

《Linux/UNIX系统编程手册》

图书基本信息

书名：《Linux/UNIX系统编程手册》

13位ISBN编号：9787115328670

10位ISBN编号：7115328676

出版时间：2014-1-2

出版社：人民邮电出版社

作者：Michael Kerrisk

页数：1176

译者：孙剑 许从年 董健,孙余强 郭光伟 陈舸

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《Linux/UNIX系统编程手册》

内容概要

《linux/unix系统编程手册(上、下册)》是介绍linux与unix编程接口的权威著作。linux编程资深专家michael kerrisk在书中详细描述了linux/unix系统编程所涉及的系统调用和库函数，并辅之以全面而清晰的代码示例。《linux/unix系统编程手册(上、下册)》涵盖了逾500个系统调用及库函数，并给出逾200个程序示例，另含88张表格和115幅示意图。

《linux/unix系统编程手册(上、下册)》总共分为64章，主要讲解了高效读写文件，对信号、时钟和定时器的运用，创建进程、执行程序，编写安全的应用程序，运用posix线程技术编写多线程程序，创建和使用共享库，运用管道、消息队列、共享内存和信号量技术来进行进程间通信，以及运用套接字api编写网络应用等内容。

《linux/unix系统编程手册(上、下册)》在汇聚大批 linux 专有特性(epoll、inotify、/proc)的同时，还特意强化了对unix标准(posix、sus)的论述，彻底达到了“鱼与熊掌，二者得兼”的效果，这也堪称本书的最大亮点。

《linux/unix系统编程手册(上、下册)》布局合理，论述清晰，说理透彻，尤其是作者对示例代码的构思巧妙，独具匠心，仔细研读定会受益良多。本书适合从事linux/unix系统开发、运维工作的技术人员阅读，同时也可作为高校计算机专业学生的参考研习资料。

《Linux/UNIX系统编程手册》

作者简介

Michael Kerrisk(<http://man7.org>)具有20多年的unix系统使用和编程经验，所开设的unix系统编程周训课程更是不计其数。自2004年起，他开始维护手册页项目，该项目旨在生成描述linux内核以及glibc编程api的手册页。他已经撰写或与他人合著了250多篇手册页，至今仍积极参与对linux内核 / 用户空间接口的测试和设计评审工作。

Michael与家人居住在德国慕尼黑。

书籍目录

上册

- 第1章 历史和标准 1
 - 1.1 unix和c语言简史 1
 - 1.2 linux简史 4
 - 1.2.1 gnu项目 4
 - 1.2.2 linux内核 5
 - 1.3 标准化 8
 - 1.3.1 c编程语言 8
 - 1.3.2 首个posix标准 9
 - 1.3.3 x/open公司和the open group 10
 - 1.3.4 susv3和posix.1-2001 10
 - 1.3.5 susv4和posix.1-2008 12
 - 1.3.6 unix标准时间表 12
 - 1.3.7 实现标准 14
 - 1.3.8 linux、标准、linux标准规范(linux standard base) 14
 - 1.4 总结 15
- 第2章 基本概念 17
 - 2.1 操作系统的核心—内核 17
 - 2.2 shell 19
 - 2.3 用户和组 20
 - 2.4 单根目录层级、目录、链接及文件 21
 - 2.5 文件i/o模型 23
 - 2.6 程序 24
 - 2.7 进程 25
 - 2.8 内存映射 27
 - 2.9 静态库和共享库 28
 - 2.10 进程间通信及同步 28
 - 2.11 信号 29
 - 2.12 线程 30
 - 2.13 进程组和shell任务控制 30
 - 2.14 会话、控制终端和控制进程 30
 - 2.15 伪终端 31
 - 2.16 日期和时间 31
 - 2.17 客户端服务器架构 32
 - 2.18 实时性 32
 - 2.19 /proc文件系统 33
 - 2.20 总结 33
- 第3章 系统编程概念 34
 - 3.1 系统调用 34
 - 3.2 库函数 36
 - 3.3 标准c语言函数库；gnu c语言函数库(glibc) 37
 - 3.4 处理来自系统调用和库函数的错误 38
 - 3.5 关于本书示例程序的注意事项 40
 - 3.5.1 命令行选项及参数 40
 - 3.5.2 常用的函数及头文件 40
 - 3.6 可移植性问题 49
 - 3.6.1 特性测试宏 49

- 3.6.2 系统数据类型 51
- 3.6.3 其他的可移植性问题 53
- 3.7 总结 54
- 3.8 练习 55
- 第4章 文件i/o：通用的i/o模型 56
 - 4.1 概述 56
 - 4.2 通用i/o 58
 - 4.3 打开一个文件：open() 58
 - 4.3.1 open()调用中的flags参数 60
 - 4.3.2 open()函数的错误 63
 - 4.3.3 creat()系统调用 64
 - 4.4 读取文件内容：read() 64
 - 4.5 数据写入文件：write() 65
 - 4.6 关闭文件：close() 66
 - 4.7 改变文件偏移量：lseek() 66
 - 4.8 通用i/o模型以外的操作：ioctl() 70
 - 4.9 总结 71
 - 4.10 练习 71
- 第5章 深入探究文件i/o 72
 - 5.1 原子操作和竞争条件 72
 - 5.2 文件控制操作：fcntl() 75
 - 5.3 打开文件的状态标志 75
 - 5.4 文件描述符和打开文件之间的关系 76
 - 5.5 复制文件描述符 78
 - 5.6 在文件特定偏移量处的i/o：pread()和pwrite() 80
 - 5.7 分散输入和集中输出(scatter-gather i/o)：readv()和writev() 81
 - 5.8 截断文件：truncate()和ftruncate()系统调用 84
 - 5.9 非阻塞i/o 84
 - 5.10 大文件i/o 85
 - 5.11 /dev/fd目录 88
 - 5.12 创建临时文件 88
 - 5.13 总结 90
 - 5.14 练习 90
- 第6章 进程 92
 - 6.1 进程和程序 92
 - 6.2 进程号和父进程号 93
 - 6.3 进程内存布局 94
 - 6.4 虚拟内存管理 97
 - 6.5 栈和栈帧 99
 - 6.6 命令行参数(argc, argv) 99
 - 6.7 环境列表 101
 - 6.8 执行非局部跳转：setjmp()和longjmp() 106
 - 6.9 总结 111
 - 6.9 练习 112
- 第7章 内存分配 113
 - 7.1 在堆上分配内存 113
 - 7.1.1 调整program break：brk()和sbrk() 113
 - 7.1.2 在堆上分配内存：malloc()和free() 114
 - 7.1.3 malloc()和free()的实现 117

