大学计算机教育丛书（影印版）

UNIX NETWORK PROGRAMMING

Networking APIs: **Sockets and XTI**

Volume 1

SECOND EDITION

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

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

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

Function and Macro Definitions

(**Bold** page numbers indicate source code implementation)

accept 99

bcmp 69
bcopy 69
bind 91
bzero 69

close 107
closelog 334
CMSG_*xxx* 364
connect 89
connect_nonb **411**
connect_timeo **350**

daemon_inetd **344**
daemon_init **336**
dg_send_recv **547**

endnetconfig 785
endnetpath 786
err_doit **922**
err_dump **922**
err_msg **922**
err_quit **922**
err_ret **922**
err_sys **922**
exec*xxx* 103

fcntl 206
fork 102
freeaddrinfo 279
free_ifi_info **439**

gai_strerror 278
getaddrinfo 274
gethostbyaddr 248
gethostbyaddr_r 304
gethostbyname 241
gethostbyname2 246
gethostbyname_r 304
gethostname 251
get_ifi_info **434, 460**
getmsg 854
getnameinfo 298
getnetconfig 785
getnetpath 786
getpeername 108
getpmsg 855
getservbyname 251
getservbyport 252
getsockname 108
getsockopt 178
gf_time **404**

heartbeat_cli **583**
heartbeat_serv **585**
host_serv 284, **284**
htonl 68
htons 68

ICMP6_FILTER_*xxx* 660
if_freenameindex 463, **467**
if_indextoname 463, **465**
if_nameindex 463, **466**
if_nametoindex 463, **464**
IN6_IS_ADDR_*xxx* 267
in_cksum **672**
inet6_option_*xxx* 648
inet6_rthdr_*xxx* 651
inet_addr 71
inet_aton 71
inet_ntoa 71
inet_ntop 72
inet_pton 72
ioctl 426, 855
isfdtype 81, **82**

listen 94

mcast_get_if 499
mcast_get_loop 499
mcast_get_ttl 499
mcast_join 499, **501**
mcast_leave 499
mcast_set_if 499
mcast_set_loop 499, **503**
mcast_set_ttl 499
memcmp 70
memcpy 70
memset 70
my_addrs **250, 940**

netdir_getbyaddr 788
netdir_getbyname 786
ntohl 68
ntohs 68

openlog 334

poll 169
pselect 168, **482**
pthread_cond_broadcast 630
pthread_cond_signal 628
pthread_cond_timedwait 630
pthread_cond_wait 628
pthread_create 602

Function and Macro Definitions

(**Bold** page numbers indicate source code implementation)

pthread_detach 604
pthread_exit 604
pthread_getspecific 617
pthread_join 603
pthread_key_create 616
pthread_mutex_lock 626
pthread_mutex_unlock 626
pthread_once 616
pthread_self 604
pthread_setspecific 617
putmsg 854
putpmsg 855

readable_timeo **353**
read_fd **387**
readline 77, **79**, **80**, **619**
readn 77, **78**
readv 357
recv 354
recvfrom 212
recvmsg 358
rtt_init **550**
rtt_minmax **550**
rtt_newpack **551**
rtt_start **551**
rtt_stop **552**
rtt_timeout **552**
rtt_ts **551**

select 150
send 354
sendmsg 358
sendto 212
setnetconfig 785
setnetpath 786
setsockopt 178
shutdown 160
signal **120**
sockatmark 572, **574**
sock_bind_wild 76
sock_cmp_addr 76
sock_cmp_port 76
socket 86
socketpair 376
sockfd_to_family **109**
sock_get_port 76
sock_ntop 75, **76**
sock_ntop_host 76
sock_set_addr 76
sock_set_port 76
sock_set_wild 76
sysctl 455
syslog 333

t_accept 802
t_alloc 789
t_bind 770
t_connect 772
tcp_connect 285, **285**, **793**
tcp_listen 288, **289**, **801**
t_error 768
t_free 789
t_getinfo 869
t_getprotaddr 790
t_getstate 869
t_listen 799
t_look 774
t_open 764
t_optmgmt 840
t_rcv 773
t_rcvconnect 868
t_rcvdis 777
t_rcvrel 776
t_rcvreldata 874
t_rcvudata 819
t_rcvuderr 824
t_rcvv 872
t_rcvvudata 872
t_snd 773
t_snddis 777
t_sndrel 776
t_sndreldata 874
t_sndudata 819
t_sndv 873
t_sndvudata 873
t_strerror 768
t_sync 872
t_unbind 872
tv_sub **667**

udp_client 293, **294**, **821**
udp_connect 295, **296**
udp_server 296, **297**, **827**
uname 249

wait 125
waitpid 125
write_fd **389**
writen 77, **78**
writev 357

xti_accept 803, **804**, **811**
xti_getopt 844, **845**
xti_ntop 792
xti_rdwr 781, **781**
xti_setopt 844, **847**

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 realtime 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 Doupnik, 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 getaddrinfo and getnameinfo Functions 305
11.17 Summary 328

Chapter 12. Daemon Processes and inetd Superserver 331

12.1 Introduction 331
12.2 syslogd Daemon 332
12.3 syslog Function 333
12.4 daemon_init Function 335
12.5 inetd Daemon 339
12.6 daemon_inetd Function 344
12.7 Summary 346

Chapter 13. Advanced I/O Functions 349

13.1 Introduction 349
13.2 Socket Timeouts 349
13.3 recv and send Functions 354
13.4 readv and writev Functions 357
13.5 recvmsg and sendmsg 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 socketpair 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: str_cli Function (Revisited) 399
15.3 Nonblocking connect 409
15.4 Nonblocking connect: Daytime Client 410
15.5 Nonblocking connect: Web Client 413
15.6 Nonblocking accept 422
15.7 Summary 424

Chapter 16. ioctl Operations 425

16.1 Introduction 425
16.2 ioctl Function 426
16.3 Socket Operations 426