aid in the use as a reference, a thorough index is provided, along with summaries on the end papers of where to find detailed descriptions of all the functions and structures. To help those reading topics in a random order, numerous references to related topics are provided throughout the text.

Although the sockets API has become the de facto standard for network programming, XTI is still used, sometimes with protocol suites other than TCP/IP. While the coverage of XTI in Part 4 is smaller than the coverage of sockets in Parts 2 and 3, much of the sockets coverage describes *concepts* that apply to XTI as well as sockets. For example, all of the concepts regarding the use of nonblocking I/O, broadcasting, multicasting, signal-driven I/O, out-of-band data, and threads, are the same, regardless of which API (sockets or XTI) is used. Indeed, many network programming problems are fundamentally similar, independent of whether the program is written using sockets or XTI, and there is hardly anything that can be done with one API that cannot be done with the other. The concepts are the same—just the function names and arguments change.

Source Code and Errata Availability

The source code for all the examples that appear in the book is available from `ftp://ftp.kohala.com/pub/rstevens/unpv12e.tar.gz`. The best way to learn network programming is to take these programs, modify them, and enhance them. Actually writing code of this form is the *only* way to reinforce the concepts and techniques. Numerous exercises are also provided at the end of each chapter, and most answers are provided in Appendix E.

A current errata for the book is also available from my home page, listed at the end of the Preface.

Acknowledgments

Supporting every author is an understanding family, or nothing would ever get written! I am grateful to my family, Sally, Bill, Ellen, and David, first for their support and understanding when I wrote my first book (the first edition of this book), and for enduring this "small" revision. Their love, support, and encouragement helped make this book possible.

Numerous reviewers provided invaluable feedback (totaling 190 printed pages or 70,000 words), catching lots of errors, pointing out areas that needed more explanation, and suggesting alternative presentations, wording, and coding: Ragnvald Blindheim, Jim Bound, Gavin Bowe, Allen Briggs, Joe Douppnik, Wu-chang Feng, Bill Fenner, Bob Friesenhahn, Andrew Gierth, Wayne Hathaway, Kent Hofer, Sugih Jamin, Scott Johnson, Rick Jones, Mukesh Kacker, Marc Lampo, Marty Leisner, Jack McCann, Craig Metz, Bob Nelson, Evi Nemeth, John C. Noble, Steve Rago, Jim Reid, Chung-Shang Shao, Ian Lance Taylor, Ron Taylor, Andreas Terzis, and Dave Thaler. A special thanks to Sugih Jamin and his students in EECS 489 ("Computer Networks") at the University of Michigan who beta tested an early draft of the manuscript during the spring of 1997.

The following people answered email questions of mine, sometimes lots of questions, which improved the accuracy and presentation of the text: Dave Butenhof, Dave Hanson, Jim Hogue, Mukesh Kacker, Brian Kernighan, Vern Paxson, Steve Rago, Dennis Ritchie, Steve Summit, Paul Vixie, John Wait, Steve Wise, and Gary Wright.

A special thanks to Larry Rafsky and the wonderful team at Gari Software for handling lots of details and for many interesting technical discussions. Thank you, Larry, for everything.

Numerous individuals and their organizations went beyond the normal call of duty to provide either a loaner system, software, or access to a system, all of which were used to test some of the examples in the text.

- Meg McRoberts of SCO provided the latest releases of UnixWare, and Dion Johnson, Yasmin Kureshi, Michael Townsend, and Brian Ziel, provided support and answered questions.
- Mukesh Kacker of SunSoft provided access to a beta version of Solaris 2.6 and answered many questions about the Solaris TCP/IP implementation.

- Jim Bound, Matt Thomas, Mary Clouter, and Barb Glover of Digital Equipment Corp. provided an Alpha system and access to the latest IPv6 kits for Digital Unix.
- Michael Johnson of Red Hat Software provided the latest releases of Red Hat Linux.
- Steve Wise and Jessie Haug of IBM Austin provided an RS/6000 system and access to the latest IPv6 for AIX.
- Rick Jones of Hewlett-Packard provided access to a beta version of HP-UX 10.30 and he and William Gilliam answered many questions about it.

Many people helped with the Internet connectivity used throughout the text. My thanks once again to the National Optical Astronomy Observatories (NOAO), Sidney Wolff, Richard Wolff, and Steve Grandi, for providing access to their networks and hosts. Dave Siegel, Justis Addis, and Paul Lucchina answered many questions, Phil Kaslo and Jim Davis provided an MBone connection, Ran Atkinson and Pedro Marques provided a 6bone connection, and Craig Metz provided lots of DNS help.

