

北京科海培训中心

(美)Uresh Vahalia 著

聊鸿斌 曲广之 王元鹏 等译



清华大学出版社

E HALL
RENTICE HA
HALL PRENTICE
RENTICE HALL PRENTICE
HALL PRENTICE HALL PRE
HALL PRENTICE HALL PRE
HALL PRENTICE HALL

Prentice Hall

北京科海培训中心

UNIX 高级教程

系统技术内幕

(美) Uresh Vahalia 著

聊鸿斌 曲广之 王元鹏 等译

清华大学出版社

序

UNIX 的种类可能比你常见的冰淇淋品牌还要多。除了业界的 X/Open 及其成员在推动它的发展外,UNIX 本身远比我们所知的更为无所不在。事实上,这倒并不是一个非常重要的目标。自从 Interactive Systems 公司首次出售商业版 UNIX,Whitesmiths 公司首次推出 UNIX 克隆以来,广大 UNIX 用户就已经面对着运行不同平台上的各种不同的 UNIX。

1969 年诞生的 UNIX 开始流行时大约是 1978 年。在其又发展的 20 年中,出现了竞争财团的(如 OSF 和 UNIX 国际)众多的 UNIX 版本。这些系统分为两大流派,AT&T 公司(现在 Novell 购买了整个 UNIX 实验室)和加州伯克利分校。Maurice Bach[Bach 86]以及 Sam Leffler, Kirk McKusick, Mike Karels 和 John Quarterman[Leff 89]的书中详细介绍了这两大类 UNIX。

现在还没有一本书向感兴趣的读者介绍各种 UNIX 操作系统实现的一个整体面貌。Uresh Vahalia 现在满足读者的这一要求。他比以往更进了一步讲解了 SVR4, 4.4BSD 和 Mach 等系统的许多内部细节。此外还详细地分析了 Solaris 和 SunOS, Digital UNIX, 以及 HP-UX。

他的讲解十分清晰,并没有某些作者对这种或那种 UNIX 的偏爱。像 Linux 之类的较新的 UNIX 克隆已经产生了许多变体,即便是伯克利的派生系统也不尽相同。面对这样的事实,本书介绍了促成 UNIX 成长与流行的来龙去脉,可以说它是一本难得的好书。

1972 年 6 月 12 日, Ken Thompson 和 Dennis Ritchie 出版了《UNIX 程序员手册》第 2 版。在此书前言中,作者写到:“已有 10 部系统安装了 UNIX,预期会更多”。但他们根本没想到到底发生了什么。

我在[Salu 94]对 UNIX 系统的历史进行了详细介绍,而 Vahalia 则在本书中对各种 UNIX 系统进行了独到而有见地的剖析。

“计算机系统”杂志主编

Peter H. Salus

参 考 文 献

- [Bach 86] Bach, M. J. , The Design of the UNIX Operating System, Prentice-Hall, Englewood Cliffs, NJ, 1986.
- [Leff 89] Leffler, S. J. , McKusick, M. K. , Karels, M. J. , and Quarterman, J. S. , The Design and Implementation of the 4.3 BSD UNIX Operating System, Addison-Wesley, Reading, MA, 1989.
- [Salu 94] Salus, P. H. , A Quarter Century of UNIX, Addison-Wesley, Reading, MA, 1994.
- [Thom 72] Thompson, K. , and Ritchie, D. M. , UNIX Programmer's Manual, Second Edition, Bell Telephone Laboratories, Murray Hill, NJ, 1972.

前 言

自 20 世纪 70 年代初以来,UNIX 已经发生了巨大的变化。最初它是由贝尔电话实验室免费发行的一个小型的试验性的操作系统,而现在它已有了一大群不断扩大的追随者。20 多年来,UNIX 吸收了来自学术界和业界的奇思妙想,经历了所有权和标准的大战,逐渐成长为现在这样一个稳定而成熟的操作系统。今天有许多商用的和研究性的 UNIX 系统变体。它们彼此在许多方面不尽相同,但它们又都如此相似,只能算是同一家族中的不同成员。一个在某种 UNIX 系统上经验丰富的程序员可以毫无阻碍地胜任不同平台、不同 UNIX 变体的开发工作。

