Sun公司核心技术丛书

Solaris 系统编程

Solaris Systems Programming

```
class MyAppletApplication extends JApplet
{
   public void init() { . . . }
   . . .
   public static void main(String args[])
   {
      AppletFrame frame = new AppletFrame(new MyAppletApplication
());
       frame.setTitle("MyAppletApplication");
       frame.setSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       frame.setVisible(true);
   }
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE preferences SYSTEM "http://java.sun.com/dtd/preferences.
dtd">
<preferences EXTERNAL_XML_VERSION="1.0">
  <root type="user">
<map/>
    <node name="com">
    <map/>
    <node name="horstmann">
    <map/>
    <node name="corejava">
    <map>
    <entry key="left" value="11"/>
```

（加）Rich Teer 著
云巅工作室 译

SOLARIS™

Sun microsystems

机械工业出版社
China Machine Press

Sun公司核心技术丛书

Solaris 系统编程

Solaris Systems Programming

（加）Rich Teer 著
云巅工作室 译

机械工业出版社
China Machine Press

本书对 Solaris 系统编程进行了详细介绍。主要内容包括：接口以及 UNIX 编程中的重要知识、Solaris 提供的 I/O 功能、进程和进程控制、进程间相互进行通信的工具、伪终端等。另外，本书还配备了适当的练习题，有助于读者加深对所学知识的理解。

本书覆盖面广，讲解透彻，示例丰富，可作为一本 Solaris 系统编程的独立参考书。

本书版权登记号：图字：01-2006-1171

图书在版编目（CIP）数据

Solaris 系统编程/（美）提尔（Teer, R.）著；云巅工作室译. －北京：机械工业出版社，2006.3

（Sun 公司核心技术丛书）

书名原文：Solaris Systems Programming

ISBN 7-111-18571-4

Ⅰ.S…　Ⅱ.①提…　②云…　Ⅲ.操作系统（软件），Solaris－程序设计　Ⅳ.TP316.89

中国版本图书馆 CIP 数据核字（2006）第 012458 号

机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码　100037）

责任编辑：华　章

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2006 年 3 月第 1 版第 1 次印刷

787mm×1020mm 1/16 · 54.25 印张

定价：99.00 元

译 者 序

从20世纪90年代开始，Sun公司的Solaris就成为市场上最受欢迎的UNIX操作系统，而且从Solaris 2.4开始，Sun公司就开始提供SCSA认证。如今，SCSA日渐流行，而且它是SCNA的基础。本书对Solaris系统编程进行了详细介绍，知识全面而又通俗易懂。本书的作者在Solaris编程领域具有渊博的知识，他的讲解方式也非常独特，往往能用简单的示例将复杂的概念解释透彻。

不仅对于Solaris程序员，而且对于UNIX程序员、C程序员来说，这都不失为一本必备参考书。甚至，国外许多Solaris程序员将本书视为Solaris系统编程的宝典。据称，作者为撰写此书，几乎用了三年半的全部时间来笔耕。从文字叙述上，也可见作者花费了大量心血。所以，读者对此书好评如潮就不足为奇了。

本书具有以下特点：

- 知识全面，讲解透彻，可作为一本Solaris系统编程的独立参考书。
- 对基础知识的叙述深入浅出，读者易于掌握。
- 本书附录提供了丰富的信息，是本书内容的有机补充。
- 本书配备了适当的练习题，读者通过这些练习题，可以加深对所学知识的理解。

参加本书翻译工作的人员包括：周良忠、刘旺、周依星、马城、齐有为、龚菊有、胡天文、胡军、李平一、徐大伟、夏傅、秦烈、高汉军、刘永光、欧阳光、隋伟业、陈盼、雷雄、唐四如、吴露、周正等。

由于本书内容涉及面广，译者水平有限、时间仓促，错误在所难免，希望广大读者不吝指正。联系E-mail：web_zhou@21cn.com。

译 者

2006年2月1日

前　言

本书讲述了 Solaris 操作系统的系统编程接口。尽管本书书名中使用了“Solaris”一词，但本书也适用于 UNIX或 UNIX 操作系统的程序员（也就是说，本书描述的某些特征是针对 Solaris 的）。

与大部分操作系统一样，Solaris 也为程序员提供了大量的服务。例如，打开文件、读取文件、分配内存、获取当前时间、启动新程序等。本书描述了许多公共接口，但是无论如何努力，也不可能详细讲解一切（面面俱到的书将冗长无比，而且永远也不可能写成这样的书）。

