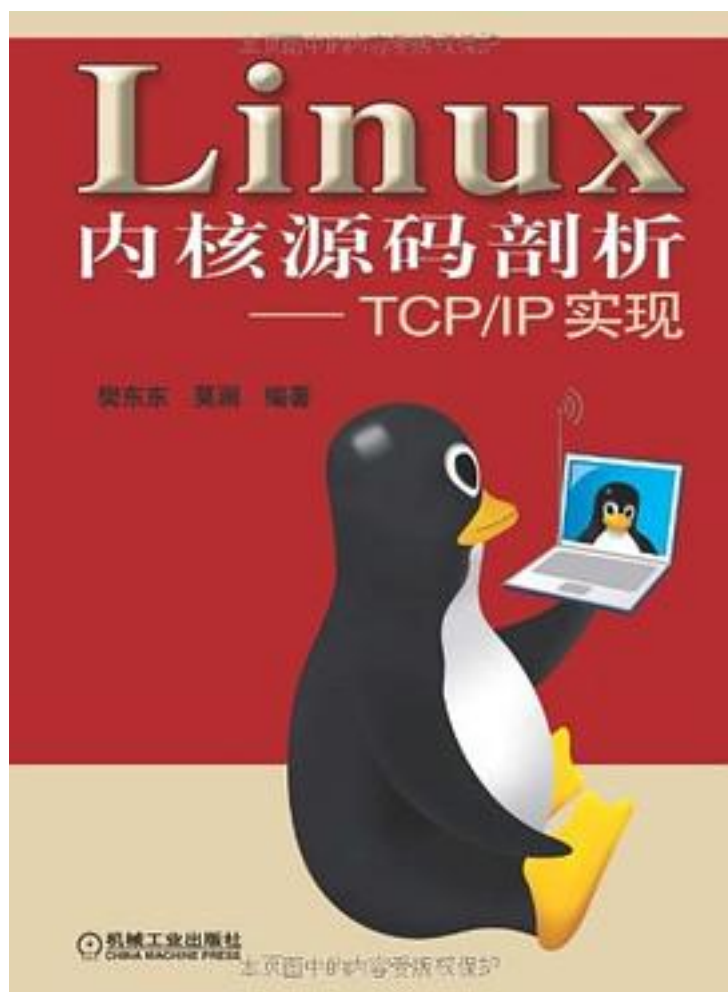


# Linux内核源码剖析（套装上下册）



[Linux内核源码剖析（套装上下册）\\_下载链接1](#)

著者:樊东东

出版者:机械工业出版社

出版时间:2011-1

装帧:平装

isbn:9787111323730

《Linux内核源码剖析:TCP/IP实现(套装上下册)》详细论述了Linux内核2.6.20版本中TCP/IP的实现。书中给出了大量的源代码，通过对源代码的详细注释，帮助读者掌握TCP

／IP的实现。《Linux内核源码剖析:TCP/IP实现(套装上下册)》根据协议栈层次，从驱动层逐步论述到传输层，包括驱动的实现、接口层的输入输出、IP层的输入输出以及IP选项的处理、邻居子系统、路由、套接口及传输层等内容，全书基本涵盖了网络体系架构全部的知识点。特别是TCP，包括TCP连接的建立和终止、输入与输出，以及拥塞控制的实现。

《Linux内核源码剖析:TCP/IP实现(套装上下册)》适用于熟悉Linux的基本使用方法，对Linux内核工作原理以及网络知识有一定的了解，而又极想更深入理解各个机制在Linux中的具体实现的用户，包括应用程序员和嵌入式程序员，以及网络管理员等。相关专业的科研人员在工作中遇到问题时，也可以查阅《Linux内核源码剖析:TCP/IP实现(套装上下册)》，理解相关内核部分的实现。此外，计算机相关专业的本科高年级学生和研究生，在学习相关课程（如操作系统、计算机网络等）时，可将《Linux内核源码剖析:TCP/IP实现(套装上下册)》作为辅助教程，与理论相结合以便更好地理解相应的知识点。

作者介绍:

目录:上册目录

前言

第1章 预备知识 1

1.1 应用层配置诊断工具 2

1.1.1 iputils 2

1.1.2 net-tools 2

1.1.3 iproute2 2

1.2 内核空间与用户空间的接口 2

1.2.1 procfs 2

1.2.2 sysctl(/proc/sys目录) 4

1.2.3 sysfs(/sys文件系统) 5

1.2.4 ioctl系统调用 6

1.2.5 netlink套接口 6

1.3 网络I/O加速 6

1.3.1 TSO/GSO 7

1.3.2 I/O AT 8

1.4 其他 8

1.4.1 slab分配器 9

1.4.2 RCU 9

第2章 网络体系结构概述 10

2.1 引言 10

2.2 协议简介 10

2.3 网络架构 11

2.4 系统调用接口 11

2.5 协议无关接口 12

2.6 传输层协议 12

2.7 套接口缓存 13

2.8 设备无关接口 14

2.9 设备驱动程序 14

2.10 网络模块源代码组织 14

第3章 套接口缓存 15

3.1 引言 15

3.2 sk\_buff结构 15

3.2.1 网络参数和内核数据结构 16

3.2.2 SKB组织相关的变量 19

- 3.2.3 数据存储相关的变量 20
- 3.2.4 通用的成员变量 21
- 3.2.5 标志性变量 24
- 3.2.6 特性相关的成员变量 25
- 3.3 skb\_shared\_info结构 25
  - 3.3.1 "零拷贝"技术 25
  - 3.3.2 对聚合分散I/O数据的支持 27
  - 3.3.3 对GSO的支持 30
  - 3.3.4 访问skb\_shared\_info结构 31
- 3.4 管理函数 31
  - 3.4.1 SKB的缓存池 31
  - 3.4.2 分配SKB 32
  - 3.4.3 释放SKB 34
  - 3.4.4 数据预留和对齐 36
  - 3.4.5 克隆和复制SKB 38
  - 3.4.6 链表管理函数 42
  - 3.4.7 添加或删除尾部数据 42
  - 3.4.8 拆分数据: skb\_split() 44
  - 3.4.9 重新分配SKB的线性数据区: pskb\_expand\_head() 46
  - 3.4.10 其他函数 46
- 第4章 网络模块初始化 48
  - 4.1 引言 48
  - 4.2 网络模块初始化顺序 48
  - 4.3 优化基于宏的标记 49
  - 4.4 网络设备处理层初始化 52
- 第5章 网络设备 55
  - 5.1 PCI设备 55
    - 5.1.1 PCI驱动程序相关结构 55
    - 5.1.2 注册PCI驱动程序 57
  - 5.2 与网络设备有关的数据结构 59
    - 5.2.1 net\_device结构 59
    - 5.2.2 网络设备有关结构的组织 71
    - 5.2.3 相关函数 72
  - 5.3 网络设备的注册 73
    - 5.3.1 设备注册的时机 73
    - 5.3.2 分配net\_device结构空间 73
    - 5.3.3 网络设备注册过程 75
    - 5.3.4 注册设备的状态迁移 79
    - 5.3.5 设备注册状态通知 79
    - 5.3.6 引用计数 80
  - 5.4 网络设备的注销 80
    - 5.4.1 设备注销的时机 80
    - 5.4.2 网络设备注销过程 81
  - 5.5 网络设备的启用 86
  - 5.6 网络设备的禁用 88
  - 5.7 与电源管理交互 89
    - 5.7.1 挂起设备 90
    - 5.7.2 唤醒设备 90
  - 5.8 侦测连接状态改变 91
    - 5.8.1 调度处理连接状态改变事件 91
    - 5.8.2 linkwatch标志 95
  - 5.9 从用户空间配置设备相关信息 95

- 5.9.1 ethtool 95
- 5.9.2 媒体独立接口 97
- 5.10 虚拟网络设备 97
- 第6章 IP编址 99
- 6.1 接口和IP地址 99
  - 6.1.1 主IP地址、从属IP地址和IP别名 99
  - 6.1.2 IP地址的组织 99
  - 6.1.3 in\_device结构 100
  - 6.1.4 in\_ifaddr结构 101
- 6.2 函数 102
  - 6.2.1 inetdev\_init() 102
  - 6.2.2 inetdev\_destroy() 104
  - 6.2.3 inet\_select\_addr() 104
  - 6.2.4 inet\_confirm\_addr() 106
  - 6.2.5 inet\_addr\_onlink() 107
  - 6.2.6 inetdev\_by\_index() 107
  - 6.2.7 inet\_ifa\_byprefix() 108
  - 6.2.8 inet\_abc\_len() 108
- 6.3 IP地址的设置 109
  - 6.3.1 netlink接口 109
  - 6.3.2 inet\_insert\_ifa() 111
  - 6.3.3 inet\_del\_ifa() 112
- 6.4 ioctl 115
- 6.5 inetaddr\_chain通知链 121
- 第7章 接口层的输入 122
- 7.1 系统参数 122
- 7.2 接口层的ioctl 123
  - 7.2.1 SIOCxIFxxx类命令 123
  - 7.2.2 SIOCETHTOOL 126
  - 7.2.3 私有命令 127
- 7.3 初始化 127
- 7.4 softnet\_data结构 128
- 7.5 NAPI方式 130
  - 7.5.1 网络设备中断例程 131
  - 7.5.2 网络输入软中断 131
  - 7.5.3 轮询处理 133
- 7.6 非NAPI方式 134
- 7.7 接口层输入报文的处理 137
  - 7.7.1 报文接收例程 137
  - 7.7.2 netif\_receive\_skb() 138
  - 7.7.3 dev\_queue\_xmit\_nit() 141
- 7.8 响应CPU状态的变化 142
- 7.9 netpoll 143
  - 7.9.1 netpoll相关结构 143
  - 7.9.2 注册netpoll实例 145
  - 7.9.3 netpoll的输入 148
  - 7.9.4 netpoll的输出 156
  - 7.9.5 tx\_work工作队列 159
  - 7.9.6 netpoll实例：netconsole 160
- 第8章 接口层的输出 163
- 8.1 输出接口 163
  - 8.1.1 dev\_queue\_xmit() 163
  - 8.1.2 dev\_hard\_start\_xmit() 167