现在已有了很多介绍 UNIX 系统各种特性的书籍。其中大部分都是介绍用户可见的部分,比如命令 shell 或编程接口等。很少有几本书介绍 UNIX 的内部细节。这里,UNIX 的内部细节是指对 UNIX 核心的研究,讲解操作系统的的核心部分。目前,每一本关于 UNIX 内部细节的书都只关注于某一个特定的 UNIX 版本。Bach 的《UNIX 操作系统设计》[Bach 86]是关于 System V Release 2(SVR2)核心的权威性论著。Leffler 等人的《4.3BSD UNIX 操作系统设计与实现》[Leff 89]是 4.3BSD 的主要设计者对该系统的核心的全面讲解。Goodheart 和 Cox 的《魔法花园探秘》[Good 94]讲解了 System V Release 4.0(SVR4)的内部细节。

设计透视

本书从系统设计的角度讲述 UNIX 核心。介绍了大量的商用的和研究性的 UNIX 变体。对每一个核心部件,本书首先探究其结构和设计,大部分的 UNIX 系统是如何实现这些部件的,以及各种方案的优点和缺点。这种比较方法是本书的独到之处,它使读者以一种审视的眼光考察系统。在研究一个操作系统时,最重要的是既注意到它的优点又注意到它的缺点。这就必须分析和比较各种不同的实现。

UNIX 变体

本书以 SVR4.2 为主体,同时也详细介绍了 4.4BSD, Solaris 2.x, Mach 和 Digital UNIX。此外,还介绍其他许多变体的有趣的特征,包括那些没有加入到商业系统中的研究性功能。本书分析了 20 世纪 80 年代中期到 90 年代中期 UNIX 的主要发展过程。为了给读者一个更完整的讲解,本书还简要介绍了传统 UNIX 的功能和实现。在必要时,我们还从历史的角度首先介绍传统的方法,分析它的缺点和局限性,然后再给出现代方案。

读者对象

本书适于作高年级本科生和研究生的操作系统课程的教材,也可用作专业人员参考手册。本书不是一本入门教材,读者需要了解核心、进程以及虚存等概念。在每章最后都有练习,启发读者进行进一步的思考和研究,有助于加深了解系统设计。练习中的许多问题都是

可以自由回答的,有些问题读者需要进一步阅读文献。每章中都有非常详尽的参考文献,方便读者查找阅读。

本书也适合于作为操作系统开发人员、编写应用程序人员以及系统管理员的参考手册。操作系统设计人员可以利用本书研究当代系统的核心结构,比较不同设计间的优劣,了解下一代操作系统的开发。编写应用程序人员可以利用这些系统的细节开发出更高效的程序,和更好地利用操作系统的特点。最后,系统管理员在了解各种不同的参数和使用模式是如何影响系统行为之后可以更好地配置和优化系统。

本书的组织结构

第1章,“简介”,追溯 UNIX 系统的发展,分析影响系统的重大修改的各种因素。从第2章到第7章介绍进程子系统。其中,第2章介绍传统 UNIX 系统(SVR3,4.3BSD 以及更早的版本)的进程和核心结构。其中,第3章到第7章介绍 SVR4,4.4BSD,Solaris 2.x 以及 Digital UNIX 等现代 UNIX 系统的特征。其中第3章讨论线程以及如何在核心和用户库中实现线程。第4章介绍信号,作业控制以及登录会话管理等。第5章介绍 UNIX 调度器以及 UNIX 对实时应用的支持。第6章介绍进程间通信(IPC),包括 System V IPC。本章也介绍 Mach 体系结构,Mach 使用 IPC 做为构建核心的基本原语。第7章讨论用于现代单处理机和多处理机的同步框架。

第8章到第11章是文件系统。其中第8章介绍用户可见的文件系统接口以及核心与文件系统之间交互的 vnode/vfs 接口。第9章介绍某些特定文件系统的实现细节,包括最初的 System V 文件系统(s5fs),伯克利快速文件系统(FFS)以及许多利用 vnode/vfs 接口来提供服务的特殊文件系统。第10章介绍几种分布式文件系统,有 Sun Microsystems 的网络文件系统(NFS),AT&T 的远程文件共享(RFS),卡内基·梅隆大学的 Andrew 文件系统(AFS)以及 Transarc 公司的分布式文件系统(DFS)。第11章介绍利用日志技术的几种具有更高的可用性和性能的先进文件系统以及一种基于堆栈式 v 节点层的新型文件系统框架。

第12章到第15章介绍内存管理。第12章讨论核心内存分配和几种分配算法。第13章介绍虚存概念并以 4.3BSD 实现为实例谈论若干问题。第14章介绍 SVR4 和 Solaris 的虚存体系结构。第15章介绍 Mach 和 4.4BSD 的内存模型。此外,还讨论了快表、虚地址缓存等硬件设施对系统的影响。

