



中国高等职业技术教育研究会推荐
高职高专系列教材

Linux 操作系统实用教程

温广民 王耀杰 编著



中国高等职业技术教育研究会推荐

高 职 高 专 系 列 教 材

Linux 操作系统实用教程

梁广民 王隆杰 编著

西安电子科技大学出版社

2004

内 容 简 介

本书是以中科红旗软件技术有限公司的红旗 Linux 服务器 3.0 为基础,从实用的角度来编写的。本书的最大特点是以企业需求为指导,讲求实用。学生认真学完本教材内容后,基本可以成为一名合格的 Linux 系统管理员。

本书分为三篇,分别是 Linux 基础、Linux 系统管理和 Linux 网络管理。Linux 的优势在于其强大的网络功能,因此本书内容在网络管理方面有所偏重。

书末附录中给出了每章习题的答案并列出了常用的 Linux 命令。

本书不仅可以作为高职高专计算机类学生的教材,而且也可以作为技术参考书或培训教材。无论是 Linux 的新手还是经验丰富的读者,都可以从本书中受益。

本书配有电子教案,需要者可与出版社联系,免费提供。

图书在版编目(CIP)数据

Linux 操作系统实用教程 / 梁广民等编著. —西安:西安电子科技大学出版社, 2004.2
(高职高专系列教材)

ISBN 7 - 5606 - 1336 - 5

. L... . 梁... . Linux 操作系统—高等学校:技术学校—教材 . TP316.89

中国版本图书馆 CIP 数据核字(2003)第 115596 号

策 划 马乐惠

责任编辑 刘锋利 马乐惠

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

<http://www.xduph.com>

E-mail: xdupfxb@pub.xaonline.com

经 销 新华书店

印 刷 西安兰翔印刷厂

版 次 2004 年 2 月第 1 版 2004 年 2 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印张 19.25

字 数 453 千字

印 数 1~4 000 册

定 价 20.00 元

ISBN 7 - 5606 - 1336 - 5/TP · 0708(课)

XDUP 1607001-1

如有印装问题可调换

本社图书封面为激光防伪覆膜,谨防盗版。

前 言

当今的操作系统主要有 Windows 和 Unix 两大阵营。从计算机专业角度来看,学生都应该会使用这两种操作系统。Unix(如 Solaris、AIX 等)是成熟的网络操作系统,然而它们更是商业化的操作系统,具有不菲的价格。而 Linux 可以说是免费的、源代码共享的 PC 版的 Unix 系统,它为我们学习和研究 Unix 操作系统提供了可能,更为难得的是 Linux 在实际中也经常作为生产平台使用。

目前 Linux 主要用于服务器和嵌入式系统两个方面,用于桌面方面则还与 Windows 有一些差距。本书是以国产的红旗 Linux 服务器 3.0 为基础,从实用的角度来编写的,具有如下特色:

在内容选取上,坚持集先进性、科学性和实用性于一体,尽可能地将最新、最实用的技术写到教材里,其中许多内容来自企业应用的一手材料。

在内容深浅程度上,把握理论够用、侧重实践、由浅入深的原则,通过大量的实例让学生分层次、分步骤地理解和掌握所学的知识。

在编写方式上,每章基本包括内容提要、实现步骤或应用举例、自我测试等环节。

在组织结构上,采用模块化,分别是 Linux 基础、Linux 系统管理和 Linux 网络管理。

由于本书面向的对象是 Linux 的入门者,所以书中尽可能通过实例来说明命令的使用和各种配置的使用方法。Linux 是一个功能强大的操作系统,我们并没有把全部内容囊括其中,而是选择最基本、最实用的内容进行了编写。本书共分三篇:Linux 基础、Linux 系统管理和 Linux 网络管理,编写时突出体现了 Linux 多用户、多任务的网络操作系统的特点。

本书第 1 章介绍了 Linux 的典型安装过程;第 2 章介绍了 Linux 常用命令,如文件和目录的操作、文件显示命令、作业和进程控制命令、文件压缩解压命令、文件查找命令、网络命令等;第 3 章介绍了 vi 编辑器的使用;第 4 章介绍了用户和组的管理、口令的管理、磁盘定额;第 5 章介绍了 Linux 系统中的设备管理;第 6 章介绍了文件系统的管理;第 7 章介绍了 Shell 编程知识;第 8~14 章则分别介绍了一个网络服务的配置,其中包括网络文件系统(NFS)、DHCP 服务、Samba 服务、DNS 服务、Web 服务(Apache)、FTP 服务、防火墙和代理服务器。

总之,本书以适应高职高专教学改革的需要为目标,充分体现高职特色,努力从内容到形式都有所创新和突破。

本书由梁广民担任主编并编写了第 1、3、5、7、10、11、12、13 章；第 2、4、6、8、9、14 章由王隆杰编写。刘兴东高级工程师、徐人凤高级工程师、常理民副教授、聂哲老师审阅了全稿，提出了许多宝贵意见，并对本书从开始到完稿始终给予关心和支持。另外，本专业教研室的石光华老师、石淑华老师、杨名川老师、刘平老师和邹润生老师在实验和绘图方面做了不少工作，在此一并表示衷心感谢！

由于编者水平有限，虽尽编者所能，书中难免还有疏漏之处，敬请读者批评指正。

编 者

2003 年 10 月

目 录

第一篇 Linux 基础

第 1 章 Linux 入门及安装	3	2.3.2 查看系统的进程	32
1.1 Linux 入门	3	2.3.3 进程的控制	33
1.1.1 什么是 Linux	3	2.3.4 作业控制	37
1.1.2 Linux 的优点	4	2.4 文件压缩和备份	39
1.1.3 Linux 操作系统的架构	5	2.4.1 压缩和解压命令	39
1.1.4 Linux 与其他操作系统的比较	6	2.4.2 文件备份	40
1.1.5 如何得到 Linux 的最新消息	7	2.5 网络命令	41
1.2 红旗 Linux 的安装	7	2.5.1 hostname、ping、host	41
1.2.1 红旗 Linux 简介	7	2.5.2 ifconfig	42
1.2.2 安装前的准备	8	2.5.3 traceroute 目标主机名或 IP 地址	43
1.2.3 安装红旗 Linux 服务器 3.0	9	2.5.4 Telnet、FTP	43
1.3 LILO 的配置和使用	17	2.5.5 wall、write、mesg	46
1.3.1 LILO 简介	17	2.5.6 mail	46
1.3.2 LILO 配置	17	2.5.7 finger	47
1.3.3 LILO 提示信息	19	2.5.8 netstat[参数选项]	47
本章小结	19	2.6 其他命令	48
习题	20	2.6.1 clear、dmesg、uname	48
第 2 章 常用的 Linux 命令	21	2.6.2 date、cal	48
2.1 文件和目录操作命令	21	2.6.3 help、man	49
2.1.1 pwd、cd	21	2.6.4 init、shutdown、halt、reboot、 poweroff	49
2.1.2 ls、tree	22	2.6.5 alias、unalias、history	51
2.1.3 mkdir、rmdir	23	2.6.6 su	51
2.1.4 cp、rm、mv、ln	23	2.6.7 who、whoami、w、last	51
2.1.5 chmod、chown、chgrp	25	2.6.8 rpm——安装软件包	52
2.1.6 find、grep	26	本章小结	55
2.1.7 cmp、diff	27	习题	55
2.1.8 stat、touch	28	第 3 章 vi 编辑器的使用	56
2.2 显示命令	29	3.1 vi 的工作模式	56
2.2.1 cat、more、less	29	3.2 vi 的启动和退出	56
2.2.2 head、tail	29	3.3 vi 长指令和短指令	58
2.2.3 sort、uniq	30	3.4 vi 高级应用	61
2.2.4 file、locate、which	31	3.4.1 设置 vi 环境	61
2.3 进程管理和作业控制	32	3.4.2 缩写与宏	63
2.3.1 进程的启动	32		

3.4.3 .exrc 文件.....	64	本章小结.....	66
3.4.4 运行 Shell 命令.....	65	习题.....	66

第二篇 Linux 系统管理

第 4 章 用户和组的管理.....	69	6.1.2 什么是文件系统.....	103
4.1 用户的管理.....	69	6.1.3 文件.....	105
4.1.1 Linux 下的用户.....	69	6.1.4 Linux 系统的目录结构.....	106
4.1.2 账号系统文件.....	69	6.2 创建文件系统.....	106
4.1.3 创建新的用户.....	72	6.2.1 Fdisk 的使用.....	107
4.1.4 修改用户的属性.....	73	6.2.2 文件系统的建立.....	110
4.1.5 停止用户.....	75	6.2.3 交换分区.....	111
4.1.6 默认新用户的设置.....	76	6.3 文件系统的安装和卸载.....	113
4.1.7 用户登录系统后环境的设定.....	78	6.3.1 手工安装和卸载文件系统.....	113
4.1.8 超级用户.....	81	6.3.2 文件系统的自动安装.....	115
4.2 组的管理.....	81	6.4 文件系统的维护.....	115
4.2.1 Linux 下的组和组文件.....	81	6.4.1 检查文件系统.....	115
4.2.2 组的添加.....	82	6.4.2 磁盘坏块的检查.....	116
4.2.3 组属性的修改.....	82	6.4.3 其他常用的文件系统管理命令.....	117
4.2.4 文件的安全问题.....	83	本章小结.....	118
4.3 磁盘配额.....	84	习题.....	118
本章小结.....	87	第 7 章 Shell 编程.....	119
习题.....	87	7.1 Shell 的基本概念.....	119
第 5 章 设备管理.....	88	7.1.1 Shell 的概念.....	119
5.1 硬件设备.....	88	7.1.2 Shell 的种类.....	121
5.1.1 设备文件.....	88	7.1.3 创建及执行 Shell 脚本.....	123
5.1.2 设备分类.....	89	7.2 Shell 语法.....	124
5.2 使用设备.....	90	7.2.1 Shell 变量.....	124
5.2.1 磁盘.....	90	7.2.2 数值运算.....	126
5.2.2 CD-ROM.....	91	7.2.3 条件命令.....	127
5.2.3 打印机.....	91	7.2.4 循环命令.....	132
5.2.4 显卡.....	93	7.2.5 函数的定义和使用.....	137
5.2.5 声卡.....	95	7.3 正则表达式.....	138
5.2.6 Modem.....	96	7.3.1 正则表达式基本元字符及使用.....	138
5.2.7 ADSL.....	98	7.3.2 正则表达式的应用.....	143
本章小结.....	100	7.4 Shell 编程综合实例.....	147
习题.....	100	7.4.1 实例一.....	147
第 6 章 文件系统管理.....	102	7.4.2 实例二.....	147
6.1 文件系统基础.....	102	本章小结.....	149
6.1.1 磁盘的分区.....	102	习题.....	150

第三篇 Linux 网络管理

第 8 章 网络文件系统 NFS	153	10.1.2 什么是 Samba	186
8.1 NFS 基本原理.....	153	10.1.3 Samba 的功能	187
8.1.1 什么是 NFS (Network File System)	153	10.1.4 Samba 的启动和退出	187
8.1.2 NFS 的工作原理.....	153	10.2 Samba 配置	188
8.2 配置 NFS 服务器.....	154	10.2.1 设置 smb.conf 文件	188
8.2.1 安装 NFS.....	155	10.2.2 共享访问控制	192
8.2.2 配置导出文件:/etc/exports	155	10.2.3 Samba 安全级别.....	194
8.2.3 激活 NFS.....	157	10.2.4 guest 用户映射	194
8.2.4 导出目录：exportfs	159	10.3 使用加密口令.....	195
8.3 配置 NFS 客户	160	10.3.1 Samba 口令文件.....	195
8.4 NFS 的性能、安全和故障排除	163	10.3.2 使用加密口令	195
8.4.1 NFS 的性能	163	10.3.3 smbpasswd 的使用	196
8.4.2 NFS 的安全	164	10.3.4 不使用加密口令.....	197
8.4.3 NFS 故障排除	165	10.4 Samba 和 Windows 互相通信	198
本章小结.....	168	10.4.1 从 Linux 机上访问 Windows 资源	198
习题	168	10.4.2 从 Windows 机上访问 Linux 资源	201
第 9 章 动态主机配置协议 DHCP.....	169	10.4.3 Linux 和 Windows 互发短消息 ..	202
9.1 DHCP 简介	169	10.5 Samba 组件中的应用程序	204
9.1.1 为什么需要 DHCP.....	169	10.5.1 报告 Samba 状态.....	204
9.1.2 BOOTP 引导程序协议.....	170	10.5.2 基于 Web 的配置工具 ——SWAT.....	204
9.1.3 DHCP 动态主机配置协议	170	10.6 Samba 常见故障排除	206
9.1.4 DHCP 的工作过程.....	171	10.6.1 Samba 服务器上的故障排除.....	206
9.1.5 DHCP 功能的进一步讨论	172	10.6.2 Samba 客户机上的故障排除.....	207
9.2 DHCP 的配置	172	本章小结	207
9.2.1 DHCP 服务器的配置.....	172	习题.....	207
9.2.2 DHCP 客户的配置.....	176	第 11 章 域名系统.....	209
9.3 DHCP 服务器的高级配置.....	179	11.1 DNS 简介	209
9.3.1 为计算机分配固定的 IP 地址.....	179	11.1.1 概述	209
9.3.2 进一步说明 dhcpd.conf	179	11.1.2 DNS 结构	210
9.3.3 DHCP 转接代理	183	11.1.3 资源记录	211
9.4 DHCP 故障排除.....	183	11.1.4 域名服务器分类.....	212
本章小结.....	185	11.2 DNS 域名解析.....	214
习题	185	11.2.1 客户解析过程调用	214
第 10 章 Samba.....	186	11.2.2 域名解析过程	215
10.1 Samba 简介	186		
10.1.1 SMB 协议	186		

11.3 DNS 配置.....	215	13.2.3 ftphosts.....	266
11.3.1 BIND 及其主要配置文件	215	13.3 wu-ftp 的相关应用	267
11.3.2 相关配置文件	222	13.3.1 连接数统计命令 ftpcount	267
11.3.3 DNS 的安全管理	223	13.3.2 在线用户查看命令 ftpwho	267
11.4 DNS 的启动、停止和测试.....	223	13.3.3 FTP 关闭文件生成命令 ftpshut ...	267
11.4.1 DNS 的启动和停止.....	223	13.3.4 用脚本实现自动 FTP	268
11.4.2 DNS 测试.....	224	13.4 wu-ftp 常见故障排除.....	268
11.5 DNS 故障排除	227	13.4.1 检查 ftp 的配置文件.....	269
本章小结	228	13.4.2 查看 log 文件	269
习题	228	本章小结	269
第 12 章 Apache	230	习题.....	270
12.1 Apache 简介.....	230	第 14 章 防火墙.....	271
12.1.1 Apache 的地位和功能.....	230	14.1 防火墙简介	271
12.1.2 Apache 的下载和安装.....	231	14.1.1 防火墙的分类和基本工作原理 ...	271
12.1.3 Apache 的启动与关闭.....	234	14.1.2 包过滤型防火墙的两种策略.....	273
12.2 Apache 的配置	234	14.2 用 ipchains 过滤数据包	274
12.2.1 文件 httpd.conf 的全局参数.....	235	14.2.1 什么是 ipchains	274
12.2.2 文件 httpd.conf 的服务器的 主要设置	237	14.2.2 使用 ipchains 的准备工作	274
12.3 Apache 的各种服务.....	239	14.2.3 ipchains 的工作流程.....	274
12.3.1 用户个人主页	239	14.2.4 ipchains 命令.....	277
12.3.2 虚拟主机	240	14.2.5 ipchains 的使用	278
12.3.3 代理服务	242	14.2.6 实例	281
12.4 Apache 访问控制	244	14.2.7 让建立的规则在系统启动时 生效	283
12.4.1 Apache 访问控制指令.....	245	14.2.8 IP 伪装	284
12.4.2 基于主机的访问控制.....	246	14.3 iptables	284
12.4.3 基于用户名的访问控制.....	247	14.3.1 iptables 的原理.....	284
12.5 Apache 常见故障排除.....	249	14.3.2 使用 iptables 准备工作.....	285
本章小结	250	14.3.3 iptables 命令.....	285
习题	251	14.3.4 iptables 使用实例	286
第 13 章 FTP	252	14.3.5 iptables 与 ipchains 的区别.....	287
13.1 FTP 简介.....	252	14.3.6 iptables 中的 IP 伪装.....	288
13.1.1 文件传输协议	252	本章小结	288
13.1.2 FTP 命令.....	253	习题.....	288
13.1.3 wu-ftp	255	附录 A 习题参考答案	289
13.2 配置 wu-ftp 服务器.....	256	附录 B 命令说明	292
13.2.1 ftpaccess 文件	256	参考文献	298
13.2.2 ftpusers.....	266		

第一篇

基础

第 1 章 Linux 入门及安装



Linux 是免费发行和使用的快速高效的操作系统，它的出现在计算机界引发了一场革命，在一些重要的 Web 站点、公司信息系统和教育应用程序中都采用了这个免费软件，因此，我们有必要了解 Linux 的历史、特点以及安装。

1.1 Linux 入门

1.1.1 什么是 Linux

Linux 是一套免费使用和自由传播的类 Unix 操作系统，它主要用于基于 Intel x86 系列 CPU 的计算机上。这个系统是由全世界各地的成千上万的程序员设计和实现的，其目的是建立不受任何商品化软件的版权制约且全世界都能自由使用的 Unix 兼容产品。

Linux 的出现，最早开始于一位名叫 Linus Torvalds 的计算机业余爱好者，当时他是芬兰赫尔辛基大学的学生，他的目的是设计一个代替 Minix(是由一位名叫 Andrew Tannebaum 的计算机教授编写的一个操作系统示教程序)的操作系统，这个操作系统可用于 386、486 或奔腾处理器的个人计算机上，并且具有 Unix 操作系统的全部功能，这就开始了 Linux 雏形的设计。

Linux 以其高效性和灵活性著称，它能够在 PC 计算机上实现全部的 Unix 特性，具有多用户、多任务的能力。Linux 是在 GNU 公共许可权限下免费获得的，是一个符合 POSIX 标准的操作系统。

所谓 GNU，是 Stallman 在 1984 年提出的一个计划，它的思想是“源代码共享，思想共享”，目的是开发一个完全自由的，与 Unix 类似但功能更强的操作系统，以便为所有的计算机使用者提供一个功能齐全、性能良好的基本系统。在其他人的协作下，他创作了通用公共许可证(General Public License, GPL)，这对推动自由软件的发展起了重要的作用。与传统的商业软件许可证不同的是，GPL 保证任何人有共享和修改自由软件的自由，任何人都有权取得、修改和重新发布自由软件的源代码，并且规定在不增加费用的条件下得到源代码(基本发行费用除外)。这一规定保证了自由软件的总体费用很低，而在使用 Internet 的情况下则是免费的。GPL 条款还规定自由软件的衍生作品继续保持自由状态，并且用户在扩散 GNU 软件时，必须让下一个用户也有获得源代码的权利。这些工作为后来 Linux 操作系统的迅速发展奠定了坚实的基础。

Linux 操作系统软件包不仅包括完整的 Linux 操作系统、文本编辑器、高级语言编译器 etc 等应用软件，还包括带有多个窗口管理器的 X Window 图形用户界面，如同我们使用 Windows 一样，允许我们使用窗口、图标和菜单对系统进行操作。1994 年，Linux 的第一

个产品版 Linux 1.0 问世，如今 Linux 家族已经有了近 140 个不同的版本，所有这些版本都基于最初的免费的源代码。不同的公司可以推出不同的 Linux 产品，但是它们都必须承诺对初始源代码的任何改动皆公布于众。

1.1.2 Linux 的优点

Linux 之所以受到广大计算机爱好者的喜爱，主要原因有如下几个：

(1) 为我们提供了学习、探索以及修改计算机操作系统内核的机会。操作系统是计算机必不可少的系统软件，是整个计算机系统的灵魂。每个操作系统都是一个复杂的计算机程序集，它提供操作过程的协议或行为准则；没有操作系统，计算机就无法工作，就不能解释和执行用户输入的命令或运行简单的程序。大多数操作系统都是一些主要的软件公司支持的商品化程序，用户只能有偿使用。如果用户购买了一个操作系统，他就必须接受供应商所要求的一切条件。因为操作系统是系统程序，用户不能擅自修改或试验操作系统的内核，这对于广大计算机爱好者来说无疑是一种束缚。

要想发挥计算机的作用，仅有操作系统还不够，还必须要有各种应用程序的支持。应用程序是用于处理某些工作(如字处理)的软件包，通常它也只能有偿使用。每个应用程序的软件包都为特定的操作系统和机器编写，使用者无权修改这些应用程序。由于 Linux 是一套自由软件，用户可以无偿地得到它及其源代码，可以无偿地获得大量的应用程序，而且可以任意地修改和补充它们，无约束地再传播，这对用户学习和了解 Unix 操作系统的内核非常有益。

(2) 可以节省大量的资金。Linux 是目前惟一可免费获得的、为 PC 机平台上的多个用户提供多任务、多进程功能的操作系统，这是人们喜欢使用它的主要原因。就 PC 机平台而言，Linux 提供了比其他任何操作系统都要强大的功能，Linux 还可以使用户远离各种商品化软件提供者促销广告的诱惑，再也不用承受每过一段时间就花钱去升级之苦，因此可以节省大量用于购买或升级应用程序的资金。

(3) 丰富的应用软件。Linux 不仅为用户提供了强大的操作系统功能，而且还提供了丰富的应用软件。用户不但可以从 Internet 上下载 Linux 及其源代码，而且还可以从 Internet 上下载许多 Linux 的应用程序。可以说，Linux 本身包含的应用程序以及移植到 Linux 上的应用程序包罗万象，任何一位用户都能从有关 Linux 的网站上找到适合自己特殊需要的应用程序及其源代码，这样用户就可以根据自己的需要下载源代码，以便修改和扩充操作系统或应用程序的功能。这对 Windows 2000、Windows 98、MS-DOS 或 OS/2 等商品化操作系统来说是无法做到的。

(4) 使我们的工作更加方便。Linux 为广大用户提供了一个在家里学习和使用 Unix 操作系统的机会。尽管 Linux 只是由计算机爱好者们开发的，但它在很多方面还是相当稳定的，从而为用户学习和使用目前世界上最流行的 Unix 操作系统提供了便利的机会。现在有许多 CD-ROM 供应商和软件公司(如 RedHat、红旗和 Turbo Linux 等)支持 Linux 操作系统。Linux 成为 Unix 系统在个人计算机上的一个代用品，并能用于替代那些较为昂贵的系统。因此，如果一个用户在公司上班时在 Unix 系统上编程，或者在工作中是一位 Unix 的系统管理员，他就可以在家里安装一套 Unix 的兼容系统，即 Linux 系统，在家中使用 Linux 就能够完成一些工作任务。

(5) 提供功能强大而稳定的网络服务。Linux 最优秀的功能莫过于其网络功能。首先，它可以支持众多的网络协议，比如 TCP/IP 协议、SPX/IPX 协议、NETBEUI 协议、X.25 协议等；其次，Linux 可以提供非常广泛的网络服务，比如 WWW、FTP、E-mail、Telnet、NFS、DHCP、Samba、防火墙以及企业的群组服务等，这些功能为 Linux 提供了无与伦比的网络亲和性。

1.1.3 Linux 操作系统的架构

Linux 一般有四个主要部分：内核、Shell、文件结构和实用工具。

1. 内核

内核是系统的核心，是运行程序和管理像磁盘和打印机等硬件设备的核心程序。

2. Shell

Shell 是系统的用户界面，它提供了用户与内核进行交互操作的一种接口。实际上 Shell 是一个命令解释器，它解释由用户输入的命令并把它们送到内核去执行。不仅如此，Shell 有自己的用于对命令进行编辑的编程语言，它允许用户编写由 Shell 命令组成的程序。Shell 编程语言具有普通编程语言的很多特点，比如它也有循环结构和分支控制结构等，用这种编程语言编写的 Shell 程序与其他应用程序具有同样的效果。有关 Shell 更详细的内容，我们将在第 7 章中讨论。

Linux 提供了像 Microsoft Windows 那样的可视的命令输入界面——X Window 的图形用户界面(GUI)。它提供了很多窗口管理器，其操作就像 Windows 一样，有窗口、图标和菜单，所有的管理都通过鼠标控制。现在比较流行的窗口管理器是 KDE 和 GNOME。每个 Linux 系统的用户可以拥有他自己的用户界面或 Shell，用以满足他们自己专门的 Shell 需要。同 Linux 本身一样，Shell 也有多种不同的版本。

3. 文件结构

文件结构是文件存放在磁盘等存储设备上的组织方法，主要体现在对文件和目录的组织上。目录提供了管理文件的一个方便而有效的途径，我们不但能够从一个目录切换到另一个目录，而且可以设置目录、文件的权限及文件的共享程度。

Linux 目录采用多级树形结构，用户可以浏览整个系统，可以进入任何一个已授权进入的目录，并访问那里的文件。

文件结构的相互关联性使共享数据变得容易，几个用户可以访问同一个文件。Linux 是一个多用户系统，操作系统本身的驻留程序存放在以根目录开始的专用目录中，有时被指定为系统目录。

内核、Shell 和文件结构一起形成了基本的操作系统结构，它们使得用户可以运行程序，管理文件以及使用系统。此外，Linux 操作系统还有许多被称为实用工具的程序，辅助用户完成一些特定的任务。

4. 实用工具

标准的 Linux 系统都有一套叫做实用工具的程序，它们是专门的程序，例如编辑器、执行标准的计算操作等。另外，用户也可以产生自己的工具。

一般来讲，实用工具可分为以下三类。

(1) 编辑器：用于编辑文件。Linux 的编辑器主要有 vi、emacs、pico 等。

(2) 过滤器：用于接收并过滤数据。Linux 的过滤器(Filter)读取从用户文件或其他地方输入的数据，经检查和处理后输出结果。从这个意义上说，它们过滤了经过它们的数据。Linux 有不同类型的过滤器，一些过滤器用行编辑命令输出一个被编辑的文件；另外一些过滤器是按模式寻找文件并以这种模式输出部分数据；还有一些执行字处理操作，检测一个文件中的格式，输出一个格式化的文件。过滤器的输入可以是一个文件，也可以是用户从键盘键入的数据，还可以是另一个过滤器的输出。过滤器可以相互连接，因此，一个过滤器的输出可能是另一个过滤器的输入。在有些情况下，用户可以编写自己的过滤器程序。

(3) 交互程序：允许用户发送信息或接收来自其他用户的信息。交互程序是用户与机器的信息接口。Linux 是一个多用户系统，它必须和所有的用户保持联系。信息可以由系统上的不同用户发送或接收。信息的发送有两种方式：一种方式是与其他用户一对一地进行对话，另一种方式是一个用户对多个用户同时进行通讯，即所谓广播式通讯。

1.1.4 Linux 与其他操作系统的比较

Linux 可以与 MS-DOS、OS/2、Windows 等其他操作系统共存于同一台机器上，它们既具有一些共性，相互之间又各有特色，有所区别。

目前运行在 PC 机上的操作系统主要有 Microsoft 的 MS-DOS、Windows，IBM 的 OS/2 等。早期的 PC 机用户普遍使用 MS-DOS，因为这种操作系统对机器的硬件配置要求不高，但是随着计算机硬件技术的飞速发展，硬件设备价格越来越低，人们可以相对容易地提高计算机的硬件配置，于是开始使用 Windows 等具有图形界面的操作系统。Linux 是新近被人们所关注的操作系统，它正逐渐为 PC 机的用户所接受。那么，Linux 与其他操作系统的主要区别是什么呢？下面从两个方面加以论述。

1. Linux 与 MS-DOS 的区别

在同一系统上运行 Linux 和 MS-DOS 已很普遍，但它们之间还是有较多区别的。

就发挥处理器功能来说，MS-DOS 没有完全发挥 x86 处理器的功能，而 Linux 完全在处理器保护模式下运行，并且发挥了处理器的所有特性。Linux 可以直接访问计算机内的所有可用内存，提供完整的 Unix 接口，而 MS-DOS 只支持部分 Unix 的接口。

就使用费用而言，Linux 和 MS-DOS 是两种完全不同的实体。与其他商业操作系统相比，MS-DOS 价格比较便宜，而且在 PC 机用户中有很大的占有率，任何其他 PC 机操作系统都很难达到 MS-DOS 的普及程度，因为其他操作系统的费用对大多数 PC 机用户来说都是一个不小的负担，而 Linux 是免费的，用户可以从 Internet 上或者其他途径获得它的版本，而且可以任意使用，不用考虑费用问题。

就操作系统的功能来说，MS-DOS 是单任务的操作系统，一旦用户运行了一个 MS-DOS 的应用程序，它就独占了系统的资源，用户不可能再同时运行其他应用程序，而 Linux 是多任务的操作系统，用户可以同时运行多个应用程序。

2. Linux 与 OS/2、Windows 的区别

从发展的背景看，Linux 与其他操作系统区别在于：Linux 是从一个比较成熟的操作系

统发展而来的，而其他操作系统(如 Windows NT、Windows 2000 等)都是自成体系，无对应的相依托的操作系统。这一区别使得 Linux 的用户能大大地从 Unix 团体贡献中获利。因为 Unix 是当今世界上使用最普遍、发展最成熟的操作系统之一，它是 20 世纪 70 年代中期发展起来的微机和巨型机的多任务系统，虽然有时接口比较混乱，并缺少相对集中的标准，但还是逐步发展壮大成为最广泛使用的操作系统之一。无论是 Unix 的作者还是 Unix 的用户，都认为只有 Unix 才是一个真正的操作系统，许多计算机系统(从个人计算机到超级计算机)都存在 Unix 版本，Unix 的用户可以从很多方面得到支持和帮助。因此，Linux 作为 Unix 的一个克隆，它的用户同样会得到相应的支持和帮助，Linux 将直接拥有 Unix 在用户中建立的牢固地位。

从使用费用上看，Linux 与其他操作系统的区别在于：Linux 是一种开放、免费的操作系统，而其他操作系统都是封闭的系统，需要有偿使用。这一区别使得我们不用花钱就能得到很多 Linux 的版本以及为其开发的应用软件。当我们访问 Internet 时，会发现几乎所有可用的自由软件都能够运行在 Linux 系统上，不同软件商对这些软件有不同的 Unix 实现方法。Unix 的开发、发展商以开放系统的方式推动其标准化，但却没有一个公司来控制这种设计。因此，任何一个软件商(或开拓者)都能在某种 Unix 中实现这些标准。而 OS/2 和 Windows 等操作系统是具有版权的产品，其接口和设计均由某一公司控制，而且只有这些公司才有权实现其设计，它们都是在封闭的环境下发展的。

1.1.5 如何得到 Linux 的最新消息

有关 Linux 的站点现在到处都是，这里向大家推荐一些。

(1) <http://www.kernel.org>。这是一个关于 Linux 核心最新消息的网站，从中可以得到核心(Kernel)发展情况的最新信息。

(2) <http://www.linuxforum.net>。这是一个非常著名的讨论组。

(3) <http://www.aka.org.cn>。AKA 是一个非常好的自由软件团体，里面有许多很实用的信息。

(4) <http://www.linuxaid.com.cn>。LinuxAid 技术支持中心是国内首家专门从事 Linux 技术支持服务的网站，以专业的技术支持和服务为核心，来满足个人及企业用户对 Linux 技术的需求。

(5) <http://www.linuxden.com>。“Linux 伊甸园”，是一个不错的 Linux 专业网站，特别是 Linux 下的自由软件非常的丰富。

1.2 红旗 Linux 的安装

1.2.1 红旗 Linux 简介

红旗 Linux 是 Linux 的一个发展产品，由中科红旗软件技术有限公司开发研制，以 Intel 和 Alpha 芯片为 CPU 构成的服务器平台，它是第一个国产的操作系统版本。它的研发成功标志着我国在发展国产操作系统的道路上迈出了坚实的一步。

相对于 Windows 操作系统及 Unix 操作系统来讲，Linux 凭借其开放性及其低成本优势，

已经在服务器操作系统的市场获得了巨大发展。但由于其操作界面复杂，一时难以让普通 PC 用户接受。GNOME 是 GNU 组织中专门开发桌面环境的项目，GNOME 作为开放源代码的软件开发成果，基于兼容性良好的 CORBA 技术，与 Linux 系统相辅相成，带给用户更加友好的界面，更多的使用，检查，修改及分发自由。与同是开放源代码图形用户界面的 KDE 相比较，GNOME 表现得更能减轻其他公司创建 Linux 桌面应用的劳动。GNOME 已经成为业内人士普遍看好的一个趋势性软件。

目前市场上使用最多的是红旗服务器 3.0 和桌面 3.2，最近将推出新产品——红旗服务器 4.0，采用的内核为 2.4。红旗软件已在中国市场上奠定了坚实的基础，成为新一代的操作系统先锋。特别是在中文处理方面，红旗 Linux 预装了炎黄中文平台和方正 TrueType 字库，支持大字符集(GBK)，实现了在 Linux 上的 TrueType 显示和打印功能。

红旗 Linux 的优点如下：

- (1) 安装简便，智能化程度高，而且支持 1024 柱面以上硬盘的安装；
- (2) 界面友好，更简便，更适合初学者；
- (3) 中文输入时，具有光标跟随的功能，可以用【Ctrl】+【Space】组合键方便地进行中、英文切换，支持多种中文字库和五笔、拼音等多种输入法；
- (4) 图形化的 Linuxconf 可以方便地对整个系统进行配置和管理；
- (5) 具有完整的在线手册和帮助文档。全中文的在线手册可以快速查到系统中每条指令的详细用法；
- (6) 带有炎黄中文 KDE 环境(启动命令为“yh”)，启动炎黄中文 KDE 环境后，就可以利用 Linux 提供的工具进行中文的阅读和输入。

总之，红旗 Linux 的推出，将对中国的计算机产业产生巨大的影响。操作系统的多元化带动着软件的多元化发展，同时对硬件发展也有推动作用。有了国产操作系统后，许多优秀的应用软件就能不受微软捆绑软件的束缚，而在自己的操作系统上一展身手了。

1.2.2 安装前的准备

虽然 Linux 发展了很长时间，但是其安装过程不像安装 Windows 那样容易。在安装前，我们必须做好如下的准备工作。

1. 收集系统硬件信息

虽然 Linux 可以自动识别许多硬件设备，但其在这方面的功能还是太简单了，特别是对于显卡和声卡的支持。为了完成安装，我们可能需要手动输入一些信息，所以第一步就是收集所有硬件的信息(生产厂商以及型号)及其配置信息。如果你使用的是 Windows 95、Windows 98 或 Windows NT，最简单的就是将设备管理器中的信息打印出来。需要收集的硬件信息主要包括：

- (1) 主机名和主机的网络设置(包括本机的 IP 地址、网络掩码、网关和 DNS 等)；
- (2) CPU 类型；
- (3) 内存容量；
- (4) 显卡、网卡及声卡类型。

2. 规划磁盘

由于捆绑的应用程序不同,各种 Linux 发行版需要的硬盘空间也不一样。Linux 必须安装在其独有的分区中。如果只有一个分区并且被 Windows 使用着,那么就需要备份你全部的数据,并且创建新的分区。还可以使用某些第三方工具来改变现有分区的大小,比如 PowerQuest 的 PartitionMagic 4.0,对于红旗 Linux 3.0,最好规划出 2 GB 以上的空间。

3. 备份数据

由于 Linux 需要独立的分区,所以我们一定要备份打算安装 Linux 分区的内容以及完整的系统分区表。如果是升级安装,还需要备份/etc 和/home 两个目录的内容。

4. 制作启动盘

对于不能从光驱启动的计算机,我们需要从软驱启动,首先应该制作一张启动盘,具体步骤如下:

- (1) 运行红旗光盘上的/dosutils/rawwrite.exe,如图 1-1 所示。

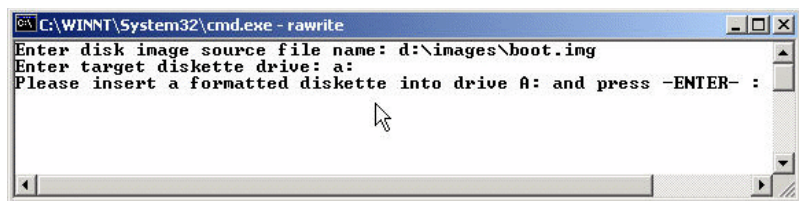


图 1-1 制作启动盘

- (2) 输入源文件的位置“g:\images\boot.img”,其中“g”是计算机光驱的提示符。
- (3) 输入目标文件的位置“a:\”。
- (4) 回车确认。

1.2.3 安装红旗 Linux 服务器 3.0

现在开始安装 Linux,请确认能够从光驱启动红旗 Linux 3.0 或已经生成了 Linux 启动软盘,并且为 Linux 腾出了空间。本章我们用一个具体的安装实例来讲述红旗 Linux 3.0 的安装过程。

1. 选择安装方式

从光驱启动后,系统会询问采用哪种安装方式,如图 1-2 所示。



图 1-2 安装方式选择

红旗 Linux Server 3.0 有四种安装方式可供选择,对于一般用户来讲,第一种(图形方式)和第二种(文本方式)比较常用。现将这两种安装方式分别介绍如下:

(1) 图形方式。图形方式安装 Linux 的优点是比较直观、方便和简单;缺点是它屏蔽了一些信息,不利于了解整个安装过程。另外,图形方式对系统内存的要求稍微高一些,一般要在 128 MB 以上,否则安装速度较慢。如果选择该方式安装,直接回车即可。

(2) 文本方式。文本方式安装 Linux 的优点是安装过程比较迅速,安装选项比较灵活,可定制范围大,而且可以避免由于显卡问题而导致安装失败现象的发生;缺点是安装过程麻烦,需要用户参与的选项多一些。对 Linux 有一定的基础,而又想了解 Linux 安装更为详细的信息的用户,可以选择该方式。如果选择该方式安装,需要在“boot:”后面输入“text”,再按回车键即可。

2. 同意软件协议

选择安装方式后,接下来就是欢迎画面,然后提示用户是否同意软件协议,如图 1-3 所示。

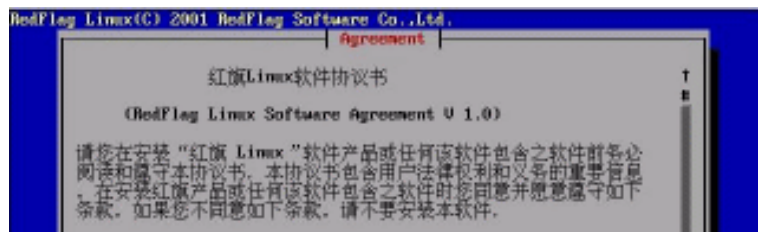


图 1-3 红旗 Linux 软件协议

用户阅读完协议之后选择同意,进入下一步。

3. 选择计划安装方式

这一步用户可以选择是安装或是升级。我们选择安装,如图 1-4 所示。

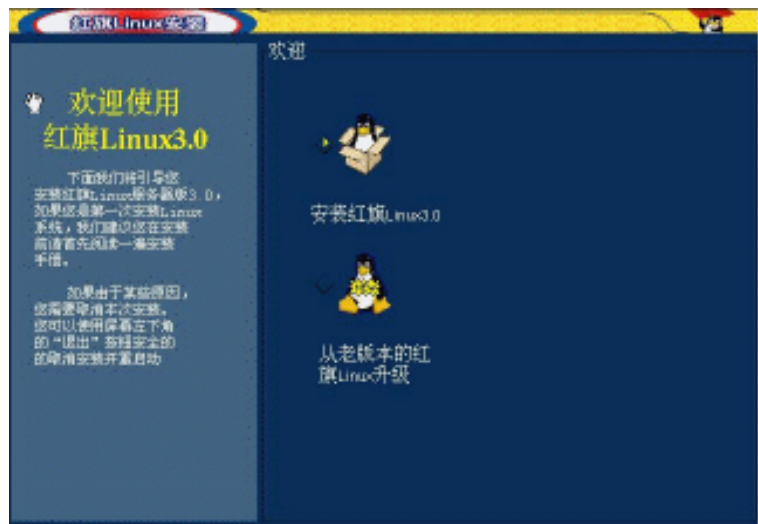


图 1-4 计划安装方式

4. 选择安装类型

安装类型有四种选择，如图 1-5 所示。

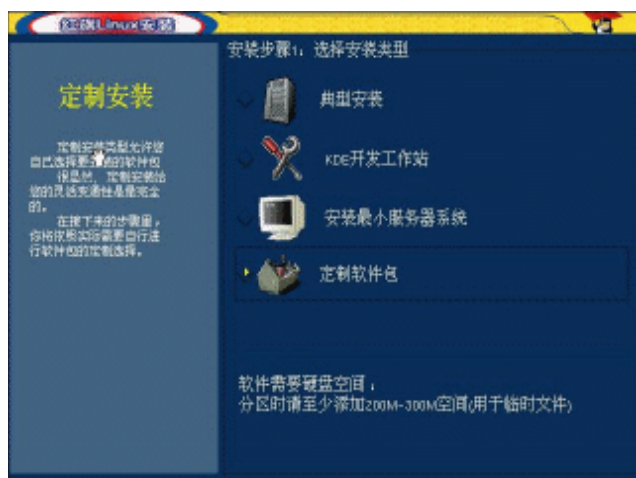


图 1-5 安装类型

(1) 典型安装：此种安装不仅包括最基本的 Linux，而且包含 X Window System、KDE 桌面环境等众多软件包，大约需要 800 MB 左右的磁盘空间；

(2) KDE 开发工作站：在典型安装的基础上增加了开发工具，大约需要 1000 MB 左右的磁盘空间；

(3) 安装最小服务器系统：最基本的 Linux(文本式)不包括 X Window System，大约需要 400 MB 左右的磁盘空间；

(4) 定制软件包：用户可以定制自己所需要的软件包。

选择定制软件包，可以根据自己的实际需要来选择想要安装的组件。

5. 选择软件包组件

红旗 Linux Server 3.0 下的软件包组件非常丰富，包括桌面环境、图形工具、网络工具、多媒体工具、软件开发工具和实用程序等，用户可以根据自己的需要选择相应的组件。为了大家学习方便，我们选择最后一项，即“全部选中”，如图 1-6 所示。

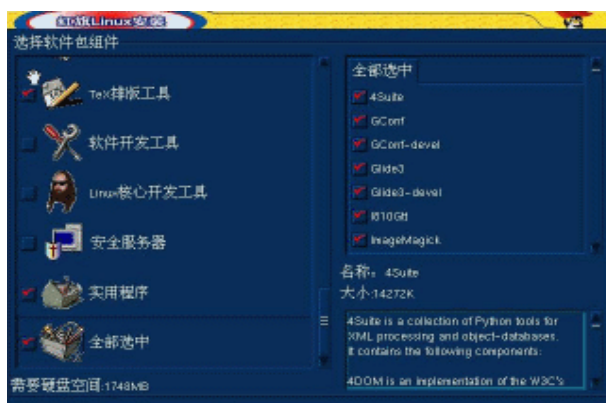


图 1-6 选择软件包组件

6. 选择分区工具

红旗 Linux Server 3.0 下可以使用的分区工具有定制分区和 Fdisk 工具两种, 如图 1-7 所示。



图 1-7 选择分区工具

(1) 定制分区(Disk Druid)。这是一个图形化分区工具, 具有直观、易操作的特点, 对初学者来说是一个理想的分区工具。我们就选择它作为分区工具。

(2) Fdisk 工具。这是一个很好的分区工具, 最可靠, 功能也比较强大。但对初学者来说可能会有一些困难, 不过启动 Fdisk 后, 利用 m 命令可以获得在线帮助。以下列出了 Fdisk 的主要命令:

- m: 提供所有可用命令和列表;
- p: 提供当前分区信息的列表;
- n: 添加新的分区;
- t: 设置或改变分区的类型;
- l: 提供不同分区类型及它们 ID 号的列表;
- w: 保存信息并退出 Fdisk;
- q: 退出但不保存。

7. 分区

在 Linux 中至少应该有根分区和交换分区, 当然可以划分更多的分区, 如果作为服务器的用途, 建议将 /usr、/usr/local、/home、/var、/boot 单独放在一个分区内。分区大小建议如下:

- /boot: 100 MB;
- /usr: 大于 800 MB;
- /usr/local: 用于系统安装新的软件, 可以根据硬盘的实际情况来预留空间;
- /home: 该空间的大小可以根据下面的公式计算:
$$50 \text{ MB} \times \text{用户数目} + \text{FTP 服务预留空间}$$
- /var: 大于 1 GB;
- swap 交换空间: 一般取 1.5 ~ 2 倍物理内存。

因为我们只是实验的目的，所以只划分了根分区、/home 分区和交换分区，分区工具采用 Disk Druid。

(1) 根分区。根分区的大小可以根据磁盘的具体情况来设置，本例中根分区的大小为 2 GB，分区类型为 Linux Native 文件系统，装载点为“/”。所谓装载点是指分区在 Linux 目录中的位置，如图 1-8 所示。

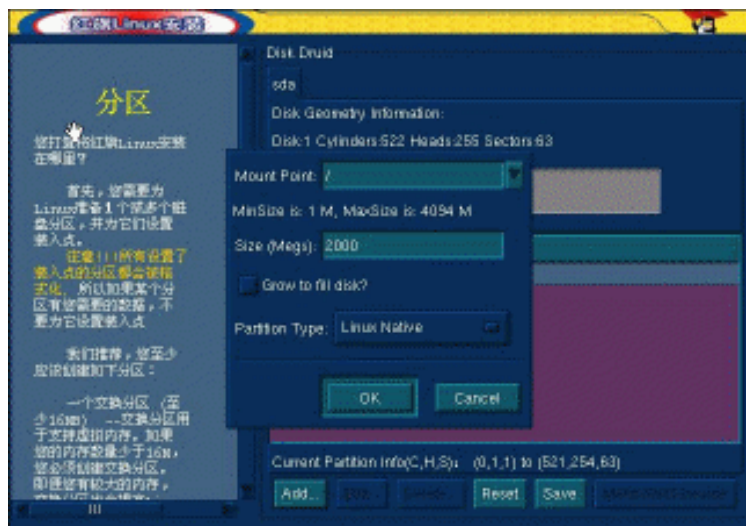


图 1-8 设置根分区

Disk Druid 最下面一排是它的功能按钮，这些按钮控制 Disk Druid 的行为，它们用来增加和删除分区，或者修改分区的属性。另外，还有按钮用来接受你所做的改变或者退出 Disk Druid，最为常用的是下面五个：

Add：用来申请一个新的分区。选择后，会出现一个对话框，包含一些你必须输入的区域。

Edit：用来修改当前激活的分区的属性。选择后，将出现一个对话框，根据分区信息是否已经写到硬盘上，你可以修改“Edit Partition”对话框中的某些或全部信息。

Delete：用来删除“Current Disk Partitions”区域中当前激活的分区。选择这个按钮会出现一个对话框，提示你确认删除。

Save：将把你所做的任何修改写入硬盘。在 Disk Druid 重写到你的硬盘分区表之前会要求你确认所做的修改。另外，你所定义的载入点也会传给安装程序，Linux 系统会使用这些系统来定义文件系统的规划。

Reset：使 Disk Druid 不保存你做的任何修改而退出。当选择这个按钮时，安装程序会退到前一个屏幕，重新开始。

(2) 交换分区。Linux 需要一个专门的交换分区。它是在物理内存用尽时操作系统要利用的虚拟内存。分区类型设置为 Linux Swap，不指定装载点，大小则设置为 512 MB，如图 1-9 所示。

/home 的分区方法和根分区的方法大致相同，三个分区添加完之后我们就完成了分区，如图 1-10 所示，保存后可进入下一步。

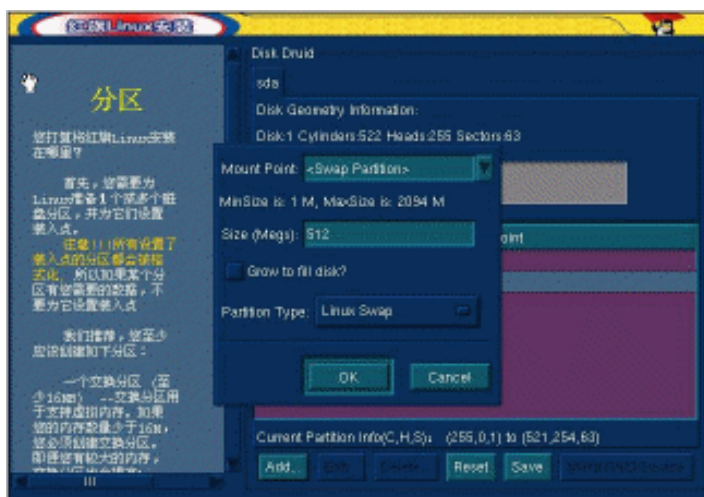


图 1-9 设置交换分区



图 1-10 Linux 分区图

8. 选择要格式化的分区

对于第一次安装 Linux 的用户来讲,所有的分区都需要格式化后才能够使用,如图 1-11 所示。



图 1-11 选择要格式化的分区

9. 配置账户

红旗 Linux Server 3.0 在安装过程中需要设置 root 密码(大小写敏感)和至少一个用户名,

如图 1-12 所示。一个好的用户口令至少要有 6 个字符长，不要使用个人信息，例如：生日、名字等。普通的英文单词也比较危险，这种口令可用字典攻击法在极短的时间内破解。用户的口令中最好有一些非字母(如数字、标点符号、控制字符等)，同时还要易于记忆。选择用户的口令时，一个好的方法是将两个不相关的词用一个数字或控制字符相连，下面的口令可以认为是好的口令：

thank_do

12%fas3q

虽然口令可以任意长，但只有前面 8 个字符有效。

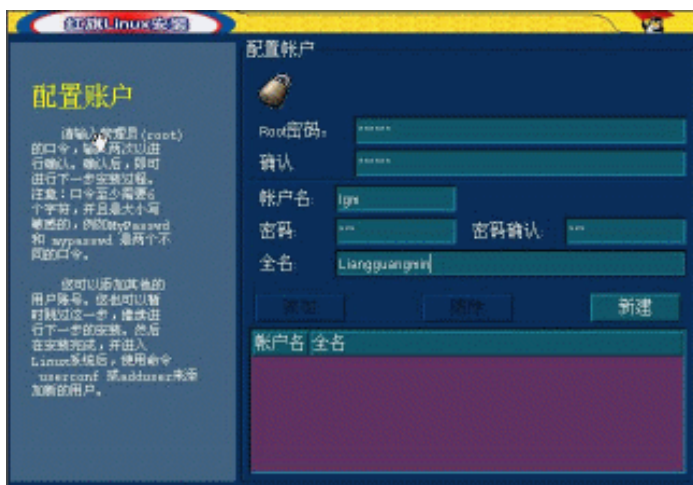


图 1-12 配置账户

10. 配置高级选项

配置高级选项包括 LILO、X Window 和 Network，其中 LILO 最为重要。LILO 的配置结果如图 1-13 所示。

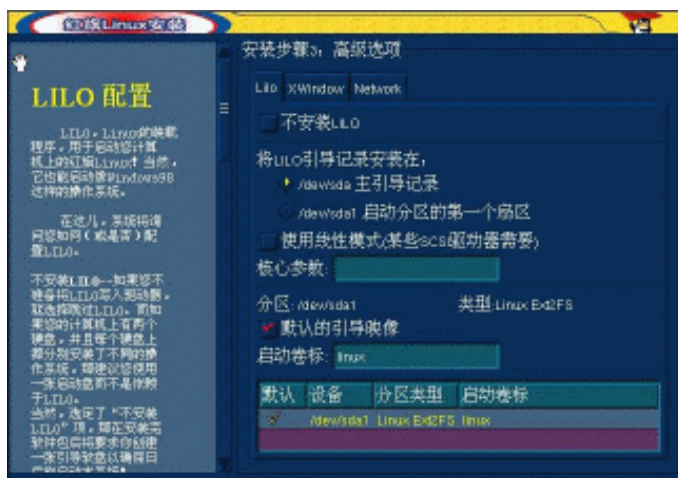


图 1-13 配置 LILO

LILO 是 Linux 的启动装载器, 要让系统顺利启动, 就要配置和安装 LILO。LILO 提供了双引导特性, 可以在引导时选择启动哪个操作系统, 这样就可以访问 Linux 和现有的 DOS 与 Windows 系统。屏幕会询问我们是否安装以及安装到哪里, 我们当然选择安装 LILO。LILO 安装的位置可以是主引导记录(MBR), 也可以是启动分区的第一个扇区。我们选择 LILO 安装在主引导记录(MBR)(这是最为简单的一种方法)。接下来设置启动卷标(启动计算机时用来选择所启动操作系统的名称), 我们设为“Linux”。最后, 把 Linux 设置为默认的引导映像(在启动菜单上默认的选项)。必须指出的是, LILO 配置不当会带来非常大的麻烦。

11. 检查安装选项

当上面的步骤完成后, 安装选项会显示一张表格让用户确认, 如图 1-14 所示。

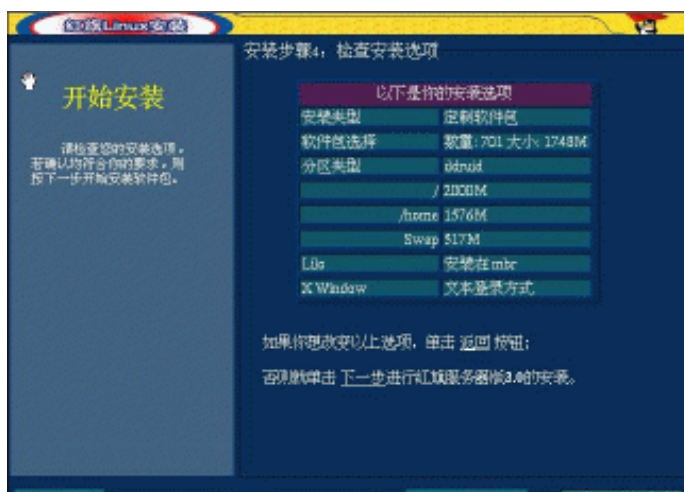


图 1-14 检查安装选项

如果没有问题, 就可以安装软件包了。

12. 安装软件包

安装软件包的过程如图 1-15 所示, 这一过程大约持续 20~30 分钟, 不需要人工干预。



图 1-15 Linux 安装软件包

13. 创建引导盘

在软件包安装完以后, 系统会提示是否创建引导盘。如果创建, 插入空白软盘即可; 如果不创建, 可以选择跳过。

14. 重新引导系统

最后，程序提示取下计算机上的软盘并重新引导系统。如果 Linux 是缺省操作系统，则系统引导到 Linux，在“boot:”提示下有 5 秒钟时间，允许选择另一种操作系统。

如果你选择另一种操作系统作为缺省引导系统，则在“boot:”提示下要输入 Linux 分区的引导卷标，这样才能引导到 Linux。

如果一切顺利，则可以看到 RedFlag 登录屏幕，用 root 用户名和前面安装时设定的口令登录，画面如图 1-16 所示。

```
Redflag Linux release 3.0
Kernel 2.4.17-1 on i686
localhost login: root
Password:
Last login: Wed Apr 23 16:10:49 on tty1
[root@localhost /root]#
```

图 1-16 红旗 Linux 启动画面

Linux 允许多次登录，即使在初始字符方式下也可以。利用【Ctrl】+【Alt】+【F1】到【Ctrl】+【Alt】+【F6】的组合键，可以在最多六个虚拟控制台之间切换。每个虚拟控制台应分别登录，可以登录为不同用户，进行不同工作，或作为同一用户登录不同虚拟控制台，这种 Unix 和 Linux 特性使其能在灵活的环境中工作。

1.3 LILO 的配置和使用

1.3.1 LILO 简介

LILO(Linux Loader)是 Linux 自带的一个优秀的引导管理器，使用它可以很方便地引导一台机器上的多个操作系统。与其他常用的引导加载程序相比，LILO 引导方式显得更具有艺术性，对其深入的理解，将有助于我们方便地处理多操作系统、网络引导、大硬盘及大内存等诸多棘手的问题。LILO 的优点如下：

(1) 作为操作系统的装载程序，LILO 独立于任何操作系统，它只使用了计算机的基本输入输出系统 BIOS。也就是说，即使不安装 Linux 操作系统，用户仍然可以在 DOS、Unix、OS/2 和 Windows 系统环境下使用 LILO，完成多个操作系统的启动任务。

(2) 可以覆盖硬盘的主引导分区。

(3) 可以同时支持 16 个不同的系统内核映像。

(4) 为每个系统内核映像提供了密码保护。

(5) 支持位于不同磁盘和分区中的引导扇区、映像文件和启动映像。

1.3.2 LILO 配置

一般地，LILO 使用一个文本文件/etc/lilo.conf 作为其配置文件。LILO 读取 lilo.conf，

按照其中的参数将特定的 LILO 写入系统引导区,任何时候修改了/etc/lilo.conf,都必须重新运行 lilo 命令,以保证 LILO 正常运行。lilo.conf 使用的配置参数很多,配置起来也相当复杂。

lilo.conf 文件中的配置参数分为两部分,一部分是全局参数,另一部分是引导映像参数。与 Linux 系统中其他的配置文件一样,“#”号后的一行文字表示注释。全局参数是全程有效的,它可以出现在文件 lilo.conf 中的任何地方。引导映像参数作用于每一个引导映像区。如果某一引导映像参数(如 password)与全局参数的定义相抵触,则以该引导映像参数的定义为准,但仅限于该引导映像区。