- 7.1.4 在堆上分配内存的其他方法 120
- 7.2 在堆栈上分配内存：alloca() 122
- 7.3 总结 123
- 7.4 练习 123
- 第8章 用户和组 124
 - 8.1 密码文件：/etc/passwd 124
 - 8.2 shadow密码文件：/etc/shadow 125
 - 8.3 组文件：/etc/group 126
 - 8.4 获取用户和组的信息 127
 - 8.5 密码加密和用户认证 132
 - 8.6 总结 135
 - 8.7 练习 135
- 第9章 进程凭证 136
 - 9.1 实际用户id和实际组id 136
 - 9.2 有效用户id和有效组id 136
 - 9.3 set-user-id和set-group-id程序 137
 - 9.4 保存set-user-id和保存set-group-id 138
 - 9.5 文件系统用户id和组id 139
 - 9.6 辅助组id 140
 - 9.7 获取和修改进程凭证 140
 - 9.7.1 获取和修改实际、有效和保存设置标识 140
 - 9.7.2 获取和修改文件系统id 145
 - 9.7.3 获取和修改辅助组id 145
 - 9.7.4 修改进程凭证的系统调用总结 146
 - 9.7.5 示例：显示进程凭证 148
 - 9.8 总结 149
 - 9.9 习题 150
- 第10章 时间 151
 - 10.1 日历时间(calendar time) 151
 - 10.2 时间转换函数 153
 - 10.2.1 将time_t转换为可打印格式 153
 - 10.2.2 time_t和分解时间之间的转换 154
 - 10.2.3 分解时间和打印格式之间的转换 155
 - 10.3 时区 161
 - 10.4 地区(locale) 163
 - 10.5 更新系统时钟 167
 - 10.6 软件时钟(jiffies) 168
 - 10.7 进程时间 168
 - 10.8 总结 171
 - 10.9 练习 172
- 第11章 系统限制和选项 173
 - 11.1 系统限制 174
 - 11.2 在运行时获取系统限制(和选项) 176
 - 11.3 运行时获取与文件相关的限制(和选项) 178
 - 11.4 不确定的限制 179
 - 11.5 系统选项 180
 - 11.6 总结 181
 - 11.7 练习 182
- 第12章 系统和进程信息 183

- 12.1 /proc文件系统 183
 - 12.1.1 获取与进程有关的信息：/proc/pid 183
 - 12.1.2 /proc目录下的系统信息 185
 - 12.1.3 访问/proc文件 186
- 12.2 系统标识：uname() 188
- 12.3 总结 190
- 12.4 练习 190
- 第13章 文件i/o缓冲 191
 - 13.1 文件i/o的内核缓冲：缓冲区高速缓存 191
 - 13.2 stdio库的缓冲 194
 - 13.3 控制文件i/o的内核缓冲 196
 - 13.4 i/o缓冲小结 200
 - 13.5 就i/o模式向内核提出建议 201
 - 13.6 绕过缓冲区高速缓存：直接i/o 202
 - 13.7 混合使用库函数和系统调用进行文件i/o 204
 - 13.8 总结 205
 - 13.9 练习 205
- 第14章 系统编程概念 207
 - 14.1 设备专用文件(设备文件) 207
 - 14.2 磁盘和分区 208
 - 14.3 文件系统 209
 - 14.4 i节点 211
 - 14.5 虚拟文件系统(vfs) 213
 - 14.6 日志文件系统 214
 - 14.7 单根目录层级和挂载点 215
 - 14.8 文件系统的挂载和卸载 216
 - 14.8.1 挂载文件系统：mount() 217
 - 14.8.2 卸载文件系统：umount()和umount2() 222
 - 14.9 高级挂载特性 223
 - 14.9.1 在多个挂载点挂载文件系统 224
 - 14.9.2 多次挂载同一挂载点 224
 - 14.9.3 基于每次挂载的挂载标志 225
 - 14.9.4 绑定挂载 225
 - 14.9.5 递归绑定挂载 226
 - 14.10 虚拟内存文件系统：tmpfs 227
 - 14.11 获得与文件系统有关的信息：statvfs() 228
 - 14.12 总结 229
 - 14.13 练习 230
- 第15章 文件属性 231
 - 15.1 获取文件信息：stat() 231
 - 15.2 文件时间戳 236
 - 15.2.1 使用utime()和utimes()来改变文件时间戳 238
 - 15.2.2 使用utimensat()和futimens()改变文件时间戳 239
 - 15.3 文件属主 241
 - 15.3.1 新建文件的属主 241
 - 15.3.2 改变文件属主：chown()、fchown()和lchown() 241
 - 15.4 文件权限 244
 - 15.4.1 普通文件的权限 244
 - 15.4.2 目录权限 246