- 8.1.3 e100的输出接口:
  - e100\_xmit\_frame() 168
- 8.2 网络输出软中断 168
  - 8.2.1 netif\_schedule() 168
  - 8.2.2 net\_tx\_action() 169
- 8.3 网络设备不支持GSO时的处理 170
  - 8.3.1 dev\_gso\_cb私有控制块 171
  - 8.3.2 dev\_gso\_segment() 171
  - 8.3.3 skb\_gso\_segment() 172
- 第9章 流量控制 174
  - 9.1 通过流量控制后输出 174
    - 9.1.1 dev\_queue\_xmit() 175
    - 9.1.2 qdisc\_restart() 176
  - 9.2 构成流量控制的三种元素 178
    - 9.2.1 排队规则 179
    - 9.2.2 类 186
    - 9.2.3 过滤器 189
  - 9.3 默认的FIFO排队规则 192
    - 9.3.1 pfifo\_fast\_init() 194
    - 9.3.2 pfifo\_fast\_reset() 194
    - 9.3.3 pfifo\_fast\_enqueue() 194
    - 9.3.4 pfifo\_fast\_dequeue() 195
    - 9.3.5 pfifo\_fast\_requeue() 195
  - 9.4 netlink的tc接口 195
  - 9.5 排队规则的创建接口 197
    - 9.5.1 类的创建接口 201
    - 9.5.2 过滤器的创建接口 204
- 第10章 Internet协议族 209
  - 10.1 net\_proto\_family结构 209
  - 10.2 inet\_protosw结构 210
  - 10.3 net\_protocol结构 212
  - 10.4 Internet协议族的初始化 214
- 第11章 IP: 网际协议 217
  - 11.1 引言 217
    - 11.1.1 IP首部 218
    - 11.1.2 IP数据报的输入与输出 219
  - 11.2 IP的私有信息控制块 220
  - 11.3 系统参数 220
  - 11.4 初始化 223
  - 11.5 IP层套接口选项 223
  - 11.6 ipv4\_devconf结构 227
  - 11.7 套接口的错误队列 229
    - 11.7.1 添加ICMP差错信息 231
    - 11.7.2 添加由本地产生的差错信息 232
    - 11.7.3 读取错误信息 233
  - 11.8 报文控制信息 235
    - 11.8.1 IP控制信息块 235
    - 11.8.2 报文控制信息的输出 235
    - 11.8.3 报文控制信息的输入 236
  - 11.9 对端信息块 237
    - 11.9.1 系统参数 239
    - 11.9.2 对端信息块的创建和查找 239
    - 11.9.3 对端信息块的删除 241

- 11.9.4 垃圾回收 242
- 11.10 IP数据报的输入处理 244
  - 11.10.1 IP数据报输入到本地 247
  - 11.10.2 IP数据报的转发 249
- 11.11 IP数据报的输出处理 253
  - 11.11.1 IP数据报输出到设备 253
  - 11.11.2 TCP输出的接口 255
  - 11.11.3 UDP输出的接口 261
- 11.12 IP层对GSO的支持 275
  - 11.12.1 inet\_gso\_segment() 275
  - 11.12.2 inet\_gso\_send\_check() 277
- 第12章 IP选项处理 278
  - 12.1 IP选项 278
    - 12.1.1 选项列表的结束符 279
    - 12.1.2 空操作 279
    - 12.1.3 安全选项 279
    - 12.1.4 严格源路由选项 280
    - 12.1.5 宽松源路由选项 281
    - 12.1.6 记录路由选项 282
    - 12.1.7 流标识选项 282
    - 12.1.8 时间戳选项 283
    - 12.1.9 路由器警告选项 283
  - 12.2 ip\_options结构 284
  - 12.3 在IP数据报中构建IP选项 285
  - 12.4 复制IP数据报中选项到指定的ip\_options结构 286
  - 12.5 处理待发送IP分片中的选项 290
  - 12.6 解析IP选项 291
  - 12.7 还原在校验IP选项时修改的IP选项 297
  - 12.8 处理转发IP数据报中的IP选项 298
  - 12.9 处理IP数据报的源路由选项 299
  - 12.10 解析并处理IP首部中的IP选项 300
  - 12.11 路由警告选项的处理 301
  - 12.12 由控制信息生成IP选项信息块 302
- 第13章 IP的分片与组装 303
  - 13.1 系统参数 303
  - 13.2 分片 303
    - 13.2.1 快速分片 306
    - 13.2.2 慢速分片 309
  - 13.3 组装 312
    - 13.3.1 ipq结构 312
    - 13.3.2 ipq散列表和链表的维护 315
    - 13.3.3 ipq散列表的重组 316
    - 13.3.4 超时IP分片的清除 317
    - 13.3.5 垃圾收集 318
    - 13.3.6 相关分片组装函数 319
    - 13.3.7 分片组装 327
- 第14章 ICMP: Internet控制

- 报文协议 330
  - 14.1 ICMP报文结构 330
  - 14.2 注册ICMP报文类型 330
  - 14.3 系统参数 330
  - 14.4 ICMP的初始化 332
  - 14.5 输入处理 333
    - 14.5.1 差错处理 337
    - 14.5.2 重定向处理 342
    - 14.5.3 请求回显 343
    - 14.5.4 时间戳请求 345
    - 14.5.5 地址掩码请求和应答 346
  - 14.6 输出处理 346
    - 14.6.1 发送ICMP报文 346
    - 14.6.2 发送回显应答和时间戳
- 应答报文 350
- 第15章 IP组播 353
  - 15.1 初始化 353
  - 15.2 虚拟接口 354
    - 15.2.1 虚拟接口的添加 355
    - 15.2.2 虚拟接口的删除：  
vif\_delete() 358
    - 15.2.3 查找虚拟接口：ipmr\_find\_vif() 358
  - 15.3 组播转发缓存 358
    - 15.3.1 组播转发缓存的创建 361
    - 15.3.2 组播转发缓存的删除 361
    - 15.3.3 组播转发缓存的查找 361
    - 15.3.4 向组播路由守护进程发送  
报告 362
  - 15.4 临时组播转发缓存 364
    - 15.4.1 临时组播转发缓存队列 365
    - 15.4.2 创建临时组播转发缓存 365
    - 15.4.3 用于超时而删除临时组播  
转发缓存的定时器 367
    - 15.4.4 释放临时组播缓存项中保存的  
临时组播报文 368
  - 15.5 外部事件 369
  - 15.6 组播套接口选项 369
    - 15.6.1 IP\_MULTICAST\_TTL 369
    - 15.6.2 IP\_MULTICAST\_LOOP 370
    - 15.6.3 IP\_MULTICAST\_IF 370
    - 15.6.4 IP\_ADD\_MEMBERSHIP 372
    - 15.6.5 IP\_DROP\_MEMBERSHIP 372
    - 15.6.6 IP\_MSFILTER 373
    - 15.6.7 IP\_BLOCK\_SOURCE和  
IP\_UNBLOCK\_SOURCE 375
    - 15.6.8 IP\_ADD\_SOURCE\_MEMBERSHIP  
和IP\_DROP\_SOURCE\_  
MEMBERSHIP 375
    - 15.6.9 MCAST\_JOIN\_GROUP 376
    - 15.6.10 MCAST\_LEAVE\_GROUP 377
    - 15.6.11 MCAST\_BLOCK\_SOURCE和  
MCAST\_UNBLOCK\_SOURCE 377
    - 15.6.12 MCAST\_JOIN\_SOURCE\_GROUP  
和MCAST\_LEAVE\_SOURCE\_

