Index

Symbols

\$(CFLAGS), make variable, 10

/dev directory, 132

/dev/full, 137

/dev/loop# (loopback devices), 139-142

/dev/null (null device), 136

/dev/pts (PTYs), 142-144

/dev/random (random number device), 137-139

/dev/urandom (random number device), 137-139

/dev/zero, 136
- mapped memory, 109

/etc/services file, 125

/proc file system, 147-148
- CD-ROM drive information, 163
- CPU information, 159
- device information, 159
- file locks information, 164-165
- file size, 147
- file systems information, 161
- hostname and domain name, 160
- IDE device information, 162
- memory usage of kernel, 161
- mounted file system information, 163-164
- output from, 148-150
- partition information, 163
- PCI bus information, 159
- process argument list, 152-154
- process directories, 150-151
- process environment, 154-155
- process executable, 155-156
- process file descriptors, 156-158
- process memory statistics, 158
- process statistics, 158
- SCSI device information, 163
- serial port information, 159-160
- system load information, 165
- system uptime information, 165-166
- version number of kernel, 148, 160

/proc/cpuinfo (system CPU information), 148-150, 159

/proc/devices (device information), 159

/proc/filesystems (file systems information), 161

/proc/ide (IDE device information), 162

/proc/loadavg (system load information), 165

/proc/locks (file locks information), 164-165

/proc/meminfo (memory usage of kernel), 161

/proc/mounts (mounted file system information), 163-164

/proc/pci (PCI bus information), 159

/proc/scsi/scsi (SCSI device information), 163

/proc/self, 151-152

/proc/sys/dev/cdrom/info (CD-ROM drive information), 163

/proc/sys/kernel/domainname (domain names), 160

/proc/sys/kernel/hostname (hostnames), 160

/proc/tty/driver/serial (serial port information), 159-160

/proc/uptime (system uptime information), 165-166

/proc/version (version number of kernel), 148, 160

/tmp directory, race conditions (security hole), 213-216

| (pipe symbol), 110

A

abort function, terminating processes, 55

accept function, 119

access speed, shared memory, 96-97

access system call, 169-170

accessing
- character devices, 134-135
- devices by opening files, 133
- FIFOs, 115-116
- terminals, 135

active processes, viewing, 46-47

addresses
- Internet-domain sockets, 123
- sockets, 117

alarm system call, 185

allocation. *See also* memory allocation; resource allocation
- semaphores (processes), 101
- shared memory, 97-98

app.c (program with library functions), listing 2.8, 37

ar command, 37

archives (static libraries), 37-38
- versus shared libraries, 41-42

argc parameter (main function), 18-19

arglist.c (argc and argv parameters), listing 2.1, 18-19

argument list, 18-19
- command-line options, 19
  - *getopt_long function, 20-23*
- processes, 152-154

arguments, thread
- defined, 62
- passing data, 64-65

argv parameter (main function), 18-19

asm statement (assembly code), 189-190
- GCC conversion of, 191
- maintenance and portability, 196
- optimization, 196
- syntax, 191-192
  - *assembler instructions, 192*
  - *clobbered registers, 194*
  - *input operands, 193*
  - *output operands, 192-193*
- versus C code, performance, 194-196
- when to use, 190

assembler instructions, asm syntax, 192

assembly code, 189-190
- asm syntax, 191-192
  - *assembler instructions, 192*
  - *clobbered registers, 194*
  - *input operands, 193*
  - *output operands, 192-193*
- GCC conversion of asm, 191
- maintenance and portability, 196
- optimization, 196
- versus C code, performance, 194-196
- when to use, 190

assert macro (error checking), 30-31

asynchronously cancelable threads, 70

atomic operations, defined, 79

attachment, shared memory, 98-99

attributes, thread
- customized, 68-69
- defined, 62

audio, playing sound files, 135

authentication, 208-211

B

better_sleep.c (high-precision sleep), listing 8.8, 182

binary semaphores. *See* semaphores (processes)

bind function, 119

bit position, determining (assembly code versus C code), 194-196

bit-pos-asm.c (bit position with bsrl), listing 9.2, 195

bit-pos-loop.c (bit position with loop), listing 9.1, 194-195

block devices
- defined, 130
- list of, 133-134
- loopback devices, 139-142
- warning about, 130

blocking functions, defined, 34

break command, GDB, 12

buffer overruns (security hole), 211-213

buffering output and error streams, 24

buffers. ***See*** **disk buffers**

bugs, finding. ***See*** **error checking**

building sample application programs, 254

C

C code versus assembly code, performance, 194-196

C library functions, relationship with low-level I/O functions, 295-296

-c option (GCC compiler), 7

C++, thread cleanup handlers, 76-77

cache. ***See*** **disk buffers**

calculator program example, profiling programs, 270-280

calculator.c (main calculator program), listing A.3, 274-275

canceling threads, 69-70
- asynchronously cancelable and synchronously cancelable threads, 70
- uncancelable threads, 71-72
- when to use, 72

cancellation points (threads), 70

carriage return character, reading DOS/Windows text files, 287

ccmalloc (dynamic memory allocation), 264-265
- comparison with other dynamic memory allocation tools, 262

CD-ROM drive information, /proc/sys/dev/cdrom/info, 163

cdrom-eject.c (ioctl example), listing 6.2, 144

character devices
- accessing, 134-135
- defined, 130
- list of, 134
- special devices, 136
  - */dev/full, 137*
  - */dev/zero, 136*
  - *null device, 136*
  - *random number devices, 137-139*

char_print function, 64

chdir system call, 296

check-access.c (file access permissions), listing 8.1, 170

child processes, 49
- cleaning up, 59-60
- communciation with parent processes, pipes, 110-112
- zombie processes, 57-59

chmod system call
- changing permission bits, 203
- setuid programs, 208
- sticky bits, 204

clean target (make), 9

cleaning up child processes, 59-60

cleanup handlers, threads, 75-76
- in C++, 76-77

cleanup.c (cleanup handlers), listing 4.8, 75-76

clearing environment variables, 26

client.c (network client program), listing 2.4, 26

clients, defined, 118

clobbered registers, asm syntax, 194

clock-speed.c (cpu clock speed from /proc/cpuinfo), listing 7.1, 149

clone system call, 93-94

close system call, 118, 284

closedir function, 297

closing file descriptors, low-level I/O functions, 284-285

cmdline process entry, 150, 152-154

code. ***See*** **source files**

code listings. ***See*** **listings**

command-line arguments, 18-19
- options, 19
  - *getopt_long function, 20-23*

commands, 53. ***See also*** **functions; system calls**
- ar, 37
- cp, device entries, 131
- dd (block copying), 140

export, 25
free, 161
hostname, 168
id, 198
ipcrm, 100
ipcrm sem, 105
ipcs, 100
ipcs -s, 105
ldd, 39, 41
ls, 299
displaying device entries, 132
viewing permission bits, 201
man, 14, 255
mke2fs, 140
mkfifo, 115
mkstemp, race conditions, 213
ps
displaying terminal devices, 143
viewing active processes, 46-47
renice, scheduling processes, 52
rm, removing device entries, 132
sort, 113
sscanf, 149
strace, 168-169
top, 179
uptime, 166
whoami, 207

common.c (utility functions), listing 11.2, 223-225

compilers
defined, 6-7
GCC, 6-7
linking object files, 8-9
options for source file compilation, 7-8

compiling source files, 9
with debugging information, 11
with make, 9-11

condition variables, synchronizing threads, 86-91

condvar.c (condition variables), listing 4.14, 90-91

configuration, environment variables as configuration information, 26-27

connect function, 118

connection-style sockets, 117

conversation objects (PAM), 210

conversion, hostnames, 123

converting asm statements to assembly code, 191

copy-on-write pages, defined, 178

copy.c (sendfile system call), listing 8.10, 184

copying
from/to file descriptors, 183-185
virtual file systems, 142

cp command, device entries, 131

CPU information, /proc/cpuinfo, 148-150, 159

cpu process entry, 151

create-file.c (create a new files), listing B.1, 284

creating
detached threads, 69
FIFOs, 115
keys (thread-specific data), 73
mutexes, 79
pipes, 110
sockets, 118
threads, 62-63