- 15.4.3 权限检查算法 246
- 15.4.4 检查对文件的访问权限：access() 248
- 15.4.5 set-user-id、set-group-id和sticky位 249
- 15.4.6 进程的文件模式创建掩码：umask() 249
- 15.4.7 更改文件权限：chmod()和fchmod() 251
- 15.5 i节点标志(ext2扩展文件属性) 252
- 15.6 总结 256
- 15.7 练习 256
- 第16章 扩展属性 258
 - 16.1 概述 258
 - 16.2 扩展属性的实现细节 260
 - 16.3 操控扩展属性的系统调用 260
 - 16.4 总结 264
 - 16.5 练习 264
- 第17章 访问控制列表 265
 - 17.1 概述 265
 - 17.2 acl权限检查算法 267
 - 17.3 acl的长、短文本格式 268
 - 17.4 acl_mask型ace和acl组分类 269
 - 17.5 getfacl和setfacl命令 270
 - 17.6 默认acl与文件创建 271
 - 17.7 acl在实现方面的限制 272
 - 17.8 acl api 273
 - 17.9 总结 280
 - 17.10 练习 280
- 第18章 目录与链接 281
 - 18.1 目录和(硬)链接 281
 - 18.2 符号(软)链接 283
 - 18.3 创建和移除(硬)链接：link()和unlink() 286
 - 18.4 更改文件名：rename() 289
 - 18.5 使用符号链接：symlink()和readlink() 290
 - 18.6 创建和移除目录：mkdir()和rmdir() 291
 - 18.7 移除一个文件或目录：remove() 292
 - 18.8 读目录：opendir()和readdir() 292
 - 18.9 文件树遍历：nftw() 297
 - 18.10 进程的当前工作目录 301
 - 18.11 针对目录文件描述符的相关操作 303
 - 18.12 改变进程的根目录：chroot() 304
 - 18.13 解析路径名：realpath() 306
 - 18.14 解析路径名字符串：dirname()和basename() 307
 - 18.15 总结 309
 - 18.16 练习 309
- 第19章 监控文件事件 311
 - 19.1 概述 311
 - 19.2 inotify api 312
 - 19.3 inotify事件 313
 - 19.4 读取inotify事件 315
 - 19.5 队列限制和/proc文件 319
 - 19.6 监控文件的旧有系统：dnotify 320

- 19.7 总结 320
- 19.8 练习 320
- 第20章 信号：基本概念 321
 - 20.1 概念和概述 321
 - 20.2 信号类型和默认行为 323
 - 20.3 改变信号处置：signal() 329
 - 20.4 信号处理器简介 330
 - 20.5 发送信号：kill() 333
 - 20.6 检查进程的存在 334
 - 20.7 发送信号的其他方式：raise()和killpg() 335
 - 20.8 显示信号描述 336
 - 20.9 信号集 337
 - 20.10 信号掩码(阻塞信号传递) 339
 - 20.11 处于等待状态的信号 341
 - 20.12 不对信号进行排队处理 341
 - 20.13 改变信号处置：sigaction() 345
 - 20.14 等待信号：pause() 346
 - 20.15 总结 347
 - 20.16 练习 347
- 第21章 信号：信号处理器函数 348
 - 21.1 设计信号处理器函数 348
 - 21.1.1 再论信号的非队列化处理 348
 - 21.1.2 可重入函数和异步信号安全函数 349
 - 21.1.3 全局变量和sig_atomic_t数据类型 353
 - 21.2 终止信号处理器函数的其他方法 354
 - 21.2.1 在信号处理器函数中执行非本地跳转 354
 - 21.2.2 异常终止进程：abort() 358
 - 21.3 在备选栈中处理信号：sigaltstack() 358
 - 21.4 sa_siginfo标志 361
 - 21.5 系统调用的中断和重启 366
 - 21.6 总结 368
 - 21.7 练习 369
- 第22章 信号：高级特性 370
 - 22.1 核心转储文件 370
 - 22.2 传递、处置及处理的特殊情况 372
 - 22.3 可中断和不可中断的进程睡眠状态 373
 - 22.4 硬件产生的信号 374
 - 22.5 信号的同步生成和异步生成 374
 - 22.6 信号传递的时机与顺序 375
 - 22.7 signal()的实现及可移植性 376
 - 22.8 实时信号 378
 - 22.8.1 发送实时信号 379
 - 22.8.2 处理实时信号 380
 - 22.9 使用掩码来等待信号：sigsuspend() 384
 - 22.10 以同步方式等待信号 387
 - 22.11 通过文件描述符来获取信号 390
 - 22.12 利用信号进行进程间通信 393
 - 22.13 早期的信号api(system v和bsd) 393
 - 22.14 总结 395

- 22.15 练习 396
- 第23章 定时器与休眠 397
 - 23.1 间隔定时器 397
 - 23.2 定时器的调度及精度 402
 - 23.3 为阻塞操作设置超时 402
 - 23.4 暂停运行(休眠)一段固定时间 404
 - 23.4.1 低分辨率休眠: sleep() 404
 - 23.4.2 高分辨率休眠: nanosleep() 404
 - 23.5 posix时钟 407
 - 23.5.1 获取时钟的值: clock_gettime() 407
 - 23.5.2 设置时钟的值: clock_settime() 408
 - 23.5.3 获取特定进程或线程的时钟id 408
 - 23.5.4 高分辨率休眠的改进版: clock_nanosleep() 409
 - 23.6 posix间隔式定时器 410
 - 23.6.1 创建定时器: timer_create() 410
 - 23.6.2 配备和解除定时器: timer_settime() 412
 - 23.6.3 获取定时器的当前值: timer_gettime() 413
 - 23.6.4 删除定时器: timer_delete() 413
 - 23.6.5 通过信号发出通知 414
 - 23.6.6 定时器溢出 417
 - 23.6.7 通过线程来通知 417
 - 23.7 利用文件描述符进行通知的定时器: timerfd api 420
 - 23.8 总结 423
 - 23.9 练习 424
- 第24章 进程的创建 425
 - 24.1 fork()、exit()、wait()以及execve()的简介 425
 - 24.2 创建新进程: fork() 427
 - 24.2.1 父、子进程间的文件共享 428
 - 24.2.2 fork()的内存语义 430
 - 24.3 系统调用vfork() 433
 - 24.4 fork()之后的竞争条件(race condition) 434
 - 24.5 同步信号以规避竞争条件 436
 - 24.6 总结 438
 - 24.7 练习 439
- 第25章 进程的终止 440
 - 25.1 进程的终止: _exit()和exit() 440
 - 25.2 进程终止的细节 441
 - 25.3 退出处理程序 442
 - 25.4 fork()、stdio缓冲区以及_exit()之间的交互 445
 - 25.5 总结 446
 - 25.6 练习 446
- 第26章 监控子进程 447
 - 26.1 等待子进程 447
 - 26.1.1 系统调用wait() 447
 - 26.1.2 系统调用waitpid() 449
 - 26.1.3 等待状态值 450
 - 26.1.4 从信号处理程序中终止进程 454
 - 26.1.5 系统调用waitid() 455
 - 26.1.6 系统调用wait3()和wait4() 456