- GROUP 377
- 15.6.13 MCAST\_MSFILTER 378
- 15.7 组播选路套接口选项 378
  - 15.7.1 MRT\_INIT 379
  - 15.7.2 MRT\_DONE 379
  - 15.7.3 MRT\_ADD\_VIF和MRT\_DEL\_VIF 380
  - 15.7.4 MRT\_ADD\_MFC和MRT\_DEL\_MFC 380
  - 15.7.5 MRT\_ASSERT 380
- 15.8 组播的ioctl 380
  - 15.8.1 SIOCGETVIFCNT 380
  - 15.8.2 SIOCGETSGCNT 380
- 15.9 组播报文的输入 381
- 15.10 组播报文的转发 383
  - 15.10.1 ip\_mr\_forward() 383
  - 15.10.2 ipmr\_queue\_xmit() 385
- 15.11 组播报文的输出 388
- 第16章 IGMP: Internet组管理协议 390
  - 16.1 in\_device结构中的组播参数 390
  - 16.2 ip\_mc\_list结构 391
  - 16.3 系统参数 393
  - 16.4 IGMP的版本与协议结构 393
    - 16.4.1 IGMP的版本 393
    - 16.4.2 第一版和第二版的IGMP报文结构 395
    - 16.4.3 第三版的IGMP查询报文结构 395
    - 16.4.4 第三版的IGMP报告结构 396
  - 16.5 IGMP报文的输入 398
  - 16.6 函数 399
    - 16.6.1 ip\_mc\_find\_dev() 399
    - 16.6.2 ip\_check\_mc() 400
  - 16.7 成员关系查询 400
  - 16.8 成员关系报告 404
    - 16.8.1 最近离开组播组列表的维护 404
    - 16.8.2 is\_in() 404
    - 16.8.3 add\_grec() 406
    - 16.8.4 普通查询的报告 409
    - 16.8.5 V1和V2的报告以及V3的当前状态记录报告 410
    - 16.8.6 主动发送组关系报告 413
  - 16.9 维护套接口组播状态 416
    - 16.9.1 套接口加入组播组 417
    - 16.9.2 套接口离开组播组 418
  - 16.10 维护网络设备组播状态 419
    - 16.10.1 被阻止的组播源列表的维护 421
    - 16.10.2 网络设备加入组播组 421
    - 16.10.3 网络设备离开组播组 425
  - 16.11 ip\_mc\_source() 430
  - 16.12 ip\_mc\_msfilter() 434
  - 16.13 网络设备组播硬件地址的管理 436
- 第17章 邻居子系统 437



- 17.1 什么是邻居子系统 437
- 17.2 系统参数 437
- 17.3 邻居子系统的结构 438
  - 17.3.1 neigh\_table结构 438
  - 17.3.2 neighbour结构 441
  - 17.3.3 neigh\_ops结构 444
  - 17.3.4 neigh\_parms结构 445
  - 17.3.5 pneigh\_entry结构 447
  - 17.3.6 neigh\_statistics结构 447
  - 17.3.7 hh\_cache结构 448
- 17.4 邻居表的初始化 449
- 17.5 邻居项的状态机 450
- 17.6 邻居项的添加与删除 452
  - 17.6.1 netlink接口 452
  - 17.6.2 ioctl 456
  - 17.6.3 路由表项与邻居项的绑定 456
  - 17.6.4 接收到的并非请求的应答 456
- 17.7 邻居项的创建与初始化 456
  - 17.7.1 neigh\_alloc() 456
  - 17.7.2 neigh\_create() 457
- 17.8 邻居项散列表的扩容 459
- 17.9 邻居项的查找 460
  - 17.9.1 neigh\_lookup() 460
  - 17.9.2 neigh\_lookup\_nodev() 461
  - 17.9.3 \_\_neigh\_lookup()和  
neigh\_lookup\_errno() 461
- 17.10 邻居项的更新 461
- 17.11 垃圾回收 465
  - 17.11.1 同步回收 465
  - 17.11.2 异步回收 466
- 17.12 外部事件 468
- 17.13 邻居项状态处理定时器 469
- 17.14 代理项 472
  - 17.14.1 代理项的查找、添加和删除 472
  - 17.14.2 延时处理代理的请求报文 472
- 17.15 输出函数 474
  - 17.15.1 丢弃 474
  - 17.15.2 慢速发送 474
  - 17.15.3 快速发送 477
- 第18章 ARP：地址解析协议 480
- 18.1 ARP报文格式 480
- 18.2 系统参数 481
- 18.3 注册ARP报文类型 483
- 18.4 ARP初始化 483
- 18.5 ARP的邻居项函数指针表 483
- 18.6 ARP表 484
- 18.7 函数 485
  - 18.7.1 arp\_error\_report() 485
  - 18.7.2 arp\_solicit() 485
  - 18.7.3 arp\_ignore() 486
  - 18.7.4 arp\_filter() 488
- 18.8 IPv4中邻居项的初始化 488
- 18.9 ARP报文的创建 490
- 18.10 ARP的输出 490

