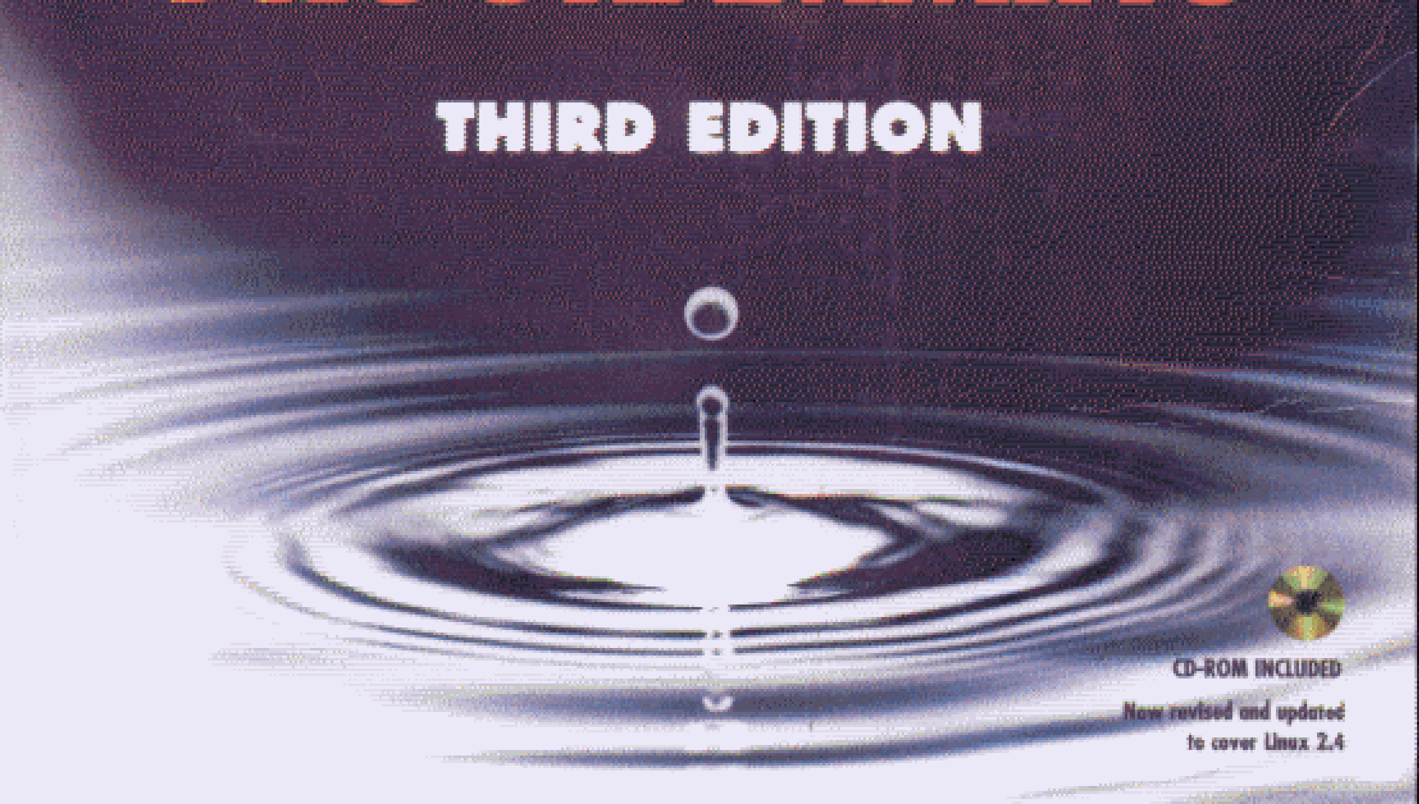
M Beck, H Böhme, M Dziadzka,
U Kunitz, R Magnus, C Schröter, D Verworner

LINUX KERNEL PROGRAMMING

THIRD EDITION



CD-ROM INCLUDED

Now revised and updated to cover Linux 2.4

CONTENTS

PREFACE xi

Preface to the 3rd edition xi
Preface to the 1st edition xii
Linus Torvalds' preface to the 1st edition xiii
Acknowledgments xiv

