

《Linux命令行大全》

图书基本信息

书名：《Linux命令行大全》

13位ISBN编号：9787115307453

10位ISBN编号：7115307458

出版时间：2013-3-1

出版社：人民邮电出版社

作者：绍茨 (William E.Shotts)

页数：428

译者：郭光伟,郝记生

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.tushu111.com

《Linux命令行大全》

内容概要

《Linux命令行大全》主要介绍Linux命令行的使用，循序渐进，深入浅出，引导读者全面掌握命令行的使用方法。

《Linux命令行大全》分为四部分。第一部分开始了对命令行基本语言的学习之旅，包括命令结构、文件系统的导引、命令行的编辑以及关于命令的帮助系统和使用手册。第二部分主要讲述配置文件的编辑，用于计算机操作的命令行控制。第三部分讲述了从命令行开始执行的常规任务。类UNIX操作系统，比如Linux，包含了很多“经典的”命令行程序，这些程序可以高效地对数据进行操作。第四部分介绍了shell编程，这是一个公认的初级技术，并且容易学习，它可以使很多常见的系统任务自动运行。通过学习shell编程，读者也可以熟悉其他编程语言的使用。

《Linux命令行大全》适合从其他平台过渡到Linux的新用户和初级Linux服务器管理员阅读。没有任何Linux基础和Linux编程经验的读者，也可以通过本书掌握Linux命令行的使用方法。

《Linux命令行大全》

作者简介

William E. Shotts, Jr.，作为一名软件开发人员和狂热的Linux用户已经有15年之久。他在软件开发领域有广泛的背景，先后涉及过技术支持、质量保证和文档编写等工作。他还是LinuxCommand.org网站的创始人，该网站是一个Linux教育和宣传网站，以新闻、评论和为人们使用Linux命令行提供广泛支持而见长。

书籍目录

第一部分 学习shell

第1章 shell是什么 3

- 1.1 终端仿真器 3
- 1.2 第一次键盘输入 4
 - 1.2.1 命令历史记录 4
 - 1.2.2 光标移动 4
- 1.3 几个简单的命令 5
- 1.4 结束终端会话 6

第2章 导航 7

- 2.1 理解文件系统树 7
- 2.2 当前工作目录 8
- 2.3 列出目录内容 9
- 2.4 更改当前工作目录 9
 - 2.4.1 绝对路径名 9
 - 2.4.2 相对路径名 9
 - 2.4.3 一些有用的快捷方式 10

第3章 linux系统 13

- 3.1 ls命令的乐趣 13
 - 3.1.1 选项和参数 14
 - 3.1.2 进一步了解长列表格式 15
- 3.2 使用file命令确定文件类型 16
- 3.3 使用less命令查看文件内容 16
- 3.4 快速浏览 18
- 3.5 符号链接 20

第4章 操作文件与目录 23

- 4.1 通配符 24
- 4.2 mkdir——创建目录 26
- 4.3 cp——复制文件和目录 26
- 4.4 mv——移除和重命名文件 27
- 4.5 rm——删除文件和目录 28
- 4.6 ln——创建链接 29
 - 4.6.1 硬链接 29
 - 4.6.2 符号链接 30
- 4.7 实战演练 30
 - 4.7.1 创建目录 30
 - 4.7.2 复制文件 31
 - 4.7.3 移动和重命名文件 31
 - 4.7.4 创建硬链接 32
 - 4.7.5 创建符号链接 33
 - 4.7.6 移除文件和目录 34
- 4.8 本章结尾语 35

第5章 命令的使用 37

- 5.1 究竟什么是命令 38
- 5.2 识别命令 38
 - 5.2.1 type——显示命令的类型 38
 - 5.2.2 which——显示可执行程序的位置 39
- 5.3 获得命令文档 39