critical sections, uncancelable threads, 71-72

critical-section.c (critical sections), listing 4.6, 71

customized thread attributes, 68-69

cwd process entry, 150

cxx-exit.cpp (C++ thread cleanup), listing 4.9, 76-77

D

daemons, buffer overruns (security hole), 211-213

data structures, mapped memory, 109

data transfer, sendfile system call, 183-185

datagram-style sockets, 117

date information, gettimeofday system call, 176-177

dd command (block copying), 140

deadlocks (threads), 82-83
on multiple threads, 91

deallocation
  semaphores (processes), 101
  shared memory, 99

debug code. *See* error checking

debuggers, GDB, 11
  compiling with, 11
  running, 11-13

debugging
  semaphores (processes), 105
  shared memory, 100
  system calls, strace command, 168-169
  threads, 77-78

definitions.h (header file for calculator program), listing A.6, 280

deleting
  files, sticky bits, 204
  temporary files, 28

denial-of-service (DoS) attack, 216

dependencies
  libraries, 40-41
  make, 9

destroying sockets, 118

detach state (threads), 68

detached threads
  creating, 69
  defined, 68

detached.c (creating detached threads), listing 4.5, 69

detachment, shared memory, 98-99

development tools
  dynamic memory allocation, 261-262
    ccmalloc, 264-265
    Electric Fence, 265-266
    malloc, 262-263
    mtrace, 263-264
    sample program, 267-269
    selecting, 266-267
  gprof (profiling), 269-270
    calculator program example, 270-280
    collecting information, 271-273
    displaying data, 271-273
  static program analysis, 259-260

device drivers
  defined, 129
  warning about, 130

device entries, 131-132
  /dev directory, 132
  accessing devices, 133
  cp command, 131
  creating, 131-132
  displaying, 132
  removing, 132

device files, types of, 130

device information, /proc/devices, 159

device numbers, defined, 130-131

devices
  accessing by opening files, 133
  block devices, list of, 133-134
  character devices
    accessing, 134-135
    list of, 134
  ioctl system call, 144
  PTYs (pseudo-terminals), 142-144
  special devices, 136
    /dev/full, 137
    /dev/zero, 136
    loopback devices, 139-142
    null device, 136
    random number devices, 137-139

directories
  /dev, 132
  /proc file system process directories, 150-151
  /tmp, race conditions (security hole), 213-216
  permissions, 203
    sticky bits, 204-205
  reading contents of, 296-297, 299

disk buffers, flushing, 173-174

diskfree.c (free disk space information), listing 11.8, 242-243

diskfree.so module (sample application program), 242-244

DISPLAY environment variable, 25

dispositions (signals), 53

dlclose function, 43

dlerror function, 43

dlopen function, 42-43

dlsym function, 43

DNS (Domain Name Service), 123

documentation, 13
  header files, 15
  Info documentation system, 14-15
  man pages, 14
  sample application program, 255-256
  source code, 15

Domain Name Service (DNS), 123

domain names, /proc/sys/kernel/domainname, 160

DoS (denial-of-service) attack, 216

DOS/Windows text files, reading, 287

drivers. *See* device drivers

dup2 system call, 112-113

dup2.c (output redirection), listing 5.8, 113

dynamic linking (libraries), 36

dynamic memory allocation, 261-262
  ccmalloc, 264-265
  Electric Fence, 265-266
  malloc, 262-263
  mtrace, 263-264
  sample program, 267-269
  selecting development tools, 266-267

dynamic runtime loading, shared libraries, 42-43

dynamically linked libraries. *See* shared libraries

E

-e option (ps command), 47

editors
  defined, 4
  Emacs, 4
    automatic formatting, 5
    opening source files, 4
    running GDB in, 13
    syntax highlighting, 5

effective user IDs versus real user IDs, 205-206
  setuid programs, 206-208

EINTR error code, 34

Electric Fence (dynamic memory allocation), 265-266
  comparison with other dynamic memory allocation tools, 262

Emacs, 4
  automatic formatting, 5
  opening source files, 4
  running GDB in, 13
  syntax highlighting, 5

environ global variable, 26

environ process entry, 150, 154-155

environment
  defined, 25-27
  printing, 25
  processes, 154-155

environment variables, 25-27
  accessing, 26
  clearing, 26
  as configuration information, 26-27
  enumerating all, 26
  MALLOC_CHECK, 263
  MALLOC_TRACE, 264
  setting, 26

errno variable, 33

error checking, 30
  assert macro, 30-31
  resource allocation, 35-36
  system call failures, 32
    error codes, 33-35

error codes, system call failures, 33-35

error function, 225

error streams, redirection with pipes, 112-113

error-checking functions, memory allocation, 225

error-checking mutexes, locking, 82

errors, stderr (error stream), 23-24

example program. *See* sample application program

exe process entry, 150, 155-156

exec functions
  avoiding security holes, 217
  creating processes, 48, 50-51

executable files, processes, 155-156

execute permissions, warning about, 204

executing programs with the shell, security holes, 216-218

execve system call, 168

exit codes, 24-25
terminating processes, 55

exit system call, terminating processes, 55-56

exiting threads, 63, 69
cleanup handlers, 75-77

export command, 25

ext2 file system, gaps in large files, 290

F

-f option (ps command), 47

fast mutexes, locking, 82

fcntl system call, 164, 171-172

fd process entry, 150, 156-158

fdatasync system call, 173-174

fdopen function, 295

FIFOs (first-in, first-out files), 114-115
accessing, 115-116
creating, 115
versus Win32 named pipes, 116

file descriptors (low-level I/O), 282
closing low-level I/O functions, 284-285
copying from/to, 183-185
I/O and error streams, 23
moving low-level I/O functions, 288-290
processes, 156-158
reading data from low-level I/O functions, 287-288
using with C library functions, 295-296
writing data to low-level I/O functions, 285-286

file locking, 171-172

file locks information, /proc/locks, 164-165

file permissions, verifying, 169-170

file size, /proc file system, 147

file systems
ext2, gaps in large files, 290
PTYs (pseudo-terminals), 142-144
virtual file systems
copying from devices, 142
creating, 140-142
defined, 139

file systems information, /proc/filesystems, 161

FILE* pointer, 282

FILE* stream, using with low-level I/O functions, 295-296

fileno function, 295

files
deleting sticky bits, 204
opening
accessing devices by, 133
low-level I/O functions, 282-284
owners, 200
permission bits, umasks, 283
permissions, 200-204
warning about execute permissions, 204
temporary files, 27
deleting, 28
mkstemp function, 28-29
tmpfile function, 29

first-in, first-out files. ***See*** **FIFOs**

flags. ***See*** **options**

flock system call, 172

flushing disk buffers, 173-174

fopen function, 295

fork system call, creating processes, 48-51

fork-exec.c (fork and exec functions), listing 3.4, 51

fork.c (fork function), listing 3.3, 49

formatting source files with Emacs, 5

-fPIC option (GCC compiler), 38

fprintf function, 282

free command, 161

free disk space information, sample application program, 242-244

fstat system call, 292

fsync system call, 173-174

functions, 53. ***See also*** **commands; system calls**
abort, terminating processes, 55
accept, 119
bind, 119
blocking functions, defined, 34
char_print, 64

