МАЙКЛ КЕРРИСК

LINUX API
ИСЧЕРПЫВАЮЩЕЕ РУКОВОДСТВО

ПИТЕР®
Санкт-Петербург • Москва • Минск
2022

Краткое содержание

Предисловие 26
Глава 1. История и стандарты 37
Глава 2. Основные понятия 57
Глава 3. Общее представление о системном программировании 79
Глава 4. Файловый ввод-вывод: универсальная модель ввода-вывода 105
Глава 5. Файловый ввод-вывод: дополнительные сведения 123
Глава 6. Процессы 147
Глава 7. Выделение памяти 172
Глава 8. Пользователи и группы 186
Глава 9. Идентификаторы процессов 200
Глава 10. Время 220
Глава 11. Системные ограничения и возможности 246
Глава 12. Информация о системе и процессе 259
Глава 13. Буферизация файлового ввода-вывода 268
Глава 14. Файловые системы 287
Глава 15. Атрибуты файла 315
Глава 16. Расширенные атрибуты 346
Глава 17. Списки контроля доступа 354
Глава 18. Каталоги и ссылки 371
Глава 19. Мониторинг событий файлов 406
Глава 20. Сигналы: фундаментальные концепции 418
Глава 21. Сигналы: обработчики сигналов 447
Глава 22. Сигналы: дополнительные возможности 472
Глава 23. Таймеры и переход в режим сна 502
Глава 24. Создание процессов 533
Глава 25. Завершение работы процесса 549
Глава 26. Мониторинг дочерних процессов 557
Глава 27. Выполнение программы 576
Глава 28. Подробнее о создании процесса и выполнении программы 601
Глава 29. Потоки выполнения: введение 627
Глава 30. Потоки выполнения: синхронизация 641
Глава 31. Потоки выполнения: потоковая безопасность и локальное хранилище 662
Глава 32. Потоки выполнения: отмена потока 677
Глава 33. Потоки выполнения: дальнейшие подробности 686
Глава 34. Группы процессов, сессии и управление заданиями 702
Глава 35. Приоритеты процессов и их планирование 733
Глава 36. Ресурсы процессов 752

Глава 37. Демоны 764
Глава 38. Написание безопасных программ с повышенными привилегиями 780
Глава 39. Система возможностей 793
Глава 40. Учет входа в систему 811
Глава 41. Основы разделяемых библиотек 825
Глава 42. Продвинутые возможности разделяемых библиотек 849
Глава 43. Краткий обзор межпроцессного взаимодействия 866
Глава 44. Каналы и очереди FIFO 876
Глава 45. Отображение в память 906
Глава 46. Операции с виртуальной памятью 932
Глава 47. Введение в межпроцессное взаимодействие стандарта POSIX 942
Глава 48. Очереди сообщений стандарта POSIX 947
Глава 49. Семафоры стандарта POSIX 969
Глава 50. Разделяемая память POSIX 983
Глава 51. Блокировка файлов 991
Глава 52. Сокеты: введение 1021
Глава 53. Сокеты: домен UNIX 1035
Глава 54. Сокеты: основы сетей TCP/IP 1046
Глава 55. Сокеты: домены сети Интернет 1062
Глава 56. Сокеты: архитектура сервера 1095
Глава 57. Сокеты: углубленный материал 1108
Глава 58. Терминалы 1142
Глава 59. Альтернативные модели ввода/вывода 1174
Глава 60. Псевдотерминалы 1221
Список используемых источников 1241

Оглавление

Предисловие **26**
Цель книги 26
Для кого эта книга 26
Linux и UNIX 27
Структура книги 27
Примеры программ 28
Упражнения 29
Стандарты и портируемость 29
Ядро Linux и версии библиотеки C 29
Использование программного интерфейса других языков программирования 30
Об авторе 30
Благодарности 31
Разрешения 36
Обратная связь 36

Глава 1. История и стандарты **37**
1.1. Краткая история UNIX и языка C 37
1.2. Краткая история Linux 41
1.2.1. Проект GNU 41
1.2.2. Ядро Linux 42
1.3. Стандартизация 46
1.3.1. Язык программирования C 46
1.3.2. Первые стандарты POSIX 47
1.3.3. X/Open Company и Open Group 49
1.3.4. SUSv3 и POSIX.1-2001 49
1.3.5. SUSv4 и POSIX.1-2008 51
1.3.6. Этапы развития стандартов UNIX 52
1.3.7. Стандарты реализаций 53
1.3.8. Linux, стандарты и нормативная база Linux 54
1.4. Резюме 55

Глава 2. Основные понятия **57**
2.1. Основа операционной системы: ядро 57
2.2. Оболочка 60
2.3. Пользователи и группы 61
2.4. Иерархия одного каталога. Что такое каталоги, ссылки и файлы 62
2.5. Модель файлового ввода-вывода 65
2.6. Программы 66
2.7. Процессы 67
2.8. Отображение в памяти 71
2.9. Статические и совместно используемые библиотеки 71
2.10. Межпроцессное взаимодействие и синхронизация 72
2.11. Сигналы 73
2.12. Потоки 74
2.13. Группы процессов и управление заданиями в оболочке 74
2.14. Сессии, управляющие терминалы и управляющие процессы 75
2.15. Псевдотерминалы 75
2.16. Дата и время 76
2.17. Клиент-серверная архитектура 76
2.18. Выполнение действий в реальном масштабе времени 77
2.19. Файловая система /proc 78
2.20. Резюме 78

Глава 3. Общее представление о системном программировании 79

3.1. Системные вызовы 79
3.2. Библиотечные функции 82
3.3. Стандартная библиотека языка C; GNU-библиотека C (glibc) 82
3.4. Обработка ошибок, возникающих при системных вызовах и вызовах библиотечных функций 84
3.5. Пояснения по поводу примеров программ, приводимых в книге 86
3.5.1. Ключи и аргументы командной строки 87
3.5.2. Типовые функции и заголовочные файлы 87
3.6. Вопросы переносимости 96
3.6.1. Макросы проверки возможностей 96
3.6.2. Типы системных данных 98
3.6.3. Прочие вопросы, связанные с портированием 102
3.7. Резюме 104
3.8. Упражнение 104