The staff at Prentice Hall, especially my editor Mary Franz, along with Noreen Regina, Sophie Papanikolaou, and Eileen Clark, have been a wonderful asset to a writer. Many thanks for letting me do so many things "my way."

As usual, but contrary to popular fads, I produced camera-ready copy of the book using the wonderful Groff package written by James Clark. I typed in all 291,972 words using the `vi` editor, created the 201 illustrations using the `gpic` program (using many of Gary Wright's macros), produced the 81 tables using the `gtbl` program, performed all the indexing, and did the final page layout. Dave Hanson's `loom` program and some scripts by Gary Wright were used to include the source code in the book. A set of `awk` scripts written by Jon Bentley and Brian Kernighan helped in producing the final index.

I welcome electronic mail from any readers with comments, suggestions, or bug fixes.

Tucson, Arizona
September 1997

W. Richard Stevens
rstevens@kohala.com
<http://www.kohala.com/~rstevens>

Contents

Preface		xv
Part 1. Introduction and TCP/IP		1
Chapter 1. Introduction		3
1.1	Introduction	3
1.2	A Simple Daytime Client	6
1.3	Protocol Independence	9
1.4	Error Handling: Wrapper Functions	11
1.5	A Simple Daytime Server	13
1.6	Road Map to Client-Server Examples in the Text	16
1.7	OSI Model	18
1.8	BSD Networking History	19
1.9	Test Networks and Hosts	20
1.10	Unix Standards	24
1.11	64-bit Architectures	27
1.12	Summary	28
Chapter 2. The Transport Layer: TCP and UDP		29
2.1	Introduction	29
2.2	The Big Picture	30
2.3	UDP: User Datagram Protocol	32
2.4	TCP: Transmission Control Protocol	32
2.5	TCP Connection Establishment and Termination	34

2.6	TIME_WAIT State	40
2.7	Port Numbers	41
2.8	TCP Port Numbers and Concurrent Servers	44
2.9	Buffer Sizes and Limitations	46
2.10	Standard Internet Services	50
2.11	Protocol Usage by Common Internet Applications	52
2.12	Summary	52

Part 2. Elementary Sockets **55**

Chapter 3. Sockets Introduction **57**

3.1	Introduction	57
3.2	Socket Address Structures	57
3.3	Value-Result Arguments	63
3.4	Byte Ordering Functions	66
3.5	Byte Manipulation Functions	69
3.6	inet_aton, inet_addr, and inet_ntoa Functions	70
3.7	inet_pton and inet_ntop Functions	72
3.8	sock_ntop and Related Functions	75
3.9	readn, writen, and readline Functions	77
3.10	isfdtype Function	81
3.11	Summary	82

Chapter 4. Elementary TCP Sockets **85**

4.1	Introduction	85
4.2	socket Function	85
4.3	connect Function	89
4.4	bind Function	91
4.5	listen Function	93
4.6	accept Function	99
4.7	fork and exec Functions	102
4.8	Concurrent Servers	104
4.9	close Function	107
4.10	getsockname and getpeername Functions	107
4.11	Summary	110

Chapter 5. TCP Client-Server Example **111**

5.1	Introduction	111
5.2	TCP Echo Server: main Function	112
5.3	TCP Echo Server: str_echo Function	113
5.4	TCP Echo Client: main Function	113
5.5	TCP Echo Client: str_cli Function	115
5.6	Normal Startup	115
5.7	Normal Termination	117
5.8	Posix Signal Handling	119
5.9	Handling SIGCHLD Signals	122
5.10	wait and waitpid Functions	124