cleanup handlers, 75-76
  in C++, 76-77
closedir, 297
connect, 118
dlclose, 43
dlerror, 43
dlopen, 42-43
dlsym, 43
error, 225
error-checking functions, memory allocation, 225
exec
  avoiding security holes, 217
  creating processes, 48, 50-51
fdopen, 295
fileno, 295
fopen, 295
fprintf, 282
getenv, 26
gethostbyname, 123
getline, buffer overruns, 212
getopt_long, 20-23
getpagesize, 97, 178
gets, buffer overruns, 212
htons, 123
library functions, defined, 167
listen, 119
localtime, 176
low-level I/O. *See* low-level I/O functions
main
  argc and argv parameters, 18-19
  interaction with operating environment, 17
  waiting for threads to exit, 65
mkstemp, 28-29
opendir, 297
pclose, 114
perror, 33
popen, 114
  security holes, 216-218
printf, 282
pthread_attr_setdetachstate, 69
pthread_cancel, 69
pthread_cleanup_pop, 75
pthread_cleanup_push, 75
pthread_cond_broadcast, 89
pthread_cond_init, 89
pthread_cond_signal, 89
pthread_cond_wait, 89
pthread_create, 62
pthread_detach, 69
pthread_equal, 68
pthread_exit, 63, 69
  thread cleanup in C++, 76
pthread_join, 65
pthread_key_create, 73
pthread_mutexattr_destroy, 82
pthread_mutexattr_init, 82
pthread_mutexattr_setkind_np, 82
pthread_mutex_init, 79
pthread_mutex_lock, 80
pthread_mutex_trylock, 83
pthread_mutex_unlock, 80
pthread_self, 68
pthread_setcancelstate, 71
pthread_setcanceltype, 70
pthread_setspecific, 73
pthread_testcancel, 70
reading directory contents, 296-297, 299
recv, 119
sample application program, 223-226
semctl, 101-102
semget, 101
semop, 103
sem_destroy, 84
sem_getvalue, 84
sem_init, 84
sem_post, 84
sem_trywait, 84
sem_wait, 84
send, 118
setenv, 26
seteuid, 206
shmat, 98-99
shmctl, 99
shmdt, 99
shmget, 97-98
signal handlers, 53-54
sleep, 181
socket, 118
socketpair, 125-126
for sockets, list of, 117
strerror, 33
strftime, 176-177
system
  creating processes, 48
  security holes, 216-218
thread functions, defined, 62
tmpfile, 29
unsetenv, 26
wait, terminating processes, 56-57

G

-g option (GCC compiler), 11

g++ (C++ compiler), 7

GCC (C compiler), 6-7
- assembly code, 189-190
  - *asm syntax, 191-194*
  - *conversion of asm, 191*
  - *maintenance and portability, 196*
  - *optimization, 196*
  - *versus C code performance, 194-196*
  - *when to use, 190*
- linking object files, 8-9
- options for source file compilation, 7-8
- -pedantic option, 260
- -Wall option, 260

GDB (GNU Debugger), 11
- commands
  - *break, 12*
  - *next, 13*
  - *print, 12*
  - *run, 12*
  - *step, 13*
  - *up, 12*
  - *where, 12*
- compiling with, 11
- running, 11-13

get-exe-path.c (program executable path), listing 7.5, 155-156

get-pid.c (process ID from /proc/self), listing 7.2, 151-152

getcwd system call, 296

getegid system call, 200

getenv function, 26

geteuid function, 200

gethostbyname function, 123

getline function, buffer overruns, 212

getopt_long function, 20-23

getopt_long.c (getopt_long function), listing 2.2, 21-23

getpagesize function, 97, 178

getrlimit system call, 174-175

getrusage system call, 175-176

gets function, buffer overruns, 212

gettimeofday system call, 176-177
- sample application program, 239

GID (group ID), 198

GNU Coding Standards, 19

GNU Debugger. *See* GDB

GNU General Public License, 309-316

GNU Make. *See* make

GNU/Linux distribution information, sample application program, 240, 242

GNU/Linux online resources, list of, 303-304

gprof (profiling) development tool, 269-270
- calculator program example, 270-280
- collecting information, 271, 273
- displaying data, 271-273

grep-dictionary.c (word search), listing 10.6, 216-217

group ID (GID), 198

groups
- process group IDs, 199-200
- UID (user ID) and GID (group ID), 198

H

hard limit, defined, 174

hardware devices
- block devices, list of, 133-134
- character devices
  - *accessing, 134-135*
  - *list of, 134*

header files, 15

hello.c (Hello World), listing A.1, 260

hexdump.c (print a hexadecimal file dump), listing B.4, 287-288

highlighting source files with Emacs, 5

HOME environment variable, 25

hostname command, 168

hostnames
- /proc/sys/kernel/hostname, 160
- conversion, 123

htons function, 123

HTTP (Hypertext Transport Protocol), 125, 221

I

-I option (GCC compiler), 7

I/O (input/output)
- FIFO access, 115-116
- input/output and error streams, 23-24
- mmap function, 109
- redirection with pipes, 112-113

I/O functions, low-level. *See* low-level I/O functions

id command, 198

IDE (Integrated Development Environment), 9

IDE device information, /proc/ide, 162

idle time information, /proc/uptime, 165-166

Info documentation system, 14-15, 256

init process, 59

initialization, semaphores (processes), 102

inline assembly code. *See* assembly code

input operands, asm syntax, 193

input. *See* I/O (input/output)

Integrated Development Environment (IDE), 9

Intel x86 architectures, register letters, 193

Internet Protocol (IP), 123

Internet-domain sockets, 123-125

interprocess communication (IPC)
- defined, 95
- mapped memory, 105
  - *example programs, 106-108*
  - *mmap function, 105-109*
  - *private mappings, 109*
  - *shared file access, 108-109*
- pipes, 110
  - *creating, 110*
  - *FIFOs, 114-116*
  - *parent-child process communication, 110-112*
  - *popen and pclose functions, 114*
  - *redirection, 112-113*
- semaphores, 101
  - *allocation and deallocation, 101*
  - *debugging, 105*
  - *initialization, 102*
  - *wait and post operations, 103-104*
- shared memory, 96
  - *access speed, 96-97*
  - *advantages and disadvantages, 101*
  - *allocation, 97-98*
  - *attachment and detachment, 98-99*
  - *deallocation, 99*
  - *debugging, 100*
  - *example program, 99-100*
  - *memory model, 97*
- sockets, 116
  - *connect function, 118*
  - *creating, 118*
  - *destroying, 118*
  - *functions, list of, 117*
  - *Internet-domain sockets, 123-125*
  - *local sockets, 119-123*
  - *send function, 118*
  - *servers, 118-119*
  - *socket pairs, 125-126*
  - *terminology, 117*

interval timers, setting, 185-186

ioctl system call, 144

IP (Internet Protocol), 123

IPC. *See* interprocess communication

ipcrm command, 100

ipcrm sem command, 105

ipcs -s command, 105

ipcs command, 100

issue.c (GNU/Linux distrubution information), listing 11.7, 240-242

issue.so module (sample application program), 240-242

itemer.c (interval timers), listing 8.11, 185-186

J-K

-j option (ps command), 47

job control notification, in shell, 93

job-queue1.c (thread race conditions), listing 4.10, 78

job-queue2.c (mutexes), listing 4.11, 80-81

job-queue3.c (semaphores), listing 4.12, 84-86

joinable threads, defined, 68

joining threads, 65-66

kernel, /proc file system. *See* /proc file system

keys (thread-specific data), creating, 73

kill system call, 47, 55

killing processes, 47

L

-L option (GCC compiler), 9

-l option (ps command), 47

LD_LIBRARY_PATH environment variable, 40

ldd command, 39-41

libraries, linking to, 8, 36-37
- archives (static libraries), 37-38
- dynamic runtime loading, 42-43
- library dependencies, 40-41
- shared libraries, 38-40
  - *versus archives, 41-42*
- standard libraries, 40

library functions, defined, 167

limit-cpu.c (resource limits), listing 8.4, 175

linking
- to libraries, 8, 36-37
  - *archives (static libraries), 37-38*
  - *dynamic runtime loading, 42-43*
  - *library dependencies, 40-41*
  - *shared libraries, 38-40*
  - *shared libraries versus archives, 41-42*
  - *standard libraries, 40*
- object files, 8-9