Глава 4. Файловый ввод-вывод: универсальная модель ввода-вывода 105

4.1. Общее представление 105
4.2. Универсальность ввода-вывода 107
4.3. Открытие файла: open() 108
4.3.1. Аргумент flags системного вызова open() 109
4.3.2. Ошибки, возвращаемые из системного вызова open() 113
4.3.3. Системный вызов creat() 114
4.4. Чтение из файла: read() 114
4.5. Запись в файл: write() 115
4.6. Закрытие файла: close() 116
4.7. Изменение файлового смещения: lseek() 117
4.8. Операции, не вписывающиеся в модель универсального ввода-вывода: ioctl() 121
4.9. Резюме 122
4.10. Упражнения 122

Глава 5. Файловый ввод-вывод: дополнительные сведения 123

5.1. Атомарность и состояние гонки 123
5.2. Операции управления файлом: fcntl() 126
5.3. Флаги состояния открытого файла 127
5.4. Связь файловых дескрипторов с открытыми файлами 128
5.5. Дублирование дескрипторов файлов 129
5.6. Файловый ввод-вывод по указанному смещению: pread() и pwrite() 133
5.7. Ввод-вывод по принципу фрагментации-дефрагментации: readv() и writev() 134
5.8. Усечение файла: truncate() и ftruncate() 137
5.9. Неблокирующий ввод-вывод 138
5.10. Ввод-вывод, осуществляемый в отношении больших файлов 138
5.11. Каталог /dev/fd 142
5.12. Создание временных файлов 143
5.13. Резюме 144
5.14. Упражнения 145

Глава 6. Процессы 147

6.1. Процессы и программы 147
6.2. Идентификатор процесса и идентификатор родительского процесса 148
6.3. Структура памяти процесса 149
6.4. Управление виртуальной памятью 153
6.5. Стек и стековые фреймы 155

6.6. Аргументы командной строки (argc, argv) 156
6.7. Список переменных среды 158
6.8. Выполнение нелокального перехода: setjmp() и longjmp() 165
6.9. Резюме 170
6.10. Упражнения 171

Глава 7. Выделение памяти 172

7.1. Выделение памяти в куче 172
7.1.1. Установка крайней точки программы: brk() и sbrk() 172
7.1.2. Выделение памяти в куче: malloc() и free() 173
7.1.3. Реализация функций malloc() и free() 177
7.1.4. Другие методы выделения памяти в куче 180
7.2. Выделение памяти в стеке: alloca() 183
7.3. Резюме 184
7.4. Упражнения 185

Глава 8. Пользователи и группы 186

8.1. Файл паролей: /etc/passwd 186
8.2. Теневой файл паролей: /etc/shadow 187
8.3. Файл групп: /etc/group 188
8.4. Извлечение информации о пользователях и группах 189
8.5. Шифрование пароля и аутентификация пользователя 195
8.6. Резюме 198
8.7. Упражнения 199

Глава 9. Идентификаторы процессов 200

9.1. Реальный идентификатор пользователя и реальный идентификатор группы 200
9.2. Действующий идентификатор пользователя и действующий идентификатор группы 200
9.3. Программы с установленным идентификатором пользователя и установленным идентификатором группы 201
9.4. Сохраненный set-user-ID и сохраненный set-group-ID 203
9.5. Пользовательские и групповые ID файловой системы 204
9.6. Дополнительные групповые идентификаторы 205
9.7. Извлечение и модификация идентификаторов процессов 205
9.7.1. Извлечение и изменение реальных, действующих и сохраненных установленных идентификаторов 206
9.7.2. Извлечение и изменение идентификаторов файловой системы 212
9.7.3. Извлечение и изменение дополнительных групповых идентификаторов 213
9.7.4. Сводный обзор вызовов, предназначенных для изменения идентификаторов процесса 214
9.7.5. Пример: вывод на экран идентификаторов процесса 216
9.8. Резюме 218
9.9. Упражнения 218

Глава 10. Время 220

10.1. Календарное время 220
10.2. Функции преобразования представлений времени 222
10.2.1. Преобразование значений типа time_t к виду, подходящему для устройств вывода информации 223
10.2.2. Преобразования между time_t и разделенным календарным временем 223
10.2.3. Преобразования между разделенным календарным временем и временем в печатном виде 225

10.3. Часовые пояса 232
10.4. Локали 235
10.5. Обновление системных часов 239
10.6. Программные часы (мгновения) 240
10.7. Время процесса 241
10.8. Резюме 244
10.9. Упражнение 245

Глава 11. Системные ограничения и возможности 246

11.1. Системные ограничения 247
11.2. Извлечение в ходе выполнения программы значений ограничений (и возможностей) системы 251
11.3. Извлечение в ходе выполнения программы значений ограничений (и возможностей), связанных с файлами 253
11.4. Неопределенные ограничения 254
11.5. Системные возможности 255
11.6. Резюме 257
11.7. Упражнения 258

Глава 12. Информация о системе и процессе 259

12.1. Файловая система /proc 259
12.1.1. Получение информации о процессе: /proc/PID 259
12.1.2. Системная информация, находящаяся в /proc 262
12.1.3 Доступ к файлам, находящимся в /proc 263
12.2. Идентификация системы: uname() 264
12.3. Резюме 266
12.4. Упражнения 267

Глава 13. Буферизация файлового ввода-вывода 268

13.1. Буферизация файлового ввода-вывода при работе в режиме ядра: буферная кэш-память 268
13.2. Буферизация в библиотеке stdio 272
13.3. Управление буферизацией файлового ввода-вывода, осуществляемой в ядре 274
13.4. Обзор буферизации ввода-вывода 278
13.5. Уведомление ядра о схемах ввода-вывода 279
13.6. Обход буферной кэш-памяти: непосредственный ввод/вывод 281
13.7. Смешивание библиотечных функций и системных вызовов для файлового ввода-вывода 284
13.8. Резюме 285
13.9. Упражнения 285