因为使用未提供说明文档的接口难以写出可移植的程序，所以本书不讲解这样的接口。同时，使用专有、未提供文档说明的接口缺少 Sun 的应用证书，因为这些接口可能在无告示的情况下随版本的变化而更改。

本书所描述的函数（接近 540 个）在 Solaris“Reference Manual Collection”的第二部分和第三部分提供了文档说明。不过，这些手册页没有提供背景材料和完整的实例，而本书正好提供了这些内容。

读者对象

本书既可作为初级和中级程序员的指导书，也可作为有经验程序员的参考书，它还适合作为本科生或研究生的编程课程的教材。

本书假定读者具备 C 编程的一些经验（但不一定需要 UNIX 平台上的编程经验），而且至少对 UNIX 的一些概念具有基本的了解，如与 shell 的交互、用文本编程器进行编辑和管道操作。

强烈建议读者使用可以调试本书示例的 Solaris 系统（但不作绝对要求）。

本书的组织方式

本书分为 6 部分：

- 第一部分：综述。这一部分提供了介绍性知识和历史背景信息，包括 2 章：
  - 第 1 章概述了许多基本 UNIX 编程概念和术语。
  - 第 2 章描述了 Solaris 的发展历史，还介绍了 Solaris 所兼容的若干不同标准。
- 第二部分：基本主题。这一部分讲解接口以及 UNIX 编程中的重要知识，包括 7 章：
  - 第 3 章介绍了用于操作字符类、字符字符串和字节数组的大部分实用工具函数，还介绍了动态内存、临时文件、解析命令行参数、错误报告以及进程挂起。许多读者可能熟悉其中的大部分材料，但它提供了我们起步的通用起点。
  - 第 4 章描述了低层次的文件 I/O，包括如何打开和关闭文件、如何读写文件以及如何更改当前文件偏移，还讨论了文件共享和缓存（使用这些函数的程序必须针对它们进行缓存）。
  - 第 5 章描述了高层次的 I/O 函数，它们由标准 I/O 库所提供。
  - 第 6 章描述了如何获取和设置系统时钟，以及在不同格式间转换时间的不同函数。
  - 第 7 章描述了用户、组和口令文件的格式，以及如何从这些文件获取信息。这一章还解释了如何判断登录系统的用户、用户最后一次登录或退出的时间。
  - 第 8 章描述了如何获取和设置不同系统和用户资源限制，如进程可以占用的 CPU 时间大小、每个进程可以打开的最大文件数量、系统主机名等信息、安装的内存大小等。
  - 第 9 章描述了程序中的若干常见安全性缺陷（如缓存溢出），还介绍了一些编写程序的技巧，这些技巧在设计上是安全的。
- 第三部分：输入/输出。这一部分描述了 Solaris 提供的 I/O 功能，包括 4 章：
  - 第 10 章描述了文件和目录的特征以及操作它们时可以使用的函数。这一章介绍了不同的文件类型

等概念，文件权限，解析符号性链接，创建、读取和删除目录，还介绍了设置用户 ID 和设置组 ID 程序。

- 第 11 章解释了如何读取磁盘上用于文件系统的数据结构、如何挂载和卸载以及如何读取挂载的文件系统表。
- 第 12 章讨论了终端 I/O，包括特殊输入字符、检查和修改终端属性。还介绍了用于操作串行的函数（如更改波特率和每个字符的位数）。
- 第 13 章介绍了更高级的 I/O 主题，如记录锁定、非阻塞、多路复用和异步 I/O。这一章还讨论了内存映射文件、访问控制表以及扩展的文件属性。