下面我们用一个具体的实例来讲述 LILO 的配置。

某台主机安装了 Linux 和 Windows 2000 操作系统,/etc/lilo.conf 的内容如下(为了方便大家的学习和理解,我们采用逐行注释):

```
boot=/dev/hda
#指定装有启动扇区的设备名,如果省略该项,则使用默认作为根文件系统的设备。

map=/boot/map
#指定 LILO 使用的映像文件,如果不指定,则该项默认使用/boot/map。

install=/boot/boot.b
#指定使用的启动文件为 boot.b。

LBA32

prompt
#迫使 LILO 程序进入提示符状态,如果不设置该项,则需要按【Alt】【Shift】或【Ctrl】键,
LILO 程序才进入提示符状态;如果设置该项,但不设置 timeout 项,LILO 程序将一直停留在
提示符状态下。

timeout=50
#超时时长为 5 秒,单位为 0.1 秒。如果为系统内核设置了密码,那么在这段时间内应完成密码
的输入。

default=linux
#缺省引导 label 为 Linux 的操作系统。

image=/boot/vmlinuz-2.4.17-1
#设置 Linux 核心引导映像。

label=Linux
#标识为 Linux。

read-only
#LILO 以只读方式载入根文件系统。

root=/dev/hda7
#指定内核映像文件存在的分区。

restricted
#与“password”联用,使“password”仅作用于在 LILO 提示后有命令行输入的时候。

password=linux
```

#为 LILO 设置口令保护，每次重新启动计算机时提示用户输入口令。设置了口令后，建议将 lilo.conf 的文件属性改为 600，以免让非 root 用户看到口令。

other=/dev/hda1

#DOS 分区为第一个 IDE 硬盘的第一分区。

label=dos

#标识为 dos。

【说明】修改完/etc/lilo.conf 文件后，一定要执行/sbin/lilo 命令，并且重新启动系统后才能够生效。

1.3.3 LILO 提示信息

LILO 在运行时会给出一些提示信息，了解它的含义对我们正确配置 lilo.conf 或查找硬件错误是有帮助的。

当 LILO 装入它自己的时候，显示单词“LILO:”，每完成一个特定的过程显示一个字母。如果 LILO 在某个地方失败了，屏幕上就停留几个字母，以指示错误发生的地方。

注意，如果磁盘发生瞬间故障，可能会在第一个字母“L”后插入一些十六进制数字(磁盘错误码)。除非 LILO 停在那里并不停地产生错误码流，否则并不说明有严重问题。以下是 LILO 运行时常见的一些提示信息及其含义。

(1) L 错误码：LILO 的第一部分已经被装入并运行了，但它不能装入第二部分的引导程序。两位数字的错误码指示问题的类型，这种情况通常是介质访问失败或硬盘参数错误。

【说明】错误码通常是两位十六进制数，表示特定的含义。例如，

0x04：表示扇区未找到，典型的原因是硬盘参数错误；

0x40：表示定位失败，这可能是介质问题，重新启动试试；

0x80：表示磁盘超时，可能是磁盘或驱动器没有准备好、介质坏了或磁盘没有转。

(2) LI：LILO 第一部分正确但是第二部分执行时出错。这一般是硬盘参数有误或/boot/boot.b 被移动后没有重新运行 map 安装程序。

(3) LIL：LILO 第二部分开始执行，但是不能从“map”文件中读取描述符表(descriptor table)。这通常是由介质错误或磁盘参数有误引起的。

(4) LIL?：LILO 在错误的地方加载。原因与“LI”大致相同。

(5) LIL-：描述符表(descriptor table)错误。典型原因是硬盘几何参数的不匹配或/boot/boot.b 被移动而没有运行 map 安装程序。

(6) LILO：LILO 执行正确。

本章小结

Linux 是 20 世纪 90 年代迅速发展起来的，与其他操作系统相比较，Linux 有着无与伦比的优势。本章首先介绍了 Linux 的基本概念、发展史、优点及结构等内容，接着重点介绍了红旗 Linux 的特点和安装，最后介绍了 LILO 的配置。

习 题

1. Linux 核心 1.0 发布时间为_____。
A. 1991 B. 1993 C. 1992 D. 1994
2. Linus 最早是由_____人 Linus Torvalds 编写的。
A. 芬兰 B. 荷兰 C. 法国 D. 美国
3. 炎黄中文 KDE 环境下, 启动输入法的程序是_____。
A. yh B. rinput C. finput D. abc
4. 一般来说, 使用 Fdisk 命令的最后一步是使用____选项命令将改动写入硬盘的当前分区表中。
A. p B. r C. x D. w
5. 如果我们需要在 Windows 或 DOS 环境下制作 Linux 的启动盘, 这时应该利用红旗 Linux Server 光盘上的_____文件。
A. auto.bat B. boot.img C. config.sys D. pcmcia.img
6. LILO 启动的时候, 出现如下的信息 “LI”, 说明_____。
A. LILO 第二部分已经加载
B. LILO 在第二部分出错的地方加载
C. 用户的分区情况改变, 没有重新安装 LILO
D. 第一部分加载, 第二部分出错
7. GPL 指的是_____。
A. 通用公共许可证
B. 对推动自由软件发展起了重要的作用
C. 保证任何人有共享和修改自由软件的自由, 任何人有权取得、修改和重新发布自由软件的源代码, 并且规定在不增加附加费用的条件下得到源代码
D. 规定自由软件的衍生作品继续保持自由状态, 并且用户在扩散 GNU 软件时, 必须让下一个用户也有获得源代码的权利
8. 通常 Linux 的安装至少需要两个分区, 分别是_____。
A. 根分区 B. /home C. /usr D. 交换分区
9. 在安装红旗 Linux 时可以使用的分区工具包括_____。
A. fdisk B. disk druid C. fsck D. mkfs
10. Linux 操作系统的架构包括_____。
A. 内核 B. Shell C. 文件结构 D. 实用工具

第2章 常用的 Linux 命令

使用过 Windows 系统的人都已经习惯了在图形环境下利用鼠标来管理系统。Linux 系统也有图形环境，但是在类似 DOS 命令行的字符界面下管理系统的机会更多，因为字符界面的功能更加强大、灵活。要学好 Linux 并成为专家，掌握好命令行下的 Linux 命令将是一个必经的挑战。

Linux 命令非常之多，以致 Linux 高手也常常只能掌握一部分命令，并且每一条命令也只用常用的参数选项，因此我们这里只介绍常用的 Linux 命令和常用的参数选项。Linux 命令非常简洁，这让初学者感到有点不太适应。为了便于记忆，我们把命令进行了分类，究竟把某一命令划分到哪一类，难以做到绝对的合理，只要读者能够记住就达到分类的目的了。部分 Linux 命令(如用户管理、设备管理、文件系统管理等)将放在本书以后的章节专门介绍。

Linux 命令像 DOS 命令一样，也分内部命令和外部命令。内部命令是 Shell 内置的命令，不需要在磁盘存有相应的可执行文件，而外部命令实际上就是一个程序或可执行文件，只不过系统安装时它们已经被安装好，成为 Linux 系统中非常重要的一部分而已。我们介绍的大部分是外部命令，这里不再区分命令的类别。

通常 Shell 具有命令补全功能，用户可以不把命令输全。只要 Shell 能区分出用户要输入的命令，用户就可以用【Tab】键让 Shell 帮助完成要输入的命令。特别是在文件名、目录名及命令名很长时，补全功能可以减少用户的击键次数。

【实例 2.1】

```
[root @redflag /root]#hist【Tab】
```

系统将会自动帮助用户完成命令：

```
[root @redflag /root]#history
```

用户学会使用【Tab】的用法，可以提高自己的工作效率。

此外，Linux 的命令是区分大小写的，通常 Linux 命令是小写的。

2.1 文件和目录操作命令

要学会使用 Linux，首先要掌握的就是基本的文件和目录操作命令。

2.1.1 pwd、cd

1. pwd——显示(打印)用户当前所处的目录

这是再常用不过的命令了，如果不知道自己当前所处的目录，就必须使用它。这个命令和 DOS 下的不带任何参数的 cd 命令的作用是一样的。其用法如下：

```
[test @redflag test]$pwd
```

```
/home/test
```

说明当前目录是/home/test。

2. cd 目录名——改变当前所处的目录或处理绝对目录和相对目录

如果用户当前处于/bin 目录，想进入/etc 目录，可以键入：

```
[test @redflag /bin]$cd /etc
```

【说明】Linux 系统中：用“.”代表当前目录；用“..”代表父目录；用“~”代表用户的个人主目录。例如，root 用户的个人主目录是/root，不带任何参数，则“cd”命令相当于“cd ~”。初学者常常不习惯，因为 DOS 下的“cd”是显示当前目录，而不是改变目录。例如：

```
[test @redflag dir1]$cd ../test
```

执行后则进入上一级的目录下的 test 目录(即和当前目录平行的目录 test)。

2.1.2 ls、tree

1. ls [参数] 路径或文件名——列出文件或子目录的信息

参数选项：

-a：显示所有的文件，包括以“.”开头的文件(即隐含文件)。

-l：以长格式显示文件或子目录的信息。

-i：显示每个文件的索引(节点)号。

执行命令[test @redflag test]\$ls -a 显示当前目录下的所有文件，输出：

```
bak chap1.txt Desktop txt
```

Linux 系统用颜色来区分文件类别。缺省时，蓝色代表目录，绿色代表可执行文件，红色代表压缩文件，浅蓝色代表链接文件，灰色代表其他文件。

【说明】之所以会用不同颜色来显示，是因为 Linux 在用户的脚本里定义了别名：
ls = 'ls - -color'。

```
[test @redflag test]$ls -l
```

```
drwxrwxr-x 2 longkey longkey 48 20A 24 22:23 bak
```

```
-rw-rw-r-- 1 longkey longkey 16 20A 24 22:23 chap1.txt
```

```
lrwxrwxr-x 1 root root 12 20A 24 22:23 Desktop->.Desktop-gb/
```

```
drwxrwxr-x 2 longkey longkey 48 20A 24 22:23 txt
```

我们来解析显示结果中各部分的含义，以最后一行为例，输出结果分 7 组(以横线标识)，依次是：文件类别和文件权限、链接数、文件拥有者、文件所属组、文件大小、文件创建或修改时间、文件名。第一组的第 1 个字符代表文件类别，第一组的第 2~10 个字符是文件权限。我们先介绍文件类别，其余部分的含义随后介绍。Linux 系统用“-”代表普通文件，“d”代表目录，“l”代表符号链接，“c”代表字符设备，“b”代表块设备。本例中 chap1.txt 是普通文件，而 bak 和 txt 是目录，Desktop 则是链接文件。

【注意】在红旗 Linux 3.0 版 ls 命令的输出中，时间字段是中文，如果不在炎黄汉字状态下，月份会呈现乱码，如以上的“20A 24 22:23”。

2. tree 目录名——以树的形式显示指定目录下的内容

```
[test @redflag test]$tree
```

这是不带任何参数的 tree 命令，以树的形式显示当前目录下的文件和子目录，会递归到各子目录。例如：

```
[test @redflag test]$tree /etc/rc.d
```

以树的形式显示目录/etc/rc.d 下的文件和子目录。

2.1.3 mkdir、rmdir

1. mkdir [参数] 目录名——建立目录

目录可以是绝对路径，也可以是相对路径。

参数选项：

-p：建立目录时，如果父目录不存在，则此时可以与子目录一起建立。

例如：

```
[test @redflag test]$mkdir dir1
```

在当前目录下建立 dir1 目录。

```
[test @redflag test]$mkdir -p dir2/bak
```

在 dir2 目录下建立 bak 目录，如果 dir2 目录不存在，那么同时建立 dir2 目录。

2. rmdir [参数] 目录名——删除目录

目录同样可以是绝对路径，也可以是相对路径。

参数选项：

-p：一起删除父目录时，父目录下应无其他目录。

例如：

```
[root @redflag /root]#rmdir test
```

删除当前目录下的 test 目录。删除目录时，被删除的目录下应无文件或目录存在。

```
[root @redflag /root]#rmdir -p longkey/test
```

删除当前目录下的 longkey/test 目录。删除目录 test 时，如果父目录 longkey 下无其他内容，则一起删除 longkey 目录。

2.1.4 cp、rm、mv、ln

1. cp [参数] 源文件 目标文件——拷贝文件或目录

相当于 DOS 下的 copy 命令。

参数选项：

-f：如果目标文件或目录存在，先删除它们再拷贝(即覆盖)，并且不提示用户。

-i：如果目标文件或目录存在，提示是否覆盖已有的文件。

-R：递归复制目录，即包含目录下的各级子目录。

例如：

```
[test @redflag test]$cp /etc/inittab ~/inittab.bak
```

把/etc 目录下的文件 inittab 拷贝到用户的个人主目录，拷贝后的文件名为 inittab.bak。

```
[test @redflag test]$cp -R /etc/rc.d /home/longkey
```

把/etc 下的目录 rc.d(含 rc.d 下的文件及子目录)拷贝到/home/longkey 目录下。

2. rm [参数] 文件名或目录名——删除文件或目录

相当于 DOS 下的 del 命令。

参数选项：

-f：删除文件或目录时不提示用户。

-i：删除文件或目录时提示用户。

-R：递归删除目录，即包含目录下的文件和各级子目录。

例如：

```
[test @redflag test]$rm *
```

删除当前目录下的所有文件，但子目录和以“.”开头的文件(即隐含文件)不删除。

```
[test @redflag test]$rm -iR bak
```

删除当前目录下的子目录 bak，包含其下的所有文件和子目录，并且提示用户确认。

3. mv [参数] 源文件或目录 目标文件或目录——移动文件或目录

相当于 DOS 下的 move 命令。

参数选项：

-i：如果目标文件或目录存在时，提示是否覆盖目标文件或目录。

-f：不论目标文件或目录是否存在，均不提示是否覆盖目标文件或目录。

值得注意的是，mv 可以用来更改文件名或目录名。

例如：

```
[test @redflag test]$mv 1.txt 2.txt
```

这里移动文件时并不改变文件的目录，如果 2.txt 原来不存在，则实际上是 1.txt 更名为 2.txt。

```
[test @redflag test]$mv ~/txtbak /bak
```

把个人主目录下的目录 txtbak 移动到/bak 目录下。

4. ln [参数] 源文件或目录 链接名——建立链接

参数选项：

-s：建立符号链接(即软链接)，不加该项时建立的是硬链接。

例如：

```
[test @redflag test]$ln telno.txt telno2.txt
```

给源文件 telno.txt 建立一个硬链接 telno2.txt，这时 telno2.txt 可以看作是 telno.txt 的别名，它和 telno.txt 不分主次。telno.txt 和 telno2.txt 实际上都指向硬盘上的相同位置，使用 telno.txt 作为文件名所做的更改，会在 telno2.txt 得到反映。硬链接有局限性，不能建立目录的硬链接。

【提示】 删除文件时，只有所有的链接全部删除时，文件或目录才被删除。


```
[test @redflag test]$ln -s telno.txt telno2.txt
```

建立软链接 `telno2.txt` 指向 `telno.txt`。软链接更像 Windows 中的快捷方式，也可以比喻为指向 `telno.txt` 的指针。改变 `telno.txt` 的权限，在 `telno2.txt` 上得不到反映，但需要注意的是，改变 `telno.txt` 的内容，在 `telno2.txt` 上却可以得到反映。

2.1.5 chmod、chown、chgrp

1. chmod 模式 文件或目录名——改变文件或目录的访问权限

Linux 系统是个多用户系统，应该能做到不同的用户能同时访问不同的文件，因此一定要有文件权限控制机制。Linux 系统的权限控制机制和 Windows 的权限控制机制有着很大的差别。Linux 的文件或目录都被一个用户拥有时，这个用户称为文件的拥有者(或所有者)，同时文件还被指定的用户组所拥有，这个用户组称为文件所属组。要说明的是，一个用户可以是不同组的成员，这可以由管理员控制，我们将在用户管理这一章介绍如何控制的问题。文件的权限由权限标志来决定，权限标志决定了文件的拥有者、文件的所属组、其他用户对文件访问的能力。可以使用“`ls -l`”命令来显示权限标志。例如：

```
[test @redflag test]$ls -l
-rw-rw-r-- 1 longkey root 16 20A 24 22:23 chap1.txt
```

本例中，文件 `chap1.txt` 的拥有者是 `longkey`，所属组是 `root`。这里我们特别关心的是输出行前面的第 1~10 个字符。第 1 个字符代表文件类别，第 2~4 个字符“`rw-`”是文件拥有者的权限，第 5~7 个字符“`rw-`”是文件所属组的权限，第 8~10 个字符“`r--`”是其他用户(即除了 `root` 用户和 `longkey` 用户组里的用户之外的用户)文件拥有者的权限。而权限均用三个字符表示，依次为读(`r`)、写(`w`)、执行(`x`)，如果某一位为“-”，则表示没有相应的权限，例如：“`rw-`”表示有读、写的权限，没有执行的权限。在本例中，文件拥有者 `longkey` 用户对文件有读、写的权限，`root` 组的所有用户对文件也有读、写的权限，而其他用户对文件只有读的权限。

设定文件权限时，在模式中常用以下的字母代表用户或用户组：

u——文件的拥有者；

g——文件的所属组；

o——其他用户；

a——代表所有用户(即 `u+g+o`)。

权限用以下字符表示：

r——读权限；

w——写权限；

x——执行权限；

最后要指明是增加(+)还是减少(-)权限，或是绝对权限(=)。

【实例 2.2】

```
[root @redflag /root]#chmod o+w chap1.txt
```

`chap1.txt` 的权限由原来的“`rw-rw-r--`”变为“`rw-rw-rw-`”，表示增加其他用户对文件的写权限。

【实例 2.3】

```
[root @redflag /root]#chmod u=rw,g=rw,o=r chap1.txt
```

chap1.txt 的权限变为“`rw-rw-r---`”，不论原来的权限是什么，这表示拥有者对文件有读、写的权限，所属组的用户对文件也有读、写的权限，而其他用户只有读的权限。

我们在以上设置权限时，用字符表示权限和用户，实际上我们也经常使用八进制来表示。读、写、执行依次各自对应一个二进制位“`???`”，如果某位为“`0`”，则表示无权限；如果某位为“`1`”，则表示有权限。例如：文件权限为 `r--w---x` 时，用二进制表示为 `100010001`，用八进制可以表示为 `421`。例如：

```
[root @redflag /root]#chmod 664 chap1.txt
```

等同于：

```
[root @redflag /root]#chmod u=rw,g=rw,o=r chap1.txt
```

【说明】 对于文件和目录来说，读、写、执行权限的含义是不相同的。目录的读权限表示可以列出目录中的文件，写权限表示可以在目录下创建新的文件或新的子目录，执行权限表示可以切换到该目录(用 `cd` 命令)。

2. chown 用户名 文件或目录名——改变文件(或目录)的拥有者或所属组

例如：

```
[root @redflag /root]#chown longkey chap1.txt
```

把文件 chap1.txt 的拥有者改为 longkey 用户。

```
[root @redflag /root]#chown longkey:root chap1.txt
```

把文件的拥有者改为 longkey 用户，同时文件的所属组改为 root 组。

【注意】 改变文件(或目录)的拥有者或所属组后，文件(或目录)究竟可以被哪个用户访问也会发生变化。

3. chgrp 组 文件或目录——改变文件或目录的所属组

chown 可以同时改变文件拥有者和所有者，chgrp 只具有改变所属组的功能。例如：

```
[root @redflag /root]#chgrp root chap1.txt
```

文件 chap1.txt 的所属组设为 root 组。

【注意】 同样，改变文件(或目录)的所属组后，文件(或目录)究竟可以被哪个用户访问也将会发生变化。

2.1.6 find、grep

1. find 路径 匹配表达式——查找文件所在的目录

路径可以是多个路径，路径之间用空格隔开。查找时，会递归到子目录。

匹配表达式：

-name：指明要查找的文件名，支持通配符“`*`”和“`?`”。

-user username：查找文件的拥有者为 username 的文件。

-group grpname：查找文件的所属组为 grpname 的文件。

-atime n：指明查找前 n 天访问过的文件(仅第 n 天这一天)。

-atime +n：指明查找前 n 天之前访问过的文件。

-atime -n：指明查找前 n 天之后访问过的文件。

-size n：指明查找文件大小为 n 块(block)的文件。

-print：搜索结果输出到标准设备。

例如：

```
[root @redflag /root]#find / -name passwd -print
```

从根目录起查找名为 passwd 的文件，并把结果输出到标准设备。

```
[root @redflag /root]#find /home /etc -user longkey -print
```

在目录/home 和目录/etc 中查找 longkey 用户所拥有的文件。

2. grep [参数] 要查找的字符串 文件名——查找文件中包含有指定字符串的行

参数选项：

-num：输出匹配行前后各 num 行的内容。

-b：显示匹配查找条件的行距离文件开头有多少字节。

-c：显示文件中包含有指定字符串的行的个数，但不显示内容。

例如：

```
[root @redflag /root]#grep -2 Hello! chap.txt
```

在文件 chap1.txt 中查找所有含有字符串“Hello!”的行，如果找到，显示该行及该行前后各 2 行的内容。文件名可以使用通配符*和?，如果要查找的字符串带空格，可以使用单引号或双引号括起来。

【提示】 grep 和 find 的差别在于 grep 是在文件的内容中查找，而 find 是根据文件名或文件的创建时间等信息查找。它们都是经常用到的命令。

```
[root @redflag /root]#grep -b find chap2.txt
```

在文件 chap2.txt 中查找含有字符串“find”的行，如果找到，输出它所在的行距离文件开始的位置和所在行的内容。

2.1.7 cmp、diff

1. cmp [参数] 文件 1 文件 2——比较两个文件内容的不同

参数选项：

-l：列出两个文件的所有差异，缺省时，发现第一处差异后就停止。

例如：

```
[test @redflag test]$cmp 1.txt 2.txt
1.txt 2.txt differ:char10,line2
```

说明从文件开头起的第 10 个字符(在第 2 行)不同，如果文件相同，则没有反应。

2. diff [参数] 源文件 目标文件——比较两个文件内容的不同

参数选项：

-q：仅报告是否相同，不报告详细的差异。

-i：忽略大小写的差异。

diff 命令的输出表示文件有哪些差别，如果要使文件相同，应该采取怎样的动作。由于其输出常常太复杂，以致于 diff 命令不太实用。我们不详细介绍输出的含义，有兴趣的读者可以用“diff --help”命令来获得详细的说明。

例如：

```
[root @redflag /root]#diff file1 file2
```

```
Files file1 and file2 differ
```

说明文件 file1 和 file2 不相同，如果文件相同，则没有反应。

【技巧】 diff 命令和 cmp 命令的区别在于两者比较文件的方式不同：diff 是逐行比较，而 cmp 是以字符为单位进行比较的。cmp 用于比较二进制文件时会更实用。

2.1.8 stat、touch

1. stat 文件名——显示文件或目录的各种信息

例如：

```
[test @redflag test]$stat /etc/passwd
```

```
File: "/etc/passwd"
```

```
Size: 1323
```

```
Blocks: 8
```

```
Regular File
```

```
Device: 301h/769d
```

```
Inode: 111261
```

```
Links: 1
```

```
Access: (0644/-rw-r--r--)
```

```
Uid: ( 0/ root)
```

```
Gid: ( 0/ root)
```

```
Access: Thu Feb 27 23:18:00 2003
```

```
Modify: Mon Feb 24 22:22:28 2003
```

```
Change: Mon Feb 24 22:22:28 2003
```

显示文件 passwd 的被访问时间、修改时间、变更时间、文件大小、文件所有者、所属组、文件权限等项内容。

2. touch [参数] 文件或目录名——修改文件的存取和修改时间

参数选项：

-d yyyymmdd：把文件的存取/修改时间改为 yyyymmdd。

-a：只把文件的存取时间改为当前时间。

-m：只把文件的修改时间改为当前时间。

例如：

```
[test @redflag test]$touch *
```

把当前目录下的所有文件的存取和修改时间改为当前系统的时间。

```
[test @redflag test]$touch -d 20030224 chap1.txt
```

把文件 chap1.txt 的存取和修改时间改为 2003 年 2 月 24 日。

```
[test @redflag test]$touch test.txt
```

把 test.txt 的存取和修改时间改为当前系统的时间，如果 test.txt 文件不存在，则生成一个空文件(即 0 字节的文件)。

touch 还有另外一种形式：

touch MMDDhhmm[YY] 文件名

例如：

```
[test @redflag test]$touch 0102120099 chap1.txt
```

把 chap1.txt 文件的存取和修改时间改为 1999 年 01 月 02 日 12:00。

2.2 显示命令

2.2.1 cat、more、less

1. cat 文件名 1 文件名 2 ——显示文件的内容

相当于 DOS 下的 type 命令。

例如：

```
[test @redflag test]$cat chap1.txt chap2.txt
```

把文件 chap1.txt、chap2.txt 在标准的输出设备(通常是显示器)上显示出来。

2. more 文件名——逐页显示文件中的内容

如果文件太长，用 cat 命令只能看到文件的最后一页，而用 more 命令时可以一页一页地显示。执行 more 命令后，进入 more 状态，用【Enter】键可以向后移动一行；用【Space】键可以向后移动一页；用“q”键可以退出。在 more 状态下还有许多功能，可用 man more 命令获得。

3. less 文件名——逐页显示文件中的内容

less 实际上是 more 的改进版，其命令的直接含义是 more 的反义。less 的功能比 more 更灵活。例如：用【Pgup】键可以向前移动一页，用【Pgdn】键可以向后移动一页，用向上光标键可以向前移动一行，用向下光标键可以向后移动一行。“q”键、【Enter】键、【Space】键的功能和 more 类似。

2.2.2 head、tail

1. head [参数] 文件名——显示文件的前几行

参数选项：

-n num：显示文件的前 num 行。

-c num：显示文件的前 num 个字符。

缺省时，head 显示文件的前 10 行。

例如：

```
[test @redflag test]$head -n 20 chap1.txt
```

显示文件 chap1.txt 的前 20 行。

2. tail [参数] 文件名——显示文件的末尾几行

参数选项：

-n num：显示文件的末尾 num 行。

-c num：显示文件的末尾 num 个字符。

tail 命令和 head 命令相反，它显示文件的末尾。缺省时，tail 命令显示文件的末尾 10 行。

例如：

```
[test @redflag test]$tail -n 20 chap1.txt
```

显示文件 chap1.txt 的末尾 20 行。

2.2.3 sort、uniq

1. sort [参数] 文件列表——将文件中的内容排序输出

参数选项：

-r：反向排序。

-o filename：把排序的结果输出到文件 filename。

如果文件 a.txt 的内容为

```
b
c
a
d
a
```

则执行 sort a.txt 命令后的显示结果为

```
a
a
b
c
d
```

例如：

```
[test @redflag test]$sort -o c.txt a.txt
```

把 a.txt 文件的内容排序，并输出到文件 c.txt。

```
[test @redflag test]$sort a.txt b.txt c.txt
```

把文件 a.txt、b.txt、c.txt 的内容联合排序输出。

2. uniq 文件名——比较相邻的行，显示不重复的行

如 b.txt 文件的内容为

```
b
c
c
a
d
a
```

则执行 `uniq b.tx` 命令后的显示结果为

```
b
c
a
d
a
```

【注意】该命令只是去掉相邻的重复行，不相邻的行并不被过滤。`uniq` 常和 `sort` 命令一起使用，例如：

```
[test @redflag test]$sort b.txt | uniq
```

则把 `b.txt` 中的行排序并去掉重复的行，其结果为

```
a
b
c
d
```

2.2.4 file、locate、which

1. file 文件名或目录——显示文件或目录的类型

例如：

```
[root @redflag /root]#file /etc/passwd
```

则可能输出：

```
/etc/passwd:ASCII text
```

说明 `passwd` 是个 ASCII 文本文件。

2. locate 字符串——查找绝对路径中包含指定字符串的文件

例如：

```
[test @redflag test]$locate chap1
```

则可能输出：

```
/etc/longkey/chap1.txt
/usr/share/doc/qt-devel-2.3.0/html/designer/chap10_1.html
/usr/share/doc/qt-devel-2.3.0/html/designer/chap1_1.html
/home/longkey/chap1.txt
/root/home/longkey/chap1.txt
```

【注意】这条命令用到 `updatedb` 文件名数据库，必须确保已做好设置，能够每天或每周运行 `updatedb`。如果使用该命令提示文件名数据库有问题，可以手工运行 `updatedb`。

3. which 命令——确定程序的具体位置

例如：

```
[test @redflag test]$which find
```

则输出 `find` 命令所处的位置：

```
/usr/bin/find
```

2.3 进程管理和作业控制

Linux 是个多用户、多任务的操作系统。多用户系统是指多个用户可以同时使用同一计算机，而多任务是指系统可以同时执行多项任务。Linux 操作系统将负责管理多个用户的请求和多个任务。用户运行一个程序，就会启动一个或多个进程。用户的感觉是一个人独占系统，实际上并非如此。大多数系统只有一个 CPU 或有限的内存资源，一个 CPU 在一个时刻实际上只能运行一个进程，造成用户一个人独占系统的感觉是操作系统的功劳。操作系统控制着每一个运行着的程序(即进程)，给每一进程分配一个合适的时间片，大约有几十毫秒，每个进程轮流被 CPU 运行一段时间，然后被挂起，系统去处理另外一个进程，经过一段时间后这个进程又被运行。

所谓的程序是指程序员编写的计算机指令集，其实就是一个保存在磁盘上的文件。运行一个程序，就会在系统中创建一个或多个进程，进程可以看成是在计算机里正在运行的程序。Linux 系统启动后，就已经创建了许多进程。

本节将介绍 Linux 这一多任务系统提供的关于进程管理的命令。

2.3.1 进程的启动

进程的启动有两种方式：手工启动和调度启动。手工启动又分为前台启动和后台启动。前台启动是最常用的方式，用户直接运行一个程序或执行一个命令时就启动了前台进程。例如：用户执行“ls -l”命令就启动了一个新的前台进程，只不过这个进程可能很快就结束了。前台进程的一个特点是进程不结束，终端不出现“#”或“\$”提示符，所以用户不能再执行别的任务。后台进程的启动是用户在输入命令行后加上“&”字符，例如：

```
[root@redflag /root]#find / -name myfile -print > /root/test &
```

这就启动了一个后台进程。后台进程常用于进程耗时长、用户不着急得到结果的场合。用户启动一个后台进程后，终端会出现“#”或“\$”提示符，而不必等待进程的结束，用户又可以接着执行别的任务。

至于调度进程，是指用户事先设定好(如在某个时间)，让系统自行启动进程的方法。有关调度进程的方法将在本节稍后介绍。

2.3.2 查看系统的进程

要管理进程，首先要知道系统里有哪些进程存在及进程的状况如何。可以使用下面的命令：

ps [参数]——查看系统的进程。

参数选项：

a：显示当前控制终端的进程(包括其他用户的)。

u：显示进程的用户名和启动时间等信息。

-w：宽行输出，不截取输出中的命令行。

-l：按长格式显示输出。

x：显示没有控制终端的进程。

-e：显示所有的进程。

-t n：显示第 n 个终端的进程。

ps 命令的输出，含义如下：

USER：启动进程的用户名。

PID：进程号。

PPID：父进程的进程号。

TTY：启动进程的终端号。

STAT：进程的状态，R 表示进程正在运行，S 表示进程在睡眠，T 表示进程僵死或停止，D 表示进程处于不能中断的睡眠(通常是输入输出)。

START：进程开始的时间。

TIME：进程已经运行的时间。

COMMAND/CMD：进程的命令名。

%CPU：进程占用 CPU 总时间的百分比。

%MEM：进程占用系统内存总量的百分比。

NI：nice 的优先级。

PRI：进程的优先级。

例如：

```
[test @redflag test]$ps au
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1110	0.0	0.4	2372	1236	pts/0	S	23:17	0:00	login -- test
test	1111	0.0	0.5	2788	1360	pts/0	S	23:17	0:00	-bash
root	1144	0.0	0.4	2792	1060	pts/0	S	23:17	0:00	su root
root	1145	0.0	0.5	2792	1364	pts/0	S	23:17	0:00	bash
root	1192	0.0	0.2	2624	760	pts/0	R	23:23	0:00	ps au

以上显示当前控制终端的进程。

```
[test @redflag test]$ps -elw
```

以上以长格式显示所有的进程。

【技巧】 ps 常和重定向、管道命令一起使用，用于查找出所需的进程，例如：

```
[test @redflag test]$ps -e u | grep test
```

查找 test 用户启动的进程。

```
[test @redflag test]$ps -e | grep httpd
```

查找 httpd(Web 服务守护进程)进程的信息，如进程号等。

2.3.3 进程的控制

1. kill 命令——给进程发送信号

前台进程在运行时，可以用【Ctrl-c】来终止它。但后台进程无法用这种方法来终止，这时候可以使用 kill 命令向进程发送强制终止信号来达到目的。

例如：

```
[root @redflag /root]#kill -l
```

显示 kill 命令所能够发送的信号种类，每个信号都有一个数值对应，例如：SIGKILL 信号的值是 9，而 SIGTERM 的值是 15，SIGTERM 信号是 kill 命令默认的信号。kill 命令的格式为：

```
kill [参数] 进程 1 进程 2...
```

参数选项：

-s signal：signal 是信号类别，如：SIGKILL。

例如：

```
[root @redflag /root]#ps
PID  TTY      TIME    CMD
835   tty1      00:00:00  login
843   tty1      00:00:00  bash
1212  tty1      00:00:00  ps
[root @redflag /root]#kill -s SIGKILL 835
```

则系统退到登录界面，以上命令也可以用以下命令代替：

```
[root @redflag /root]#kill -9 835
```

2. killall -s signal 命令名——根据进程名来发送信号

参数选项：

-s signal：signal 是信号类别，如：SIGKILL。

用 kill 命令时要先用 ps 命令查出进程号，这样不是很方便。killall 可以根据进程名来发送信号。

例如：

```
[root @redflag /root]#killall -9 vim
```

终止所有 vi 会话。

【注意】执行“killall -9 vi”命令时，如果有多个 vi 会话，则会终止全部 vi 会话。因此在使用 killall 时，要注意避免终止你不想终止的会话。

3. nice 命令——以指定的优先级运行程序

Linux 系统有两个与进程有关的优先级。用“ps -l”命令可以看到两个域：PRI 和 NI。PRI 是进程实际的优先级，它是由操作系统动态计算的，这个优先级的计算机和 NI 值有关。NI 值可以被用户更改，其值范围为 -20 ~ 20。NI 值越高，优先级越低。一般用户只能加大 NI 值(即降低优先级)，只有超级用户可以减小 NI 值(即提高优先级)。NI 值被改变后，会影响 PRI。优先级高的进程被优先运行，缺省时进程的 NI 为 0。nice 命令的用法如下：

```
nice -n 程序名 以指定的优先级运行程序
```

n：NI 值，正值代表 NI 值增加，负值代表 NI 值减小。

例如：

```
[root @redflag /root]#nice --15 ps -l
```

则输出：

```

F S UID  PID  PPID  C  PRI  NI ADDR  SZ  WCHAN  TTY    TIME  CMD
100 S  0    1261 1260  0  70   0   -    593  wait4  pts/0  00:00:00 login
000 S  0    1295 1262  0  73   0   -    698  wait4  pts/0  00:00:00 su
100 S  0    1296 1295  0  76   0   -    698  wait4  pts/0  00:00:00 bash
100 R  0    1318 1296  0  78  -15  -    769  -      pts/0  00:00:00 ps

```

可以看到 ps 命令以 NI 值为-15 的优先级运行。

4. renice 命令——改变进程的优先级

运行中的进程的优先级可以被调整，注意只有 root 用户可以提高进程的优先级，一般用户只能降低优先级。renice 命令就是用来改变进程的优先级的，其用法如下：

```
renice n 进程号
```

n 为期望的进程 NI 值。

```
[root @redflag /root]#ps -l
```

```

F S UID  PID  PPID  C  PRI  NI ADDR  SZ  WCHAN  TTY    TIME  CMD
100 S  0    1261 1260  0  69   0   -    593  wait4  pts/0  00:00:00 login
000 S  0    1295 1262  0  69   0   -    698  wait4  pts/0  00:00:00 su
100 S  0    1296 1295  0  73   0   -    698  wait4  pts/0  00:00:00 bash
100 R  0    1373 1296  0  75  -15  -    769  -      pts/0  00:00:00 ps

```

```
[root @redflag /root]#renice -6 1261
```

```
[root @redflag /root]#ps -l
```

```

F S UID  PID  PPID  C  PRI  NI ADDR  SZ  WCHAN  TTY    TIME  CMD
100 S  0    1261 1260  0  69  -6   -    593  wait4  pts/0  00:00:00 login
000 S  0    1295 1262  0  69   0   -    698  wait4  pts/0  00:00:00 su
100 S  0    1296 1295  0  75   0   -    698  wait4  pts/0  00:00:00 bash
100 R  0    1380 1296  0  77   0   -    769  -      pts/0  00:00:00 ps

```

检查进程号为 1261 的进程优先级是否发生了改变。

5. top 命令——实时监控进程程序

和 ps 命令不同，top 命令可以实时监控进程状况。top 屏幕自动每 5 秒刷新一次，也可以用 “top -d 20”，使得 top 屏幕每 20 秒刷新一次。top 的屏幕输出如下：

```
11:50pm up 1:06, 4 users, load average: 0.02, 0.01, 0.00
```

```
50 processes: 49 sleeping, 1 running, 0 zombie, 0 stopped
```

```
CPU states: 0.0% user, 0.0% system, 0.0% nice, 99.9% idle
```

```
Mem: 255212K av, 167244K used, 87968K free, 92K shrd, 42504K buff
```

```
Swap: 771112K av, 0K used, 771112K free 31712K cached
```

```
PID USER PRI NI SIZE RSS SHARE STAT %CPU %MEM TIME COMMAND
```

```

1 root 8 0 536 536 464 S 0.0 0.2 0:04 init
2 root 9 0 0 0 0 SW 0.0 0.0 0:00 keventd
3 root 9 0 0 0 0 SW 0.0 0.0 0:00 kapm-idled
4 root 19 19 0 0 0 SWN 0.0 0.0 0:00 ksoftirqd_CPU0

```

```

.....
579 snort    9   0 5452 5444 1176 S    0.0  2.1   0:00 snort
606 named    9   0 3836 3836 2176 S    0.0  1.5   0:00 named

```

top 屏幕前五行的含义如下：

第一行：正常运行时间行。显示系统当前时间，系统已经正常运行的时间，系统当前用户数，最近 1 分钟、5 分钟、15 分钟准备运行的进程平均数。

第二行：进程统计数。显示当前的进程总数，睡眠的进程数，正在运行的进程数，僵死的进程数，暂停的进程数。

第三行：CPU 统计行。包括用户进程，系统进程，修改过 NI 值的进程，空闲进程各自使用 CPU 的百分比。

第四行：内存统计行。包括内存总量，已用内存，空闲内存，共享内存，缓冲区的内存总量。

第五行：交换区和缓冲区统计行。包括交换区总量，已使用的交换区，空闲交换区，高速缓冲区总量。

输出的其他行的含义和 ps 的输出类似，这里不再介绍。

在 top 屏幕下，用“q”键可以退出，用“h”键可以显示 top 下的帮助。

6. bg、jobs、fg

在手工启动前台进程时，如果进程没有执行完毕，则可以使用【Ctrl-z】键暂停进程的执行。例如：

```
[root @redflag /root]#du -a / | sort -rn > /root/test.out 按【Ctrl-z】
```

```
[1]+ Stopped    du -a / | sort -rn > /root/test.out
```

表示 du 命令被暂停，这时候我们可以把暂停的进程放到后台继续运行，我们在前台还可以运行别的命令。bg 命令用于把进程放到后台。

```
[root @redflag /root]#bg du 或 bg %1
```

```
[1]+ du -a / | sort -rn > /root/test.out &
```

使用 jobs 命令可以看到在后台运行的进程。

```
[root @redflag /root]#jobs
```

```
[1]+ Running du -a / | sort -rn > /root/test.out
```

这说明了 du 程序在后台运行。在上面例子中，我们把被暂停的进程放到后台运行，如果我们想直接在后台运行它，在命令行后加“&”字符，不必用【Ctrl-z】键先把它暂停再用 bg 命令把它放到后台。例如：

```
[root @redflag /root]#du -a / | sort -rn > /root/test.out &
```

用 bg 命令可以把进程放到后台，用 fg 命令则可以把在后台运行的进程放到前台。

```
[root @redflag /root]#du -a / | sort -rn > /root/test.out &
```

```
[root @redflag /root]#jobs
```

查出在后台运行的进程的进程号，我们假设是 1，则

```
[root @redflag /root]#fg %1
```

把进程从后台放到了前台。

在后台运行的进程，如果进程有到标准输出设备的输出，则最好把输出重定向到文件。

7. nohup 命令

一般来说，当一个进程的父进程被终止时，该进程也会被终止。用“ps -l”命令可以看到进程的父进程。例如：

F S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
100 S	0	1552	1551	0	69	0	-	593	wait4	pts/0	00:00:00	login
000 S	0	1586	1553	0	69	0	-	698	wait4	pts/0	00:00:00	su
100 S	0	1587	1586	0	74	0	-	698	wait4	pts/0	00:00:00	bash
100 R	0	2642	1587	0	76	0	-	770	-	pts/0	00:00:00	ps

其中，进程 2642 的父进程是 1587，1587 实际上是 Shell 进程。在 Shell 下启动的进程的父进程大多是 Shell 进程，而 Shell 进程的父进程又是 login 进程，所以当你 logout 时，由 login 进程衍生出的进程就会被终止。如果你希望 logout 后自己的进程仍然能在后台继续运行，可以使用 nohup 命令。nohup 的用法如下：

```
[root @redflag /root]#nohup 命令 &
```

【实例 2.4】

```
[root @redflag /root]#nohup du -a / | sort -rn > /root/test.out &
```

命令 du-a/ | sort -rn > /root/test.out 不会在用户 logout 后终止。

2.3.4 作业控制

有时我们需要把费时的工作放在深夜进行，这时候我们可以事先进行调度安排，即调度启动进程，系统会自动启动我们安排好的进程。

1. at、atq、atrm

我们可以使用 at 命令将要执行的命令安排成队列，例如：

```
[root @redflag /root]#at 8:40
```

```
at>du -a >test.out
```

```
at>tree / >>>test.out
```

按【Ctrl-d】退出。很明显，我们安排系统在 8:40 执行两个命令，注意这些命令只执行一次。使用 at 命令时，可以使用不同的时间格式，例如：20:40、8:00am、8:40am feb 23、10am+5days、12:30 pm tomorrow、midnight、noon 等。如果执行如下命令：

```
[root @redflag /root]#at -f mywork 9:45am+2days
```

则安排系统于两天后的上午 9:45 执行文件 mywork 里的作业。用 at 命令设定好作业后，atd 守护进程将负责运行它们。我们可以使用 atq 命令查看已经安排好的作业。

```
[root @redflag /root]#atq
```

```
14 2003-02-28 07:45 a root
```

```
13 2003-02-27 08:40 a root
```

输出行中依次是作业号、作业的启动时间、用户名，遗憾的是这里不能知道作业的内容。如果想知道作业的内容，请到/var/spool/at 目录里去找(类似 a00011010a2548 这样的文件)。

【说明】 超级用户可以列出全部用户安排的作业，而一般用户只能列出自己安排的作业。

如果要删除作业，可以使用 `atrm` 命令。例如：

```
[root @redflag /root]#atrm 13
```

这里的 13 是作业号。

【注意】 超级用户可以在任何情况下使用 `at` 系列的命令。一般用户使用 `at` 系列命令的权利由文件 `/etc/at.allow` 和 `/etc/at.deny` 控制。如果 `/etc/at.allow` 存在，则只有列在这个文件中的用户才能使用 `at` 系列的命令。如果 `/etc/at.allow` 文件不存在，则检查 `/etc/at.deny` 这个文件。只要不列在这个文件中的用户都可以使用 `at` 系列的命令。缺省的配置是 `/etc/at.deny`，它是一个空文件，这表明所有的用户都可以使用 `at` 系列的命令。

2. crontab 命令

`at` 命令用于安排运行一次的作业比较方便，但如果要重复运行程序，则使用 `crontab` 更为简捷。`crontab` 的用法如下：

```
crontab [参数] {-e |-l |-r }
```

参数选项：

`-u username`：用户 `username` 的作业，不指定时指当前用户。

`-e`：编辑用户 `cron` 作业。

`-l`：显示用户 `cron` 作业。

`-r`：删除用户 `cron` 作业。

例如：

```
[root @redflag /root]#crontab -e
```

进入编辑 `cron` 作业状态，编辑器采用的是 `vi` 编辑器。关于 `vi` 的使用请参见第 3 章。一个 `cron` 作业的格式如下：

分 小时 日 月 星期 命令

如果用户不需要指定其中的几项，可以用“*”代替，例如：

```
0,10,20 * * * * updatedb
```

每逢 0、10、20 分运行一次命令 `updatedb`。

```
* * * * * date>>/root/crontest.out
```

表示每分钟往 `/root/crontest.out` 文件添加一行关于时间的行。

```
[root @redflag /root]#crontab -l
```

列出当前用户的 `cron` 作业：

```
# DO NOT EDIT THIS FILE - edit the master and reinstall.
```

```
# (/tmp/crontab.1106 installed on Thu Feb 27 23:17:23 2003)
```

```
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
```

```
* * * * * date>>/root/crontest.out
```

2.4 文件压缩和备份

2.4.1 压缩和解压命令

使用压缩文件不仅可以减小文件占用的磁盘空间，也可以减小文件在网络传输时所带来的传输流量。Linux 的压缩和解压工具很多，下面我们介绍常用的几个工具。

1. `compress [参数] 文件名(压缩文件命令) / uncompress [参数] 文件名(文件解压命令)`

参数选项：

`-v`：显示被压缩的文件的压缩比或解压时的信息。

例如：

```
[root @redflag /root]#compress -v test
test:  -- replaced with test.Z Compress: 53.56%
```

文件 `test` 被压缩成 “`test.Z`”，压缩比为：53.56%。

```
[root @redflag /root]#uncompress -v test 或 uncompress -v test.Z
test.Z:  -- replaced with test
```

解压文件是 “`test.Z`”。

【注意】 `compress` 命令压缩文件时，原文件会被替换为以 “.Z” 作为文件名结尾的文件。解压时，文件名的结尾 “.Z” 可以加，也可以不加，不加时 `uncompress` 会自动查找 “.Z” 文件。

2. `gzip -v 文件名(压缩文件) / gunzip -v 文件名(解压文件)`

`gzip`、`gunzip` 和 `compress`、`uncompress` 类似，不过压缩后的文件的文件名是以 “.gz” 结尾而已。

3. `zip 压缩文件名(压缩文件) / unzip 被解压文件名(解压文件)`

`zip` 生成的文件是以 “.zip” 为文件名的结尾，这种文件是我们在 Windows 等系统中常见的压缩文件。`zip` 命令的功能非常强大，可以创建自解压的文件、设置文件的保护口令等。常用 `man zip` 命令来获得 `zip` 命令的详细帮助。`zip` 命令并不替换原文件。

例如：

```
[root @redflag /root]#zip test.zip test
adding: test (deflated 66%)
```

把文件 `test` 压缩到文件 “`test.zip`”。

```
[root @redflag /root]#unzip test.zip
Archive: test.zip
inflating: test
```

如果 `test` 已经存在，`unzip` 命令会提示是否覆盖 `test` 文件。

2.4.2 文件备份

任何计算机系统都可能出现问題，从而导致数据的丢失，因此备份是系统维护中不可缺少的一个环节。备份应做到系统崩溃后能快速、简单、完整地恢复系统。针对要备份的内容，备份可以分为两类：系统数据备份和用户数据备份。系统数据是指 Linux 系统要正常运行所需的文件(如/bin 和/boot 目录)、系统配置(如/etc 目录)等；而用户数据是指计算机用户创建的文件(如/home 目录)等，相对于系统数据来说，用户数据的变化要频繁得多。备份系统数据时，可以不备份不必要的数据(如/proc/core)，它是当前物理内存的一个映像，并且系统备份常在系统有变化后进行，例如安装了新的补丁。

备份有两种策略：完全备份和增量备份。完全备份是把要备份的数据完完全全备份出来，一旦系统发生故障，可以使用备份的数据把数据恢复到备份前的状态；增量备份则是备份最后一次备份以后发生了变化的数据，因此被备份的数据量要少得多。实际工作中，完全备份和增量备份常常是结合起来使用的。例如：一周(例如在星期一)进行一次完全备份，而每天进行一次增量备份，如果系统于星期四出现故障，恢复系统时先用完全备份恢复系统，然后再顺序用周二、周三的增量备份恢复系统。Linux 中的 tar 工具是最常用的备份和恢复工具，同时 tar 也是软件商发布补丁、新软件的常用工具，所以掌握 tar 的使用是非常重要的。我们把备份产生的文件称为文档文件(或文档)。格式如下：

tar [参数] 文件或目录名

参数选项：

-c：创建一个新的文档。

-r：用于将文件附加到已存在的文档后面。

-u：仅仅添加比文档文件更新的文件，如原文档中不存在旧的文件，则追加它到文档中，如存在则更新它。

-x：从文档文件中恢复被备份的文件。

-t：用于列出一个文档文件中的被备份出的文件名。

-z：用 zip 命令压缩或用 unzip 解压。

-f：使用档案文件或设备，这个选项通常是必选的。

-v：列出处理过程中的详细信息。

-C directory：把当前目录切换到 directory。

例如：

```
[root @redflag /root]#tar -cvf longkey.tar /home/longkey
```

把/home/longkey 目录下的文件和子目录(包括隐含文件和目录)备份到 longkey.tar 文档中。备份产生的文档文件的文件名最好用 “.tar” 结尾，以示区别。以上是一个完全备份。

```
[root @redflag /root]#tar -uvf longkey.tar /home/longkey
```

把/home/longkey 目录中比文档文件 longkey.tar 还新的文件添加到 longkey.tar 中。

```
[root @redflag /root]#tar -czvf longkey.tar.gz /home/longkey
```

以 gzip 压缩文件的形式把/home/longkey 的内容备份到 longkey.tar.gz 中。备份产生的文档是压缩过的，这样可以减小文档文件的大小。注意，文件是以 “.tar.gz” 结尾的。

```
[root @redflag /root]#tar -xzf longkey.tar.gz
```


从 longkey.tar.gz 文档中恢复数据。注意，恢复出来的数据是放在当前目录下的，而不是恢复到原来的目录。

```
[root @redflag /root]#tar -xzf longkey.tar.gz -C /home
```

可以把文件恢复到指定的目录/home 下。

tar 命令的参数还有很多，而且它常常和其他的命令如 find 等一起使用，以实现完全备份和增量备份，用户甚至还常写脚本来实现自己备份的策略。

2.5 网络命令

Linux 系统也是一个网络操作系统，其网络功能也相当强大。目前 Linux 系统大多是被用来提供网络服务的，Linux 系统可以提供各种各样的网络服务，例如：Web 服务、FTP 服务、DNS 服务。这些内容将放在以后的章节介绍，我们先介绍基本的网络命令。

2.5.1 hostname、ping、host

1. hostname [主机名]——显示或设置系统的主机名

例如：

```
[root @redflag /root]#hostname
redflag
```

表示本人的系统主机名是“redflag”。

```
[root @redflag /root]#hostname server.redflag.com
```

把主机名设置为“server.redflag.com”。

2. ping [参数] 主机名(或 IP 地址)——测试本主机和目标主机连通性

参数选项：

-c count：共发出 count 次信息，不加此项，则发无限次信息。

-i interval：两次信息之间的时间间隔为 interval，不加此项，间隔为 1 秒。

例如：

```
[root @redflag /root]#ping -c 10 -i 0.5 192.168.0.1
PING 192.168.0.1 (192.168.0.1) from 192.168.0.10 : 56(84) bytes of data.
Warning: time of day goes back, taking countermeasures.
64 bytes from 192.168.0.1: icmp_seq=0 ttl=128 time=648 usec
64 bytes from 192.168.0.1: icmp_seq=1 ttl=128 time=578 usec
.....
64 bytes from 192.168.0.1: icmp_seq=8 ttl=128 time=569 usec
64 bytes from 192.168.0.1: icmp_seq=9 ttl=128 time=576 usec

--- 192.168.0.1 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.544/0.575/0.648/0.028 ms
```

以上命令共发出 10 次信息，信息之间的间隔为 0.5 秒。

3. host 主机名或 IP 地址——IP 地址查找工具

例如：

```
[root @redflag /root]#host www.sina.com
www.sina.com. has address 66.77.9.79
DNS 查找出 “ www.sina.com ” 的 IP 地址为 “ 66.77.9.79 ”。
[root @redflag /root]#host 202.96.134.133
133.134.96.202.in-addr.arpa. domain name pointer ns.szptt.net.cn.
```

这是反向 IP 地址解析。

```
[root @redflag /root]# host mylinux 192.168.0.10
Using domain server:
Name: 192.168.0.10
Address: 192.168.0.10#53
Aliases:
mylinux.test.com. has address 192.168.0.1
```

以上指明了从 DNS 服务器 192.168.0.10 上查找主机 mylinux 的 IP 地址，如果没有指明 DNS 服务，则系统使用本机网络设置中设定的 DNS 服务器。有关 IP 地址的查找请进一步参见本书有关 DNS 的章节。

2.5.2 ifconfig

ifconfig 是用于配置网卡和显示网卡信息的工具。例如：

```
[root @redflag /root]#ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:E0:4C:30:03:F7
          inet addr:192.168.0.10  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1257 errors:0 dropped:0 overruns:0 frame:0
          TX packets:558 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:138072 (134.8 Kb)  TX bytes:48879 (47.7 Kb)
          Interrupt:5 Base address:0xc000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1046 (1.0 Kb)  TX bytes:1046 (1.0 Kb)
```

显示本主机有一个网卡 eth0。lo 是本地环路(虚拟的)网卡，不是物理上实实在在的网卡。在以上的输出中有几个重要的信息：IP 地址、网卡 MAC 地址、网卡的配置以及网卡的一些统计数(如接收和发送包的总量)。

【注意】如果 ifconfig 命令不带参数“-a”，则只显示当前激活的网卡的信息，不激活的网卡的信息不显示。

Ifconfig 命令的格式如下：

ifconfig 网卡号 [参数] [IP 地址]

参数选项：

up：激活网卡。

down：关闭网卡。

例如：

```
[root @redflag /root]#ifconfig eth0 down
```

关闭网卡。

```
[root @redflag /root]#ifconfig eth0 192.168.0.100 netmask 255.255.255.0
```

把网卡 eth0 的 IP 地址改为 192.168.0.100，子网掩码为 255.255.255.0。

2.5.3 traceroute 目标主机名或 IP 地址

该命令显示本机到达目标主机的路由路径，例如：

```
[root @redflag /root]#traceroute www.sina.com
traceroute to www.sina.com (66.77.9.79), 30 hops max, 38 byte packets
 1  * * *
 2  211.162.65.62 (211.162.65.62)  1.116 ms  1.010 ms  0.945 ms
 3  211.162.78.201 (211.162.78.201)  1.061 ms  1.053 ms  1.030 ms
 .....
18  svl-core-01.inet.qwest.net (205.171.8.242) 1250.125 ms 1265.016 ms 1255.987 ms
19  svc-cntr-01.inet.qwest.net (205.171.14.2) 1255.304 ms 1254.937 ms 1249.087 ms
20  msfc-07.jsv.qwest.net (66.77.106.98) 1247.149 ms 1271.762 ms 1267.991 ms
21  66.77.9.79 (66.77.9.79) 1250.633 ms 1253.705 ms 1260.267 ms
```

以上输出中每一行代表一个段，使用该命令可以找到数据从本机到达目标主机所经过的每一个网关。如果数据包无法到达，也很容易确定问题点。

2.5.4 Telnet、FTP

1. Telnet 主机名或 IP 地址——远程登录客户程序

例如：

```
[root @redflag /root]#telnet 192.168.0.200
```

远程登录到服务器 192.168.0.200。服务器 192.168.0.200 应开启 Telnet 服务，否则会连接失败。如果成功连接 Telnet，程序会提示输入用户名和口令，登录成功后就可以远程管理或使用服务器。

2. FTP 主机名或 IP 地址——FTP 客户程序

例如：