CHAPTER 1 LINUX – THE OPERATING SYSTEM 1

1.1 Main characteristics 2
1.2 LINUX distributions 5

CHAPTER 2 COMPILING THE KERNEL 6

2.1 Where is everything? 6
2.2 Compiling 8
2.3 Additional configuration facilities 10

CHAPTER 3 INTRODUCTION TO THE KERNEL [illegible]

3.1 Important data structures 16
- 3.1.1 The task structure 16
- 3.1.2 The process table 24
- 3.1.3 Files and inodes 25
- 3.1.4 Dynamic memory management 27
- 3.1.5 Queues and semaphores 28
- 3.1.6 System time and timers 30

3.2 Main algorithms 31
- 3.2.1 Signals 31
- 3.2.2 Hardware interrupts 32
- 3.2.3 Software interrupts 33
- 3.2.4 Booting the system 34
- 3.2.5 Timer interrupts 36
- 3.2.6 The scheduler 39

3.3 Implementation of system calls 42
- 3.3.1 How do system calls actually work? 42
- 3.3.2 Examples of simple system calls 44
- 3.3.3 Examples of more complex system calls 45

CHAPTER 4 MEMORY MANAGEMENT [illegible]

4.1 The architecture-independent memory model 57
- 4.1.1 Pages of memory 57
- 4.1.2 Virtual address space 57
- 4.1.3 Converting the linear address 59
- 4.1.4 Page directories 61
- 4.1.5 The page table 63

4.2 The virtual address space of a process 65
- 4.2.1 The user segment 65
- 4.2.2 Virtual memory areas 67
- 4.2.3 The system call *brk* 70
- 4.2.4 Mapping functions 71
- 4.2.5 The kernel segment 71
- 4.2.6 Memory allocation in the kernel segment during booting 72
- 4.2.7 Dynamic memory management in the kernel segment 72

4.3 Block device caching 75
- 4.3.1 Block buffering 75
- 4.3.2 `Bdflush` and `kupdate` 77
- 4.3.3 List structures for the buffer cache 78
- 4.3.4 Using the buffer cache 79

4.4 Paging under LINUX 80
- 4.4.1 Memory management and the memory cache 82
- 4.4.2 Reserving a page of memory 84
- 4.4.3 Optimization of memory page management via kernel threads 87
- 4.4.4 Page errors and reloading a page 88

CHAPTER 5 INTERPROCESS COMMUNICATION 90

5.1 Synchronization in the kernel 92

5.2 Communication via files 97
- 5.2.1 Locking entire files 98
- 5.2.2 Locking file areas 99

5.3 Pipes 103

5.4 Debugging using *ptrace* 105

5.5 System V IPC 108
- 5.5.1 Access rights, numbers, and keys 109
- 5.5.2 Semaphores 110
- 5.5.3 Message queues 114
- 5.5.4 Shared memory 117
- 5.5.5 The `ipcs` and `ipcrm` commands 120

5.6 IPC with sockets 121
5.6.1 A simple example 122
5.6.2 The implemetation of UNIX domain sockets 126

CHAPTER 6 **THE UNIX FILE SYSTEM 129**

6.1 Basic principles 130
6.2 The representation of file systems in the kernel 132
6.2.1 Mounting 134
6.2.2 The superblock 134
6.2.3 Superblock operations 136
6.2.4 The directory cache 139
6.2.5 DEntry operations 141
6.2.6 The inode 142
6.2.7 Inode operations 144
6.2.8 The file structure 147
6.2.9 File operations 148
6.2.10 Opening a file 152
6.3 The *Ext2* file system 156
6.3.1 The structure of the *Ext2* file system 156
6.3.2 Directories in the *Ext2* file system 159
6.3.3 Block allocation in the *Ext2* file system 160
6.3.4 Extensions of the *Ext2* file system 161
6.4 The *Proc* file system 162
6.4.1 Structures of the *Proc* file system 163
6.4.2 Implementation of the *Proc* file system 164