- 第四部分：进程和进程控制。这一部分介绍进程及进程控制的相关知识，包括 5 章：
  - 第 14 章描述了运行 UNIX 进程的环境，还介绍了进程的启动和终止。
  - 第 15 章介绍了如何创建新进程以及如何启动另一个运行的程序。这一章还介绍了如何等待进程的终止状态、如何避免竞争条件以及进程记账。
  - 第 16 章解释了会话、进程组和控制终端的概念。
  - 第 17 章描述了可靠和不可靠信号的概念，包括如何发送信号、捕获信号、阻塞信号以及忽略信号。
  - 第 18 章描述了守护进程的特征，并介绍了进程如何变成守护进程。还介绍了将消息记入日志的工具以及如何只启动守护进程的一个拷贝。
- 第五部分进程间通信。这一部分介绍进程间相互进行通信的工具，包括 4 章：
  - 第 19 章介绍了最古老但也许使用最频繁的进程间通信方法：管道和 FIFO。这一章还介绍了交互式和并发式服务器的区别。
  - 第 20 章介绍了 System V 消息队列、信号量集和共享内存片段。
  - 第 21 章介绍了向其他进程传递文件描述符的概念，其他进程既包括关联进程，也包括非关联进程。这一章还介绍了如何从文件系统中的文件名上附加和去除管道。
  - 第 22 章介绍了 Solaris 门，它用于同一主机上进程间的快速远程过程调用。
- 第六部分：伪终端。这一部分介绍伪终端，只包括 1 章：
  - 第 23 章介绍了什么是伪终端，以及如何创建伪终端。然后，介绍了如何通过伪终端执行其他程序，这对于脚本程序有用，否则，这些程序必须交互式运行。

本书还包括 5 个附录：

- 附录 A 简短介绍了国际化和本地化的相关知识。也就是介绍了文化区域的概念，以及如何编写可移植到不同语言和区域的程序。
- 附录 B 简要介绍了 BSD 源代码兼容性包。这个包是一个过渡性工具，它设计用于让编写的程序兼容基于 BSD 的 SunOS 4.x API。
- 附录 C 总结了本书介绍的所有函数。在介绍每个函数的函数原型（包括所有返回值）的同时，还介绍了每个函数的用途（针对 Solaris 版本和标准）。
- 附录 D 介绍了大部分示例程序中包含的头文件的源代码，还介绍了为在示例程序中使用而开发的库函数的源代码。
- 附录 E 提供了许多章末练习的答案。

源代码和勘误

本书中所有示例的源代码均可从作者的主页下载（网址参见后面）。学习本书所介绍的接口和技术的最好途径是编写和修改这些程序。实际编写代码（也许使用这些示例作为起点）是完全理解这些概念和技术的唯一途径。

本书中的所有示例均在多个系统（包括 SPARCstation 20、SPARCstation Voyager、几种 Ultra 1、Sun Blade

100 和 Ultra 60）上进行了测试，运行了若干不同的 Solaris 版本。它们使用 Sun 的 C 编译器（5.4 版本）和 gcc 的 2.95.2 版本进行编译。这些示例还在许多运行 Solaris x86 的系统上通过了测试。

现在，本书的勘误可从作者的主页下载。

致谢

感谢我的家庭，他们在三年多中付出了爱心和支持。感谢 Jenny 和 Judge（我们的宠物），他们陪伴我度过了漫长的写作时间。现在，本书定稿，我的妻子和儿女又可以与我共享天伦之乐了。还感谢家庭中的其他成员和朋友，在我写作期间，他们给予我支持和鼓励。感谢你们每一个人！

感谢如下技术审校人员，他们提供了有价值的反馈，找出了大量错误，指出了需要更加详述的地方，并对不同的措辞、阐述、代码提出了建议：Philip Brown、Alan Coopersmith、Casper Dik、Stefaan Eeckels、Peter Baer Galvin、Alexander Gelfenbain、Anthony Mandic、Chris Morgan、David Robinson。还要特别感谢 Dragan Cvetković，他审阅了整个手稿。

下列人员在百忙中回答我在电子邮件中提出的问题（有时所提问题不少），这一切均有助于提高本书的准确性和全面性：Dave Butenhof、Dennis Clarke、Alan Coopersmith、Casper Dik、Darren Dunham、Bill Fenner、Markus Gyger、Zhishun Alex Liu、Darren Moffat、Jim Moore、Alec Muffet、Greg Onufer、David Robinson、Karl Schendel、Andy Tanenbaum 和 Tony Walton。

特别要感谢 W. Richard Stevens，感谢他在本书构思之初给予的鼓励。也要感谢 Addison-Wesley 的 Michael Slaughter，是他最初与我进行联系，并促成了本书的付梓。

感谢 Bill Moffitt 和 Karen Hill，是他们安排我参加了 Solaris 9 Beta 课程。还要感谢 Sun Microsystems 公司提供了 SunONE Studio 7 Compiler Collection 的授权。