- 18.11 ARP的输入 491
  - 18.11.1 arp\_rcv() 491
  - 18.11.2 arp\_process() 492
- 18.12 ARP代理 497
  - 18.12.1 arp\_process() 498
  - 18.12.2 arp\_fwd\_proxy() 499
  - 18.12.3 parp\_redo() 500
- 18.13 ARP的ioctl 500
- 18.14 外部事件 501
- 18.15 路由表项与邻居项的绑定 502
- 第19章 路由表 503
  - 19.1 什么是路由表 503

- 19.1.1 路由的要素 503
- 19.1.2 特殊路由 505
- 19.1.3 路由缓存 505
- 19.2 系统参数 506
- 19.3 路由表组成结构 506
  - 19.3.1 fib\_table结构 508
  - 19.3.2 fn\_zone结构 510
  - 19.3.3 fib\_node结构 511
  - 19.3.4 fib\_alias结构 511
  - 19.3.5 fib\_info结构 512
  - 19.3.6 fib\_nh结构 515
- 19.4 路由表的初始化 516
- 19.5 netlink接口 517
  - 19.5.1 netlink路由表项消息结构 517
  - 19.5.2 inet\_rtm\_newroute() 519
  - 19.5.3 inet\_rtm\_delroute() 520
- 19.6 获取指定的路由表 520
- 19.7 路由表项的添加 520
- 19.8 路由表项的删除 526
- 19.9 外部事件 528
  - 19.9.1 网络设备状态变化事件 528
  - 19.9.2 IP地址变化事件 529
  - 19.9.3 fib\_add\_ifaddr() 529
  - 19.9.4 fib\_del\_ifaddr() 531
  - 19.9.5 fib\_disable\_ip() 534
  - 19.9.6 fib\_magic() 534
- 19.10 选路 535
  - 19.10.1 输入选路：  
ip\_route\_input\_slow() 535
  - 19.10.2 组播输入选路：  
ip\_route\_input\_mc() 539
  - 19.10.3 输出选路：  
ip\_route\_output\_slow() 541
  - 19.10.4 fib\_lookup() 546
  - 19.10.5 fn\_hash\_lookup() 548
- 19.11 ICMP重定向消息的发送 548

## 下册目录

- 第20章 路由缓存 551
  - 20.1 系统参数 551

- 20.2 路由缓存的组织结构 552
  - 20.2.1 rtable结构 552
  - 20.2.2 flowi结构 555
  - 20.2.3 dst\_entry结构 556
  - 20.2.4 dst\_ops结构 559
- 20.3 初始化 561
- 20.4 创建路由缓存项 563
  - 20.4.1 创建输入路由缓存项 563
  - 20.4.2 创建输出路由缓存项 565
- 20.5 添加路由表项到缓存中：  
rt\_intern\_hash() 568
- 20.6 输入路由缓存查询：  
ip\_route\_input() 571
- 20.7 输出路由缓存查询 573
  - 20.7.1 ip\_route\_output\_key() 573
  - 20.7.2 \_\_ip\_route\_output\_key() 573
- 20.8 垃圾回收 575
  - 20.8.1 路由缓存项的过期 575
  - 20.8.2 判断缓存路由表项是否  
可被删除 575
  - 20.8.3 同步清理 576
  - 20.8.4 异步清理 580
  - 20.8.5 路由缓存项的释放 582
- 20.9 刷新缓存 582
  - 20.9.1 通过定时器定时刷新 584
  - 20.9.2 网络设备的硬件地址发生  
改变 584
  - 20.9.3 网络设备状态发生变化 584
  - 20.9.4 给设备添加或删除一个  
IP地址 584
  - 20.9.5 全局转发状态或设备的转发  
状态发生变化 584
  - 20.9.6 一条路由被删除 585
  - 20.9.7 通过写/proc的flush文件 585
- 20.10 ICMP重定向消息的处理 585
- 20.11 ICMP目的不可达，需要分片  
消息的处理 588
- 第21章 路由策略 590
  - 21.1 路由策略组织结构 590
    - 21.1.1 fib\_rules\_ops结构 590
    - 21.1.2 fib\_rule结构 592
    - 21.1.3 fib4\_rule结构 594
  - 21.2 三个默认路由策略 595
  - 21.3 IPv4协议族的fib\_rules\_ops  
结构实例 595
    - 21.3.1 fib4\_rule\_action() 595
    - 21.3.2 fib4\_rule\_match() 596
    - 21.3.3 fib4\_rule\_configure() 596
    - 21.3.4 fib4\_rule\_compare() 598
    - 21.3.5 fib4\_rule\_fill() 598
    - 21.3.6 fib4\_rule\_default\_pref() 599
  - 21.4 netlink接口 599
    - 21.4.1 netlink路由策略消息结构 599
    - 21.4.2 fib\_nl\_newrule() 600