```
[root @redflag /root]#ftp 192.168.10
```

FTP 到远程 FTP 服务器 192.168.0.10，同样服务器 192.168.0.10 要开启 FTP 服务。连接成功后，FTP 程序会提示输入用户名和口令。如果连接成功，将得到“ftp>”提示符。现在可以自由使用 FTP 提供的命令，可以用 help 命令或“?”取得可供使用的命令清单，也可以在 help 命令后面指定具体的命令名称，获得这条命令的说明。最常用的命令有：

ls：列出远程机的当前目录。

cd：在远程机上改变工作目录。

lcd：在本地机上改变工作目录。

ascii：设置文件传输方式为 ASCII 模式。

binary：设置文件传输方式为二进制模式。

close：终止当前的 FTP 会话。

hash：每次传输完数据缓冲区中的数据后就显示一个“#”号。

get(mget)：从远程机传送指定文件到本地机。

put(mput)：从本地机传送指定文件到远程机。

open：连接远程 FTP 站点。

quit：断开与远程机的连接并退出 FTP。

?：显示本地帮助信息。

!：转到 Shell 中。

下面简单将 FTP 常用命令作以介绍。

(1) 启动 FTP 会话。open 命令用于打开一个与远程主机的会话。该命令的一般格式是：

open 主机名/IP

(2) 终止 FTP 会话。close、disconnect、quit 和 bye 命令用于终止与远程机的会话。close 和 disconnect 命令用于关闭与远程机的连接，但是仍没有退出 FTP 程序。quit 和 bye 命令都用于关闭用户与远程机的连接，然后退出用户机上的 FTP 程序。

(3) 改变目录。“cd [目录]”命令用于在 FTP 会话期间改变远程机上的目录，lcd 命令可改变本地目录，使用户能指定查找或放置本地文件的位置。

(4) 远程目录列表。ls 命令列出远程目录的内容，就像使用一个交互 Shell 中的 ls 命令一样。ls 命令的一般格式是：

ls [目录]

如果指定了目录作为参数，那么 ls 就列出该目录的内容。

(5) 从远程系统下载文件。get 和 mget 命令用于从远程机上获取文件，get 命令的一般格式为：

get 源文件名 目标文件名

源文件名是要下载的文件名，目标文件名是文件下载后在本地机上保存时的文件名。如果不给出目标文件名，那么就使用源文件的名字。mget 命令一次可获取多个远程文件。mget 命令的一般格式为：

mget 文件名列表

使用用空格分隔的或带通配符的文件名列表来指定要获取的文件。

(6) 向远程系统上载文件。put 和 mput 命令用于向远程机发送文件，put 命令的一般格式为：

put 文件名

mput 命令一次发送多个本地文件，mput 命令的一般格式为：

mput 文件名列表

使用用空格分隔的或带通配符的文件名列表来指定要发送的文件。

(7) 改变文件传输模式。默认情况下，FTP 按 ASCII 模式传输文件，用户也可以指定其他模式。ascii 和 binary 命令的功能是设置传输的模式。用 ASCII 模式传输纯文本文件是非常好的，而二进制文件以二进制模式传输更为可靠。

(8) 切换“#”提示。hash 命令使 FTP 在每次传输完数据缓冲区中的数据后，就在屏幕上打印一个“#”字符。本命令在发送和接收文件时都可以使用。hash 命令是一个开关。

【实例 2.5】

以下是下载一个文件的过程：

```
[root @redflag /root]#ftp
ftp> open 192.168.0.10
Connected to 192.168.0.10.
220 mylinux.wlj.com FTP server (Version wu-2.6.1-16) ready.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (192.168.0.10:test): test
331 Password required for test.
Password:
230-----
230-This is wanglongjie's FTP Server
230-----
230-
230 User test logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> pwd
257 "/home/test" is current directory.
ftp> ls
227 Entering Passive Mode (192,168,0,10,249,113)
150 Opening ASCII mode data connection for directory listing.
total 9963
lrwxrwxrwx   1 root    root          12 Feb 18 20:29 Desktop -> .Desktop_gb/
-rw-rw-r--   1 503    503           0 Mar  1 10:46 a.zip
-rw-r--r--   1 503    503    5044297 Feb 28 23:59 aaa
drwxr-xr-x   2 root    root           80 Feb 19 23:08 public_html
drwxrwxr-x   2 503    503           288 Feb 26 13:59 ttt
```

```
226 Transfer complete.
ftp> get a.zip /root/1.zip
local: /root/1.zip remote: a.zip
227 Entering Passive Mode (192,168,0,10,20,213)
150 Opening BINARY mode data connection for a.zip (0 bytes).
226 Transfer complete.
ftp> close
221-You have transferred 0 bytes in 1 files.
221-Total traffic for this session was 783 bytes in 1 transfers.
221 Thank you for using the FTP service on mylinux.wlj.com.
ftp> bye
[root @redflag /root]#
```

2.5.5 wall、write、mesg

1. wall——向任何用户终端发送字符消息

例如：

```
[root @redflag /root]#wall
```

表示进入消息输入状态，可以输入一行或多行消息，按【Ctrl-d】键结束。在进行系统管理时，如果有紧急消息要通知所有在线用户，wall 命令十分有用。

【注意】不要轻易使用 wall，以免对他人造成不必要的干扰。wall 命令和下面介绍的 mesg 命令有关。

2. write 用户名 [终端]——向用户发送字符消息

例如：

```
[root @redflag /root]#write user1
```

表示进入消息输入状态，可以输入一行或多行消息，按【Ctrl-d】键结束。write 命令和下面介绍的 mesg 命令也有关。

3. mesg [参数]——控制他人向自己的终端发送消息的能力

参数选项：

y：允许他人往自己的终端发送消息。

n：不允许他人往自己的终端发送消息，但无法阻止 root 用户向自己发送信息。

例如：

```
[root @redflag /root]#mesg n
```

表示其他用户用 wall 命令发送消息时，不会对自己的终端产生影响。

```
[root @redflag /root]#mesg
```

```
is n
```

显示当前终端是否允许他人往自己的终端发送消息，以上表示不允许。

2.5.6 mail

mail 用户名或 E-mail 地址——SMTP 客户端程序。

可以使用这个程序在系统内发送和接收邮件，也可以往 Internet 上的主机发送邮件或从 Internet 的主机接收邮件。例如：

```
[root @redflag /root]#mail longkey
Subject:This is a test mail
Hello,longkey!
Cc:
```

输入时按【Ctrl-d】键可以结束输入，把邮件发出。当 longkey 用户登录时，系统会提示“ You have mail ”。这时 longkey 用户可以直接使用 mail 命令来接收邮件和回复邮件。键入该命令时，出现“ & ”提示符，用“ ? ”命令可以得到 mail 的帮助。mail 的使用较为复杂，这里就不详细讨论了。

2.5.7 finger

finger [用户名@主机]——显示主机系统中用户的信息。

例如：

```
[root @redflag /root]#finger

Login      Name      Tty      Idle  Login    Time      Office    Office Phone
root       root      tty1      2     Mar  1    09:44
test                      tty2      4     Mar  1    11:08
test                      pts/0      Mar  1    11:13 (wwwb.szpt.net)
```

显示用户当前登录的主机上的所有登录用户的信息。finger 命令要求主机要提供 finger 服务，否则会连接失败。

```
[root @redflag /root]#finger longkey@192.168.0.10
Login: longkey                               Name: (null)
Directory: /home/longkey                     Shell: /bin/bash
Last login Sat Mar  1 08:32 (EST) on tty3
No mail.
No Plan.
```

显示主机 192.168.0.10 上的用户 longkey 的信息。由于 finger 命令常常为黑客所利用，所以大多数的系统关闭了 finger 服务。

2.5.8 netstat [参数选项]

netstat 命令显示网络连接、路由表、网卡统计数等信息。

参数选项：

-i：显示网卡的统计数。

-r：显示路由表。

-a：显示所有信息。

例如：

```
[root @redflag /root]#netstat -i
```

表示显卡的统计数。

以下是网卡的各种统计数据，如：接收错误的包的数量。

Kernel Interface table

Iface	MTU	Met	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
eth0	1500	0	460	0	0	0	320	0	0	0	BMRU
lo	16436	0	54	0	0	0	54	0	0	0	LRU

2.6 其他命令

2.6.1 clear、dmesg、uname

1. clear——清除屏幕

该命令相当于 DOS 下 cls 命令。

2. dmesg——显示内核引导时的状态信息

该命令对于内核引导出现故障时查找问题十分有用。

3. uname -a——显示系统的信息

例如：

```
[root @redflag /root]#uname -a
```

```
Linux mylinux.test.com 2.4.7-2 #1 一 8月 27 14:04:34 CST 2001 i686 unknown
```

说明主机名是 mylinux.test.com，Linux 的内核是 2.4.7-2，CPU 是 i686 结构。

2.6.2 date、cal

1. date [时间]——显示或设置系统的时间

格式如下：

[时间]：MMDDhhmmCCYY.SS

```
[root @redflag /root]#date
```

```
三 2月 26 17:04:44 EST 2003
```

当前系统的时间为 2003 年 2 月 26 日 17:04:44，星期三。

【注意】 date 命令应在中文环境下(用“yh”命令)执行，否则中文部分的显示是乱码。

```
[root @redflag /root]#date 022714022003.50
```

```
四 2月 27 14:02:50 EST 2003
```

把系统的时间设置为 2003 年 2 月 27 日 14:02:50。

2. cal [月][年]——显示指定年月的月历

如未指明年月，则显示当月的月历。

```
[root @redflag /root]#cal 3 2003
```



```
      三月 2003
日 一 二 三 四 五 六
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

【注意】 cal 命令应在中文环境下(用“yh”命令)执行，否则中文部分的显示是乱码。

2.6.3 help、man

1. help [内置命令]——用于查看 Linux 内置命令的帮助

例如：

```
[root @redflag /root]#help
```

输出全部内置命令。我们曾介绍 Linux 命令有内部命令和外部命令之分，如果读者不能区别命令是哪类，通过这一命令就可以区别它们了。

```
[root @redflag /root]#help alias
```

输出内置命令 alias 的帮助。

2. man [命令名]——命令的帮助手册

Linux 的命令不仅多，而且每个命令的功能都十分强大，其参数也多如牛毛，幸运的是它有一个帮助系统能够帮助我们。例如：

```
[root @redflag /root]#man ls
```

获得 ls 的帮助。

典型的 man 手册包括以下几个部分：

NAME：命令的名字。

SYNOPSIS：名字的概要，简单说明命令的使用方法。

DESCRIPTION：详细描述命令的使用，如各种参数选项的作用。

SEE ALSO：列出可能要查看的其他有关的手册页条目。

AUTHOR，COPYRIGHT：作者和版权等信息。

学会 man 的使用非常必要，熟练的 Linux 管理员也常常离不开它。遗憾的是，手册中命令使用的例子不多，且都是用英文说明的，即使在红旗 Linux 中也没有汉化。

2.6.4 init、shutdown、halt、reboot、poweroff

关闭 Linux 系统要采取正确的步骤，否则会引起文件系统损坏。由于 Linux 系统使用磁盘缓冲技术，Linux 并不把数据立即写到磁盘上，因此不能直接用关闭电源来关机。正确的步骤应是执行如下指令：

```
[root @redflag /root]#sync;sync;sync
```

```
[root @redflag /root]#shutdown -h now (或下面我们介绍的关闭系统指令)
```

这三个 sync 可确保磁盘缓冲的内容全部写到磁盘中。此外，缺省时按【Ctrl】+【Alt】+【Del】键可以重新启动系统，用户可以禁止这一功能，方法是先找到/etc/inittab 文件，把以下行屏蔽即可：

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

1. init n——改变系统的运行等级

n：指定的系统运行等级。

0——停止系统。

1——单用户。

2——多用户，但不支持 NFS。

3——全多用户模式，即系统正常的模式。

5——进入 X11(即窗口模式)。

6——重启系统。

例如：

```
[root @redflag /root]#init 6
```

重新启动系统。

```
[root @redflag /root]#init 0
```

关闭系统。

【提示】 /etc/inittab 文件中的第一个非注释行“init:3:initdefault”是用来指明 Linux 系统启动后的运行等级的。如果把 3 改成 1，则系统启动后会进入单用户状态。

2. shutdown [参数] 时间 [警告消息]——在指定时间关闭系统

参数选项：

-r：系统关闭后重启。

-h：关闭后停机。

时间可以有以下几种形式：

now：表示立即。

hh:mm：指定绝对时间，hh 表示小时，mm 表示分钟。

+m：表示 m 分钟以后。

例如：

```
[root @redflag /root]#shutdown -r +5 "System will reboot in 5 minutes."
```

该命令警告用户 5 分钟后系统重启。

3. halt——立即停止系统

该命令不自动关闭电源，需要人工关闭电源。

4. reboot——立即重启系统

相当于命令：

```
shutdown -r now
```

5. poweroff——立即停止系统，并且关闭电源

该命令要求计算机支持关机功能。相当于命令：

```
shutdown -h now
```

2.6.5 alias、unalias、history

1. alias 命令别名 = “命令行” ——创建命令的别名

例如：

```
[root @redflag /root]#alias
```

显示已有的命令别名，读者在其中可以发现 `ls= 'ls --color'`，这就是用户执行 `ls` 命令后为什么会用颜色表示不同的文件类别的原因了。用户可以把自己常用的长命令通过定义别名来用短命令替代。

```
[root @redflag /root]#alias mydir= 'ls --color'
```

创建自己的命令 `mydir` 代替 “`ls -color`”。

2. unalias 命令别名——删除已创建的别名

例如：

```
[root @redflag /root]#unalias mydir
```

删除之前已经定义的别名 `mydir`。

3. history——显示用户最近执行的命令

可以保留的历史命令数和环境变量 `HISTSIZE` 有关。只要在编号加 “!”，很容易地就可以重新运行 `history` 中的显示出的命令行。例如：

```
[root @redflag /root]#!25
```

表示重新运行第 25 个历史命令。

2.6.6 su

`su [用户名]`——改变用户的 ID 或成为超级用户。

`su` 可以让用户在一个登录的 Shell 中不退出就改变成为另一用户。如果 `su` 命令不跟用户名，则 `su` 命令缺省地成为超级用户。执行 `su` 命令后系统会要求输入密码。`su` 之后，当前所有的用户变量都会传递过去。`su` 命令在远程管理时相当有用，一般情况下超级用户（即 `root` 用户）不被允许远程登录。这时候，可以用普通用户 `Telnet` 到主机，再用 `su` 成为超级用户后进行远程管理，例如：

```
[test@mylinux test]$ su
```

```
[root@redflag test]#
```

2.6.7 who、whoami、w、last

1. who——显示谁登录系统

例如：

```
[root @redflag /root]#who
```

root	tty1	Mar	1	12:17
test	tty2	Mar	1	12:17
longkey	tty3	Mar	1	12:17
test	pts/0	Mar	1	11:35

输出表示有 4 个用户登录系统，输出行中分别是用户名、登录的终端号和登录时间。

【说明】 pts/0 终端表示是用 Telnet 登录(第一个)的。

2. whoami——显示当前登录的用户名

例如：

```
[root @redflag /root]#whoami
root
```

表示当前用户是 root。

3. w——显示谁登录系统并且在做什么

例如：

```
[root @redflag /root]#w
12:18pm  up  44 min , 4 users , load average :  0.06 ,  0.03 ,  0.00
USER      TTY      FROM          LOGIN@      IDLE   JCPU   PCPU   WHAT
root      tty1      -              12:17pm     1:07   0.04s  0.03s  -bash
test      tty2      -              12:17pm     1:04   0.10s  0.03s  -bash
test      pts/0     wwwb.szpt.net  11:35am     0.00s  0.13s  0.01s  w
```

4. last——显示过去有多少用户在本机登录

last 命令显示的是直到/var/log/wtmp 文件创建以来有多少用户在本机登录。

2.6.8 rpm——安装软件包

以前的 Linux 软件几乎都是源程序代码的形式，要安装时必须先取得压缩文件，解压得到源程序，再编译成可执行的文件，然后将相关的文件放到正确的目录中。RPM 系统(Red Hat Package Manager)就是为了解决软件的安装问题而开发的，有了 RPM 就可以用一条命令完成软件的安装，RPM 自动地帮我们完成了复杂的安装步骤。软件开发人员将软件源程序代码、补丁(Patch)及安装指示包装成一个 RPM 套件(也就是一个 rpm 文件)。安装软件包只要一个 rpm 文件，执行 rpm 命令就可以轻松安装了。系统的 RPM 数据库记载了所有的以 RPM 方法安装的数据，因此可以非常方便地删除、查询和升级软件。

rpm 命令非常复杂，其格式如下：

```
rpm [参数选项]
```

我们这里以实例来说明常用参数选项的使用。

【实例 2.6】

```
[root @redflag /root]#rpm -qa
glibc-common-2.2.3-12
htmlview-1.1.0-2
mailcap-2.1.4-2
rhl-gsg-6.1en-3
sash-3.4-8
basesystem-7.0-3
```

```
chkconfig-1.2.22-1
db2-2.4.14-4
e2fsprogs-1.27-8
finger-0.17-9
gdbm-1.8.0-3
hdparm-5.2-1
isapnptools-1.22-2
libattr-2.0.8-2
```

.....

RPM 数据库存储了安装在系统内所有软件包的版本、文件和位置等相关数据，可以随时查询。“-qa”参数是查询目前系统中安装的全部软件包。

【实例 2.7】

```
[root @redflag /root]#rpm -q nfs-utils
```

```
nfs-utils-0.3.1-8
```

查询某一特定软件包，用“-q”参数，如果要显示软件包的完整信息，加“-i”参数。

```
[root @redflag /root]#rpm -qi nfs-utils
```

```
Name       : nfs-utils           Relocations: (not relocateable)
Version     : 0.3.1              Vendor: RedFlag SoftWare
Release     : 8                  Build Date: 2001 年 08 月 03 日 星期五 14 时 05 分 15 秒
Install date: 2003 年 03 月 19 日 星期三 08 时 56 分 43 秒 Build Host: xiejue.redflag-linux.com
Group       : System Environment/Daemons   Source RPM: nfs-utils-0.3.1-8.src.rpm
Size        : 520527              License: GPL
Packager    : RedFlag SoftWare<http://www.redflag-linux.com>
Summary     : NFS utilities and supporting daemons for the kernel NFS server.Description :
```

The nfs-utils package provides a daemon for the kernel NFS server and related tools, which provides a much higher level of performance than the traditional Linux NFS server used by most users.

This package also contains the showmount program. Showmount queries the mount daemon on a remote host for information about the NFS (Network File System) server on the remote host. For example, showmount can display the clients which are mounted on that host.

【实例 2.8】

```
#rpm -ql nfs-utils
/etc/rc.d/init.d/nfs
/etc/rc.d/init.d/nfslock
/sbin/rpc.lockd
/sbin/rpc.statd
/sbin/rpcdebug
/usr/sbin/exportfs
```

```

/usr/sbin/nfsstat
/usr/sbin/nhfsstone
/usr/sbin/rpc.mountd
.....

```

查询 nfs-utils 软件包包含的文件。

【实例 2.9】

用 rpm 命令安装软件包时，软件包要求是以 “.rpm” 结尾的文件。把红旗 3.0 的光盘安装在 /mnt/cdrom 目录后，这些软件包就在 /mnt/cdrom/RedFlag/RPMS 目录下。

```

[root @redflag RPMS]#rpm -ivh zsh-4.0.1-1.i386.rpm
Preparing...                               ##### [100%]
1:zsh                                       ##### [100%]

```

以上命令安装了 zsh 软件包。可以看到安装是多么轻松的一件事。如果要安装的软件已经安装了，系统会出现提示信息：

```

Preparing...                               ##### [100%]
package zsh-4.0.1-1 is already installed

```

【实例 2.10】

删除软件包使用 “-e” 参数。例如：

```
[root @redflag /root]#rpm -e zsh
```

表示如果成功删除就没有输出。

```

[root @redflag /root]#rpm -e apache
error: removing these packages would break dependencies:
    apache is needed by apacheconf-0.7-2
    webserver is needed by mod_perl-1.24_01-2
    webserver is needed by mod_ssl-2.8.10-1
    webserver is needed by auth_ldap-1.6.0-5

```

表示如果一个软件包和其他软件包依存时，就无法删除它，这样就避免了许多错误。

【实例 2.11】

要升级软件包时用 “-U” 参数。例如：

```
#rpm -Uvh zsh-4.0.1-1.i386.rpm
```

RPM 会自动删除旧版，安装新版，而旧版的配置保留。如果旧版不存在，就会自动安装新版软件包。“-F” 参数则会要求 RPM 把系统中的旧版和要升级的版本进行比较，只有升级版比旧版新时才进行安装。如果系统中不存在旧的版本，RPM 也不会安装软件包。

【实例 2.12】

要确认软件包的完整性，可以使用 “-V” 参数。

```

[root @redflag /root]#rpm -V nfs-utils
.....T c /var/lib/nfs/etab
S.5....T c /var/lib/nfs/rmtab
.....T c /var/lib/nfs/xtab

```

RPM 比较系统现有软件包和原有软件包的所有的文件数据，如果完全一样，就不会有任何输出。有输出说明文件被改动，文件名前的“c”表示是设置文件。其余的表示如下：

S 为文件大小，L 为符号连接，5 为 MD5 checksum，T 为文件修改时间，D 为设备，U 为拥有者，G 为所属组，M 为模式(包括文件权限等) 被改变。

本章小结

本章主要介绍 Linux 的常用命令，其中主要有文件和目录的操作命令：pwd、cd、ls、tree、mkdir、rmdir、cp、rm、mv、ln、chmod、chown、chgrp、find、grep、cmp、diff、stat、touch；显示文件的命令：cat、more、less、head、tail、sort、uniq、file、locate、which；进程和作业的控制命令：&、ps、kill、nice、renice、top、bg、fg、job、nohup、at、atq、atrm、crontab；文件压缩和备份命令：compress、uncompress、gzip、gunzip、zip、unzip、tar；网络命令：hostname、ping、host、ifconfig、traceroute、telnet、ftp、wall、write、mesg、mail、finger、netstat；其他命令：clear、dmesg、uname、date、cal、man、help、init、shutdown、halt、reboot、poweroff、alias、unalias、history、su、who、whoami、w、last、rpm。各个命令还介绍了常用的选项。

习 题

1. 使用“ls -l”命令有以下输出：
-rw-rw-r-- 1 longkey longkey 16 20A 24 22:23 chap1.txt
请解析各输出部分的含义。
2. 命令 chmod 664 test.txt 的结果是什么？
3. 增加所有用户对 test.txt 文件的读权限应使用什么命令？
4. find 命令和 grep 命令各用于什么目的？
5. 建立软链接和硬链接后，用“ls -l”命令显示链接，有何差别？
6. 用 crontab 命令，如果要每星期一 12 时执行命令 echo hellolmail root，作业应如何书写？
7. 网上有一软件包 soft.tar.gz，下载后应首先如何处理？
8. 测试自己的主机和某一主机是否通信正常，通常使用_____命令。
A. telnet B. host C. ping D. ftp
9. 稳妥的关机命令是什么？
10. 管理员想终止用户 user1 的登录进程，应采用哪些命令？
11. 查询软件包 tcp_wrappers 是否已经安装的完整命令是什么？

第 3 章 vi 编辑器的使用

vi 是 Linux 系统中一种功能强大、界面友好的编辑器，熟练掌握 vi 命令及其使用技巧往往能起到事半功倍的作用，大大提高编程工作的效率，因此我们要学习它的使用方法。初步接触 vi，你可能会觉得它的界面不太友好，不容易掌握，可是一旦你掌握了 vi 的命令，就可以感觉到它强大的功能与高效。而且 vi 相对来说较小，无论你使用哪种 Linux 系统，都可以使用 vi。在很多系统中，可能只有 vi 供你选择。

3.1 vi 的工作模式

在使用 vi 之前，首先应该了解一下 vi 的工作模式。

vi 有两种工作模式：编辑模式和指令模式。在 vi 中用户可以在这两种模式间切换。

编辑模式：用来输入和编辑文件的模式，屏幕上会显示用户的键入，按键不是被解释为命令执行，而是作为文本写到用户的文件中。

指令模式：用来编辑、存盘和退出文件的模式。运行 vi 后，首先进入指令模式。此时输入的任何字符都被视为指令对待，键入的命令不会在屏幕上显示。

状态行：屏幕底部一行，通常是第 24 行，被 vi 编辑器用来反馈编辑操作结果。错误消息或者提供信息的信息会在状态行中显示出来。vi 还会在 24 行显示那些以冒号(:)或者问号(?)开头的命令。

如果从指令模式切换到编辑模式，则可以按【Insert】键；如果从编辑模式切换到指令模式，则可以按【Esc】键。如果不能断定目前处于什么模式，则可以多按几次【Esc】键，这时系统会发出蜂鸣声，证明已经进入指令模式。

【注意】 Linux 下的命令是大小写敏感的。

3.2 vi 的启动和退出

1. 启动 vi

要进入 vi，可以直接在系统提示字符下键入 vi，按空格，然后再输入文件名(本例中以 test.txt 作为文件名)，像下面一行：

```
vi test.txt
```

图 3-1 显示了用 vi 新建文件 test.txt 的初始画面。

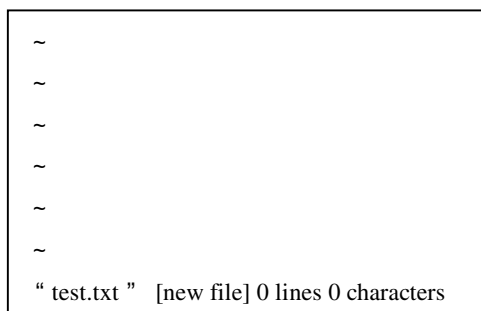


图 3-1 进入 vi 的初始化画面

vi 可以自动帮你载入所要编辑的文件或是开启一个新文件。如果 test.txt 文件已存在，vi 就会在屏幕上显示文件的第一页(前 23 行)。如果 test.txt 是一个新文件，vi 就会清屏，光标会出现在屏幕的左上角，屏幕左方会出现波浪符号“~”，凡是列首有该符号就表示此列目前是空的。

【说明】 状态行说明文件名，并且说明这是一个新文件。

图 3-1 中不是全屏幕显示(24 行)。

【注意】 一般在使用 vi 时，后边都要跟文件名。

接下来就可以按照自己的要求录入和编辑文件。在编辑模式中，可以使用四个方向键、【Ins】、【Del】、【Tab】、【Return】和【Backspace】等按键进行简单的文本编辑了。

【技巧】 如何创建文件“-file”呢？

如果想创建文件“-file”，键入“vi -file”会显示 invalid option，原来由于文件名的第一个字符为“-”，Linux 把文件名当作选项了，可以用“vi -- -file”来创建文件，同理，在使用其他命令的时候也是一样的，比如想查看“-file”文件的内容，则应该用“cat -- -file”。

2. 退出 vi

要离开 vi 可以在指令模式下键入“:q”，“q!”或“:wq”离开(注意冒号)。

(1) :q 如果用户只是读文件的内容而未对文件进行修改，可以使用“:q”退出 vi；如果用户对文件的内容作了修改，则用“:q”退出 vi，那么 vi 在屏幕的底行会提示下面的信息，vi 编辑器还保留在屏幕上：

No write since last change (:q! overrides).

(2) :q! 如果用户对文件的内容作了修改，然后决定要放弃对文件的修改，可以使用“:q!”强行退出 vi，在这种情况下文件的内容不变。

(3) :wq 在大多数情况下，用户在编辑结束时，用“:wq”命令保存文件，然后退出 vi。

(4) :n,mw filename 该指令将第 n~m 行的文本保存到指定的文件 filename 中。

(5) ZZ 该指令表示快速保存文件的内容，然后退出 vi，功能和“:wq”一样。

(6) :w! vi 编辑器通常防止覆盖一个已存在的文件。比如用户键入“:w test.txt”并按回车键，而 test.txt 文件已存在时，vi 会显示如下的信息提出警告：

"test.txt" File exist - use ":w!" to overwrite"

【注意】 ZZ 命令的前面不用冒号，而且也不需要键入【Return】完成命令。只需键入“ZZ”，整个操作就完成了。

3.3 vi 长指令和短指令

vi 的指令分为两种：长指令和短指令。

长指令以冒号开头，键入冒号后，在屏幕的最末尾一行会出现冒号提示符，等待用户键入指令，输入完指令后回车，vi 就会执行该指令。

短指令和快捷键相似，键入短指令之后，vi 不会给任何提示就直接执行。

接下来我们以分组的形式来介绍 vi 常用的指令。

(1) 输入输出命令的作用如表 3-1 所示。

表 3-1 输入输出命令

命 令	作 用
a	在光标后输入文本
A	在当前行末尾输入文本
i	在光标前输入文本
I	在当前行开始输入文本
o	在当前行后输入新一行
O	在当前行前输入新一行

(2) 光标移动命令的作用如表 3-2 所示。

表 3-2 光标移动命令

命 令	作 用
B	移动到当前单词的开始
e	移动到当前单词的结尾
w	向后移动一个单词
h	向前移动一个字符
j	向下移动一行
k	向上移动一行
l	向后移动一个字符

(3) 删除操作命令的作用如表 3-3 所示。

表 3-3 删除操作命令

命 令	作 用
x	删除光标所在的字符
dw	删除光标所在的单词
d\$	删除光标至行尾的所有字符
D	同 d\$
dd	删除当前行

【注意】可在删除命令前加上数字，如：dd5 表示删除 5 行。

(4) 改变与替换命令的作用如表 3-4 所示。

表 3-4 改变与替换命令

命 令	作 用
r	替换光标所在的字符
R	替换字符序列
cw	替换一个单词
ce	同 cw
cb	替换光标所在的前一字符
c\$	替换自光标位置至行尾的所有字符
C	同 c\$
cc	替换当前行

(5) 查询命令的作用如表 3-5 所示。

表 3-5 查 询 命 令

命 令	作 用
/abc	向后查询字串“abc”
?abc	向前查询字串“abc”
n	重复前一次查询
N	重复前一次查询，但方向相反

【注意】n 和 N 命令要配合“/”和“?”使用。

(6) 拷贝与粘贴命令的作用如表 3-6 所示。

表 3-6 拷贝与粘贴命令

命令	作 用
yw	将光标所在单词拷入剪贴板
y\$	将光标至行尾的字符拷入剪贴板
Y	同 y\$
yy	将当前行拷入剪贴板
p	将剪贴板中的内容粘贴在光标后
P	将剪贴板中的内容粘贴在光标前

【技巧】 如何同时对同一行进行连续多次拷贝？

利用 yy 命令，后面紧跟要拷贝的次数，然后再用 p 命令。

如何进行块拷贝？

有两种方法可以实现。第一种方法：按下 v 键，光标所在的位置就会反白，然后可以移动光标来选择范围，接着按 y 键将所选块拷入剪贴板，最后按 p 键将剪贴板中的内容粘贴在光标后。第二种方法：首先将光标移动到要粘贴的位置，拖动鼠标左键选定要拷贝的内容，然后按下鼠标的右键即可完成块拷贝。

(7) 文件保存及退出命令的作用如表 3-7 所示。

表 3-7 文件保存及退出命令

命 令	作 用
:q	不保存退出
:q!	不保存强制性退出
:w	保存编辑
:w filename	存入文件 filename 中
:w! filename	强制性存入文件 filename 中
:wq	保存退出
:x	同 :wq
ZZ	同 :wq

(8) 其他命令的作用如表 3-8 所示。

表 3-8 其他 vi 命令

命 令	作 用
u	取消上一次的操作
U	可以恢复对光标所在行的所有改变
J	把两行连接到一起
:set	用来设置或浏览 vi 系统当前的选项
:X	对所编辑的文件进行简单加密

【技巧】 如何显示 vi 的系统设置？

使用“:set”命令即可。发出不带参数的 set 命令只显示用户设置的选项。也可以将 set 命令缩写为 se。要在同一行设置许多选项，用 se 命令并用一个空格分隔选项，如下面的例子：

```
:se showmode report = 1 wm = 5 ic nu
```

如果要查看所有选项的列表，键入“:set all”。

如何用 vi 对所编辑的文件进行简单加密？

在 vi 当中要对所编辑的文件进行简单加密，可以在命令行模式下键入“:X”(无括号，只有冒号和 X)，然后在提示“Enter the encrypt key”时输入口令字串就可以加密了。但是别忘了要保存，保存的时候会发现在屏幕底部有“[crypted]”字样，下次打开该加密文件会要求你输入口令字串，如果密码错误，那么显示的将是乱码。

3.4 vi 高级应用

3.4.1 设置 vi 环境

vi 编辑器的行为可以通过设置编辑参数来定义，并且有许多种方法可以进行这种设置。最直接的方法是使用 vi 的 set 命令进行设置。这种情况下，vi 在进行设置前必须处于指令状态。使用这种方法的用户可以设置任何选项，但是选项的改变是临时的，并且只在用户当前编辑会话下有效。当用户退出 vi 编辑器时，设置会被丢弃。

本小节介绍一些有用的 vi 参数，表 3-9 对它们进行了汇总(按字母顺序列出)。大多数选项名有缩写形式，用户进行设置时既可以使用选项名的全称，也可以使用缩写。

表 3-9 vi 选 项

选 项	缩 写	功 能
autoindent	ai	将新行与前一行的开始对准
ignorecase	ic	在搜索选项下，忽略大小写
magic	-	在搜索时，允许使用特殊字符
number	nu	显示行号
report	-	告知用户最后一个命令作用行的行号
scroll	-	设定使用[Ctrl-d]命令翻滚的行数
shiftwidth	sw	设定缩进空格数，一般与 autoindent 一同使用
show mode	smd	在屏幕右角显示 vi 编辑器模式
terse	-	缩短错误信息
wrapmargin	wm	将右边界设定为一定的字符个数

1. autoindent 选项

autoindent 选项将用户键入的每个新行与前一行的开始对齐。该选项对于使用 C 等其他结构化程序设计语言编写程序时十分有用。使用【Ctrl-d】可减少一级缩进，每次执行【Ctrl-d】，会增加一个由 shiftwidth 选项指定的数值。本选项的默认值为 noai。

2. ignorecase 选项

vi 编辑器提供大小写敏感的搜索，也就是说它区分大写字母和小写字母。要使 vi 忽略大小写，键入“ :set ignorecase ”并按回车键。要返回大小写敏感状态，键入“ :set noignorecase ”并按回车键。

3. magic 选项

某些符号(如方括号[])在用于搜索时有特殊的含义。当用户将这些符号开头置为 nomagic 时，这些符号不再有特殊含义。

4. number 选项

vi 编辑器一般情况下不显示每行的行号。显示行号可以使用户对自己文件的大小及自

己正在编辑文件的哪一部分等心里有数。要显示行号，键入“:set number”，然后按回车键。如果不希望显示行号，键入“:set nonumber”并按回车键。

【注意】行号并不是文件的一部分，它们只有在 vi 编辑器中才会显示。

5. report 选项

vi 编辑器对用户的编辑工作并不给予任何反馈。例如，如果用户键入“dd”，vi 删除当前行文本，但不会在屏幕上显示任何确认消息。如果希望在屏幕上看到自己编辑的反馈信息，用户可以使用 report 选项来实现。该参数被设为使 vi 编辑器报告发生变化的行的最小行数。

要将 report 选项设为 2 行时有效，键入“:set report = 2”并按回车键。于是，当用户的编辑工作作用两行时，vi 显示相应报告。例如，删除两行并复制两行，将在屏幕底部产生类似下面的报告信息：

```
2 lines deleted
2 lines yanked
```

6. scroll 选项

scroll 选项用于设定用户在使用【Ctrl-d】时希望滚动的行数。例如，要想使屏幕滚动 5 行，键入“:set scroll = 5”并按回车键。

7. shiftwidth 选项

该选项设定在设置了自动缩进时，使用【Ctrl-d】时的空格数。该选项的默认设置为“sw = 8”。例如，要把该设置改为 10，键入“:set sw = 10”并按回车键。

8. showmode 选项

vi 编辑器并不显示任何可见的反馈信息来告知当前是处于文本输入模式还是指令模式，这可能导致混淆，尤其是对于新手。用户可以设置 showmode 选项来提供可见的反馈到屏幕，或者说 showmode 选项在状态行上指示你所处的模式。

要打开 showmode 选项，键入“:set showmode”并按回车键。接着，根据用户需要在文本输入和指令模式之间切换，而 vi 在屏幕的右下角显示不同的信息。如果用户键入“a”或“A”切换到编辑模式，vi 显示 APPEND MODE；如果用户键入“i”或“I”，vi 将显示 INSERT MODE；如果用户键入“O”或“o”，vi 显示 OPEN MODE 等等。

这些信息将一直显示在屏幕上，直到用户按【Esc】键切换到指令模式。当屏幕上没有信息时，vi 处于指令模式。要关闭 showmode 选项，键入“:set noshowmode”并按回车键。

9. terse 选项

该选项使 vi 编辑器显示缩短的错误消息。该选项默认值为 noterse。

10. wrapmargin 选项

该选项定义右边距。用户的终端屏幕通常为 80 列。当键入到行的末尾时(超过第 80 列)，屏幕即开始一个新行，这就是所说的行回绕。在用户按回车键时，屏幕同样开始一个新行。因此，屏幕上一行的长度可以为 1~80 个字符之间的任何长度。但是，vi 编辑器只在用户

按回车键时，才在用户文件中生成一个新行。如果用户在按回车键前键入了 120 个字符，这时键入的文本看起来是在 2 行，但实际在文件中，这 120 个字符只在 1 行中。

过长的行在文件打印时可能会出现问题，并且屏幕显示的行号与实际文件中的行号相对应时容易产生混淆。最简单的限制行长度的方法是在到达屏幕行末尾前按回车键。另一种方法是设定 `wrapmargin` 参数以使 vi 编辑器自动插入回车。

例如，要将 `wrapmargin` 设为 10(10 是从屏幕右边界计数的字符的个数)，键入“`wm = 10`”并按回车键。于是当用户键入到第 70 列时，vi 编辑器强迫回车，开始一个新行，以便留出右边距。如果用户正在键入一个字时超过第 70 列，vi 编辑器将把该字整个移至新行。这也意味着右边界可能会对不齐。`wrapmargin` 选项的默认值是 0，要关闭这个选项，键入“`:set wrapmargin = 0`”并按回车键即可。

3.4.2 缩写与宏

vi 编辑器为用户提供一些捷径，以使用户的输入更快速、更简单。“`:ab`”和“`:map`”是两个用于该目的的命令。

1. 缩写操作符

缩写操作符“`:ab`”(缩写)命令使得用户给任何字符串指定缩写，该功能可以帮助用户提高输入速度。用户可以为自己经常输入的文本选择一个易记的缩写，在 vi 编辑器中设置缩写后，就可使用该缩写代替原来的文本。例如，要缩写本书中常用的文本 Unix Operating System，键入“`:ab uno Unix Operating System`”并按回车键。

在这个例子中，“`uno`”是赋给 Unix Operating System 的缩写，因此，当 vi 处于文本输入模式时，任何时间用户键入“`uno`”接着键入一个空格时，vi 都将 `uno` 变为 Unix Operating System。如果 `uno` 是另一个字的一部分，如 `unofficial`，则并不会发生改变。vi 通过 `uno` 前后的空格来识别出 `uno` 是一个缩写，并把它扩展。

要取消一个缩写，用户可以使用“`:unab`”(未缩写)操作符。例如，要取消 `uno` 缩写，键入“`:unab uno`”并按回车键即可。

要想列出已经设置了哪些缩写，键入“`:ab`”并按回车键。

【注意】 缩写在 vi 编辑器指令模式下进行指定，并用于 vi 的文本输入模式下。

编写的指定是临时的，它们只在当前编辑会话中有效。

【实例 3.1】

- (1) 键入“`:ab lc linux course`”并按回车键，将 `lc` 指定为 `linux course` 的缩写。
- (2) 键入“`:ab 123 one,two,three,etc.`”并按回车键，将 `123` 指定为 `one,two,three,etc.` 的缩写。
- (3) 键入“`:ab`”并按回车键，显示所有指定的缩写：

```
lc    linux course
123   one,two,three,etc.
```

- (4) 键入“`:unab 123`”并按回车键，取消 `123` 缩写。

2. 宏操作符

宏操作符(`map`)使用户能将一系列键指定给某一键。如同缩写操作符给用户一个文本输入模式下的捷径一样，`map` 给用户一个在指令模式下的捷径。例如，将指令 `dd` 指定为 `q`，

键入 “:map q dd” 并按回车键。此后, 当 vi 处于指令模式时, 每当用户键入 q 时, vi 应删除光标所在的行。

要取消一个 map 指定, 用户可以使用 “:unmap” 操作符。键入 “:unmap q” 并按回车键。

要查看 map 键的列表和它们指定的内容, 键入 “:map” 并按回车键。

用户也可以使用 map 指令为自己的终端指定功能键。在这种情况下, 用户键入 “#n” 作为键名, n 代表功能键号。例如, 要指定 dd 到【F2】, 键入 “:map #2 dd” 并按回车键即可。此后, 如果用户在 vi 的指令模式下按【F2】键, vi 应删除光标所在的行。

【注意】 用户在 vi 中创建的映射键是临时的, 只对当前编辑会话有效。

映射键在 vi 处于指令模式下进行指定和使用。

如果【Return】或【Esc】是映射键的一部分, 用户必须在回车键和【Esc】前加【Ctrl-v】。

【实例 3.2】

下面例子显示部分指定键。

(1) 键入 “:map V /linux” 并按回车键, 将 V 键指定为搜索 linux 的搜索指令。

(2) 键入 “:map #3 yy” 并按回车键, 将【F3】指定为拷贝一行。

(3) 键入 “:map” 并按回车键, 显示已经指定的键:

```
V          /linux
#3         yy
```

【实例 3.3】

假设用户希望在文件中查找 “linux”, 并将它替换为 “LINUX”。进行下面的操作:

(1) 键入 “:/linux” 并按回车键, 查找单词 “linux”。

(2) 键入 “cwLINUX”, 然后按【Esc】键, 将 “linux” 改为 “LINUX” 并返回 vi 指令模式。

在映射键指定中, 命令行中按【Ctrl-v】【Return】来代表回车, 用【Ctrl-v】【Esc】来代表【Esc】键。这样, 要映射前面的命令到一个键中, 比如说 V 键, 键入 “:map v /linux”, 接着按【Ctrl-v】【Return】, 然后键入 “cwLINUX”, 再按【Ctrl-v】【Esc】。该命令行中使用了不可打印字符【Ctrl-v】和【Esc】, 所以用户看到的屏幕如下所示:

```
:map v /linux ^McwLINUX^M
```

3.4.3 “.exrc” 文件

用户在 vi 编辑器中所设置的所有选项都是临时的, 当用户退出 vi 时, 它们都会失效。要使这些设置成为永久的, 而不需在每次使用 vi 时重新设置, 可以将选项的设置保存到文件 “.exrc” 中。

【注意】 以 “.” (点)开头的文件被称为隐藏文件。

当用户打开 vi 编辑器时, 它自动查看用户当前工作目录中的 “.exrc” 文件, 并根据在文件中找到的内容设置编辑环境。如果 vi 没有在当前目录中发现 “.exrc” 文件, 它将查找用户的主目录, 并根据在那里发现的 “.exrc” 文件设置编辑环境。如果 vi 一个 “.exrc” 文件也没有找到, 则它对选项使用默认值。

vi 检查 “.exrc” 文件存在的方式给用户提供了强大的工具，用户可以根据自己的不同的编辑需要定义 “.exrc” 文件。例如，可以创建一个通用的 “.exrc” 文件存在主目录。用户可以用 vi 创建一个 “.exrc” 文件，或修改现有的 “.exrc” 文件。

【实例 3.4】

创建一个 “.exrc” 文件，键入 “vi .exrc” 并按回车键，然后输入用户想要的 set 和其他命令。下面是一个具体的例子：

```
set report = 0
set showmode
set number
set ic
set wm = 10
set scroll = 5
ab uop UNIX Operating System
map q dd
```

【注意】 文件名头部不要忘记 “.” (点)。

“.exrc” 属于启动文件。

还有其他类似于 “.exrc” 用途的启动文件。

3.4.4 运行 Shell 命令

用户可以在 vi 的命令行运行 Linux Shell 命令。这一方面的特性使用户可以临时抛弃 vi 来运行 Shell 命令。“!” 通知 vi 后面是一个 Shell 命令。例如，要在 vi 编辑器中运行 date 命令，键入 “:!date” 后按回车键。vi 编辑器将清除屏幕，执行 date 命令，我们可以看到类似如下的屏幕显示：

```
Sat Feb 8 14:00:52 EDT 2003
[Hit any key to continue]
```

按任意一个键即可返回 vi 编辑器，并可在前面离开的地方继续编辑。如果用户希望，也可以将 Shell 命令执行的结果读进来并加到用户文本中。使用 “:r”(读取)命令，后面紧跟 “!” 和相应的 Shell 命令来将命令的结果写到用户的文本中。

【实例 3.5】

要读取系统的时间和日期，键入 “:r!date” 后按回车键，vi 响应，将当前系统日期和时间放在当前行下面。

vi 编辑器保持在文本输入模式。

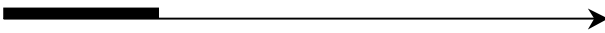
```
This is a test!
We are from Shenzhen.
The date and time is  Sat Feb 8 14:00:52 EDT 2003
~
~
```


第二篇



系统管理

第4章 用户和组的管理



Linux 系统是一个多用户的操作系统，任何一个要使用系统资源的使用者(即用户)，都必须首先申请一个账号，然后用这个账号登录系统。用户的账号一方面可以帮助系统对使用系统的用户进行跟踪，并控制用户对系统资源的访问；另一方面也可以帮助用户组织文件，并为用户提供安全性的保护。组是具有共同特性的用户的逻辑集合，使用组有利于管理员分批管理用户，提高工作效率。一个用户(也称为账号或账户)可以属于多个组，例如：某公司有技术组和领导组，陈某是该公司的技术主管，则他既属于技术组也属于领导组。

账号的管理主要包括用户的管理、组的管理和口令的管理。

4.1 用户的管理

4.1.1 Linux 下的用户

Linux 下的用户可以分为三类：超级用户、系统用户和普通用户。超级用户的用户名为 root，它具有一切权限，只有进行系统维护(例如：建立用户等)或其他必要情形下才用超级用户登录，以避免系统出现安全问题。系统用户是 Linux 系统正常工作所必需的内建的用户，主要是为了满足相应的系统进程对文件属主的要求而建立的，系统用户不能用来登录，例如：bin、daemon、adm、lp 等用户。而普通用户是为了让使用者能够使用 Linux 系统资源而建立的，我们的大多数用户属于此类。

每个用户都有一个数值，称为 UID。超级用户的 UID 为 0，系统用户的 UID 一般为 1~499，普通用户的 UID 为 500~60 000 之间的值。

4.1.2 账号系统文件

不像 Windows 2000 那样有专门的数据库用来存放用户的信息，Linux 系统采用纯文本文件来保存账号的各种信息，其中最重要的文件有/etc/passwd、/etc/shadow、/etc/group 这几个。因此账号的管理实际上就是对这几个文件的内容进行添加、修改和删除记录行的操作。我们可以使用 vi 或其他编辑器来更改它们，也可以使用专门的命令来更改它们。不管以哪种形式管理账号，了解这几个文件的内容十分必要。Linux 系统为了自己的安全，缺省情况下只允许超级用户更改它们。

1. /etc/passwd 文件

/etc/passwd 文件是账号管理中最最重要的一个文件，它是一个纯文本文件。每一个注册用户在该文件都有一个对应的记录行，这一记录行记录了此用户的必要信息。

【实例 4.1】

```
[root @redflag /root]#cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
.....
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nol
rpc:x:32:32:Portmapper RPC user:/bin/false
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/bin/false
mailnull:x:47:47::/var/spool/mqueue:/dev/null
test:x:500:500:/home/test:/bin/bash
```

passwd 文件中的每一行由 7 个字段的数据组成，字段之间用 “:” 分隔，其格式如下：

账号名称：密码：UID：GID：个人资料：主目录：Shell

字段说明：

账号名称：用户登录 Linux 系统时使用的名称。

密码：这里的密码是经过加密后的密码(一般是采用 MD5 加密方式)，而不是真正的密码，若为 “x”，说明密码经过了 shadow 的保护(我们随后就介绍)。

UID：用户的标识，是一个数值，Linux 系统内部使用它来区分不同的用户。

GID：用户所在基本组的标识，是一个数值，Linux 系统内部使用它来区分不同的组，相同的组具有相同的 GID。

个人资料：可以记录用户的完整姓名、地址、办公室电话、家庭电话等信息。

主目录：类似 Windows 2000 的个人目录，通常是 /home/username，这里 username 是用户名，用户执行 “cd ~” 命令时当前目录会切换到个人主目录。

Shell：定义用户登录后激活的 Shell，默认是 Bash Shell。

从 passwd 文件中可以看到，第一行是 root 用户，紧接的是系统用户，普通用户通常在文件的尾部。当然它们所在的顺序并不是很重要的。

【注意】如果出现两个紧挨的 “:”，说明这一字段为空。

2. /etc/shadow

在 passwd 文件中，有一个字段是用来存放经过加密的密码。我们先来看以下 passwd 文件的权限：

```
[root @redflag /root]#ls -l /etc/passwd
-rw-r--r-- 1 root root 1092 3月 12 18:00 /etc/passwd
```

可以看到任何用户对它都有读的权限。如果不让所有的用户对它有读的权限，Linux 系统会出现一些问题。虽然密码已经经过加密，但还是不能避免别有用心的人轻易地获取加密后的密码后进行解密(如：字典法、穷举法等)。于是 Linux 系统对密码提供了更多一层的保护，即把加密后的密码重定向到另一个文件/etc/shadow。

```
[root @redflag /root]#ls -l /etc/shadow
-r----- 1 root root 758 3月 12 18:00 /etc/shadow
```

现在只有超级用户能够读取 shadow 的内容，密码显然安全多了，因为其他人即使想获得加密后的密码也不容易了。

红旗 Linux 3.0 缺省采用 shadow 文件。密码如果经过 shadow 保护，在/etc/passwd 文件中，每一记录行的密码字段会变成“x”，并且在/etc 目录下存在文件 shadow。

【实例 4.2】

```
[root @redflag /root]#cat /etc/shadow
root::12123:0:99999:7:::
bin*:12123:0:99999:7:::
daemon*:12123:0:99999:7:::
adm*:12123:0:99999:7:::
.....
sshd:!:12123:0:99999:7:::
rpc:!:12123:0:99999:7:::
rpcuser:!:12123:0:99999:7:::
mailnull:!:12123:0:99999:7:::
test::12123:0:99999:7:::
```

和 passwd 文件类似，shadow 文件中的每行由 9 个字段组成，格式如下：

用户名：密码：最后一次修改时间：最小时间间隔：最大时间间隔：警告时间：
不活动时间：失效时间：标志

字段说明：

用户名：和/etc/passwd 文件中相对应的用户名。

密码：存放加密后的口令(密码)。

最后一次修改时间：用户最后一次修改口令的时间(从 1970-1-1 起计的天数)。

最小时间间隔：两次修改口令允许的最小天数。

最大时间间隔：口令保持有效的最多天数，即多少天后必须修改口令。

警告时间：从系统提前警告到口令正式失效的天数。

不活动时间：口令过期多少天后，该账号被禁用。

失效时间：指示口令失效的绝对天数(从 1970-1-1 开始计算)。

标志：未使用。

shadow 文件中，密码字段为“*”表示用户被禁止登录，为“!”表示密码未设置，为“!”表示用户被锁定。

3. pwconv 和 pwunconv

安装 Linux 系统时，系统缺省采用 shadow 来保护密码。如果安装 Linux 时未启用 shadow，可以使用 pwconv 命令启用 shadow。注意用 root 用户登录来执行该命令。

```
[root @redflag /root]#pwconv
```

执行的结果是/etc/passwd 文件中的密码字段被改为“x”，同时产生/etc/shadow 文件。

相反，如果要取消 shadow 功能，使用 pwunconv 命令。

【实例 4.3】

```
[root @redflag /root]#pwunconv
[root @redflag /root]#cat /etc/passwd
root:$1$HgZ5TnjG$1lE4Umcpx0s9ESNHB0By0:0:0:root:/root:/bin/bash
.....
test:$1$dm5d2WCQ$6RUCGuCRAz9RUAOytwbk3/:500:500:./home/test:/bin/bash
```

可以看到 passwd 密码字段存放了加密后的密码，已不是“x”了。

【注意】 虽然可以取消 shadow 功能，但是建议不要这么做，除非有合适的理由。取消 shadow 功能的结果将严重影响系统的安全性。

4.1.3 创建新的用户

创建新的用户要完成以下几个工作：

- (1) 在/etc/passwd(和/etc/shadow)中添加一行新的记录；
- (2) 创建用户的个人主目录，并赋权限；
- (3) 在用户的个人主目录设置默认的配置文件的；
- (4) 设置用户的初始口令。

创建用户可以用手工创建或使用专门的命令创建。手工创建就是管理员一步一步完成以上的工作；使用专门的命令，则是由 Linux 提供的命令来完成以上的工作。使用后者效率较高，如果不是创建有特殊要求的用户，建议使用后者。

创建用户的命令为 useradd 或 adduser，一般来说这两个命令是没有差别的，先用 root 用户登录后，再执行它们。useradd 命令的格式如下：

useradd [参数] 用户名

参数选项：

- c comment：注释行，一般为用户的全名、地址、办公室电话、家庭电话等。
 - d dir：设置个人主目录，默认值是/home/用户名。
 - e YYYY-MM-DD：设置账号的有效日期，此日期后用户将不能使用该账号。要启用 shadow 才能使用此功能。
 - f days：指定密码到期后多少天永久停止账号，要求启用 shadow 功能。
 - g group：设定用户的所属基本组，group 必须是存在的组名或组的 GID。
 - G group：设定用户的所属附属组，group 必须是存在的组名或组的 GID，附属组可以有多个，组之间用“，”分隔开。
 - k Shell-dir：和“-m”一起使用，将 Shell-dir 目录中文件复制到主目录，默认是/etc/skel 目录。
 - m：若用户主目录不存在，创建主目录。
 - s Shell：设置用户登录后启动的 Shell，默认是 Bash Shell。
 - u UID：设置账号的 UID，默认是已有用户的最大 UID 加 1。
- 例如：

```
[root @redflag /root]#useradd user1
```

在/etc/passwd 文件中会看到增加了一行：

```
user1:x:501:501::/home/user1:/bin/bash
```

系统自动指定用户 user1 的 UID 为 501 ,同时还自动创建组名为 user1 的用户组(其名称和用户名相同,其 GID 值也和 UID 值相同),在/home 目录下还创建了目录 user1,用户的登录 Shell 是 Bash Shell。

```
[root @redflag /root]#ls -l /home
```

```
.....
```

```
drwx-----  9 user1      user1          472  3月 13 23:06 user1
```

```
.....
```

user1 对/home/user1 目录有所有权限,其他用户无任何权限。同时在/etc/shadow 文件中也会增加一行:

```
user1:!!:12125:0:99999:7:::
```

注意密码字段的内容为“!!”,表示密码没有设置。

```
[root @redflag /root]#useradd -g user1 -c “user2,-755-123456” user2
```

以上命令创建 user2 用户,并把它加入到组 user1 中,同时加上用户的注释。

4.1.4 修改用户的属性

1. 修改用户的密码

(1) passwd 用户名——修改用户的密码。

修改用户的密码需要两次输入密码确认。密码是保证系统安全的一个重要措施,在设置密码时,不要使用过于简单的密码。密码的长度应在 8 位或 8 位以上,由数字和英文组合而成,不要采用英文单词等有意义的词汇。一个便于记忆并且有效的密码“wabjtm!”,是“我爱北京天安门!”的汉字拼音首字母组合。密码的更改间隔天数不要太大,并且不得重复使用。

用户的密码也可以自己更改,这时使用不带用户名的 passwd 命令。

```
[root @redflag /root]#passwd
```

修改 root 用户自己的密码。

(2) passwd -d 用户名——删除用户的密码。

【实例 4.4】

```
[root @redflag /root]#passwd -d user2
```

```
Changing password for user user2
```

```
Removing password for user user2
```

```
passwd: Success
```

以上命令删除了用户 user2 的密码。

2. 修改用户的 Shell 设置

如果用户的默认 Shell 不合适,可以把它改成任何已经加入到/etc/Shell 文件中的 Shell。使用 chsh 命令改变用户的 Shell,格式如下:

```
chsh 用户名
```


【实例 4.5】

```
[root @redflag /root]#chsh user2
Changing Shell for user2.
New Shell [/bin/bash]: /bin/csh
Shell changed.
```

指定的 Shell 一定要在/etc/Shells 中存在，否则会导致用户无法登录。也可以使用下面要介绍的命令 usermod 来改变用户的 Shell 设置。

3. usermod [参数] 用户名——改变用户的属性

参数选项：

- c comment：改变用户的注释，如：全名、地址、办公室电话、家庭电话等。
- d dir：改变用户的主目录，如果同时使用“-m”选项，原来主目录的内容会移动到新的主目录。
- e YYYY-MM-DD：修改用户的有效日期。
- f days：在密码到期的 days 天后停止使用账户。
- g GID 或组名：修改用户的所属基本组。
- G GID 或组名：修改用户的所属附加组，组之间用“,”分隔。
- l name：更改账户的名称，必须在该用户未登录的情况下才能使用。
- m：把主目录的所有内容移动到新的目录。
- p 密码：修改用户的密码。
- s Shell：修改用户的登录 Shell。
- u UID：改变用户的 UID 为新的值，改变用户的 UID 时主目录下所有该用户所拥有的文件或子目录将自动更改 UID，但对于主目录之外的文件和目录只能用 chown 命令手工进行设置。

例如：

```
[root @redflag /root]#usermod -d /home2/user2 user2
```

该命令把用户 user2 的主目录改为/home2/user2。

4. chfn 用户名——修改用户的个人信息**【实例 4.6】**

```
[root @redflag /root]#chfn user2
Changing finger information for user2.
Name [Bom]: BOM
#修改姓名。
Office [xili shenzhen]: XiLi ShenZhen
#修改办公室地址。
Office Phone [0755-2222222]: 0755-1111111
#修改办公室电话。
Home Phone [0755-3333333]: 0755-5555555
#修改家庭电话。
Finger information changed.
```

4.1.5 停止用户

将用户停用有几个不同的程度：

- (1) 暂时停止用户登录系统的权利，日后再恢复。
- (2) 从系统中删除用户，但保留用户的文件。
- (3) 从系统中删除用户，并删除用户所拥有的文件。

1. 暂停用户

暂停用户常常用于某用户在未来较长的一段时间内不登录系统的情形(如出差)。只需要利用编辑工具将 passwd 文件中的密码字段加上“*”即可(如果采用了 shadow 文件，就编辑 shadow 文件)。恢复时，把“*”删除即可。

也可以使用带“-l”参数的 passwd 命令来暂停用户。

【实例 4.7】

```
[root @redflag /root]#passwd -l user1
Changing password for user user2
Locking password for user user2
passwd: Success
```