最后两章讨论 I/O 系统问题。其中,第16章介绍设备驱动程序框架,核心与 I/O 系统之间的交互以及 SVR4 的设备驱动程序接口/驱动程序核心(DDI/DDK)接口规范。第17章介绍用于编写网络协议、网络和终端驱动程序的 STREAMS 框架。

尽管我们尽了最大努力,但难免会有错误,请将你的修改、意见及建议发送给我们,E-Mail 地址为:vahala@acm.org。

参考文献

- [Bach 86] Bach, M. J., The Design of the UNIX Operating System, Prentice-Hall, 1986.
- [Bapat 94] Bapat, S. G., Object-Oriented Networks, Prentice-Hall, 1994.

- [Good 94] Goodheart, B. , and Cox, J. , The Magic Garden Explained--The Internals of UNIX System V Release 4, An Open Systems Design, Prentice-Hall, 1994.
- [Leff 89] Leffler, S. J. , McKusick, M. K. , Karels, M. J. , and Quarterman, J. S. , The Design and Implementation of the 4.3 BSD UNIX Operating System, Addison-Wesley, MA, 1989.

译者说明

从 UNIX 诞生至今,UNIX 已经走过了 30 个年头。这期间 UNIX 发生许多重大的变化,有繁荣,也有坎坷。但不管怎样,UNIX 以其特有的简洁性和开放性获得越来越多的人的赞同和青睐。事实上,UNIX 已远远超出了一个操作系统软件所起的作用,它潜移默化地在方方面面影响着人们对软件的设计和开发,它所蕴含的软件设计思想仍然是十分值得借鉴的。

本书对 UNIX 核心设计和实现进行了详细地介绍,包括进程管理,文件系统,内存管理,外部设备等等。更难能可贵的是,著者以其丰富的经验针对不同的功能模块,通过比较不同系统的设计和实现向我们展示了软件设计的艺术性。

全书的翻译工作是由聊鸿斌(第 1,12,13,14,15 章),曲广之(第 4,5,6,7 章),王元鹏(第 8,9,10,11 章),靳若明(第 2,3 章)和郑明(第 16,17 章)完成的。由于时间匆促,能力有限,难免有不尽如人意的地方,请多多指教。

目 录

第 1 章 简介	(1)
1.1 简介	(1)
1.1.1 简史	(1)
1.1.2 创始之初	(1)
1.1.3 繁衍	(2)
1.1.4 BSD	(3)
1.1.5 System V	(4)
1.1.6 商业化	(4)
1.1.7 Mach	(5)
1.1.8 标准	(5)
1.1.9 OSF 和 UI	(6)
1.1.10 SVR4 及其之后	(7)
1.2 演变的动力	(7)
1.2.1 功能	(7)
1.2.2 网络	(8)
1.2.3 性能	(8)
1.2.4 硬件变化	(9)
1.2.5 改进质量	(9)
1.2.6 模式变化	(9)
1.2.7 其他应用领域	(10)
1.2.8 简洁就是美	(10)
1.2.9 灵活性	(11)
1.3 回顾与展望	(12)
1.3.1 UNIX 好在哪里	(12)
1.3.2 UNIX 的误区在哪儿	(13)
1.4 本书的范围	(14)
1.5 参考文献	(15)
第 2 章 进程与内核	(17)
2.1 简介	(17)
2.2 模式,空间和上下文	(19)
2.3 进程抽象	(20)
2.3.1 进程状态	(21)
2.3.2 进程上下文	(22)
2.3.3 用户凭证	(23)
2.3.4 u 区和 proc 结构	(24)
2.4 内核态下运行	(25)
2.4.1 系统调用接口	(26)