5.11	Connection Abort before accept Returns	129
5.12	Termination of Server Process	130
5.13	SIGPIPE Signal	132
5.14	Crashing of Server Host	133
5.15	Crashing and Rebooting of Server Host	134
5.16	Shutdown of Server Host	135
5.17	Summary of TCP Example	135
5.18	Data Format	137
5.19	Summary	140
Chapter 6.	I/O Multiplexing: The select and poll Functions	143
6.1	Introduction	143
6.2	I/O Models	144
6.3	select Function	150
6.4	str_cli Function (Revisited)	155
6.5	Batch Input	157
6.6	shutdown Function	160
6.7	str_cli Function (Revisited Again)	161
6.8	TCP Echo Server (Revisited)	162
6.9	pselect Function	168
6.10	poll Function	169
6.11	TCP Echo Server (Revisited Again)	172
6.12	Summary	175
Chapter 7.	Socket Options	177
7.1	Introduction	177
7.2	getsockopt and setsockopt Functions	178
7.3	Checking If an Option Is Supported and Obtaining the Default	178
7.4	Socket States	183
7.5	Generic Socket Options	183
7.6	IPv4 Socket Options	197
7.7	ICMPv6 Socket Option	199
7.8	IPv6 Socket Options	199
7.9	TCP Socket Options	201
7.10	fcntl Function	205
7.11	Summary	207
Chapter 8.	Elementary UDP Sockets	211
8.1	Introduction	211
8.2	recvfrom and sendto Functions	212
8.3	UDP Echo Server: main Function	213
8.4	UDP Echo Server: dg_echo Function	214
8.5	UDP Echo Client: main Function	216
8.6	UDP Echo Client: dg_cli Function	217
8.7	Lost Datagrams	217
8.8	Verifying Received Response	218
8.9	Server Not Running	220
8.10	Summary of UDP example	221

8.11	connect Function with UDP	224	
8.12	dg_cli Function (Revisited)	227	
8.13	Lack of Flow Control with UDP	228	
8.14	Determining Outgoing Interface with UDP	231	
8.15	TCP and UDP Echo Server Using select	233	
8.16	Summary	235	
Chapter 9.	Elementary Name and Address Conversions		237
9.1	Introduction	237	
9.2	Domain Name System	237	
9.3	gethostbyname Function	240	
9.4	RES_USE_INET6 Resolver Option	245	
9.5	gethostbyname2 Function and IPv6 Support	246	
9.6	gethostbyaddr Function	248	
9.7	uname Function	249	
9.8	gethostname Function	250	
9.9	getservbyname and getservbyport Functions	251	
9.10	Other Networking Information	255	
9.11	Summary	256	
Part 3.	Advanced Sockets		259
Chapter 10.	IPv4 and IPv6 Interoperability		261
10.1	Introduction	261	
10.2	IPv4 Client, IPv6 Server	262	
10.3	IPv6 Client, IPv4 Server	265	
10.4	IPv6 Address Testing Macros	267	
10.5	IPV6_ADDRFORM Socket Option	268	
10.6	Source Code Portability	270	
10.7	Summary	271	
Chapter 11.	Advanced Name and Address Conversions		273
11.1	Introduction	273	
11.2	getaddrinfo Function	273	
11.3	gai_strerror Function	278	
11.4	freeaddrinfo Function	279	
11.5	getaddrinfo Function: IPv6 and Unix Domain	279	
11.6	getaddrinfo Function: Examples	282	
11.7	host_serv Function	284	
11.8	tcp_connect Function	285	
11.9	tcp_listen Function	288	
11.10	udp_client Function	293	
11.11	udp_connect Function	295	
11.12	udp_server Function	296	
11.13	getnameinfo Function	298	
11.14	Reentrant Functions	300	
11.15	gethostbyname_r and gethostbyaddr_r Functions	303	

11.16	Implementation of <code>getaddrinfo</code> and <code>getnameinfo</code> Functions	305
11.17	Summary	328
Chapter 12.	Daemon Processes and <code>inetd</code> Superserver	331
12.1	Introduction	331
12.2	<code>syslogd</code> Daemon	332
12.3	<code>syslog</code> Function	333
12.4	<code>daemon_init</code> Function	335
12.5	<code>inetd</code> Daemon	339
12.6	<code>daemon_inetd</code> Function	344
12.7	Summary	346
Chapter 13.	Advanced I/O Functions	349
13.1	Introduction	349
13.2	Socket Timeouts	349
13.3	<code>recv</code> and <code>send</code> Functions	354
13.4	<code>readv</code> and <code>writew</code> Functions	357
13.5	<code>recvmsg</code> and <code>sendmsg</code> Functions	358
13.6	Ancillary Data	362
13.7	How Much Data Is Queued?	365
13.8	Sockets and Standard I/O	366
13.9	T/TCP: TCP for Transactions	369
13.10	Summary	371
Chapter 14.	Unix Domain Protocols	373
14.1	Introduction	373
14.2	Unix Domain Socket Address Structure	374
14.3	<code>socketpair</code> Function	376
14.4	Socket Functions	377
14.5	Unix Domain Stream Client–Server	378
14.6	Unix Domain Datagram Client–Server	379
14.7	Passing Descriptors	381
14.8	Receiving Sender Credentials	390
14.9	Summary	394
Chapter 15.	Nonblocking I/O	397
15.1	Introduction	397
15.2	Nonblocking Reads and Writes: <code>str_cli</code> Function (Revisited)	399
15.3	Nonblocking <code>connect</code>	409
15.4	Nonblocking <code>connect</code> : Daytime Client	410
15.5	Nonblocking <code>connect</code> : Web Client	413
15.6	Nonblocking <code>accept</code>	422
15.7	Summary	424
Chapter 16.	<code>ioctl</code> Operations	425
16.1	Introduction	425
16.2	<code>ioctl</code> Function	426
16.3	Socket Operations	426