Глава 14. Файловые системы 287

14.1. Специальные файлы устройств 287
14.2. Диски и разделы 289
14.3. Файловые системы 290
14.4. Индексные дескрипторы 292
14.5. Виртуальная файловая система 294
14.6. Журналируемые файловые системы 295
14.7. Иерархия одиночного каталога и точки монтирования 297
14.8. Монтирование и размонтирование файловых систем 298
14.8.1. Монтирование файловой системы: mount() 299
14.8.2. Размонтирование файловой системы: системные вызовы umount() и umount2() 305

14.9. Дополнительные функции монтирования 306
14.9.1. Монтирование файловой системы в нескольких точках монтирования 306
14.9.2. Создание стека монтирования в одной точке 307
14.9.3. Флаги монтирования, которые являются параметрами конкретной точки монтирования 307
14.9.4. Связанные (синонимичные) точки монтирования 308
14.9.5. Рекурсивное связанное монтирование 309
14.10. Файловая система виртуальной памяти: tmpfs 310
14.11. Получение информации о файловой системе: statvfs() 311
14.12. Резюме 313
14.13. Упражнение 314

Глава 15. Атрибуты файла 315

15.1. Извлечение информации о файле: stat() 315
15.2. Файловые метки времени 320
15.2.1. Изменение меток времени файла с помощью системных вызовов utime() и utimes() 323
15.2.2. Изменение меток времени файла с помощью системного вызова utimensat() и функции futimens() 325
15.3. Принадлежность файла 326
15.3.1. Принадлежность новых файлов 327
15.3.2. Изменение принадлежности файла: системные вызовы chown(), fchown() и lchown() 327
15.4. Права доступа к файлу 330
15.4.1. Права доступа к обычным файлам 330
15.4.2. Права доступа к каталогам 332
15.4.3. Алгоритм проверки прав доступа 333
15.4.4. Проверка доступности файла: системный вызов access() 335
15.4.5. Биты set-user-ID, set-group-ID и закрепляющий 336
15.4.6. Маска режима создания файла процесса: umask() 337
15.4.7. Изменение прав доступа к файлу: системные вызовы chmod() и fchmod() 339
15.5. Флаги индексного дескриптора (расширенные атрибуты файла в файловой системе ext2) 340
15.6. Резюме 344
15.7. Упражнения 344

Глава 16. Расширенные атрибуты 346

16.1. Обзор 346
16.2. Подробности реализации расширенных атрибутов 348
16.3. Системные вызовы для манипуляции расширенными атрибутами 349
16.4. Резюме 353
16.5. Упражнение 353

Глава 17. Списки контроля доступа 354

17.1. Обзор 354
17.2. Алгоритм проверки прав доступа с помощью списков контроля доступа 356
17.3. Длинная и краткая текстовые формы списков контроля доступа 357
17.4. Запись ACL_MASK и класс группы для ACL-списка 358
17.5. Команды getfacl и setfacl 359
17.6. ACL-списки по умолчанию и создание файла 361
17.7. Границы реализации списка контроля доступа 362
17.8. API для ACL-списков 363

17.9. Резюме 370
17.10. Упражнение 370

Глава 18. Каталоги и ссылки 371

18.1. Каталоги и (жесткие) ссылки 371
18.2. Символические (мягкие) ссылки 374
18.3. Создание и удаление (жестких) ссылок: системные вызовы link() и unlink() 377
18.4. Изменение имени файла: системный вызов rename() 380
18.5. Работа с символическими ссылками: системные вызовы symlink() и readlink() 381
18.6. Создание и удаление каталогов: системные вызовы mkdir() и rmdir() 382
18.7. Удаление файла или каталога: функция remove() 384
18.8. Чтение каталогов: функции opendir() и readdir() 384
18.9. Обход дерева файлов: функция nftw() 390
18.10. Текущий рабочий каталог процесса 394
18.11. Работа с использованием файлового дескриптора каталога 396
18.12. Изменение корневого каталога процесса: системный вызов chroot() 398
18.13. Анализ имени пути: функция realpath() 400
18.14. Синтаксический разбор строк с именем пути: функции dirname() и basename() 402
18.15. Резюме 404
18.16. Упражнения 404

Глава 19. Мониторинг событий файлов 406

19.1. Обзор 406
19.2. Интерфейс inotify 407
19.3. События inotify 409
19.4. Чтение событий inotify 410
19.5. Ограничения очереди и файлы /proc 416
19.6. Старая система мониторинга событий файлов: dnotify 416
19.7. Резюме 417
19.8. Упражнение 417

Глава 20. Сигналы: фундаментальные концепции 418

20.1. Концепции и общие сведения 418
20.2. Типы сигналов и действия по умолчанию 420
20.3. Изменение диспозиций сигналов: signal() 426
20.4. Введение в обработчики сигналов 427
20.5. Отправка сигналов: kill() 430
20.6. Проверка существования процесса 432
20.7. Другие способы отправки сигналов: raise() и killpg() 433
20.8. Отображение описаний сигналов 434
20.9. Наборы сигналов 435
20.10. Сигнальная маска (блокирование доставки сигналов) 438
20.11. Ожидающие сигналы 439
20.12. Сигналы не ставятся в очередь 440
20.13. Изменение диспозиций сигналов: sigaction() 443
20.14. Ожидание сигнала: pause() 445
20.15. Резюме 445
20.16. Упражнения 446

Глава 21. Сигналы: обработчики сигналов 447

21.1. Проектирование обработчиков сигналов 447
21.1.1. Сигналы не ставятся в очередь (еще раз о...) 447
21.1.2. Реентерабельные функции и функции, безопасные для асинхронных сигналов 448

21.1.3. Глобальные переменные и тип данных sig_atomic_t 453
21.2. Другие методы завершения работы обработчика сигнала 454
21.2.1. Выполнение нелокального перехода из обработчика сигнала 454
21.2.2. Аварийное завершение процесса: abort() 458
21.3. Обработка сигнала на альтернативном стеке: signalstack() 459
21.4. Флаг SA_SIGINFO 462
21.5. Прерывание и повторный запуск системных вызовов 467
21.6. Резюме 470
21.7. Упражнение 471