被锁定的账户其在 passwd 文件或 shadow 文件中的记录行的密码字段会加上“!”，如下(这里是 shadow)：

```
.....
user2:!!$UKp9vz4X$BIf3RyDs9hkveDqwiosFt.:12126:0:99999:7:-1:-1:134523744
.....
```

恢复时，使用带“-u”参数的 passwd 命令，如下：

```
[root @redflag /root]#passwd -u user2
Changing password for user user2
Unlocking password for user user2
passwd: Success
```

2. 删除用户

删除一个账户可以直接将 passwd 文件中的用户记录整行删除(如采用 shadow，还要删除 shadow 文件中的记录)。也可以使用 userdel 命令：

userdel [参数] 用户名

参数选项：

-r：删除用户时将用户主目录下的所有内容一并删除，同时删除用户的邮箱(在 /var/spool/mail 下)。

例如：

```
[root @redflag /root]#userdel -r user1
```

表示删除 user1，且将 user1 下的内容删除。

【注意】删除用户时如果同时删除用户的主目录，请确认用户的主目录下的文件没有价值或事先做好备份。

3. 完全删除

Linux 系统并不提供完全删除用户所有文件的命令，带“-r”参数的 userdel 命令只能删除用户主目录下的文件和邮箱，对于用户在别的目录下所拥有的文件只能手工删除。

4.1.6 默认新用户的设置

使用 useradd 建立新用户时，新建的用户有一定的默认设置，这个设置来自 /etc/default/useradd 文件，文件中的内容如下：

```
# useradd defaults file
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

文件中的含义如下：

GROUP=100：指定默认的组为 100，该设置只有在使用“-u”选项禁止了默认私有组(也就是和新用户同名的组)时才有效，该值必须在/etc/group 文件中存在。

HOME=/home：指定新用户主目录所在的目录。

INACTIVE=-1：只有启用了 shadow 功能后才有效，指定用户密码过期后该账户将在多长时间后无效(以天数计)，-1 代表永远不过期。

EXPIRE=：只有启用了 shadow 功能后才有效，指定账户被禁止的时间。

SHELL=/bin/bash：指定默认登录的 Shell。

SKEL=/etc/skel：指定保存用户各种配置文件的目录，创建新用户时，该目录下的文件将被拷贝到用户的主目录下。

如果启用 shadow 功能，建立用户账户时还将使用另一配置文件/etc/login.defs。

【实例 4.8】

我们把有关的说明用“#”进行标注，以示和文件原有内容的区别。

```
[root @redflag /root]#cat /etc/login.defs
```

```
# *REQUIRED*
#   Directory where mailboxes reside, _or_ name of file, relative to the
#   home directory.  If you _do_ define both, MAIL_DIR takes precedence.
#   QMAIL_DIR is for Qmail
#
#QMAIL_DIR      Maildir
MAIL_DIR        /var/spool/mail
#用户邮箱所在的目录。
#MAIL_FILE      .mail
```

```
# Password aging controls:
#
# PASS_MAX_DAYS Maximum number of days a password may be used.
# PASS_MIN_DAYS Minimum number of days allowed between password changes.
# PASS_MIN_LEN Minimum acceptable password length.
# PASS_WARN_AGE Number of days warning given before a password expires.
#
PASS_MAX_DAYS 99999
#账户密码最长的使用天数。
PASS_MIN_DAYS 0
#允许更改账户密码的最短天数。
PASS_MIN_LEN 5
#账户密码最小长度。
PASS_WARN_AGE 7
#账户密码过期前提前警告的天数。
#
# Min/max values for automatic uid selection in useradd
#
UID_MIN 500
#用 useradd 命令添加账户时自动产生 UID，最小的 UID 值。
UID_MAX 60000
#用 useradd 命令添加账户时自动产生 UID，最大的 UID 值。
#
# Min/max values for automatic gid selection in groupadd
#
GID_MIN 500
#用 useradd 命令添加账户时自动产生 GID，最小的 GID 值。
GID_MAX 60000
#用 useradd 命令添加账户时自动产生 GID，最大的 GID 值。
#
# If defined, this command is run when removing a user.
# It should remove any at/cron/print jobs etc. owned by
# the user to be removed (passed as the first argument).
#
#USERDEL_CMD /usr/sbin/userdel_local

#
# If useradd should create home directories for users by default
```

```
# On RH systems, we do. This option is ORed with the -m flag on
# useradd command line.
#
CREATE_HOME      yes
#是否创建用户主目录。
```

4.1.7 用户登录系统后环境的设定

系统管理员经常碰到用户反映“我不想每次设定我自己的命令别名，我要怎么做？”、“我想改变我注册后的提示符，我要怎么做？”这类问题。这些要求可以通过修改系统或用户的作业环境来控制。

用户登录系统后可以使用不同的 Shell，不同的 Shell 对作业环境进行控制所采用的文件不太一样。我们这里以系统默认的 Shell——Bash Shell 为例，其他 Shell 所采用的文件请参见有关的资料(例如，用 man tcsh 命令)。

使用 Bash Shell 时，有以下几个文件和用户的作业环境有关：

- (1) /etc/profile。
- (2) /etc/bashrc。
- (3) /etc/inputrc。
- (4) \$HOME/.bash_profile。
- (5) \$HOME/.bashrc。
- (6) \$HOME/.inputrc。
- (7) \$HOME/.bash_login。

前 3 个文件和系统所有的用户有关，影响所有登录的用户；后 4 个文件和某个特定用户有关，如果只想改变单一用户的作业环境，要更改这几个文件。这些文件(inputrc 和 .inputrc 除外)采用 Shell 语言编写，因此请参照 Shell 编程这一章节的内容来阅读这些文件的内容。其中，/etc/profile 是系统首先会执行的文件。

【实例 4.9】

同样，我们把有关的说明用“#”进行标注，以示和文件原有内容的区别。

```
# /etc/profile
# System wide environment and startup programs
# Functions and aliases go in /etc/bashrc
#如果用户的搜索路径中没有/usr/X11R6/bin，则把这一路径加到搜索路径中。
if ! echo $PATH | /bin/grep -q "/usr/X11R6/bin" ; then
PATH="$PATH:/usr/X11R6/bin"
fi
#设定系统产生的核心文件的大小最大为 100 MB(软限制)。
ulimit -S -c 1000000 > /dev/null 2>&1
#如果是普通用户，则 umask 值设为 022，否则为 002。
if [ `id -gn` = `id -un` -a `id -u` -gt 14 ]; then
umask 002
```

```
else
    umask 022
fi
#设定环境变量 USER(用户名)。
USER='id -un`
#设定环境变量 LOGNAME(登录名)。
LOGNAME=$USER
#设定环境变量 MAIL(用户邮箱所在处)。
MAIL="/var/spool/mail/$USER"
#设定环境变量 HOSTNAME(主机名)。
HOSTNAME=`bin/hostname`
#设定环境变量 HISTSIZE，即保存 1000 条的历史命令。使用 history 命令时可以显示用户的历史命令，能保存的历史命令数和这一变量有关。
HISTSIZE=1000
#定义键盘键位的文件是/etc/inputrc 文件。
if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
    INPUTRC=/etc/inputrc
fi
#输出以下的变量。
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC
#执行/etc/profile.d 目录下的所有 "*.sh" 文件。
for i in /etc/profile.d/*.sh ; do
    if [ -x $i ]; then
        . $i
    fi
done

unset i
#定义堆栈大小为 2048。
ulimit -s 2048 > /dev/null 2>&1
ulimit -s 2048 > /dev/null 2>&1
```

/etc/bashrc 文件的主要功能是设置一些命令别名和在 profile 文件没有定义的变量，内容如下。

【实例 4.10】

```
# /etc/bashrc
# System wide functions and aliases
# Environment stuff goes in /etc/profile

# are we an interactive Shell?
```

```

if [ "$PS1" ]; then
    if [ -x /usr/bin/tput ]; then
        if [ "x'tput kbs'" != "x" ]; then # We can't do this with "dumb" terminal
            stty erase 'tput kbs'
        elif [ -x /usr/bin/wc ]; then
            if [ "tput kbs|wc -c '" -gt 0 ]; then # We can't do this with "dumb" terminal
                stty erase 'tput kbs'
            fi
        fi
    fi
fi
case $TERM in
    xterm*)
        PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME}: ${PWD}\007"'
        ;;
    *)
        ;;
esac

#设定用户登录后的提示符，意思是：[用户名@主机名 当前目录]。
[ "$PS1" = "\s-\v\\\$ " ] && PS1="[u@h \W]\\\$ "

if [ "x$SHLVL" != "x1" ]; then # We're not a login Shell
    for i in /etc/profile.d/*.sh; do
        if [ -x $i ]; then
            . $i
        fi
    done
fi
fi

#设定一些常用的命令别名。
alias ls='ls --color --show-control-chars'
alias ls=ls --color -l --show-control-chars'
alias ls=ls --color'
alias ls=ls --color'

```

/etc/inputrc 文件主要定义或者改变一些功能键的定义，从而更好地使用命令行，一般并不对此文件进行改变。详细说明请用 man bash 命令获得。

“ .bash_profile ”、“ .bashrc ”和“ .bash_login ”是存在个人主目录下的隐含文件，更改它们的内容可以定制某一用户的作业环境。它们的内容和 profile、bashrc 内容类似，这里不再详细说明。

4.1.8 超级用户

我们已经知道 root 用户是超级用户，它具有至高无上的权利，不仅对系统任何文件都有权限(不管是否明确分配 root 对文件的权限)，还可以管理系统。root 用户的 UID 和 GID 都为 0。实际上，普通用户如果其 UID 和 GID 也都为 0，它就成了和 root 平起平坐的超级用户了。大多情况下，这样做并没有什么好处，而且还有坏处。但有时在组织中需要多个系统管理员管理同一系统，多个超级用户有利于多个管理员的责任明确。

缺省情况下，超级用户只有从/etc/securetty 文件中列出的 tty 上登录才能获得成功。普通用户成为超级用户后仍无法从 Telnet 登录。因此如果要远程管理用户，可以用普通用户从 Telnet 登录，再用 su 命令切换到超级用户来实现远程管理。

root 用户的重要性已经十分明显。和 Windows 2000 不同，如果 root 的密码丢失，有不用重新安装系统的方法。解决的办法是用红旗 Linux 启动盘启动，进入到安装状态，然后把文件系统 mount 到一个目录下(如/mnt)，随后修改/etc 目录下的文件 passwd，把 root 用户的密码字段内容删除。或者在 LILO 出现时，输入“linux single”把系统启动到单用户状态，再更改/etc/passwd 后重新启动系统到正常状态。

4.2 组 的 管 理

4.2.1 Linux 下的组和组文件

Linux 的组有私有组、系统组、标准组之分。建立账户时，若没有指定账户所属的组，系统会建立一个组名和用户名相同的组，这个组就是私有组，这个组只容纳了一个用户。而标准组可以容纳多个用户，组中的用户都具有组所拥有的权利。系统组是 Linux 系统正常运行所必需的，安装 Linux 系统或添加新的软件包会自动建立系统组。

一个用户可以属于多个组，用户所属的组又有基本组和附加组之分。在用户所属组中的第一个组称为基本组，基本组在/etc/passwd 文件中指定；其他组为附加组，附加组在/etc/group 文件中指定。属于多个组的用户所拥有的权限是它所在的组的权限之和。

Linux 系统关于组的信息存放在文件/etc/group 中。

【实例 4.11】

```
[root @redflag /root]#cat /etc/group
root:x:0:root,test
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
.....
sshd:x:74:
rpc:x:32:
rpcuser:x:29:
```



```
mailnull:x:47:  
test:x:500:  
group1:x:1000:  
user1:x:501:
```

group 文件中的每一行记录了一个组的信息，每行包括 4 个字段，字段之间用 “:” 分隔。格式如下：

组名：组的密码：GID：组成员

字段说明：

组名：组的名称，如:root、bin 等。

组的密码：设置加入组的密码，一般情况下不使用组密码，该字段通常没用。

GID：组的标识符，为数值，类似 UID。

组成员：组所包含的用户，用户之间用 “,” 分隔。大部分系统组无成员。

4.2.2 组的添加

可以手工编辑/etc/group 文件来完成组的添加，也可以用命令 groupadd 来添加用户。groupadd 命令的格式如下：

groupadd [参数] 组名

参数选项：

-g GID：指定新组的 GID，默认值是已有的最大的 GID 加 1。

-r：建立一个系统专用组，与-g 不同时使用时，则分配一个 1 ~ 499 的 GID。

例如：

```
[root @redflag /root]#groupadd -g 1000 group1
```

表示添加一个新组，组 ID 为 1000，组名为 group1。

4.2.3 组属性的修改

修改组的属性，使用 groupmod 命令，格式如下：

groupmod [参数] 组名

参数选项：

-g GID：指定组新的 GID。

-n name：更改组的名字为 name。

要改变组中的成员用户或改变组的密码使用 gpasswd 命令，例如：

gpasswd [参数] [用户名] 组名

不带参数时，即修改组密码。

参数选项：

-a：将用户加入到组中。

-d：将用户从组中删除。

-r：取消组密码。

例如：

```
[root @redflag /root]#gpasswd group1
Changing the password for group group1
New Password:
Re-enter new password:
[root @redflag /root]#gpasswd -a user1 group1
```

将用户 user1 加入到组 group1 中。

```
[root @redflag /root]#gpasswd -d user1 group1
```

将用户 user1 从组 group1 中删除。

4.2.4 文件的安全问题

我们在第 2 章已经介绍了文件和目录的权限问题，这里就权限问题做一些补充。

1. umask——改变默认权限掩码

文件权限可以通过 chmod 命令来修改。当用户创建一个新文件后，如果不使用 chmod 修改权限，则这个文件的权限是什么呢？这个文件的权限由系统默认权限和默认权限掩码共同确定，它等于系统默认权限减去默认权限掩码。Linux 系统中目录的默认权限是 777，文件的默认权限是 666。因此，有以下公式：

新目录的权限=777 - 默认权限掩码

新文件的权限=666 - 默认权限掩码

例如：

```
[root @redflag /root]#umask
022
```

以上不带任何参数的 umask 命令显示当前的默认权限掩码值。如果用户创建新的文件，文件的权限应为：

666-022=644(即 rw-r--r--)

```
[root @redflag /root]#vi test
```

```
[root @redflag /root]#ls -l test
```

```
-rw-r--r--  1 root    root           7  3 月  8 12:57 test
```

如果用户创建新的目录，目录的权限应为：

777-022=755(即 rwxr-xr-x)

```
[root @redflag /root]#mkdir testdir
```

```
[root @redflag /root]#ls -l
```

```
drwxr-xr-x  2 root    root           48  3 月  8 13:03 testdir
```

```
[root @redflag /root]#umask 002
```

把当前用户的默认权限掩码值改为 002。

【技巧】 每一个用户可以有自己的 umask 值，可以把 umask 命令放在用户的作业环境脚本里。作业环境脚本在用户个人主目录下，是文件名为 “.bash_profile” 的隐含文件。

2. setuid、setgid 和 sticky 位

在这之前，我们介绍了文件权限可以用 3 个八进制来表示。实际上文件权限可以有 4

个八进制位(如,“0644”),只不过第 1 位八进制为“0”,可以省略。第 1 位的八进制如果用二进制表示,有 3 位二进制,依次表示:

用户标识位:组标识位:粘贴位

用户标识位:如为“1”,表示任一用户执行该程序时,正在运行的程序的拥有者将是文件的所有者,而不是启动程序的用户。例如,有一文件 testprg,其拥有者为 root,该文件设置了用户标识位,用户 user1 运行了该程序,则正在运行的程序 testprg 将归 root 拥有, testprg 程序将具有 root 用户的权限。这使得普通用户在权限不够时的情况下仍然可以运行某些要求有高权限的命令,但同时带来了安全的问题,所以要慎重使用用户标识位。

组标识位:和用户标识位类似,只不过用户换成了组而已。

粘贴位:如果某个程序不断被运行,可以设置粘贴位以提高速度。设置了粘贴位后,系统会在内存创建一个该程序的拷贝,下次调用时速度会加快。

仍然使用 chmod 命令来设置以上 3 个位,例如:

```
[root @redflag /root]#chmod 4644 testprg
```

注意以上命令有了 4 个八进制,第 1 位八进制用于设置用户标识位。

```
[root @redflag /root]#ls -l testprg
```

```
-rwSr--r-- 1 test test 13 3月 15 10:00 testprg
```

用户标识位用“--s-----”或“--S-----”来显示,即它占用了文件拥有者的执行权限位来表示。如果该位为“s”,表示设置了用户标识位的同时文件拥有者对文件还有执行权限;如果该位为“S”,表示设置了用户标识位而文件拥有者对文件没有执行权限。

组标识位用“-----s---”或“-----S---”来显示,即它占用了文件所属组的执行权限位来表示。如果该位为“s”,表示设置了组标识位的同时文件所属组对文件还有执行权限;如果该位为“S”,表示设置了组标识位而文件所属组对文件没有执行权限。

粘贴位用“-----t”或“-----T”来显示,即它占用了其他用户的执行权限位来表示。如果该位为“t”,表示设置了粘贴位的同时其他用户对文件还有执行权限;如果该位为“T”,表示设置了粘贴位而其他用户对文件没有执行权限。

使用 chmod 命令时,可以用“s”表示用户或组的标识位,用“t”表示粘贴位。

```
[root @redflag /root]#chmod 4644 testprg 等价于
```

```
[root @redflag /root]#chmod u=rws,g=r,o=r testprg
```

4.3 磁盘配额

所谓的磁盘配额,是指用户在主机上可以使用的磁盘空间的额度。限制用户所能占用的磁盘空间常常是必要的,特别是当主机作为公共服务器使用时。Linux 通过 quota 来实现磁盘配额管理。quota 可以从两个方面进行限制:一个方面可以限制用户或组占用的磁盘块数(1 块=1024 字节);另一方面可以限制用户或组所拥有的文件数(inode 数)。大多情况下我们使用块数的限制。

quota 是以文件系统为基础的,如果系统中有多个文件系统,则必须在所有文件系统上分别进行 quota 的设置。quota 目前只在 ext2 类型的文件系统上实现。

配置 quota 一般有以下几个步骤:

(1) 检查内核是否支持 quota。红旗 Linux 3.0 内核缺省是支持的，也可以用下面的命令确认：

```
[root @redflag /root]#dmesg | grep "quota"
```

应有输出：

```
VFS: Diskquotas version dquot_6.5.0 initialized
```

如内核不支持，则用 rpm 安装相应的软件包。

(2) 修改/etc/fstab 文件(关于 fstab 的内容，可进一步参见文件系统管理这一章)。

对于要启用 quota 的文件系统，要配置相应的安装选项。如下：

```
/dev/hda3 /mnt/disk1 ext2 defaults,usrquota,grpquota 1 2
```

带下划线的部分是和 quota 有关的选项。userquota 表示支持用户 quota，而 grpquota 表示支持组 quota。

(3) 重新启动系统或卸载文件系统并重新安装文件系统让 quota 选项生效，例如：

```
[root @redflag /root]#unmount /dev/hda3
```

```
[root @redflag /root]#mount /dev/hda3
```

(4) 建立 aquota.user 和 aquota.group 文件。使用 quotacheck 命令来完成这一任务，该命令的作用是检查配置了 quota 的文件系统中，各个用户或组对文件和文件数的使用情况，并在每个文件系统的根目录上建立 aquota.user 和 aquota.group 文件。第一次执行时，如果文件系统存在的文件数较多，会比较费时。

例如：

```
[root @redflag /root]#quotacheck -avug
```

```
Scan of /mnt/disk1 [/dev/hda3] done
```

```
Checked 2 directories and 0 files
```

参数选项：

-a：检查所有已安装(mount)了并且配置了配额的文件系统。

-g：检查组的配额。

-u：检查用户配额。

-v：显示检查时产生的信息。

(5) 修改系统的启动脚本，让系统启动时自动执行配额检查并启动配额功能。系统的启动脚本为/etc/rc.d/rc.local，在 rc.local 文件末加入以下语句：

```
/sbin/quotacheck -avug
```

```
/sbin/quotaoon -avug
```

重新启动系统让脚本生效。也可以不重启系统，但要手工执行一遍以上命令。

(6) 设置用户配额。设置用户配额的命令是：

```
edquota [参数] [用户名或组名]
```

参数选项：

-u：修改用户的配额。

-a：修改组的配额。

-t：修改缓冲延时。

例如：

```
[root @redflag /root]#edquota -u test
Edit block and inode quota for user test:
Device /dev/hda3 (/mnt/disk1):
Used 0KB, limits: soft=0 hard=0
Used 0 inodes, limits: soft=0 hard=0
~
```

以上命令将启动 vi 编辑器用于编辑用户 test 在每个文件系统的配额(这里只有一个文件系统配置了配额)。“Used 0KB”是用户已经使用的磁盘块数,“limits: soft”是软配额,“hard”是硬配额,“Used 0 inodes”是用户已经使用的文件数(inode 数)。

软配额:用户所使用的磁盘块数或文件数到这一配额后,系统会警告,但仍允许用户继续使用。可继续使用的时间和下面介绍的缓冲延时有关。

硬配额:用户所使用的磁盘块数或文件数到这一配额后,系统将不允许用户继续使用。

缓冲延时:用户使用的磁盘块数或文件数达到软配额后,仍允许用户继续使用的时间。如果延时到达,用户将不被允许继续使用磁盘了。例如:

```
[root @redflag /root]#edquota -t
Edit grace times for user quota:
Device /dev/hda3 (/mnt/disk1):
Block grace: 7 days Inode grace: 7 days
~
```

其中,“Block grace: 7 days”表示磁盘块数的缓冲延时;“Inode grace: 7 days”表示文件数的缓冲延时。

(7) 关闭和打开磁盘配额功能使用。

quotaoff -a 文件系统——关闭文件系统磁盘配额功能。

参数选项:

-a: 关闭所有已安装(mount)了并且配置了配额的文件系统的配额功能。

【实例 4.12】

```
[root @redflag /root]#quotaoff /dev/hda3
```

表示关闭/dev/hda3 分区的磁盘配额功能。

quotaon [参数] 文件系统——打开文件系统磁盘配额功能。

参数选项:

-a: 打开所有已安装(mount)了并且配置了配额的文件系统的配额功能。

【实例 4.13】

```
[root @redflag /root]#quotaon /dev/hda3
```

表示打开/dev/hda3 分区的磁盘配额功能。

(8) 其他命令。

repquota: 可以产生文件系统有关磁盘配额的统计。

```
[root @redflag /root]#repquota -a
*** Report for user quotas on device /dev/hda3 (/mnt/disk1)
```

Block grace time: 7 days; Inode grace time: 7 days

Block limits					File limits				
User		used	soft	hard	grace	used	soft	hard	grace

root	--	13	0	0		2	0	0	

以上命令显示了各个用户磁盘配额和文件配额的使用数。

本章小结

本章主要介绍了 Linux 系统中用户和组的管理。用户管理和组管理的主要文件有 /etc/passwd、/etc/group、/etc/shadow，用户和组的增加或删除可以通过直接编辑这几个文件实现，也可以使用命令 useradd、passwd、usermod、groupadd、gpasswd、groupmod 实现。新用户的默认设置可以通过修改/etc/login.defs 实现。用户登录的作业环境可以在/etc/profile、/etc/bashrc 等文件进行设置。磁盘配额指用户在主机上可以使用的磁盘空间或文件数的额度，要打开文件系统的配额功能应在/etc/fstab 中加上相应的选项，和磁盘配额有关的命令主要有：quotacheck、edquota、quotaoff、quotaon、repquota。

习 题

1. /etc/passwd 文件中的其中一行为“test:x:500:500::/home/test:/bin/bash”，请解析各字段的含义。
2. /etc/group 文件中的其中一行为“group1:x:1000:”，请解析各字段的含义。
3. 在/etc/fstab 文件中有一行“/dev/hda3 /mnt/disk1 ext2 defaults 1 2”，如要打开/dev/hda3 的磁盘配额功能，应加上什么选项？还要其他什么步骤？

第 5 章 设备管理

要正确管理 Linux 系统，就必须了解设备的有关概念。如果不具备这些基本的知识，将无法为系统增加新的硬件，管理和调整 Linux 系统硬件配置。相反，很好地进行设备管理不仅可以帮助我们正确地使用设备，而且可以充分地发挥各种设备的性能。

5.1 硬件设备

5.1.1 设备文件

Linux 操作系统本身对于如何控制硬盘、软驱、光驱和其他连接到系统的外围设备并无内建的指令。所有用于和外设通信的指令都包含在一个叫做设备驱动程序的文件中。该程序通常是一段汇编语言或 C 代码，用于和外设传递数据、交流信息。Linux 系统通过设备文件实现对设备和设备驱动程序的跟踪。设备文件主要包括设备权限和设备类型的有关信息，以及两个可供系统内核识别的唯一的设备号。系统在很多情况下，可能有不正一个同种类型的设备，因此 Linux 可以对所有的设备使用同种驱动程序，但是操作系统又必须能够区分每一个设备。

那么 Linux 又是通过什么样的方法来区分这些同种类型设备呢？实际上 Linux 是使用设备号来区分的。每一个设备都有一个主设备号和子设备号。主设备号用来确定使用什么样的驱动程序，子设备号是硬件驱动程序用来区分不同的设备和判断如何进行处理。例如，6 个终端都使用相同的设备驱动程序，那么它们的主设备号都是一样的，但是每一个终端都有一个不同的子设备号，可使操作系统唯一的确定它们。

【实例 5.1】

从下面的清单可以看出，所有终端设备的驱动程序都有相同的主设备号 4，但子设备号是从 0~5。

crw-rw-rw-	1 root	tty	4, 1	4OA 11	2001	tty0
crw-rw-rw-	1 root	tty	4, 2	4OA 11	2001	tty1
crw-rw-rw-	1 root	tty	4, 3	4OA 11	2001	tty2
crw-rw-rw-	1 root	tty	4, 4	4OA 11	2001	tty3
crw-rw-rw-	1 root	tty	4, 5	4OA 11	2001	tty4
crw-rw-rw-	1 root	tty	4, 6	4OA 11	2001	tty5

【说明】在目录清单中，都是主设备号在前，子设备号在后，这种赋值方法确保了它们的惟一性。如果两个设备有完全一样的主、子设备号，Linux 将不能与它们正常通信。

Linux 习惯于把所有的设备文件都置于/dev 目录下，其中很大一部分只是带有不同设备号的基本驱动程序的拷贝，但是每一个文件都是相互独立的。

Linux 下的驱动程序的命名与其他操作系统下的命名不同，常见的设备名称与驱动程序的对应关系如表 5-1 所示。

表 5-1 Linux 下常见设备及对应的驱动程序命名

设 备	命 名
第一软驱(A:)	/dev/fd0
第二软驱(B:)	/dev/fd1
IDE1 的第一个硬盘(master)	/dev/hda
IDE1 的第二个硬盘(slave)	/dev/hdb
IDE2 的第一个硬盘(master)	/dev/hdc
SCSI 的第一个硬盘	/dev/sda
SCSI 的第二个硬盘	/dev/sdb
光驱 CD-ROM	/dev/cdrom
打印机	/dev/lp0

5.1.2 设备分类

计算机上凡是与 Linux 进行通信的每个硬件都被视为一个设备，它们可以分为两种类型：块设备和字符设备。

终端、打印机和异步调制解调器都属于字符设备，它们的通信方式是使用字符，一次只发送一个并回送一个字符。相反，硬盘驱动器和磁带机则使用块数据通信，这对发送大量信息无疑是一种极为快捷的方法，这样的设备称为块设备。通常，块设备用于对大批量数据的处理，而字符设备传输数据则比较缓慢。例如，大多数模拟调制解调器是字符设备，而 ISDN 则属于块设备。在相同的时间里，块设备可以比字符设备传输更多的数据。

有些设备在不同的情况下可分别为字符设备和块设备，例如，一些磁带机就属于这种情况，也就是说这样的主设备有两套设备驱动程序，用户可针对不同的读写要求来选择设备驱动程序。对于大量、快速的数据传送，最好选用块设备；对于某个文件检索或单一目录的备份，字符设备则更为适合。另一种区分块设备和字符设备的方法是看设备如何处理缓冲，字符设备是靠自己实现缓冲，块设备通常以 512 字节或 1024 字节(甚至更大)的组块进行通信，它们通过系统内核实现缓冲。对用户来说，这种缓冲则更易察觉。

设备驱动程序和设备文件很详细地标明了设备是字符设备还是块设备。要识别一个设备的类型，只需要查看一下设备文件中的权限位就可以了。如果权限位中的第一个字符是 b，则该设备就是块设备；若是 c，则说明它是字符设备。如图 5-1 所示的是我们从/dev 目录清单中摘录的一段，用户可以由权限位的第一个字符来判断设备是何种类型。

crw-----	1 root	root	14,	20	4OA 11	2001	audio1
crw-----	1 root	root	14,	7	4OA 11	2001	audioc1
brw-rw----	1 root	disk	29,	0	4OA 11	2001	aztcd
brw-rw----	1 root	disk	41,	0	4OA 11	2001	bpcd

图 5-1 /dev 目录清单

5.2 使用设备

本节我们将介绍一些常见设备的使用。

5.2.1 磁盘

计算机有不同种类的磁盘驱动器，常见的有软盘、IDE 硬盘和 SCSI 硬盘等，下面我们分别来介绍如何使用这些设备。

1. 软盘

软盘是可移动的低容量的存储介质。作为存储设备，它比硬盘要慢得多，但它具有可移动和便于传输数据的优点。相应的软盘的块设备都以字母“fd”开始，/dev/fd0 是第一个，其他软盘的编号逐步增大。对于软盘，有许多可能的格式，内核需要知道磁盘的格式才能够正确地读取它。目前计算机使用的软盘基本都是 1.44 MB 的。

使用软盘的步骤如下：

- (1) 以超级用户身份登录；
- (2) 创建一个安装点(如/mnt/floppy)来加载软盘；
- (3) 放入软盘；
- (4) 执行如下命令来加载软驱：

```
[root@redflag /root]#mount -t vfat /dev/fd0 /mnt/floppy
```

成功安装后，软盘的文件出现在/mnt/floppy 目录下，这些文件对所有的用户可读，但只有 root 才可以修改、删除这些文件。

卸载软盘的命令如下：

```
[root@redflag /root]#umount /mnt/floppy
```

【注意】在卸载软盘时，当前工作目录不能是/mnt/floppy 或其子目录。

2. 硬盘

硬盘一般是比较大的存储设备，这使得它能够在其不同位置存放不同的文件系统。加载硬盘的步骤和软盘基本相同，通过加载，我们可以很容易地使用 Windows 98 或 Windows 2000 下的文件，假设安装点为/mnt/windows，对于 IDE 硬盘执行的命令如下：

```
[root@redflag /root]#mount -t vfat /dev/hda5 /mnt/windows
```

对于 SCSI 硬盘，执行的命令如下：

```
[root@redflag /root]#mount -t vfat /dev/sda4 /mnt/windows
```

使用“-t vfat”选项，是因为 Windows 下文件系统是 FAT 32 格式的。

【说明】我们可以通过修改/etc/fstab 文件，使得系统每次启动时自动加载。/etc/fstab 文件的内容如下：

/dev/hda7	/	reiserfs	defaults,notail	1	1
/dev/hda5	/mnt/windows	vfat	defaults	0	0
/dev/cdrom	/mnt/cdrom	iso9660	noauto,owner,ro	0	0
/dev/hda6	swap	swap	defaults	0	0

/dev/fd0	/mnt/floppy	vfat	noauto,owner	0	0
none	/proc	proc	defaults	0	0
none	/dev/pts	devpts	gid=5,mode=620	0	0

5.2.2 CD-ROM

CD-ROM 驱动器从根本上讲是只读设备，它与其他块设备的安装方式相同。CD-ROM 一般包含标准的 ISO9660 文件系统和一些可选的扩充。现在的光驱基本上都符合 ATAPI 标准。

使用 mount 命令可以把光盘中的所有目录和文件加载到 Linux 目录中，以 root 身份执行如下的命令：

```
[root@redflag /root]#mount -t iso9660 /dev/cdrom /mnt/cdrom
```

如果命令生效，光盘中的内容将出现在目录/mnt/cdrom 下。

上述命令执行后，若不能加载成功，可能的原因如下：

- (1) /mnt/cdrom 不存在。
- (2) /dev/cdrom 不存在。
- (3) 当前目录是安装点。

卸载光盘的命令如下：

```
[root@redflag /root]#umount /dev/cdrom
```

如果系统提示“设备已经安装或目录忙”的信息，可能是由于用户的当前目录是在安装点/mnt/cdrom 或子目录而造成的，此时必须切换到其他目录下才能进行。

【注意】 如果 CD-ROM 没有能够成功卸载，光盘就无法被取出。

5.2.3 打印机

当我们想要把自己设计好的文件在 Linux 下打印时，首先要正确配置打印机。在红旗 Linux 中，配置打印机的命令是 printtool，此命令只能在 X Window 下使用。下面我们通过实例来讲述如何安装和配置打印机。

【实例 5.2】

本实例安装本地打印机，打印机的型号是 HP LaserJet 4，配置过程如下：

(1) 运行 printtool 工具，其主界面如图 5-2 所示。利用 printtool，用户可以添加、编辑和删除打印机队列。

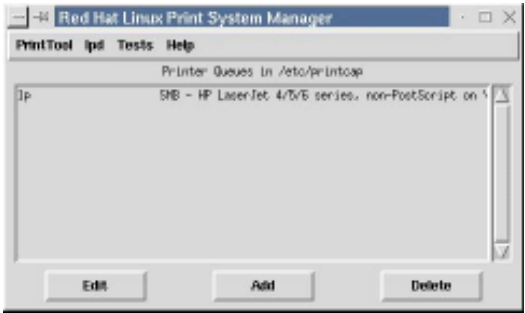


图 5-2 PrintTool 主界面

(2) 选择【Add】来添加打印机，然后选择要增加的打印机的类型。Linux 下的打印机类型如图 5-3 所示，有四种类型。

Local Printer：指连接于本地系统并口或串口上打印机的打印队列。

Remote Unix (lpd) Queue：指可以通过 TCP/IP 网络来访问的连接于非本地系统的打印队列。

SMB/Windows 95/NT Printer：指可以通过 SMB 网络来访问的连接于非本地系统的打印队列。

NetWare Printer(NCP)：指可以通过 NetWare 网络来访问的连接于非本地系统的打印队列，在建立此打印队列之前，必须安装 NCP 文件系统。

(3) 选择配置本地打印机，即“Local Printer”，进入到具体的配置界面，如图 5-4 所示。

其中：

Names：是打印机名称，可以指定多个名字，之间用“|”(管道)符号隔开。

Spool Directory：指定存放要打印的文件目录，不要让多个打印队列共享一个假脱机目录。

File Limit in Kb：所接受的最大作业量，以 Kb 为单位。0 表示不进行限制。

Printer Device：打印设备名称。

Input Filter：过滤器用来将打印文件转换成打印机可以处理的格式。通过 Select 来选择最适合你的打印机的过滤器，如图 5-5 所示。

从图 5-5 可以看出，左边为打印机的具体类型，右边为打印机的具体选项，在这里我们选择“HP LaserJet 4/5/6 series, non-PostScript”类型的打印机，然后再进行右边的具体配置，最后按“OK”来确定。

Suppress Headers：可以选择是否每个打印作业前打印一张起始信息页。



图 5-3 Linux 下的打印机类型



图 5-4 编辑本地打印机

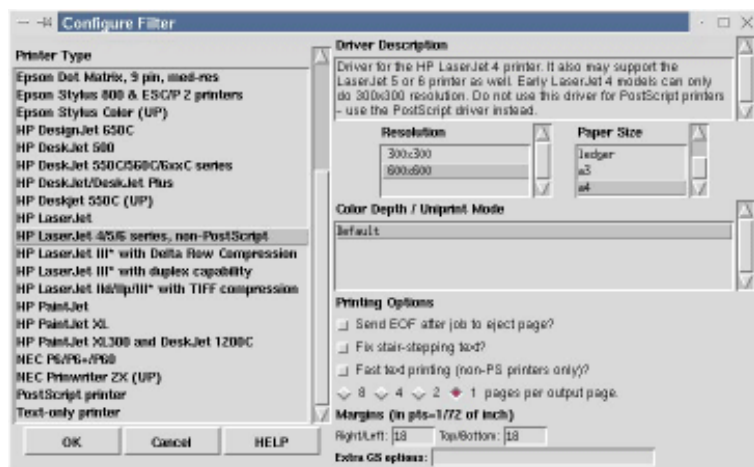


图 5-5 配置打印机

(4) 上述步骤执行完成后，我们就完成了本地打印机的配置，如图 5-6 所示。

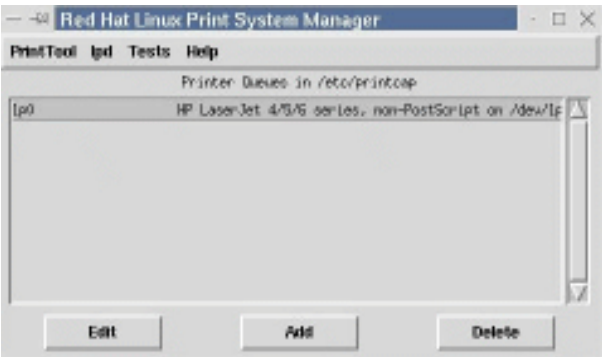


图 5-6 配置好的打印机

添加完打印机队列后，用户还需要重新启动打印机守护进程 lpd，然后就可以进行打印测试了。

5.2.4 显卡

显卡是 Linux 操作系统支持最差的，其配置正确与否直接影响到用户能否启动 X Window。配置显卡最常用的工具是提供了图形化界面的 Xconfigurator。

【实例 5.3】

本实例使用 Xconfigurator 来配置显卡，步骤如下：

(1) 键入“Xconfigurator”命令来启动配置，欢迎画面如图 5-7 所示。选择“OK”进入下一步。

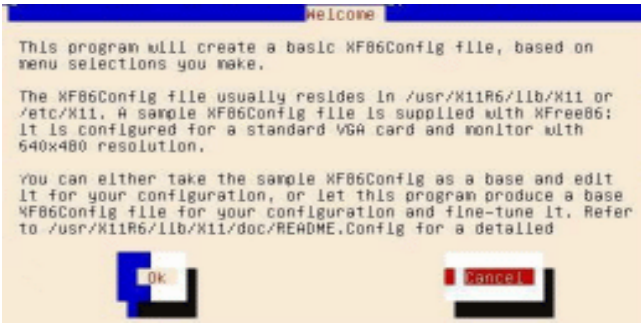


图 5-7 Xconfigurator 欢迎画面

(2) PCI 显卡检测，结果如图 5-8 所示。



图 5-8 PCI 显卡检测结果

(3) 选择显示器类型，如图 5-9 所示。我们选择“Acer 57c”。



图 5-9 选择显示器

(4) 选择显存大小，如图 5-10 所示。我们选择“16 mb”。

(5) 选择时钟芯片，如图 5-11 所示，我们一般选择第一项。

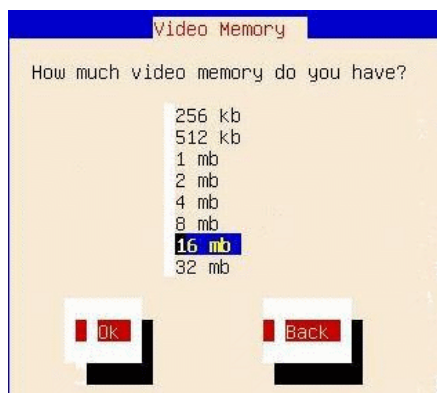


图 5-10 选择显存

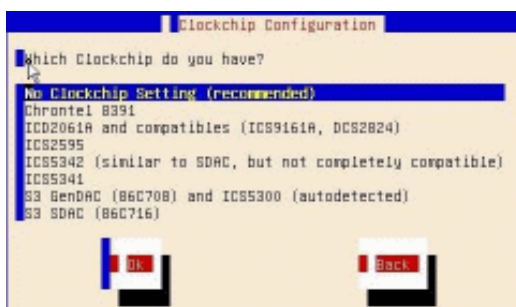


图 5-11 选择时钟芯片

(6) 设置显示器分辨率，如图 5-12 所示，我们设为 24 位“1024 × 768”。

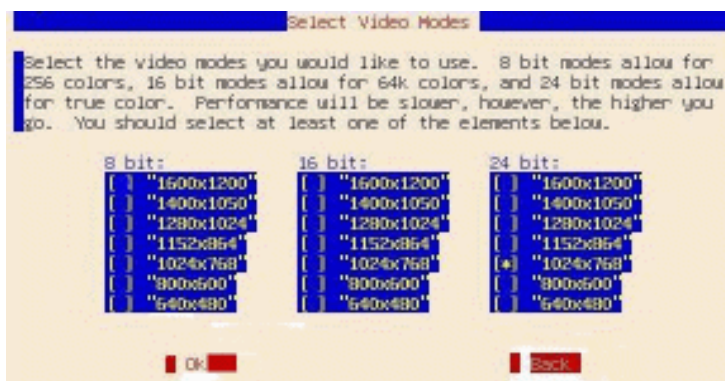


图 5-12 设置分辨率

(7) 测试。完成上述步骤后，系统会进行测试，来确定显卡能否正常工作。如果测试通过，我们就可以通过 start x 命令进入 X Window 环境，如图 5-13 所示。



图 5-13 X Window 启动画面

如果测试未通过，我们就需要下载驱动程序，安装后才能使显卡正常工作。下面我们以实例来讲解在 Linux 下如何安装显卡驱动程序。

【实例 5.4】

某机的显卡是 Trident Blade 3D 的，安装显卡驱动程序步骤如下：

(1) 下载驱动程序。下载驱动程序首先要找到显卡 for Linux 的驱动程序。现在绝大多数的 3D 显卡都已有了 for Linux 的驱动程序，可到各显卡厂商的主页或 Linux 的相关站点上去寻找。现在找到的驱动程序名为“XF86_SVGA”，文件格式是“.tar.gz”，是 Linux 的压缩文件。

(2) 文件解压。由于下载的是压缩文件，就需要进行解压缩，可以在 Linux 下用 tar 命令解压。

(3) 拷贝文件。将“XF86_SVGA”文件拷贝到/usr/X11R6/bin 目录下。注意，这是针对 RedFlag 版本来说，其他版本的路径不一定相同。执行如下的命令：

```
cp /root/XF86_SVGA/XF86_SVGA /usr/X11R6/bin
```

大家可根据自己的情况灵活掌握，关键是路径一定要正确，还要分清字母的大小写。如果系统提示有同名文件，问是否覆盖，一定要选择“y”。这些旧文件可能是以前安装显卡时加载的，对我们没有实际用处。

(4) 配置显卡。文件拷贝完成后，输入 Xconfigurator，启动显卡配置程序。选择加载的显示模块为 SVGA，再选择显示器型号、显存大小、刷新频率，选定 24 位色、800×600 的分辨率。一切就绪，X 服务器开始检测，十几秒钟后，检测完毕，X 服务器再没有像以往那样给出出错信息。通过 start x 命令进入 X Window 环境。

5.2.5 声卡

声卡是多媒体计算机不可或缺的设备，Linux 要成为普及的操作系统，当然也会支持声

卡。在安装 Linux 系统时会检测声卡，如果检测不到，用户就必须自行设置。在文字模式中执行 `sndconfig` 命令(或执行 `setup` 命令后，选择“Sound card configuration”)开始设置声卡。

【实例 5.5】

本实例介绍如何配置声卡，步骤如下：

(1) 执行 `setup` 命令后，选择“Sound card configuration”，如图 5-14 所示，进入声卡配置。

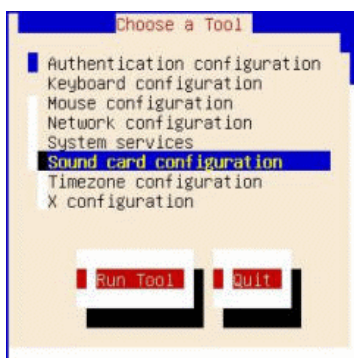


图 5-14 选择配置声卡选项

(2) 接下来系统自动检测声卡，若检测到，会显示如图 5-15 所示的画面。

(3) 接下来系统会播放声音样本进行测试，然后询问我们是否听到声音，如图 5-16 所示。

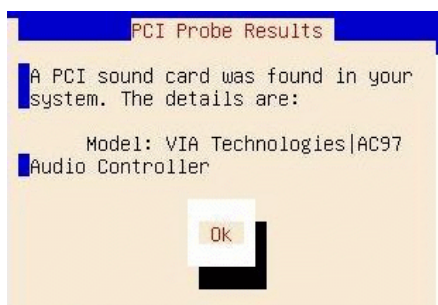


图 5-15 检测到声卡



图 5-16 询问听到声音样本

如果我们能够听到一段声音，表示声卡设置成功，此时单击“`Yes`”按钮完成设置。如果没有听到声音，单击“`No`”按钮，改用自行指定声卡的方式设置，这里就不一一演示了。

5.2.6 Modem

红旗 Linux 提供了非常方便的配置 Modem 的图形化的工具。我们用实例 5.6 来讲述 Modem 的配置过程。

【实例 5.6】

本实例利用红旗 INTERNET 工具来介绍使用调制解调器拨号上网，步骤如下：

(1) 我们可以通过【`K` 按钮】 【系统设置】 【拨号网络配置】来启动红旗 INTERNET 工具，如图 5-17 所示，选择 Modem 项。

(2) 按“下一步”执行 `KPPP` 命令来打开“`KPPP`”对话框，如图 5-18 所示。

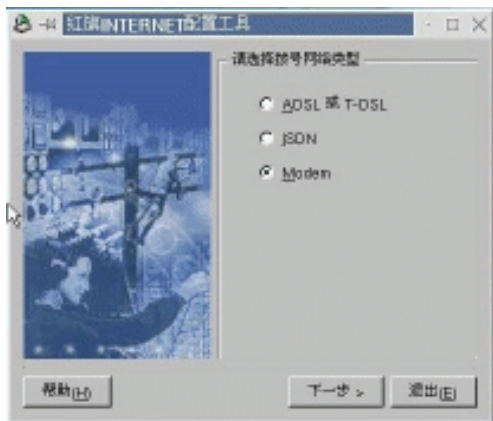


图 5-17 红旗 INTERNET 工具启动画面



图 5-18 KPPP 对话框

- (3) 我们按设置按钮来设置拨号账号，如图 5-19 所示。
- (4) 我们选择新建连接，设置连接名字为“my_dail”，电话号码为“96169”，身份验证选择“PAP”，如图 5-20 所示。

【注意】目前大部分的 ISP 账号拨号时都会自动获取域名服务器的地址。如果你的 ISP 不提供此功能，则自行在 DNS 选项卡中设置。



图 5-19 设置拨号账号



图 5-20 新建拨号连接

- (5) 新建连接设置好的画面如图 5-21 所示。
- (6) 开始拨号，画面如图 5-22 所示。
- 成功拨号上网后，我们就可以在网上冲浪了。
- 【注意】如果你的主机同时安装了网卡，最好先关闭网卡再进行拨号，否则可能会导致拨号失败。



图 5-21 设置好的拨号连接



图 5-22 拨号上网画面

5.2.7 ADSL

现在已经有越来越多的人采用 ADSL 方式上网。在 Linux 下可以使用 PPPoE 这个软件来进行 ADSL 拨号。

【实例 5.7】

本实例介绍计时计制的 ADSL 上网，我的账号是 steven@163.gd，配置步骤如下：

(1) 拨号设置。

```
[root@redflag /root]# adsl-setup
```

```
Welcome to the Roaring Penguin ADSL client setup. First, I will run
some checks on your system to make sure the PPPoE client is installed
properly...
```

#上面一段显示欢迎信息。

```
Looks good! Now, please enter some information:
```

#提示用户要输入一些信息。

```
USER NAME
```

#提示输入用户账号。

```
>>> Enter your PPPoE user name (default bxxxxnxx@sympatico.ca): steven@163.gd
```

```
INTERFACE
```

#提示用户输入网络接口。

```
>>> Enter the Ethernet interface connected to the ADSL modem
```

For Solaris, this is likely to be something like /dev/hme0.

For Linux, it will be ethn, where n is a number.

```
(default eth0):
```

#网络接口通常是 eth0，按回车键即可。

```
Do you want the link to come up on demand, or stay up continuously?
```

If you want it to come up on demand, enter the idle time in seconds after which the link should be dropped. If you want the link to stay up permanently, enter 'no' (two letters, lower-case.)

NOTE: Demand-activated links do not interact well with dynamic IP addresses. You may have some problems with demand-activated links.

>>> Enter the demand value (default no):no

#让用户设置闲置多少秒要断线，若想一直维持连接，则回答“no”。

DNS

#提示用户输入“DNS”。

Please enter the IP address of your ISP's primary DNS server.

If your ISP claims that 'the server will provide DNS addresses', enter 'server' (all lower-case) here.

If you just press enter, I will assume you know what you are doing and not modify your DNS setup.

>>> Enter the DNS information here:

#输入“DNS”，一般不需要填写，拨号上网后，ISP会自动指定。

PASSWORD

#输入两次密码，但是密码不会显示在屏幕上。

>>> Please enter your PPPoE password:

>>> Please re-enter your PPPoE password:

FIREWALLING

#提示用户设置防火墙。

Please choose the firewall rules to use. Note that these rules are very basic. You are strongly encouraged to use a more sophisticated firewall setup; however, these will provide basic security. If you are running any servers on your machine, you must choose 'NONE' and set up firewalling yourself. Otherwise, the firewall rules will deny access to all standard servers like Web, e-mail, ftp, etc. If you are using SSH, the rules will block outgoing SSH connections which allocate a privileged source port.

The firewall choices are:

0 - NONE: This script will not set any firewall rules. You are responsible for ensuring the security of your machine. You are STRONGLY recommended to use some kind of firewall rules.

1 - STANDALONE: Appropriate for a basic stand-alone web-surfing workstation

2 - MASQUERADE: Appropriate for a machine acting as an Internet gateway for a LAN

```
>>> Choose a type of firewall (0-2): 0
#我们选择“0”，表示不设置防火墙。

Start this connection at boot time
#提示用户是否在启动时进行拨号。

Do you want to start this connection at boot time?
Please enter no or yes.

no

** Summary of what you entered **
#查看设置是否正确。

Ethernet Interface: eth0
User name:          steven@gd.net
Activate-on-demand: No
DNS:                Do not adjust
Firewalling:        NONE
#如果设置正确，选择“y”完成 ADSL 设置。

>>> Accept these settings and adjust configuration files (y/n)? y
Adjusting /etc/ppp/pppoe.conf
Adjusting /etc/ppp/pap-secrets and /etc/ppp/chap-secrets
    (But first backing it up to /etc/ppp/pap-secrets-bak)
    (But first backing it up to /etc/ppp/chap-secrets-bak)

Congratulations, it should be all set up!