2.4.2 中断处理	(27)
2.5 同步	(29)
2.5.1 阻塞操作	(30)
2.5.2 中断	(30)
2.5.3 多处理器	(32)
2.6 进程调度	(32)
2.7 信号	(33)
2.8 新进程和程序	(33)
2.8.1 fork 和 exec	(34)
2.8.2 进程创建	(35)
2.8.3 fork 优化	(35)
2.8.4 执行一个新程序	(36)
2.8.5 进程终止	(37)
2.8.6 等待进程终止	(38)
2.8.7 僵尸(Zombie)进程	(39)
2.9 小结	(39)
2.10 练习	(39)
2.11 参考文献	(40)
第3章 线程和轻量级进程	(41)
3.1 简介	(41)
3.1.1 动机	(41)
3.1.2 多线程和多处理器	(42)
3.1.3 并发和并行	(44)
3.2 基本抽象概念	(44)
3.2.1 内核线程	(45)
3.2.2 轻量级进程	(45)
3.2.3 用户线程	(47)
3.3 轻量级进程设计——要考虑的问题	(50)
3.3.1 fork 的语义	(50)
3.3.2 其他的系统调用	(51)
3.3.3 信号传递和处理	(51)
3.3.4 可视性	(52)
3.3.5 堆栈增长	(52)
3.4 用户级线程库	(53)
3.4.1 编程接口	(53)
3.4.2 线程库的实现	(53)
3.5 调度器调用	(54)
3.6 Solaris 和 SVR4 的多线程处理	(55)
3.6.1 内核线程	(56)
3.6.2 轻量级进程的实现	(56)
3.6.3 用户线程	(57)
3.6.4 用户线程的实现	(58)

3.6.5 中断处理	(59)
3.6.6 系统调用处理	(59)
3.7 Mach 中的线程	(60)
3.7.1 Mach 的抽象概念——任务和线程	(60)
3.7.2 Mach 的 C-threads	(61)
3.8 Digital UNIX	(62)
3.8.1 UNIX 接口	(62)
3.8.2 系统调用和信号	(64)
3.8.3 pthreads 线程库	(64)
3.9 Mach 3.0 的续体	(65)
3.9.1 编程模型	(65)
3.9.2 使用续体	(66)
3.9.3 优化	(67)
3.9.4 分析	(68)
3.10 小结	(68)
3.11 练习	(68)
3.12 参考文献	(69)
第 4 章 信号和会话管理	(72)
4.1 简介	(72)
4.2 信号生成和处理	(72)
4.2.1 信号处理	(73)
4.2.2 信号生成	(75)
4.2.3 典型情景	(76)
4.2.4 睡眠和信号	(76)
4.3 不可靠信号	(77)
4.4 可靠的信号	(78)
4.4.1 主要特性	(78)
4.4.2 SVR3 的实现	(79)
4.4.3 BSD 信号管理	(80)
4.5 SVR4 信号机制	(81)
4.6 信号机制的实现	(82)
4.6.1 信号生成	(82)
4.6.2 信号传递和处理	(83)
4.7 异常	(83)
4.8 Mach 中的异常处理	(84)
4.8.1 异常端口	(84)
4.8.2 错误处理	(85)
4.8.3 调试器的交互	(86)
4.8.4 分析	(86)
4.9 进程组和终端管理	(87)
4.9.1 基本概念	(87)
4.9.2 SVR3 模型	(87)

4.9.3 局限性	(89)
4.9.4 4.3BSD 中的进程组和终端	(89)
4.9.5 缺点	(91)
4.10 SVR4 会话的体系结构	(91)
4.10.1 目的(动机)	(92)
4.10.2 会话和进程组	(92)
4.10.3 数据结构	(93)
4.10.4 控制终端	(94)
4.10.5 4.4BSD 中会话的实现	(95)
4.11 小结	(95)
4.12 练习	(95)
4.13 参考文献	(97)
第 5 章 进程调度	(98)
5.1 简介	(98)
5.2 时钟中断处理	(99)
5.2.1 调出链表	(99)
5.2.2 报警	(101)
5.3 调度器的目标	(101)
5.4 传统的 UNIX 调度	(102)
5.4.1 进程优先级	(103)
5.4.2 调度器的实现	(104)
5.4.3 运行队列管理	(105)
5.4.4 分析	(106)
5.5 SVR4 的调度器	(106)
5.5.1 类无关层	(107)
5.5.2 调度类的接口	(108)
5.5.3 分时类	(110)
5.5.4 实时类	(111)
5.5.5 系统调用 <code>priocntl</code>	(112)
5.5.6 分析	(113)
5.6 Solaris 2.x 调度的改善	(114)
5.6.1 抢占式内核	(114)
5.6.2 多处理器的支持	(114)
5.6.3 隐式调度	(115)
5.6.4 优先级逆转	(116)
5.6.5 优先级继承的实现	(118)
5.6.6 优先继承的局限性	(120)
5.6.7 Turnstiles	(120)
5.6.8 分析	(121)
5.7 mach 中的调度	(121)
5.7.1 多处理器的支持	(122)
5.8 Digital UNIX 的实时调度器	(124)