Глава 22. Сигналы: дополнительные возможности 472

22.1. Файлы дампа ядра 472
22.2. Частные случаи доставки, диспозиции и обработки 474
22.3. Прерываемые и непрерываемые состояния сна процесса 475
22.4. Аппаратно генерируемые сигналы 476
22.5. Синхронная и асинхронная генерация сигнала 477
22.6. Тайминг и порядок доставки сигнала 478
22.7. Реализация и переносимость функции signal() 479
22.8. Сигналы реального времени 481
22.8.1. Отправка сигналов реального времени 483
22.8.2. Обработка сигналов реального времени 485
22.9. Ожидание сигнала с использованием маски: sigsuspend() 488
22.10. Синхронное ожидание сигнала 492
22.11. Получение сигналов через файловый дескриптор 496
22.12. Межпроцессное взаимодействие посредством сигналов 498
22.13. Ранние API сигналов 499
22.14. Резюме 500
22.15. Упражнения 501

Глава 23. Таймеры и переход в режим сна 502

23.1. Интервальные таймеры 502
23.2. Планирование и точность таймеров 507
23.3. Установка времени ожидания для блокирующих операций 508
23.4. Приостановка выполнения на определенный отрезок времени (переход в режим сна) 509
23.4.1. Переход в режим сна (низкая точность): вызов sleep() 509
23.4.2. Переход в режим сна (высокая точность): вызов nanosleep() 510
23.5. Часы стандарта POSIX 512
23.5.1. Получение текущего значения часов: вызов clock_gettime() 513
23.5.2. Изменение значения часов: вызов clock_settime() 514
23.5.3. Получение идентификатора часов для определенного процесса или потока 514
23.5.4. Улучшенный переход в режим сна (высокая точность): вызов clock_nanosleep() 515
23.6. Интервальные таймеры POSIX 516
23.6.1. Создание таймера: вызов timer_create() 517
23.6.2. Запуск и остановка таймера: вызов timer_settime() 519
23.6.3. Получение текущего значения таймера: вызов timer_gettime() 520
23.6.4. Удаление таймера: вызов timer_delete() 520
23.6.5. Уведомление с помощью сигнала 521
23.6.6. Дополнительные срабатывания таймера 524
23.6.7. Уведомление с помощью потока 525
23.7. Таймеры, которые уведомляют с помощью файловых дескрипторов: интерфейс timerfd 528

23.8. Резюме 532
23.9. Упражнения 532

Глава 24. Создание процессов 533

24.1. Обзор вызовов fork(), exit(), wait() и execve() 533
24.2. Создание нового процесса: fork() 534
24.2.1. Совместный доступ к файлу родителя и потомка 537
24.2.2. Семантика памяти вызова fork() 540
24.3. Системный вызов vfork() 542
24.4. Состояние гонки после вызова fork() 544
24.5. Синхронизация с помощью сигналов как способ избежать состояния гонки 546
24.6. Резюме 548
24.7. Упражнения 548

Глава 25. Завершение работы процесса 549

25.1. Завершение процесса: вызовы _exit() и exit() 549
25.2. Завершение процесса в подробностях 550
25.3. Обработчики выхода 551
25.4. Взаимодействие между буферами stdio и вызовами fork() и _exit() 554
25.5. Резюме 556
25.6. Упражнение 556

Глава 26. Мониторинг дочерних процессов 557

26.1. Ожидание дочернего процесса 557
26.1.1. Системный вызов wait() 557
26.1.2. Системный вызов waitpid() 559
26.1.3. Статус ожидания 560
26.1.4. Завершение процесса из обработчика сигнала 564
26.1.5. Системный вызов waitid() 565
26.1.6. Системные вызовы wait3() и wait4() 567
26.2. Процессы-«сироты» и процессы-«зомби» 567
26.3. Сигнал SIGCHLD 569
26.3.1. Установка обработчика сигнала SIGCHLD 570
26.3.2. Доставка сигнала SIGCHLD для остановленных потомков 573
26.3.3. Игнорирование завершенных дочерних процессов 573
26.4. Резюме 574
26.5. Упражнения 575

Глава 27. Выполнение программы 576

27.1. Выполнение новой программы: execve() 576
27.2. Библиотечные функции семейства exec() 579
27.2.1. Переменная среды PATH 580
27.2.2. Задание аргументов программы в виде списка 582
27.2.3. Передача переменных среды вызывающего процесса новой программе 582
27.2.4. Выполнение файла через ссылку на его дескриптор: fexecve() 583
27.3. Интерпретируемые скрипты 583
27.4. Дескрипторы файлов и вызовы exec() 587
27.5. Сигналы и вызов exec() 589
27.6. Выполнение консольных команд: system() 590
27.7. Реализация функции system() 593
27.8. Резюме 599
27.9. Упражнения 599

Глава 28. Подробнее о создании процесса и выполнении программы 601
28.1. Учет ресурсов, используемых процессом 601
28.2. Системный вызов clone() 607
28.2.1. Аргумент flags вызова clone() 612
28.2.2. Расширения к вызову waitpid() для клонированных потомков 619
28.3. Скорость создания процессов 619
28.4. Влияние вызовов exec() и fork() на атрибуты процесса 621
28.5. Резюме 625
28.6. Упражнение 626

Глава 29. Потоки выполнения: введение 627
29.1. Краткий обзор 627
29.2. Общие сведения о программном интерфейсе Pthreads 630
29.3. Создание потоков 631
29.4. Завершение потоков 633
29.5. Идентификаторы потоков 633
29.6. Присоединение к завершенному потоку 635
29.7. Отсоединение потока 637
29.8. Атрибуты потоков 637
29.9. Сравнение потоков и процессов 638
29.10. Резюме 639
29.11. Упражнения 640