- 26.2 孤儿进程与僵尸进程 457
- 26.3 sigchld信号 459
 - 26.3.1 为sigchld建立信号处理程序 459
 - 26.3.2 向已停止的子进程发送sigchld信号 462
 - 26.3.3 忽略终止的子进程 462
- 26.4 总结 464
- 26.5 练习 464
- 第27章 程序的执行 465
 - 27.1 执行新程序：execve() 465
 - 27.2 exec()库函数 468
 - 27.2.1 环境变量path 469
 - 27.2.2 将程序参数指定为列表 470
 - 27.2.3 将调用者的环境传递给新程序 471
 - 27.2.4 执行由文件描述符指代的程序：fexecve() 471
 - 27.3 解释器脚本 472
 - 27.4 文件描述符与exec() 474
 - 27.5 信号与exec() 477
 - 27.6 执行shell命令：system() 477
 - 27.7 system()的实现 480
 - 27.8 总结 485
 - 27.9 练习 485
- 第28章 详述进程创建和程序执行 487
 - 28.1 进程记账 487
 - 28.2 系统调用clone() 493
 - 28.2.1 clone()的flags参数 497
 - 28.2.2 因克隆生成的子进程而对waitpid()进行的扩展 503
 - 28.3 进程的创建速度 503
 - 28.4 exec()和fork()对进程属性的影响 505
 - 28.5 总结 508
 - 28.6 练习 508
- 第29章 线程：介绍 509
 - 29.1 概述 509
 - 29.2 pthreads api的详细背景 511
 - 29.3 创建线程 513
 - 29.4 终止线程 514
 - 29.5 线程id(thread id) 514
 - 29.6 连接(joining)已终止的线程 515
 - 29.7 线程的分离 517
 - 29.8 线程属性 518
 - 29.9 线程vs进程 518
 - 29.10 总结 519
 - 29.11 练习 519
- 第30章 线程：线程同步 521
 - 30.1 保护对共享变量的访问：互斥量 521
 - 30.1.1 静态分配的互斥量 524
 - 30.1.2 加锁和解锁互斥量 524
 - 30.1.3 互斥量的性能 526
 - 30.1.4 互斥量的死锁 527
 - 30.1.5 动态初始化互斥量 527

- 30.1.6 互斥量的属性 528
- 30.1.7 互斥量类型 528
- 30.2 通知状态的改变：条件变量(condition variable) 529
 - 30.2.1 由静态分配的条件变量 530
 - 30.2.2 通知和等待条件变量 531
 - 30.2.3 测试条件变量的判断条件(predicate) 534
 - 30.2.4 示例程序：连接任意已终止线程 534
 - 30.2.5 经由动态分配的条件变量 537
- 30.3 总结 538
- 30.4 练习 538
- 第31章 线程：线程安全和每线程存储 539
 - 31.1 线程安全(再论可重入性) 539
 - 31.2 一次性初始化 541
 - 31.3 线程特有数据 542
 - 31.3.1 库函数视角下的线程特有数据 542
 - 31.3.2 线程特有数据api概述 543
 - 31.3.3 线程特有数据api详述 543
 - 31.3.4 使用线程特有数据api 545
 - 31.3.5 线程特有数据的实现限制 549
 - 31.4 线程局部存储 549
 - 31.5 总结 550
 - 31.6 练习 551
- 第32章 线程：线程取消 552
 - 32.1 取消一个线程 552
 - 32.2 取消状态及类型 552
 - 32.3 取消点 553
 - 32.4 线程可取消性的检测 556
 - 32.5 清理函数(cleanup handler) 556
 - 32.6 异步取消 559
 - 32.7 总结 560
- 第33章 线程：更多细节 561
 - 33.1 线程栈 561
 - 33.2 线程和信号 562
 - 33.2.1 unix信号模型如何映射到线程中 562
 - 33.2.2 操作线程信号掩码 563
 - 33.2.3 向线程发送信号 563
 - 33.2.4 妥善处理异步信号 564
 - 33.3 线程和进程控制 564
 - 33.4 线程实现模型 566
 - 33.5 linux posix线程的实现 567
 - 33.5.1 linuxthreads 567
 - 33.5.2 nptl 569
 - 33.5.3 哪一种线程实现 570
 - 33.6 pthread api的高级特性 572
 - 33.7 总结 572
 - 33.8 练习 572
- 下册
- 第34章 进程组、会话和作业控制 573
 - 34.1 概述 573