最后，感谢 Prentice Hall 的工作人员：Kathleen Caren、Raquel Kaplan，特别要感谢我的编辑 Greg Doench。他们对于我增加篇幅、多次延期以及“自由发挥”表现出了极大的耐心，非常感谢他们（作者是一位居住在加拿大的英国人，使用的是英式英语）。当然，还有很多需要感谢的人没有在此罗列出来。

后记

本书沿袭了使用 vi 和 troff 编写真正 UNIX 书籍的传统。在运行 Solaris 10 Build 60 的 Sun Ultra 60 上，我使用 James Clark 的优秀 groff 包生成可立即使用的 PostScript。我使用 vi 编辑器输入了所有 417 259[⊖]个字，使用 gpic 程序制作了 84 幅插图，使用 gtbl 程序生成了 91 个表，制作了所有索引（使用 Jon Bentley 和 Brian Kernighan 编写的 awk 脚本集），并且完成了最后的排版。我自己的脚本 c2ms、expand 程序、nl 实用工具以及其中一个 sed 脚本用于包含本书中源代码的 12 619 行（281 个程序）。

欢迎读者来信提出建议，指出错误。

电子邮件：rich.teer@rite-group.com

个人主页：http://www.rite-group.com/rich

Rich Teer

加拿大不列颠哥伦比亚省基洛纳市

2004 年 7 月

⊖ 这是指原英文书字符数。——编辑注

目　录

第三部分 输入/输出

第五部分 进程间通信

第六部分 伪 终 端

附 录

第一部分 综 述

第1章 Solaris 系 统

1.1 引言

本章将对 Solaris 系统编程中所使用的众多术语和特征进行基本介绍。因为如果不介绍那些未曾提及的特征，对于像 Solaris 这样的庞大系统，就几乎无法描述如何进行系统编程。在概述中简要解释的这些主题，将会在本书的其他章节进行更为详细的论述。

Solaris 是由 Sun Microsystems 公司开发的一个 UNIX 操作系统版本。最初，Solaris 仅能运行在 Sun 公司的 SPARC（Scalable Processor ARChitecture，可扩展处理器体系结构）处理器上，但目前它已被移植到了 Intel 公司的 x86 处理器及其兼容处理器上。Solaris 的一个版本（确切地说是指 Solaris 2.5.1）甚至移植到了 Apple、IBM 和 Motorola 公司生产的 PowerPC 处理器上。UNIX 操作系统本身是由 AT&T 贝尔实验室于 20 世纪 60 年代末 70 年代初所开发的。

尽管本书重点介绍 Solaris 2.5 到 Solaris 9，但是大多数内容都和 Solaris 的其他版本以及 UNIX 的其他实现相关，其中包括 Linux。

> 严格地说，一般通俗地称为“Solaris”的操作系统更确切的名称应该是“Solaris 操作环境”。Solaris 操作环境由若干个组件构成，例如，称为 SunOS 的内核以及视窗系统。目前，视窗系统基于 X 窗口系统（X Window System），可以使用 CDE（Common Desktop Environment，通用桌面环境）或者 GNOME（GNU Network Object Model Environment，GNU 网络对象模型环境）用户接口（Solaris 的早期版本使用 OpenWindows 用户接口，并且对于系统管理员安装其他可选用户接口没有任何限制，比如 KDE（K Desktop Environment，K 桌面环境）用户接口）。在本书中，除非需要作特别明确的区分，一般都仅称为“Solaris”。
>
> 在全书的正文中，我们采用类似本段文字的一些附加说明性的注释来补充更为详细的内容，或者给出一些历史参考资料。因为有些时候，只有明白了某个事物为什么以这种方式工作，才能更容易理解它的工作方式。

Solaris 是 UNIX 系统 V 版本号 4（SVR4）的后代，本书中的大部分材料都适用于这两者。Solaris 特有的细节作了标记，指示 SVR4 和 Solaris 之间的区别。Solaris 的历史将在第 2 章讲述。

Solaris 提供了一套应用程序编程接口（Application Programming Interface，API）和应用程序二进制接口（Application Binary Interface，ABI），它们在 C 编程语言中定义。因此，在本书中，我们使用 C 语言，不过，开发人员也可以使用其他语言，如 C++ 或 Java。但是，如果使用其他语言，一些细节可能会有所不同。