Глава 30. Потоки выполнения: синхронизация 641
30.1. Защита доступа к разделяемым переменным: мьютексы 641
30.1.1. Статически выделяемые мьютексы 645
30.1.2. Закрытие и открытие мьютекса 645
30.1.3. Производительность мьютексов 647
30.1.4. Взаимное блокирование мьютексов 648
30.1.5. Динамическая инициализация мьютексов 649
30.1.6. Атрибуты мьютексов 650
30.1.7. Типы мьютексов 650
30.2. Оповещение об изменении состояния: условные переменные 651
30.2.1. Статически выделяемые условные переменные 652
30.2.2. Оповещение и ожидание условных переменных 652
30.2.3. Проверка предиката условной переменной 656
30.2.4. Пример программы: подсоединение любого завершенного потока 657
30.2.5. Динамически выделяемые условные переменные 660
30.3. Резюме 661
30.4. Упражнения 661

Глава 31. Потоки выполнения: потоковая безопасность и локальное хранилище 662
31.1. Потоковая безопасность (и новый взгляд на реентерабельность) 662
31.2. Единовременная инициализация 665
31.3. Данные уровня потока 666
31.3.1. Данные уровня потока с точки зрения библиотечной функции 666
31.3.2. Обзор программного интерфейса для работы с данными уровня потока 667
31.3.3. Подробности о программном интерфейсе для работы с данными уровня потока 667
31.3.4. Использование программного интерфейса для работы с данными уровня потока 670
31.3.5. Ограничения реализации данных уровня потока 674

31.4. Локальное хранилище потока 674
31.5. Резюме 676
31.6. Упражнения 676

Глава 32. Потоки выполнения: отмена потока 677

32.1. Отмена потока 677
32.2. Состояние и тип отмены 677
32.3. Точки отмены 678
32.4. Проверка возможности отмены потока 681
32.5. Обработчики, освобождающие ресурсы 681
32.6. Асинхронная отмена 685
32.7. Резюме 685

Глава 33. Потоки выполнения: дальнейшие подробности 686

33.1. Стеки потоков 686
33.2. Потоки и сигналы 687
33.2.1. Как модель сигналов в UNIX соотносится с потоками 687
33.2.2. Изменение масок сигналов потока 688
33.2.3. Отправка сигнала потоку 689
33.2.4. Разумная обработка асинхронных сигналов 689
33.3. Потоки и управление процессами 690
33.4. Модели реализации потоков 692
33.5. Разные реализации POSIX-потоков в Linux 693
33.5.1. LinuxThreads 694
33.5.2. Библиотека NPTL 696
33.5.3. Выбор между разными реализациями многопоточности 698
33.6. Продвинутые возможности программного интерфейса Pthreads 700
33.7. Резюме 700
33.8. Упражнения 701

Глава 34. Группы процессов, сессии и управление заданиями 702

34.1. Краткий обзор 702
34.2. Группы процессов 703
34.3. Сессии 707
34.4. Контролирующие терминалы и контролирующие процессы 708
34.5. Активные и фоновые группы процессов 710
34.6. Сигнал SIGHUP 711
34.6.1. Обработка сигнала SIGHUP командной оболочкой 712
34.6.2. Сигнал SIGHUP и завершение контролирующего процесса 714
34.7. Управление заданиями 716
34.7.1. Управление заданиями в рамках командной оболочки 716
34.7.2. Реализация управления заданиями 718
34.7.3. Обрабатываем сигналы, связанные с управлением заданиями 723
34.7.4. Осиротевшие группы процессов (и новый взгляд на сигнал SIGHUP) 727
34.8. Резюме 731
34.9. Упражнения 732

Глава 35. Приоритеты процессов и их планирование 733

35.1. Приоритеты процессов (значение nice) 733
35.2. Обзор планирования в режиме реального времени 736
35.2.1. Политика SCHED_RR 738
35.2.2. Политика SCHED_FIFO 739
35.2.3. Политики SCHED_BATCH и SCHED_IDLE 739

35.3. Программный интерфейс планирования в режиме реального времени 739
35.3.1. Диапазон приоритетов реального времени 740
35.3.2. Изменение и получение политик и приоритетов 740
35.3.3. Освобождение ресурсов процессора 746
35.3.4. Временной отрезок в политике SCHED_RR 746
35.4. Привязка к процессору 747
35.5. Резюме 749
35.6. Упражнения 750

Глава 36. Ресурсы процессов 752

36.1. Ресурсы, использующиеся процессом 752
36.2. Ограничения на ресурсы для отдельных процессов 754
36.3. Подробности об отдельных ограничениях на ресурсы 759
36.4. Резюме 763
36.5. Упражнения 763

Глава 37. Демоны 764

37.1. Краткий обзор 764
37.2. Создание демона 764
37.3. Рекомендации по написанию демонов 768
37.4. Использование сигнала SIGHUP для повторной инициализации демона 769
37.5. Запись в журнал сообщений и ошибок с помощью системы syslog 771
37.5.1. Краткий обзор 772
37.5.2. Программный интерфейс syslog 773
37.5.3. Файл /etc/syslog.conf 778
37.6. Резюме 779
37.7. Упражнение 779

Глава 38. Написание безопасных программ с повышенными привилегиями 780

38.1. Нужно ли программе устанавливать идентификаторы пользователя или группы? 780
38.2. Работайте с минимальными привилегиями 781
38.3. Будьте осторожны при выполнении программы 783
38.4. Избегайте раскрытия деликатной информации 785
38.5. Изоляция процесса 785
38.6. Не забывайте о сигналах в состоянии гонки 786
38.7. Подводные камни, связанные с файловыми операциями и вводом/выводом файлов 787
38.8. Не доверяйте внешнему вводу или среде выполнения 788
38.9. Остерегайтесь переполнений буфера 789
38.10. Остерегайтесь DoS-атак 790
38.11. Проверяйте результаты выполнения и предусматривайте безопасное завершение в случае неудачи 791
38.12. Резюме 791
38.13. Упражнения 792

Глава 39. Система возможностей 793

39.1. Зачем нужна система возможностей 793
39.2. Система возможностей в Linux 794
39.3. Возможности, связанные с процессами и файлами 794
39.3.1. Возможности процесса 794
39.3.2. Возможности файлов 795
39.3.3. Назначение разрешенных и действующих возможностей процесса 798