Type 'adsl-start' to bring up your ADSL link and 'adsl-stop' to bring
it down.  Type 'adsl-status' to see the link status.
```

(2) 拨号上网。当我们要上网时只要执行“adsl-start”命令即可，若要断线则执行“adsl-stop”命令。拨号上线后可执行“adsl-status”命令来查看连接状态。

本章小结

本章主要介绍了硬件设备的基本概念、分类以及常见硬件设备(磁盘、CD-ROM、打印机、显卡、声卡、Modem 和 ADSL)的使用。我们在使用设备时一定要仔细查找资料，勇于大胆尝试。

习 题

1. 假设当前系统中只有一个软驱设备，它在 Linux 下一般表示为_____。
A. /dev/fd0 B. /mnt/floppy C. /dev/floppy D. /dev/fd1

-
2. 一般/dev/lp0 表示_____。
A. 声卡 B. 打印机 C. 显示器 D. 键盘
 3. ADSL 的拨号软件是_____。
A. KPPP B. PPPoE C. SLIP D. PPP
 4. Linux 中设备文件的主要内容包括_____。
A. 设备权限 B. 设备类型 C. 主设备号 D. 子设备号
 5. printtool 可以配置的打印机类型包括_____。
A. local printer B. remote unix(lpd)queue
C. SMB/Windows 95/NT printer D. NetWare Printer
 6. Linux 系统中的设备的类型包括_____。
A. 块设备 B. 字符设备 C. 流设备 D. 缓冲设备
 7. 执行 mount -t auto /dev/cdrom /mnt/cdrom 命令之后, Linux 报告出错信息, 则可能的原因是_____。
A. /mnt/cdrom 不存在 B. /mnt/cdrom 为空
C. /dev/cdrom 设备不存在 D. 当前目录是安装点/dev/cdrom

第 6 章 文件系统管理

Linux 文件系统是 Linux 和 Windows 的一个重要差别。首先，Linux 有自己独特的文件系统格式，支持的类型也比 Windows 多；其次，Linux 文件系统的组织方式也和 Windows 的不同，Linux 没有什么 C 盘、D 盘，所有的在不同分区的数据构成惟一的一个目录树，理解这一点是至关重要的；再者，在 Linux 中可以很容易地根据需要决定是否挂接某个分区。

6.1 文件系统基础

6.1.1 磁盘的分区

Linux 系统使用各种存储介质来保存永久的数据，例如：硬盘、软盘、光盘、磁带等。其中硬盘是不可缺少的介质，硬盘有容量大、速度快、价格低的特点。我们常常对硬盘进行分区，使得每个分区在逻辑上是独立的。这样我们就可以在每个分区安装一个操作系统，而多个操作系统就可以共处在同一个硬盘上。软盘的容量小，不进行分区；光盘则作为一个大盘更易于使用，也不进行分区。

硬盘分区的信息保存在硬盘的第一个扇区(即第一面第一磁道第一扇区)，这个扇区称为 MBR(主引导记录)，主引导记录包含有一段小程序。计算机启动时 BIOS 会执行这一段小程序，小程序又会读入分区表，检查哪个分区是活动分区(也叫启动分区)，并读入活动分区的第一扇区(称为分区的启动扇区)。启动扇区也包含另一个程序，这个程序实际上是操作系统的一部分。它将负责操作系统的启动。一个硬盘的分区最多只能有四个基本分区。有些时候这个数量太少了，于是人们就发明了扩展分区。扩展分区是在基本分区的基础上把分区再细分成多个子分区，每个子分区都是逻辑分区。一般情况下，只能允许存在一个扩展分区，即磁盘可以有三个基本分区和一个扩展分区。硬盘的分区结构如图 6-1 所示，硬盘的分区信息可以使用命令“fdisk -l”获得。

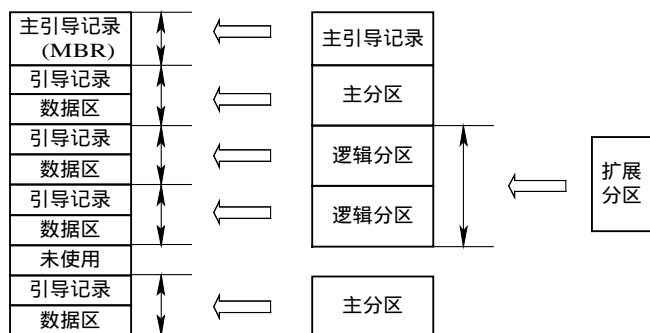


图 6-1 硬盘分区结构

【实例 6.1】

```
[root @redflag /root]#fdisk -l /dev/hda
```

```
Disk /dev/hda: 255 heads, 63 sectors, 2482 cylinders
```

```
Units = cylinders of 16065 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	383	3076416	83	Linux
/dev/hda2		384	447	514080	82	Linux swap
/dev/hda3		448	454	56227+	83	Linux
/dev/hda4		455	467	104422+	82	Linux swap

以上输出中带“*”号的是启动分区。我们随后将详细介绍 fdisk 命令的使用。

Linux 对硬盘分区的命名和 DOS 对硬盘分区的命名有很大的不同。在 DOS 下软盘为“A:”、“B:”，而硬盘为“C:”、“D:”，等等。Linux 则使用/dev/hda0 等来命名它们。以/dev/hd开头的表示 IDE 接口的硬盘，以/dev/sd开头的表示 SCSI 接口的硬盘，随后的 abcd 等代表第几个硬盘，而数字 1、2、3、4 代表硬盘的第几个分区。例如，/dev/hda1 表示第一个 IDE 硬盘的第一个分区。表 6-1 列举了常用的分区命名方法。

表 6-1 分区的命名

设 备	分区的命名
软盘	/dev/fd0
第一个 IDE 硬盘(整个硬盘)	/dev/hda
第一个 IDE 硬盘第一个分区	/dev/hda1
第一个 IDE 硬盘第二个分区	/dev/hda2
.....
第二个 IDE 硬盘(整个硬盘)	/dev/hdb
第二个 IDE 硬盘第一个分区	/dev/hdb1
第二个 IDE 硬盘第二个分区	/dev/hdb2
.....
第一个 SCSI 硬盘(整个硬盘)	/dev/sda
第一个 SCSI 硬盘第一个分区	/dev/sda1
第一个 SCSI 硬盘第二个分区	/dev/sda2
.....
第二个 SCSI 硬盘(整个硬盘)	/dev/sdb
第二个 SCSI 硬盘第一个分区	/dev/sdb1
第二个 SCSI 硬盘第二个分区	/dev/sdb2

6.1.2 什么是文件系统

文件系统是操作系统用于明确磁盘或分区上的文件的方法和数据结构，即在磁盘上组织文件的方法。分区或磁盘在作为文件系统使用前需要初始化，并将记录数据结构写到磁

盘上，这个过程叫建立文件系统。我们在 DOS 下常常进行的格式化磁盘进程也是一个建立文件系统的过程。不同的操作系统所支持的文件系统是不同的，一个文件系统在一个操作系统下可以正常地被使用，转移到另一操作系统时往往会出问题。

Linux 支持多种类型的文件系统。而红旗 Linux 3.0 版又增加了几种新类型的文件系统。下面是几个重要的文件系统：

minix：最早的 Minix 系统的文件系统。

ext2：Linux 系统的文件系统，目前是使用最广泛的文件系统。

swap：用于交换分区和交换文件的文件系统。

sysv：Unix 里广泛使用的 SystemV。

iso9660：标准的 CD-ROM 的文件系统。

vfat：扩展的 DOS 文件系统，支持长文件名，被 Windows 98 采用。

msdos：与 MS-DOS/FAT 16 兼容的文件系统。

hpfs：OS/2 文件系统。

nfs：网络文件系统，允许多台计算机间共享的文件系统。

ntfs：用于 Windows NT 和 Windows 2000 的文件系统。

reiserfs：安全性和效能比 ext2 都好的文件系统。

ext3：ext2 的后续者，红旗 Linux 3.0 版(Linux 内核 2.4.17)已经把它加入。

smb：支持 SMB 协议的高性能文件系统。

一般情况下没有理由用不同类型的文件系统来组成 Linux 系统，除非原有的文件系统已经存在。由于历史的原因，当前 ext2 的使用最为广泛，而 reiserfs 是红旗 Linux 3.0 安装时默认的文件系统类型，ext3 是当前最新的文件系统类型。ext2 比起以往的文件系统在文件性能方面有很大的提高，但也存在不少的问题，例如，文件系统在异常关机等情况下容易遭到破坏，使用“fsck”命令检查文件系统要检查整个文件系统，对于较大的文件系统，常常需花费几个小时的检查时间。为了解决这个问题，出现日志型文件系统，如 reiserfs 和 ext3。日志型文件系统比传统的文件系统安全，因为它用独立的日志文件跟踪磁盘内容的变化。就像关系型数据库(RDBMS)，日志文件系统可以用事务处理的方式提交或撤消文件系统的变化。日志机制保证了在每个实际数据修改之前，相应的日志已经写入硬盘。正因为如此，在系统突然崩溃时，在下次启动几秒钟后就能恢复成一个完整的系统，系统也就能很快地使用了。reiserfs 除了具有日志型文件系统的特性，还具有适合处理大量小文件(如 5000 个 50 字节的文件)和特大文件(例如 2 GB 以上)的特点。ext3 则是 ext2 的后续者，它也是日志型文件系统，更为难得的是，它的磁盘格式和 ext2 的相同，ext2 和 ext3 的转换相当的容易，对于 ext2 的升级十分有利。没有一种文件系统适合所有的应用，因此我们要选择适合自己的文件系统。对于新安装的系统，我们建议采用 reiserfs 或 ext3。

Linux 采用虚拟文件系统(VFS)技术，因此 Linux 可以支持多种文件。每一文件系统都提供一个公共的接口给 VFS，不同文件系统的所有细节由软件进行转换。而从 Linux 内核和 Linux 运行的程序来看，不同的文件系统之间没有差别。例如，用户 can 把自己原有的 Windows 分区挂接到 Linux 中的一个目录下，也可以同时把 NFS(网络文件系统)挂接在另一目录，它们可以和平地融合为一体。

文件系统是所有数据的基础，所有文件和目录都驻留在文件系统上。在 Linux 系统中，所有的文件系统都被连接到一个总的目录上，这个目录就叫根目录，是由系统自动建立的。根目录下有许多分支，分支又有子分支，从而整个目录呈树状结构，如图 6-2 所示。

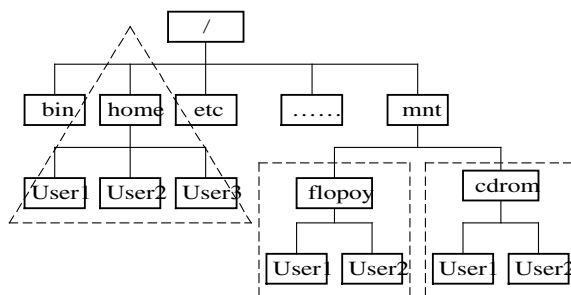


图 6-2 树状的目录结构

文件系统连接目录树上的一点，这个点就叫安装点。图 6-2 中的每个虚框就是一个文件系统，所有不在虚框的部分也是一个文件系统，一共有四个文件系统。就这样，不同的文件系统形成了一个无缝的整体。

6.1.3 文件

文件是有名字的一组相关信息的集合，它有多种的分类方法，如根据文件的用途我们把文件分为以下四种。

(1) 普通文件。文件可以是千差万别的，如普通的 Word 文件、图像文件、声音文件、网页 HTML 文件，也可以是脚本文件、程序员编写的可执行文件。我们可进一步把文件分类为文本文件和二进制文件。文本文件即 ASCII 码文件，可以使用 cat、more 等命令查看，Linux 系统的多种配置文件、源程序、HTML 文件都属于此类。二进制文件，一般不能被直接查看，而必须使用相应的软件才行，如图像文件、声音文件、可执行文件都属于此类。

(2) 目录文件。Linux 中把目录也看成文件，这是和 DOS/Windows 不太相同的地方。目录可以包含下一级目录和普通文件。

(3) 链接文件。我们在第 2 章已经介绍了链接，有软链接和硬链接之分。链接的一个好处是不占用过多的磁盘空间。

(4) 设备文件。Linux 中把系统中的设备也当成文件，用户可以像访问普通文件一样来访问系统中的设备，并且所有设备文件都放在 /dev 目录下。设备文件可以分为块设备和字符设备两类。例如，打印机是字符设备，磁盘是块设备。把设备当成文件的好处是使得 Linux 系统能够保证设备的独立性。计算机外设不断更新，但是操作系统不可能为了刚出现的设备文件而经常修改。当需要增加新的设备时，只需要在内核添加必要的设备驱动程序就可以了。这样一来，使用不同外设时内核就可以用完全一样的方式来进行处理。

设备文件中有一特殊的文件是 /dev/null，称为空设备。它是一个类似“黑洞”的设备，所有放入该设备的东西将不复存在，例如：

```
[root @redflag /root]#mv test.zip /dev/null
```

该命令执行的结果是 test.zip 文件永远被删除了。

还有一种很特殊的文件是管道文件，它主要用于在进程间传递信息，是一个先进先出(FIFO)的缓冲区，管道文件类似我们日常生活中的管道，一端进入的是某个进程的输出，另一端输出的是另一个进程的输入。例如：

```
[root@redflag /root]#cat /etc/passwd | more
```

该命令使用了管道“|”，命令 `cat /etc/passwd` 的输出是管道的输入，经过管道后，成为了命令 `more` 的输入。

使用命令“`ls -l`”可以显示文件的类别，每个输出行中的第一个字符表示的就是文件的类别，例如，“b”代表块设备，“p”代表管道文件，“c”代表字符设备，“d”代表目录文件。

6.1.4 Linux 系统的目录结构

Linux 系统中，目录是一个层次(或树状结构)，根是所有目录的起始点，根目录下主要有以下子目录。

`/bin`：包含二进制文件，即可执行程序，这些程序是系统必需的文件。

`/sbin`：也用于存储二进制文件，但不同的是它们不给普通用户使用，只有超级用户 `root` 可以使用。

`/etc`：用于存放 Linux 系统的配置文件，该目录的文件相当重要，例如：`passwd`、`host`、`fstab`、`inittab` 等等，我们将在不同的章节使用到这个目录下的文件。

`/boot`：Linux 系统引导时加载器使用的文件，系统中非常重要的内核 `vmlinux` 就是放在该目录下。

`/dev`：存放设备文件，用户可以通过这些文件访问外部设备。

`/lib`：存放根文件系统中的程序运行所需要的库文件。

`/temp`：存放各种临时文件。

`/mnt`：管理员临时安装文件系统的安装点，下面有几个意义明确的子目录，如软盘、光驱等。如下：

<code>drwxrwxr-x</code>	<code>2 root</code>	<code>root</code>	<code>48</code>	<code>10 月</code>	<code>9</code>	<code>1998</code>	<code>cdrom</code>
<code>drwxrwxr-x</code>	<code>2 root</code>	<code>root</code>	<code>48</code>	<code>2 月</code>	<code>6</code>	<code>1996</code>	<code>floppy</code>

`/root`：超级用户的个人主目录。

`/usr`：该目录占用的空间一般比较大，用于安装各种应用程序。

`/proc`：是一个虚拟的目录，存放当前内存的映像，该文件系统由内核自动产生。

`/var`：存放一些会随时改变的文件。例如，`spool` 目录、其他的暂存文件。

`/opt`：是放置额外安装的应用程序包的地方。

6.2 创建文件系统

要在硬盘创建文件系统，首先要进行硬盘分区。硬盘分区有很多的工具，如：`Fdisk`、`Cfdisk` 等，用得最多的还是 `Fdisk`。

6.2.1 Fdisk 的使用

1. fdisk -l——显示所有分区的信息

【实例 6.2】

Disk /dev/hda: 255 heads, 63 sectors, 2482 cylinders

Units = cylinders of 16065 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	383	3076416	83	Linux
/dev/hda2		384	447	514080	82	Linux swap
/dev/hda3		448	486	313267+	83	Linux

以上显示了惟一的一个磁盘/dev/hda 的参数和分区情况，磁盘有 255 个磁头，2482 个柱面，每柱面 63 个扇区，第四行起是分区的情况，依次是分区名、是否是启动分区、起始柱面、终止柱面、分区的总块数、分区 ID(分区类型的数字值)、分区的类型。如/dev/hda1 分区是启动分区(带“*”号)，起始柱面是 1，终止柱面是 383，分区大小是 3 076 416 块(每块的大小是 1024 字节，即总共 300 MB 左右的空间)。每柱面的扇区数等于磁头数乘以每柱面扇区数，每两个扇区为一块，因此分区的块数等于分区占用的总柱面数乘以磁头数，再乘以每柱面扇区数后除以二。例如：

/dev/hda2 的总块数=64 × 255 × 63/2=514 080

2. fdisk 驱动器名——创建磁盘分区

【实例 6.3】

[root @redflag /root]#fdisk /dev/hda

The number of cylinders for this disk is set to 2482.

There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:

- 1) software that runs at boot time (e.g., old versions of LILO)
- 2) booting and partitioning software from other OSs

(e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):m

使用“m”可以获得如下帮助：

Command action

- a toggle a bootable flag
- b edit bsd disklabel
- c toggle the dos compatibility flag
- d delete a partition
- l list known partition types
- m print this menu
- n add a new partition
- o create a new empty DOS partition table
- p print the partition table

- q quit without saving changes
- s create a new empty Sun disklabel
- t change a partition's system id
- u change display/entry units
- v verify the partition table
- w write table to disk and exit
- x extra functionality (experts only)

其中常用的命令有：

- a： 切换分区的启动标志。
- d： 删除分区。
- l： 显示已知的分区类型。
- m： 显示命令的帮助。
- n： 添加新的分区。
- p： 显示当前硬盘的分区情况。
- q： 退出并且不保存分区的结果。
- t： 改变分区的类型。
- w： 保存分区的结果并退出。

使用“l”可以获得如下帮助：

Command (m for help): l

0 Empty	1b Hidden Win95 FA	63 GNU HURD or Sys	b7 BSDI fs
1 FAT12	1c Hidden Win95 FA	64 Novell Netware	b8 BSDI swap
2 XENIX root	1e Hidden Win95 FA	65 Novell Netware	c1 DRDOS/sec (FAT-
3 XENIX usr	24 NEC DOS	70 DiskSecure Mult	c4 DRDOS/sec (FAT-
4 FAT16 <32M	39 Plan 9	75 PC/IX	c6 DRDOS/sec (FAT-
5 Extended	3c PartitionMagic	80 Old Minix	c7 Syrinx
6 FAT16	40 Venix 80286	81 Minix / old Lin	da Non-FS data
7 HPFS/NTFS	41 PPC PReP Boot	82 Linux swap	db CP/M / CTOS / .
8 AIX	42 SFS	83 Linux	de Dell Utility
9 AIX bootable	4d QNX4.x	84 OS/2 hidden C:	e1 DOS access
a OS/2 Boot Manag	4e QNX4.x 2nd part	85 Linux extended	e3 DOS R/O
b Win95 FAT32	4f QNX4.x 3rd part	86 NTFS volume set	e4 SpeedStor
c Win95 FAT32 (LB	50 OnTrack DM	87 NTFS volume set	eb BeOS fs
e Win95 FAT16 (LB	51 OnTrack DM6 Aux	8e Linux LVM	ee EFI GPT
f Win95 Ext'd (LB	52 CP/M	93 Amoeba	ef EFI (FAT-12/16/
10 OPUS	53 OnTrack DM6 Aux	94 Amoeba BBT	f1 SpeedStor
11 Hidden FAT12	54 OnTrackDM6	9f BSD/OS	f4 SpeedStor
12 Compaq diagnost	55 EZ-Drive	a0 IBM Thinkpad hi	f2 DOS secondary
14 Hidden FAT16 <3	56 Golden Bow	a5 BSD/386	fd Linux raid auto
16 Hidden FAT16	5c Priam Edisk	a6 OpenBSD	fe LANstep
17 Hidden HPFS/NTF	61 SpeedStor	a7 NeXTSTEP	ff BBT
18 AST Windows swa			

数字是各种的分区类型的 ID，典型的有：5 表示扩展分区、7 表示 NTFS 分区、b 表示 Windows 95 分区、82 表示 Linux 交换分区、83 表示 Linux 分区。

使用 “p” 命令可以显示当前分区的情况：

Command (m for help): p

Disk /dev/hda: 255 heads, 63 sectors, 2482 cylinders

Units = cylinders of 16065 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	383	3076416	83	Linux
/dev/hda2		384	447	514080	82	Linux swap
/dev/hda3		448	486	313267+	83	Linux

使用 “d” 命令可以删除分区：

Command (m for help): d

Partition number (1-4): 3

使用 “n” 命令可以创建分区：

Command (m for help): n

Command action

e extended

p primary partition (1-4)

创建分区时选择 p，则创建的是基本分区，选择 e，则创建的是扩展分区，我们选择 p。

Partition number (1-4): 3

输入分区号 “3”。

First cylinder (448-2482, default 448):

输入分区的起始柱面，不输入时就是缺省值 448。

Using default value 448

Last cylinder or +size or +sizeM or +sizeK (448-2482, default 2482): +50M

输入分区的终止柱面，也可以输入分区的大小，用+50 000 000 表示 50 000 000 字节，+50 M 表示 50 MB，+50 000 K 表示 50 000 KB。

使用 “t” 命令可以改变分区的类型：

Command (m for help): t

Partition number (1-4): 3

Hex code (type L to list codes): 83

Changed system type of partition 3 to 83 (Linux)

红旗 Linux 3.0 缺省创建的文件系统类型是 83(Linux)。

再次使用 “p” 命令显示分区的情况：

Command (m for help): p

Disk /dev/hda: 255 heads, 63 sectors, 2482 cylinders

Units = cylinders of 16065 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	383	3076416	83	Linux

/dev/hda2	384	447	514080	82	Linux swap
/dev/hda3	448	454	56227+	83	Linux

可以看到新的分区/dev/hda3 已经创建好了。

使用“w”命令保存分区结果并退出，如果不想保存结果，可使用“q”命令。

Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.

Re-read table failed with error 16: Device or resource busy.

Reboot your system to ensure the partition table is updated.

WARNING: If you have created or modified any DOS 6.x partitions, please see the fdisk manual page for additional information.

Syncing disks.

硬盘分区后，要重新启动系统。

6.2.2 文件系统的建立

硬盘进行分区后，下一步的工作就是文件系统的建立，这和格式化磁盘类似。在一个分区上建立文件系统会冲掉分区上的所有数据，并且不能恢复，因此建立文件系统前要确认分区上的数据不再使用。建立文件系统的命令是mkfs。mkfs的命令格式如下：

mkfs [参数] 文件系统

参数选项：

-t：指定要创建的文件系统类型，缺省是 ext2。

-c：建立文件系统之前首先要检查坏块。

-l file：从文件 file 中读磁盘坏块列表，该文件一般是由磁盘坏块检查程序产生的。

-V：输出建立文件系统详细信息。

【实例 6.4】

```
[root @redflag /root]#mkfs -V -t ext3 -c /dev/hda3
```

```
mkfs version 2.10s (Aug 30 2001)
```

```
mkfs.ext3 -c /dev/hda3
```

```
mke2fs 1.27 (8-Mar-2002)
```

```
Filesystem label=
```

```
OS type: Linux
```

```
Block size=1024 (log=0)
```

```
Fragment size=1024 (log=0)
```

```
14056 inodes, 56227 blocks
```

```
2811 blocks (5.00%) reserved for the super user
```

```
First data block=1
```

```
7 block groups
```

```
8192 blocks per group, 8192 fragments per group
```

```
2008 inodes per group
Superblock backups stored on blocks:
8193, 24577, 40961
Checking for bad blocks (read-only test): done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 27 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
```

以上命令是在磁盘分区/dev/hda3 建立 ext3 类型的文件系统，建立文件系统时检查磁盘坏块，并且显示详细信息。

对于在软盘建立文件系统稍有不同，我们不必对软盘进行分区，而是先直接格式化：

```
[root @redflag /root]#fdformat -n /dev/fd0
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
```

其次，使用 badblocks 命令检查软盘上的坏块，把坏块信息保存在文件 bad 中：

```
[root @redflag /root]#badblocks /dev/fd0 1440 >bad
```

再用 mkfs 命令建立文件系统：

```
[root @redflag /root]#mkfs -t ext2 -l bad /dev/fd0
mke2fs 1.27 (8-Mar-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
184 inodes, 1440 blocks
72 blocks (5.00%) reserved for the super user
First data block=1
1 block group
8192 blocks per group, 8192 fragments per group
184 inodes per group
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 26 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
```

6.2.3 交换分区

如果在 Linux 运行时物理内存不够，Linux 会把内存的数据先写到磁盘上，当需要数据时再读回到物理内存中，这个过程就叫交换，而用于交换的磁盘空间就叫交换空间。这些技术和 Windows 的虚拟内存技术类似，但 Linux 支持两种形式的交换空间：独立的磁盘交

换分区和交换文件。独立的磁盘交换分区是专门分出一个磁盘分区用于交换，而交换文件则是创建一个文件用于交换。使用交换分区比使用交换文件效率要高，因为独立的交换分区保证了磁盘块的连续，Linux 系统读写数据的速度较快。交换空间的大小一般是物理内存的 1.5 ~ 2 倍。如果内存是 256 MB，则交换空间大小为 500 MB 左右较为合适。

1. 交换分区的建立和激活

交换分区的建立和其他分区的建立没有太大的差别，惟一不同是用 fdisk 命令建立分区时要使用 “t” 命令把分区类型改成 82(Linux Swap)，如下的/dev/hda4是新建的交换分区。

```
[root @redflag /root]# fdisk -l
Disk /dev/hda: 255 heads, 63 sectors, 2482 cylinders
Units = cylinders of 16065 * 512 bytes

   Device   Boot      Start         End      Blocks   Id  System
/dev/hda1   *           1          383       3076416   83   Linux
/dev/hda2             384          447        514080   82   Linux swap
/dev/hda3             448          454        56227+   83   Linux
/dev/hda4             455          467       104422+   82   Linux swap
```

Linux 系统下可以有多个交换分区。创建好交换分区后，要使用 mkswap 命令“格式化”分区。

```
[root @redflag /root]#mkswap -c /dev/hda4
Setting up swapspace version 1, size = 106921984 bytes
```

最后还要激活交换分区：

```
[root @redflag /root]#swapon /dev/hda4
```

以上命令是手工激活的方法，但交换分区通常是在系统启动时就自动激活，自动激活可以在/etc/fstab 文件中配置(见下节介绍)。

关闭交换分区的命令为：

```
[root @redflag /root]#swapoff /dev/hda4
```

2. 交换文件的建立和激活

交换文件一般用在临时增加交换空间的情形。对交换文件的要求是不能有空洞，即文件要占据一片连续的物理空间。建立交换文件和激活的过程如下：

(1) 创建一个指定大小的文件。如下：

```
[root @redflag /root]#dd if=/dev/zero of=/swap bs=1024 count=50000
50000+0 records in
50000+0 records out
```

以上命令在根目录下创建了一个 25 MB(等于 50 000 块)的交换文件 swap。/dev/zero 是一个特殊的设备文件，对它的读操作总是返回零值字节。

(2) 创建交换文件并修改文件的权限为 600。如下：

```
[root @redflag /root]#mkswap /swap
Setting up swapspace version 1, size = 25595904 bytes
[root @redflag /root]#chmod 600 /swap
```

(3) 激活交换文件。如下：

```
[root @redflag /root]#swapon /swap
```

同样关闭交换文件的使用，可以使用命令：

```
[root @redflag /root]#swapoff /swap
```

交换文件关闭后，如果不再继续使用，可以删除。

6.3 文件系统的安装和卸载

Linux 文件系统的组织方式和 DOS、Windows 文件系统的组织方式有很大的差别。Windows 把磁盘分区后用不同驱动器名字来命名，如“C：”、“D：”、“E：”等等，我们把它当成逻辑独立的硬盘来使用，每个逻辑盘有自己的根目录。而 Linux 系统只有一个总的根目录，或者说只有一个目录树，不同磁盘的不同分区都只是这个目录树的一部分。在 Linux 中创建文件后，用户还不能直接使用它，要把文件系统安装(mount)后才能使用。安装文件系统首先要选择一个安装点(mount point)。所谓的安装点就是要安装的文件系统的根目录所在的目录。如图 6-3 和图 6-4 所示，安装后/home 就是第二个文件系统的安装点，因为第二个文件系统的根目录在这一目录下。这样整个文件是由多个文件系统构成的，但对于用户来说整个文件系统却是无缝的，感觉不到是在不同的文件系统工作。文件系统的安装点不同，目录树的结构也会发生变化，如图 6-5 所示。

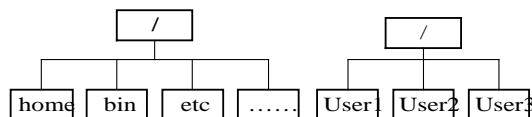


图 6-3 未安装的两个独立的文件系统

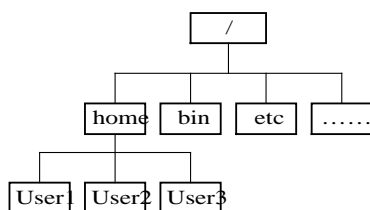


图 6-4 安装后的文件系统

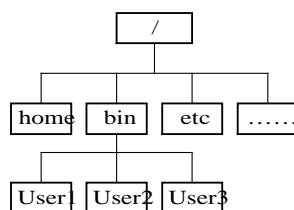


图 6-5 文件系统安装点不同引起目录树结构的不同

6.3.1 手工安装和卸载文件系统

手工安装文件系统常常用于临时使用文件系统的场合，尤其是软盘和光盘的使用。手工安装文件系统使用mount命令，其格式如下：

```
mount [参数] 设备名 安装点
```

参数选项：

-a：安装/etc/fstab 中的所有设备。

-f：不执行真正的安装，只是显示安装过程中的信息。

-n：不在/etc/mtab 登记此安装。

-r：用户对被安装的文件系统只有读权限。

-w：用户对被安装的文件系统有写权限。

-t type：指定被安装的文件系统的类型，常用的有：minix、ext、ext2、ext3、msdos、hpfs、nfs、iso9660、vfat、reiserfs、umdos、smbfs。

-o：指定安装文件系统的安装选项。

例如：

```
[root @redflag /root]#mount -t ext3 /dev/hda3 /mnt/disk1
```

该命令将/dev/hda3 分区的文件系统安装在/mnt/disk1 目录下，文件系统的类型是 ext3，安装点是/mnt/disk1。

安装文件系统时，用户不能处在安装点(即当前目录是安装点)，否则安装文件系统后，用户看到的内容仍是没有安装文件系统前安装点目录原来的内容。安装文件系统后，安装点原有的内容会不可见。卸载文件系统后，安装点原有的内容又会可见。

使用 mount 命令时，mount 会试着测试文件系统的类型，因此常常可以不指明文件系统的类型，但 mount 并非总能成功检测出文件系统的类型，例如：

```
[root @redflag /root]#mount /dev/hda3 /mnt/disk1
```

安装文件系统只能由超级用户 root 来进行，一般用户不能执行此项操作。对于软盘这样的设备，如果只允许超级用户来使用的话不是很合理，这时可以在/etc/fstab 加入相应的行来控制。我们随后介绍/etc/fstab 的用法。

如果不打算在一个文件系统上写入任何数据的话，可以使用-r 开关。这将停止任何对此文件的写要求，也将停止对 i 节点的文件存取时间的更改。

```
[root @redflag /root]#mount -r -t ntfs /dev/hdb1 /mnt/disk2
```

以上命令把/dev/hdb1(Windows 2000)安装在/mnt/disk2 目录下，只是为了在 Linux 中能够读取 Windows 2000 系统下的数据，但不允许改动，以免影响 Windows 2000 的正常工作。

Linux 系统会把已经安装的文件系统信息写到/etc/mtab 文件中，用不带任何参数的 mount 命令也可以显示已经安装的文件系统的信息。

【实例 6.5】

```
[root @redflag /root]#mount
/dev/hda1 on / type reiserfs (rw,notail)
none on /proc type proc (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
```

以上是当前已经安装文件系统的信息。

要卸载文件系统则相当简单，使用以下命令：

umount 安装点或 umount 设备名

```
[root @redflag /root]#umount /mnt/disk1
```

卸载文件系统时，不能有用户正在使用文件系统，例如，用户当前目录是/mnt/disk1，则以上命令会失败。要先切换到别的目录，再执行 umount，例如：

```
[root @redflag /root]#cd
```

```
[root @redflag /root]#umount /mnt/disk1
```

【注意】对于软盘要特别注意,如果一个软盘已经被 mount,要先用 umount 命令卸载文件系统,以保证所有的磁盘缓冲数据全部写到软盘,然后才能把软盘从驱动器取出,否则可能导致软盘上的数据永久性的损坏。

6.3.2 文件系统的自动安装

我们可以使用 mount 命令手工安装文件系统,对于用户经常使用的文件系统则最好能让 Linux 系统在启动时就自动安装好。/etc/fstab 文件就是为了解决这个问题的,其格式如下:

文件系统 安装点 文件系统类型 安装选项 备份频率 检查顺序

【实例 6.6】

```
[root @redflag /root]#cat /etc/fstab
```

/dev/hda1	/	reiserfs	defaults,noail	1 1
/dev/cdrom	/mnt/cdrom	iso9660	noauto,owner,ro	0 0
/dev/hda2	swap	swap	defaults	0 0
/dev/fd0	/mnt/floppy	vfat	noauto,owner	0 0
none	/proc	proc	defaults	0 0
none	/dev/pts	devpts	gid=5,mode=620	0 0

例 6.6 中第二行文件系统/dev/cdrom 安装在/mnt/cdrom 目录下,文件系统类型是 iso9660,安装选项是“noauto,owner,ro”,不使用 dump 命令进行备份,系统安装文件系统时不进行检查。文件系统安装选项可以有多个,选项之间用逗号隔开,常用选项有:

defaults: 缺省值,等于 rw,suid,dev,exec,auto,nouser,async。

noauto: 系统启动时不自动加载该文件系统。

ro: 该文件系统只能读。

rw: 该文件系统可以读写。

user: 允许普通用户安装该文件系统。

noexec: 不允许在该文件系统运行程序。

检查顺序是系统自动安装文件系统时用 fsck 命令检查文件系统时的顺序,为“0”表示不进行检查,为“1”表示第 1 个检查,为“2”表示第 2 个检查,依次类推。在以下例子中:

```
/dev/fd0 /mnt/floppy ext2 user,noauto 0 0
```

软盘的安装选项是“user,noauto”,表示系统不自动安装,普通用户可以安装。

6.4 文件系统的维护

6.4.1 检查文件系统

Linux 是一个稳定的操作系统,一般情况下文件系统并不会出现什么问题。如果系统异常断电或不遵守正确的关机步骤,磁盘缓冲的数据没有写入磁盘,文件系统常常会不正常,这时就需要进行文件系统的检查。Linux 系统启动时,会自动检查/etc/fstab 文件中设定要自动检查的文件系统,就像 Windows 系统开机时用 scandisk 检查磁盘一样。我们也可以使用 fsck 命令手工对文件系统进行检查,fsck 命令的格式如下:

fsck [参数] 设备名

参数选项：

-t fstype：指定文件系统类型。

-A：检查/etc/fstab中的所有文件系统。

-V：显示fsck执行时的信息。

-N：只是显示fsck每一步的工作，而不进行实际操作。

-R：和-A同时使用时，跳过根文件系统。

-P：和-A同时使用时，不跳过根文件系统(要注意使用)。

-n：检查文件系统时，对要求回答的所有问题，全部回答“no”。

-y：检查文件系统时，对要求回答的所有问题，全部回答“yes”。

-p：检查文件系统时，不需要确认就执行所有的修复。

fsck 检查结束后，会给出如下错误代码(fsck 实际的返回值可能是以上代码值的和，表示出现多个错误)：0——没有发现错误；1——文件系统错误已经更正；2——系统需要重新启动；4——文件系统错误没有更正；8——操作错误；16——语法错误；128——共享库错误。

手工检查文件系统时应在没有安装的文件系统上进行，如果文件系统已经安装，应先把它卸载。fsck命令检查完文件系统后，如果修复了文件系统，应该重新启动Linux系统。通常fsck检查完文件系统会将没有引用的项直接连接到文件系统中的/lost+found这样的特定目录下，用户可以从这里找回丢失的数据，但这不是一件容易的事。

【实例 6.7】

```
[root @redflag /root]#umount /dev/hda3
[root @redflag /root]#fsck -V -t ext3 /dev/hda3
fsck 1.27 (8-Mar-2002)
[/sbin/fsck.ext3 (1) -- /dev/hda3] fsck.ext3 /dev/hda3
e2fsck 1.27 (8-Mar-2002)
/dev/hda3: clean, 11/14056 files, 5907/56227 blocks
```

以上命令检查了/dev/hda3 上的文件系统。

检查文件系统所需的时间与文件系统的大小和类型有很大关系。ext2 类型文件系统在检查时要检查整个文件系统，往往要几分钟到几十分钟时间。而 reiserfs 和 ext3 类型文件系统所需的时间要短得多。

6.4.2 磁盘坏块的检查

在磁盘进行分区之后，创建文件系统之前，最好能够使用 badblocks 命令检查磁盘上的坏块。创建文件系统时可以利用坏块检查结果来跳过坏块，避免数据保存到磁盘坏块上。坏块检查命令 badblocks 的格式如下：

badblocks [参数] 设备名 块数

参数选项：

-o filename：将坏块情况输出到文件 filename。

-s：显示已经检查过的磁盘块数。

-w：使用写模式。

【注意】“-w”参数是向被测试磁盘的每一块写入数据，然后读出数据比较，但这意味着会破坏原来的数据，因此要谨慎使用。

例如：

```
[root @redflag /root]#badblocks -o bad /dev/fd0 1440
```

以上命令检查软盘，软盘的块数是 1440(即 1440 KB)，坏块结果输出到文件 bad 中。

```
[root @redflag /root]#badblocks -s /dev/hda3
```

```
Checking for bad blocks (read-only test): done
```

以上命令检查/dev/hda3 的坏块情况，并显示检查的进度。

6.4.3 其他常用的文件系统管理命令

1. du [参数] 目录名——统计目录使用磁盘空间的情况

参数选项：

-a：显示所有文件的统计数，而不仅仅是目录的统计数。

-s：只显示磁盘的总体使用情况。

-b：以字节为单位显示信息，缺省是块(1024 字节)。

例如：

```
[root @redflag /root]#du -b /root
```

以上命令统计/root 下各个子目录的大小(以字节显示)。

2. df——统计未使用磁盘空间

例如：

```
[root @redflag /root]#df
```

Filesystem	1k-blocks	Used	Available	Use%	Mounted on
/dev/hda1	3076316	1003628	2072688	33%	/
/dev/hda3	54447	4127	47509	8%	/mnt/disk1

以上命令统计的是当前系统已经安装的文件系统。

3. dd——转换和拷贝文件

dd 命令可以用来产生交换文件，也常常用来制作软盘的映像文件。制作软盘映像并不需要建立文件系统。

dd if=输入文件名 of=输出文件名 count=块数

例如：

```
[root @redflag /root]#dd if=/dev/fd0 of=floppy-image count=1440
```

```
1440+0 records in
```

```
1440+0 records out
```

以上命令从软盘产生一个映像文件 floppy-image。更换软盘后使用以下命令，把产生的映像文件写入到新的软盘：

```
[root @redflag /root]#dd if=floppy-image of=/dev/fd0 count=1440
```

```
1440+0 records in
```

```
1440+0 records out
```

本章小结

本章介绍了磁盘分区的基本知识、Linux 文件系统的基本知识，特别是 Linux 下不同文件系统如何形成一个无缝的目录。还着重介绍了用 `fdisk` 命令创建磁盘分区、用 `mkfs` 命令在分区上建立文件系统以及如何用 `mount` 命令安装文件系统。Linux 文件系统可以通过配置文件 `/etc/fstab` 来实现自动安装。最后本章介绍了文件系统维护的常用命令 `fsck`。

习 题

1. Linux 下不同文件系统如何形成一个无缝的目录？
2. 要使用硬盘上的某一空闲空间一般要经过什么步骤？使用什么命令？
3. `/etc/fstab` 文件中有如下一行：

```
/dev/hda1 / reiserfs defaults,noatime 1 1
```

请解析各个字段的含义。

第7章 Shell 编程

Shell 的原意是外壳，用来形容物体外部的框架，使整体具有轮廓而不至于松散变形。对于操作系统来说，Shell 搭起了用户与操作系统的桥梁，它提供了基本的操作界面，让用户可以下达各种命令，在系统中进行操作，产生彼此间的交互关系。

7.1 Shell 的基本概念

7.1.1 Shell 的概念

Shell 是一个命令语言解释器，它拥有自己内建的 Shell 命令集，Shell 也能被系统中其他应用程序调用。用户在提示符下输入的命令都由 Shell 解释后传给 Linux 核心。Shell、用户及 Linux 操作系统内核之间关系如图 7-1 所示。

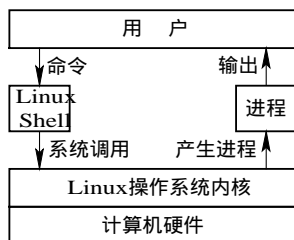


图 7-1 Shell、用户及 Linux 操作系统内核之间关系

有一些命令(比如改变工作目录命令 `cd`)是包含在 Shell 内部的。还有一些命令(例如拷贝命令 `cp` 和移动命令 `rm`)是存在于文件系统中某个目录下的单独的程序。对用户而言，不必关心一个命令是建立在 Shell 内部还是一个单独的程序。

Shell 接到用户输入的命令后首先检查命令是否是内部命令，若不是再检查是否是一个应用程序(这里的应用程序可以是 Linux 本身的实用程序，如 `ls` 和 `rm`，也可以是购买的商业程序，如 `xv`，或者是自由软件，如 `emacs`)。然后，Shell 在搜索路径里寻找这些应用程序(搜索路径就是一个能找到可执行程序的路径列表)。如果键入的命令不是一个内部命令并且在路径里没有找到这个可执行文件，将会显示一条错误信息。如果能够成功找到命令，该内部命令或应用程序将被分解为系统调用并传给 Linux 内核。Shell 对命令的解释过程如图 7-2 所示。

Shell 的另一个重要特性是它自身就是一个解释型的程序设计语言。Shell 程序设计语言支持绝大多数在高级语言中能见到的程序元素，如函数、变量、数组和程序控制结构。Shell 编程语言简单易学，任何在提示符中能键入的命令都能放到一个执行的 Shell 程序中。

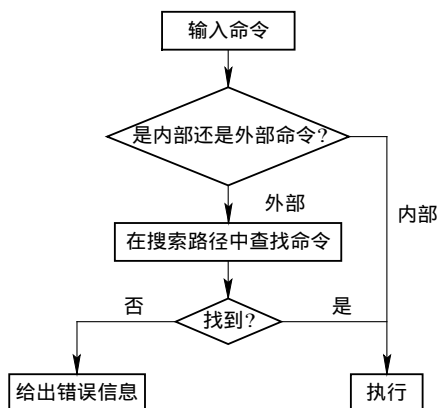


图 7-2 Shell 对命令的解释过程

当普通用户成功登录后，系统将执行一个称为 Shell 的程序。正是 Shell 进程提供了命令行提示符。作为默认值，对普通用户用“\$”作提示符，对超级用户(root)用“#”作提示符。

一旦出现了 Shell 提示符，就可以键入命令名称及命令所需要的参数。Shell 将执行这些命令。如果一条命令花费了很长的时间来运行，或者在屏幕上产生了大量的输出，可以在键盘上按【Ctrl-d】发出中断信号来中断它(在正常结束之前，中止它的执行)。

当用户准备结束登录对话进程时，可以键入 logout 命令、exit 命令或文件结束符(EOF)(按【Ctrl-d】实现)，结束登录。

下面的例子显示了 Shell 如何来工作：

```
[root@redflag /root]#cal 3 2003
```

```
三月 2003
```

```

日 一 二 三 四 五 六
                        1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31

```

```
[root@redflag /root]#
```

该命令的含义是显示 2003 年 3 月的日历，在接收到这个命令后，Shell 便执行它，执行完后又返回到命令接收状态。

【注意】“cal”是系统中一个命令的名字，用于显示日历。“3”和“2003”是 cal 的两个参数。

另外，用户键入有关命令后，如果 Shell 找不到以其中的命令名为名字的程序，就会给出错误信息。例如，如果用户键入：

```
[root@redflag /root]#mytest
```

```
bash:mytest:command not found
```

此时可以看到，用户得到了一个没有找到该命令的错误信息，这样的错误信息一般出现在用户输入错误命令时。

7.1.2 Shell 的种类

Linux 中的 Shell 有多种类型，其中最常见的是 Bourne Shell (sh)、C Shell (csh)和 Korn Shell (ksh)。三种 Shell 各有优缺点。Bourne Shell 是 Unix 最初始的 Shell，并且在每种 Unix 上都可以使用。Bourne Shell 在 Shell 编程方面相当优秀，但在处理与用户的交互方面做得不如其他几种 Shell。Bash(Bourne Again Shell)是 Bourne Shell 的扩展，与 Bourne Shell 完全向下兼容，并且增加了许多特性。它还包含了很多 C Shell 和 Korn Shell 中的优点，有灵活和强大的编程接口，同时又有很友好的用户界面。

C Shell 是一种比 Bourne Shell 更适于编程的 Shell，它的语法与 C 语言很相似。Linux 为喜欢使用 C Shell 的人员提供了 Tcsh。Tcsh 是 C Shell 的一个扩展版本。Tcsh 包括命令编辑、可编程单词补全、拼写矫正、历史命令替换、作业控制和类似 C 语言的语法，它提供比 Bourne Shell 更多的提示符参数。

Korn Shell 集合了 C Shell 和 Bourne Shell 的优点并且和 Bourne Shell 完全兼容。Linux 系统提供了 pdksh(ksh 的扩展)，它支持任务控制，可以在命令行上挂起、后台执行、唤醒或终止程序。

Bash 是大多数 Linux 系统(包括红旗 Linux 系统)的默认 Shell。Bash 有以下的优点：

(1) 补全命令。当你在 Bash 命令提示符下输入命令或程序名时，你不必输全命令或程序名，按【Tab】键，Bash 将自动补全命令或程序名。

(2) 通配符。在 Bash 下可以使用通配符“*”和“?”。“*”可以替代多个字符，而“?”则替代一个字符。

(3) 历史命令。Bash 能自动跟踪用户每次输入的命令，并把输入的命令保存在历史列表缓冲区。缓冲区的大小由 HISTSIZE 变量控制。当用户每次登录后，home 目录下的“.bash_history”文件将初始化历史列表缓冲区。也能通过 history 和 fc 命令执行、编辑历史命令。

(4) 别名。在 Bash 下，可用 alias 和 unalias 命令给命令或可执行程序起别名和清除别名，这样就可以用自己习惯的方式输入命令。

(5) 输入/输出重定向。输入重定向用于改变命令的输入，输出重定向用于改变命令的输出。输出重定向更为常用，它经常用于将命令的结果输入到文件中，而不是屏幕上。输入重定向的命令是“<”，输出重定向的命令是“>”。

【实例 7.1】

输入重定向：

```
[root@redflag /root]#wc</etc/passwd
20 23 726
```

输出重定向：

```
[root@redflag /root]#ls>dir.out
```

上面命令将 ls 命令的输出保存为文件 dir.out。

```
[root@redflag /root]#ls>>dir1.out
```

“>>”表示要将一条命令的输出结果追加到文件 dir1.out 的后面，该文件的原有内容不被破坏，如果使用“>”，则文件原有内容被覆盖。

(6) 管道。管道用于将一系列的命令连接起来，也就是把前面命令的输出作为后面命令的输入。管道的命令是“|”。

【实例 7.2】

```
[root@redflag /root]# cat dir.out|grep "test "|wc -l
```

管道将 cat 命令(列出一个文件的内容)的输出送给 grep 命令，grep 命令在输入里查找单词 test，grep 的输出则是所有包含单词 test 的行，这个输出又被送给 wc 命令，wc 命令统计出输入中的行数。假设 dir.out 的内容如下：

```
This is a test.
Today is Monday.
Eveybody is here.
We should have a test.
The final test is over.
```

那么该管道运行的结果应该是 3。

(7) 提示符。Bash 有两级提示符。第一级提示符就是登录 Shell 时见到的，默认为“\$”。可以通过重新给 PS1 变量赋值来改变第一级提示符。当 Bash 需要进一步提示以便补全命令时，它会显示第二级提示符。第二级提示符默认为“>”，可以通过重新给 PS2 变量赋值来改变第二级提示符。一些特殊意义的字符也可以加入提示符赋值中。

(8) 作业控制。作业控制是指在一个作业执行过程中，控制执行的状态。可以挂起一个正在执行的进程，并在以后恢复执行该进程。按下【Ctrl-z】组合键，挂起正在执行的进程，用 bg 命令使进程恢复在后台执行，用 fg 命令使进程恢复在前台执行。

【技巧】 如何查看用户登录系统时默认使用的 Shell？

最简单的方式是执行 echo 命令，查询系统环境变量的值：

```
[root@redflag /root]# echo $SHELL
/bin/bash
```

执行 finger 命令查看用户数据，也能看到用户默认的 Shell：

```
[root@redflag /root]#finger -l root
Login: root                      Name: root
Directory: /root                 Shell: /bin/bash
On since Fri Mar  7 08:14 (EST) on tty1      20 minutes 24 seconds idle
On since Fri Mar  7 08:17 (EST) on tty2      16 minutes idle
On since Fri Mar  7 08:14 (EST) on pts/0     20 minutes 6 seconds idle
On since Fri Mar  7 08:19 (EST) on pts/1 from :0.0
No mail.
Plan:
listening .....
```

查看/etc/passwd 文件，注意用“:”分隔的最后一列，就是该用户的默认的 Shell，该文件的内容如下：

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
```

```
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
:
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/bin/false
test:x:500:500::/home/test:/bin/bash
```

7.1.3 创建及执行 Shell 脚本

1. 创建 Shell 脚本文件

Shell 脚本是指使用用户环境 Shell 提供的语法所编写的脚本。如果你经常用到相同执行顺序的操作命令，便可以将这些命令写成脚本文件，这样以后要做同样的事情时，只要在命令行输入其文件名即可。本节会以一个简单的实例，来介绍如何创建与执行 Shell 脚本。

【实例 7.3】

该实例会显示当前的日期时间、执行路径、用户账号及所在的目录位置。

在文本编辑器中输入下列内容，并保存为 showinfo：

```
#!/bin/bash
#This script is a test!
echo -n "Date and time is: "
date
echo -n "The Executable path is: "$PATH
echo "Your name is: `whoami`"
echo -n "Your Current directory is: "
pwd
#end
```

【说明】 文件中以“#”开头的行是注释行，在执行时会被忽略。特别是其中的第一行“#!/bin/bash”用来指定脚本以 bash 执行，如果要设置以 tcsh 执行，则应设成“#!/bin/tcsh”。要指定执行的 Shell 时，一定要将它写在第一行；如果没有指定，则以当前正在执行的 Shell 来解释。

echo 命令用来显示提示信息，参数“-n”表示在显示信息时不自动换行(默认会自动换行)。如第三行的“echo -n ...”，表示此行输出后不换行。下一行的 date 命令执行结果就会接在“Date and time is:”之后。

第六行的‘whoami’字符串左右的反引号(`)用于命令置换，也就是将它所括起来的字符串视为命令执行，并将其输出字符串在原地展开。第三行也可以改写成和第六行相同的写法 echo "Date and time is: `date`"。

2. 执行 Shell 脚本

编辑脚本之后，如何来运行呢？一般有四种方法，下面来分别介绍。

(1) 将脚本的权限设置为可执行，这样就可以在 Shell 的提示符下直接执行。可以使用 chmod 命令更改 Shell 脚本的权限：

```
chmod u+x showinfo
chmod u+x,g+x showinfo
chmod a+x showinfo
```

其中第一行是将用户自己的权限设置为可执行，第二行是将用户自己和同组的权限设置为可执行，第三行将所有人的权限设置为可执行，包括用户自己、同组用户和其他用户。

那么如何指定使用哪一个 Shell 来解释执行脚本呢？

如果 Shell 脚本的第一个非空白字符不是“#”，则它会使用 Bourne Shell。

如果 Shell 脚本的第一个非空白字符是“#”，但不是以“#!”开头时，则它会使用 C Shell。

如果 Shell 脚本以“#!”开头，则“#!”后面所跟的字符串就是所使用 Shell 的绝对路径名。Bourne Shell 的路径名称为/bin/sh，而 C Shell 则为/bin/csh。

(2) 执行 Shell 脚本想要执行的 Shell，其后跟随 Shell 脚本的文件名作为命令行参数。例如，使用 tcsh 执行上面的 Shell 脚本：

```
tcsh showinfo
```

此命令启动一个新的 Shell，并令其执行 showinfo 文件。

(3) 在 pdksh 和 Bash 下使用“.”命令，或在 tcsh 下使用 source 命令。例如，在 pdksh 和 Bash 下执行上面的 Shell 脚本：

```
./showinfo
```

或在 tcsh 下执行上面的 Shell 脚本：

```
source showinfo
```

(4) 使用命令替换。这是一个相当有用的方法，如果想要使某个命令的输出成为另一个命令的参数时，就可以使用这个方法。将命令列于两个“`”号之间，而 Shell 会以这个命令执行后的输出结果代替这个命令以及两个“`”符号。例如：

```
str="Current directory is `pwd`"
echo $str
```

结果如下：

```
Current directory is /root
```

在上面的例子中，pwd 这个命令输出/root，而后整个字符串代替原来的‘pwd’，所以 str 变量的内容则会包括 pwd 命令的输出。

需要注意的是命令替换方式，灵活运用这种方式，我们可以将一系列小工具装配成一个强大的工具，用来解决复杂的问题。

7.2 Shell 语法

7.2.1 Shell 变量

所谓变量，就是可存放数据的识别符。例如，x=10，x 是个变量名，而 10 则为存放的数据。在 pdksh 和 Bash 中，给变量赋值的方法是一样的，即在变量名后紧跟着等号和变量值。例如，要把变量 10 赋给变量 a，则使用下面的命令：

```
a=10
```

【注意】在等号的两边不能有空格。使用变量之前无须事先定义。

在 tcsh 中，可以使用如下的命令：

```
set a=10
```

给字符串赋值的方法与给整数赋值的方法相同。例如：

```
str=name
```

为了读取变量的值，需要使用“\$”符号。例如，用如下的命令将 a 变量的值输出到屏幕上：

```
echo $a
```

除了自定义的变量外，还有一些系统默认的特殊变量，这些变量也是常常应用在脚本程序中的，这些常用的变量如表 7-1 所示。

表 7-1 常用的变量

变 量	功 能
HOME	用户的主目录
PATH	执行命令时所搜索的目录
TZ	时区
MAILCHECK	检查新邮件的时间间隔
PS1	在 Shell 命令行的提示符
PS2	在命令尚未输入完时，Shell 要求再输入的提示符
MANPATH	man 指令的搜索路径
TERM	终端的类型

要想查看变量的值，用 echo 命令即可。

表 7-1 介绍的变量是系统默认的变量，还有一些变量是在执行 Shell 程序时系统给设置好的，并且不能加以修改，这些变量也是相当有用的，如表 7-2 所示。

表 7-2 系统设置好的变量

变 量	功 能
#	存储 Shell 程序中命令行参数的个数
?	存储上一个执行命令的返回值
0	存储 Shell 程序的程序名
*	存储 Shell 程序的所有参数
@	存储所有命令行输入的参数
\$	存储 Shell 程序的 PID
!	存储上一个后台执行命令的 PID

【实例 7.4】

下面的程序会显示所有参数及其总数，将其存为 showarg 文件，源代码如下：

```
#!/bin/bash
echo "All argument list : $@"          //存储所有命令行输入的参数
echo "The total number of argument is : $#"/>
#end
存盘后执行 chmod a+x showarg
[root@redflag /root]# ./showarg red flag linux //输入三个参数
All argument list : red flag linux          //显示参数
The total number of argument is : 3        //显示参数的个数
```

7.2.2 数值运算

如果需要处理数值运算，可以使用 `expr` 命令，下面列出可以使用的数值运算符及其用法：

```
expr expression
```

【说明】 `expression` 是由字符串以及运算符所组成的，每个字符串或运算符之间必须用空格隔开。

下面列出了运算符的种类及功能，运算符的优先顺序以先后次序排列，但可以利用小括号来改变运算的优先次序，并将其运算结果输出到标准输出设备上，常见的运算符如下：

```
* 乘法；
/ 除法；
% 取余数；
+ 加法；
- 减法；
< 小于；
<= 小于等于；
= 等于；
!= 不等于；
> 大于；
>= 大于等于；
& AND 运算；
| OR 运算；
```

【实例 7.5】

用 Shell 编程计算 $2*(3+4)$ 的值。

程序的源代码如下：

```
#!/bin/bash
sum=`expr 2 \* \( 3 + 4 \)`
echo "The sum is $sum"
#end
```

【注意】 第二行的等号两边没有空格。

第二行使用的是反引号(在 Esc 键的正下方)。

当表达式中含有“*”、“(”、“)”等非字母或数字符号时，必须在其前面加上“\”，以免被 Shell 解释成其他意义。例如，在第二行如果没有“\”，那么结果将是一个字符串，而不是 14，结果将是错误的。

7.2.3 条件命令

1. 条件测试

在 Bash 和 pdksh 环境中，测试条件表达式只能通过使用 test 命令来完成。test 命令的用法如下：

```
test expression
```

或

```
[expression]
```

【实例 7.6】

```
[root@redflag /root]# test 8 = 8           //测试字符串 8 等于 8
[root@redflag /root]# echo $?              //显示其返回值，变量$存储上一个执行命令返回值
0                                           //其值为真，返回 0
[root@redflag /root]# test 8 = 9           //测试字符串 8 等于 9
[root@redflag /root]# echo $?
1                                           //其值为真，返回 0
[root@redflag /root]# [ 8 = 8 ]            //省略“test”，改用中括号，作用完全一样
[root@redflag /root]# echo $?
0
[root@redflag /root]# [ 8 = 9 ]            //省略“test”，改用中括号，作用完全一样
[root@redflag /root]# echo $?
1
```

【注意】上例中等号两边和中括号两边内侧要有空格。

test 命令可以和多种系统运算符一起使用。这些运算符可以分为四类：整数运算符、字符串运算符、文件运算符和逻辑运算符。在不同的 Shell 环境中，条件表达式的语法有些不同，本书主要介绍 Bash、pdksh 和 tcsh，其中 Bash 和 pdksh 的语法相同。下面分别介绍。

首先，在 Bash 和 pdksh 环境中：

(1) 数值运算符：用来判断数值表达式的真假。可用的运算符如下：

- int1 -eq int2 如果 int1 = int2，则为真。
- int1 -ge int2 如果 int1 >= int2，则为真。
- int1 -gt int2 如果 int1 > int2，则为真。
- int1 -le int2 如果 int1 <= int2，则为真。
- int1 -lt int2 如果 int1 < int2，则为真。
- int1 -ne int2 如果 int1 != int2，则为真。

【实例 7.7】

```
[root@redflag /root]# [ text -gt abc ]    //测试整数 text 是否大于 abc
```

```
[ : text : integer expression expected      //数据类型错误
[root@redflag /root]# [ 100 -gt 200 ]        //100 大于 200 吗 ?
[root@redflag /root]# echo $?               //显示其返回值
1                                           //其值为假, 返回 1
```

【注意】数据类型的区别完全取决于运算符的使用。如果将数值运算符用于一般字符串，将像上例一样得到错误信息。

(2) 字符串运算符：用来判断字符串表达式的真假。可用的运算符如下：

- str1 = str2 如果 str1 和 str2 相同，则为真。
- str1 != str2 如果 str1 和 str2 不相同，则为真。
- str 如果 str 不为空，则为真。
- -n str 如果 str 的长度大于零，则为真。
- -z str 如果 str 的长度等于零，则为真。

【实例 7.8】

```
[root@redflag /root]# [ text = TEXT ]      //测试字符串 text 是否等于 TEXT
[root@redflag /root]# echo $?              //显示其返回值
1                                           //其值为假, 返回 1(区分大小写)
[root@redflag /root]# [ $PWD ]             //测试环境变量 PWD 是否有值
[root@redflag /root]# echo $?
0                                           //其值为真, 返回 0
```

【注意】字符串运算符不可以使用通配符，同时，在等号的两边要留空格。

(3) 文件运算符：用来判断文件是否存在、文件类型及属性。可用的运算符如下：

- -d filename 如果 filename 为目录，则为真。
- -f filename 如果 filename 为普通的文件，则为真。
- -r filename 如果 filename 为只读，则为真。
- -s filename 如果 filename 的长度大于零，则为真。
- -w filename 如果 filename 为可写，则为真。
- -x filename 如果 filename 为可执行，则为真。

【实例 7.9】

```
[root@redflag /root]# [ -d /etc ]          //判断/etc 是否为目录
[root@redflag /root]# echo $?              //显示其返回值
0                                           //其值为真, 返回 0
[root@redflag /root]# [ -w /etc ]          //判断 root 对目录/etc 是否有写的权限
[root@redflag /root]# echo $?              //显示其返回值
0                                           //其值为真, 返回 0, 有写的权限
```

(4) 逻辑运算符：用来结合表达式或取得表达式相反值。可用的运算符如下：

- ! expr 如果 expr 为假，则返回真。
- expr1 -a expr2 如果 expr1 和 expr2 同时为真，则返回真。
- expr1 -o expr2 如果 expr1 或 expr2 有一个为真，则返回真。

【实例 7.10】

```
[root@redflag /root]# [ -f file1 -a -w file1 ] //file1 存在且具有可写的权限
[root@redflag /root]# echo $? //显示其返回值
0 //其值为真，返回 0
```

其次，对于 tcsh 来说，它的语法更接近于 C 语言，所支持的表达式运算符也分为整数运算符、字符串运算符、文件运算符和逻辑运算符四类。

(1) 数值运算符：用来判断数值表达式的真假。可用的运算符如下：

- `int1 <= int2` 如果 `int1 <= int2`，则为真。
- `int1 >= int2` 如果 `int1 >= int2`，则为真。
- `int1 > int2` 如果 `int1 > int2`，则为真。
- `int1 < int2` 如果 `int1 < int2`，则为真。

(2) 字符串运算符：用来判断字符串表达式的真假。可用的运算符如下：

- `str1 == str2` 如果 `str1` 和 `str2` 相同，则为真。
- `str1 != str2` 如果 `str1` 和 `str2` 不相同，则为真。

(3) 文件运算符：用来判断文件是否存在、文件类型及属性。可用的运算符如下：

- `-r filename` 如果 filename 为只读，则为真。
- `-w filename` 如果 filename 为可写，则为真。
- `-x filename` 如果 filename 为可执行，则为真。
- `-e filename` 如果 filename 存在，则为真。
- `-o filename` 如果当前用户拥有 filename，则为真。
- `-z filename` 如果 filename 的长度为零，则为真。
- `-f filename` 如果 filename 为普通的文件，则为真。
- `-d filename` 如果 filename 为目录，则为真。

(4) 逻辑运算符：用来结合表达式或取得表达式相反值。可用的运算符如下：

- `! expr` 如果 `expr` 为假，则返回真。
- `expr1 && expr2` 如果 `expr1` 和 `expr2` 同时为真，则返回真。
- `expr1 || expr2` 如果 `expr1` 或 `expr2` 有一个为真，则返回真。

2. 条件语句

1) if 语句

if 语句可根据表达式的值是真或假来决定要执行的程序段落，其语法如下：

if expression1	//若 expression1 为真
then	
commands	//则执行这些命令
elif expression2	//否则若 expression2 为真
then	
commands	//则执行这些命令
else	//若以上的表达式都不成立
commands	//则执行这些命令
fi	//结束 if 语句

fi 是 if 语句的结束符(刚好是 if 倒过来), 必须与 if 成对出现, 而 elif 及 else 子句可有可无。elif 是 else if 的简写, 当 if 的表达式不成立时, 才会接着测试 elif 的表达式。如果 if 及 elif 的测试条件都不成立, 最后才会执行 else 子句内的命令。一个 if 可以有好几个 elif 子句, 但只能有一个 else 子句。

【实例 7.11】

此实例将显示目录内是否有 example_if 文件, 源代码如下:

```
#!/bin/bash
#This is example1 about if.
if [ -f example_if ]           //判断文件是否存在
then
    echo "There is a example_if file in current directory. "
else
    echo "no example_if file in current directory. "
fi
#end
```

【实例 7.12】

此实例在键盘上读取一个字符, 然后根据字符的值来判断。

```
#!/bin/bash
#This is example2 about if.
echo -n "Please input the answer: "    // -n 不换行
read I                                //从键盘读入
if [ $I = y ]
then
    echo "The answer is right"
elif [ $I = n ]
then
    echo "The answer is wrong"
else
    echo "Bad Input."
fi
#end
```

【技巧】 如何从键盘一次读入多个变量?

用语句 read var1 var2

变量之间用空格隔开。例如:

```
read name gender                //读入两个变量
echo $name
echo $gender
```

假如在键盘输入 linux man, 则上边的语句执行的结果为:

```
linux                //变量 name 的值
man                  //变量 gender 的值
```

如果输入文本过长，Shell 将所有的超长的部分赋给最后一个变量。

2) case 语句

case 语句用来从很多的测试条件中选择符合的条件执行。其语法如下：

```
case string in
    str1)                //测试 string 字符串
                        //若 str1 符合
    commands;;           //则执行这些命令
    str2)
    commands;;
    *)                   //若 str1 和 str2 都不符合
    commands;;           //则执行这些命令
esac                    //结束 case 语句
```

case 语句适用于字符串的比较，其测试条件可用通配符。双分号(;;)为测试条件的结束，在每一个测试条件成立后，一直到双分号之前的命令，都会被 Shell 执行。使用通配符作为测试条件时，不要在字符串左右加上双引号，因为这样会使字符串无法正确匹配。

由于所有字符串都可以与通配符“*”匹配，因此“*”之后的命令可以视为 case 语句默认的执行命令。

【实例 7.13】

此实例检查命令行的第一个参数是否为“-i”或“-e”，如果是“-i”，则计算由第二个参数指定的文件中以 i 开头的行数；如果是“-e”，则计算由第二个参数指定的文件中以 e 开头的行数。如果第一个参数既不是“-i”也不是“-e”，则在屏幕上显示一条错误的信息。该实例源代码如下：

```
#!/bin/bash
#This is example1 about case.
case $1 in
    -i )                // $1 是命令行第一个参数
    count=`grep ^i $2|wc -l`           //查找并计算以 i 开头的行数,“`”是反引号
    echo "The number of lines in $2 that start with an i is $count. "
    ;;
    -e )
    count=`grep ^e $2|wc -l`
    echo "The number of lines in $2 that start with an e is $count. "
    ;;
    * )                 //默认匹配
    echo "That option is not recognized. "
    ;;
esac                    //和 case 成对出现
#end
```

将其存为 case_1，然后执行 chmod a+x case_1。

执行程序 case_1：

```
[root@redflag /root]# ./case_1 -i file1
```

The number of lines in file1 that start with an i is 8.

【实例 7.14】

此实例使用 case 语句建立一个菜单，这也是 case 最为常见的一种应用。该实例源代码如下：

```
#!/bin/bash
#display a menu
echo "-----"
echo "1 Restore"
echo "2 Backup"
echo "3 Upload"                                //前四行显示一个菜单
echo
#read and execute the user's selection
echo -n "Enter Choice:  "
read CHOICE                                  //读取键盘输入
case "$CHOICE" in
    1|R) echo "1 Restore";;                  //如果输入是 1 或者是 R，则显示 Restore
    2|B) echo "2 Backup";;
    3|U) echo "3 Upload" ;;
    *) echo "sorry $CHOICE is not a valid choice! " //默认匹配
    exit 1                                     //退出程序
esac
#end
```

【注意】 在 case 模式中可以使用逻辑操作，在实例 7.14 第 12 行中，“1|R”表示 1 或 R 都可以匹配，“|”表示逻辑或运算。

7.2.4 循环命令

Shell 中提供了几种执行循环的命令，比较常见的命令有 for、while、until、shift 命令。下面分别介绍。

1. for

for 语句有两种语法。

第一种：

```
for var in list
do
commands
done                                //对 list 中的每一个元素
                                    //执行这些操作
                                    //循环语句结束
```

在使用此形式时，对在 list 中的每一项，for 语句都执行一次。List 可以是包括几个单词并且有空格分隔开的变量，也可以是直接输入的几个值。每执行一次循环，var 都被赋予 list 中的当前值，直到最后一个为止。for 语句的 in 子句和 case 的子句相同，都可以使用通配符。

第二种：

```
for var
do
statements                                //对每一个命令行参数执行这些操作
done                                       //循环语句结束
```

使用这种形式时，对变量 var 中的每一项，for 语句都执行一次。此时 Shell 程序假定 var 包含 Shell 程序在命令行的所有位置参数。一般情况下，此种方式也可以写成：

```
for var in "$@"
do
statements
done
```

【实例 7.15】

该实例是 for 语句的 in 子句使用通配符的例子，它的功能是显示当前目录下所有文本文件(*.txt)的名称和内容，该程序源代码如下：

```
#!/bin/bash
#This is example 1 about for.
for file in *.txt                                //对目录下的每个 txt 文件
do
echo "-----"
echo $file                                       //显示文件名
echo "-----"
cat $file                                       //显示文件内容
done
#end
```

【实例 7.16】

此程序可以以任何数目的文本文件作为命令行参数。它读取每一个文件，把其中的内容转换成大写字母，然后将结果存在以“.caps”作为扩展名的同样名字的文件中，该程序源代码如下：

```
#!/bin/bash
#This is example 2 about for.
for file
do
tr a-z A-Z <$file >$file.caps                //将文件的内容转换为大写字母并另存
done
#end
```

【实例 7.17】

此程序可以实现把直接输入的几个值显示出来，该程序源代码如下：

```
#!/bin/bash
#This is example 3 about for.
for p in 1 2 3 4 5
do
echo $p
done
#end
```

【实例 7.18】

此实例实现求命令行所有整数之和，源代码如下：

```
#!/bin/bash
#This is example 3 about for.
sum=0                                //清零
for p in $*                          //命令行所有参数
do
sum=`expr $sum + $p`                //加法运算
done
echo "the total is $sum"
#end
```

2. while 及 until

while 语句与 until 语句的语法结构和用途相似。while 语句会在测试条件为真时循环执行，语法如下：

```
while expression
do
    commands
done
```

而 until 语句会在其测试条件为假时循环执行，语法如下：

```
until expression
do
    commands
done
```

【实例 7.19】

此实例实现计算 1 ~ 5 的平方，程序源代码如下：

```
#!/bin/bash
#This is example 1 about while.
int=1                                //int 赋初值
while [ $int -le 5 ]                 //当 int<=5 时，执行循环
do
```

```
sq='expr $int \* $int'           //计算 int 平方
echo $sq
int='expr $int + 1'             //int 加 1
done
echo "finished"
#end
```

【实例 7.20】

此实例列出所带的所有参数以及它的位置号，程序源代码如下：

```
#!/bin/bash
#This is example 2 about while.
count=1
while [ -n "$*" ]               /*表示存储 Shell 程序的所有参数
do
echo "This is parameter number $count $1"
shift                          //将命令行参数左移一位
count='expr $count + 1'
done
#end
```

3. shift

shift 命令用来将命令行参数左移。假设当前的命令行有 3 个参数，分别是：

```
$1=-r $2=file1 $3=file2
```

则在执行 shift 命令之后，其值会变成：

```
$1=file1 $2=file2
```

shift 命令也可以指定参数左移的位数，下面的命令将使命令行参数左移 2 个位置：

```
shift 2
```

shift 命令常与 while 语句或 until 语句一起使用。

【实例 7.21】

该实例将一个文件中的英文字母全部转换为大写字母，程序源代码如下：

```
#!/bin/bash
#This is example about shift.
while [ "$1" ]
do
if [ "$1" = "-i" ]           //如果是选项-i
then
infile="$2"                 //指定输入文件名
shift 2                      //命令行参数左移 2 位
elif [ "$1" = "-o" ]        //若是选项-o
then
```

```

        outfile="$2"                                //指定输出文件名
        shift 2
    else
        echo "Program $0 does not recognize option $1 "
    fi
done
tr a-z A-Z < $infile > outfile                      //转换为大写字母
#end

```

tr 命令将第一个参数所设置的范围对应到第二个参数所设置的范围。

将上面的程序存为 upcase 并将其权限设置为可执行，下面是执行结果：

```

[root@redflag /root]# cat a.txt                      //显示源文件
I love linux.