- 21.4.3 fib\_nl\_dehrule() 602
- 21.5 受网络设备状态改变的影响 604
- 21.6 策略路由的查找 604
- 第22章 套接口层 606
  - 22.1 socket结构 607
  - 22.2 proto\_ops结构 608
  - 22.3 套接口文件系统 610
    - 22.3.1 套接口文件系统类型 610
    - 22.3.2 套接口文件系统超级块操作接口 610
    - 22.3.3 套接口文件的inode 611
    - 22.3.4 sock\_alloc\_inode() 611
    - 22.3.5 sock\_destroy\_inode() 612
  - 22.4 套接口文件 612
    - 22.4.1 套接口文件与套接口的绑定 612
    - 22.4.2 根据文件描述符获取套接口 614
  - 22.5 进程、文件描述符和套接口 615
  - 22.6 套接口层的系统初始化 616
  - 22.7 套接口系统调用 617
    - 22.7.1 套接口系统调用入口 617
    - 22.7.2 socket系统调用 621
    - 22.7.3 bind系统调用 629
    - 22.7.4 listen系统调用 632
    - 22.7.5 accept系统调用 633
    - 22.7.6 connect系统调用 635
    - 22.7.7 shutdown系统调用 636
    - 22.7.8 close系统调用 638
    - 22.7.9 select系统调用的实现 640
- 第23章 套接口I/O 641
  - 23.1 输出/输入数据的组织 641
    - 23.1.1 msghdr结构 641
    - 23.1.2 verify\_iovec() 643
    - 23.1.3 memcpy\_toiovec() 644
    - 23.1.4 memcpy\_fromiovec() 644
    - 23.1.5 memcpy\_fromiovecend() 644
    - 23.1.6 csum\_partial\_copy\_fromiovecend() 644
  - 23.2 输出系统调用 644
    - 23.2.1 sock\_sendmsg() 644
    - 23.2.2 sendto系统调用 645
    - 23.2.3 send系统调用 646
    - 23.2.4 sendmsg系统调用 646
  - 23.3 输入系统调用 649
- 第24章 套接口选项 650
  - 24.1 setsockopt系统调用 650
  - 24.2 ioctl系统调用 655
    - 24.2.1 ioctl在文件系统内的调用过程 655
    - 24.2.2 套接口文件ioctl调用接口的实现 655
    - 24.2.3 套接口层的实现 658
  - 24.3 getsockname系统调用 659
  - 24.4 getpeername系统调用 660
- 第25章 传输控制块 661
  - 25.1 系统参数 662

- 25.2 传输描述块结构 662
  - 25.2.1 sock\_common结构 662
  - 25.2.2 sock结构 663
  - 25.2.3 inet\_sock结构 670
- 25.3 proto结构 674
  - 25.3.1 proto实例组织结构 677
  - 25.3.2 proto\_register() 677
  - 25.3.3 proto\_unregister() 679
- 25.4 传输控制块的内存管理 680
  - 25.4.1 传输控制块的分配和释放 680
  - 25.4.2 普通的发送缓存区的分配 682
  - 25.4.3 发送缓存的分配与释放 685
  - 25.4.4 接收缓存的分配与释放 686
  - 25.4.5 辅助缓存的分配与释放 688
- 25.5 异步IO机制 688
  - 25.5.1 sk\_wake\_async() 689
  - 25.5.2 sock\_def\_wakeup() 690
  - 25.5.3 sock\_def\_error\_report() 690
  - 25.5.4 sock\_def\_readable() 691
  - 25.5.5 sock\_def\_write\_space()和sk\_stream\_write\_space() 691
  - 25.5.6 sk\_send\_sigurg() 692
  - 25.5.7 接收到FIN段后通知进程 692
  - 25.5.8 sock\_fasync() 693
- 25.6 传输控制块的同步锁 694
  - 25.6.1 socket\_lock\_t结构 694
  - 25.6.2 控制用户进程和下半部间同步锁 695
  - 25.6.3 控制下半部间同步锁 698
- 第26章 TCP：传输控制协议 699
  - 26.1 系统参数 699
  - 26.2 TCP的inet\_protosw实例 705
  - 26.3 TCP的net\_protocol结构 705
  - 26.4 TCP传输控制块 706
    - 26.4.1 inet\_connection\_sock结构 706
    - 26.4.2 inet\_connection\_sock\_af\_ops结构 710
    - 26.4.3 tcp\_sock结构 711
    - 26.4.4 tcp\_options\_received结构 721
    - 26.4.5 tcp\_skb\_cb结构 723
  - 26.5 TCP的proto结构和proto\_ops结构的实例 725
  - 26.6 TCP状态迁移图 725
  - 26.7 TCP首部 726
  - 26.8 TCP校验和 727
    - 26.8.1 输入TCP段的校验和检测 728
    - 26.8.2 输出TCP段校验和的计算 729
  - 26.9 TCP的初始化 729
  - 26.10 TCP传输控制块的管理 731
    - 26.10.1 inet\_hashinfo结构 732
    - 26.10.2 管理除LISTEN状态之外的TCP传输控制块 733
    - 26.10.3 管理LISTEN状态的TCP传输控制块 734

- 26.11 TCP层的套接口选项 735
- 26.12 TCP的ioctl 736
- 26.13 TCP传输控制块的初始化 737
- 26.14 TCP的差错处理 737
- 26.15 TCP传输控制块层的缓存管理 741
  - 26.15.1 缓存管理的算法 741
  - 26.15.2 发送缓存的管理 744
  - 26.15.3 接收缓存的管理 745
- 第27章 TCP的定时器 746
  - 27.1 初始化 746
  - 27.2 连接建立定时器 747
    - 27.2.1 连接建立定时器处理函数 747
    - 27.2.2 连接建立定时器的激活 751
  - 27.3 重传定时器 751
    - 27.3.1 重传定时器处理函数 751
    - 27.3.2 重传定时器的激活 756
  - 27.4 延迟确认定时器 756
    - 27.4.1 延迟确认定时器的处理函数 756
    - 27.4.2 延迟确认定时器的激活 758
  - 27.5 持续定时器 758
    - 27.5.1 持续定时器处理函数 758
    - 27.5.2 激活持续定时器 762
  - 27.6 保活定时器 763
    - 27.6.1 保活定时器处理函数 763
    - 27.6.2 激活保活定时器 764
  - 27.7 FIN\_WAIT\_2定时器 764
    - 27.7.1 FIN\_WAIT\_2定时器处理函数 765
    - 27.7.2 激活FIN\_WAIT\_2定时器 765
  - 27.8 TIME\_WAIT定时器 766
- 第28章 TCP连接的建立 767
  - 28.1 服务端建立连接过程 767
  - 28.2 连接相关的数据结构 770
    - 28.2.1 request\_sock\_queue结构 770
    - 28.2.2 listen\_sock结构 771
    - 28.2.3 tcp\_request\_sock结构 771
    - 28.2.4 request\_sock\_ops结构 774
  - 28.3 bind系统调用的实现 775
    - 28.3.1 bind端口散列表 775
    - 28.3.2 传输接口层的实现 775
  - 28.4 listen系统调用的实现 779
    - 28.4.1 inet\_listen() 779
    - 28.4.2 实现侦听：  
inet\_csk\_listen\_start() 780
    - 28.4.3 分配连接请求块散列表：  
reqsk\_queue\_alloc() 781
  - 28.5 accept系统调用的实现 782
    - 28.5.1 套接口层的实现：  
inet\_accept() 782
    - 28.5.2 传输接口层的实现：  
inet\_csk\_accept() 783
  - 28.6 被动打开 785
    - 28.6.1 SYN cookies 785
    - 28.6.2 第一次握手：接收SYN段 786