39.3.4. Для чего нужны разрешенные и действующие возможности файла ... 799
39.3.5. Для чего нужны наследуемые возможности процесса и файла ... 799
39.3.6. Назначение и просмотр возможностей файла из командной оболочки ... 800
39.4. Современная реализация системы возможностей ... 801
39.5. Изменение возможностей процесса во время выполнения exec() ... 801
39.5.1. Ограничивающий набор возможностей ... 802
39.5.2. Сохранение традиционной для администратора семантики ... 802
39.6. Как изменение пользовательского идентификатора влияет на возможности процесса ... 803
39.7. Программное изменение возможностей процесса ... 803
39.8. Создание среды, в которой возможности являются единственным механизмом повышения привилегий ... 807
39.9. Определение возможностей, которые требуются программе ... 810
39.10. Резюме ... 810
39.11. Упражнение ... 810

Глава 40. Учет входа в систему ... 811

40.1. Краткий обзор файлов utmp и wtmp ... 811
40.2. Программный интерфейс utmpx ... 811
40.3. Структура utmpx ... 812
40.4. Извлечение информации из файлов utmp и wtmp ... 814
40.5. Получение имени текущего пользователя: getlogin() ... 817
40.6. Запись в файлы utmp и wtmp данных о пребывании в системе ... 818
40.7. Файл lastlog ... 822
40.8. Резюме ... 824
40.9. Упражнения ... 824

Глава 41. Основы разделяемых библиотек ... 825

41.1. Библиотека объектов ... 825
41.2. Статические библиотеки ... 826
41.3. Краткий обзор разделяемых библиотек ... 827
41.4. Создание и использование разделяемых библиотек. Первые шаги ... 828
41.4.1. Создание разделяемой библиотеки ... 829
41.4.2. Адресно-независимый код ... 829
41.4.3. Использование статической библиотеки ... 830
41.4.4. Разделяемые библиотеки и имя soname ... 832
41.5. Полезные инструменты для работы с разделяемыми библиотеками ... 834
41.6. Версии и соглашение об именовании разделяемых библиотек ... 835
41.7. Установка разделяемых библиотек ... 838
41.8. Совместимые и несовместимые библиотеки ... 841
41.9. Обновления разделяемых библиотек ... 841
41.10. Задание каталогов для поиска библиотеки в объектном файле ... 842
41.11. Поиск разделяемых библиотек на этапе выполнения ... 845
41.12. Разрешение символов на этапе выполнения ... 845
41.13. Использование статической библиотеки вместо динамической ... 847
41.14. Резюме ... 847
41.15. Упражнение ... 848

Глава 42. Продвинутые возможности разделяемых библиотек ... 849

42.1. Динамически загружаемые библиотеки ... 849
42.1.1. Открытие разделяемой библиотеки: dlopen() ... 850
42.1.2. Анализ ошибок: dlerror() ... 852
42.1.3. Получение адреса символа: dlsym() ... 852

42.1.4. Закрытие разделяемой библиотеки: dlclose() 855
42.1.5. Получение информации о загруженных символах: dladdr() 855
42.1.6. Получение доступа к символам главной программы 856
42.2. Управление видимостью символов 856
42.3. Версионные сценарии компоновщика 857
42.3.1. Управление видимостью символов с помощью версионных сценариев 858
42.3.2. Версионирование символов 859
42.4. Инициализация и финализация функций 861
42.5. Предварительная загрузка разделяемых библиотек 862
42.6. Мониторинг работы динамического компоновщика: LD_DEBUG 863
42.7. Резюме 864
42.8. Упражнения 865

Глава 43. Краткий обзор межпроцессного взаимодействия 866

43.1. Классификация IPC-механизмов 866
43.2. Средства взаимодействия 866
43.3. Средства синхронизации 869
43.4. Сравнение IPC-механизмов 870
43.5. Резюме 875
43.6. Упражнения 875

Глава 44. Каналы и очереди FIFO 876

44.1. Краткий обзор 876
44.2. Создание и использование каналов 878
44.3. Каналы как средство синхронизации процессов 883
44.4. Использование каналов для соединения фильтров 885
44.5. Взаимодействие с консольными командами с помощью канала: popen() 888
44.6. Каналы и буферизация стандартного ввода/вывода 891
44.7. Очереди FIFO 892
44.8. Клиент-серверные приложения на основе очередей FIFO 894
44.9. Неблокирующий ввод/вывод 901
44.10. Семантика вызовов read() и write() в контексте каналов и очередей FIFO 903
44.11. Резюме 904
44.12. Упражнения 905

Глава 45. Отображение в память 906

45.1. Краткий обзор 906
45.2. Создание отображения: mmap() 908
45.3. Удаление отображения с участка памяти: munmap() 911
45.4. Отображение файлов 912
45.4.1. Приватные файловые отображения 913
45.4.2. Разделяемые файловые отображения 914
45.4.3. Крайние случаи 917
45.4.4. Взаимодействие защиты памяти и режима доступа к памяти 919
45.5. Синхронизация отображенного участка памяти: msync() 919
45.6. Дополнительные флаги вызова mmap() 920
45.7. Анонимные отображения 922
45.8. Изменение отображенного участка памяти: mremap() 924
45.9. Флаг MAP_NORESERVE и перерасход пространства подкачки 925
45.10. Флаг MAP_FIXED 927
45.11. Нелинейные отображения: remap_file_pages() 928
45.12. Резюме 930
45.13. Упражнения 931

Глава 46. Операции с виртуальной памятью 932
46.1. Изменение защиты памяти: mprotect() 932
46.2. Блокирование памяти: mlock() и mlockall() 934
46.3. Определение местонахождения памяти: mincore() 937
46.4. Предсказание модели использования памяти в будущем: madvise() 940
46.5. Резюме 941
46.6. Упражнения 941

Глава 47. Введение в межпроцессное взаимодействие стандарта POSIX 942
47.1. Краткий обзор программных интерфейсов 942
47.2. Резюме 946