- 5.3.1 help——获得shell内置命令的帮助文档 39
- 5.3.2 help——显示命令的使用信息 40
- 5.3.3 man——显示程序的手册页 40
- 5.3.4 apropos——显示合适的命令 41
- 5.3.5 whatis——显示命令的简要描述 42
- 5.3.6 info——显示程序的info条目 42
- 5.3.7 readme和其他程序文档文件 43
- 5.4 使用别名创建自己的命令 43
- 5.5 温故以求新 45
- 第6章 重定向 47
 - 6.1 标准输入、标准输出和标准错误 48
 - 6.1.1 标准输出重定向 48
 - 6.1.2 标准错误重定向 50
 - 6.1.3 将标准输出和标准错误重定向到同一个文件 50
 - 6.1.4 处理不想要的输出 51
 - 6.1.5 标准输入重定向 51
 - 6.2 管道 53
 - 6.2.1 过滤器 53
 - 6.2.2 uniq——报告或忽略文件中重复的行 54
 - 6.2.3 wc——打印行数、字数和字节数 54
 - 6.2.4 grep——打印匹配行 54
 - 6.2.5 head/tail——输出文件的开头部分/结尾部分 55
 - 6.2.6 tee——从stdin读取数据，并同时输出到stdout和文件 56
 - 6.3 本章结尾语 57
- 第7章 透过shell看世界 59
 - 7.1 扩展 59
 - 7.1.1 路径名扩展 60
 - 7.1.2 波浪线扩展 61
 - 7.1.3 算术扩展 61
 - 7.1.4 花括号扩展 62
 - 7.1.5 参数扩展 63
 - 7.1.6 命令替换 64
 - 7.2 引用 65
 - 7.2.1 双引号 65
 - 7.2.2 单引号 67
 - 7.2.3 转义字符 67
 - 7.3 本章结尾语 68
- 第8章 高级键盘技巧 69
 - 8.1 编辑命令行 69
 - 8.1.1 光标移动 70
 - 8.1.2 修改文本 70
 - 8.1.3 剪切和粘贴(killing and yanking)文本 71
 - 8.2 自动补齐功能 71
 - 8.3 使用历史命令 73
 - 8.3.1 搜索历史命令 73
 - 8.3.2 历史记录扩展 75
 - 8.4 本章结尾语 76
- 第9章 权限 77
 - 9.1 所有者、组成员和其他所有用户 78

- 9.2 读取、写入和执行 79
 - 9.2.1 chmod——更改文件模式 81
 - 9.2.2 采用gui设置文件模式 84
 - 9.2.3 umask——设置默认权限 85
- 9.3 更改身份 87
 - 9.3.1 su——以其他用户和组id的身份来运行shell 88
 - 9.3.2 sudo——以另一个用户的身份执行命令 89
 - 9.3.3 chown——更改文件所有者和所属群组 90
 - 9.3.4 chgrp——更改文件所属群组 91
- 9.4 权限的使用 91
- 9.5 更改用户密码 93
- 第10章 进程 95
 - 10.1 进程如何工作 96
 - 10.1.1 使用ps命令查看进程信息 96
 - 10.1.2 使用top命令动态查看进程信息 98
 - 10.2 控制进程 100
 - 10.2.1 中断进程 100
 - 10.2.2 使进程在后台运行 101
 - 10.2.3 使进程回到前台运行 101
 - 10.2.4 停止(暂停)进程 102
 - 10.3 信号 102
 - 10.3.1 使用kill命令发送信号到进程 103
 - 10.3.2 使用killall命令发送信号给多个进程 105
 - 10.4 更多与进程相关的命令 105
- 第二部分 配置与环境
- 第11章 环境 109
 - 11.1 环境中存储的是什么 109
 - 11.1.1 检查环境 110
 - 11.1.2 一些有趣的变量 111
 - 11.2 环境是如何建立的 112
 - 11.2.1 login和non-login shell 112
 - 11.2.2 启动文件中有什么 113
 - 11.3 修改环境 114
 - 11.3.1 用户应当修改哪些文件 114
 - 11.3.2 文本编辑器 115
 - 11.3.3 使用文本编辑器 115
 - 11.3.4 激活我们的修改 117
 - 11.4 本章结尾语 118
- 第12章 vi简介 119
 - 12.1 为什么要学习vi 119
 - 12.2 vi背景 120
 - 12.3 启动和退出vi 120
 - 12.4 编辑模式 121
 - 12.4.1 进入插入模式 122
 - 12.4.2 保存工作 122
 - 12.5 移动光标 123
 - 12.6 基本编辑 124
 - 12.6.1 添加文本 124
 - 12.6.2 插入一行 125