5.8.1 多处理器支持	(124)
5.9 其他的一些调度实现	(125)
5.9.1 fair-share 调度	(125)
5.9.2 最终期限驱动调度	(125)
5.9.3 三级(Three-Level)调度器	(126)
5.10 小结	(127)
5.11 练习	(127)
5.12 参考文献	(128)
第6章 进程间通信	(130)
6.1 简介	(130)
6.2 通用 IPC 方法	(130)
6.2.1 信号	(131)
6.2.2 管道	(131)
6.2.3 SVR4 的管道	(133)
6.2.4 进程跟踪	(133)
6.3 System V 的进程间通信	(135)
6.3.1 公共元素	(135)
6.3.2 信号量	(136)
6.3.3 消息队列	(139)
6.3.4 共享内存	(141)
6.3.5 讨论	(143)
6.4 Mach IPC	(143)
6.4.1 基本概念	(144)
6.5 消息	(145)
6.5.1 消息的数据结构	(146)
6.5.2 消息传递接口	(147)
6.6 端口	(148)
6.6.1 端口名字空间	(148)
6.6.2 端口数据结构	(148)
6.6.3 端口变换	(149)
6.7 消息传递	(149)
6.7.1 端口权力的传递	(150)
6.7.2 脱机内存	(152)
6.7.3 控制流	(154)
6.7.4 通知	(154)
6.8 端口操作	(155)
6.8.1 释放一个端口	(155)
6.8.2 备份端口	(155)
6.8.3 端口集合	(155)
6.8.4 端口的添加	(157)
6.9 扩展性	(158)
6.10 Mach 3.0 的改进	(159)

6.10.1 一次发送权	(159)
6.10.2 Mach 3.0 的通知	(160)
6.10.3 发送权的用户引用记数	(160)
6.11 讨论	(160)
6.12 小结	(161)
6.13 练习	(161)
6.14 参考文献	(162)
第 7 章 同步和多处理器	(164)
7.1 简介	(164)
7.2 传统 UNIX 内核中的同步	(165)
7.2.1 中断屏蔽	(165)
7.2.2 睡眠和唤醒	(165)
7.2.3 传统方法的局限性	(166)
7.3 多处理器系统	(167)
7.3.1 内存模型	(167)
7.3.2 同步支持	(169)
7.3.3 软件体系结构	(170)
7.4 多处理器同步问题	(170)
7.4.1 唤醒丢失问题	(171)
7.4.2 巨群问题	(171)
7.5 信号灯	(172)
7.5.1 提供互斥访问的信号灯	(173)
7.5.2 使用的信号灯的事件等待	(173)
7.5.3 用于控制可计数资源的信号灯	(173)
7.5.4 信号灯的缺点	(174)
7.5.5 护卫	(174)
7.6 自旋锁	(175)
7.6.1 自旋锁的使用	(176)
7.7 条件变量	(176)
7.7.1 实现问题	(178)
7.7.2 事件	(178)
7.7.3 阻塞锁	(179)
7.8 读写锁	(179)
7.8.1 设计考虑	(179)
7.8.2 实现	(180)
7.9 引用计数	(181)
7.10 其他考虑	(182)
7.10.1 死锁避免	(182)
7.10.2 递归锁	(183)
7.10.3 阻塞还是自旋	(184)
7.10.4 锁什么	(184)
7.10.5 粒度和持续时间	(184)

7.11 例子分析	(185)
7.11.1 SVR 4.2/MP	(185)
7.11.2 Digital UNIX	(186)
7.11.3 其他实现	(187)
7.12 小结	(188)
7.13 练习	(188)
7.14 参考文献	(189)
第 8 章 文件系统接口和框架.....	(191)
8.1 简介	(191)
8.2 文件的用户接口	(191)
8.2.1 文件和目录	(192)
8.2.2 文件属性	(193)
8.2.3 文件描述符	(195)
8.2.4 文件 I/O	(197)
8.2.5 分散-聚集 I/O(Scatter-Gather I/O)	(197)
8.2.6 文件加锁	(198)
8.3 文件系统	(199)
8.3.1 逻辑磁盘	(200)
8.4 特殊文件	(200)
8.4.1 符号链接	(200)
8.4.2 管道和 FIFO	(202)
8.5 文件系统框架	(202)
8.6 vnode/vfs 体系结构	(203)
8.6.1 目标	(203)
8.6.2 设备 I/O 的经验	(204)
8.6.3 vnode/vfs 接口概述	(206)
8.7 实现概述	(207)
8.7.1 目标	(207)
8.7.2 v 节点和打开文件	(208)
8.7.3 v 节点	(209)
8.7.4 v 节点引用计数	(210)
8.7.5 vfs 对象	(211)
8.8 文件系统相关对象	(212)
8.8.1 每个文件的私有数据	(212)
8.8.2 vnodeops 向量	(213)
8.8.3 vfs 层中的文件系统相关部分	(214)
8.9 安装一个文件系统	(214)
8.9.1 虚拟文件系统转换	(215)
8.9.2 mount 的实现	(215)
8.9.3 VFS_MOUNT 处理	(216)
8.10 对文件的操作	(216)
8.10.1 路径名遍历	(216)