Глава 48. Очереди сообщений стандарта POSIX 947
48.1. Краткий обзор 947
48.2. Открытие, закрытие и удаление очереди сообщений 948
48.3. Связь между дескрипторами и очередями сообщений 950
48.4. Атрибуты очередей сообщений 951
48.5. Обмен сообщениями 956
48.5.1. Отправка сообщений 956
48.5.2. Получение сообщений 957
48.5.3. Отправка и получение сообщений с ограниченным временем ожидания 959
48.6. Оповещение о сообщении 960
48.6.1. Получение оповещения в виде сигнала 962
48.6.2. Получение уведомлений в отдельном потоке 964
48.7. Возможности, характерные для Linux 965
48.8. Ограничения, относящиеся к очередям сообщений 967
48.9. Резюме 968
48.10. Упражнения 968

Глава 49. Семафоры стандарта POSIX 969
49.1. Краткий обзор 969
49.2. Именованные семафоры 969
49.2.1. Открытие именованного семафора 970
49.2.2. Закрытие семафора 972
49.2.3. Удаление именованного семафора 972
49.3. Операции с семафорами 973
49.3.1. Декрементация семафора 973
49.3.2. Инкрементация семафора 975
49.3.3. Получение текущего значения семафора 976
49.4. Анонимные семафоры 977
49.4.1. Инициализация анонимного семафора 978
49.4.2. Уничтожение анонимного семафора 981
49.5. Сравнение POSIX-семафоров с мьютексами из библиотеки Pthreads 981
49.6. Ограничения, связанные с семафорами 982
49.7. Резюме 982
49.8. Упражнение 982

Глава 50. Разделяемая память POSIX 983
50.1. Краткий обзор 983
50.2. Создание объектов разделяемой памяти 984
50.3. Использование объектов разделяемой памяти 986
50.4. Удаление объектов разделяемой памяти 989
50.5. Сравнение программных интерфейсов для работы с разделяемой памятью 989
50.6. Резюме 990

Глава 51. Блокировка файлов 991
51.1. Краткий обзор 991
51.2. Блокировка файла с помощью вызова flock() 993
51.2.1. Семантика наследования и снятия блокировок 996
51.2.2. Ограничения вызова flock() 997
51.3. Блокировка записей с помощью вызова fcntl() 997
51.3.1. Структура flock 999
51.3.2. Аргумент cmd 1000
51.3.3. Подробности об установке и снятии блокировок 1001
51.3.4. Взаимная блокировка 1002
51.3.5. Пример: программа для интерактивной блокировки 1002
51.3.6. Пример: библиотека функций для установки блокировок 1006
51.3.7. Производительность блокировок и их ограничения 1008
51.3.8. Семантика наследования и снятия блокировок 1009
51.3.9. Зависание блокировок и приоритет отложенных запросов на их получение 1010
51.4. Строгая блокировка 1011
51.5. Файл /proc/locks 1014
51.6. Выполнение только одного экземпляра программы 1015
51.7. Устаревшие способы блокировки 1017
51.8. Резюме 1019
51.9. Упражнения 1019

Глава 52. Сокеты: введение 1021
52.1. Краткий обзор 1021
52.2. Создание сокета: socket() 1024
52.3. Привязывание сокета к адресу: bind() 1025
52.4. Универсальные структуры для хранения адресов сокетов: struct sockaddr 1025
52.5. Потоковые сокеты 1026
52.5.1. Ожидание входящих соединений: listen() 1028
52.5.2. Прием соединения: accept() 1029
52.5.3. Соединение с удаленным сокетом: connect() 1029
52.5.4. Операции ввода/вывода с потоковыми сокетами 1030
52.5.5. Закрытие соединения: close() 1030
52.6. Датаграммные сокеты 1031
52.6.1. Обмен датаграммами: recvfrom() и sendto() 1031
52.6.2. Использование вызова connect() в сочетании с датаграммными сокетами 1033
52.7. Резюме 1033

Глава 53. Сокеты: домен UNIX 1035
53.1. Адреса сокетов в домене UNIX: struct sockaddr_un 1035
53.2. Потоковые сокеты в домене UNIX 1037
53.3. Датаграммные сокеты в домене UNIX 1040
53.4. Права доступа к сокетам домена UNIX 1043
53.5. Создание соединенной пары сокетов: socketpair() 1043
53.6. Абстрактное пространство имен сокетов в Linux 1044
53.7. Резюме 1045
53.8. Упражнения 1045

Глава 54. Сокеты: основы сетей TCP/IP 1046
54.1. Интерсети 1046
54.2. Сетевые протоколы и уровни 1047
54.3. Канальный уровень 1050

54.4. Сетевой уровень: IP ... 1050
54.5. IP-адреса ... 1052
54.6. Транспортный уровень ... 1054
54.6.1. Номера портов ... 1055
54.6.2. Протокол пользовательских датаграмм (UDP) ... 1056
54.6.3. Протокол управления передачей (TCP) ... 1056
54.7. Документы, выносимые на рассмотрение (RFC) ... 1060
54.8. Резюме ... 1060

Глава 55. Сокеты: домены сети Интернет ... 1062

55.1. Сокеты интернет-домена ... 1062
55.2. Порядок байтов в сети ... 1062
55.3. Представление данных ... 1064
55.4. Адреса интернет-сокетов ... 1066
55.5. Краткий обзор функций для преобразования сетевых адресов и имен служб ... 1068
55.6. Функции inet_pton() и inet_ntop() ... 1070
55.7. Пример клиент-серверного приложения (на основе датаграммных сокетов) ... 1071
55.8. Система доменных имен (DNS) ... 1073
55.9. Файл /etc/services ... 1076
55.10. Преобразование имен узлов и служб, не зависящее от протокола ... 1077
55.10.1. Функция getaddrinfo() ... 1077
55.10.2. Удаление списков со структурами addrinfo: freeaddrinfo() ... 1080
55.10.3. Выявление ошибок: gai_strerror() ... 1080
55.10.4. Функция getnameinfo() ... 1081
55.11. Пример клиент-серверного приложения на основе потоковых сокетов ... 1082
55.12. Библиотека для работы с сокетами интернет-домена ... 1088
55.13. Сравнение сокетов в UNIX- и интернет-доменах ... 1092
55.14. Дополнительная информация ... 1093
55.15. Резюме ... 1093
55.16. Упражнения ... 1094