links, symbolic
- reading, 182-183
- stat function, 292

listdir.c (printing directory listings), listing B.8, 297-299

listen function, 119

listings
- app.c (program with library functions), 37
- arglist.c (argc and argv parameters), 18-19
- better_sleep.c (high-precision sleep), 182
- bit-pos-loop.c (bit position with loop), 194-195
- bit-pos-asm.c (bit position with bsrl), 195
- calculator.c (main calculator program), 274-275
- cdrom-eject.c (ioctl example), 144
- check-access.c (file access permissions), 170
- cleanup.c (cleanup handlers), 75-76
- client.c (network client program), 26
- clock-speed.c (cpu clock speed from /proc/cpuinfo), 149
- common.c (utility functions), 223-225
- condvar.c (condition variables), 90-91
- copy.c (sendfile system call), 184
- create-file.c (create a new file), 284
- critical-section.c (critical sections), 71
- cxx-exit.cpp (C++ thread cleanup), 76-77
- definitions.h (header file for calculator program), 280
- detached.c (creating detached threads), 69
- diskfree.c (free disk space information), 242-243
- dup2.c (output redirection), 113
- fork.c (fork function), 49
- fork-exec.c (fork and exec functions), 51
- get-exe-path.c (program executable path), 155-156
- getopt_long.c (getopt_long function), 21-23
- get-pid.c (process ID from /proc/self), 151-152
- grep-dictionary.c (word search), 216-217
- hello.c (Hello World), 260
- hexdump.c (print a hexadecimal file dump), 287-288
- issue.c (GNU/Linux distribution information), 240, 242
- itemer.c (interal timers), 185-186
- job-queue1.c (thread race conditions), 78
- job-queue2.c (mutexes), 80-81

job-queue3.c (semaphores), 84-86
limit-cpu.c (resource limits), 175
listdir.c (printing directory listings), 297-299
lock-file.c (write locks), 171-172
lseek-huge.c (creating large files), 289-290
main.c (C source file), 6
main.c (main server program), 235-238
Makefile (Makefile for sample application program), 252-253
malloc-use.c (dynamic memory allocation), 267-269
mmap-read.c (mapped memory), 107
mmap-write.c (mapped memory), 106
module.c (loading server modules), 226-227
mprotect.c (memory access), 180-181
number.c (unary number implementation), 276-278
open-and-spin.c (opening files), 157
pam.c (PAM example), 209
pipe.c (parent-child process communication), 111
popen.c (popen command), 114
primes.c (prime number computation in a thread), 67
print-arg-list.c (printing process argument lists), 153
print-cpu-times.c (process statistics), 176
print_env.c (printing execution environment), 26
print-environment.c (process environment), 154-155
print-pid.c (printing process IDs), 46
print-symlink.c (symbolic links), 183
print-time.c (date/time printing), 177
print-uname (version number and hardware information), 188
print-uptime.c (system uptime and idle time), 165-166
processes.c (summarizing running processes), 244-250
random_number.c (random number generation), 138-139
readfile.c (resource allocation during error checking), 35-36
read-file.c (reading files into buffers), 292-293
reciprocal.cpp (C++ source file), 6
reciprocal.hpp (header file), 7
sem_all_deall.c (semaphore allocation and deallocation), 102
sem_init.c (semaphore initialization), 102
sem_pv.c (semaphore wait and post operations), 104
server.c (server implementation), 228-233
server.h (function and variable declarations), 222-223
setuid-test.c (setuid programs), 207
shm.c (shared memory), 99-100
sigchld.c (cleaning up child processes), 60
sigusr1.c (signal handlers), 54
simpleid.c (printing user and group IDs), 200
socket-client.c (local sockets), 121
socket-inet.c (Internet-domain sockets), 124
socket-server.c (local sockets), 120
spin-condvar.c (condition variables), 87
stack.c (unary number stack), 279-280
stat-perm.c (viewing file permissions with stat system call), 202
sysinfo.c (system statistics), 187
system.c (system function), 48
temp_file.c (mkstemp function), 28-29
temp-file.c (temporary file creation), 214-215
test.c (library contents), 37
thread-create.c (creating threads), 63
thread-create2 (creating two threads), 64-65
thread-create2.c (revised main function), 65
thread-pid (printing thread process IDs), 92
tifftest.c (libtiff library), 40
time.c (show wall-clock time), 239-240
timestamp.c (append a timestamp), 285
tsd.c (thread-specific data), 73-74
write-all.c (write all buffered data), 286
write-args.c (writev function), 294-295
write_journal_entry.c (data buffer flushing), 173
zombie.c (zombie processes), 58

loading server modules (sample application program), 226-227

local sockets, 119
- example program, 120-123

localtime function, 176

lock-file.c (write locks), listing 8.2, 171-172

locking
- physical memory, 177-179
- threads
  - *nonblocking mutex tests, 83*
  - *with mutexes, 79-83*

locks, fcntl system call, 171-172

locks information, /proc/locks, 164-165

long form (command-line options), 19

loopback devices, 139-142

low-level I/O functions, 281-282
- chdir, 296
- closing file descriptors, 284-285
- file descriptors, 282
- getcwd, 296
- mkdir, 296
- moving file descriptors, 288-290
- opening files, 282-284
- reading data from file descriptors, 287-288
- relationship with C library functions, 295-296
- rename, 296
- rmdir, 296
- stat (file status information), 291-293
- unlink, 296
- vector reads, 295
- vector writes, 293-295
- writing data to file descriptors, 285-286

ls command, 299
- displaying device entries, 132
- viewing permission bits, 201

lseek system call, 288-290

lseek-huge.c (creating large files), listing B.5, 289-290

lstat system call, 292
- race conditions, 214

M

macros
- assert (error checking), 30-31
- on GCC command line, 8
- NDEBUG, 30

main function
- argc and argv parameters, 18-19
- interaction with operating environment, 17
- waiting for threads to exit, 65

main server program (sample application program), 235-239

main.c (C source file), listing 1.1, 6

main.c (main server program), listing 11.5, 235-238

maintenance, assembly code, 196

major device numbers, defined, 130-131

make, compiling source files, 9-11

Makefile, 10-11
- sample application program, listing 11.10, 252-253

malloc (dynamic memory allocation), 262-263
- comparison with other dynamic memory allocation tools, 262

malloc-use.c (dynamic memory allocation), listing A.2, 267-269

MALLOC_CHECK environment variable, 263

MALLOC_TRACE environment variable, 264

man command, 14, 255

man pages, 14
- writing, 255

mapped memory, 105
- example programs, 106-108
- mmap function, 105-106, 109
- private mappings, 109
- shared file access, 108-109

maps process entry, 150

memory
  dynamic allocation, 261-262
    ccmalloc, 264-265
    Electric Fence, 265-266
    malloc, 262-263
    mtrace, 263-264
    sample program, 267-269
    selecting development tools, 266-267
  mapped memory, 105
    example programs, 106-108
    mmap function, 105-106, 109
    private mappings, 109
    shared file access, 108-109
  page-aligned memory, allocating, 179
  pages, 178
  physical memory, locking, 177-179
  shared memory, 96
    access speed, 96-97
    advantages and disadvantages, 101
    allocation, 97-98
    attachment and detachment, 98-99
    deallocation, 99
    debugging, 100
    example program, 99-100
    memory model, 97
  thrashing, defined, 178

memory allocation
  error-checking functions, 225
  page-aligned memory, 179

memory buffers. *See* disk buffers

memory model, shared memory, 97

memory permissions, setting, 179-181

memory statistics, processes, 158

memory usage of kernel, /proc/meminfo, 161

minor device numbers, defined, 130-131

mkdir system call, 296

mke2fs command, 140

mkfifo command, 115

mknod system call, creating device entries, 131-132

mkstemp function, 28-29
  race conditions, 213

mlock system calls, 177-179

mlockall system call, 178

mmap system call, 105-106, 109, 179

mmap-read.c (mapped memory), listing 5.6, 107

mmap-write.c (mapped memory), listing 5.5, 106

mode. *See* permission bits

module.c (loading server modules), listing 11.3, 226-227

modules, sample application program, 239
  diskfree.so, 242-244
  issue.so, 240, 242
  loading server modules, 226-227
  processes.so, 244-252
  time.so, 239-240