- 12.6.3 删除文本 126
- 12.6.4 剪切、复制和粘贴文本 127
- 12.6.5 合并行 128
- 12.7 查找和替换 128
 - 12.7.1 行内搜索 128
 - 12.7.2 搜索整个文件 129
 - 12.7.3 全局搜索和替换 129
- 12.8 编辑多个文件 130
 - 12.8.1 切换文件 131
 - 12.8.2 载入更多的文件 132
 - 12.8.3 文件之间的内容复制 132
 - 12.8.4 插入整个文件 133
- 12.9 保存工作 134
- 第13章 定制提示符 135
 - 13.1 提示符的分解 135
 - 13.2 尝试设计提示符 137
 - 13.3 添加颜色 138
 - 13.4 移动光标 140
 - 13.5 保存提示符 141
 - 13.6 本章结尾语 141
- 第三部分 常见任务和主要工具
- 第14章 软件包管理 145
 - 14.1 软件包系统 146
 - 14.2 软件包系统工作方式 146
 - 14.2.1 软件包文件 146
 - 14.2.2 库 147
 - 14.2.3 依赖关系 147
 - 14.2.4 高级和低级软件包工具 147
 - 14.3 常见软件包管理任务 148
 - 14.3.1 在库里面查找软件包 148
 - 14.3.2 安装库中的软件包 148
 - 14.3.3 安装软件包文件中的软件包 149
 - 14.3.4 删除软件包 149
 - 14.3.5 更新库中的软件包 150
 - 14.3.6 更新软件包文件中的软件包 150
 - 14.3.7 列出已安装的软件包列表 150
 - 14.3.8 判断软件包是否安装 151
 - 14.3.9 显示已安装软件包的相关信息 151
 - 14.3.10 查看某具体文件由哪个软件包安装得到 151
 - 14.4 本章结尾语 152
- 第15章 存储介质 155
 - 15.1 挂载、卸载存储设备 156
 - 15.1.1 查看已挂载的文件系统列表 157
 - 15.1.2 确定设备名称 160
 - 15.2 创建新的文件系统 162
 - 15.2.1 用fdisk命令进行磁盘分区 162
 - 15.2.2 用mkfs命令创建新的文件系统 164
 - 15.3 测试、修复文件系统 165
 - 15.4 格式化软盘 166

- 15.5 直接从/向设备转移数据 166
- 15.6 创建cd-rom映像 167
 - 15.6.1 创建一个cd-rom文件映像副本 167
 - 15.6.2 从文件集合中创建映像文件 168
- 15.7 向cd-rom写入映像文件 168
 - 15.7.1 直接挂载iso映像文件 168
 - 15.7.2 擦除可读写cd-rom 169
 - 15.7.3 写入映像文件 169
- 15.8 附加认证 169
- 第16章 网络 171
 - 16.1 检查、监测网络 172
 - 16.1.1 ping——向网络主机发送特殊数据包 172
 - 16.1.2 traceroute——跟踪网络数据包的传输路径 173
 - 16.1.3 netstat——检查网络设置及相关统计数据 174
 - 16.2 通过网络传输文件 175
 - 16.2.1 ftp——采用ftp(文件传输协议)传输文件 175
 - 16.2.2 lftp——更好的ftp(文件传输协议) 177
 - 16.2.3 wget——非交互式网络下载工具 177
 - 16.3 与远程主机的安全通信 178
 - 16.3.1 ssh——安全登录远程计算机 178
 - 16.3.2 scp和sftp——安全传输文件 181
- 第17章 文件搜索 183
 - 17.1 locate——较简单的方式查找文件 184
 - 17.2 find——较复杂的方式查找文件 185
 - 17.2.1 test选项 186
 - 17.2.2 action选项 190
 - 17.2.3 返回到playground文件夹 194
 - 17.2.4 option选项 196
- 第18章 归档和备份 197
 - 18.1 文件压缩 198
 - 18.1.1 gzip——文件压缩与解压缩 198
 - 18.1.2 bzip2——牺牲速度以换取高质量的数据压缩 200
 - 18.2 文件归档 201
 - 18.2.1 tar——磁带归档工具 201
 - 18.2.2 zip——打包压缩文件 205
 - 18.3 同步文件和目录 207
 - 18.3.1 rsync——远程文件、目录的同步 207
 - 18.3.2 在网络上使用rsync命令 209
- 第19章 正则表达式 211
 - 19.1 什么是正则表达式 211
 - 19.2 grep——文本搜索 212
 - 19.3 元字符和文字 213
 - 19.4 任意字符 214
 - 19.5 锚 214
 - 19.6 中括号表达式和字符类 215
 - 19.6.1 否定 216
 - 19.6.2 传统字符范围 216
 - 19.6.3 posix字符类 217
 - 19.7 posix基本正则表达式和扩展正则表达式的比较 220