Глава 56. Сокеты: архитектура сервера ... 1095

56.1. Итерационные и параллельные серверы ... 1095
56.2. Итерационный UDP-сервер echo ... 1095
56.3. Параллельный TCP-сервер echo ... 1098
56.4. Другие разновидности архитектуры параллельного сервера ... 1100
56.5. Демон inetd ... 1102
56.6. Резюме ... 1106
56.7. Упражнения ... 1107

Глава 57. Сокеты: углубленный материал ... 1108

57.1. Частичное чтение и запись в контексте потоковых сокетов ... 1108
57.2. Системный вызов shutdown() ... 1110
57.3. Специальные системные вызовы для работы с сокетами: recv() и send() ... 1113
57.4. Системный вызов sendfile() ... 1114
57.5. Получение адреса сокета ... 1117
57.6. Подробности реализации протокола TCP ... 1119
57.6.1. Формат TCP-сегментов ... 1119
57.6.2. Порядковые номера и подтверждения в протоколе TCP ... 1121
57.6.3. Машина состояний и диаграмма перехода состояний в протоколе TCP ... 1122
57.6.4. Установка TCP-соединения ... 1124
57.6.5. Разрыв TCP-соединения ... 1125

57.6.6. Вызов shutdown() для TCP-сокета ... 1126
57.6.7. Состояние TIME_WAIT ... 1127
57.7. Мониторинг сокетов: утилита netstat ... 1128
57.8. Мониторинг данных, проходящих по протоколу TCP, с помощью утилиты tcpdump ... 1130
57.9. Параметры сокета ... 1131
57.10. Параметр сокета SO_REUSEADDR ... 1132
57.11. Наследование флагов и параметров сокета при выполнении вызова accept() ... 1134
57.12. Выбор между TCP и UDP ... 1135
57.13. Продвинутые возможности ... 1136
57.13.1. Внеканальные данные ... 1136
57.13.2. Системные вызовы sendmsg() и recvmsg() ... 1136
57.13.3. Передача файловых дескрипторов ... 1137
57.13.4. Получение учетных данных отправителя ... 1137
57.13.5. Последовательный обмен пакетами ... 1137
57.13.6. Протоколы транспортного уровня SCTP и DCCP ... 1138
57.14. Резюме ... 1139
57.15. Упражнения ... 1140

Глава 58. Терминалы ... 1142

58.1. Краткий обзор ... 1143
58.2. Извлечение и изменение атрибутов терминала ... 1143
58.3. Команда stty ... 1146
58.4. Специальные символы терминала ... 1148
58.5. Флаги терминала ... 1153
58.6. Режимы ввода/вывода терминала ... 1159
58.6.1. Канонический режим ... 1159
58.6.2. Неканонический режим ... 1159
58.6.3. Режимы с обработкой, без обработки и cbreak ... 1161
58.7. Скорость передачи данных в терминале ... 1167
58.8. Управление последовательным портом ... 1169
58.9. Размер окна терминала ... 1170
58.10. Идентификация терминала ... 1172
58.11. Резюме ... 1172
58.12. Упражнения ... 1173

Глава 59. Альтернативные модели ввода/вывода ... 1174

59.1. Краткий обзор ... 1174
59.1.1. Уведомления, срабатывающие по уровню или фронту ... 1177
59.1.2. Применение неблокирующего режима в сочетании с альтернативными моделями ввода/вывода ... 1178
59.2. Мультиплексирование ввода/вывода ... 1179
59.2.1. Системный вызов select() ... 1179
59.2.2. Системный вызов poll() ... 1185
59.2.3. Условия готовности файлового дескриптора ... 1189
59.2.4. Сравнение вызовов select() и poll() ... 1192
59.2.5. Проблемы, присущие вызовам select() и poll() ... 1193
59.3. Ввод/вывод на основе сигналов ... 1194
59.3.1. Установка владельца файлового дескриптора ... 1196
59.3.2. Когда генерируется сигнал о возможности ввода/вывода? ... 1198
59.3.3. Эффективное использование ввода/вывода на основе сигналов ... 1199
59.4. Программный интерфейс epoll ... 1201

59.4.1. Создание экземпляра epoll: вызов epoll_create() ... 1202
59.4.2. Редактирование списка интереса epoll: вызов epoll_ctl() ... 1203
59.4.3. Ожидание событий: вызов epoll_wait() ... 1204
59.4.4. Подробности семантики интерфейса epoll ... 1209
59.4.5. Производительность интерфейса epoll по сравнению с мультиплексированным вводом/выводом ... 1211
59.4.6. Уведомления, срабатывающие по фронту ... 1212
59.5. Ожидание сигналов и готовности файловых дескрипторов ... 1213
59.5.1. Системный вызов pselect() ... 1214
59.5.2. Трюк с зацикленным каналом ... 1216
59.6. Резюме ... 1218
59.7. Упражнения ... 1220

Глава 60. Псевдотерминалы ... 1221

60.1. Краткий обзор ... 1221
60.2. Псевдотерминалы стандарта UNIX 98 ... 1225
60.2.1. Открытие неиспользуемого первичного устройства: вызов posix_openpt() ... 1225
60.2.2. Изменение владельца и прав доступа к вторичному устройству: вызов grantpt() ... 1226
60.2.3. Разблокировка вторичного устройства: вызов unlockpt() ... 1227
60.2.4. Получение имени вторичного устройства: вызов ptsname() ... 1227
60.3. Открытие первичного устройства: вызов ptyMasterOpen() ... 1228
60.4. Соединение процессов с помощью псевдотерминала: вызов ptyFork() ... 1229
60.5. Ввод/вывод псевдотерминала ... 1232
60.6. Реализация программы script(1) ... 1234
60.7. Атрибуты терминала и размер окна ... 1238
60.8. Резюме ... 1239
60.9. Упражнения ... 1239

Список используемых источников ... 1241