8.10.2	目录查找缓存	(217)
8.10.3	VOP-LOOKUP 操作	(218)
8.10.4	打开文件	(219)
8.10.5	文件 I/O	(219)
8.10.6	文件属性	(220)
8.10.7	用户凭证	(220)
8.11	分析	(220)
8.11.1	SVR4 实现的缺点	(221)
8.11.2	4.4 BSD 模型	(222)
8.11.3	OSF/1 方法	(223)
8.12	小结	(223)
8.13	练习	(224)
8.14	参考文献	(225)
第 9 章	文件系统实现	(227)
9.1	简介	(227)
9.2	System V 文件系统(s5fs)	(228)
9.2.1	目录	(228)
9.2.2	i 节点	(229)
9.2.3	超级块	(231)
9.3	s5fs 内核组织	(232)
9.3.1	内存 i 节点	(232)
9.3.2	i 节点查找	(233)
9.3.3	文件 I/O	(233)
9.3.4	i 节点的分配与回收	(234)
9.4	对 s5fs 的分析	(236)
9.5	伯克利快速文件系统(FFS)	(236)
9.6	硬盘结构	(237)
9.7	磁盘组织	(238)
9.7.1	块和碎片	(238)
9.7.2	分配策略	(239)
9.8	FFS 的增强功能	(240)
9.9	分析	(241)
9.10	临时文件系统	(242)
9.10.1	内存文件系统	(243)
9.10.2	tmpfs 文件系统	(243)
9.11	特殊目的文件系统	(244)
9.11.1	specfs 文件系统	(244)
9.11.2	/proc 文件系统	(245)
9.11.3	处理器文件系统	(246)
9.11.4	半透明文件系统	(247)
9.12	以往的磁盘缓存	(248)
9.12.1	基本操作	(248)

9.12.2 缓冲区头结构	(249)
9.12.3 优点	(249)
9.12.4 缺点	(250)
9.12.5 保证文件系统的一致性	(250)
9.13 小结	(251)
9.14 练习	(251)
9.15 参考文献	(252)
第 10 章 分布式文件系统	(255)
10.1 简介	(255)
10.2 分布式文件系统的一般特征	(255)
10.2.1 设计考虑	(256)
10.3 网络文件系统(NFS)	(256)
10.3.1 用户透视	(257)
10.3.2 设计目标	(258)
10.3.3 NFS 组成	(258)
10.3.4 无状态	(260)
10.4 协议族	(261)
10.4.1 扩展数据表示(XDR)	(261)
10.4.2 远程过程调用(RPC)	(262)
10.5 NFS 实现	(264)
10.5.1 控制流	(264)
10.5.2 文件句柄	(265)
10.5.3 Mount 操作	(265)
10.5.4 路径名查找	(266)
10.6 UNIX 语义	(266)
10.6.1 打开文件权限	(266)
10.6.2 删除打开文件	(267)
10.6.3 读和写	(267)
10.7 NFS 性能	(268)
10.7.1 性能瓶颈	(268)
10.7.2 客户端高速缓存	(268)
10.7.3 延迟写	(268)
10.7.4 重传高速缓存	(269)
10.8 专用 NFS 服务器	(271)
10.8.1 Auspex 功能性多处理器结构	(271)
10.8.2 IBM 的 HA-NFS 服务器	(272)
10.9 NFS 安全性	(273)
10.9.1 NFS 访问控制	(273)
10.9.2 UID 重新映射	(274)
10.9.3 root 重新映射	(274)
10.10 NFSv3	(275)
10.11 远程文件共享(RFS)文件系统	(276)