- 34.2 进程组 575
- 34.3 会话 577
- 34.4 控制终端和控制进程 578
- 34.5 前台和后台进程组 580
- 34.6 sighup信号 581
 - 34.6.1 在shell中处理sighup信号 581
 - 34.6.2 sighup和控制进程的终止 583
- 34.7 作业控制 585
 - 34.7.1 在shell中使用作业控制 585
 - 34.7.2 实现作业控制 587
 - 34.7.3 处理作业控制信号 591
 - 34.7.4 孤儿进程组(sighup回顾) 594
- 34.8 总结 598
- 34.9 习题 599
- 第35章 进程优先级和调度 600
 - 35.1 进程优先级(nice值) 600
 - 35.2 实时进程调度概述 603
 - 35.2.1 sched_rr策略 604
 - 35.2.2 sched_fifo策略 605
 - 35.2.3 sched_batch和sched_idle策略 605
 - 35.3 实时进程调用api 605
 - 35.3.1 实时优先级范围 606
 - 35.3.2 修改和获取策略和优先级 606
 - 35.3.3 释放cpu 611
 - 35.3.4 sched_rr时间片 611
 - 35.4 cpu亲和力 612
 - 35.5 总结 614
 - 35.6 习题 615
- 第36章 进程资源 617
 - 36.1 进程资源使用 617
 - 36.2 进程资源限制 619
 - 36.3 特定资源限制细节 623
 - 36.4 总结 627
 - 36.5 习题 627
- 第37章 daemon 628
 - 37.1 概述 628
 - 37.2 创建一个daemon 629
 - 37.3 编写daemon指南 632
 - 37.4 使用sighup重新初始化一个daemon 632
 - 37.5 使用syslog记录消息和错误 635
 - 37.5.1 概述 635
 - 37.5.2 syslog api 636
 - 37.5.3 /etc/syslog.conf文件 640
 - 37.6 总结 641
 - 37.7 习题 641
- 第38章 编写安全的特权程序 642
 - 38.1 是否需要一个set-user-id或set-group-id程序? 642
 - 38.2 以最小权限操作 643
 - 38.3 小心执行程序 645

- 38.4 避免暴露敏感信息 646
- 38.5 确定进程的边界 647
- 38.6 小心信号和竞争条件 647
- 38.7 执行文件操作和文件i/o的缺陷 648
- 38.8 不要完全相信输入和环境 648
- 38.9 小心缓冲区溢出 649
- 38.10 小心拒绝服务攻击 650
- 38.11 检查返回状态和安全地处理失败情况 651
- 38.12 总结 651
- 38.13 习题 652
- 第39章 能力 653
 - 39.1 能力基本原理 653
 - 39.2 linux能力 654
 - 39.3 进程和文件能力 654
 - 39.3.1 进程能力 654
 - 39.3.2 文件能力 655
 - 39.3.3 进程许可和有效能力集的目的 657
 - 39.3.4 文件许可和有效能力集的目的 657
 - 39.3.5 进程和文件可继承集的目的 658
 - 39.3.6 在shell中给文件赋予能力和查看文件能力 658
 - 39.4 现代能力实现 659
 - 39.5 在exec()中转变进程能力 659
 - 39.5.1 能力边界集 660
 - 39.5.2 保持root语义 660
 - 39.6 改变用户id对进程能力的影响 661
 - 39.7 用编程的方式改变进程能力 661
 - 39.8 创建仅包含能力的环境 665
 - 39.9 发现程序所需的能力 667
 - 39.10 不具备文件能力的老式内核和系统 667
 - 39.11 总结 669
 - 39.12 习题 669
- 第40章 登录记账 670
 - 40.1 utmp和wtmp文件概述 670
 - 40.2 utmpx api 671
 - 40.3 utmpx结构 671
 - 40.4 从utmp和wtmp文件中检索信息 673
 - 40.5 获取登录名称: getlogin() 676
 - 40.6 为登录会话更新utmp和wtmp文件 677
 - 40.7 lastlog文件 681
 - 40.8 总结 683
 - 40.9 习题 683
- 第41章 共享库基础 684
 - 41.1 目标库 684
 - 41.2 静态库 685
 - 41.3 共享库概述 686
 - 41.4 创建和使用共享库——首回合 687
 - 41.4.1 创建一个共享库 687
 - 41.4.2 位置独立的代码 687
 - 41.4.3 使用一个共享库 688

- 41.4.4 共享库soname 689
- 41.5 使用共享库的有用工具 691
- 41.6 共享库版本和命名规则 692
- 41.7 安装共享库 694
- 41.8 兼容与不兼容库比较 696
- 41.9 升级共享库 697
- 41.10 在目标文件中指定库搜索目录 698
- 41.11 在运行时找出共享库 700
- 41.12 运行时符号解析 700
- 41.13 使用静态库取代共享库 701
- 41.14 总结 702
- 41.15 习题 703
- 第42章 共享库高级特性 704
 - 42.1 动态加载库 704
 - 42.1.1 打开共享库：dlopen() 705
 - 42.1.2 错误诊断：dlerror() 706
 - 42.1.3 获取符号的地址：dlsym() 707
 - 42.1.4 关闭共享库：dlclose() 709
 - 42.1.5 获取与加载的符号相关的信息：dladdr() 710
 - 42.1.6 在主程序中访问符号 710
 - 42.2 控制符号的可见性 710
 - 42.3 链接器版本脚本 711
 - 42.3.1 使用版本脚本控制符号的可见性 712
 - 42.3.2 符号版本化 713
 - 42.4 初始化和终止函数 715
 - 42.5 预加载共享库 716
 - 42.6 监控动态链接器：ld_debug 716
 - 42.7 总结 717
 - 42.8 习题 718
- 第43章 进程间通信简介 719
 - 43.1 ipc工具分类 719
 - 43.2 通信工具 720
 - 43.3 同步工具 721
 - 43.4 ipc工具比较 723
 - 43.5 总结 727
 - 43.6 习题 727
- 第44章 管道和fifo 728
 - 44.1 概述 728
 - 44.2 创建和使用管道 730
 - 44.3 将管道作为一种进程同步的方法 735
 - 44.4 使用管道连接过滤器 737
 - 44.5 通过管道与shell命令进行通信：popen() 739
 - 44.6 管道和stdio缓冲 743
 - 44.7 fifo 743
 - 44.8 使用管道实现一个客户端/服务器应用程序 745
 - 44.9 非阻塞i/o 751
 - 44.10 管道和fifo中read()和write()的语义 752
 - 44.11 总结 753
 - 44.12 习题 754