- 19.8 或选项 221
- 19.9 限定符 222
 - 19.9.1 ?——匹配某元素0次或1次 222
 - 19.9.2 *——匹配某元素多次或零次 222
 - 19.9.3 +——匹配某元素一次或多次 223
 - 19.9.4 {}——以指定次数匹配某元素 223
- 19.10 正则表达式的应用 224
 - 19.10.1 用grep命令验证号码簿 224
 - 19.10.2 用find查找奇怪文件名的文件 225
 - 19.10.3 用locate查找文件 226
 - 19.10.4 利用less和vim命令搜索文本 226
- 19.11 本章结尾语 227
- 第20章 文本处理 229
 - 20.1 文本应用程序 230
 - 20.1.1 文件 230
 - 20.1.2 网页 230
 - 20.1.3 电子邮件 230
 - 20.1.4 打印机输出 231
 - 20.1.5 程序源代码 231
 - 20.2 温故以求新 231
 - 20.2.1 cat——进行文件之间的拼接并且输出到标准输出 231
 - 20.2.2 sort——对文本行进行排序 232
 - 20.2.3 uniq——通知或省略重复的行 238
 - 20.3 切片和切块 239
 - 20.3.1 cut——删除文本行中的部分内容 239
 - 20.3.2 paste——合并文本行 242
 - 20.3.3 join——连接两文件中具有相同字段的行 243
 - 20.4 文本比较 245
 - 20.4.1 comm——逐行比较两个已排序文件 245
 - 20.4.2 diff——逐行比较文件 246
 - 20.4.3 patch——对原文件进行diff操作 248
 - 20.5 非交互式文本编辑 249
 - 20.5.1 tr——替换或删除字符 249
 - 20.5.2 sed——用于文本过滤和转换的流编辑器 251
 - 20.5.3 aspell——交互式拼写检查工具 258
 - 20.6 本章结尾语 260
 - 20.7 附加项 261
- 第21章 格式化输出 263
 - 21.1 简单的格式化工具 264
 - 21.1.1 nl——对行进行标号 264
 - 21.1.2 fold——将文本中的行长度设定为指定长度 266
 - 21.1.3 fmt——简单的文本格式化工具 267
 - 21.1.4 pr——格式化打印文本 270
 - 21.1.5 printf——格式化并打印数据 270
 - 21.2 文档格式化系统 273
 - 21.2.1 roff和tex家族 274
 - 21.2.2 groff——文档格式化系统 274
 - 21.3 本章结尾语 279
- 第22章 打印 281

- 22.1 打印操作简史 282
 - 22.1.1 灰暗时期的打印 282
 - 22.1.2 基于字符的打印机 282
 - 22.1.3 图形化打印机 283
- 22.2 linux方式的打印 284
- 22.3 准备打印文件 284
 - 22.3.1 pr——将文本文件转换为打印文件 285
- 22.4 向打印机发送打印任务 285
 - 22.4.1 lpr——打印文件(berkeley类型) 286
 - 22.4.2 lp——打印文件(system v类型) 287
 - 22.4.3 另外一个参数选项：a2ps 287
- 22.5 监测和控制打印任务 290
 - 22.5.1 lpstat——显示打印系统状态 290
 - 22.5.2 lpq——显示打印队列状态 291
 - 22.5.3 lprm与cancel——删除打印任务 291
- 第23章 编译程序 293
 - 23.1 什么是编译 294
 - 23.2 是不是所有的程序都需要编译 295
 - 23.3 编译一个c程序 295
 - 23.3.1 获取源代码 296
 - 23.3.2 检查源代码树 297
 - 23.3.3 生成程序 298
 - 23.3.4 安装程序 302
 - 23.4 本章结尾语 302
- 第四部分 编写shell脚本
- 第24章 编写第一个shell脚本 305
 - 24.1 什么是shell脚本 305
 - 24.2 怎样写shell脚本 306
 - 24.2.1 脚本文件的格式 306
 - 24.2.2 可执行权限 307
 - 24.2.3 脚本文件的位置 307
 - 24.2.4 脚本的理想位置 308
 - 24.3 更多的格式诀窍 309
 - 24.3.1 长选项名 309
 - 24.3.2 缩进和行连接 309
 - 24.5 本章结尾语 310
- 第25章 启动一个项目 311
 - 25.1 第一阶段：最小的文档 311
 - 25.2 第二阶段：加入一点数据 313
 - 25.3 变量和常量 314
 - 25.3.1 创建变量和常量 314
 - 25.3.2 为变量和常量赋值 316
 - 25.4 here文档 317
 - 25.5 本章结尾语 319
- 第26章 自顶向下设计 321
 - 26.1 shell函数 322
 - 26.2 局部变量 325
 - 26.3 保持脚本的运行 326
 - 26.4 本章结尾语 328