mount system call, 141, 147

mount descriptors, 163-164

mounted file system information, /proc/mounts, 163-164

moving file descriptors, low-level I/O functions, 288-290

mprotect system call, 179-181

mprotect.c (memory access), listing 8.7, 180-181

msync system call, 108

mtrace (dynamic memory allocation), 263-264
  comparison with other dynamic memory allocation tools, 262

multiple threads, deadlocks on, 91

munlock system call, 178

munlockall system call, 178

munmap system call, 106

mutexes
  with condition variables, 88
  locking threads, 79-82
    deadlocks, 82-83
  nonblocking tests, 83

mutual exclusion locks. *See* mutexes

N

named pipes. *See* FIFOs

nanosleep system call, 181-182

NDEBUG macro, 8, 30

network byte order (sockets), 123

Network File System (NFS), 172

newline character, reading DOS/Windows text files, 287

next command, GDB, 13

NFS (Network File System), 172

nice system call, scheduling processes, 52

niceness values, processes, 52

nonblocking mode (wait functions), 59

nonblocking mutex tests (threads), 83

NUL versus NULL, 152

null device, 136

number.c (unary number implementation), listing A.4, 276-278

O

-o option
  GCC compiler, 8
  ps command, 47

-O2 option (GCC compiler), 8

object files, linking, 8-9

online resources, list of, 303-304

Open Publication License Version 1.0, 305-308

open system call, 282-284

open-and-spin.c (opening files), listing 7.6, 157

opendir function, 297

opening
  files
    accessing devices by, 133
    low-level I/O functions, 282-284
  source files with Emacs, 4

optimization. *See also* performance
  assembly code, 196
  GCC compiler options, 8
  gprof (profiling) development tool, 269-270
    calculator program example, 270-271, 274-280
    collecting information, 271, 273
    displaying data, 271-273

output from /proc file system, 148-150. *See also* I/O (input/output)

output operands, asm syntax, 192-193

owners of files, 200

P

packets, 117

page-aligned memory, allocating, 179

pages, copy-on-write, 178

pages of memory, 178
  shared memory, 97

PAM (Pluggable Authentication Modules), 209-211

pam.c (PAM example), listing 10.4, 209

parent process ID (ppid), 46

parent processes, 49
  communication with child processes, 110-112

partition (partition device information), 163

passing data to threads, 64-65

passwords, user authentication, 208-209

PATH environment variable, 25

PCI bus information, /proc/pci, 159

pclose function, 114

-pedantic option (GCC compiler), 260

performance, assembly code versus C code, 194-196. *See also* optimization

permission bits
  changing with chmod function, 203
  umasks, 283
  viewing, 201

permissions
  directories, 203
    sticky bits, 204-205
  file permissions, 200-204
    verifying, 169-170
    warning about execute permissions, 204
  memory permissions, setting, 179-181

perror function, 33

physical memory, locking, 177-179

PIC (position-independent code), 38

pid (process ID), 46

pipe system call, 110

pipe symbol (|), 110

pipe.c (parent-child process communication), listing 5.7, 111

pipes, 110
- creating, 110
- FIFOs, 114-115
  - *accessing, 115-116*
  - *creating, 115*
  - *versus Win32 named pipes, 116*
- parent-child process communcation, 110-112
- popen and pclose functions, 114
- redirection, 112-113

Pluggable Authentication Modules (PAM), 209-211

popen command, 114
- security holes, 216-218

popen.c (popen command), listing 5.9, 114

port numbers
- sockets, 123
- standard, 125

portability, assembly code, 196

position-independent code (PIC), 38

post operation (semaphores), 83, 103-104

postfix notation, defined, 270

ppid (parent process ID), 46

primes.c (prime number computation in a thread), listing 4.4, 67

print command, GDB, 12

print-arg-list.c (printing process argument lists), listing 7.3, 153

print-cpu-times.c (process statistics), listing 8.5, 176

print-environment.c (process environment), listing 7.4, 154-155

print_env.c (printing execution environment), listing 2.3, 26

print-pid.c (printing process IDs), listing 3.1, 46

print-symlink.c (symbolic links), listing 8.9, 183

print-time.c (date/time printing), listing 8.6, 177

print-uname (version number and hardware information), listing 8.13, 188

print-uptime.c (system uptime and idle time), listing 7.7, 165-166

printenv program, 25

printf function, 282

printing the environment, 25

private mappings, mapped memory, 109

process group IDs, 199-200

process IDs, 46

process semaphores. ***See*** **semaphores (processes)**

process statistics, 175-176

process user IDs, 199-200

processes. ***See also*** **interprocess communication (IPC)**
- /proc file system directories, 150-151
- /proc/self, 151-152
- argument list, 152-154
- child, 49
- creating
  - *with fork and exec functions, 48-51*
  - *with system function, 48*
- defined, 45
- environment, 154-155
- executable files, 155-156
- file descriptors, 156-158
- implementing threads as, 92-93
  - *clone system call, 93-94*
  - *signal handling, 93*
- init process, 59
- memory statistics, 158
- parent, 49
- process IDs, 46
- relationship with threads, 61-62
- scheduling, 52
- signals, 52-54
- statistics, 158

terminating, 47, 55-56
cleaning up child processes, 59-60
wait functions, 56-57
zombie processes, 57-59
versus threads, when to use, 94
viewing active, 46-47

processes.c (summarizing running processes), listing 11.9, 244-250

processes.so module (sample application program), 244-252

profiling programs, gprof development tool, 269-270
calculator program example, 270-271, 274-280
collecting information, 271, 273
displaying data, 271-273

program listings. *See* listings

programs
argument list, 18-19
command-line options, 19
getopt_long function, 20-23
development tools. *See* development tools
environment, 25-27
error checking, 30
assert macro, 30-31
resource allocation, 35-36
system call failures, 32-35
exit codes, 24-25
interaction with operating environment, 17
linking to libraries, 36-37
archives (static libraries), 37-38
dynamic runtime loading, 42-43
library dependencies, 40-41
shared libraries, 38-40
shared libraries versus archives, 41-42
standard libraries, 40
sample application program. *See* sample application program
standard I/O, 23-24
temporary files, 27
mkstemp function, 28-29
tmpfile function, 29

protocols
associations with standard port numbers, 125
HTTP (Hypertext Transport Protocol), 125
IP (Internet Protocol), 123
sockets, 117
TCP (Transmission Control Protocol), 123

ps command
displaying terminal devices, 143
viewing active processes, 46-47

pseudo-terminals (PTYs), 142-144

pseudorandom numbers, 137

pthread functions, 62

pthread_attr_setdetachstate function, 69

pthread_cancel function, 69

pthread_cleanup_pop function, 75

pthread_cleanup_push function, 75

pthread_cond_broadcast function, 89

pthread_cond_init function, 89

pthread_cond_signal function, 89

pthread_cond_wait function, 89

pthread_create function, 62

pthread_detach function, 69

pthread_equal function, 68

pthread_exit function, 63, 69
thread cleanup in C++, 76

pthread_join function, 65

pthread_key_create function, 73

pthread_mutexattr_destroy function, 82

pthread_mutexattr_init function, 82

pthread_mutexattr_setkind_np function, 82

pthread_mutex_init function, 79

pthread_mutex_lock function, 80

pthread_mutex_trylock function, 83

pthread_mutex_unlock function, 80

pthread_self function, 68

pthread_setcancelstate function, 71

pthread_setcanceltype function, 70

pthread_setspecific function, 73

pthread_testcancel function, 70

PTYs (pseudo-terminals), 142-144

Q-R

race conditions (security hole), 213-216

race conditions (threads), 78-79
  avoiding with mutexes, 79-82
    deadlocks, 82-83