- 28.6.3 第二次握手：
  - 发送SYN+ACK段 793
- 28.6.4 第三次握手：接收ACK段 798
- 28.7 connect系统调用的实现 813
  - 28.7.1 套接口层的实现：
    - inet\_stream\_connect() 813
  - 28.7.2 传输接口层的实现 815
- 28.8 主动打开 816
  - 28.8.1 第一次握手：发送SYN段 816
  - 28.8.2 第二次握手：
    - 接收SYN+ACK段 823
  - 28.8.3 第三次握手：发送ACK段 828
- 28.9 同时打开 828
  - 28.9.1 SYN\_SENT状态接收SYN段 828
  - 28.9.2 SYN\_RECV状态接收SYN+ACK段 830
- 第29章 TCP拥塞控制的实现 831
  - 29.1 拥塞控制引擎 831
  - 29.2 拥塞控制状态机 832
    - 29.2.1 Open状态 833
    - 29.2.2 Disorder状态 833
    - 29.2.3 CWR状态 833
    - 29.2.4 Recovery状态 834
    - 29.2.5 Loss状态 834
  - 29.3 拥塞窗口调整撤销 836
    - 29.3.1 撤销拥塞窗口的检测 837
    - 29.3.2 tcp\_undo\_cwr() 837
    - 29.3.3 从Disorder拥塞状态撤销 838
    - 29.3.4 从Recovery状态撤销 838
    - 29.3.5 从Recovery拥塞状态撤销 839
    - 29.3.6 从Loss拥塞状态撤销 839
  - 29.4 显式拥塞通知 840
    - 29.4.1 IP对ECN的支持 841
    - 29.4.2 TCP对ECN的支持 841
  - 29.5 拥塞控制状态的处理及转换 843
    - 29.5.1 拥塞控制状态的处理：
      - tcp\_fastretrans\_alert() 843
    - 29.5.2 拥塞避免 852
  - 29.6 拥塞窗口的检测：
    - tcp\_cwnd\_test() 852
  - 29.7 F-RTO算法 853
    - 29.7.1 进入F-RTO算法处理阶段 853
    - 29.7.2 进行F-RTO算法处理 855
  - 29.8 拥塞窗口的检验 857
    - 29.8.1 tcp\_event\_data\_sent() 857
    - 29.8.2 tcp\_cwnd\_validate() 858
  - 29.9 支持多拥塞控制算法的机制 859
    - 29.9.1 接口 859
    - 29.9.2 注册拥塞控制算法：tcp\_register\_congestion\_control() 861
    - 29.9.3 注销拥塞控制算法：tcp\_unregister\_congestion\_control() 861
    - 29.9.4 选取某种拥塞控制算法：tcp\_set\_congestion\_control() 861

- 29.9.5 Linux支持的拥塞控制算法 862
- 第30章 TCP的输出 864
  - 30.1 引言 864
  - 30.2 最大段长度 (MSS) 867
  - 30.3 sendmsg系统调用在TCP中的实现 870
    - 30.3.1 分割TCP段 871
    - 30.3.2 套接口层的实现 871
    - 30.3.3 传输接口层的实现 871
  - 30.4 对TCP选项的处理 889
    - 30.4.1 构建SYN段的选项 889
    - 30.4.2 构建非SYN段的选项 892
  - 30.5 Nagle算法 893
  - 30.6 ACK的接收 894
    - 30.6.1 tcp\_ack() 894
    - 30.6.2 发送窗口的更新 899
    - 30.6.3 根据SACK选项标记重传队列中段的记分牌 900
    - 30.6.4 重传队列中已经确认段的删除 910
  - 30.7 往返时间测量和RTO的计算 913
  - 30.8 路径MTU发现 915
    - 30.8.1 路径MTU发现原理 915
    - 30.8.2 路径MTU发现时的黑洞 916
    - 30.8.3 有关数据结构的初始化 916
    - 30.8.4 创建路径MTU发现TCP段并发送 916
    - 30.8.5 路径MTU发现失败后处理 920
    - 30.8.6 处理需要分片ICMP目的不可达报文 920
    - 30.8.7 更新当前有效的MSS 921
    - 30.8.8 路径MTU发现成功后处理 922
  - 30.9 TCP重传接口 922
- 第31章 TCP的输入 926
  - 31.1 引言 926
  - 31.2 TCP接收的总入口 927
    - 31.2.1 接收到prequeue队列 930
    - 31.2.2 有效TCP段的处理 931
  - 31.3 报文的过滤 932
    - 31.3.1 过滤器的数据结构 933
    - 31.3.2 安装过滤器 935
    - 31.3.3 卸载过滤器 937
    - 31.3.4 过滤执行 938
  - 31.4 ESTABLISHED状态的接收 938
    - 31.4.1 首部预测 939
    - 31.4.2 接收处理无负荷的ACK段 941
    - 31.4.3 执行快速路径 942
    - 31.4.4 执行慢速路径 945
    - 31.4.5 数据从内核空间复制到用户空间 948
    - 31.4.6 通过调节接收窗口进行流量控制 952
    - 31.4.7 确定是否需要发送ACK段（用于接收的数据从内核空间复制到用户空间时） 956