第27章 流控制：if分支语句	329
27.1 使用if	330
27.2 退出状态	330
27.3 使用test命令	332
27.3.1 文件表达式	332
27.3.2 字符串表达式	334
27.3.3 整数表达式	335
27.4 更现代的test命令版本	336
27.5 (())——为整数设计	338
27.6 组合表达式	339
27.7 控制运算符：另一种方式的分支	341
27.8 本章结尾语	342
第28章 读取键盘输入	343
28.1 read——从标准输入读取输入值	344
28.1.1 选项	346
28.1.2 使用ifs间隔输入字段	347
28.2 验证输入	349
28.3 菜单	350
28.4 本章结尾语	351
28.5 附加项	352
第29章 流控制：while和until循环	353
29.1 循环	353
29.2 while	354
29.3 跳出循环	356
29.4 until	357
29.5 使用循环读取文件	358
29.6 本章结尾语	358
第30章 故障诊断	359
30.1 语法错误	359
30.1.1 引号缺失	360
30.1.2 符号缺失冗余	360
30.1.3 非预期的展开	361
30.2 逻辑错误	362
30.2.1 防御编程	363
30.2.2 输入值验证	364
30.3 测试	364
30.3.1 桩	365
30.3.2 测试用例	365
30.4 调试	366
30.4.1 找到问题域	366
30.4.2 追踪	366
30.4.3 运行过程中变量的检验	368
30.5 本章结尾语	369
第31章 流控制：case分支	371
31.1 case	371
31.1.1 模式	373
31.1.2 多个模式的组合	374
31.2 本章结尾语	375
第32章 位置参数	377

- 32.1 访问命令行 377
 - 32.1.1 确定实参的数目 378
 - 32.1.2 shift——处理大量的实参 379
 - 32.1.3 简单的应用程序 380
 - 32.1.4 在shell函数中使用位置参数 381
- 32.2 处理多个位置参数 381
- 32.3 更完整的应用程序 383
- 32.4 本章结尾语 386
- 第33章 流控制：for循环 389
 - 33.1 for：传统shell形式 389
 - 33.2 for：c语言形式 392
 - 33.3 本章结尾语 393
- 第34章 字符串和数字 395
 - 34.1 参数扩展(parameter expansion) 395
 - 34.1.1 基本参数 396
 - 34.1.2 空变量扩展的管理 396
 - 34.1.3 返回变量名的扩展 397
 - 34.1.4 字符串操作 398
 - 34.2 算术计算和扩展 400
 - 34.2.1 数字进制 401
 - 34.2.2 一元运算符 401
 - 34.2.3 简单算术 401
 - 34.2.4 赋值 402
 - 34.2.5 位操作 404
 - 34.2.6 逻辑操作 405
 - 34.3 bc：一种任意精度计算语言 407
 - 34.3.1 bc的使用 407
 - 34.3.2 脚本例子 408
 - 34.4 本章结尾语 409
 - 34.5 附加项 409
- 第35章 数组 411
 - 35.1 什么是数组 411
 - 35.2 创建一个数组 412
 - 35.3 数组赋值 412
 - 35.4 访问数组元素 413
 - 35.5 数组操作 414
 - 35.5.1 输出数组的所有内容 415
 - 35.5.2 确定数组元素的数目 415
 - 35.5.3 查找数组中使用的下标 416
 - 35.5.4 在数组的结尾增加元素 416
 - 35.5.5 数组排序操作 416
 - 35.5.6 数组的删除 417
 - 35.6 本章结尾语 418
- 第36章 其他命令 419
 - 36.1 组命令和子shell 419
 - 36.1.1 执行重定向 420
 - 36.1.2 进程替换 420
 - 36.2 trap 422
 - 36.3 异步执行 425

36.4	命名管道	426
36.4.1	设置命名管道	427
36.4.2	使用命名管道	427
36.5	本章结尾语	428

《Linux命令行大全》

媒体关注与评论

诚实地讲，《Linux命令行大全》是我发现的最佳Linux入门指南，没有之一。—Linux Journal当读者认真学习完本书之日，也即成为UNIX命令行大师之时。—ITworld《Linux命令行大全》是最平易近人、易于阅读的Linux命令行大全。—Linux Magazine如果你是一名打算掌握命令行操作的新手，《Linux命令行大全》无疑是最佳的读物。—Ubuntu Musings《Linux命令行大全》是最适合Linux新手快速入门以及迅速提升的读物，它并非呆板地讲解这些命令行工具，而是会深入分析命令行的理论基础以及工作机制。对于想“知其然，知其所以然”的读者来说，本书是你的不二之选。—Nicholas C. Zakas, Web开发工程师，《Maintainable JavaScript》等畅销书作者