random number devices, 137-139

random_number.c (random number generation), listing 6.1, 138-139

read system call, 287-288

read-file.c (reading files into buffers), listing B.6, 292-293

readdir system call, 297

readfile.c (resource allocation during error checking), listing 2.6, 35-36

reading
  data from file descriptors, low-level I/O functions, 287-288
  directory contents, 296-297, 299
  DOS/Windows text files, 287
  symbolic links, 182-183

readlink system call, 182-183

readv system call, 295

real user IDs, versus effective user IDs, 205-206
  setuid programs, 206-208

reciprocal.cpp (C++ source file), listing 1.2, 6

reciprocal.hpp (header file), listing 1.3, 7

recursive mutexes, locking, 82

recv function, 119

redirecting I/O and error streams, 23

redirection with pipes, 112-113

register letters, Intel *x*86 architectures, 193

registering cleanup handlers, 75

removing device entries, 132

rename system call, 296

renice command, scheduling processes, 52

resource allocation, error checking, 35-36

resource limits, setting, 174-175

return values (threads), 66-67

rm command, removing device entries, 132

rmdir system call, 296

root process entry, 150

root user account, 199
  permissions, 204
  setuid programs, 206-208

rules (make), 9

run command, GDB, 12

runnable tasks, defined, 165

running processes, summarizing (sample application program), 244-252

running the server (sample application program), 254-255

runtime checks, assert macro, 30-31

runtime loading, shared libraries, 42-43

runtime tools. *See* development tools

S

sample application program, 219
  building, 254
  common functions, 223-224, 226
  documentation, 255-256
  implementation, 221, 223
  loading server modules, 226-227
  main server program, 235-239
  Makefile, 252-253
  modules, 239
    diskfree.so, 242-244
    issue.so, 240, 242
    processes.so, 244-252
    time.so, 239-240
  overview, 219-221
  running the server, 254-255
  server implementation, 228-235

scheduling processes, 52

SCSI device information, /proc/scsi/scsi, 163

security
  authentication, 208-209, 211
  directory permissions, 203
    sticky bits, 204-205
  file permissions, 200-204
    warning about execute permissions, 204

GID (group ID), 198
holes in, 211
buffer overruns, 211-213
executing programs with the shell, 216-218
race conditions, 213-216
permission bits, umasks, 283
process group IDs, 199-200
process user IDs, 199-200
root user account, 199
permissions, 204
user IDs (UID), 198
real versus effective IDs, 205-208

segments (shared memory), 97
advantages and disadvantages, 101
allocation, 97-98
attachment and detachment, 98-99
deallocation, 99
debugging, 100
example program, 99-100

selecting dynamic memory allocation tools, 266-267

semaphores (processes), 101
allocation and deallocation, 101
debugging, 105
initialization, 102
versus condition variables, 91
wait and post operations, 103-104

semaphores (threads), 83-86

semctl function, 101-102

semget function, 101

semop function, 103

sem_all_deall.c (semaphore allocation and deallocation), listing 5.2, 102

sem_destroy function, 84

sem_getvalue function, 84

sem_init function, 84

sem_init.c (semaphore initialization), listing 5.3, 102

sem_post function, 84

sem_pv.c (semaphore wait and post operations), listing 5.4, 104

sem_trywait function, 84

sem_wait function, 84

send function, 118

sendfile system call, 183-185

serial port information, /proc/tty/driver/serial, 159-160

server implementation (sample application program), 228-235

server modules, loading (sample application program), 226-227

server.c (server implementation), listing 11.4, 228-233

server.h (function and variable declarations), listing 11.1, 222-223

servers
defined, 118
running (sample application program), 254-255
sockets, 118-119

setenv function, 26

seteuid function, 206

setitimer system call, 185-186

setreuid system call, 206

setrlimit system call, 174-175

setuid programs, 206-208

setuid-test.c (setuid programs), listing 10.3, 207

shared file access, memory mapping, 108-109

shared libraries, 38-40
versus archives, 41-42

shared memory, 96
access speed, 96-97
advantages and disadvantages, 101
allocation, 97-98
attachment and detachment, 98-99
deallocation, 99
debugging, 100
example program, 99-100
memory model, 97

shared objects. *See* shared libraries

shell
executing programs within (security holes), 216-218
job control notification, 93

shm.c (shared memory), listing 5.1, 99-100

shmat function, 98-99

shmctl function, 99

shmdt function, 99

shmget function, 97-98

short form (command-line options), 19

SIGABRT signal, 302

sigaction system call (signal dispositions), 53

SIGALRM signal, 302

SIGCHLD signal, 302

sigchld.c (cleaning up child processes), listing 3.7, 60

SIGFPE signal, 302

SIGHUP signal, 301

SIGILL signal, 302

SIGINT signal, 302

SIGKILL signal, 302

signal handling (threads), 93

signal-handler functions, 53-54

signals, 52-54
- cleaning up child processes, 59-60
- table of, 301-302
- terminating processes, 55

SIGPIPE signal, 302

SIGSEGV signal, 302

SIGTERM signal, 302

SIGUSR1 signal, 302

sigusr1.c (signal handlers), listing 3.5, 54

SIGUSR2 signal, 302

SIGVTALRM signal, 302

SIGXCPU signal, 302

simpleid.c (printing user and group IDs), listing 10.1, 200

sleep function, 181-182

socket addresses, 117

socket function, 118

socket-client.c (local sockets), listing 5.11, 121

socket-inet.c (Internet-domain sockets), listing 5.12, 124

socket-server.c (local sockets), listing 5.10, 120

socketpair function, 125-126

sockets, 116
- connect function, 118
- creating, 118
- destroying, 118
- functions, list of, 117
- Internet-domain sockets, 123-125
- local sockets, 119
  - *example program, 120-123*
- send function, 118
- servers, 118-119
- socket pairs, 125-126
- terminology, 117

soft limit, defined, 174

sort command, 113

sound files, playing, 135

source code. *See* source files

source code listings. *See* listings

source files
- compiling
  - *GCC options, 7-8*
  - *linking object files, 8-9*
  - *with debugging information, 11*
  - *with make, 9-11*
- debugging, 11
  - *running GDB, 11-13*
- formatting with Emacs, 5
- opening with Emacs, 4
- sample application program, 221, 223
- syntax highlighting with Emacs, 5
- as technical support, 15

special devices, 136
- /dev/full, 137
- /dev/zero, 136
- loopback devices, 139-142
- null device, 136
- random number devices, 137-139

speed of access, shared memory, 96-97

spin-condvar.c (condition variables), listing 4.13, 87

sscanf command, 149

stack.c (unary number stack), listing A.5, 279-280

standard libraries, linking to, 40

standard port numbers, 125

stat process entry, 151

stat system call, 291-293
- viewing permission bits, 201-202

stat-perm.c (viewing file permissions with stat system call), listing 10.2, 202

static libraries. *See* archives

static linking (libraries), 36

static program analysis tools, 259-260

statistics
- memory statistics, processes, 158
- processes, 158, 175-176
- system statistics, retrieving, 186-187

statm process entry, 151, 158

status process entry, 151, 158

stderr (error stream), 23-24

stdin (input stream), 23-24

stdout (output stream), 23-24

step command, GDB, 13

sticky bits (security), 204-205

strace command, 168-169

streams, redirection with pipes, 112-113

strerror function, 33

strftime function, 176-177

structures. *See* data structures

su program, 207-208

superuser. *See* root user account

symbolic links
- race conditions (security hole), 213-216
- reading, 182-183
- stat function, 292

synchronizing threads
- condition variables, 86-91
- deadlocks, 82-83
  - *on multiple threads, 91*
- mutexes, 79-82
- nonblocking mutex tests, 83
- race conditions, 78-79
- with semaphores, 83-86