[root@redflag /root]# ./upcase -i a.txt -o z.out      //转换成大写
[root@redflag /root]# cat z.out                      //显示转换后文件的内容
ILOVE LINUX.

```

4. break 和 continue


在 Shell 的 for、while、until 循环语句中也可以使用如 C 语言的 break 和 continue 语句以跳离现有的循环。break 语句用于中断循环的执行，将程序流程移至循环语句结束之后的下一个命令。而 continue 语句则忽略之后的命令，将程序流程移至循环开始的地方。

break 和 continue 语句都可以加上数字，以指示要跳出的循环数目。例如，下面的 continue 语句将跳出两层循环：

```

while expression1
do
    while expression2
    do
        :
        continue 2
        :
    done
done

```



//从第二层跳到第一层

【注意】 要跳过几层循环，就加上数字，若指定的数字大于最大的循环层数，就跳至最外一层循环执行，若只跳一层，数字 1 可以省略。

【实例 7.22】

该实例用来模拟工业上的操作流程控制，程序源代码如下：

```

#!/bin/bash
for x in a b c d e f g h i
do
    for y in 1 2 3 4 5 6 7 8 9

```

```

do
echo "current job is $x$y"
echo "input n to do next job"
echo "      s to skip the other jobs in current level"
echo "      x to terminate all jobs"
read action
if [ $action = n ]
then
    echo "do next job"
        continue                //往前跳一层
    elif [ $action = s ]
    then
        echo "skip the other jobs in current level"
        continue 2                //往前跳两层
    elif [ $action = x ]
    then
        echo "terminate all jobs"
        break 2                //往后跳两层
    fi
done
done
#end

```

7.2.5 函数的定义和使用

Shell 脚本也有自定义函数的功能。当脚本变得很大时，可以将脚本文件中常用的程序写成函数，这样可以使脚本更小、更易于维护，同时整个结构更加严谨。定义函数的语法如下：

```

fname () {
    Shellcommands
}

```

函数的使用方法与外部命令一样，只要直接使用函数名即可。在使用函数时，一样可以传入参数。函数处理参数的方式与脚本文件处理命令行参数的方法是一样的。在函数中，\$1 是指传入函数的第 1 个参数，\$2 是指传入函数的第 2 个参数。同时也可以使用 shift 命令来移动函数参数。

【实例 7.23】

该实例示范如何将命令行中输入的数字传入函数，并显示最大的数值，将文件存为 maxvalue，程序源代码如下：


```
#!/bin/bash
max()
{
    while [ $1 ]
    do
        if [ $maxvalue ]
        then
            if [ $1 -gt $maxvalue ]
            then
                maxvalue=$1
            fi
        else
            maxvalue=$1
        fi
        shift
    done
    return $maxvalue
}

max $@
echo "Max Value is :$maxvalue"
#end
```

实例执行结果：

```
[root@redflag /root]#maxvalue 100 30 50 7 60 20 150
Max Value is :119
```

7.3 正则表达式

正则表达式是一种可以用于模式匹配和替换的工具，可以让用户通过使用一系列的特殊字符构建匹配模式，然后把匹配模式与待比较字符串或文件进行比较，根据比较对象中是否包含匹配模式，执行相应的程序。例如，在一个文本中抽取一个单词，它的头两个字符是大写的，后面紧跟四个数字。如果不使用正则表达式，在 Shell 中将不能实现这个操作。而利用正则表达式就很容易完成。正则表达式起始于 Unix 系统，目前广泛应用于各种脚本语言中，不但在 PHP, Perl, JavaScript 中找到它的身影，而且正则表达式本身就是如 sed、awk、grep、perl 等程序的选项开关内容之一。

7.3.1 正则表达式基本元字符及使用

正则表达式本身是一些特殊或不特殊字符的集合，这些字符被称为元字符，常见的元

字符如表 7-3 所示。

表 7-3 常见的元字符

元字符	含 义
^	只匹配行首
\$	只匹配行尾
*	一个单字符后紧跟*, 匹配 0 个或多个此单字符
[]	匹配[]内字符, 可以是一个单字符, 也可以是字符序列, 可以使用-表示[]内字符序列范围, 如用[1-5]表示[12345]
\	用来屏蔽一个元字符的特殊含义, 因为有时在 Shell 中一些元字符有特殊含义, \可以使其失去原有意义
.	匹配任意单字符
pattern{n}	用来匹配前面 pattern 出现的次数, n 为次数
pattern{n,}	含义同上, 但次数最少为 n 次
pattern{n,m}	含义同上, 但 pattern 出现次数在 n 和 m 之间

现在详细讲解表 7-3 中每项的含义。

1. 使用句点匹配单字符

句点“.”可以匹配任意单字符。

【实例 7.24】

(1) 使用“ls -l”命令可以匹配一定的权限：

```
...X..X..X
```

此格式匹配用户本身、用户组及其他组的执行权限。

```
drwx----- 不匹配
```

```
-rwxrwxr-- 不匹配
```

```
drwx--x--x 匹配
```

```
lrwxrwxrwx 匹配
```

(2) 假定正在过滤一个文本文件, 对于一个有 10 个字符的脚本集, 要求前 4 个字符之后为 XC, 匹配操作如下：

```
....XC....
```

以上例子解释为前 4 个字符任意, 5、6 字符为 XC, 后 4 个字符也任意按下例运行:

```
TEXTVITEST 不匹配
```

```
ABC4XCDCBA 匹配
```

```
PPPPXXAAAA 不匹配
```

```
WEWEXCQUTE 匹配
```

【注意】“.”允许匹配 ASCII 集中的任意字符、字母或数字。

2. 在行首以 “^” 匹配字符串

“^” 只允许在一行的开始匹配字符或单词。

【实例 7.25】

(1) 使用 “ls -l” 命令，并匹配目录。之所以可以这样做，是因为在 “ls -l” 命令的结果中，如果每行的第一个字符是 d，就代表一个目录：

```
d
d          匹配
          不匹配
d          匹配
l          不匹配
使用 “^” , 结果将匹配以 ^ 开始的行。
```

```
          不匹配
          匹配
          不匹配
d          不匹配
可以将各种模式混合使用。
```

```
          不匹配
          匹配
          不匹配
          不匹配
```

以上模式表示，在每行开始，匹配任意 1 个字符，后跟 1 个字符，最后为任意 1 个字符。“^” 在正则表达式中使用频繁，因为大量的抽取操作通常在行首。

在行尾以 “\$” 匹配字符串

可以说 “\$” 与 “^” 正相反，它在行尾匹配字符串，“\$” 符号放在匹配单词后。

【实例 7.26】

假定要匹配以单词 “l” 结尾的所有的行，操作为：

```
l
匹配所有的空行。
```

具体分析为匹配行首，又匹配行尾，中间没有任何模式，因此为空行。

只匹配一个字符。

在行首和行尾之间有一个 “.”，代表任意单字符。

使用 “.” 匹配字符串中的单字符或其重复多次表达

使用 “*” 可以匹配任意字符或字符串的重复多次表达。

【实例 7.27】

匹配 compu*ter 的字符串如下：

```
computer
compuuter
compuuuter
```

匹配字符 u 一次或多次的单词是符合的。

5. 使用 “\” 屏蔽一些特殊字符的含义

有时候需要查找一些字符或字符串，而它们包含了系统指定为特殊字符的一个字符。什么是特殊字符呢？一般意义上讲，下列字符可以认为是特殊字符：

\$. ' " * [] ^ | () \ + ?

【实例 7.28】

(1) 屏蔽特殊字符 “.”。假定匹配包含字符 “.” 的行，而点代表匹配任意单字符的特殊字符，因此需要屏蔽其含义，操作如下：

```
\.
```

(2) 匹配以 “*.pas” 结尾的所有行。操作如下：

```
\*\.pas
```

这样就可以屏蔽字符 “*” 和 “.” 的特殊含义。

6. 使用 “[]” 匹配一个范围或集合

使用 “[]” 匹配特定字符串或字符串集，可以用逗号将括号内要匹配的不同字符串分开，这样做可以增加模式的可读性，但并不强制要求这样做。

使用 “-” 表示一个字符串范围，表明字符串从 “-” 左边字符开始，到 “-” 右边字符结束。

如果熟知一个字符串匹配操作，应经常使用 “[]” 模式。

【实例 7.29】

(1) 假定匹配任意一个数字，操作如下：

```
[0123456789]
```

然而通过使用 “-” 符号可以简化操作：

```
[0-9]
```

(2) 匹配任意字母。如下：

```
[A-Za-z]
```

表明从 A-Z、a-z 的字母范围。

同理，匹配任意字母或数字：

```
[A-Za-z0-9]
```

(3) 在字符序列中使用 “[]”。在字符序列中使用 “[]” 来指定字符范围，如下：

```
s[a-zA-Z]t
```

表示要匹配一个单词，以 s 开头，中间有任意字母，以 t 结尾。

- (4) 匹配包含 system 后跟句点的所有单词，这里 s 可以大小写，操作如下：

`[Ss]system\.`

- (5) 匹配所有单词。操作如下：

`[A-Za-z]*`

“[]”在指定模式匹配的范围或限制方面很有用。结合“*”与“[]”使用更是有益。

- (6) 否定或不匹配括号内容。如下：

`[^A-Za-z]`

匹配任意非字母型字符。

【注意】当“^”使用在方括号里，意思是否定或不匹配括号的内容。

7. 使用“\{\}”模式匹配结果出现的次数

使用“*”可匹配所有匹配结果任意次，但如果指定次数，就应使用“\{\}”，此模式有三种形式，即：

<code>pattern\{n\}</code>	匹配模式出现 n 次
<code>pattern\{n,\}</code>	匹配模式出现最少为 n 次
<code>pattern\{n,m\}</code>	匹配模式出现次数在 n 和 m 之间

【实例 7.30】

- (1) 匹配字母 A 出现 2 次，并以 B 结尾。

`A\{2\}B`

匹配值为 AAB。

- (2) 匹配字母 A 至少出现 4 次。

`A\{4,\}B`

可以得结果 AAAAB 或 AAAAAAB 等，但 AAAB 就不符合匹配模式。

- (3) 匹配字母 A 出现 2 次到 4 次之间。

`A\{2,4\}B`

匹配值为 AAB、AAAB、AAAAB，其他结果都不符合匹配模式。

- (4) 匹配模式要求前 4 个字符是数字，接下来是 xx，最后也是 4 个数字，操作如下：

`[0-9]\{4\}xx[0-9]\{4\}`

结果为：

1234xx7890	匹配
123axx7865	不匹配
9930xc1234	不匹配
1111xx9999	匹配

在书写正则表达式时，可能会有难度或达不到预期效果，一个好的习惯就是在写真正的正则表达式前先写下预期的输出结果。这样，当写错时，可以逐渐修改，以消除意外结果，直至返回正确值。为节省设计基本模式的时间，表 7-4 给出一些例子，这些例子并无特别顺序。

表 7-4 经常使用的正则表达式举例

表达式	含 义
^	行首
\$	行尾
^the	以 the 开头行
[Ss]ligna[IL]	匹配单词 signal、signiaL、Signal、SignalL
[Ss]ligna[IL]\.	同上，但加一句点
^USER\$	只包含 USER 的行
tty\$	以 tty 结尾的行
\.	带句点的行
^d..x..x..x	对用户、用户组及其他用户组成员有可执行权限的目录
^[^I]	排除关联目录的目录列表
.*0	0 之前或之后加任意字符
000*	000 或更多个
[iI]	大写或小写 i
[iI][nN]	大写或小写 i 或 n
^\$	空行
^.*\$	匹配行中任意字符串
^.....\$	包括 6 个字符的行
[a-zA-Z]	任意单字符
[a-z]*	至少一个小写字母
[^0-9\\$]	非数字或美元标识
[^0-9A-Za-z]	非数字或字母
[123]	1 ~ 3 中的一个数字
[Dd]evice	单词 Device 或 device
De..ce	前两个字母为 De，后跟两个任意字符，最后为 ce
^q	以 q 开始的行
^.\$	仅有一个字符的行
^\[0-9][0-9]	以一个句点和两个数字开始的行
Device	单词 Device
De[Vv]ice\.	单词 Device. 或 DeVice.

7.3.2 正则表达式的应用

在 Shell 编程中，一段好的脚本与完美的脚本间的差别之一，就是要熟知正则表达式并学会使用它们。相比较起来，用一个命令抽取一段文本比用三四个命令得出同样的结果要节省许多时间。在上一节既然已经学会了正则表达式中经常使用的基本特殊字符，又通过

一些例子简化了其复杂操作，那么现在可以看一些真正的例子了。本节主要讲述在 `grep` 和 `awk` 中使用正则表达式的例子。

1. `grep` 和正则表达式

相信 `grep` 是 Unix 和 Linux 中使用最为广泛的命令之一，`grep` 允许对文本文件进行模式查找。如果找到匹配模式，`grep` 显示包含模式的所有行。

`grep` 的选项包括：

- c: 只输出匹配的行数。
- i: 不区分大小写(只适用于单字符)。
- h: 查询多文件时不显示文件名。
- l: 查询多文件时只输出包含匹配字符的文件名。
- n: 显示匹配行及行号。
- s: 不显示不存在或无匹配文本的错误信息。
- v: 显示不包含匹配文本的所有行。

开始讨论之前，先生成一个文件，插入一段文本，并在每列后加入【Tab】键，`grep` 命令实例中将以此为例，该文件名为 `express.dat`，其文件结构如下：

- 第 1 列：城市位置编号
- 第 2 列：月份
- 第 3 列：存储代码及出库年份
- 第 4 列：产品代号
- 第 5 列：产品价格
- 第 6 列：标识号
- 第 7 列：合格数量

文件内容如下：

48	Dec	3BC1997	LPSX	68.00	LVX2A	138
483	Sept	5AP1996	USP	65.00	LVX2C	189
49	Oct	3ZL1998	LPSX	43.00	KVM9D	512
219	Dec	2CC1999	CAD	23.00	PLV2C	68
484	Nov	7PL1996	CAD	49.00	PLV2C	234
483	May	5PA1998	USP	37.00	KVM9D	644
216	Sept	3ZL1998	USP	86.00	KVM9E	234

【实例 7.31】

(1) 从文件 `express.dat` 中抽取代码为 483 和 484 的城市。

```
[root@redflag /root]#grep '48[34]' express.dat
483  Sept  5AP1996      USP      65.00    LVX2C    189
484  Nov   7PL1996      CAD      49.00    PLV2C    234
483  May   5PA1998      USP      37.00    KVM9D    644
```

利用上节中讲到使用 “[]” 来指定字符串范围，这里以 48 开始，以 3 或 4 结尾，这样抽出 483 或 484。

(2) 从文件 express.dat 中抽取行首不是 48 的行。

```
[root@redflag /root]#grep '^[^48]' express.dat
219   Dec       2CC1999       CAD       23.00       PLV2C       68
216   Sept      3ZL1998       USP       86.00       KVM9E       234
```

方括号内的^表示否定，即不是字符 4 或 8。

(3) 从文件 express.dat 中抽取以 K 开头、以 D 结尾的所有代码。

```
[root@redflag /root]#grep 'K...D' express.dat
49   Oct       3ZL1998       LPSX      43.00       KVM9D       512
483  May       5PA1998       USP       37.00       KVM9D       644
```

(4) 从文件 express.dat 中抽取城市代码为 219 或 216。

```
[root@redflag /root]#grep -E '219|216' express.dat
219   Dec       2CC1999       CAD       23.00       PLV2C       68
216   Sept      3ZL1998       USP       86.00       KVM9E       234
```

grep 命令加-E 参数，这一扩展允许使用扩展模式匹配。

(5) 从文件 express.dat 中抽取城市代码，要求第一个字符为任意字符，第二个字符为 0~5 之间，第三个字符在 0~6 之间。

```
[root@redflag /root]#grep '[0-9][0-5][0-6]' express.dat
216   Sept      3ZL1998   USP       86.00KVM9E   234
```

(6) 从文件 express.dat 中抽取数字 4 至少出现两次的行。

```
[root@redflag /root]#grep '4\{2,\}' express.dat
483   May       5PA1998       USP       37.00KVM9D   644
```

2. awk 和正则表达式

如果要格式化报文或从一个大的文本文件中抽取数据包，那么 awk 可以完成这些任务。awk 是一种自解释的编程语言，其最基本的功能是在文件或字符中按照指定规则浏览和抽取信息。awk 抽取信息后，才能进行其他文本操作。

awk 脚本由各种操作和模式组成。其命令行格式如下：

```
awk [-F field-separator] 'commands' input-file
```

在这里[-F 域分割符]是可选的，因为 awk 使用空格作为缺省的域分割符，因此要浏览域空间有空格的文本，不必指定这个选项。但如果要浏览诸如/etc/passwd 的文件，则必须指定“-F:”。如果使用“-F”选项，则 awk 每次读一条记录或一行，而 commands 是真正的 awk 命令。

一个 awk 语句中可能有多条语句。模式部分决定动作语句何时触发及触发事件。如果缺省模式部分，动作将时刻保持执行状态。模式可以是条件语句、复合语句或正则表达式。

动作在大括号{}里指明。动作大多数用来打印。

awk 执行时，其浏览域标记为“\$1,\$2,\$3.....\$n”。这种方法称为域标识。使用这些域标识将更容易对域进行进一步处理。使用\$1 表示参照第 1 域。如果需要打印一个有 5 个域的所有记录，不必指明\$1，\$2，\$3，\$4，\$5，可以使用\$0，即所有域。

下面就结合具体的实例来讨论 `awk` 的用法。开始讨论之前，先生成一个文件，插入一段文本，并在每列后加入【Tab】键，`awk` 命令实例中将以此为例，该文件名为 `student.dat`，其文件结构如下：

第 1 列：学生姓名
第 2 列：学号
第 3 列：所在城区
第 4 列：性别
第 5 列：英语成绩

文件内容如下：

Tom	116001	FuTian	M	90
John	116005	NanShan	M	85
Mary	116018	LuoHu	W	65
Steven	116030	YanTian	M	78

【实例 7.32】

(1) 从文件 `student.dat` 中抽取家住罗湖区(LuoHu)的学生。

```
[root@redflag /root]awk 'if ($3~/LuoHu/) print $0' student.dat
Mary 116018      LuoHu      W      65
```

匹配记录找到时，如果不特别声名，`awk` 缺省打印整条记录。对初学者而言，使用语句开始有点难度，但也有许多方法可以跳过它，并保持同样结果。下面就是上例的另外一种实现方法。

```
[root@redflag /root]awk ' $0 ~ /LuoHu/' student.dat
Mary 116018      LuoHu      W      65
```

(2) 从文件 `student.dat` 中抽取家不住罗湖区(LuoHu)的学生。

```
[root@redflag /root]awk ' $0 !~ /LuoHu/' student.dat
Tom      116001      FuTian    M      90
John     116005      NanShan   M      85
Steven   116030      YanTian   M      78
```

有时要浏览信息并抽取不匹配操作的记录，与“~”相反的符号是“!~”，意即不匹配。缺省情况下，`awk` 将打印所有匹配记录，因此这里不必加入动作部分。

(3) 从文件 `student.dat` 中抽取英语成绩在 80 分以上的学生，并且只打印学生的姓名。

```
[root@redflag /root]awk 'if ($5 > 80) print $1' student.dat
Tom
John
```

(4) 从文件 `student.dat` 中抽取第一个域的第四个字符是 v 的学生。

```
[root@redflag /root]awk ' $1 ~ /^...v/' student.dat
Steven   116030      YanTian   M      78
```

(5) 从文件 `student.dat` 中抽取家住福田区(FuTian)的男同学。

```
[root@redflag /root]awk 'if ($3="FuTian" && $4="M") print $0' student.dat
Tom      116001      FuTian    M      90
```

复合模式或复合操作符用于形成复杂的逻辑操作，而复合表达式则为下述各表达式相互结合起来的表达式：

&& : AND，语句两边必须同时匹配为真。

|| : OR，语句两边同时或其中一边匹配为真。

! : 非，求反。

7.4 Shell 编程综合实例

7.4.1 实例一

该实例的功能是按照/etc/hosts 文件中的条目逐一 ping 所有的机器。该程序的源代码如下：

```
#!/bin/bash
#pingall
#grab /etc/hosts and ping each address
cat /etc/hosts | grep -v '^#' | while read LINE
do
    ADDR=`awk '{print $1}'`
    for MACHINE in $ADDR
    do
        ping -s -c 1 $MACHINE
    done
done
#end
```

该实例列出/etc/hosts 文件并查找其中的非注释行(不以“#”开头的行)。然后使用一个 while 语句循环读入所有的行，接下来使用 awk 分析出每一行的第一个域，并把它赋给变量 ADDR。最后使用 for 循环逐一 ping 相应的地址。

7.4.2 实例二

该实例的功能是显示当前使用系统的硬件信息。该程序的源代码如下：

```
#!/bin/sh
#HINV for linux v1.1
# written by Dino dino@brownnut.com
#to include in another version
# GNU GENERAL PUBLIC LICENSE
# tested on RedFlag and Debian sofar.
#fixed AMD processor detect in this v1.1
#
clear
```

```
verbose=0
help=0
if [ "$1" = "-v" ];then
verbose=1
fi
if [ "$1" = "-h" ];then
echo "hinv {-v|-h}"
echo "-v = verbose"
echo "-h =help"
exit
fi
mach=`uname -m`
mem=`cat /proc/meminfo | awk '/^MemTotal/ {print $2}`
proc=`cat /proc/cpuinfo | awk '/^processor/' |grep -c processor`
echo "Total Processors:$proc"
echo "$mach          :Processor"
egrep -i "vendor_id|name|MHz|cache" /proc/cpuinfo
echo ""
#echo "Main Memory Size:$mem Mbytes"
echo "Main Memory Size:$mem Kbytes"
if [ -r /proc/ide/ide0/hda/model ];then
echo ""
echo "Host:ide0 Channel:hda"
cat /proc/ide/ide0/hda/model
fi
if [ -r /proc/ide/ide0/hdb/model ];then
echo ""
echo "Host:ide0 Channel:hdb"
cat /proc/ide/ide0/hdb/model
fi
if [ -r /proc/ide/ide1/hdc/model ]; then
echo ""
echo "Host:ide Channel:hdc"
cat /proc/ide/ide1/hdc/model
fi
if [ -r /proc/ide/ide1/hdd/model ]; then
echo ""
echo "Host:ide Channel:hdd"
cat /proc/ide/ide1/hdd/model
fi
if [ -r /proc/scsi/scsi ];then
```

```
echo ""
egrep "Host|Vendor" /proc/scsi/scsi
fi
echo ""
echo "Serial Ports:"`egrep -c serial /proc/ioproports`
echo "Keyboard Detected:"`egrep -c keyboard /proc/ioproports`
echo "Ethernet Controllers:"`/sbin/ifconfig | awk '/^eth/ {print $1}`"
echo ""
egrep controller /proc/pci
if [ $verbose -ne 0 ]; then
egrep "bridge" /proc/pci
fi
```

该程序的执行结果如下：

```
Total Processors:1
i686           :Processor
vendor_id      : GenuineIntel
model name     : Celeron (Mendocino)
cpu MHz       : 367.504
cache size    : 128 KB

Main Memory Size:158248 Kbytes

Host:ide0 Channel:hda
SAMSUNG SV0432A

Host:ide Channel:hdc
SAMSUNG SCR-3232

Serial Ports:2
Keyboard Detected:1
Ethernet Controllers:eth0

Ethernet controller: Realtek Semiconductor Co. , Ltd. RTL-8139 (rev 16).
VGA compatible controller: Silicon Integrated Systems [SiS] 86C326 (rev 11).
```

本章小结

Shell 是语言解释器，是系统中最重要程序，也是系统管理员必须熟练掌握的内容。本章主要通过大量的实例介绍了 Shell 基本概念、Shell 语法、正则表达式等内容。随着对 Shell 理解的深入，相信读者会写出自己满意的、功能强大的脚本，为系统管理提供更加快捷和方便的手段。

习 题

1. 在 bash 中普通用户用_____作为默认的提示符。
A. \$ B. # C. @ D. ?
2. 表示追加输出重定向的符号是_____。
A. > B. >> C. < D. <<
3. 表示管道的符号是_____。
A. || B. | C. >> D. //
4. 下列符号中, _____不是 Shell 环境下正则表达式的元字符。
A. \$ B. ^ C. * D. @
5. 在_____Shell 环境中, 使用如下的变量赋值方式: set variable=5。
A. bash B. pdksh C. tcsh D. ksh
6. Bourne Shell 的程序名为_____。
A. Bash B. zsh C. rc D. sh
7. 输入一个命令之后, Shell 首先检查_____。
A. 它是不是外部命令 B. 它是不是在搜索路径上
C. 它是不是一个命令 D. 它是不是一个内部命令
8. 关于 Linux 的 Shell 说法错误的是_____。
A. 一个命令语言解释器 B. 编译型的程序设计语言
C. 能执行内部命令 D. 能执行外部命令
9. 为匹配以 001 开头的行, 我们可以使用如下的正则表达式是_____。
A. ^001 B. \$001 C. *001 D. \001
10. 下面_____不是 Shell 的循环控制结构。
A. for B. switch C. while D. until
11. 与正则表达式[^a-z]匹配的表达式有_____。
A. a B. 9 C. * D. %
12. 与正则表达式[123]匹配的表达式有_____。
A. 1 B. 2 C. 23 D. 123
13. 下述正则表达式意思相同的有_____。
A. [0 1 2 3 4 5] B. [0-5] C. [0 1-5] D. [0-2 3-5]
14. 与正则表达式[C c]omputer 相匹配的项有_____。
A. [C c]omputer B. Computer C. computer D. Ccomputer
15. 与正则表达式 compu*t 匹配的单词有_____。
A. comput B. comput C. computat D. computt

第三篇



网络管理

第 8 章 网络文件系统 NFS



计算机网络发展的目的是资源共享,资源共享最常见形式就是文件的共享,Linux 理所当然要有这样的功能。在 Linux 的主机上可以使用另一个 Linux 主机上的文件夹和文件,当然我们要做适当的控制,本章讨论如何做到 Linux 主机间的文件共享。

8.1 NFS 基本原理

8.1.1 什么是 NFS(Network File System)

在 Windows 主机之间可以通过共享文件夹来实现存储远程主机上的文件,而在 Linux 系统间通过网络文件系统(NFS)可以实现类似的功能。然而 NFS 和 Windows 的共享还是有差别的,它和 Windows 2000 Server 的分布式文件系统更为类似。我们先来看一个例子:Linux 主机 A 有一个目录/test,我们可以先把它导出(类似把它共享出来),然后我们在 Linux 主机 B 上把 A:/test 安装(挂接)到安装点/mnt/nfs 下,这个挂接和挂接本地磁盘一样。这样在主机 B 上的用户就可以使用/mnt/nfs 下的文件,而实际上用户使用的是主机 A 上的目录/test。也就是说,用户可以以访问本地文件的方式访问远程主机上的文件。

NFS 由 SUN 公司开发,它最终被 IETF 所接受,纳入 RFC 成为一种文件服务标准,是分布式计算机系统的一个组成部分。网络文件系统有以下优点:

- (1) 被所有用户访问的数据可以存放在一台中央的主机上,其他不同主机上的用户可以通过 NFS 访问同一中央主机上的数据;
- (2) 客户访问远程主机上的文件是透明的,和访问本地主机上的文件是一样的;
- (3) 远程主机上文件的物理位置发生变化(如从一台主机移动到另一主机上)也不会影响客户访问方式的变化。

8.1.2 NFS 的工作原理

NFS 是基于客户/服务器模式的。NFS 服务器是输出一组文件的计算机,而客户是访问文件的计算机。客户和服务器通过远程过程调用(RPC, Remote Procedure Call)通信,当客户主机上的应用程序访问远程文件时,客户主机内核向远程服务器发送一个请求,客户进程被阻塞,等待服务器应答,而服务器一直处于等待状态,如果接收到客户请求,就处理请求并将结果返回客户机。NFS 服务器上的目录如果被远程用户访问,就称为“导出”(export);客户主机访问服务器导出目录的过程称为“安装”(mount),有时也称“挂接”或“导入”。

NFS 由许多组件共同协作完成，如图 8-1 所示。

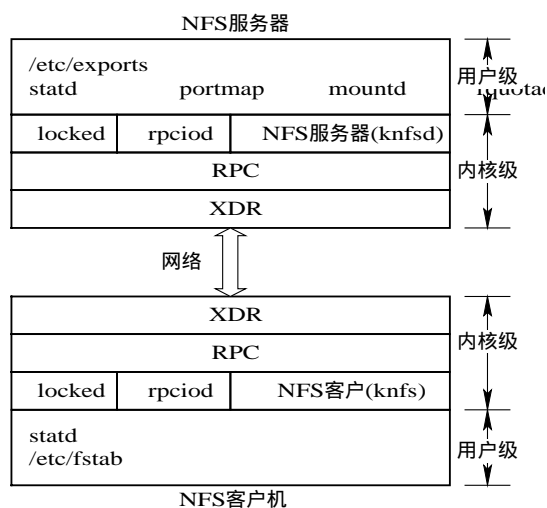


图 8-1 NFS 组件

XDR(外部数据表示)：在异构机器组成的网络上交换数据。

RPC 协议：负责定义客户机和服务器之间的信息格式，NFS 请求是以 RPC 包的形式发送的。

`locked`：RPC 锁监控程序，实现 NFS 锁管理器(NLM，NFS Lock Manager)协议，NLM 协议支持多个客户通过 NFS 一致性地锁定文件。

`rpciod`：NFS I/O 监控程序。

`knfsd`：是 NFS 系统的核心，监听远程主机的 RPC 请求、根据 NFS 协议进行解释，它用 RPC 将响应返回客户机，并与 NFS 服务器上运行的其他组件通信如 `rpciod`、`rpc.locked` 和 `rpc.statd`。

`statd`：NFS 状态监控程序，利用网络状态监视协议(NSM)维护 NFS 锁的状态。

`portmap`：将 RPC 程序号与运行服务的实际端口相对应。

`mountd`：处理客户机的 mount 请求。

`rquotad`：提供 NFS 系统中用户的配额信息。

由于 NFS 有明确的服务器和客户角色之分，因此 NFS 的配置包括两个部分：NFS 服务器的配置和 NFS 客户机的配置。

8.2 配置 NFS 服务器

NFS 服务器的配置步骤如下：

- (1) 安装 NFS 软件包。
- (2) 在 `/etc/exports` 文件中配置 NFS 服务器上要导出的文件系统或目录。
- (3) 启动 NFS 服务。
- (4) 导出 `/etc/exports` 中配置的文件系统或目录。

8.2.1 安装 NFS

要使用 NFS 服务器或安装 NFS 服务软件包，可以用红旗 3.0 版的 CD-ROM 中的目录 /Redflag/RPMS 下的软件包“nfs-utils-0.3.1-8.i386.rpm”。安装之前请用以下命令检查该软件包是否已经安装：

```
[root @redflag /root]#rpm -qa | grep nfs-utils
nfs-utils-0.3.1-8
```

以上命令表明软件包已经安装，如果没有安装则用以下命令安装软件包(当然要先把 CD-ROM mount 上)。

```
[root @redflag /root]#mount /dev/cdrom
[root @redflag /root]#cd /mnt/cdrom/RedFlag/RPMS
[root @redflag /root]#rpm -ivh nfs-utils-0.3.1-8.i386.rpm
```

8.2.2 配置导出文件:/etc/exports

/etc/exports 文件是用来告诉 Linux 系统哪些文件系统或目录将为 NFS 客户导出，这个文件是一个文本文件，通常由管理员编辑。文件中以“#”号开头的行被忽略，长行可以用“\”分解为多行。语法如下：

```
dir_name host1(opt1 , opt2 , ..... ) host2(opt1 , opt2 , ..... ) \
host3(opt2 , ..... )
```

【实例 8.1】

```
/home linux2.wlj.com(ro)
/test *(rw)
```

在/etc/exports 文件中的每个项目以目录名开始，目录名 dir_name 就是要导出给 NFS 客户使用的目录，目录可以是某一文件系统的根，也可以不是文件系统的根。目录名之后可以列出多个或零个用空格分隔开的导出指定项。每个导出指定项由两个可选部分组成：主机指定部分(host)和导出选项部分(opt)，这两个部分都是可选的。如果只列出主机指定部分而没有列出导出选项部分，则导出选项缺省为“ro”——只读，即 NFS 客户只能读取这个目录的文件而不能写入目录中的文件。如果只指定导出选项而没有指定主机，则主机缺省为所有主机。如果目录后没有指定任何导出项，则以只读选项导出目录给所有主机，相当于*(ro)。

【注意】 由于导出指定项用空格分隔，因此主机部分和导出选项之间不能有空格。如：
/home linux2.wlj.com(ro)和 /home linux2.wlj.com (ro)有很大的差别，后者相当于：/home linux2.wlj.com(ro) *(ro)。

主机部分非常灵活，可以指定单个主机，也可以指定多个主机。描述主机的方法有四种。

(1) 单个主机：可以列出短名、完全限定名或 IP 地址，如：

```
linux2 或 linux2.wlj.com 或 192.168.0.1
```

(2) 主机网络：指定特定一个子网或几个子网上的主机，用 address/netmask 语法指定，如：

192.168.0.0/255.255.255.0 表明导出文件系统或目录给 192.168.0.0 子网上的所有主机使用。

(3) 通配符主机：可以使用“?”、“*”对主机名进行匹配，如：

“*.wlj.com”可以匹配 wlj.com 域中的所有主机。但是要注意的是通配符不匹配主机名中的“.”，因此“*.wlj.com”不匹配 server1.test.wlj.com。

(4) 网组：可以列出 NFS 网组映射中定义的整组主机，网组以“@”开头，例如：

@linuxgrp。

导出选项列表用“.”分隔开，选项之间也不能包含空格。表 8-1 列出了导出选项的含义，导出选项分为性能选项和安全选项两种类型。

表 8-1 NFS 导出选项

选 项	是否缺省	类型	含 义
async	是	性能	异步将数据写入磁盘(不是在客户机请求时写入)
sync	否	性能	客户机执行写操作时立即将数据写入磁盘
wdelay	是	性能	延迟同步写入，实现累积
no_wdelay	否	性能	不延迟同步写入
ro	是	安全	允许对这个文件系统进行只读操作
rw	否	安全	允许对这个文件系统进行读写操作
root_squash	是	安全	将 UID=0 的用户(root)映射为用户 nobody
no_root_squash	否	安全	不将 UID=0 的用户(root)映射为用户 nobody
all_squash	否	安全	将所有的用户映射为用户 nobody
no_all_squash	是	安全	不将所有的用户映射为用户 nobody
anonuid=N	-2	安全	将匿名账户 nobody 的 UID 设为 N
anongid=N	-2	安全	将匿名账户 nobody 的 GID 设为 N
secure	是	安全	允许 1024 以下端口产生的请求
insecure	否	安全	允许 1024 以上端口产生的请求
subtree_check	是	安全	认证文件句柄属于整个文件系统的导出子树
no_subtree_check	否	安全	不认证文件句柄属于整个文件系统的导出子树
hide	是	安全	不导出这个目录下挂接的其他文件系统
nohide	否	安全	导出这个目录下挂接的其他文件系统

【实例 8.2】

/home linux2.wlj.com(ro)

将使得 linux2.wlj.com 主机上的用户对/home 目录有只读权限。

【实例 8.3】

/test *.wlj.com(rw)

将使得 wlj.com 域上的主机上的用户对/test 目录有读写权限。

导出目录时将面临一个问题：NFS 服务器用户和 NFS 客户机上用户的映射。假如在 NFS 服务器上导出/test 目录，该目录由 NFS 服务器上的用户 user1 和 user2 拥有，则 NFS 客户机上有同样的用户 user1 和 user2 才能使用导出的目录。另外，我们可能不希望客户机上的 root 用户拥有 NFS 服务器上的 root 用户对/test 目录的权限。这时我们可以使用导出选项控制 NFS 客户机和 NFS 服务器的用户映射，从而达到控制权限的目的。“root_squash”选项（root 用户挤压）是默认选项，该选项的作用是将客户机上的 root 用户映射到 NFS 服务器上的 nobody 用户，这时 NFS 客户机上的 root 用户对 NFS 服务器上导出的目录的权限只具有服务器上 nobody 用户的权限。

【注意】“no_root_squash”取消客户机上的 root 用户到服务器上的 nobody 用户的映射，而映射为 NFS 服务器上的 root 用户，因此客户机上的 root 用户权限相当大，要小心使用该选项。

如果希望将客户机上的 root 用户映射到服务器上的其他用户或组，可以使用 anonuid 和 anongid 选项。

【实例 8.4】

```
/test linux2.wlj.com(anonuid=500,anongid=600)
```

本例中客户机上的 root 用户映射到服务器上 UID=500 的用户，同时组 GID=600，意味着客户机上的 root 用户具有服务器上 UID=500 的用户和 GID=600 的组的权限。

“all_squash”和“no_all_squash”选项的含义和“root_squash”、“no_root_squash”选项的含义类似。只不过“all_squash”是将客户机上的所有用户映射到服务器上的 nobody 用户，如果要映射成服务器上的其他用户，同样要使用 anonuid 和 anongid 选项。“no_all_squash”是缺省选项，意味着缺省时客户机上的用户要和服务器上的用户一一对应，才能访问目录。

【实例 8.5】

```
/test linux2.wlj.com(all_squash,anonuid=400,anongid=700)
```

本例中把客户机上的所有用户（含 root 用户）映射到服务器上 UID=400 的用户，GID=700 的组。

【注意】“root_squash”和“no_all_squash”是缺省值。缺省情况下客户机上的 root 用户具有服务器上 nobody 用户的权限，而客户机上的其他用户映射到服务器上同名的用户，它们具有与服务器上同名用户的权限。因此在客户机上，root 用户可能比其他用户的权限还小。

【提示】如果在 NFS 服务器上把 CD-ROM 安装在/mnt/cdrom，并把/mnt/cdrom 导出，就可以把光驱共享出来使用了。

表 8-1 的其他选项放在稍后的章节介绍，我们先激活服务器上被导出的目录。

8.2.3 激活 NFS

NFS 服务的启动和停止是通过/etc/rc.d/init.d 目录下的脚本 nfs 来实现的，执行该脚本必须用 root 用户登录。

【实例 8.6】

```
/etc/rc.d/init.d/nfs start
```

```
Starting NFS services: [ OK ]
```

Starting NFS quotas: [OK]

Starting NFS mountd: [OK]

Starting NFS daemon: [OK]

【实例 8.7】

```
/etc/rc.d/init.d/nfs stop
```

Shutting down NFS mountd: [OK]

Shutting down NFS daemon: [OK]

Shutting down NFS services: [OK]

Shutting down NFS quotas: [OK]

【提示】 也可以用 `service nfs start` 来启动 NFS 服务。

如果想让 Linux 系统在启动时同时也启动 NFS 服务，可以使用 `ntsysv` 命令，如图 8-2 所示。找到 NFS 服务，用空格键在 NFS 处做标记“*”，确认“Ok”即可。

【实例 8.8】

[root @redflag /root]#ntsysv(见图 8-2)。



图 8-2 NFS 的自动启动

【提示】 `ntsysv` 不仅可以用来设置 NFS 服务的自动启动，也可以用于设置其他服务的自动启动，如 `httpd` 等。

启动 NFS 服务后，可以使用 `rpcinfo` 命令进行检查。

【实例 8.9】

```
[root @redflag /root]#rpcinfo -p
```

program	vers	proto	port	
100000	2	tcp	111	portmapper
100000	2	udp	111	portmapper
100024	1	udp	32768	status
100024	1	tcp	32768	status
100011	1	udp	635	rquotad
100011	2	udp	635	rquotad
100005	1	udp	32775	mountd

100005	1	tcp	32773	mountd
100005	2	udp	32775	mountd
100005	2	tcp	32773	mountd
100005	3	udp	32775	mountd
100005	3	tcp	32773	mountd
100003	2	udp	2049	nfs
100003	3	udp	2049	nfs
100021	1	udp	32776	nlockmgr
100021	3	udp	32776	nlockmgr
100021	4	udp	32776	nlockmgr

如果 rquotad、mountd、nfs、nlockmgr 等项存在，则说明 NFS 服务已经在运行。

【注意】 如果/etc/exports 文件中没有任何导出指定项，则使用 nfs start 命令没有任何输出，同样使用“rpcinfo -p”命令时输出中也没有 mountd、nfs、nlockmgr 等项存在，所以要先配置 exports 文件，才启动 NFS 服务。

8.2.4 导出目录：exportfs

NFS 服务启动时会自动导出/etc/exportfs 文件中设定的文件系统或目录，但是如果在 NFS 服务启动后修改了 exports 文件，要使用 exportfs 命令导出目录。exportfs 命令的用法如下：

```
/usr/sbin/exportfs [-avi] [-o options,...] [client:/path..]
/usr/sbin/exportfs -r [-v]
/usr/sbin/exportfs [-av] -u [client:/path ..]
/usr/sbin/exportfs [-v]
```

参数选项:

- a：对所有的目录操作。
- o options：指定目录导出时的选项，选项的定义和 exports 文件的定义相同。
- i：忽略/etc/exports 文件列出的信息，取命令行中指定的导出选项。
- r：再次导出所有的目录，用于重新初始化 NFS 服务器内部的导出清单。
- u：从 NFS 服务器中删除导出的目录。
- v：显示 exportfs 命令执行时的信息。

【实例 8.10】

```
[root @redflag /root]#cat /etc/exports
/home *(ro)
/test *(rw)
[root @redflag /root]#exportfs -av
exporting */test
exporting */home
```

导出所有目录。

【实例 8.11】

```
[root @redflag /root]#exportfs -rv
```

重新导出所有目录，如果在/etc/exports 文件中增加或删除了某项，可以使用该命令。这是用得最多的例子。

【实例 8.12】

```
[root @redflag /root]#exportfs -v -i -o rw,no_root_squash *.wlj.com:/test2
exporting *.wlj.com:/test
```

把 NFS 服务器上的/test2 目录导出给 “*.wlj.com” 客户机使用，导出权限为 “rw, no_root_squash”。我们使用 “-i” 选项让 exportfs 命令忽略/etc/exports 文件。这通常用于临时将一个目录导出，而不将其加入到/etc/exports 文件中。

【实例 8.13】

```
[root @redflag /root]#exportfs -vu *.wlj.com:/test
unexporting *.wlj.com:/test
unexporting */test
```

取消/test 目录的导出。

【实例 8.14】

```
[root @redflag /root]#exportfs
/home          *
/test          *
```

不带任何参数的 exportfs 命令显示当前 NFS 服务器已经导出的目录。

8.3 配置 NFS 客户

Linux 下的 NFS 客户配置非常容易，不需要加载任何新的或附加的软件。惟一的要求是内核必须编译为支持 NFS 的功能或作为可装入模块。缺省情况下，红旗 Linux 3.0 将 NFS 作为可装入模块，在需要时自动装入。也可以使用命令 insmod nfs 手工装入模块，装入后可使用 lsmod 命令检查。

【实例 8.15】

```
[root @redflag /root]#lsmod

Module              Size  Used by    Not tainted
nfsd                 77216   0      (autoclean)
nfs                  86716   0
lockd                57280   0      [nfsd nfs]
sunrpc               77076   0      [nfsd nfs lockd]
8139too              20128   1      (autoclean)
usb-uhci             25636   0      (unused)
usbcore              65184   1      [usb-uhci]
```

如果输出中有 “nfs” 项(注意：不是 “nfsd” 项)，则说明 “nfs” 已经作为可装入模块加载了。

```
[root @redflag /root]#cat /proc/filesystems
```

```
nodev    rootfs
nodev    bdev
nodev    proc
nodev    sockfs
nodev    tmpfs
nodev    shm
nodev    pipefs
          ext2
          iso9660
          reiserfs
nodev    devpts
nodev    usbdevfs
nodev    nfs
```

输出中也应该有“nfs”项。

NFS 软件包(nfs-utils-0.3.1-8.i386.rpm, 我们前面已经介绍过它, 它是 NFS 服务器软件包)带有一个称为 showmount 的程序, 该程序能够给出 NFS 服务器导出的目录信息。showmount 命令的格式如下:

```
showmount [参数选项] NFS 服务器主机名/IP 地址
```

参数选项:

-e: 显示 NFS 服务器上导出的目录。

【实例 8.16】

```
[root @redflag /root]#showmount -e 192.168.0.15
```

```
Export list for 192.168.0.15:
```

```
/test *
```

```
/home *
```

显示 NFS 服务器 192.168.0.15 导出的目录, 当然我们也可以在 NFS 服务器上执行不带任何参数的 exportfs 命令来查看导出的 NFS 目录, 结果是一样的。

如果已经知道了从 NFS 服务器上导出的目录信息, 就可以使用我们在文件系统这一章中已经介绍过的 mount 命令手工安装(挂接)NFS 服务器上的目录了。挂接 NFS 目录时, mount 命令有几个小的变化: 文件系统的类型指定为“nfs”、分区为远程 NFS 服务器上的导出目录和有一些挂接选项。例如:

```
mount -t nfs -o options nfsserver:/dir dir_mountpoint
```

【实例 8.17】

```
[root @redflag /root]#mount -t nfs -o rw linux1.wlj.com:/test /mnt/nfs
```

把 NFS 服务器上 linux1.wlj.com 导出的目录/test 挂接在本机的/mnt/nfs 目录下, 权限为读写。

NFS 客户机安装 NFS 分区(目录)时的选项很多, 除了以前介绍 mount 命令时介绍的所有文件系统都具有的一般选项外, 还有一些只适用于 NFS 分区的选项。挂接 NFS 分区(目

录)时,可以混合使用一般选项和 NFS 分区特有的选项。常用的 NFS 特有的选项见表 8-2,其余的选项参见 man nfs 命令的输出。

表 8-2 NFS 客户特有的挂载选项

选项	缺省值	含 义
bg	关	后台挂载
fg	开	前台挂载
intr	关	定义一个可中断的挂载
soft	关	软挂载
hard	开	硬挂载
rsize=N	1024	从 NFS 服务器读取数据的字节数
wsiz=N	1024	向 NFS 服务器写入数据的字节数
timeo=N	7	重传 RPC 请求之前等待的 1/10 秒数
retrans=N	3	RPC 请求失败前超时和重传的次数

使用后台挂载时,如果第一次挂载失败(例如 NFS 服务器宕机),挂载进程就会在后台处理,直到成功为止。采用“bg”选项可以避免 NFS 服务器宕机而使客户机 Linux 系统悬挂在 mount 命令上。特别是在/etc/fstab 文件中自动挂载 NFS 分区时,最好采用“bg”选项,否则可能导致 Linux 系统无法启动。

如果服务器 A 在/etc/fstab 挂载了服务器 B 的 NFS 分区,同时服务器 B 也在/etc/fstab 挂载了服务器 A 的 NFS 分区,这时出现了“交叉挂载”,两台主机既是做为服务器为对方服务,也同时是对方的客户。交叉挂载会带来一个问题:服务器 A 必须等到服务器 B 完成引导才能正常引导,而服务器 B 也必须等到服务器 A 完成引导才能正常引导,从而陷入一个死循环。这种情况下应确保在两台主机的/etc/fstab 文件中将 NFS 分区用“bg”选项来挂载。

NFS 客户机挂载 NFS 分区时,缺省采用“硬”挂载(即采用“hard”选项),即它们会无休止地尝试连接 NFS 服务器直到连接上为止。这种做法并不总是最好的,在需要对整个系统进行关机时可能会引起故障。如果 NFS 服务器关机比 NFS 客户机关机早,客户机的关机操作会无法正常进行。这种情况下可以采用“软”挂载(即采用“soft”选项),NFS 客户机在重试一定次数(用 retrans=N 参数设定重试次数)后会终止连接。但是数据操作执行完成之前不得把控制权交给应用程序时(如电子邮件目录),就不要使用“软”挂载选项。

当 NFS 客户机上某个进程进行系统调用时,内核就会接过动作,在内核处理系统调用的时间里,原来的进程对自己是没有控制权的。如果内核的存取操作出现问题,该进程就必须继续等待,直到内核调用返回为止,进程自己是不能放弃操作并退出运行的。“intr”选项给挂载加上一个可中断的标志,允许等待 NFS 操作的进程放弃操作并退出。缺省时不使用“intr”选项。

在 NFS 客户机上挂载 NFS 服务器上的 NFS 分区同样可以在/etc/fstab 文件中配置,使得客户机的 Linux 系统启动时自动挂载服务器上的 NFS 分区。

【实例 8.18】

在/etc/fstab 中添加:


```
linux1.wlj.com:/test /mnt/nfs nfs rw,bg,intr,soft 0 0
```

自动把 NFS 服务器 linux1.wlj.com 上的导出目录/test 挂接在/mnt/nfs 目录下, 挂接选项是: “rw”, “bg”, “intr”, “soft”。

8.4 NFS 的性能、安全和故障排除

8.4.1 NFS 的性能

NFS 通过网络访问远程文件, 因此速度比访问本地磁盘慢。由于 NFS 有客户机和服务器两个部分, NFS 的性能也就和双方的参数配置都有关。

1. NFS 服务器的性能选项

首先要强调的是 NFS 性能和 Linux 内核有很大的关系, 调整内核也可改善 NFS 服务器的性能, 特别是把 NFS 编译到内核, 而不是作为模块装入。有关内核的编译请参见有关 Linux 内核的资料, 我们这里主要介绍导出选项对 NFS 服务器性能的影响。

NFS 客户机上的用户执行 write 系统调用时, 它希望系统调用顺利返回时数据已经写入到 NFS 服务器上的磁盘, 这样可以保证数据的可靠性。如果 write 系统调用后 NFS 服务器崩溃, 则已经写入到磁盘的数据还是安全的, 这种方式称为同步写入。同步写入会降低性能, 因为用户进程必须等待系统调用的返回。与之相反的另一种方法是 NFS 服务器并不立即把数据写到磁盘而是放在缓冲区, 然后立即返回调用, 这就是异步写入。NFS 服务器上的 Linux 系统在调用完成后的一段时间(5 秒左右)内把数据写入磁盘, 如果在这段时间里 NFS 服务器崩溃, 将会导致数据的丢失。“nfs” 导出选项 “sync” 是同步写入, “async” 是异步写入, 缺省是 “async”。

【提示】可靠性和性能在此不能两全, 如果要保证数据的可靠性而不惜影响性能, 可以在/etc/exports 文件中加上 “sync” 选项。

NFS 服务器还有另一个特性和可靠性、性能有关。Linux 使用基于页的虚拟内存, 它可以把同一物理数据页的几个小的写入集合起来一次性写入, 以减少写操作的次数。“wdelay” 选项允许 Linux 系统延迟写入, 把几次写入操作集合起来。“nowdelay” 选项则不允许 Linux 系统延迟写入, 不允许集合操作。写入延迟可以提高性能, 但同样也增加数据丢失的可能性, 如果 NFS 服务器在把集合起来的数据写入到磁盘之前崩溃, 则所有这些数据全部丢失。“wdelay” 选项是缺省值。

【提示】同样, 如果要保证数据的可靠性而性能可以差些, 可以在/etc/exports 文件中加上 “no_wdelay” 选项。

2. NFS 客户机的性能选项

NFS 客户机上的 NFS 性能是由挂接 NFS 文件系统时(mount 或/etc/fstab)采用的选项控制的。NFS 客户机也有类似的 “sync” 和 “async” 选项(它们不是 NFS 特有的选项), 注意它们不是用来控制 NFS 服务器的性能。NFS 客户机缓冲用户进程写入的数据时, 是将这些数据保留在 NFS 客户机的缓冲区中, 即这个数据不从网络立即传送给 NFS 服务器。“async” 是缺省值。

NFS 客户机按固定最大长度的单元从网络上读取和写入数据，这分别是“`rsize`”和“`wsize`”选项指定的值。读取和写入长度越大，NFS 系统的性能越好。但是 NFS 建立在 RPC 上，而 RPC 使用 XDR，XDR 又使用 IP 网络组件。大的 NFS 传输单元会被分解成多个小组，通常是 1500 字节左右(在以太网上)。如果使用 NFS 和 UDP，当 NFS 传输单元中有一分组丢失时，即使其他分组安全到达，也会导致 NFS 重发整个单元，这样就增加了网络的负担。如果线路的情况不好，可能出现反复重发，加剧恶化网络的通信状况。因此需要平衡网络的性能和 NFS 的性能，如果网络快而稳定，“`rsize`”和“`wsize`”的值可以加大，高达 32 KB；如果网络慢而不稳定(如拨号网络)，“`rsize`”和“`wsize`”的值可降至 1 KB、2 KB。

8.4.2 NFS 的安全

1. NFS 服务器的安全

保护 NFS 服务器是 NFS 服务安全的最重要部分，保护方式主要有两种：RPC 访问安全和目录的导出权限。

Linux 的 TCP Wrapper 根据两个配置文件确定谁能访问哪些服务，这两个文件是 `/etc/hosts.allow` 和 `/etc/hosts.deny`。这两个文件很相似，格式如下：

服务 1，服务 2，.....：主机名 1，主机名 2，.....

TCP Wrapper 程序在用户访问主机的服务时，按以下的顺序进行操作：

- (1) 如果客户机的主机名或 IP 地址匹配 `/etc/hosts.allow` 列出的服务名，则客户的访问被允许；
- (2) 如果客户机的主机名或 IP 地址匹配 `/etc/hosts.deny` 列出的服务名，则客户的访问被拒绝；
- (3) 如果客户机的主机名或 IP 地址在两个文件均不匹配，则客户的访问被允许。

【提示】缺省下，安装 Linux 系统时 TCP Wrapper 软件是安装的，可以使用命令“`rpm -q tcp_wrappers`”检查软件包是否安装。同时 `/etc/hosts.allow` 和 `/etc/hosts.deny` 文件都为空，因此任何客户均可以访问任何服务。

【实例 8.19】

```
[root @redflag /root]#cat /etc/hosts.deny
ALL:ALL

[root @redflag /root]#cat /etc/hosts.allow
portmap:192.168.0.
lockd:192.168.0.
nfsd:192.168.0.
mountd,rquotad:192.168.0.
stad:192.168.0.
```

以上例子除了开放 NFS 相关的服务给 192.168.0.* 的主机外，关闭了其他所有的服务(包括了 Web、FTP、DNS 等服务)。这里采用的是先关闭所有的服务再一一开放允许的服务，安全性较好，如果要开放某一服务可以在 `/etc/hosts.allow` 加入相应的命令行。

NFS 服务器的配置文件 `/etc/exports` 中有许多选项是用于控制安全性，我们已经介绍了“`root_squash`”、“`no_root_squash`”、“`all_squash`”、“`no_all_squash`”、“`anonuid`”和“`anongid`”，

这里补充其他的安全选项。NFS 服务器缺省要求 NFS 客户主机用 1024 以下的端口来发送请求，这些端口称为安全端口，root 用户使用这些端口，非 root 用户可能使用 1024 以上的端口(不安全端口)发出请求。因此可能需要用“insecure”选项使得 NFS 客户可以使用 NFS 服务，特别是非 Linux 系统的 NFS 客户。

NFS 服务器导出某一目录时，这一目录下可能还挂接着其他文件系统。为了在 NFS 客户机上能够使用这两个目录，通常会在 NFS 服务器上的/etc/exports 文件分别有两个目录的导出指定项，然后在 NFS 客户机上分别挂接，如果只挂接上一级目录，下一级目录会被隐藏。“hide”选项和“nohide”选项可以控制导出目录时其他文件系统是否一起导出。“hide”选项是缺省值，即不显示被导出目录下挂接的其他文件系统，避免出现死循环。

【实例 8.20】

/test 目录下有一目录/test/win2000，它是 Windows 2000 文件系统的挂接点，如果我们导出目录时采用：

```
/test *(ro)
```

```
/test/win2000 *(ro)
```

在 NFS 客户机上只挂接/test，这时无法在/test 目录下看到/test/win2000 目录的内容。我们可以使用“nohide”选项导出：

```
/test *(ro)
```

```
/test/win2000 *(ro,nohide)
```

在 NFS 客户机上只挂接/test，这时在/test 目录下就可以看到/test/win2000 目录的内容了。

“subtree_check”选项用于控制 NFS 服务器证明文件句柄确实属于所导出的文件系统，防止客户机创建伪 NFS 文件句柄。假设 Linux NFS 服务器有一个文件系统安装在/test，其下有目录/test/proj。我们将/test/proj 用“subtree_check”选项导出，每次 NFS 客户机向 NFS 服务器发送 NFS 操作时，服务器保证收到的文件句柄属于/test/proj 目录或其子目录。如果客户主机伪造 NFS 文件句柄(该句柄实际上是指向/test/doc/hetong1.doc)后发送请求给服务器，NFS 服务器会检查句柄并拒绝服务。“subtree_check”选项是缺省值，“no_subtree_check”选项可以关掉子树检查。如果导出整个文件系统，可以采用“no_subtree_check”选项加速对文件系统的访问，因为文件系统不会有其他的部分被访问的危险。

2. NFS 客户机的安全性

在 NFS 客户机挂接 NFS 分区时，“rw”和“ro”选项可以控制 NFS 文件系统的访问权限。缺省时，NFS 客户机挂接分区时采用“rw”，而 NFS 服务器导出目录时缺省采用的是“ro”。因此如果所有的选项保持缺省值，则 NFS 服务器不让写入导出卷，当 NFS 客户机想写入导出卷时会出现错误。当然如果 NFS 服务器上以“rw”选项导出目录，NFS 客户机以“ro”选项挂接 NFS 分区，最终 NFS 客户机上的用户也只具有读的权限。

8.4.3 NFS 故障排除

1. IP 地址的问题

在 NFS 服务器上的/etc/exports 文件配置导出目录时，主机指定如果采用主机名的方法，

要保证 NFS 客户机的 IP 地址和主机名符合，否则会遭到拒绝。NFS 服务器获得连接到自己的客户机 IP 地址后，试图解析为完全授权域名 FQDN，但是如果在 /etc/exports 文件里列出的机名不完整，NFS 服务器将拒绝服务。如，服务器认为是 linux2.wlj.net，可在 /etc/exports 文件里列出的是 linux2，这时就要检查 /etc/hosts 文件和 DNS 的设置。

在 NFS 客户机挂接 NFS 目录时，如果使用主机名而不是使用 IP 地址来指定 NFS 客户机，同样要确保客户机能够正确解析主机名的 IP 地址。如果不能正确解析，就要检查 /etc/hosts 文件和 DNS 的配置。

2. 故障排除常用命令

NFS 是基于 RPC 调用的，因此 rpcinfo 命令常常用于确定 RPC 服务的信息。我们前面已经介绍了“rpcinfo -p”命令，注意这一命令是 NFS 服务器上执行的，在客户机也可以使用 rpcinfo 命令确定远程 NFS 服务器的 RPC 服务信息。

【实例 8.21】

在客户机上执行以下命令以确定远程 NFS 服务器的 RPC 服务信息：

```
[root @redflag /root]#rpcinfo -u 192.168.0.15 portmap
program 100000 version 2 ready and waiting
[root @redflag /root]#rpcinfo -u 192.168.0.15 mountd
program 100005 version 1 ready and waiting
program 100005 version 2 ready and waiting
program 100005 version 3 ready and waiting
[root @redflag /root]#rpcinfo -u 192.168.0.15 nfs
program 100003 version 2 ready and waiting
program 100003 version 3 ready and waiting
```

nfsstat 命令可以显示 nfs 统计信息，命令格式如下：

nfsstat [参数选项]

参数选项：

-c：显示客户机上的 NFS 操作。

-s：显示服务器上的状态。

【注意】带“-c”和“-s”应分别在 NFS 客户机和 NFS 服务器执行。

【实例 8.22】

```
[root @redflag /root]#nfsstat -c
```

Client rpc stats:

calls	retrans	authrefrsh
68	0	0

Client nfs v2:

Client	nfs	v2:	setattr	root	lookup	readlink
0	0%	0	0%	0	0%	0
read	wrcache	write	create	remove	rename	
0	0%	0	0%	0	0%	0

link		symlink		mkdir		rmdir		readdir		fsstat	
0	0%	0	0%	0	0%	0	0%	0	0%	0	0%

Client nfs v3:

null		getattr		setattr		lookup		access		readlink	
0	0%	3	4%	0	0%	7	10%	54	79%	0	0%
read		write		create		mkdir		symlink		mknod	
1	1%	0	0%	0	0%	0	0%	0	0%	0	0%
remove		rmdir		rename		link		readdir		readdirplus	
0	0%	0	0%	0	0%	0	0%	1	1%	0	0%
fsstat		fsinfo		pathconf		commit					

【实例 8.23】

[root @redflag /root]#nfsstat -s

calls		badcalls		badauth		badclnt		xdrcall	
70		1		1		0		0	

Server nfs v2:

null		getattr		setattr		root		lookup		readlink	
1	100%	0	0%	0	0%	0	0%	0	0%	0	0%
read		wrcache		write		create		remove		rename	
0	0%	0	0%	0	0%	0	0%	0	0%	0	0%
link		symlink		mkdir		rmdir		readdir		fsstat	
0	0%	0	0%	0	0%	0	0%	0	0%	0	0%

Server nfs v3:

null		getattr		setattr		lookup		access		readlink	
1	1%	3	4%	0	0%	7	10%	54	78%	0	0%
read		write		create		mkdir		symlink		mknod	
1	1%	0	0%	0	0%	0	0%	0	0%	0	0%
remove		rmdir		rename		link		readdir		readdirplus	
0	0%	0	0%	0	0%	0	0%	1	1%	0	0%
fsstat		fsinfo		pathconf		commit					
1	1%	1		1%	0	0%	0	0%			

其中 calls 是 RPC 的调用数，而 bad calls 是不良 RPC 的调用数。如果该值太多，NFS 系统就有严重问题。

3. 诊断步骤

诊断 NFS 故障的一般步骤如下：

- (1) 检查 NFS 客户机和 NFS 服务器之间的通信是否正常(用 ping 命令)。
- (2) 检查 NFS 服务器上的 NFS 服务是否在运行，必要时用 “ /etc/rc.d/init.d/nfs restart ”

命令重新启动。

(3) 验证 NFS 服务器的/etc/exports 文件语法是否正确，包括导出权限。

(4) 检查客户端的 NFS 文件系统服务是否正常，必要时用 “ /etc/rc.d/init.d/netfs restart ” 命令重新启动。

(5) 验证/etc/fstab 文件中的配置是否正确。

本章小结

本章介绍 NFS 的基本原理：Linux 系统间文件的共享；NFS 服务器上的配置 /etc/exports 文件中导出项的配置、导出选项的含义、NFS 服务的启动、用 exports 导出目录；NFS 客户端的配置：挂接远程 NFS 分区、挂接选项的含义；如何通过 NFS 服务器的配置选项和 NFS 客户的挂接选项调整 NFS 的性能、可靠性、安全性；最后还介绍了 NFS 故障排除的基本方法。

习 题

1. 启动 NFS 服务的命令是_____ 或_____ 。

2. 在/etc/exports 文件中有一行：

/home

请解析其作用。

3. 在 NFS 客户端加载 NFS 有两种方法，分别是_____ 和_____ 。

第9章 动态主机配置协议 DHCP



IP 地址已是每台计算机必须配置的参数了，手工设置每一台计算机的 IP 地址成为管理员最不愿意做的一件事，于是自动配置 IP 地址的方法出现了，这就是 DHCP。DHCP 服务器能够从预先设置的 IP 地址池里自动给主机分配 IP 地址，它不仅能够保证 IP 地址不重复分配，也能及时回收 IP 地址以提高 IP 地址的利用率。

9.1 DHCP 简介

9.1.1 为什么需要 DHCP

TCP/IP 协议目前已经成为互联网的公用通信协议，在局域网上也是必不可少的协议。用 TCP/IP 协议进行通信时，每一台计算机(主机)都必须拥有一个 IP 地址用于在网络上标识自己。如果 IP 地址的设置是由系统管理员在每一台计算机上手工进行设置，把它设定为一个固定的 IP 地址时，就称为静态 IP 地址。设定静态的 IP 地址是常见的方法之一，但在许多场合并不适用。如果网络的规模较大，系统管理员给每一台计算机分配 IP 地址的工作量就会很大，而且常常因为用户不遵守规则而会出现错误，例如：导致 IP 地址的冲突等。在把大批计算机从一个网络移动到另一个网络或者改变部门计算机所属子网时，同样存在改变 IP 地址的工作量大的问题。随着笔记本电脑的普及，移动办公也是大家习以为常的事，当电脑从一个网络移动到另一网络时，则每次移动也需要改变 IP 地址，并且移动的电脑在每个网络都需要占用一个 IP 地址。

我们再来看看 IP 地址的占用问题。如果某个网络上有 200 台计算机，采用静态 IP 地址时，每台计算机都需要预留一个 IP 地址，即共需要 200 个 IP 地址。然而这 200 台计算机并不同时开机，假如只有 20 台计算机同时开机，我们就浪费了 180 个 IP 地址。这种情况对于 ISP(Internet Service Provider)来说，是一个十分严重的问题，如果 ISP 有 100 000 个用户，难道需要 100 000 个 IP 地址不成？因此必须要有方法解决这个问题。DHCP(Dynamic Host Configuration Protocol)就是应这个需求而诞生的。采用 DHCP 的方法配置计算机 IP 地址的方案称为动态 IP 地址。

在动态 IP 地址的方案中，每台计算机并不设定固定的 IP 地址，而是在计算机开机时才被分配一个 IP 地址，这台计算机被称为 DHCP 客户端(DHCP Client)。而负责给 DHCP 客户端分配 IP 地址的计算机称为 DHCP 服务器。也就是说 DHCP 是采用客户/服务器(Client/Server)模式，有明确的客户端和服务器角色的划分。DHCP 服务器在给 DHCP 客户分配 IP 地址(即 IP 地址租用)的时候，还会有租用时间的限制，超过租用时间时，DHCP 服务器就把这个 IP 地址回收。回收的 IP 地址可以重新分配给另一个 DHCP 客户，这样 IP 地址就被重复使用，

大大提高了 IP 地址的利用率。移动的计算机在不同的网络上开机时，将会获得它所在网络上的 DHCP 服务器分配的有效 IP 地址，也就不必手工更改 IP 地址的设置了。由于 DHCP 客户是在开机的时候自动获得 IP 地址的，因此并不能保证每次获得的 IP 地址是相同的。

动态 IP 地址方案可以减少管理员的工作量是显而易见的，只要 DHCP 服务器正常，IP 地址的冲突是不会发生的。要大批量更改计算机的所在子网或其他 IP 参数，只要在 DHCP 服务器上进行即可。

9.1.2 BOOTP 引导程序协议

DHCP 是对 BOOTP 的扩展，所以我们要先介绍 BOOTP(BOOTstrap Protocol)。BOOTP 也称为自举协议，它使用 UDP 来使一个工作站自动获取配置信息。

为了获取配置信息，协议软件广播一个 BOOTP 请求报文，收到请求报文的 BOOTP 服务器查找出发出请求的计算机的各项配置信息(如 IP 地址、默认路由地址、子网掩码等)，将配置信息放入一个 BOOTP 应答报文，并将应答报文返回给发出请求的计算机。这样，一台计算机就获得了所需的配置信息。由于计算机发送 BOOTP 请求报文时还没有 IP 地址，因此它会使用全广播地址作为目的地址，而使用全“0”作为源地址，BOOTP 服务器可使用广播(Broadcast)将应答报文返回给计算机，或使用收到的广播帧上的 MAC 地址进行单播(Unicast)。

但是，BOOTP 设计用于相对静态的环境，管理人员创建一个 BOOTP 配置文件，该文件定义了每一个主机的一组 BOOTP 参数。配置文件只能提供主机标识符到主机参数的静态映射，如果主机参数没有要求变化，BOOTP 的配置信息通常保持不变。配置文件不能快速更改，此外管理员必须为每一主机分配一个 IP 地址，并对服务器进行相应的配置，使它能够理解从主机到 IP 地址的映射。

由于 BOOTP 是静态配置 IP 地址和 IP 参数的，不可能充分利用 IP 地址和减少配置的工作量，因此有必要引入自动机制。

9.1.3 DHCP 动态主机配置协议

DHCP 是对 BOOTP 的扩充，此协议从两个方面对 BOOTP 进行有力的扩充。第一，DHCP 可使计算机通过一个消息获取它所需要的配置信息，例如，一个 DHCP 报文除了能获得 IP 地址，还能获得子网掩码、网关等。第二，DHCP 允许计算机快速动态获取 IP 地址，为了使用 DHCP 的动态地址分配机制，管理员必须配置 DHCP 服务器使得它能够提供一个 IP 地址。任何时候一旦有新的计算机连到网络上，新的计算机与服务器联系，并申请一个 IP 地址。服务器从管理员指定的 IP 地址中选择一个地址，并将它分配给该计算机。

DHCP 允许有三种类型的地址分配。第一种，和 BOOTP 类似，DHCP 允许手工配置，管理员可为特定的某个计算机配置特定的地址。第二种，管理员可为第一次连接到网络的计算机分配一个固定的地址。第三种，DHCP 允许完全动态配置，服务器可使计算机在一段时间内“租用”一个地址。

动态地址分配是 DHCP 的最重要和新颖的功能，与 BOOTP 所采用的静态分配地址不同的是，动态 IP 地址的分配不是一对一的映射，服务器不能预先知道客户机的身份。我们通过配置 DHCP 服务器使得任意一个主机都可以获得 IP 地址并开始通信。为了使自动配置

成为可能，DHCP 服务器一开始就拥有网络管理员交给它的一组 IP 地址，管理员定义服务器操作的规定，DHCP 客户机通过与服务器交换信息协商 IP 地址的使用。在交换中，服务器为客户机提供 IP 地址，客户机确认它已经接收此地址。一旦客户机接收了一个地址，它就开始使用此地址进行通信。

将所有的 TCP/IP 参数保存在 DHCP 服务器有以下的好处：

(1) 管理员能够快速检查 IP 地址及其他配置参数而不必前往每一台计算机，此外由于 DHCP 的数据库可以在一个中心位置(即 DHCP 服务器)完成更改，因此重新配置时也无需对每一台主机进行配置。

(2) DHCP 不会将相同的 IP 地址同时分配给两台主机，从而避免了冲突。

9.1.4 DHCP 的工作过程

DHCP 的工作过程如图 9-1 所示。

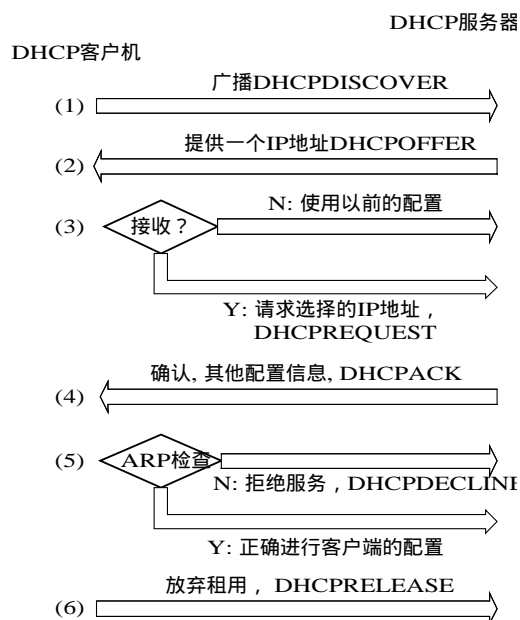


图 9-1 DHCP 的工作过程

(1) DHCP 客户机启动时，客户机在当前的子网中广播 DHCPDISCOVER 报文向 DHCP 服务器申请一个 IP 地址。

(2) DHCP 服务器收到 DHCPDISCOVER 报文后，它将从针对那台主机的地址区间中为它提供一个尚未被分配出去的 IP 地址，并把提供的 IP 地址暂时标记为不可用。服务器以 DHCPOFFER 报文送回给主机。如果网络里包含有不止一个的 DHCP 服务器，则客户机可能收到好几个 DHCPOFFER 报文，但客户机通常只承认第一个 DHCPOFFER。

(3) 客户端收到 DHCPOFFER 后，向服务器发送一个含有有关 DHCP 服务器提供的 IP 地址的 DHCPREQUEST 报文。如果客户端没有收到 DHCPOFFER 报文并且还记得以前的网络配置，此时使用以前的网络配置(如果该配置仍然在有效期限内)。

(4) DHCP 服务器向客户机发回一个含有原先被发出的 IP 地址及其分配方案的一个应答报文(DHCPACK)。

(5) 客户端接收到包含了配置参数的 DHCPACK 报文, 利用 ARP 检查网络上是否有相同的 IP 地址。如果检查通过, 则客户机接受这个 IP 地址及其参数, 如果发现问题, 客户机向服务器发送 DHCPDECLINE 信息, 并重新开始新的配置过程。服务器收到 DHCPDECLINE 信息, 将该地址标为不可用。

(6) DHCP 服务器只能将那个 IP 地址分配给 DHCP 客户一定时间, DHCP 客户必须在该次租用过期前对它进行更新。客户机在 50% 租借时间过去以后, 每隔一段时间就开始请求 DHCP 服务器更新当前租借, 如果 DHCP 服务器应答则租用延期。如果 DHCP 服务器始终没有应答, 则在有效租借期的 87.5%, 客户应该与任何一个其他的 DHCP 服务器通信, 并请求更新它的配置信息。如果客户机不能和所有的 DHCP 服务器取得联系, 租借时间到后, 它必须放弃当前的 IP 地址并重新发送一个 DHCPDISCOVER 报文开始上述的 IP 地址获得过程。

(7) 客户端可以主动向服务器发出 DHCPRELEASE 报文, 将当前的 IP 地址释放。

9.1.5 DHCP 功能的进一步讨论

从以上的讨论中, 可以看到 DHCP 可以提高 IP 地址的利用率, 减少 IP 地址的管理工作量, 便于移动用户的使用。但要注意的是, 由于客户机每次获得的 IP 地址不是固定的(当然现在的 DHCP 已经可以针对某一计算机分配固定的 IP 地址), 如果想利用某主机对外提供网络服务(例如 Web 服务、DNS 服务)等, 动态的 IP 地址是不可行的, 这时通常要求采用静态 IP 地址配置方法。此外对于一个只有几台计算机的小网络, DHCP 服务器则显得有点多余。

利用 TCP/IP 进行通信, 光有 IP 地址是不够的, 常常需要网关、WINS、DNS 等设置。DHCP 服务器除了能动态提供 IP 地址外, 还能同时提供 WINS、DNS 主机名、域名等附加信息, 完善 IP 地址参数的配置。

9.2 DHCP 的配置

DHCP 有服务器和客户机的角色划分, 其配置也就必须在服务器和客户端分别进行。

9.2.1 DHCP 服务器的配置

DHCP 服务器的配置步骤主要有:

- (1) 安装 DHCP 服务器软件包。
- (2) 配置/etc/dhcpd.conf 文件。
- (3) 生成 dhcpd.leases 文件。
- (4) 启动 dhcpd 服务。

1. 安装 DHCP 软件包

用 “rpm -qa |grep dhcpd” 命令确认 DHCP 服务器软件包是否已经存在, 如果不存在, 需先把红旗 Linux 3.0 光盘 mount 上, 在 RedFlag/RPMS 目录下找到软件包, 用 “rpm -ivh dhcp-2.0pl5-4.i386.rpm” 命令进行安装。

【实例 9.1】

```
[root @ redflag /root]#rpm -qa|grep dhcp
dhcp-2.0pl5-4
dhcpcd-1.3.18pl8-10
```

以上说明服务器软件包 dhcp-2.0pl5-4 已经存在。

2. 配置/etc/dhcpd.conf 文件

要向一个子网提供服务 ,DHCP 服务器的守护进程 dhcpd 需要知道它提供服务的子网号和子网掩码 ,此外为了分配动态 IP 地址 ,还必须在每一个子网提供一个或多个 IP 地址范围。以下是一个非常简单的配置文件。

【实例 9.2】

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    #IP 地址范围。
    range 192.168.0.100 192.168.0.199;
}
```

DHCP 服务器将向 192.168.0.0/255.255.255.0 子网提供 192.168.0.100 到 192.168.0.199 的 IP 地址。配置时,除了括号所在行外的其他每一行要加“;”号,以“#”号开头的语句是注释语句。

【实例 9.3】

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.20 192.168.0.69;
    range 192.168.0.100 192.168.0.199;
}
```

这是一个多地址范围的例子。

DHCP 的 IP 地址租用有时间的限制,租用时间可以是任意长度,具体长度根据需要服务的主机的类型来确定。例如:在一个办公环境中,系统不断增加、减少,但移动较少,那么其租用时间长度为一个月或更多个月比较合适。而在一个工厂的测试车间里,一个最长 30 分钟的租用期足以完成一个网络应用的简单测试过程,其租用时间就为几十分钟。可以指定两个租用时间长度:缺省租用时间和最大租用时间。前者是 DHCP 客户请求租用 IP 地址时如果未指定要租用的时间,DHCP 服务器会自动指定的租用时间长度;而后者是 DHCP 客户请求租用 IP 地址时可以指定的最大租用时间长度。它们是作为子网声明的子句定义的。常用的时间可以是 86 400(一天)、604 800(一周)、2 592 000(30 天)。如果 DHCP 服务器同时为多个子网服务,每个子网的租用时间又不相同,可以分别在不同的子网配置不同的租用时间。

【实例 9.4】

```
subnet 192.168.0.0 netmask 255.255.255.0 {
    #下一行是设置缺省租用时间。
    default-lease-time 600;
    #下一行是设置最大租用时间。
    max-lease-time 7200;
    range 192.168.0.100 192.168.0.199;
}
```

这个特殊的子网声明指定了一个 10 分钟的缺省租用时间和一个两小时的最大租用时间。

DHCP 提供一种机制，使得服务器可以向客户提供有关如何配置网卡的信息(例如：子网掩码)和客户如何访问各种网络服务(例如 DNS、网关等)的信息。这些选项可以通过每一子网中的 option 语句来定义。DHCP 可以指定的 option 很多，常用的选项见表 9-1，更多的选项信息请参见“man dhcp-options”命令。

表 9-1 dhcpd.conf 配置文件中可以指定的常用选项

关 键 字	含 义
subnet-mask ip-address	子网掩码
routers ip-address1,ip-address2,...	网关
domain-name-servers ip-address1,ip-address2,...	指明 DNS
host-name string	指明主机名
domain-name string	指明所在域的域名
broadcast-address ip-address	指明子网的广播地址
static-routers ip-address ip-address,...	配置客户的静态路由表
arp-cache-timeout int	指明 arp 缓冲表项的生存时间
nis-servers ip-address1,ip-address2,...	指明 NIS 服务器的地址
netbios-name-servers ip-address1,ip-address2,...	指明 WINS 服务器地址
netbios-node-type int	指明客户的 NetBIOS 节点的类型： 1 B-node：广播 2 P-node：WINS 3 M-node：先广播后 WINS 4 H-node：先 WINS 后广播

【实例 9.5】

```
subnet 192.168.0.0 netmask 255.255.255.0 {  
    default-lease-time 600;  
    max-lease-time 7200;  
    #下一行指定掩码信息为 255.255.255.0。  
    option subnet-mask 255.255.255.0;  
    #下一行指定广播地址为 192.168.0.255。  
    option broadcast-address 192.168.0.255;  
    #下一行指定网关为 192.168.0.254。  
    option routers 192.168.0.254;  
    #下一行指定域名服务器(DNS)为 192.168.0.1。  
    option domain-name-servers 192.168.0.1;  
    #下一行指定主机所在的域。  
    option domain-name "wlj.com";  
    range 192.168.0.100 192.168.0.199;  
}
```

3. 生成/var/lib/dhcp/dhcpd.leases 文件

通常安装 DHCP 软件包时并不产生 dhcpd.leases 文件。服务器守护进程 dhcpd 会使用这个文件来存储目前的租用信息，dhcpd 为每一客户分配一个 IP 地址时都会在这个文件记录该租用信息。在系统出现故障后或重新启动时，就可以从该文件中重新获得租用信息。用以下命令生成文件：

```
touch /var/lib/dhcp/dhcpd.leases
```

4. 启动 dhcpd 服务

我们已迫不及待地要启动 DHCP 服务，然而还有如下一件事情要确定。

```
[root @ redflag /root]#ifconfig
eth0      Link encap:Ethernet  HWaddr 00:E0:4C:30:03:F7
          inet addr:192.168.0.15  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:699 errors:0 dropped:0 overruns:0 frame:0
          TX packets:289 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:69881 (68.2 Kb)  TX bytes:27158 (26.5 Kb)
          Interrupt:5 Base address:0xe000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:420 (420.0 b)  TX bytes:420 (420.0 b)
```

查看第 3 行是否有 MULTICAST 信息，应该庆幸的是大多数的系统有该信息。如果没有，必须手工加入 255.255.255.255 这个 IP 地址至 routing-table 中，请以 root 用户登录，输入：

```
[root @ redflag /root]#route add -host 255.255.255.255 dev eth0
```

这是为了让 dhcpd 能够正确地为过分挑剔的 DHCP 客户端(例如：Windows 95)服务。DHCP 必须能够发送数据包到 255.255.255.255 这个 IP 地址，但是在以往的 Linux 系统中这个地址被用来做为监听子网广播的 IP 地址，所以在 2.2 内核之前版本要输入以上语句借以启用 MULTICAST 功能，让数据包传递到 255.255.255.255 这个地址。

【提示】 要想启动时系统自动添加该路由，请将此命令加入/etc/rc.d/rc.local 文件末尾。启动 DHCP 服务相当简单，例如：

```
[root @ redflag /root]#/etc/rc.d/init.d/dhcpd start 或
[root @ redflag /root]#service dhcpd start
Starting dhcpd:  [ OK ]
```

【提示】 想在 Linux 系统启动时同时启动 DHCP 服务器，可以使用 ntsysv 命令，找到 dhcpd 服务，标记为“*”号确认即可。

停止 DHCP 服务则如下：

```
[root @ redflag /root]#/etc/rc.d/init.d/dhcpd stop 或
```

```
[root @ redflag /root]#service dhcpd stop
```

9.2.2 DHCP 客户的配置

DHCP 客户可以有多类，如 Windows 98、Windows 2000 或 Linux，下面分别进行介绍。

1. Windows 2000 客户的配置

首先在 Windows 2000 下把 TCP/IP 地址设置为自动获得(见图 9-2)，如果 DHCP 服务器还提供 DNS、WINS 等，也把它们设置为自动获得。

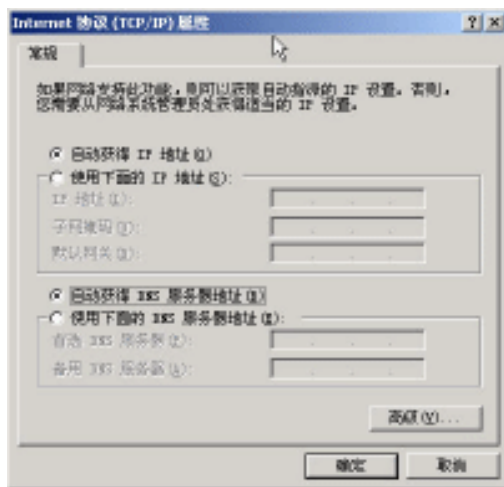


图 9-2 将 TCP/IP 地址改为自动获得

在“命令提示符”下，执行 C:/>ipconfig/renew 可以更新 IP 地址。而执行 C:/>ipconfig/all 可以看到 IP 地址、WINS、DNS、域名是否正确。要释放地址用 C:/>ipconfig/release 命令。

【实例 9.6】

```
C:\>ipconfig/renew
```

```
Windows 2000 IP Configuration
```

```
Ethernet adapter 本地连接:
```

```
Connection-specific DNS Suffix. : wlj.com
IP Address. . . . . : 192.168.0.11
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.0.1
```

```
C:\>ipconfig/all
```

```
Windows 2000 IP Configuration
```

```
Host Name . . . . . : w2000-client
Primary DNS Suffix . : 
Node Type . . . . . : Broadcast
IP Routing Enabled. . . . . : Yes
```

WINS Proxy Enabled. : No
 DNS Suffix Search List. : wlj.com

Ethernet adapter 本地连接:

Connection-specific DNS Suffix . : wlj.com
 Description : Realtek RTL8139(A) PCI Fast Ethernet Adapter
 Physical Address. : 00-E0-4C-83-C9-65
 DHCP Enabled. : Yes
 Autoconfiguration Enabled . . . : Yes
 IP Address. : 192.168.0.11
 Subnet Mask : 255.255.255.0
 Default Gateway : 192.168.0.1
 DHCP Server : 192.168.0.2
 DNS Servers : 192.168.0.1
 Lease Obtained. : 2003 年 4 月 16 日 10:47:57
 Lease Expires : 2003 年 4 月 16 日 10:57:57

C:\>ipconfig/release

Windows 2000 IP Configuration

IP address successfully released for adapter "本地连接"

2. Windows 98 客户的配置

先在 Windows 98 下把 TCP/IP 地址设置为自动获得,再执行 winipcfg 命令,如图 9-3 所示。点击“释放”按钮可以释放 IP 地址,点击“更新”可以获得新的 IP 地址。

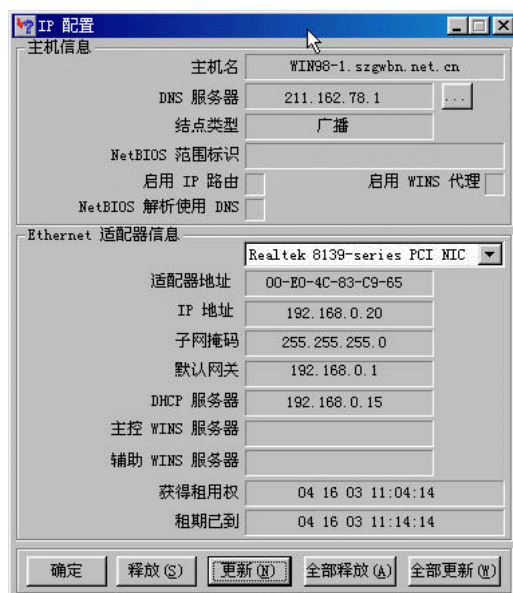


图 9-3 用 winipcfg 进行动态 IP 配置

3. Linux 客户端的设置

设置 DHCP 客户机有多种方式。下面介绍常见的三种。

第一种是使用 `linuxconf` 命令,选择要配置的 DHCP 客户网络设备,把 IP 地址设为 DHCP 方式。重新启动系统,系统将从 DHCP 服务器上获得 IP 地址。

第二种是手工修改 `/etc/sysconfig/network-scripts/ifcfg-eth0` 文件,把 `BOOTPROTO=“none”` 改为 `BOOTPROTO=“dhcp”`,再执行命令 `ifup`。

【实例 9.7】

```
[root @ redflag /root]#ifdown eth0
[root @ redflag /root]#ifup eth0
[root @ redflag /root]#ifconfig
Determining IP information for eth0... done.
eth0      Link encap:Ethernet  HWaddr 00:E0:4C:83:C9:65
          inet addr:192.168.0.20  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:41 errors:0 dropped:2 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:4530 (4.4 Kb)  TX bytes:16334 (15.9 Kb)
          Interrupt:11 Base address:0xc000
          .....
```

可以看到 IP 地址已经成功获取。

第三种是运行 `dhcpcd` 命令。首先要检查客户机是否安装了 `dhcpcd` 软件包。

```
[root @ redflag /root]#rpm -qalgrep dhcpcd
dhcpcd-1.3.18pl8-10
```

如果没有安装,请用 `rpm` 命令把 `dhcpcd` 软件包安装上。

【实例 9.8】

```
[root @ redflag /root]#mount /dev/cdrom
[root @ redflag /root]#cd /mnt/cdrom/RedFlag/RPMS
[root @ redflag /root]#rpm -ivh dhcpcd-1.3.18pl8-10.i386.rpm
Preparing...      ##### [100%]
 1:dhcpcd          ##### [100%]
[root @ redflag /root]#dhcpcd
[root @ redflag /root]#ifconfig
Determining IP information for eth0... done.
eth0      Link encap:Ethernet  HWaddr 00:E0:4C:83:C9:65
          inet addr:192.168.0.22  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:41 errors:0 dropped:2 overruns:0 carrier:0
```



```
collisions:0 txqueuelen:100
RX bytes:4530 (4.4 Kb) TX bytes:16334 (15.9 Kb)
Interrupt:11 Base address:0xc000
.....
```

可以看到 IP 地址已经成功获取。

要停止 dhcpcd 进程，可以使用 “dhcpcd -k” 命令，同时 IP 地址被释放。

9.3 DHCP 服务器的高级配置

9.3.1 为计算机分配固定的 IP 地址

如果你的主机做为服务器为其他用户提供网络服务(例如 Web 服务、DNS 服务和 FTP 服务)，这时 IP 地址最好能够固定。我们可以把它们的 IP 地址设为静态 IP 而不用动态 IP，此外我们也可以让 DHCP 服务器为它分配固定的 IP 地址。只要在 dhcpd.conf 文件中加入相应的配置。

【实例 9.9】

```
host webserver {
    #Web 服务器上网卡的 MAC 地址。
    hardware Ethernet 00:80:c8:1c:29:96;
    #固定的 IP 地址。
    fixed-address 192.168.0.210;
}
```

以上根据网卡 MAC 地址分配固定的 IP 地址。

9.3.2 进一步说明 dhcpd.conf

前面介绍了 dhcpd.conf 的基本用法，这里进一步介绍它的使用。dhcpd.conf 是一个递归下降格式的配置文件，由参数和声明两大类语句构成。参数类语句主要告诉 dhcpd 怎么做(例如，地址租用的时间长短)、是否做什么事情(例如，是否给未知客户分配地址)以及提供给客户什么参数(例如，网关是 192.168.0.254)。而声明语句则是描述网络的拓扑、描述网络上的客户、要分配给客户的 IP 地址以及提供一个参数组给一组声明等。

描述网络拓扑结构的声明语句有：“shared-network”和“subnet”语句。如果要给一个子网里的客户动态分配 IP 地址，那么在 subnet 声明里必须有一个 range 声明，用于说明地址范围。如果要给 DHCP 客户静态指定 IP 地址，那么每一个客户都要有一个 host 声明。group 语句用于对主机进行分组，指定共同的参数。一个典型的 dhcpd.conf 如下：

全局参数

```
shared-network shared-networkname {
    共享网络特定参数...
    subnet 192.168.0.0 netmask 255.255.255.0 {
        子网特定参数...
```

```

        range 192.168.0.10 192.168.0.199;
    }
    subnet 192.168.1.0 netmask 255.255.255.0 {
        子网特定参数...
        range 192.168.1.10 192.168.1.199;
    }
}
subnet 192.168.2.0 netmask 255.255.255.0 {
    子网特定参数...
    range 192.168.2.10 192.168.2.199;
}
group {
    组特定参数...
    host ws1.domain {
        特定主机参数...
    }
    host ws2.domain {
        特定主机参数...
    }
}

```

当客户启动时，首先采用的参数是 host 语句中的特定主机参数，其次是 group 语句中的组主机参数，再到 subnet 语句中的子网主机参数，下一个是 shared-network 语句中的共享网络主机参数，最后才是全局参数。即下层参数的声明会覆盖上层参数的声明、下层如无声明参数则采用上层的参数。

1. 声明语句

shared-network 语句用于告诉 dhcpd 服务器以下几个 IP 子网，其实是共享同一个物理网络。任何一个在共享物理网络里的子网都必须声明在 share-network 语句里。当属于其子网里的客户启动时，将获得在 share-network 语句里指定参数，除非这些参数被 subnet 或 host 里的参数覆盖。shared-network 的用法如下：

```

shared-network shared-networkname {
    [参数]
    [声明]
}

```

用 share-network 是一种权宜之计，例如某公司用 B 类网络 172.16.0.0，公司里的部门 A 被划在子网 172.16.0.0 里，子网掩码为 255.255.255.0，该子网最多只能容纳 254 台主机。但如果部门 A 急速增长，超过了 254 个节点，而物理网络还来不及增加，因此就要在原来这个物理网络上增加一个子网(例如，172.16.1.0)，而这两个子网其实是在同一个物理网络上。

【实例 9.10】

```

shared-network share1 {
    subnet-mask 255.255.255.0;
}

```

```
subnet 172.16.0.0 netmask 255.255.255.0 {  
    range 172.16.0.10 172.16.0.253;  
}  
subnet 172.16.1.0 netmask 255.255.255.0 {  
    range 172.16.1.10 172.16.1.253;  
}  
}
```

subnet 语句用于提供足够的信息来告诉 dhcpd 一个 IP 地址是否属于该子网。也可以提供指定的子网参数和该子网的 IP 地址范围，IP 地址必须在 range 声明里指定。subnet 语句的用法如下：

```
subnet subnet-number netmask netmask {  
    [参数]  
    [声明]  
}
```

【实例 9.11】

```
subnet 172.16.3.0 netmask 255.255.255.0 {  
    option router 172.16.3.254;  
    range 172.16.3.10 172.16.3.253;  
}
```

range 语句的用法如下：

```
range low-address [high-address];
```

对于一个有动态分配 IP 地址的 subnet 语句里，至少要有有一个 range 语句，用来指明要分配的 IP 地址的范围。如果只有一个 IP 地址要分配，高地址部分可以省略。

host 语句的用法如下：

```
host hostname {  
    [参数]  
    [声明]  
}
```

host 语句的作用是为特定的客户机提供网络信息，常常用于给固定的主机分配固定的 IP。

【实例 9.12】

```
host ftpserver{  
    hardware Ethernet 00:80:c8:1c:29:97;  
    fixed-address 192.168.0.2;  
}
```

group 的用法如下：

```
group {  
    [参数]  
    [声明]  
}
```

组语句是用于简化参数的应用，使用组把一组参数应用于一组主机、共享网络、子网、甚至是组(即组嵌套)。

【实例 9.13】

```
group {  
    option router 192.168.0.254;  
    host webserver{  
        hardware Ethernet 00:80:c8:1c:29:96;  
        fixed-address 192.168.0.1;  
    }  
    host ftpserver{  
        hardware Ethernet 00:80:c8:1c:29:97;  
        fixed-address 192.168.0.2;  
    }  
}
```

例子中的两个主机 router(网关)都为：192.168.0.254。

allow 和 deny 语句用来控制 dhcpd 对不同类客户请求的响应，unknown-clients 关键字表示未知的客户。如下：

```
allow unknown-clients;  
deny unknown-clients;
```

前者允许 dhcpd 分配动态 IP 给未知的客户，而后者则不允许，缺省是允许的。

2. 参数类语句

default-lease-time 语法如下：

```
default-lease-time time;
```

指定缺省租约时间，这里的 time 是以秒为单位的。如果 DHCP 客户在请求租用 IP 地址但没有指定租约的时间，租约时间就是缺省租约时间。

max-lease-time 语法如下：

```
max-lease-time time;
```

指定最大的租约时间，如果 DHCP 客户机在请求 IP 地址时指定了租约时间，则该值是被允许的最大租约时间。

hardware 语法如下：

```
hardware hardware-type hardware-address;
```

指明网卡类型(ethernet、token-ring)和硬件地址，硬件地址(即网卡地址)由 12 个十六进制位构成，每 2 位以“：”隔开，如 00:80:c8:1c:29:96。

fixed-address 语法如下：

```
fixed-address address [, address ...];
```

该语句指定一个或多个 IP 地址给一个 DHCP 客户，该语句只能出现在 host 声明里。

3. 选项类语句

选项类语句以 option 开头，后面跟一个选项名，选项名后是选项数据，选项非常的多，

常用选项见表 9-1。

9.3.3 DHCP 转接代理

当网络中存在多个子网被路由器隔开的时候,由于客户计算机只能通过广播发送 DHCP 请求,这些请求一般不能跨越路由器。我们可以在每一网络上都安置 DHCP 服务器,然而这样不便于管理。我们也可以设置路由器转发 DHCP 请求,即转发相应的 UDP 端口 67 和 68 的广播数据包,但这样设置就增加了网络广播,不利于减少网络流量。

还有另外一种方法来使得 DHCP 客户计算机能够使用子网之外的 DHCP 服务器来获得 IP 地址。这就是使用 DHCP 转接代理服务器来转发 DHCP 的请求。转接代理服务器把 DHCP 请求转移到其他网络的 DHCP 服务器上。转接代理服务器相当于一个中介,它接收到一个请求时,将把请求转发到命令行指定的 DHCP 服务器,由于它了解 DHCP 服务器的 IP 地址,因此能通过正常的 IP 数据包将原广播包转发到服务器中,而不使用广播。当转接代理服务器收到 DHCP 服务器分配的 IP 地址后,将在接受请求的网络上进行广播,使发出 DHCP 请求的主机得到 IP 地址。转接代理服务器实际上是一个程序 Dhcrelay。命令格式如下:

dhcrelay DHCP 服务器地址

【实例 9.14】

网络的拓扑结构如图 9-4 所示。

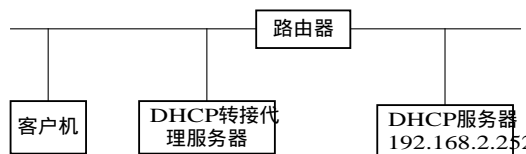


图 9-4 使用 DHCP 转接代理服务器的网络

```
[root @ redflag /root]#dhcrelay 192.168.2.252
Internet Software Consortium DHCP Relay Agent 2.0p15
Copyright 1997, 1998, 1999 The Internet Software Consortium.
All rights reserved.
Please contribute if you find this software useful.
For info, please visit http://www.isc.org/dhcp-contrib.html
Listening on Socket/eth0
Sending on    Socket/eth0
```

9.4 DHCP 故障排除

如果 DHCP 服务器不能正常分配 IP 地址,要先根据 DHCP 服务器的配置步骤逐一进行检查,特别是配置文件 dhcpd.conf 中关键字不要错误,以及每行的行末是否加了“;”。还可以以 debug 模式运行 DHCP 服务器,观察 DHCP 服务器的运行情况。首先停止 DHCP 服务,再以 debug 模式运行 DHCP 服务器,如下。

【实例 9.15】

```
[root @ redflag /root]#service dhcpd stop
Shutting down dhcpd: [ OK ]
[root @ redflag /root]#dhcpd -d
Internet Software Consortium DHCP Server 2.0pl5
Copyright 1995, 1996, 1997, 1998, 1999 The Internet Software Consortium.
All rights reserved.
Please contribute if you find this software useful.
For info, please visit http://www.isc.org/dhcp-contrib.html
Listening on Socket/eth0/192.168.0.0
Sending on Socket/eth0/192.168.0.0
DHCPDISCOVER from 00:e0:4c:83:c9:65 via eth0
DHCPOFFER on 192.168.0.11 to 00:e0:4c:83:c9:65 via eth0
DHCPREQUEST for 192.168.0.11 from 00:e0:4c:83:c9:65 via eth0
DHCPACK on 192.168.0.11 to 00:e0:4c:83:c9:65 via eth0
BOOTREQUEST from 00:ab:00:00:00:00 via eth0
No applicable record for BOOTP host 00:ab:00:00:00:00 via eth0
DHCPRELEASE of 192.168.0.11 from 00:e0:4c:83:c9:65 via eth0 (found)
```

从例子中可以清楚地看到 DHCP 服务器的工作过程，DHCP 报文的发送和接收、IP 地址分配过程等。

Linux 系统把系统的信息记录在/var/log/messages 文件中，因此我们可以查看文件中有关 DHCP 的内容来进行排错。同时，DHCP 服务器会把已经出租的 IP 地址存放在文件/var/lib/dhcp/dhcpd.leases 中，它也可以帮助我们排除故障。

【实例 9.16】

```
[root @ redflag /root]#cat /var/lib/dhcp/dhcpd.leases

# All times in this file are in UTC (GMT), not your local timezone.    This is
# not a bug, so please don't ask about it.    There is no portable way to
# store leases in the local timezone, so please don't request this as a
# feature.    If this is inconvenient or confusing to you, we sincerely
# apologize.    Seriously, though - don't ask.
# The format of this file is documented in the dhcpd.leases(5) manual page.
lease 192.168.0.11 {
    starts 1 2003/04/14 23:52:35;
    ends 2 2003/04/15 00:02:35;
    hardware ethernet 00:e0:4c:83:c9:65;
    uid 01:00:e0:4c:83:c9:65;
    client-hostname "w2000-client";
}
```

本章小结

本章首先介绍了静态 IP 地址方案和动态 IP 地址方案的区别，动态 IP 地址的优点主要是减少 IP 地址和 IP 参数管理的工作量、提高 IP 地址的利用率。DHCP 的工作过程主要有：DHCPDISCOVER、DHCPOFFER、DHCPREQUEST 和 DHCPACK 四个步骤。本章着重介绍了 DHCP 的配置：DHCP 服务器软件的安装、dhcpd.conf 文件的配置、dhcp.lease 文件的生成、DHCP 服务的启动。还介绍了不同的 DHCP 客户端：Windows 2000、Windows 98、Linux 客户的配置。最后介绍了 DHCP 的高级配置和 DHCP 故障的排除。

习 题

1. 动态 IP 地址方案有什么优点和缺点？
2. 要给 DHCP 客户机分配 192.168.0.1 到 192.168.0.199 的 IP 地址，同时指明 DNS 为 192.168.0.250、网关为 192.168.0.251，应如何书写 dhcpd.conf 文件？
3. 为什么要使用 DHCP 转接代理服务器？
4. 在以下的配置文件中，192.168.0.20 主机的 IP 地址租用时间是多少？

```
shared-network share1 {  
    option default-lease-time 28000;  
    subnet 192.168.0.0 netmask 255.255.255.0 {  
        range 192.168.0.1 192.168.0.199;  
    }  
  
    host webserver{  
        option default-lease-time 31536000;  
        hardware Ethernet 00:80:c8:1c:29:96;  
        fixed-address 192.168.0.20;  
    }  
}
```

第 10 章 Samba



网络的宗旨是更容易地在计算机间共享信息，Samba 实际上是一组程序，它们让你的 Linux 服务器懂得 SMB (Server Messages Block) 协议，而 SMB 是一套通讯协议，可以和运行 Windows 98、Windows 2000 等操作系统的计算机实现文件共享和打印机共享服务。在运行 Samba 服务器程序的时候，Linux 机器在“网络邻居”中看起来如同一台 Windows 机器。同时，Windows 机器的用户可以“登录”到 Linux 服务器上。

10.1 Samba 简介

10.1.1 SMB 协议

SMB(Server Message Block)通信协议可以看作是局域网上共享文件和打印机的一种协议。它是微软(Microsoft)和英特尔(Intel)在 1987 年制定的协议，主要是作为 Microsoft 网络的通讯协议，而 Samba 则是将 SMB 协议搬到 Unix 上来应用。通过“NetBIOS over TCP/IP”使得 Samba 不但能与局域网络主机分享资源，更能与全世界的电脑分享资源。因为互联网上千千万万的主机所使用的通讯协议就是 TCP/IP。SMB 是在会话层(session layer)和表示层(presentation layer)以及小部分应用层(application layer)的协议。SMB 使用了 NetBIOS 的应用程序接口(Application Program Interface, 简称 API)。另外，它是一个开放性的协议，允许协议扩展，这使得它变得更大而且复杂，大约有 65 个最上层的作业，而每个作业都超过 120 个函数。

10.1.2 什么是 Samba

Samba 是用来实现 SMB 协议的一种软件，由澳大利亚的 Andrew Tridgell 开发，是一套让 Unix 系统能够应用 Microsoft 网络通讯协议的软件。它使执行 Unix 系统的机器能与执行 Windows 系统的共享资源。Samba 属于 GNU Public License (简称 GPL)的软件；因此，可以合法且免费地使用它。作为 Unix 的克隆，Linux 也可以运行这套软件。这套软件由一系列的组件构成，主要的组件如表 10-1 所示。

Samba 的运行包含两个后台守护进程：nmbd 和 smbd，它们是 Samba 的核心。nmbd 程序使其他计算机可以浏览 Linux 服务器，smbd 守护进程在 SMB 服务请求到达时对它们进行处理，并且为被使用或共享的资源进行协调。如果指定的是一个文件，该资源就是一个文件；而一个打印机请求就要求访问一台打印机。实际上，在请求访问打印机时，smbd 把要打印的信息存储到打印队列中；在请求访问一个文件时，smbd 把数据发到内核，最后将它存到磁盘上。

表 10-1 Samba 软件包的组件

smbd	SMB 服务器，给 SMB 客户提供文件和打印服务
nmbd	NetBIOS 名称服务器,提供 NetBIOS 名称服务和浏览支持，帮助 SMB 客户定位服务器
smbclient	SMB 客户程序，用来存取 SMB 服务器上的共享资源
testprns	测试服务器上打印机访问的程序
testparm	测试 Samba 配置文件 smb.conf 正确性的工具
smb.conf	Samba 的配置文件
smbstatus	列出当前 smbd 服务器上的连接
smbpasswd	用来设定 Samba 用户密码
Swat	Samba 的 Web 管理工具
mk smbpasswd.sh	从/etc/passwd 文件形成 smbpasswd 文件的程序
Smb	启动或停止 Samba 服务的脚本程序
smbadduser	更新/etc/samba/smbusers 文件和/etc/samba/smbpasswd 文件

在 Linux 操作系统中，只要在安装的时候选择了 Samba，那么它就会在安装 Linux 的同时安装 Samba。如果没有选择的话，也可以在光盘上找到 Samba 的 RPM 安装包，使用 RPM 安装它就可以了。如果你的 Linux 发布没有包含这个软件，可以到 <ftp://samba.org/pub/samba> 去下载。

10.1.3 Samba 的功能

通过使用 Samba，Linux 可以实现如下功能：

(1) 提供 Windows NT 风格的文件和打印机共享。当 Windows 95、Windows 98、Windows 2000 等共享 Linux 操作系统的资源时，外表看起来和 Windows 的资源没有区别。

(2) 解析 NetBIOS 名字。在 Windows 网络中，为了能够利用网上资源，同时自己的资源也能被别人所利用；各个主机都定期向网上广播自己的身份信息。而负责收集这些信息并为别的主机提供检索情报的服务器就被称为浏览服务器，Samba 可以有效地完成这项功能。在跨越网关的时候 Samba 还可以作 WINS 服务器使用。

(3) 提供 SMB 客户功能。利用 Samba 提供的 smbclient 程序可以从 Linux 下像使用 FTP 一样访问 Windows 的资源。

(4) 备份 PC 上的资源。利用一个叫 smbtar 的 Shell 脚本，可以使用 tar 格式备份和恢复一台远程 Windows 上的共享文件。

(5) 提供一个命令行工具，在其上可以有限制地支持 NT 的某些管理功能。

(6) 支持 SWAT(Samba Web Administration Tool)。

(7) 支持 SSL(Secure Socket Layer)。

10.1.4 Samba 的启动和退出

我们可以通过命令 `ntsysv` 来设定在系统启动时自动启动 Samba，也可以通过手工启动 Samba，具体的命令如下。

(1) 启动。