- 第45章 system v ipc介绍 756
 - 45.1 概述 757
 - 45.2 ipc key 759
 - 45.3 关联数据结构和对象权限 761
 - 45.4 ipc标识符和客户端/服务器应用程序 763
 - 45.5 system v ipc get调用使用的算法 764
 - 45.6 ipcs和ipcrm命令 766
 - 45.7 获取所有ipc对象列表 767
 - 45.8 ipc限制 767
 - 45.9 总结 768
 - 45.10 习题 768
- 第46章 system v消息队列 769
 - 46.1 创建或打开一个消息队列 769
 - 46.2 交换消息 771
 - 46.2.1 发送消息 772
 - 46.2.2 接收消息 774
 - 46.3 消息队列控制操作 777
 - 46.4 消息队列关联数据结构 778
 - 46.5 消息队列的限制 780
 - 46.6 显示系统中所有消息队列 781
 - 46.7 使用消息队列实现客户端/服务器应用程序 783
 - 46.8 使用消息队列实现文件服务器应用程序 784
 - 46.9 system v消息队列的缺点 790
 - 46.10 总结 790
 - 46.11 习题 791
- 第47章 system v信号量 792
 - 47.1 概述 793
 - 47.2 创建或打开一个信号量集 795
 - 47.3 信号量控制操作 796
 - 47.4 信号量关联数据结构 798
 - 47.5 信号量初始化 801
 - 47.6 信号量操作 803
 - 47.7 多个阻塞信号量操作的处理 809
 - 47.8 信号量撤销值 810
 - 47.9 实现一个二元信号量协议 811
 - 47.10 信号量限制 814
 - 47.11 system v信号量的缺点 815
 - 47.12 总结 816
 - 47.13 习题 817
- 第48章 system v共享内存 818
 - 48.1 概述 818
 - 48.2 创建或打开一个共享内存段 819
 - 48.3 使用共享内存 820
 - 48.4 示例：通过共享内存传输数据 821
 - 48.5 共享内存存在虚拟内存中的位置 825
 - 48.6 在共享内存中存储指针 828
 - 48.7 共享内存控制操作 829
 - 48.8 共享内存关联数据结构 830
 - 48.9 共享内存的限制 832

- 48.10 总结 833
- 48.11 习题 833
- 第49章 内存映射 835
 - 49.1 概述 835
 - 49.2 创建一个映射：mmap() 837
 - 49.3 解除映射区域：munmap() 840
 - 49.4 文件映射 840
 - 49.4.1 私有文件映射 841
 - 49.4.2 共享文件映射 842
 - 49.4.3 边界情况 845
 - 49.4.4 内存保护和文件访问模式交互 846
 - 49.5 同步映射区域：msync() 847
 - 49.6 其他mmap()标记 848
 - 49.7 匿名映射 849
 - 49.8 重新映射一个映射区域：mremap() 852
 - 49.9 map_noreserve和过度利用交换空间 853
 - 49.10 map_fixed标记 854
 - 49.11 非线性映射：remap_file_pages() 855
 - 49.12 总结 857
 - 49.13 习题 858
- 第50章 虚拟内存操作 859
 - 50.1 改变内存保护：mprotect() 859
 - 50.2 内存锁：mlock()和mlockatt() 861
 - 50.3 确定内存驻留性：mincore() 864
 - 50.4 建议后续的内存使用模式：madvise() 866
 - 50.5 小结 868
 - 50.6 习题 868
- 第51章 posix ipc介绍 869
 - 51.1 api概述 869
 - 51.2 system v ipc与posix ipc比较 872
 - 51.3 总结 873
- 第52章 posix消息队列 874
 - 52.1 概述 874
 - 52.2 打开、关闭和断开链接消息队列 875
 - 52.3 描述符和消息队列之间的关系 877
 - 52.4 消息队列特性 878
 - 52.5 交换消息 882
 - 52.5.1 发送消息 882
 - 52.5.2 接收消息 883
 - 52.5.3 在发送和接收消息时设置超时时间 885
 - 52.6 消息通知 886
 - 52.6.1 通过信号接收通知 887
 - 52.6.2 通过线程接收通知 889
 - 52.7 linux特有的特性 891
 - 52.8 消息队列限制 892
 - 52.9 posix和system v消息队列比较 893
 - 52.10 总结 894
 - 52.11 习题 894
- 第53章 posix信号量 895

- 53.1 概述 895
- 53.2 命名信号量 895
 - 53.2.1 打开一个命名信号量 896
 - 53.2.2 关闭一个信号量 898
 - 53.2.3 删除一个命名信号量 898
- 53.3 信号量操作 899
 - 53.3.1 等待一个信号量 899
 - 53.3.2 发布一个信号量 901
 - 53.3.3 获取信号量的当前值 901
- 53.4 未命名信号量 903
 - 53.4.1 初始化一个未命名信号量 904
 - 53.4.2 销毁一个未命名信号量 906
- 53.5 与其他同步技术比较 906
- 53.6 信号量的限制 907
- 53.7 总结 908
- 53.8 习题 908
- 第54章 posix共享内存 909
 - 54.1 概述 909
 - 54.2 创建共享内存对象 910
 - 54.3 使用共享内存对象 913
 - 54.4 删除共享内存对象 915
 - 54.5 共享内存apis比较 915
 - 54.6 总结 916
 - 54.7 习题 917
- 第55章 文件加锁 918
 - 55.1 概述 918
 - 55.2 使用flock()给文件加锁 920
 - 55.2.1 锁继承与释放的语义 922
 - 55.2.2 flock()的限制 923
 - 55.3 使用fcntl()给记录加锁 923
 - 55.3.1 死锁 928
 - 55.3.2 示例：一个交互式加锁程序 928
 - 55.3.3 示例：一个加锁函数库 931
 - 55.3.4 锁的限制和性能 933
 - 55.3.5 锁继承和释放的语义 934
 - 55.3.6 锁定饿死和排队加锁请求的优先级 935
 - 55.4 强制加锁 935
 - 55.5 /proc/locks文件 938
 - 55.6 仅运行一个程序的单个实例 939
 - 55.7 老式加锁技术 941
 - 55.8 总结 942
 - 55.9 习题 943
- 第56章 socket：介绍 945
 - 56.1 概述 945
 - 56.2 创建一个socket：socket() 948
 - 56.3 将socket绑定到地址：bind() 948
 - 56.4 通用socket地址结构：struct sockaddr 949
 - 56.5 流socket 950
 - 56.5.1 监听接入连接：listen() 951