CHAPTER 7 **DEVICE DRIVERS UNDER LINUX 169**

7.1 Character and block devices 171
7.2 Hardware 172
7.2.1 Port I/O 172
7.2.2 The PCI bus 173
7.2.3 The dinosaur – the ISA bus 184
7.2.4 ISA-PnP 193
7.3 Polling, interrupts, and waiting queues 198
7.3.1 Polling 199
7.3.2 Interrupt mode 200
7.3.3 Interrupt sharing 201
7.3.4 Software interrupts 202
7.3.5 Bottom halves – the lower interrupt halves 203
7.3.6 Task queues 204
7.3.7 Timers 206

7.4 Implementing a driver 207

7.4.1 Example of a PC loudspeaker driver 207

7.4.2 A simple driver 210

7.4.3 The `setup` function 212

7.4.4 `Init` 213

7.4.5 Open and `release` 215

7.4.6 Read and `write` 216

7.4.7 `IOCTL` 219

7.4.8 `Poll` 221

7.4.9 `Llseek` 223

7.4.10 `Mmap` 223

7.4.11 `Fasync` 225

7.4.12 `Readdir`, `Fsync` 228

7.5 Dynamic and static drivers 228

CHAPTER 8 NETWORK IMPLEMENTATION 231

8.1 Introduction and overview 231

8.1.1 The layer model of the network implementation 232

8.1.2 Getting the data from A to B 232

8.2 Important structures 236

8.2.1 The `socket` structure 236

8.2.2 The `sk_buff` structure – buffer management in the network 238

8.2.3 The INET socket – a special part of the socket 241

8.2.4 Protocol operations in the `proto` structure 246

8.2.5 The general structure of a socket address 248

8.3 Network devices under LINUX 248

8.3.1 Ethernet 255

8.3.2 SLIP and PLIP 256

8.3.3 The loopback device 257

8.3.4 The dummy device 257

8.3.5 An example device 257

CHAPTER 9 MODULES AND DEBUGGING 260

9.1 What are modules? 261

9.2 Implementation in the kernel 262

9.2 1 Signatures of symbols 264

9.3 The meaning of object sections for modules and kernels 264

9.4 Parameter transfer and examples 267

9.5 What can be implemented as a module? 268

9.6 The kernel daemon 270

9.7 Simple data swapping between modules 270
9.8 An example module 271
9.9 Debugging 272
9.9.1 Changes are the beginning of the end 272
9.9.2 The best debugger – `printk()` 273
9.9.3 Debugging with gdb 274

CHAPTER 10 **MULTIPROCESSING 276**

10.1 The Intel multiprocessor specification 276
10.2 Problems with multiprocessor systems 277
10.3 Changes to the kernel 278
10.3.1 Kernel initialization 278
10.3.2 Scheduling 279
10.3.3 Interrupt handling 279
10.4 Atomic operations 279
10.4.1 The atomic data type 280
10.4.2 Accessing the atomic data type 280
10.4.3 Changing and testing atomic variables 280
10.5 Spin locks 281
10.5.1 Access functions 281
10.5.2 Read-write spin locks 282

APPENDIX A **SYSTEM CALLS 283**

A.1 Process management 284
A.2 The file system 329
A.3 Communication 362
A.4 Memory management 365
A.5 All that remains 371

APPENDIX B **KERNEL-RELATED COMMANDS 373**

B.1 `free` – overview of the system's memory 373
B.2 `ps` – output of process statistics 374
B.3 `top` – the CPU charts 378
B.4 `init` – primus inter pares 380
B.5 `shutdown` – shutting down the system 388
B.6 `strace` – monitoring a process 389
B.7 Configuring the network interface 392
B.8 `traceroute` – Ariadne's paths in the internet 393
B.9 Configuring a serial interface 395

B.10 Configuring a parallel interface 397
B.11 Building a directory tree 398

APPENDIX C **THE *PROC* FILE SYSTEM 405**

C.1 The /PROC/ directory 406
C.2 The NET/ directory 414
C.3 The SELF/ directory 416
C.4 The SYS/ directory 420

APPENDIX D **THE BOOTING PROCESS 422**

D.1 Carrying out the booting process 423
D.2 LILO – the LINUX loader 425

APPENDIX E **USEFUL KERNEL FUNCTIONS 434**

REFERENCES 449

INDEX 455