在 Solaris 上编译 C 程序

书中提供了许多示例程序。在运行它们之前，必须通过 C 编译器进行编译。除了实际的 C 编译器外，Solaris发布时提供了编译 C 程序所需的一切（即头文件、库和工具）。运行之前还必须安装下面的包（根据所使用的 Solaris 版本，并非存在下面的所有包）：

- 对于工具（如 as、ld、make、sccs 和 truss）：SUNWbtool，SUNWsprot 和 SUNWtoo。
- 对于库和头文件：SUNWhea，SUNWarc，SUNWlibm，SUNWlibms，SUNWxwinc，SUNWolinc 和 SUNWxglh。

- 对于64位开发：SUNWarcx，SUNWbtoox，SUNWdplx，SUNWscpux，SUNWsprox，SUNWtoox，SUNWlmsx，SUNWlmx和SUNWlibCx。

（确保安装所有必需包的最简单方式是在安装OS时至少选择“开发人员系统支持”软件组。）

对于Solaris，可用的两个最流行编译器是Sun的C编译器cc和GNU C编译器gcc。Sun的编译器是一款商业产品，因此，必须购买许可才能使用。而gcc是免费产品，但是，它生成的代码不如Sun的编译器生成的代码优良。gcc编译器可以从因特网上下载，或自Solaris 8开始，从Solaris软件附带光盘中安装它（Sun编译器的拷贝也包含在Solaris光盘包中，还可从Sun获得30天或60天免费试用版本）。

在/usr/ucb之前，PATH中必须存在目录/usr/ccs/bin（从PATH中一起删除/usr/ucb更好），而且安装编译器的目录也必须存在于PATH中。

有时，应用程序需要与不在默认Solaris目录中的库进行链接。进行链接时，需要指定如下编译器命令行标志：-L/path/to/lib和-R/path/to/lib。如果忽略第二个标志，程序将成功编译，但接着必须设置LD_LIBRARY_PATH来指向适当的目录。这样做不太方便，所以在使用-L时应该确保总是指定了-R（对于预编译二进制，不可能这样做，因此，使用设置了程序运行期间LD_LIBRARY_PATH的小型封装脚本是最佳解决之道）。

在编译代码时依赖于LD_LIBRARY_PATH也不理想，因为这样通常会导致开发人员由于错误的原因而使用它（例如，使用LD_LIBRARY_PATH的一个充足理由是正在开发一个新版本库，而尚未在系统库目录中安装它）。

真正的危险在于安装应用程序时尚未配置实时路径，而强迫用户设置LD_LIBRARY_PATH。一旦用户进行了全局设置，用户就开始依赖它，那么，当必须修改或删除LD_LIBRARY_PATH时，就会造成麻烦。

对于如何处理应用程序的共享库，请参考如下3条建议：

1）尽可能使用系统提供的共享库。

2）如果应用程序使用私有共享库，要确保未设置LD_LIBRARY_PATH（至少在最后生成时保证如此），还要指定正确的运行时库路径。

3）从Solaris 7开始，在库路径中使用$ORIGTIN（不要与shell环境变量混淆），创建不依赖于LD_LIBRARY_PATH的可重定位软件的工作变得更加容易，$ORIGTIN在运行时确定。假定将所有库放在lib子目录中，用-R$ORIGIN/../lib进行链接后可以在不产生其他破坏的情况下移动安装目录。

如果我们开发商业应用程序，这些建议尤其重要，因为用户通常对于修正错误无能为力。

1.2　登录

使用Solaris系统的第一件事情是登录。登录要在系统中标识用户，这一步将设置环境和权限。登录时，必须输入用户名和口令，或在传统的login：提示符下，或使用更新一些的图形化登录管理器（至少在控制台上如此。使用telnet或ssh的远程登录仍然存在传统界面，除非采取措施改变默认的行为）。

验证用户名和口令时，如果它们与存储在/etc/passwd和/etc/shadow中的用户名、口令匹配，则允许访问系统。除了支持传统/etc/passwd用户验证方法，Solaris还支持网络信息服务（NIS，以前称做黄页）、NIS+以及轻量级目录访问协议（Directory Access Protocol，LDAP）（从Solaris 8开始）。要搜索哪些数据库、按什么顺序来搜索，这些信息均在/etc/nsswitch.conf中定义。

得到验证后，设置用户ID和组ID，并将用户放入运行指定shell的主目录中。口令文件包含本机所有账号的列表（一行一个），或包含提供必需信息的其他名字服务。第7章将讨论允许询问和更新这些文件的函数。