synchronously cancelable threads, 70

syntax highlighting with Emacs, 5

sysinfo system call, 166, 186-187

sysinfo.c (system statistics), listing 8.12, 187

system call failures, 32
- error codes, 33-35

system calls. *See also* commands; functions
- access, 169-170
- alarm, 185
- chdir, 296
- chmod, changing permission bits, 203
- close, 118, 284
- debugging, strace command, 168-169
- defined, 167-168
- dup2, 112-113
- execve, 168
- exit, terminating processes, 55-56
- fcntl, 164, 171-172
- fdatasync, 173-174
- flock, 172
- fork, creating processes, 48-51
- fstat, 292
- fsync, 173-174
- getcwd, 296
- getegid, 200
- geteuid, 200
- getrlimit, 174-175
- getrusage, 175-176
- gettimeofday, 176-177, 239
- ioctl, 144
- kill, 47, 55
- list of, 168
- lseek, 288-290
- lstat, 292
  - *race conditions, 214*
- mkdir, 296
- mknod, creating device entries, 131-132
- mlock, 177-179
- mlockall, 178
- mmap, 105-106, 109, 179
- mount, 141, 147
- mprotect, 179-181
- msync, 108
- munlock, 178
- munlockall, 178
- munmap, 106
- nanosleep, 181-182
- nice, scheduling processes, 52
- open, 282-284
- pipe, 110
- read, 287-288

readdir, 297
readlink, 182-183
readv, 295
rename, 296
rmdir, 296
sendfile, 183-185
setitimer, 185-186
setreuid, 206
setrlimit, 174-175
sigaction (signal dispositions), 53
stat, 291-293
sysinfo, 166, 186-187
time, 195
ulimit, 174
uname, 169, 187
unlink, 28, 119, 296
write, 169, 285-286
writev, 293-295

system function
creating processes, 48
security holes, 216-218

system information, uname system call, 187

system load information, /proc/loadavg, 165

system statistics, retrieving, 186-187

system uptime information, /proc/uptime, 165-166

System V semaphores. *See* semaphores (processes)

system.c (system function), listing 3.2, 48

T

targets (make), 9

TCP (Transmission Control Protocol), 123

technical support, 13
header files, 15
Info documentation system, 14-15
man pages, 14
source code, 15

temp-file.c (temporary file creation), listing 10.5, 214-215

temporary files, 27
deleting, 28
mkstemp function, 28-29
tmpfile function, 29

temp_file.c (mkstemp function), listing 2.5, 28-29

terminals
accessing, 135
PTYs (pseudo-terminals), 142-144

terminating processes, 55-56
cleaning up child processes, 59-60
wait functions, 56-57
zombie processes, 57-59

test.c (library contents), listing 2.7, 37

thrashing, defined, 178

thread arguments
defined, 62
passing data, 64-65

thread attributes
customized, 68-69
defined, 62

thread functions, defined, 62

thread IDs, 62
uses for, 68

thread-create.c (creating threads), listing 4.1, 63

thread-create2 (creating two threads), listing 4.2, 64-67, 69, 72

thread-create2.c (revised main function), listing 4.3, 65

thread-pid (printing thread process IDs), listing 4.15, 92

thread-specific data, 72-74

threads
atomic operations, defined, 79
canceling, 69-70
asynchronously cancelable and synchronously cancelable threads, 70
uncancelable threads, 71-72
when to use, 72
cleanup handlers, 75-76
in C++, 76-77
creating, 62-63
debugging, 77-78
defined, 61

detach state, defined, 68
detached threads
creating, 69
defined, 68
exiting, 63, 69
implementing as processes, 92-93
clone system call, 93-94
signal handling, 93
joinable threads, defined, 68
joining, 65-66
passing data to, 64-65
pthread functions, 62
relationship with processes, 61-62
return values, 66-67
synchronizing
condition variables, 86-91
deadlocks, 82-83
deadlocks on multiple threads, 91
mutexes, 79-82
nonblocking mutex tests, 83
race conditions, 78-79
semaphores, 83-86
thread IDs, uses for, 68
thread-specific data, 72-74
versus processes, when to use, 94

tifftest.c (libtiff library), listing 2.9, 40

time system call, 195

time information, gettimeofday system call, 176-177

time.c (show wall-clock time), listing 11.6, 239-240

time.so module (sample application program), 239-240

timers, setting interval timers, 185-186

timestamp.c (append a timestamp), listing B.2, 285

tmpfile function, 29

tools. *See* development tools

top command, 179

transferring data, sendfile system call, 183-185

Transmission Control Protocol (TCP), 123

troff, formatting man pages, 255

troubleshooting. *See* error checking

tsd.c (thread-specific data), listing 4.7, 73-74

U

UID (user ID), 198

ulimit system call, 174

umasks, permission bits, 283

uname system call, 169, 187

unary numbers, defined, 270

uncancelable threads, 71-72
defined, 70

UNIX epoch, defined, 176

UNIX-domain sockets. *See* local sockets

unlink system call, 28, 119, 296

unsetenv function, 26

up command, GDB, 12

uptime command, 166

uptime information, /proc/uptime, 165-166

user authentication, 208-209, 211

USER environment variable, 25

user IDs (UID), 198
real versus effective IDs, 205-206
setuid programs, 206-208

usernames, UID (user ID), 198

users
process user IDs, 199-200
root, 199
UID (user ID) and GID (group ID), 198

V

variables
condition variables, synchronizing threads, 86-91
environment variables, 25-27
accessing, 26
clearing, 26
as configuration information, 26-27
enumerating all, 26
setting, 26
errno, 33
thread-specific data, 72-74

vector reads, low-level I/O functions, 295

vector writes, low-level I/O functions, 293-295

version number of kernel, /proc/version, 148, 160

virtual file systems
- copying from devices, 142
- creating, 140-142
- defined, 139

W-Z

wait functions, terminating processes, 56-57

wait operation (semaphores), 83, 103-104

-Wall option (GCC compiler), 260

Web sites, list of online resources, 303-304

where command, GDB, 12

whoami command, 207

Win32 named pipes, versus FIFOs, 116

Windows text files, reading, 287

write system call, 169, 285-286

write-all.c (write all buffered data), listing B.3, 286

write-args.c (writev function), listing B.7, 294-295

writev system call, 293-295

write_journal_entry.c (data buffer flushing), listing 8.3, 173

writing
- data to file descriptors, low-level I/O functions, 285-286
- man pages, 255

zombie processes, 57-59

zombie.c (zombie processes), listing 3.6, 58

VOICES THAT MATTER

HOW TO CONTACT US

VISIT OUR WEB SITE

WWW.NEWRIDERS.COM

On our Web site, you'll find information about our other books, authors, tables of contents, and book errata. You will also find information about book registration and how to purchase our books, both domestically and internationally.

EMAIL US

Contact us at: **nrfeedback@newriders.com**

- If you have comments or questions about this book
- To report errors that you have found in this book
- If you have a book proposal to submit or are interested in writing for New Riders
- If you are an expert in a computer topic or technology and are interested in being a technical editor who reviews manuscripts for technical accuracy

Contact us at: **nreducation@newriders.com**

- If you are an instructor from an educational institution who wants to preview New Riders books for classroom use. Email should include your name, title, school, department, address, phone number, office days/hours, text in use, and enrollment, along with your request for desk/examination copies and/or additional information.

Contact us at: **nrmedia@newriders.com**

- If you are a member of the media who is interested in reviewing copies of New Riders books. Send your name, mailing address, and email address, along with the name of the publication or Web site you work for.

BULK PURCHASES/CORPORATE SALES

If you are interested in buying 10 or more copies of a title or want to set up an account for your company to purchase directly from the publisher at a substantial discount, contact us at 800-382-3419 or email your contact information to `corpsales@pearsontechgroup.com`. A sales representative will contact you with more information.