16.4	File Operations	427	
16.5	Interface Configuration	428	
16.6	get_ifi_info Function	429	
16.7	Interface Operations	439	
16.8	ARP Cache Operations	440	
16.9	Routing Table Operations	442	
16.10	Summary	443	
Chapter 17.	Routing Sockets		445
17.1	Introduction	445	
17.2	Datalink Socket Address Structure	446	
17.3	Reading and Writing	447	
17.4	sysctl Operations	454	
17.5	get_ifi_info Function	459	
17.6	Interface Name and Index Functions	463	
17.7	Summary	467	
Chapter 18.	Broadcasting		469
18.1	Introduction	469	
18.2	Broadcast Addresses	470	
18.3	Unicast versus Broadcast	472	
18.4	dg_cli Function Using Broadcasting	475	
18.5	Race Conditions	478	
18.6	Summary	486	
Chapter 19.	Multicasting		487
19.1	Introduction	487	
19.2	Multicast Addresses	487	
19.3	Multicasting versus Broadcasting on A LAN	490	
19.4	Multicasting on a WAN	493	
19.5	Multicast Socket Options	495	
19.6	mcast_join and Related Functions	499	
19.7	dg_cli Function Using Multicasting	502	
19.8	Receiving Mbone Session Announcements	504	
19.9	Sending and Receiving	507	
19.10	SNTP: Simple Network Time Protocol	510	
19.11	SNTP (Continued)	515	
19.12	Summary	528	
Chapter 20.	Advanced UDP Sockets		531
20.1	Introduction	531	
20.2	Receiving Flags, Destination IP Address, and Interface Index	532	
20.3	Datagram Truncation	539	
20.4	When to Use UDP Instead Of TCP	539	
20.5	Adding Reliability to a UDP Application	542	
20.6	Binding Interface Addresses	553	
20.7	Concurrent UDP Servers	557	
20.8	IPv6 Packet Information	560	
20.9	Summary	562	

Chapter 21.	Out-of-Band Data	565
21.1	Introduction	565
21.2	TCP Out-of-Band Data	565
21.3	socketmark Function	572
21.4	TCP Out-of-Band Data Summary	580
21.5	Client-Server Heartbeat Functions	581
21.6	Summary	586
Chapter 22.	Signal-Driven I/O	589
22.1	Introduction	589
22.2	Signal-Driven I/O for Sockets	590
22.3	UDP Echo Server Using SIGIO	592
22.4	Summary	598
Chapter 23.	Threads	601
23.1	Introduction	601
23.2	Basic Thread Functions: Creation and Termination	602
23.3	str_cli Function Using Threads	605
23.4	TCP Echo Server Using Threads	607
23.5	Thread-Specific Data	611
23.6	Web Client and Simultaneous Connections (Continued)	620
23.7	Mutexes: Mutual Exclusion	622
23.8	Condition Variables	627
23.9	Web Client and Simultaneous Connections (Continued)	631
23.10	Summary	633
Chapter 24.	IP Options	635
24.1	Introduction	635
24.2	IPv4 Options	635
24.3	IPv4 Source Route Options	637
24.4	IPv6 Extension Headers	645
24.5	IPv6 Hop-by-Hop Options and Destination Options	645
24.6	IPv6 Routing Header	649
24.7	IPv6 Sticky Options	653
24.8	Summary	654
Chapter 25.	Raw Sockets	655
25.1	Introduction	655
25.2	Raw Socket Creation	656
25.3	Raw Socket Output	657
25.4	Raw Socket Input	659
25.5	Ping Program	661
25.6	Traceroute Program	672
25.7	An ICMP Message Daemon	685
25.8	Summary	702