《Linux命令行大全》

编辑推荐

《Linux命令行大全》主要介绍Linux命令行的使用，循序渐进，深入浅出，引导读者全面掌握命令行的使用方法，涵盖全部的Linux核心命令，解读深奥的Linux详细参数，设计串联的Linux命令组合，跟踪可疑的Linux执行过程，提供丰富的Linux应用示例。《Linux命令行大全》适合从其他平台过渡到Linux的新用户和初级Linux服务器管理员阅读。没有任何Linux基础和Linux编程经验的读者，也可以通过本书掌握Linux命令行的使用方法。

精彩短评

- 1、比较不错的学习资料，很喜欢。
- 2、用linux已经有快差不多四年的时间了。平心而论，这真的是一本不错的书。内容很踏实，如果看英文的，会觉得更加原汁原味。书中大部分的命令是我们常用的命令。虽然不是讲得很全，但是已经够用了。
- 3、书不错 包装也还行就是这纸张有点次
- 4、过了一遍，感觉偏向手册一类
- 5、送货速度好快啊，刚开始看，还无法评价内容，书质量还是很好的。
- 6、严重批评下人邮的责编，书中印刷错误太多了，大小写，ln印刷成in，质量真差！
- 7、入门不错
- 8、本来想拿来重新复习下命令的.结果这书无论是从编排的顺序和内容的深度都差得太远.甚至不比久复烂名的21天XXXXXX系列.更要命的是,各种错误一堆堆的.强烈不建议购买!
- 9、可以拿来作为参考手册了，不过还是比较基础的命令介绍！
- 10、CLI 入门读物。读的是电子版本 说起来「快乐的 Linux 命令行」也明显比类似大全一类的 要动人很多吧...命令介绍蛮全面 侧重基础 也可做手册参考（不嫌厚的话...）。
- 11、我买了鸟哥和这本书，本来想把这本书作为补充，没想到这本书内容很不错。大家一定要仔细看，看几页就会发现一下就入门了。我认为是入门第一书
- 12、讲的不深但常用，不错，可以带去深圳当工具书
- 13、Linux 工具书，查漏补缺，有些章节略过。Shell那部分没看，准备读门佳译的《Linux Shell脚本攻略》。20160527
- 14、内容比较简单，入门级
- 15、很适合拿来入门。。。
- 16、指导性很强，而且文字精炼易懂！
- 17、学习了一些基本命令和shell 编程。
- 18、本来是要买参考手册的，结果买了本入门书，太基础了
- 19、shell脚本没看，有必要时再深入学习。的确是入门佳作，零基础也能看
- 20、这是一本不温不火的书，简单介绍了常用的命令，Bash 有别于其它编程语言的特质，以及基础脚本编写组件；值得作为初学者的入门书籍。它真的很简单，所以作为进阶的话还是选用别的书籍会比较好。
- 21、书 本身不错 但是翻译或排版有很多小错误
- 22、浅显易懂，抓住重点，适合作为一本简单的Linux使用参考书。
- 23、喜欢，实用，多练
- 24、比鸟哥之类的更适合作为新手入门使用，而且不仅仅只是罗列命令行，能够很顺畅的让人适应命令行方式，也很适合学完之后或学习之中的按目录查阅，推荐新手购买
- 25、用一天的时间翻完了，后面bash script的部分没看，毕竟不是做运维的。
讲的不错，满满的干货，鸟哥的书在bash命令写的并没有很全面，和这本结合起来看就很好了。
- 26、英文书是09年的，翻译差强人意，当然小错误还不少。哎，还是看不惯英文教程啊，还是得拿译文版的翻翻才高兴。话说总让我想起来那本已经基本绝迹的Python在Unix和Linux系统管理中的应用；拿ipython当shell用，一个月就基本心随意动了，但是单写shell，就是没有得心应手的感觉~
- 27、新手快速入门。内容挺全面的。
- 28、一般参考书，适合初学者
- 29、Shell脚本编程，作为命令行工具，非常强大。
- 30、内容不够详细，称不上是大全
- 31、第一版，所有里面有一些小错误。不过对于新手来说还是很不错的，短时间内把linux该知道的都讲解清楚了，最为一本入门书很合适。
- 32、相对基础...适合于0基础读者...
- 33、书86页的表错了，明显是从85页复制过来的。遇到不懂的地方最好在网上下载英文原版的电子书