- 31.5 TCP选项的处理 957
  - 31.5.1 慢速路径中快速解析TCP选项 957
  - 31.5.2 全面解析TCP选项 958
- 31.6 慢速路径的数据处理 961
  - 31.6.1 接收处理预期的段 963
  - 31.6.2 接收处理在接收窗口之外的段 965
  - 31.6.3 接收处理乱序的段 966
  - 31.6.4 tcp\_ofo\_queue() 969
- 31.7 带外数据处理 970
  - 31.7.1 检测紧急指针 970
  - 31.7.2 读取带外数据 972
- 31.8 SACK信息 973
  - 31.8.1 SACK允许选项 973
  - 31.8.2 SACK选项 974
  - 31.8.3 SACK的产生 974
  - 31.8.4 发送方对SACK的响应 975
  - 31.8.5 实现 975
- 31.9 确认的发送 975
  - 31.9.1 快速确认模式 976
  - 31.9.2 处理数据接收事件 977
  - 31.9.3 发送确认紧急程度和状态 978
  - 31.9.4 延迟或快速确认 979
- 31.10 recvmsg系统调用在TCP中的实现 980
  - 31.10.1 套接口层的实现 980
  - 31.10.2 传输接口层的实现 980
- 31.11 sk\_backlog\_rcv接口 991
- 第32章 TCP连接的终止 992
  - 32.1 连接终止过程 993
    - 32.1.1 正常关闭 993
    - 32.1.2 同时关闭 994
  - 32.2 shutdown传输接口层的实现 994
    - 32.2.1 tcp\_shutdown() 994
    - 32.2.2 tcp\_send\_fin() 995
  - 32.3 close传输接口层的实现：tcp\_close() 995
  - 32.4 被动关闭：FIN段的接收处理 999
  - 32.5 主动关闭 1002
    - 32.5.1 timewait控制块的数据结构 1002
    - 32.5.2 timewait控制块取代TCP传输控制块 1006
    - 32.5.3 启动FIN\_WAIT\_2或TIME\_WAIT定时器 1008
    - 32.5.4 CLOSE\_WAIT、LAST\_ACK、FIN\_WAIT1、FIN\_WAIT2与CLOSING状态处理 1010
    - 32.5.5 FIN\_WAIT2和TIME\_WAIT状态处理 1013
    - 32.5.6 timewait控制块的2MSL超时处理 1020
- 第33章 UDP：用户数据报 1023

33.1 引言 1023  
33.1.1 UDP首部 1023  
33.1.2 UDP的输入与输出 1024  
33.2 UDP的inet\_protosw结构 1024  
33.3 UDP的传输控制块 1025  
33.4 UDP的proto结构和proto\_ops结构的实例 1027  
33.5 UDP的状态 1027  
33.6 UDP传输控制块的管理 1027  
33.7 bind系统调用的实现 1028  
33.8 UDP套接口的关闭 1031  
33.9 connect系统调用的实现 1032  
33.9.1 udp\_disconnect() 1033  
33.9.2 ip4\_datagram\_connect() 1033  
33.10 select系统调用的实现 1034  
33.11 UDP的ioctl 1037  
33.12 UDP的套接口选项 1037  
33.13 UDP校验和 1038  
33.13.1 输入UDP数据报校验和的计算 1038  
33.13.2 输出UDP数据报校验和的计算 1039  
33.14 UDP的输出：sendmsg系统调用 1040  
33.14.1 udp\_sendmsg() 1040  
33.14.2 udp\_push\_pending\_frames() 1047  
33.15 UDP的输入 1048  
33.15.1 UDP接收的入口：udp\_rcv() 1048  
33.15.2 UDP组播数据报输入：\_\_udp4\_lib\_mcast\_deliver() 1052  
33.15.3 udp\_queue\_rcv\_skb() 1053  
33.16 recvmsg系统调用的实现 1055  
33.17 UDP的差错处理：udp\_err() 1059  
33.18 轻量级UDP 1061  
参考文献 1063  
• • • • • ([收起](#))

[Linux内核源码剖析（套装上下册）\\_下载链接1](#)

标签

TCP/IP

linux

内核

Linux

网络

计算机

协议栈

网络编程

## 评论

我只用到了下册，上册没看，下册写的确实不错，连拥塞状态机这种不好找的资料都写得很明白。

-----  
整体来说还可以，含量不高，好多地方想过场子一样，没有点明白！拥塞控制那章真是太烂了 so suck!

-----  
网络协议栈部分参考资料 2017.5月 学习组播

-----  
上册相当一部分照搬深入Linux网络技术内幕

-----  
当成手册查看的。很多内容在网上都可以搜到。不想费事，就买了。相比较国内的其他书而言，更适合初学者对着这本书去看代码。

-----  
只是讲代码，设计理念很少，还得看RFC和相关propose论文。和国外书籍还是有差距

，不过总的来说，算是讲TCP协议最全的一本了。tcp/ip architecture那本书倒是过时了。（讲的2.4内核）

-----  
[Linux内核源码剖析（套装上下册）\\_下载链接1](#)

## 书评

作为IT人，看过的IT技术书不少了，原来的印象是国内的IT书好的很少，尤其是有大牛挂名的书（非第一作者的），质量其实更是没有保证。这本书的作者虽然没什么名气，但往往国内的精品书就是这样的人写的。我与《深入理解网络Linux技术内幕》（包括英文版）对照着看了，《...

-----  
[Linux内核源码剖析（套装上下册）\\_下载链接1](#)