WRITE TO US

New Riders Publishing
201 W. 103rd St.
Indianapolis, IN 46290-1097

CALL/FAX US

Toll-free (800) 571-5840
If outside U.S. (317) 581-3500
Ask for New Riders
FAX: (317) 581-4663

New Riders

WWW.NEWRIDERS.COM

Solutions from experts you know and trust.

www.informit.com

OPERATING SYSTEMS

WEB DEVELOPMENT

PROGRAMMING

NETWORKING

CERTIFICATION

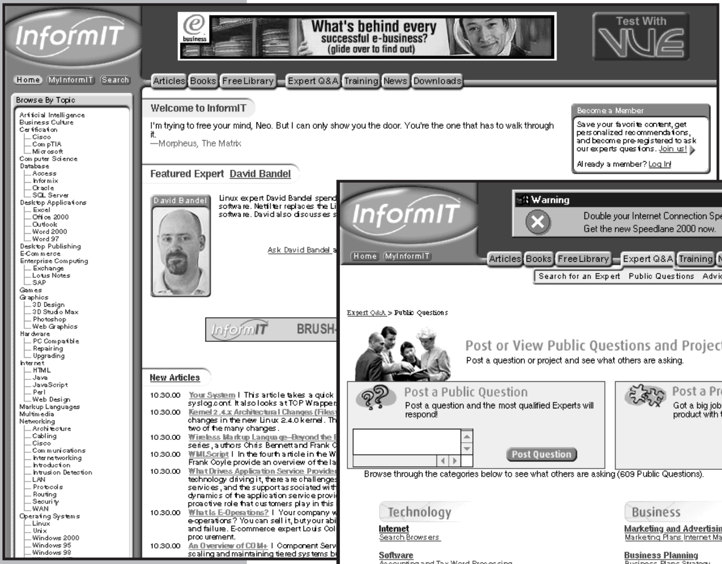
AND MORE…

Expert Access. Free Content.

New Riders has partnered with **InformIT.com** to bring technical information to your desktop. Drawing on New Riders authors and reviewers to provide additional information on topics you're interested in, **InformIT.com** has free, in-depth information you won't find anywhere else.

- **Master the skills you need, when you need them**
- **Call on resources from some of the best minds in the industry**
- **Get answers when you need them, using InformIT's comprehensive library or live experts online**
- **Go above and beyond what you find in New Riders books, extending your knowledge**

As an **InformIT** partner, **New Riders** has shared the wisdom and knowledge of our authors with you online. Visit **InformIT.com** to see what you're missing.



Browse By Topic

Artificial Intelligence
Business Culture
Certification
- Cisco
- CompTIA
- Microsoft

Computer Science
Database
- Access
- Informix
- Oracle
- SQL Server

Desktop Applications
- Excel
- Office 2000
- Outlook
- Word 2000
- Word 97

Desktop Publishing
E-Commerce
Enterprise Computing
- Exchange
- Lotus Notes
- SAP

Games
Graphics
- 3D Design
- 3D Studio Max
- Photoshop
- Web Graphics

Hardware
- PC Compatible
- Repairing
- Upgrading

Internet
- HTML
- Java
- JavaScript
- Perl
- Web Design

Markup Languages
Multimedia
Networking
- Architecture
- Cabling
- Cisco
- Communications
- Internetworking
- Introduction
- Intrusion Detection
- LAN
- Protocols
- Routing
- Security
- WAN

Operating Systems
- Linux
- Unix
- Windows 2000
- Windows 95
- Windows 98

Welcome to InformIT

I'm trying to free your mind, Neo. But I can only show you the door. You're the one that has to walk through it.
—Morpheus, The Matrix

Become a Member

Save your favorite content, get personalized recommendations, and become pre-registered to ask our experts questions. Join us!

Already a member? Log In!

Featured Expert David Bandel

David Bandel

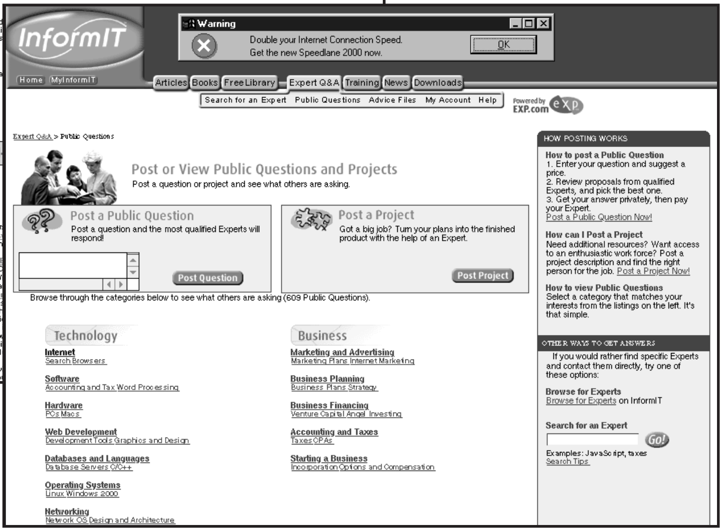
Linux expert David Bandel spend
software. Netfilter replaces the Li
software. David also discusses s

Ask David Bandel a

InformIT BRUSH-

New Articles

10.30.00 Your System | This article takes a quick
syslog.conf. It also looks at TCP Wrapper
10.30.00 Kernel 2.4.x Architectural Changes (Files
changes in the new Linux 2.4.0 kernel. Th
two of the many changes.
10.30.00 Wireless Markup Language—Beyond the
series, authors Chris Bennett and Frank C
10.30.00 WMLScript | In the fourth article in the W
Frank Coyle provide an overview of the la
10.30.00 What Drives Application Service Provider
technology driving it, there are challenges
services, and the support associated with
dynamics of the application service provi
proactive role that customers play in this
10.30.00 What Is E-Operations? | Your company w
e-operations? You can sell it, but your abi
and failure. E-commerce expert Louis Col
procurement.
10.30.00 An Overview of COM+ | Component Serv
scaling and maintaining tiered systems b



Warning

Double your Internet Connection Speed.
Get the new Speedlane 2000 now.

OK

Expert Q&A > Public Questions

Post or View Public Questions and Projects

Post a question or project and see what others are asking.

Post a Public Question

Post a question and the most qualified Experts will respond!

Post a Project

Got a big job? Turn your plans into the finished product with the help of an Expert.

Browse through the categories below to see what others are asking (609 Public Questions).

Technology

Internet
Search Browsers

Software
Accounting and Tax Word Processing

Hardware
PCs Macs

Web Development
Development Tools Graphics and Design

Databases and Languages
Database Servers C/C++

Operating Systems
Linux Windows 2000

Networking
Network OS Design and Architecture

Business

Marketing and Advertising
Marketing Plans Internet Marketing

Business Planning
Business Plans Strategy

Business Financing
Venture Capital Angel Investing

Accounting and Taxes
Taxes CPAs

Starting a Business
Incorporation Options and Compensation

HOW POSTING WORKS

How to post a Public Question
1. Enter your question and suggest a price.
2. Review proposals from qualified Experts, and pick the best one.
3. Get your answer privately, then pay your Expert.
Post a Public Question Now!

How can I Post a Project
Need additional resources? Want access to an enthusiastic work force? Post a project description and find the right person for the job. Post a Project Now!

How to view Public Questions
Select a category that matches your interests from the listings on the left. It's that simple.

OTHER WAYS TO GET ANSWERS

If you would rather find specific Experts and contact them directly, try one of these options:

Browse for Experts
Browse for Experts on InformIT

Search for an Expert

Examples: JavaScript, taxes
Search Tips

InformIT™

www.informit.com ▪ www.newriders.com

New Riders

Colophon

The ruins of the Stabian Baths in Pompeii, captured by photographer Mel Curtis, are featured on the cover of this book. Said to be the largest and oldest of the baths, the Stabian baths also offered massages and poetry readings. Residents of Pompeii visited these public baths daily. The baths are named for their location on Stabian Street.

This book was written and edited in LaTeX, and then converted to Microsoft Word by New Riders and laid out in QuarkXPress. The font used for the body text is Bembo and MCPdigital. It was printed on 50# Husky Offset Smooth paper at R.R. Donnelley & Sons in Crawfordsville, Indiana. Prepress consisted of PostScript computer-to-plate technology (filmless process). The cover was printed at Moore Langen Printing in Terre Haute, Indiana, on Carolina, coated on one side.