《Linux命令行大全》

对照着。考虑到电子书不方便才买了纸质的。。。

34、粗略读完。略过Shell的地方了。要是写shell的时候可以看看。

35、写的深入浅出 感觉比鸟哥写的好

36、从基本入门开始 很适合初学者学习

37、今天刚刚看完，后边几章没有认真看

章节试读

1、《Linux命令行大全》的笔记-管道线

把 ls 命令的运行结果输送到文件 ls-output.txt 中去，由文件代替屏幕。

```
[me@linuxbox ~]$ ls -l /usr/bin &gt; ls-output.txt
```

如果我们需要删除一个文件内容（或者创建一个新的空文件），可以使用这样的技巧：

```
[me@linuxbox ~]$ &gt; ls-output.txt
```

怎样才能把重定向结果追加到文件内容后面，而不是从开头重写文件？为了这个目的，我们使用“>>”重定向符，像这样：

```
[me@linuxbox ~]$ ls -l /usr/bin &gt;&gt; ls-output.txt
```

重定向标准错误缺乏专用的重定向操作符。重定向标准错误，我们必须参考它的文件描述符。一个程序可以在几个编号的文件流中的任一个上产生输出。然而我们必须把这些文件流的前三个看作标准输入，输出和错误，shell 内部参考它们为文件描述符 0, 1 和 2，各自地。shell 提供了一种表示法来重定向文件，使用文件描述符。因为标准错误和文件描述符 2 一样，我们用这种表示法来重定向标准错误：

```
[me@linuxbox ~]$ ls -l /bin/usr 2&gt; ls-error.txt
```

可能有这种情况，我们希望捕捉一个命令的所有输出到一个文件。为了完成这个，我们必须同时重定向标准输出和标准错误。有两种方法来完成任务。第一个，传统的方法，在旧版本 shell 中也有效：

```
[me@linuxbox ~]$ ls -l /bin/usr &gt; ls-output.txt 2&gt;&amp;1
```

现在的 bash 版本提供了第二种方法，更精简合理的方法来执行这种联合的重定向。

```
[me@linuxbox ~]$ ls -l /bin/usr &amp;&gt; ls-output.txt
```

有时候“沉默是金”，我们不想要一个命令的输出结果，只想把它们扔掉。这种情况尤其适用于错误和状态信息。系统为我们提供了解决问题的方法，通过重定向输出结果到一个特殊的叫做“/dev/null”的文件。这个文件是系统设备，叫做位存储桶，它可以接受输入，并且对输入不做任何处理。为了隐瞒命令错误信息，我们这样做：

```
[me@linuxbox ~]$ ls -l /bin/usr 2&gt; /dev/null
```

cat 经常被用来显示简短的文本文件。因为 cat 可以接受不只一个文件作为参数，所以它也可以用来把文件连接在一起。比方说我们下载了一个大型文件，这个文件被分离成多个部分（USENET 中的多媒体文件经常以这种方式分离），我们想把它们连起来。如果文件命名为：

```
movie.mpeg.001 movie.mpeg.002 ... movie.mpeg.099
```

我们能用这个命令把它们连接起来：

```
cat movie.mpeg.0* &gt; movie.mpeg
```

因为通配符总是以有序的方式展开，所以这些参数会以正确顺序安排。

创建简短的文本文件。

输入命令，其后输入要放入文件中的文本。记住，最后输入 Ctrl-d。通过使用这个命令，我们实现了世界上最低能的文字处理器！看一下运行结果，我们使用 cat 来复制文件内容到标准输出：

```
[me@linuxbox ~]$ cat lazy_dog.txt
```

```
The quick brown fox jumped over the lazy dog.
```

管道线

命令可以从标准输入读取数据，然后再把数据输送到标准输出，命令的这种能力被一个 shell 特性所利用，这个特性叫做管道线。使用管道操作符 “|”（竖杠），一个命令的标准输出可以管道到另一个命令的标准输入：

```
command1 | command2
```

过滤器

管道线经常用来对数据完成复杂的操作。有可能会把几个命令放在一起组成一个管道线。通常，以这种方式使用的命令被称为过滤器。过滤器接受输入，以某种方式改变它，然后输出它。第一个我们想试验的过滤器是 sort。想象一下，我们想把目录/bin 和/usr/bin 中的可执行程序都联合在一起，再把它们排序，然后浏览执行结果：

```
[me@linuxbox ~]$ ls /bin /usr/bin | sort | less
```

uniq - 报道或忽略重复行

uniq 命令经常和 sort 命令结合在一起使用。uniq 从标准输入或单个文件名参数接受数据有序列表（详情查看 uniq 手册页），默认情况下，从数据列表中删除任何重复行。所以，为了确信我们的列表中不包含重复句子（这是说，出现在目录/bin 和/usr/bin 中重名的程序），我们添加 uniq 到我们的管道线中：

```
[me@linuxbox ~]$ ls /bin /usr/bin | sort | uniq | less
```