1.3　shell

我们在命令行层次与Solaris交互的方式是使用shell。其他交互方法包括使用本地GUI或者远程操作（例

如，使用浏览器）。shell 是一种命令行解释器，它以独立的进程或内建命令的形式读取命令并试图执行。shell 也是编程语言，它有控制结构、变量、算术工具，支持正则表达式。以 shell 编程语言编写的程序称为 shell 脚本，本书不介绍它的写法。如果 shell 是交互式会话，那么命令由用户输入。命令也可以从包含 shell 脚本的文件中读取。

Solaris 提供 3 种 shell 作为标准，它们是：

- Bourne shell（sh）。
- C shell（csh）。
- Korn shell（ksh）。

从 Solaris 8 开始，Solaris 还支持开放源代码的 bash、tcsh 和 zsh 几个 shell，但是它们没有随着核心操作系统一起安装。上述 shell 和其他 shell 都可以从因特网上下载。

Bourne shell（/bin/sh）是以它的作者 Steve Bourne 命名的，Steve Bourne 在贝尔实验室开发了此 shell。从 Version 7 开始，它几乎随同 UNIX 的每个版本一起发布，Version 7 是贝尔实验室在 System III 之前发布的最后一个 UNIX 公共版本。

> UNIX 的早期版本是根据随着系统发布的手册的版本号命名的。

Bourne shell 的一个静态链接版本可以在/sbin/sh 中找到。默认情况下，这是超级用户 root 使用的 shell。不要试图把它改为功能更具体的 shell（如/bin/ksh），因为如果新的 shell 所依赖的任何库文件被删除或破坏了，root 用户就不能登录：这是当系统需要紧急维护时，我们想做的最后一件事。获得更具体 root shell 的一种更好方法是在 root 的 .profile 内运行我们选择的 shell：如果在运行中出现了问题，仍然可以使用/sbin/sh。

> 很多人认为可以改变 root 的 shell，并声称最坏的情况下需要从备用源（如 CD-ROM 或网络启动服务器）启动来恢复系统。这是正确的，但是静态链接 shell 还是更安全一些。这是因为它是自我包含的，而不是依赖几个动态库，这些动态库中的一些或全部可能由于某种原因而无法使用。静态编译我们选择的 shell，然后把它放到 root 文件系统也是一种选择（这是作者认为唯一的可接受的/sbin/sh 的替代方式），但是这好像有点事倍功半。
>
> 从 Solaris 9 开始，对于 root 的 shell 来说，Solaris 的灵活性更强：如果列在/etc/passwd 中的 shell 不能执行，/sbin/sh将作为故障保护措施被执行。然而，总的来说，把/sbin/sn 作为 root 的 shell 是最安全的选择。
>
> 从 Solaris 10 开始，Solaris 不再支持与系统库文件（包括/sbin/sh）静态链接的 32 位程序（Solaris 从来都不支持静态链接的 64 位可执行程序）。因此，不改变 root 的 shell 的技术原因已经变得不再重要。

Sun 公司的共同创始人之一 Bill Joy 在加利福尼亚大学伯克利分校时编写了 C shell——/bin/csh。C shell 基于第 6 版 shell（而不是 Bourne shell），它有一个类似于 C 编程语言的控制流（这也就是它的名称由来），并提供了 Bourne shell 不支持的特性——比如作业控制、历史机制和命令行编辑。尽管在交互式使用方面很好，但 C shell 的某些特性使其编写 shell 脚本时却不尽如人意（参见［Christiansen 1995］得到更多相关信息）。

Korn shell——/bin/ksh 被认为是 Bourne shell 的继承者，它是 David Korn 在贝尔实验室写成的。它具备所有 C shell 所具有的优良特性（作业控制、历史机制和命令行编辑），但它的语法是 Bourne shell 的超集，所以用后者写出的脚本也可以在 Bourne shell 下正常工作。

1.4 文件、目录和文件系统

UNIX 程序设计范例之一是每种东西均是一个文件。普通的文件是文件，目录是文件，所有的设备也是文件（设备称为特殊文件）。文件是一个无序的字节流。该字节流不以任何方式构造；所有的解释均留给应用软件（与此相反，其他一些操作系统给文件强加了一个面向记录的视图，或在二进制与非二进制文件之间加以区分）。