- 56.5.2 接受连接 : accept() 952
- 56.5.3 连接到对等socket : connect() 952
- 56.5.4 流socket i/o 953
- 56.5.5 连接终止 : close() 953
- 56.6 数据报socket 953
 - 56.6.1 交换数据报 : recvfrom和sendto() 954
 - 56.6.2 在数据报socket上使用connect() 955
- 56.7 总结 956
- 第57章 socket : unix domain 957
 - 57.1 unix domain socket地址 : struct sockaddr_un 957
 - 57.2 unix domain中的流socket 959
 - 57.3 unix domain中的数据报socket 962
 - 57.4 unix domain socket权限 965
 - 57.5 创建互联socket对 : socketpair() 965
 - 57.6 linux抽象socket名空间 966
 - 57.7 总结 967
 - 57.8 习题 967
- 第58章 socket : tcp/ip网络基础 968
 - 58.1 因特网 968
 - 58.2 联网协议和层 969
 - 58.3 数据链路层 971
 - 58.4 网络层 : ip 971
 - 58.5 ip地址 973
 - 58.6 传输层 975
 - 58.6.1 端口号 975
 - 58.6.2 用户数据报协议(udp) 976
 - 58.6.3 传输控制协议(tcp) 977
 - 58.7 请求注解(rfc) 979
 - 58.8 总结 980
- 第59章 socket : internet domain 982
 - 59.1 internet domain socket 982
 - 59.2 网络字节序 982
 - 59.3 数据表示 984
 - 59.4 internet socket地址 986
 - 59.5 主机和服务转换函数概述 988
 - 59.6 inet_pton()和inet_ntop()函数 989
 - 59.7 客户端-服务器示例(数据报socket) 990
 - 59.8 域名系统(dns) 992
 - 59.9 /etc/services文件 994
 - 59.10 独立于协议的主机和服务转换 995
 - 59.10.1 getaddrinfo()函数 996
 - 59.10.2 释放addrinfo列表 : freeaddrinfo() 998
 - 59.10.3 错误诊断 : gai_strerror() 999
 - 59.10.4 getnameinfo()函数 999
 - 59.11 客户端-服务器示例(流式socket) 1000
 - 59.12 internet domain socket库 1006
 - 59.13 过时的主机和服务转换api 1010
 - 59.13.1 inet_aton()和inet_ntoa()函数 1010
 - 59.13.2 gethostbyname()和gethostbyaddr()函数 1010

- 59.13.3 getserverbyname()和getserverbyport()函数 1012
- 59.14 unix与internet domain socket比较 1013
- 59.15 更多信息 1014
- 59.16 总结 1014
- 59.17 习题 1015
- 第60章 socket : 服务器设计 1016
 - 60.1 迭代型和并发型服务器 1016
 - 60.2 迭代型udp echo服务器 1016
 - 60.3 并发型tcp echo服务器 1019
 - 60.4 并发型服务器的其他设计方案 1021
 - 60.5 inetd(internet超级服务器)守护进程 1023
 - 60.6 总结 1027
 - 60.7 练习 1027
- 第61章 socket : 高级主题 1028
 - 61.1 流式套接字上的部分读和部分写 1028
 - 61.2 shutdown()系统调用 1030
 - 61.3 专用于套接字的i/o系统调用 : recv()和send() 1033
 - 61.4 sendfile()系统调用 1034
 - 61.5 获取套接字地址 1036
 - 61.6 深入探讨tcp协议 1039
 - 61.6.1 tcp报文的格式 1039
 - 61.6.2 tcp序列号和确认机制 1041
 - 61.6.3 tcp协议状态机以及状态迁移图 1041
 - 61.6.4 tcp连接的建立 1043
 - 61.6.5 tcp连接的终止 1044
 - 61.6.6 在tcp套接字上调用shutdown() 1045
 - 61.6.7 time_wait状态 1045
 - 61.7 监视套接字 : netstat 1047
 - 61.8 使用tcpdump来监视tcp流量 1048
 - 61.9 套接字选项 1049
 - 61.10 so_reuseaddr套接字选项 1050
 - 61.11 在accept()中继承标记和选项 1051
 - 61.12 tcp vs udp 1052
 - 61.13 高级功能 1053
 - 61.13.1 带外数据 1053
 - 61.13.2 sendmsg()和recvmsg()系统调用 1053
 - 61.13.3 传递文件描述符 1054
 - 61.13.4 接收发送端的凭据 1054
 - 61.13.5 顺序数据包套接字 1055
 - 61.13.6 sctp以及dccp传输层协议 1055
 - 61.14 总结 1056
 - 61.15 练习 1056
- 第62章 终端 1058
 - 62.1 整体概览 1059
 - 62.2 获取和修改终端属性 1060
 - 62.3 stty命令 1062
 - 62.4 终端特殊字符 1063
 - 62.5 终端标志 1068
 - 62.6 终端的i/o模式 1073