在这个例子中，我们使用 uniq 从 sort 命令的输出结果中，来删除任何重复行。如果我们想看到重复的数据列表，让 uniq 命令带上 “-d” 选项，就像这样：

```
[me@linuxbox ~]$ ls /bin /usr/bin | sort | uniq -d | less
```

wc（字计数）命令是用来显示文件所包含的行，字和字节数。例如：

```
[me@linuxbox ~]$ wc ls-output.txt
7902 64566 503634 ls-output.txt
```

“-l” 选项限制命令输出只能报道行数。添加 wc 到管道线来统计数据，是个很便利的方法。查看我们的有序列表中程序个数，我们可以这样做：

```
[me@linuxbox ~]$ ls /bin /usr/bin | sort | uniq | wc -l
2728
```

grep 是个很强大的程序，用来找到文件中的匹配文本。这样使用 grep 命令：
grep pattern [file...]

当 grep 遇到一个文件中的匹配 “模式”，它会打印出包含这个类型的行。grep 能够匹配的模式可以很复杂，但是现在我们把注意力集中在简单文本匹配上面。在后面的章节中，我们将会研究高级模式，叫做正则表达式。

grep 有 - 对方便的选项：“-i” 导致 grep 忽略大小写当执行搜索时（通常，搜索是大小写敏感的），“-v” 选项会告诉 grep 只打印不匹配的行。

有时候你不需要一个命令的所有输出。可能你只想要前几行或者后几行的输出内容。head 命令打印文件的前十行，而 tail 命令打印文件的后十行。默认情况下，两个命令都打印十行文本，但是可以通过 “-n” 选项来调整命令打印的行数。

tail 有一个选项允许你实时的浏览文件。当观察日志文件的进展时，这很有用，因为它们同时在被写入。在以下的例子里，我们要查看目录/var/log 里面的信息文件。在一些 Linux 发行版中，要求有超级

用户权限才能阅读这些文件，因为文件/var/log/messages 可能包含安全信息。

```
[me@linuxbox ~]$ tail -f /var/log/messages  
Feb 8 13:40:05 twin4 dhclient: DHCPACK from 192.168.1.1
```

....

使用 ” -f ” 选项，tail 命令继续监测这个文件，当新的内容添加到文件后，它们会立即 出现在屏幕上。这会一直继续下去直到你输入 Ctrl-c。

2、《Linux命令行大全》的笔记-如何查看命令名相关的帮助

type – 说明怎样解释一个命令名
which – 显示会执行哪个可执行程序
man – 显示命令手册页
apropos – 显示一系列适合的命令
info – 显示命令 info
whatis – 显示一个命令的简洁描述
alias – 创建命令别名

到底什么是命令？

命令可以是下面四种形式之一：

1. 是一个可执行程序，就像我们所看到的位于目录/usr/bin 中的文件一样。属于这一类的程序，可以编译成二进制文件，诸如用 C 和 C++ 语言写成的程序，也可以是由脚本语言写成的程序，比如说 shell , perl , python , ruby , 等等。

2. 是一个内建于 shell 自身的命令。bash 支持若干命令，内部叫做 shell 内部命令 (builtins)。例如，cd 命令，就是一个 shell 内部命令。

3. 是一个 shell 函数。这些是小规模的 shell 脚本，它们混合到环境变量中。在后续的章节里，我们将讨论配置环境变量以及书写 shell 函数。但是现在，仅仅意识到它们的存在就可以了。

4. 是一个命令别名。我们可以定义自己的命令，建立在其它命令之上。

识别命令

这经常很有用，能确切地知道正在使用四类命令中的哪一类。Linux 提供了一对方法来弄明白命令类型。

type - 显示命令的类型

type 命令是 shell 内部命令，它会显示命令的类别，给出一个特定的命令名（做为参数）。它像这样工作：

```
type command
```

“ command ” 是你要检测的命令名。这里有些例子：[me@linuxbox ~]\$ type type

```
type is a shell builtins
```

```
[me@linuxbox ~]$ type ls
```

```
ls is aliased to `ls --color=tty`
```

```
[me@linuxbox ~]$ type cp
```

```
cp is /bin/cp
```


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.tushu111.com