目录是由一个或多个用户组合在一起（创建一个层次结构）的文件以及其他目录的集合。目录是通过内核将文件标记为目录来实现的；该文件包含一个文件及其属性的列表。更确切地说，一个目录文件由一个条目（该目录下每个文件的名称）列表和它们对应的索引节点号组成。

索引节点（inode）是一个固定长度的数据结构（128 个字节），其中保存了与文件有关的绝大部分信息（例如所有者、权限、存取时间、数据块列表，以及文件类型和大小）。（在第 10 章讨论 stat、fstat 和 lstat 函数时，将更详细讨论这些文件属性。）

不在目录条目中保存文件属性的原因是为了灵活性：通过将两者分离，就可以让多个文件名指向同一个索引节点。这种安排也允许文件在文件系统中没有名称时仍然存在。这一点对于创建临时文件（当进程退出时该文件自动去除）很有用。只有删除文件索引节点的最后一个引用时，才释放用于该文件的磁盘块，除非某个进程已经打开了该文件。在这种情况下，只有当最后一个打开该文件的进程关闭它之后，才删除该文件。因此，通过创建、打开然后删除文件，创建了一个临时文件，当进程退出时，它自然而然地消失了。

每个目录至少包含两个文件名（任何时候创建新目录时，均自动创建它们）：“.”（被称作点）和“..”（被称作点－点）。前者指向当前目录，后者指向当前目录的父目录（有一个例外是根目录，它的点－点仍然指向它自己）。

同一逻辑设备上的文件和目录在文件系统中集中放置。在可以访问文件系统中的文件和目录之前，必须先通过挂载将它们附加到层次性文件树上。一般的 Solaris 系统在某个时刻会挂载若干文件系统。

Solaris 支持 3 种类型的文件系统：基于存储器（常规）、网络和伪文件系统。常规文件系统是为文件和目录提供永久性存储空间的文件系统，网络文件系统使远程服务器上存储的文件看起来像本地文件一样，伪文件系统是使用虚拟文件来表示不同抽象的文件系统。图 1-1 总结了不同的文件系统类型。

文件系统类型	文 件 类 型	示　例
常规文件系统	文件存储在永久性介质上	ufs、hsfs、tmpfs
网络文件系统	远程存储文件	nfs
伪文件系统	文件是某些内容的抽象	procfs、doorfs、fdfs

图 1-1　不同文件系统类型总结

文件名和路径名

许多函数都要有文件名或路径名作为它们的参数之一。文件名是包含一个或多个字符的名字，这些字符用于给文件命名。除斜杠（/）和 NUL 字符外，这些字符可以为任意字符。文件名有时作为路径名的一部分。路径名是包含一个或多个字符的字符串，这些字符用于标识某个文件。路径名具有一个可选的起始斜杠，后面是零个或多个文件名，文件名用斜杠分隔（多个连续斜杠看作一个斜杠）。为了比较这两个术语，让我们看看口令文件。口令文件的路径名是/etc/passwd，但它的文件名就是 passwd。在大部分讨论中，这两个术语的区别只是学术上的，实际上，我们会交替使用它们。

一些系统允许路径名以/结束，但有的系统不允许这样做；可移植程序因此不应该依赖于末尾斜杠的可行性。实际上，从 Solaris 10 开始，如果路径名末尾的斜杠不是指目录，那么引用这样的路径名将返回错误。

路径名有 2 种类型：绝对路径名和相对路径名。如果路径名以/起始，则称为绝对路径名，因为文件名是从根目录开始指定的；如果路径名不是以/起始，则称为相对路径名。

当处理文件名和路径名时，需要注意两个常量，即 MAXPATHLEN（在＜sys/param.h＞中定义）和 MAXNAMELEN（也在＜sys/param.h＞中定义）。MAXPATHLEN 是绝对路径名的最大长度，MAXNAMELEN 是路径名每个部分的最大长度。这两个常量均包含终止 NUL 字符的空间。

例：简单目录清单

列出给定目录的所有项非常简单。程序 1-1 显示了完成这项工作的程序的实现。

本书中全部采用程序 1-1 中显示的源代码格式。对每个非空行进行编号，并且在左边的边空中，描述代码各个部分的文本都具有起始行号和结束行号。有时会在描述性段落前面加上粗体标题，它提供了所解释代码的总结。

每个代码段的开始和结束部分中的水平线指定了在其中可以找到这段代码的目录和文件名称（对于下面的示