- 62.6.1 规范模式 1073
- 62.6.2 非规范模式 1074
- 62.6.3 加工模式、cbreak模式以及原始模式 1075
- 62.7 终端线速(比特率) 1081
- 62.8 终端的行控制 1082
- 62.9 终端窗口大小 1084
- 62.10 终端标识 1085
- 62.11 总结 1086
- 62.12 练习 1087
- 第63章 其他备选的i/o模型 1088
- 63.1 整体概览 1088
- 63.1.1 水平触发和边缘触发 1091
- 63.1.2 在备选的i/o模型中采用非阻塞i/o 1092
- 63.2 i/o多路复用 1092
- 63.2.1 select()系统调用 1092
- 63.2.2 poll()系统调用 1097
- 63.2.3 文件描述符何时就绪? 1101
- 63.2.4 比较select()和poll() 1103
- 63.2.5 select()和poll()存在的问题 1105
- 63.3 信号驱动i/o 1105
- 63.3.1 何时发送“i/o就绪”信号 1109
- 63.3.2 优化信号驱动i/o的使用 1110
- 63.4 epoll编程接口 1113
- 63.4.1 创建epoll实例: epoll_create() 1113
- 63.4.2 修改epoll的兴趣列表: epoll_ctl() 1114
- 63.4.3 事件等待: epoll_wait() 1115
- 63.4.4 深入探究epoll的语义 1120
- 63.4.5 epoll同i/o多路复用的性能对比 1121
- 63.4.6 边缘触发通知 1122
- 63.5 在信号和文件描述符上等待 1124
- 63.5.1 pselect()系统调用 1125
- 63.5.2 self-pipe技巧 1126
- 63.6 总结 1128
- 63.7 练习 1129
- 第64章 伪终端 1130
- 64.1 整体概览 1130
- 64.2 unix98伪终端 1133
- 64.2.1 打开未使用的主设备: posix_openpt() 1134
- 64.2.2 修改从设备属主和权限: grantpt() 1135
- 64.2.3 解锁从设备: unlockpt() 1135
- 64.2.4 获取从设备名称: ptsname() 1136
- 64.3 打开主设备: ptymasteropen() 1136
- 64.4 将进程连接到伪终端: ptyfork() 1138
- 64.5 伪终端i/o 1140
- 64.6 实现script(1)程序 1142
- 64.7 终端属性和窗口大小 1146
- 64.8 bsd风格的伪终端 1146
- 64.9 总结 1148
- 64.10 练习 1149

附录a 跟踪系统调用	1151
附录b 解析命令行选项	1153
附录c 对null指针做转型	1159
附录d 内核配置	1161
附录e 更多信息源	1162
附录f 部分习题解答	1167

精彩短评

- 1、下半部分基本没怎么看。以后搞到这块再细读吧。不过对一些概念的理解加深了。
- 2、看APUE到一半决定放弃，转看这本的英文版，目前看完第28章，清晰易懂，决定坚持看完。。。
- 3、有英文电子版，内容实在太多，不过都很有用满满干货，需要多看几遍，有的需要时再查看
- 4、不夸张得说，这本书超越了APUE，尤其是对Linux系统编程中的关键概念讲解特别透彻，而且对linux 2.6.*内核的最新系统调用有很多介绍。
- 5、读过APUE和UNP之后读到这本书。此书更适合入门。准备面试的效果也是萌萌哒。
- 6、熟悉系统，巨厚的书快速扫过，非常详细linux编程的多数点都有涉及
- 7、很好的书，比apue新。介绍的也很通俗易懂，覆盖面也广，不过要自己发现重点。
- 8、粗略看了一遍，没有太艰涩的内容，对于工作了一段时间想提高的朋友会有帮助。有时间要仔细重看一遍，做下笔记
- 9、拿来学习Linux API
- 10、千辛万苦下到了pdf 两本太重了 你们倒是出个电子版啊！||好看，在图书馆找到了实体书 顾得连着看完了 好看！讲解清晰到位 翻译也挺好的 上下两卷翻译没沟通吧 文件描述符那里说不清
- 11、好大一本手册~
- 12、接口的介绍，大而全。
- 13、linuxC编程入门
- 14、不给五星的原因是由于，对于书对于知识网络的构建没有太好，如果读过深入了解计算机系统再来看这本书，就非常棒了
- 15、程序员必读，十分精彩
- 16、比APUE更详细，当手册查，这里讲IO模型讲的非常详细，可以重点看看。
- 17、unix 环境高级编程的开胃小菜
- 18、我觉得apue作为教材远不如这本.....apue还是做参考书吧.....
- 19、apue 混合服用疗效更好
- 20、必成经典，比APUE好
- 21、Apue可以安心退休了
- 22、花了几天时间，也只算是把这本书的内容大致过了一遍，还有很多细节的东西没有弄清楚，特别是信号、权限控制等等。UNIX真是一个复杂的系统！
这本书号称超越AUPE，读起来确实不那枯燥，特别是有很多特性的最佳实践，这是最实用的地方。翻译还可以，只是可能是很多人合译的原因，感觉有的章节译得好，有的章节不那么好。

精彩书评

1、花了几天时间，也只算是把这本书的内容大致过了一遍，还有很多细节的东西没有弄清楚，特别是信号、权限控制等等。UNIX真是一个复杂的系统！这本书号称超越AUPE，读起来确实不那枯燥，特别是有很多特性如何选择的最佳实践，这是最实用的地方！有些Linux上特有的特性如epoll，AUPE上也没有。翻译还可以，只是可能是很多人合译的原因，感觉有的章节译得好，有的章节不那么好。

章节试读

1、《Linux/UNIX系统编程手册》的笔记-第61页

线程某甲，甲就是甲，某甲是什么？双重假设吗？
线程甲，线程乙，或者甲线程，乙线程。

2、《Linux/UNIX系统编程手册》的笔记-第1325页

1. 关于 level-triggered 和 edge-triggered notification

2. I/O Multiplexing机制

select 和 poll

3. Signal-Driven I/O

4. epoll

3、《Linux/UNIX系统编程手册》的笔记-第10页

X/Opena联网服务。Open

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu000.com