```
[root@redflag /root]# /etc/rc.d/init.d/smb start 或
```

```
[root@redflag /root]#service smb start
```

(2) 重新启动。

```
[root@redflag /root]# /etc/rc.d/init.d/smb restart 或
```

```
[root@redflag /root]#service smb restart
```

(3) 停止。

```
[root@redflag /root]# /etc/rc.d/init.d/smb stop 或
```

```
[root@redflag /root]#service smb stop
```

在启动 Samba 服务之前首先要完成其配置文件，接下来我们将详细的介绍。

10.2 Samba 配置

配置 Samba 的工作其实就是对它的配置文件 `smb.conf` 进行相应的设置。`smb.conf` 关系着 Samba 服务器的权限设置，以及共享的目录、打印机和机器所属的工作组等各种细致的选项。

10.2.1 设置 `smb.conf` 文件

1. `smb.conf` 的语法

文件 `smb.conf` 位于 `/etc/samba/` 目录下，它的语法非常明确。与 Windows 的 “*.ini” 文件十分相似。如下所示：

(1) 文件被分成几部分，每一部分都包含几个参数，用来定义 Samba 输出的共享及其详细操作。

(2) 文件被分成段，每一段的名称用一个方括号括起来，例如 `[global]`、`[home]`、`[printers]` 等。

(3) `[global]` 部分定义的参数用来控制 Samba 的总体特性，而其他每一部分都定义了一个专门的服务。

(4) 在每一段内用 “名称=值” 的格式来设置参数，例如，`read only = yes`。

(5) 行首前面加 “;” 或 “#” 表示该行为注释。

2. `smb.conf` 文件结构

`smb.conf` 文件最基本的三个特殊段分别是：

(1) `global`(全局参数)。

(2) `directory shares`(目录共享)——包括标准的 `[home]` 部分。

(3) `printer shares`(打印共享)部分。

除了 `[global]` 段外，所有的段都可以看作是一个共享资源，段名是该共享资源的名称，而段里的参数就是共享资源的属性。`[global]`、`[homes]` 和 `[printers]` 这三个段是比较特殊的。

3. `smb.conf` 配置实例

下面我们将用实际应用中一个具体的例子来讲述 `smb.conf` 的配置。

【实例 10.1】

以下是作者配置好的 smb.conf 配置文件，本实例分[global]和共享服务两节来介绍。为了学习方便，本部分采用逐行注释，其中“#”和“;”开始的行在执行时都被忽略。

(1) [global]节的内容如下：

```
[global]
```

```
netbios name = Redflag
```

指定本机在网上邻居中的显示名，如图 10-1 所示。

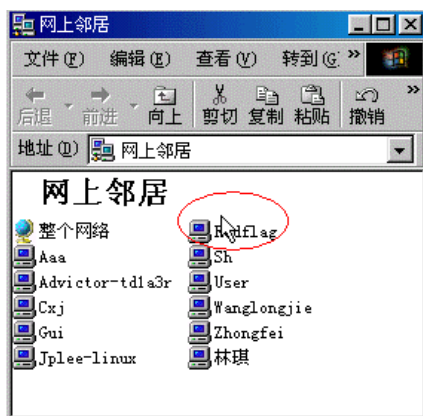


图 10-1 Samba 服务器在 Windows 网上邻居中的显示名

```
workgroup = WORKGROUP
```

#指定本机所属的工作组，默认值为 MYGROUP，用户可以根据自己的喜爱进行相应的修改。

```
server string = This is my Samba Server
```

#指定本机的备注。

```
hosts allow = 192.168.100. 192.168.200. 127.
```

#hosts allow 是一个用来指定在局域网中哪些机器可以使用 Samba 服务器共享的资源(详见 10.2.2 节)。

```
printcap name = /etc/samba/printcap
```

#打印机配置文件。

```
load printers = yes
```

#是否要共享打印机。

```
;printing = bsd
```

#除非打印机不是标准型号，否则没必要设置该项。

```
;guest account = test
```

#如果希望建立一个客户账号，就选择该项，同时在/etc/passwd 文件中要定义该账号，否则将使用用户“nobody”作为客户账号。

```
;map to guest = Bad User
```

#guest 用户映射，有效值是“Never”、“Bad User”、“Bad Password”(详见 10.2.4 节)。

```
log file = /var/log/samba/log.%m
```

#指定了 log 日志文件的存放地址。

【说明】在 `log file = /var/log/samba/log.%m` 行中，`%m` 是 samba 里定义的宏，宏用百分号后面跟一字符表示，在具体运作的时候就用实际的参数来代替，灵活地应用宏可以很方便地管理比较复杂的网络。常用的宏如表 10-2 所示。

表 10-2 smb.conf 文件中常用的宏

宏	描 述
%S	当前共享的名称
%P	当前服务的根路径
%u	当前服务的用户名
%g	给定%u 的所在工作组名
%H	给定的%u 的私人目录
%v	Samba 版本号
%h	运行 Samba 的主机的名称
%m	客户机的 NetBIOS 名
%L	服务器的 NetBIOS 名
%T	当前的日期和时间
%I	客户机的 IP 地址
%d	当前服务器进程 ID

```
max log size = 50
```

#指定了日志文件的最大长度，单位为 KB。

```
security = user
```

#安全级别为用户级，这样定义是为了让每一个 Windows 的客户端能自由使用它们在 Linux 服务器上的目录，关于安全级别的细节详见 10.2.3 节。

```
encrypt passwords = yes
```

#如果用户使用加密口令，则使用该项。

```
smb passwd file = /etc/samba/smbpasswd
```

#指定用户口令校验的密码文件的位置。

```
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
```

#大多数人发现使用该选项可以显著提高服务器的执行效率。

```
; interfaces = 192.168.12.2/24 192.168.13.2/24
```

#配置 Samba 使用多个网络界面，如果有多个网络界面，那么必须用“interfaces”选项按照该行中格式列出。

```
;local master = no
```

#是浏览控制选项。如果不想使 Samba 服务器成为局域网内部的主浏览服务器，将此选项设为“no”。

```
;domain master = yes
```

#将 Samba 服务器定义为主域控制器，此选项将允许 Samba 在子网列表中比较。如果已经有一台 Windows 域控制器，不要使用此项。

```
;domain controller = <NT-Domain-Controller-SMBName>
```

#仅当网络中有一台在安装时设置为主域控制器的 NT 服务器时使用该选项。

```
;domain logons = yes
```

#如果想使 Samba 成为 Windows 98 等工作站的登录服务器，使用此项。

```
;wins support = yes
```

#Windows 的 Internet 域名服务支持记录部分，它告诉 nmbd 守护进程支持 WINS 服务器。

```
;wins server = w.x.y.z
```

#告诉 nmbd 守护进程作为 WINS 客户机。

```
;wins proxy = no
```

#代表一个非 WINS 客户通知 Samba 响应名字解析请求，要使此选项正常工作必须保证网络中至少有一台 WINS 服务器，缺省值是“no”。

```
dns proxy = no
```

#该选项决定 Samba 是否通过 DNS 的 nslookups 去解析主机的 NetBIOS 名。

(2) 共享服务节的内容如下：

一个共享资源段由要提供访问的路径和附加的访问权限组成，可以是文件资源或打印资源。共享资源段可以给 guest 级的用户服务，一般来说 guest 用户是不需要密码的，如果给 guest 提供服务，guest 的权限是根据 Linux 里的 guest 账户来赋予的。如果共享资源段不是给 guest 提供服务的，则访问这样的段需要用户的密码。由于赋予用户对共享资源段的访问权限是基于该用户在 Linux 系统里对该资源的访问权限，所以服务器是不能赋予超过 Linux 系统赋予该用户的权限的。

[homes]段的内容如下：

```
[homes]
```

```
comment = Home Directories
```

#就是对共享的描述，可以是任意的字符串。

```
browseable = yes
```

#指定共享是否可以浏览，默认是“yes”。

```
writable = yes
```

#指定共享路径是否可以写，值是“yes”或“no”。

在[homes]段中，当用户请求一个共享时，服务器将在存在的共享资源段中寻找，如果找到匹配的共享资源段，就使用这个共享资源段。如果找不到，就将请求的共享名看成是用户的用户名，并在本地的 password 文件里找这个用户，如果用户名存在且用户提供的密码是正确的，则以这个 home 段克隆出一个共享提供给用户。这个新的共享的名称是用户的用户名，而不是 home，如果[home]段里没有指定共享路径，就把该用户的主目录(home directory)作为共享路径。

通常的共享资源段能指定的参数基本上都可以指定给[home]段。但一般情况下[home]段有如下配置就可以满足普通的应用。

```
[homes]
```

```
writable = yes
```

【注意】 如果在[home]段里加了

```
guess access = ok
```

则所有的用户都可以不要密码就能访问所有的主目录！

[printers]段的内容如下：

```
[printers]
#该段用于提供打印服务。
comment = All Printers
path = /usr/spool/samba
# path 是提供共享服务的路径，可以用%u、%m 这样的宏来代替路径里的 Linux 用户和客户机的
  NetBIOS 名。用户在连接到这个共享时具体的路径会被他的用户名代替，要注意这个用户名
  路径一定要存在，否则，客户机在访问时会找不到网络路径。
browseable = yes
guest ok = no
browseable = no
writable = no
printable = yes
#指定用户能不能打印，默认是“no”，要让一个打印共享可以让用户使用，必须设为“yes”。
printer driver = HP LaserJet 4
#打印机的驱动类型，这个参数可以让 Windows 知道远程打印机上的类型，具体的值可以参考在
  Windows 里安装打印机出现选择打印机类型时的打印机类型。
```

【注意】如果定义了[printers]这个段，用户就可以连接在 printcap 文件里指定的打印机。当一个连接请求到来时，smbd 去查看配置文件里已有的段，如果和请求匹配，就用那个段；如果找不到匹配的段，但[home]段存在，就用[home]段。否则请求的共享名就当作是个打印机共享名，然后去寻找适合的 printcap 文件，看看请求的共享名是不是个有效的打印共享名。如果匹配，那么就克隆出一个新的打印机共享提供给客户。[printers]服务必须是 printable，如果指定为其他，服务器将拒绝加载配置文件。

[public]段的内容如下：

```
[public]
#所有用户都可以读写的目录，方括号内的内容是在 Windows 下看到的该共享目录的名称。
comment = Public Zone
path = /home/public
public = yes
#这个参数指明是否允许 guest 账户访问，值为“yes”或“no”。
writable = yes
```

【说明】上面的代码段定义了一个所有用户都可以读写的共享目录。用户在这个目录中创建的文件都将具有默认用户的权限，所以任何可以访问该目录的用户都可以删除别的用户在这里放置的文件。显然，这个目录对于默认用户必须是可写的。

10.2.2 共享访问控制

在设置共享时我们必须考虑到网络安全，因此有必要对共享的目录进行访问控制。下面介绍几种常见的访问控制方法。

1. 通过读写方式来进行访问控制

通过设置对某个目录的读写来进行访问控制是较为常见的方法。这种访问控制通过以下参数实现：

read only

若该项设置成 “yes”，则用户只能读而不能进行创建、修改和删除等操作。

writable

若该项设置成 “yes”，则用户可以具有写操作的权限，开放该权限时一定要慎重考虑。

【实例 10.2】

下面是对 /home/share 目录进行的共享设定：

```
[myshare]
    comment = share file
    path = /home/share
    browseable = yes
    read only = yes
    writable = no
    .....
    .....
```

2. 通过控制主机地址来进行访问控制

主机地址访问控制通过以下参数实现：

hosts allow

hosts deny

“hosts allow” 指定允许访问的主机列表，而 “hosts deny” 指定拒绝访问的主机列表，主机列表用空格或逗号等隔开。主机列表可以是主机名、IP 地址、子网地址，也可以用 “EXPECT” 关键字来限制子网中的个别主机。

【实例 10.3】

在 smb.conf 的 [global] 段中加入下面的行：

```
[global]
netbios name = redflag
workgroup = workgroup
.....
.....
hosts allow = 192.168.100. 192.168.150. EXCEPT 192.168.150.33
#允许网段 192.168.100.0 和 192.168.150.0 除了 192.168.150.33 主机之外的所有主机访问。
hosts deny = 192.168.50.
#拒绝网段 192.168.50 的主机的访问。
.....
.....
```

3. 通过控制用户名来进行访问控制

如果某些目录只想让具有合法账号的用户访问，那么通过用户名来实现访问控制是一种比较理想的方法。用户名访问控制通过以下几组参数实现：

(1) public 和 guest 这两个参数功能相同，使得相应的资源被 guest 访问，即不需要用户名和密码的验证即可获得资源的访问权限。如果指定的是“guest only”，则表示相应的服务只允许由 guest 用户来访问。

(2) invalid users 和 valid users valid users 指定共享资源的有效用户，即允许访问该资源的用户。invalid users 和 valid users 相反，指定哪些用户不可访问共享资源。列表中的用户名可以用空格隔开。

【实例 10.4】

下面是对/home/share 目录进行的共享设定：

```
[myshare]
comment = share file for user1 and user2
path = /home/share
valid users = user1 user2
# user1 和 user2 是有效用户。
browseable = yes
writable = yes
printable = yes
.....
.....
```

10.2.3 Samba 安全级别

Samba 是重要的安全配置，其参数有四个值，分别是 share、user、server 和 domain，下面分别介绍。

(1) security = user：这是 Samba 的默认配置，这种情况下要求用户在访问共享资源之前必须先提供用户名和密码进行验证。

(2) security = share：这是几乎没有安全性的级别，任何用户都可以不要用户名和口令访问服务器上的资源。

(3) security = server：这个安全级和 user 安全级类似，但用户名和密码是递交到另外一个 SMB 服务器去验证，比如递交给一台 NT 服务器。如果递交失败，就退到 user 安全级，从用户端看来，server 和 user 这两个级别是没什么分别的。

(4) security = domain：这种安全级别要求网络上存在一台 NT PDC，Samba 把用户名和密码递交给 NT PDC 去验证。从用户端看来，user 级以上的安全级其实是没什么分别的，只是服务器验证的方式不同，但这三种安全级都要求用户在本 Linux 机器上要有系统账户，否则是不能访问的。

10.2.4 guest 用户映射

guest 用户映射仅适用于安全模式(user、server 和 domain 安全级)。如果一个用户没有通

过身份验证,就可以将其映射为 guest 用户,从而允许它访问 guest 共享。这里的 guest 用户由[global]段中的“ guest account ”参数指定。

guest 用户映射由全局参数“ map to guest ”控制,它只能放在[global]节中,可以是如下三个值。

(1) map to guest = Never: 该值不进行映射,拒绝非法用户访问任何资源,这是默认行为。

(2) map to guest = Bad User: 该值用于如果用户使用了一个不存在的账号登录,就将他映射为 guest 用户;如果提供的账号正确而口令错误,则禁止连接。

(3) map to guest = Bad Password: 该值将使用错误口令登录的用户映射为 guest 用户。但是这样的设置会产生一个问题,即如果用户不小心键入了错误的口令,服务器会“偷偷地”将他映射为 guest 用户,而不给出任何错误信息,这样该用户会在不知道的情况下受到种种访问控制。

10.3 使用加密口令

现在的 Windows 98、Windows 2000 等操作系统在网络中仅传递加密口令作为用户认证的信息。这些客户机在和不支持加密口令并且以 user 安全级运行的 Samba 服务器通信时,会遇到很大的困难。为了能顺利的和这些客户机进行通信,Samba 也加入了对加密口令的支持。

10.3.1 Samba 口令文件

为了使用加密口令,Samba 需要一份自己的口令文件,并且该文件应该和 Linux 的 /etc/passwd 文件保持同步。可以使用 mksmbpasswd.sh 脚本来制作这份文件:

```
[root@redflag /root]# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

其中 smbpasswd 就是需要的口令文件,其权限应该设为 0600,拥有者是 root。

Samba 口令文件中的记录和/etc/passwd 文件的记录一一对应,它的格式和/etc/passwd 文件基本相同,只有密码字段不同。密码字段由两个部分组成,每部分都是 32 个 X。这两部分就是 Samba 使用的加密口令,前一部分用于和 Lanman 通信,后一部分用于和 Windows 通信。Linux 的用户口令是无法破译的,因此,刚建立的口令文件中所有的用户都没有正确的口令,用 32 个 X 来表示。root 用户可以使用 smbpasswd 命令为每个用户设定一个初始口令,这样用户就可以使用 smbpasswd 命令更改自己的 Samba 口令了。smbpasswd 的使用方法和 passwd 命令类似。

root 用户也可以将口令设置为空,方法是手工修改口令文件,将前 11 个 X 改为“NO PASSWORD”。

这样当 smbpasswd 命令询问用户旧密码时,就可以直接按回车键跳过。

10.3.2 使用加密口令

要让 Samba 使用加密口令,需要在/etc/samba/smb.conf 文件的[global]节中加入以下参数:

```
encrypt passwords = yes
```

```
smb passwd file = /etc/samba/smbpasswd
```

上面两行默认时前面有分号，因此要将分号去掉才能生效，否则以分号开始的行表示注释。第一行通知 Samba 使用加密口令，第二行给出口令文件的路径。

以上两行生效后，在 Windows 端访问 Samba 共享资源时将要求输入密码，如图 10-2 所示。

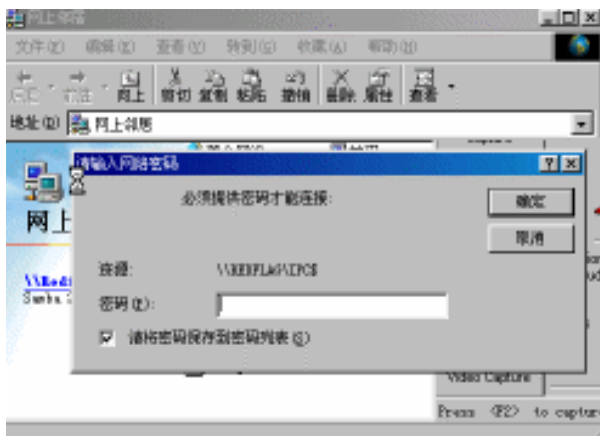


图 10-2 访问 Samba 资源时需要输入密码

【注意】 如果允许用户使用空口令，则应该修改如下参数：

```
null passwords = yes
```

但是这可能会带来安全问题。

10.3.3 smbpasswd 的使用

smbpasswd 命令除了可以修改 Samba 用户的口令外，还可以帮助 root 用户添加或删除用户。smbpasswd 命令的常用参数如下。

1. -a

该选项增加一个用户账号，该账号必须出现在/etc/passwd 文件中，也就是说是一个合法的 Linux 账号，只有 root 可以使用本选项。

【实例 10.5】

添加一个 Samba 用户。

(1) 添加一个 Linux 用户 test。

```
[root@redflag /root]# useradd test
```

(2) 生成 Samba 口令文件

```
[root@redflag /root]# cat /etc/passwd | mksmbpasswd.sh > /etc/samba/smbpasswd
```

(3) 添加一个 Samba 用户 test，同时设定 test 的 Samba 密码。

```
[root@redflag /root]# smbpasswd -a test
```

完成上述三个步骤后，就可以在 Windows 的网上邻居点击 Samba 主机，输入 Samba 密码后就可以访问 Samba 共享资源了，如图 10-3 所示。

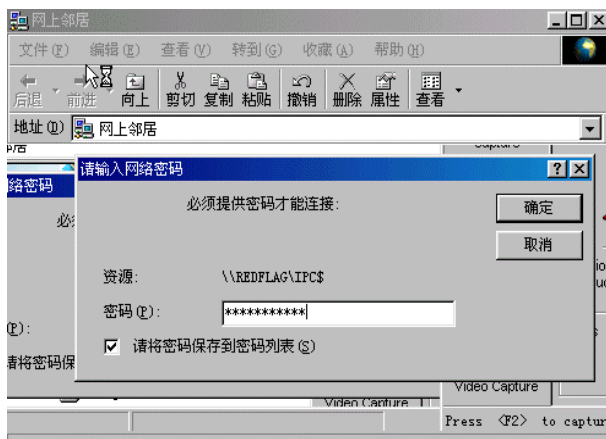


图 10-3 输入 Samba 密码

【说明】 Samba 用户的密码和 Linux 用户的密码不是同一回事，二者可以相同，也可以不同。

2. -d

该选项禁止一个用户账号，并不将其删除，仅供 root 使用。

3. -e

该选项恢复一个被禁止的用户账号，仅供 root 使用。

4. -n

该选项将账号的口令设为空，即将口令字段的前 11 个字符改为“NO PASSWORD”，仅供 root 使用。

5. -r remote_machine_name

该选项允许用户指定远程主机，如果没有该选项，那么 smbpasswd 默认修改本地 Samba 服务器上的口令。remote_machine_name 是远程主机的 NetBIOS 名。

6. -U username

该选项只能和“-r”选项连用。当修改远程主机上的口令时，用户可以用该选项指定欲修改的账号。还允许在不同系统中使用不同账号的用户修改自己的口令。

7. -s

该选项使 smbpasswd 以 silent(沉默)模式工作，在此模式下，smbpasswd 将从标准输入读取数据，而不是默认的/dev/tty。这可以帮助用户编写调用 smbpasswd 的脚本。

10.3.4 不使用加密口令

使用加密口令主要是为了顺利地完成了和 Windows 的通信，也许大家觉得有些麻烦，我们可以用另外一种方法来实现 Linux 和 Windows 的通信。这种方法就是修改 Windows 的注册表，强制其使用不加密的口令。不同的操作系统，修改注册表的位置是不同的，我们主要讨论以下三种操作系统。

(1) 对于 Windows 98 操作系统,可以在[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]里加一个 DWORD 变量,变量名称为 EnablePlainTextPassword,变量值为 1,即

EnablePlainTextPassword=00000001

如图 10-4 所示。



图 10-4 注册表中变量 EnablePlainTextPassword

(2) 对于 Windows NT 操作系统,可以在[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Rdr\Parameters]里加一个 DWORD 变量,EnablePlainTextPassword,变量值为 1,即

EnablePlainTextPassword=00000001

(3) 对于 Windows 2000 操作系统,可以在[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanWorkStation\Parameters]里加一个 DWORD 变量,EnablePlainTextPassword,变量值为 1,即

EnablePlainTextPassword=00000001

通过上面的三种方法,就可以让 Linux 和 Windows 能在网络上传送明文密码,但是使用明文传输会带来诸如安全等方面的麻烦,所以我们仍然推荐采用加密传输。

10.4 Samba 和 Windows 互相通信

通过配置 Samba 服务器,可以提供 Windows 客户机来访问 Linux 系统上共享的资源。也能够在 Linux 上通过 SMB 协议访问 Windows 系统上的共享资源。只是习惯上 Linux 计算机用做服务器,Windows 计算机用做客户机,很少有让 Linux 访问 Windows 计算机的需要。但是越来越多的计算机使用者不但使用 Linux 作服务器,也使用 Linux 来处理很多其他任务,那么就可能会需要访问其他 Windows 计算机上的资源。

10.4.1 从 Linux 机上访问 Windows 资源

Samba 软件中用于访问网络上其他 SMB 资源的软件为 smbclient,它是一个类似于 FTP 操作方式,通过远程操作的方式进行文件传递的软件。smbclient 命令的语法是:

```
smbclient <servicename> [password] [options]
```

其中 servicename 是要连接的资源名称,资源名称的格式如下:

```
//server/service
```

其中 server 是远程服务器的 NetBIOS 名字,对于 Windows 来说,就是它们出现在“网上邻居”中的名字,而 service 是共享目录的名称。

password 是存取该资源所需要的口令,如果没有给出 password 参数,也没有使用“-N”

选项，smbclient 会提示用户输入口令。

对于 smbclient 命令的参数，我们需要掌握如下几个：

- (1) -L：该选项列出远程服务器提供的所有资源。
- (2) -N：该选项禁止 smbclient 提示用户输入口令，当连接不需要口令的资源时可以使用该选项。
- (3) -I：该选项如果不清楚一个 Windows 计算机的 NetBIOS 名字，可以使用“-I”指定要访问计算机的 IP 地址，这时 NetBIOS 的名字将被忽略。

【实例 10.6】

本实例中，Linux 计算机的 IP 地址为 192.168.100.33，其 Samba 配置文件中的 NetBIOS 名字为 Redflag，Windows 2000 计算机的 IP 地址是 192.168.100.39，NetBIOS 名字为 lgm，共享的文件夹为“web”和“test”，如图 10-5 所示。

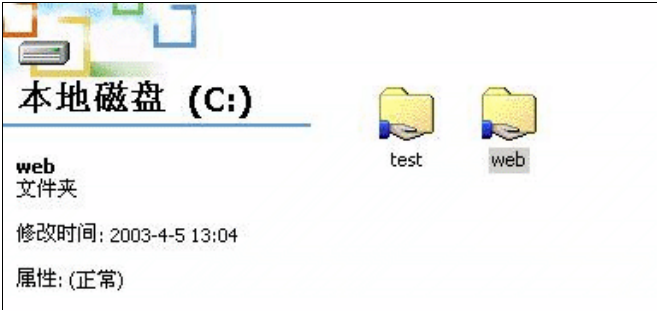


图 10-5 在 Windows 2000 中共享的目录

- (1) 列出 Windows 计算机共享出来的资源。

```
[root@redflag /root]# smbclient -L lgm
```

结果如下：

```
Password:
added interface ip=192.168.100.33 bcast=192.168.100.255 nmask=255.255.255.0
Got a positive name query response from 192.168.100.39 ( 192.168.100.39 )
Domain=[WORKGROUP] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
```

Sharename	Type	Comment
-----	----	-----
IPC\$	IPC	远程 IPC
web	Disk	
test	Disk	
#共享的目录“web”和“test”。		
ADMIN\$	Disk	远程管理
C\$	Disk	默认共享
Server		Comment
-----		-----

```

DZYL11
LGM
LINUXSEVER          samba server
PC23
REDFLAG             samba server
REDFLAG36           Samba Server
Workgroup            Master
-----
ABC                 PC17
DOMAIN              24-DLQEECHSOERL
SZCEAC              PC22
WORKGROUP            DZYL11

```

(2) 下载文件夹 Web 里的文件。

```

[root@redflag /root]# smbclient //lgm/web
added interface ip=192.168.100.33 bcast=192.168.100.255 nmask=255.255.255.0
Got a positive name query response from 192.168.100.39 ( 192.168.100.39 )
Password:
Domain=[WORKGROUP] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> ?

```

“ ? ” 查看可以使用的命令。

?	altname	archive	blocksize	cancel
cd	chmod	chown	del	dir
du	exit	get	help	history
lcd	link	lowercase	ls	mask
md	mget	mkdir	more	mput
newer	open	print	printmode	prompt
put	pwd	q	queue	rmdir
rd	recurse	rename	rm	translate
setmode	symlink	tar	tarmode	!

```

smb: \> ls
.                D            0            Sat Apr            5 01:43:56 2003
..               D            0            Sat Apr            5 01:43:56 2003
BOOTEX.LOG       A            1394       Wed Dec            25 08:37:26 2002
WWW1.txt         A            59398      Wed Dec            18 02:52:14 2003

64007 blocks of size 32768. 18967 blocks available

```

```
smb: \> get bootex.log /root/bootex.log
```

#下载文件。

```
getting file /root/bootex.log of size 1394 as /root/bootex.log (194.5 kb/s) (average 194.5 kb/s)
```

(3) 使用 smbmount 安装 Samba 文件系统。

```
[root@redflag /root]# smbmount //lgm/web /mnt/samba
```

Password:

安装成功之后，就可以像使用本地的文件一样。

【注意】 在使用 smbmount 和 smbclient 命令的时候都需要输入密码，该密码是 Windows 2000 本机已经登录用户的密码，而在 Windows 98 里应该是设置共享时的密码。

10.4.2 从 Windows 机上访问 Linux 资源

从 Windows 机上访问 Linux 资源相对来说比较简单，本节仅通过实例来讲述如何从 Windows 机上访问 Linux 资源。

【实例 10.7】

本实例中 Windows 2000 的计算机的 NetBIOS 名称为 lgm，Linux 计算机的 NetBIOS 名称为 Redflag，则整个访问操作如下：

(1) 在 Windows 2000 的计算机的桌面上点击“网上邻居”，进入 Workgroup 组，其成员如图 10-6 所示，找到 Linux 的主机 Redflag。

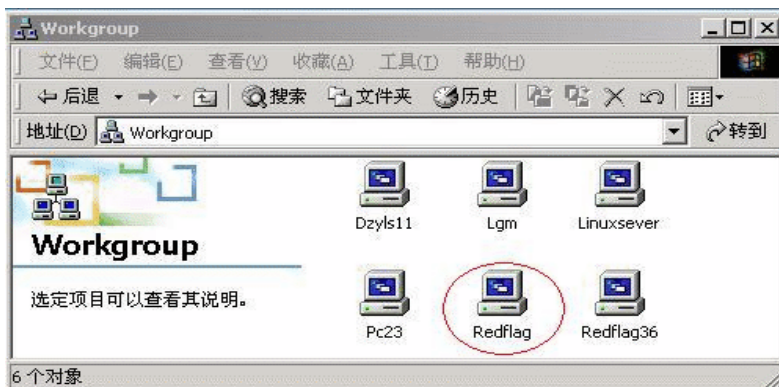


图 10-6 Windows 2000 的计算机 Workgroup 组成员

(2) 点击 Redflag，出现要求输入用户名和密码的对话框，如图 10-7 所示。



图 10-7 访问 Samba 服务器时输入用户名和密码的对话框

(3) 如果输入用户名和密码正确的话，则可以看到 Redflag 机上共享的资源，如图 10-8 所示。

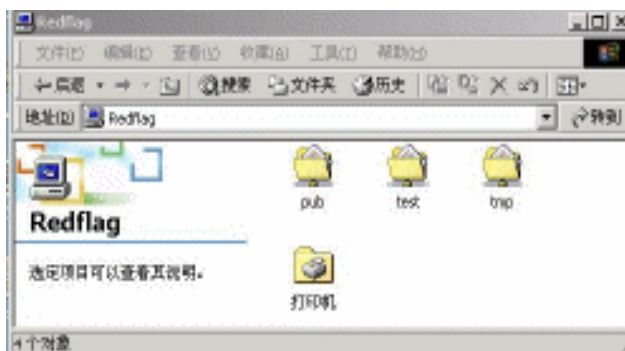


图 10-8 Redflag 机上共享的资源

接下来用户可以按照自己的需要进行相应的操作。

【技巧】 在进入 pub 目录后，为什么不能访问更深层的目录，而在设置 Samba 共享时已经设置了 “browsable = yes” 和 “writable = yes” ？

由于赋予用户对共享资源段的访问权限是基于该用户在 Linux 系统的对该资源的访问权限，所以服务器是不能赋予超过 Linux 系统赋予该用户的权限的。赋予权限可以用下面的命令来实现：

```
[root@redflag /root]#chmod -R 666 /home/pub
```

10.4.3 Linux 和 Windows 互发短消息

相信大家都使用过 Windows 98 的 “winpopup.exe” 和 Windows 2000 的 “net send” 程序，它们是一个很方便的通信工具。在 Linux 里的 Samba 也可以处理 winpopup 的信息。

为了处理 winpopup 的信息，Samba 提供了 “message command” 全局参数，它定义了信息到来时 Samba 采取的措施，例如在 smb.conf 的 [global] 段加入下面两行：

```
message command = bash "kedit %s; rm %s" &
message command = /bin/mail -s message from %f on %m 'root < %s;rm %s&
```

【说明】 该命令必须立即返回，所以在其后加上符号 “&”。

宏 %s 表示存放信息的文件名。

宏 %f 表示发送信息的用户。

宏 %m 表示客户机的 NetBIOS 名字。

1. 从 Linux 向 Windows 发送短消息

我们用到的命令是 smbclient，操作如下：

```
[root@redflag /root]# echo "This is a test!" | smbclient -M lgm
```

其中 “lgm” 表示 Windows 的 NetBIOS 名字。

(1) 在 Windows 98 下首先要运行 winpopup.exe 程序，接下来才能接收到短消息，如图 10-9 所示。

(2) 在 Windows 2000 下，可以直接接收短消息，如图 10-10 所示。



图 10-9 Windows 98 接收到 Linux 发送的信息

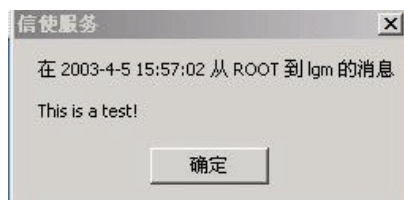


图 10-10 Windows 2000 接收到 Linux 发送的信息

2. 从 Windows 向 Linux 发送短信息

(1) 在 Windows 98 下向 Linux 发送短信息时，仍然要用 winpopup.exe 程序，但是对象应该是 Linux 下的 NetBIOS 名称才可以，如图 10-11 所示。

(2) 在 Windows 2000 下用 net send 命令向 Linux 发送短信息，如图 10-12 所示。

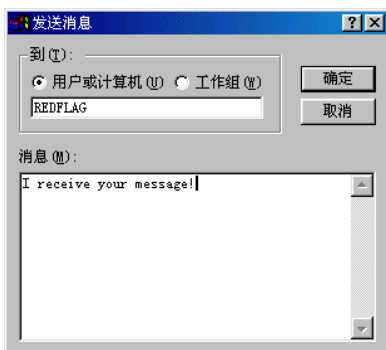


图 10-11 在 Windows 98 下回复 Linux 短信

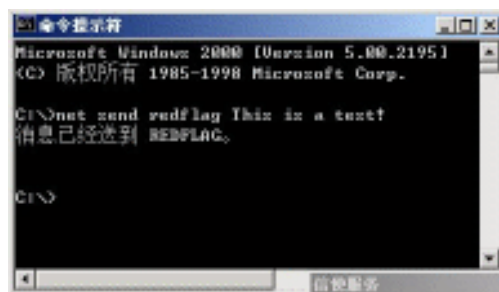


图 10-12 Windows 2000 下用 net send 命令向 Linux 发送短信息

(3) 在 Linux 下可以通过 mail 命令来查看收到的信息。

```
[root@redflag /root]#mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/root": 1 message 1 new
>N 1 nobody@localhost.loc Sat Apr 5 15:56 13/442 "message from LGM on l"
& 1
Message 1:
From nobody Sat Apr 5 15:56:14 2003
Date: Sat, 5 Apr 2003 15:56:14 -0500
From: Nobody <nobody@localhost.localdomain>
To: root@localhost.localdomain
Subject: message from LGM on lgm

This is a test!
```

10.5 Samba 组件中的应用程序

Samba 组件中的应用程序有很多，本节只简单的介绍几个程序。

10.5.1 报告 Samba 状态

smbstatus 列出当前所有的 Samba 连接，该命令执行如下：

```
[root@redflag /root]# smbstatus
Samba version 2.2.5

Service      uid          gid          pid          machine
-----
IPC$         test        test        2034         lgm        (192.168.100.39) Sat Apr  5 16:08:47 2003

No locked files
```

10.5.2 基于 Web 的配置工具——SWAT

SWAT 是 Samba Web Administration Tool 的缩写，它允许用户通过 Web 配置 Samba。实现 SWAT 的过程如下：

(1) 修改/etc/xinetd.d/swat 文件。修改后的 SWAT 文件内容如下：

```
# default: off
# description: SWAT is the Samba Web Admin Tool. Use swat \
#              to configure your Samba server. To use SWAT, \
#              connect to port 901 with your favorite web browser.
service swat
{
    port                = 901
    socket_type         = stream
    wait                = no
    #    only_from        = 127.0.0.1
    #    from              = 192.168.100.0 255.255.255.0
    #允许 192.168.100.0 网段上的所有机器通过 Web 配置 Samba。
    user                = root
    server              = /usr/sbin/swat
    log_on_failure      += USERID
    disable             = no
    #默认时，“disable = yes”，要想启动 SWAT，所以这里一定设置为“no”。
}
```

(2) 查看/etc/services 文件，保证 901 端口启动。因为 SWAT 使用的是 901 端口，所以该端口一定要打开，在文件/etc/services 中的内容如下：

```
swat 901/tcp
```

(3) 启动超级网络服务。

启动。

```
[root@redflag /root]#/etc/rc.d/init.d/xinetd start
```

或

```
[root@redflag /root]#service xinetd start
```

重启动。

```
[root@redflag /root]#/etc/rc.d/init.d/xinetd restart
```

或

```
[root@redflag /root]#service xinetd restart
```

关闭。

```
[root@redflag /root]#/etc/rc.d/init.d/xinetd stop
```

或

```
[root@redflag /root]#service xinetd stop
```

(4) 使用 SWAT。

在浏览器输入 `http://192.168.100.33:901`，SWAT 首先要求用户输入账号和口令，一般可以用 root 账号登录，登录完成就进入了配置页面，结果如图 10-13 所示。

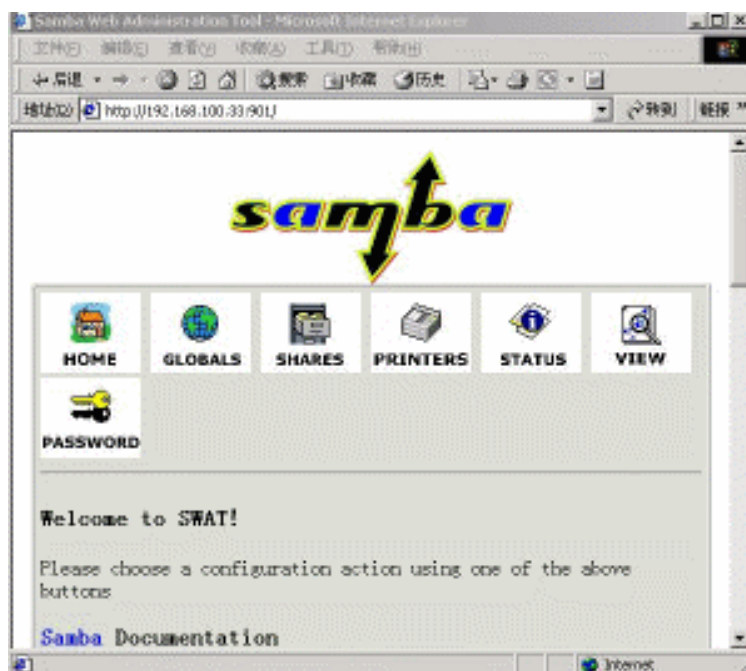


图 10-13 SWAT 配置页面

SWAT 提供了比较完整的帮助信息，而且也是非常好的配置工具，可以进行一些简单的配置。SWAT 虽然使用很方便，但是功能却没有那么强大，某些应用仍需要直接手工修改配置文件。

10.6 Samba 常见故障排除

Samba 的故障排除可以从服务器和客户机上分别来完成。

10.6.1 Samba 服务器上的故障排除

1. 用 testparm 命令检查语法错误

【实例 10.8】

本实例是用 testparm 检查 192.168.100.33 机的 Samba 的配置语法，命令如下：

```
[root@redflag /root]#testparm
```

结果如下：

```
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Loaded services file OK.
Press enter to see a dump of your service definitions
# Global parameters
[global]
netbios name = REDFLAG
workgroup = WORKGROUP
:
[homes]
comment = Home Directories
:
```

通过该命令我们可以查看到相应的服务定义，并且根据提示及时更改错误的语法。

2. 检查相关的参数

因为 Samba 的许多参数是非常重要的，如果设置不正确的话，Windows 和 Linux 将不能通信。这些参数包括“hosts allow”、“hosts deny”、“valid users”、“invalid users”等和权限相关的参数。

3. 及时重新启动 Samba

用户在修改完配置文件后一定要重新启动才能生效。

4. 检查权限的设定

在进行共享目录的权限的设置时，一定要考虑清楚，合理的利用 read、writable、guest、browsable 等。

5. 在 Linux 机上使用 smbclient 失败

在 Windows 2000 下要将 guest 账号起用。

10.6.2 Samba 客户机上的故障排除

(1) 在 Windows 98 或 Windows 2000 上看不到 Linux 的机器。这时候可以用搜索 Linux 机器 IP 地址的方法来解决。

(2) 在 Windows 端得到的权限不是理想的权限。如果在服务器上设置了“map to guest = Bad Password”，而且恰恰用户输入了错误的口令，服务器会“偷偷地”将它映射为 guest 用户，而不给出任何错误提示信息，这样该用户会在不知道的情况下受到种种访问控制。

本章小结

Samba 是目前应用最为广泛的网络服务之一，其最常用的应用是作为文件服务器和打印服务器。本章从 Samba 的概念入手，主要介绍了 Samba 配置、访问控制、加密口令使用、Windows 和 Linux 之间通信等内容。Samba 的功能非常之强大，随着读者学习的深入和对 Samba 应用的熟练，更能体会到这一点。

习 题

1. Samba 的配置文件是_____。
A. smb.conf B. samba.conf C. smbpasswd D. smbclient
2. 我们可以用_____命令对 Samba 的配置文件进行语法测试。
A. smbclient B. smbpasswd C. testparm D. smbmount
3. 在 smb.conf 文件中有如下的表达 %m，该表达的含义是_____。
A. 当前服务使用的用户名 B. 当前共享的名称
C. 当前会话使用的用户名 D. 客户机的 NetBIOS 名
4. _____是 Samba 服务器的默认安全级别。
A. share B. user C. server D. domain
5. 我们可以通过设置条目_____来控制可以访问 Samba 共享服务的合法主机名。
A. allow hosts B. valid hosts C. allow D. public
6. 为了使用户在提供了正确的用户名而密码不正确的情况下，也能登录使用 Samba 资源，需要设置_____参数。
A. map to guest = Never B. map to guest = Bad User
C. map to guest = Bad Password C. map to guest = Only
7. 如果不清楚一个 Windows 计算机的 NetBIOS 名字，smbclient 可以使用_____选项指定要访问计算机的 IP 地址，这时 NetBIOS 的名字将被忽略。
A. -N B. -I C. -L D. -P
8. Samba 的配置文件不包括_____。
A. global(全局)参数 B. directory shares (目录共享)部分
C. printer shares(打印共享)部分 D. application shares(应用程序共享)部分

9. 使用 Samba 服务器，一般来说，可以提供打印服务和_____。
- A. 域名服务 B. 文件服务 C. IP 地址解析 D. Web 服务
10. _____允许用户通过 Web 配置 Samba。
- A. SWAT B. xinetd C. inetd D. SSL
11. _____是 Samba 核心的两个后台进程。
- A. smbd 和 nmbd B. nmbd 和 inetd C. inetd 和 smbd D. inetd 和 httpd
12. _____命令可以修改 Samba 用户的口令。
- A. smbpasswd B. passwd C. mkpasswd D. password

第 11 章 域名系统



DNS 是域名系统(Domain Name System)的缩写,在一个 TCP/IP 架构的网络(例如 Internet)环境中,DNS 是一个非常重要而且常用的系统,其主要功能就是将人易于记忆的 Domain Name 与人不容易记忆的 IP Address 作转换。而上面执行 DNS 服务的这台网络主机,就可以称之为 DNS Server。DNS 最常见的版本是 BIND——伯克利 Internet 域名服务器,该服务器称为 named。

11.1 DNS 简介

11.1.1 概述

随着互联网在世界范围的快速发展,网络已经走进人们的生活。在 TCP/IP 网络上,每个设备必须分配一个惟一的地址,计算机在网络上通讯时只能识别如“202.96.134.163”之类的数字地址,而人们在使用网络资源的时候,为了便于记忆和理解,更倾向于使用有代表意义的名称,即域名系统 DNS,如 www.szptt.net.cn 代表深圳之窗网站的域名,这就是为什么当我们打开浏览器,在地址栏中输入如“www.szptt.net.cn”的域名后,就能看到我们所需要的页面。这是因为在我们输入域名后,有一台称为“DNS 服务器”的计算机自动把我们的域名“翻译”成了相应的 IP 地址,然后调出那个 IP 地址所对应的网页,最后再传回给我们的浏览器,我们才能得到结果。

在早期的 IP 网络世界里,每台电脑都只用 IP 位址来表示,不久人们就发现这样很难记忆。于是,一些 Unix 的管理者就建立一个 HOSTS 对应表,将 IP 和主机名字对应起来,只要用户输入主机的名字,计算机就可以将该名字转换成机器能够识别的 IP 地址。但是随着 Internet 规模的不断扩大,这种做法显然是不可行的。为了解决这个问题,1983 年,Internet 开始采用域名系统 DNS。

DNS 的核心思想是分级的,它主要用于将主机名和电子邮件映射成 IP 地址。那么 DNS 是怎么运作的?DNS 是使用层的方式来运作的。有的组织有其自己的 DNS 服务器,这个 DNS 服务器维护着所在域的主机和 IP 地址的映射记录。当请求名称解析时,DNS 服务器先在自己的记录中检查是否有对应的 IP 地址。如果未找到,它就会向其他 DNS 服务器询问该信息。DNS 解析程序的查询流程如图 11-1 所示。

例如,某单位的 Domain Name 为 www.szpt.edu.cn,这个 Domain Name 当然不是凭空而来的,是从“.edu.cn”所委派下来的。“ .edu.cn”又是从“.cn”委派(delegation)的。“ .cn”是从哪里来的呢?答案是从“.”来的,也就是所谓的“根域”(Root Domain)来的。根域已经是 Domain Name 的最上层。而“.”这层是由 InterNIC(Internet Network Information Center,

互联网信息中心)所管理的。全世界的 Domain Name 就是这样，一层一层地委派下来。

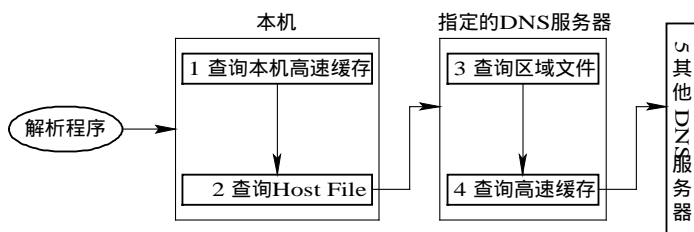


图 11-1 DNS 解析程序的查询流程

【实例 11.1】

DNS 是怎么查出域名 `www.szpt.edu.cn` 的 IP 地址的？

(1) 你所用的电脑(可能是 PC ,也可能是工作站)送出一个问题给这台电脑所设定的 DNS Server，问：`www.szpt.edu.cn` 的 IP 是什么？

(2) 这台 DNS 会先看看是不是在它的 cache 中，如果是，就给出答案；如果不是，就从最上头查起。在 DNS Server 上面一定有设定“.”，所以，这个时候它就往“.”层的任何一台 DNS(目前“.”有 13 台)问：“.cn”要问谁？

(3) “.”层的 DNS 会回答“.cn”要向谁去查询(同时你用的 DNS 会 cache 起来这个答案)。

(4) 接下来你所用的 DNS 就会向“.cn”这层的 DNS 问：“.edu.cn”要问谁？

(5) “.cn”的这层就会回答“.edu.cn”要向谁查(同时你用的 DNS 也把该答案 cache 起来)。

(6) 直到 `www.szpt.edu.cn` 回答 `www.szpt.edu.cn` 的 IP 是 61.144.120.117(又 cache 起来)。

经过了这么多的过程，终于得到了这个 IP，接下来才能作进一步的连线。要注意的是，在每一层都会问一个问题，并且把答案记下来(cache 起来)，而且还会忘掉(看该层的设定是要 cache 多久)。

11.1.2 DNS 结构

为了方便管理及确保网络上每台主机的域名绝对不会重复，因此整个 DNS 结构就设计成 4 层，分别是根域、顶层域、第二层域和主机。

1. 根域

这是 DNS 的最上层，当下层的任何一台 DNS 服务器无法解析某个 DNS 名称时，便可以向根域的 DNS 寻求协助。理论上，只要所查找的主机有按规定注册，那么无论它位于何处，从根域的 DNS 服务器往下层查找，一定可以解析出它的 IP 地址。

2. 顶层域

这一层的命名方式有争议。在美国以外的国家，大多数以 ISO3116 所定制的国码来区分，例如，cn 为中国，jp 为日本，hk 为香港等，如图 11-2 所示。

但是在美国，虽然它也有 us，但却很少用来当成顶级域名，反而是以组织性质来区分，如图 11-3 所示。

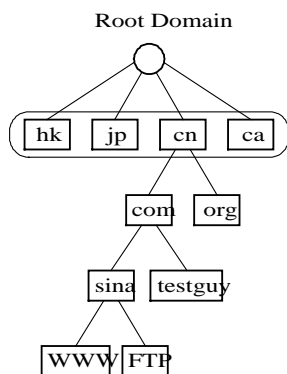


图 11-2 DNS 顶级域名以国码命名

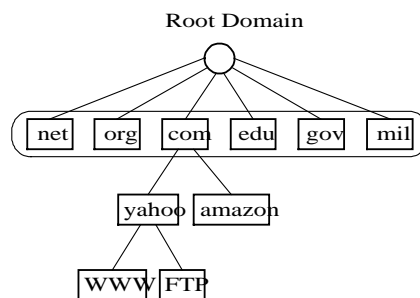


图 11-3 DNS 顶级域名以组织性质命名

其中：

com：表示商业组织。

edu：表示教育机构。

org：表示非赢利组织。

net：表示计算机网络组织。

gov：表示美国政府组织。

mil：表示军事部门。

3. 第二层域

第二层域可以说是整个 DNS 系统中最重要的部分，在这些域名之下都可以开放给所有人申请，名称则由申请者自己定义，例如，“.szpt.edu.cn”。

4. 主机

最后一层是主机，也就是隶属于第二层域的主机，这一层是由各个域的管理员自行建立，不需要通过管理域名的机构。例如，我们可以在“.szpt.edu.cn”这个域下再建立“www.szpt.edu.cn”、“ftp.szpt.edu.cn”等主机。

【注意】域名有相对域名和绝对域名之分。相对域名就是指在某一级域名下面的下属域名。例如，上面所说的 szpt 是 edu 下属的一个相对域名，而 ftp 又是 szpt 下属的一个相对域名。绝对域名是完整的域名，一直写出根域名，例如上面所说的 www.szpt.edu.cn 就是一个绝对域名。绝对域名有时也称为完全合格的域名 FQDN(Full Qualified Domain Name)。在局部范围使用相对域名更为方便。

11.1.3 资源记录

如前所述，每个 DNS 数据库都由资源记录构成。一般来说，资源记录包含与特定主机有关的信息，如 IP 地址、主机的所有者或者提供服务的类型，当进行 DNS 解析时，它取回的是与该域名相关的资源记录。

数据库文件的每一行都由一条资源记录组成，可以通过普通的编辑器 vi 来修改，而每个资源记录通常包含五项，大多数情况下用 ASCII 文本显示，每条记录一行，格式如下：

Domain Time to live Record type Class Record data

各项的含义如下：

(1) 域名(Domain)：该项给出要定义的资源记录的域名，该域通常用来作为域名查询时的关键字。

(2) 存活期(Time to live)：在该存活期过后，该记录不再有效。

(3) 类别(Class)：该项说明网络类型。目前大部分的资源记录都采用“IN”，表明 Internet，该域的缺省值为“IN”。

(4) 记录数据(Record data)：说明和该资源记录相关的信息，通常由资源记录类型来决定。

(5) 记录类型(Record type)：该项说明资源记录的类型，常用的资源记录类型如表 11-1 所示。

表 11-1 常用的资源记录类型

记录类型	说 明
A	将主机名转换为地址
CNAME	指定主机的别名
MX	建立邮件服务器记录
NS	域名服务器
PTR	将地址变成主机名，主机名必须是规范主机名
SOA	起始授权机构，此记录指定区域的起点
HINFO	主机描述，是以 ASCII 码表示的 CPU 和 OS
TXT	定义一个 zone，所有的定义只与该 zone 有关

具体解释如下：

(1) A：此记录列出特定主机名的 IP 地址。这是域名解析的重要记录。

(2) CNAME：此记录指定标准主机名的别名。

(3) MX：建立邮件服务器记录，此记录列出了负责接收发到域中的电子邮件的主机。

(4) NS：此记录指定负责给定区域的名称服务器。

(5) PTR：反向地址解析。

(6) SOA：它所包含的信息有域名、管理员电子邮件地址，以及指示辅助 DNS 服务器如何更新区域数据文件的设置等。

(7) HINFO：该记录允许人们找出一个域内相应的机器和操作系统类型。

(8) TXT：该记录允许域以任意方式标识自身，HINFO 和 TXT 两种记录类型都是为了用户的方便，其中任何一条都不是必要的。

11.1.4 域名服务器分类

从理论上讲，一台域名服务器就可以包括多个整个 DNS 数据库，并响应所有的查询，但是这样的话，服务器会由于负载过重而一无用处。另外，服务器失效，整个互联网就会崩溃。

域名服务器不仅应该能够进行一些域名到 IP 地址的转换,而且还必须具有与其他域名服务器联系的信息。这样,当自己不能进行域名到 IP 地址的转换时,就能够知道到什么地方找别的域名服务器,这样就使得这些域名服务器组成一个大的域名服务系统。Internet 上的域名服务器系统也是按照域名的层次来划分的。每个域名服务器都只对域名体系中的一部分进行管辖,一个独立管理的 DNS 子树称为一个区域(zone)。一个常见的区域是一个二级域,如 edu.cn。许多二级域将它们的区域划分成更小的区域。例如,大学可能根据不同的学院来划分区域,公司可能根据不同的部门来划分区域。

常见的域名种类服务器有如下五种。

1. 根服务器(Root Servers)

根域名服务器位于域名空间的最顶层,也就是“.”,它们主要用来管理根域和顶级域名。目前有 13 个根域名服务器,这些根域名服务器都由 NIC 维护,它的主机名通常为“a.root-servers.net”,一直到“m.root-servers.net”。最新的根域名服务器列表可以从 <ftp://ftp.rs.internic.net/domain/named.root> 获得。

2. 主域名服务器(Primary Servers)

通常每个区域有且只有一个主域名服务器。虽然 DNS 规则中没有明确禁止使用多个主域名服务器,但是由于维护多个主域名服务器一般来说是比较困难的,而且很容易产生错误,所以不鼓励一个域有多个主域名服务器。对每个区域的所有 DNS 数据库文件的修改都在该区域的主域名服务器上修改。主域名服务器对该域中的辅助域名服务器进行周期性的更新。

从 BIND 8.1.2 开始,主域名服务器在 DNS 配置文件/etc/named.conf 中由“zone”语句中的“type master”变量来定义。

3. 辅助域名服务器(Secondary Servers)

辅助域名服务器用作同一区域中主服务器的备份服务器,以防主服务器无法访问或宕机。辅助域名服务器定期与主域名服务器通讯,确保它的区域信息保持最新。如果不是最新信息,辅助域名服务器就会从主服务器获取最新区域数据文件的副本。这种将区域文件复制到多台域名服务器的过程称为区域复制。

4. 专用缓存域名服务器(Cache-only Servers)

所有的域名服务器都缓存非它们授权管理的远地域名的信息。而专用缓存服务器只用来缓存任何 DNS 域的信息,它们不管理任何授权的域名信息,所以它们对任何域提供的信息都是非授权的。专用缓存服务器可以分担辅助域名服务器从主域名服务器获得数据库文件拷贝的负担,可以为用户提供本地的域名信息服务而不用设置主域名服务器和辅助域名服务器。

5. 转发域名服务器(Forwarding Servers)

转发域名服务器是主域名服务器和辅助域名服务器的一种变形,它负责所有非本地域名的本地查询。如果用户定义了一台转发域名服务器,那么所有对非本地域名的查询都首先发送给它。转发域名服务器通常有大量的域名缓存信息,从而可以减少非本地域名查询

的重复次数。转发域名服务器不能解析域名查询，本地域名服务器仍然可以访问远地的域名服务器。

从 BIND 8.1.2 开始，转发域名服务器在 DNS 配置文件/etc/named.conf 中由“option”语句来定义。

11.2 DNS 域名解析

DNS 域名解析通常在用户输入一些命令之后发生，例如输入命令“ftp ftp.szpt.edu.cn”，这时客户机首先要从 DNS 服务器获得 ftp.szpt.edu.cn 的 IP 地址，这一过程就是所谓的域名解析。

11.2.1 客户解析过程调用

域名解析过程开始于客户方的解析过程调用。解析过程调用通常已经被构建到系统内部，应用程序通过系统调用来访问它们。

客户解析过程有以下特点：

- (1) 它只查询在/etc/resolv.conf 文件中指定的域名服务器。

【实例 11.2】

下面是某/etc/resolv.conf 文件的内容：

```
domain abc.com
nameserver 202.96.134.133
search abc.com
```

- (2) 它是由在 DNS 服务器中的文件/etc/nsswitch.conf 中的“hosts”项中的一个关联激活的。

【实例 11.3】

下面是某/etc/nsswitch.conf 文件的部分内容：

```
# /etc/nsswitch.conf
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
:
:
#hosts:      db files nisplus nis dns
hosts:      files dns nisplus nis
```

```
.....
.....
```

在该文件中“hosts: files dns nisplus nis”行规定了当主机上的应用程序进行域名解析时的顺序。

11.2.2 域名解析过程

本节将用一个具体的实例来说明域名解析过程。

【实例 11.4】

如果用户发出“ftp ftp.szpt.edu.cn”，那么相应的域名解析过程如下：

(1) 客户系统查询/etc/nsswitch.conf 文件来决定域名解析的顺序，在这个例子中，解析的顺序为：先查/etc/hosts 文件，然后查询 DNS。

(2) 如果客户方没有在/etc/hosts 里找到相应的项，它就查看/etc/resolv.conf 来确定域名解析的查询顺序，以及所指定的本地域名服务器的 IP 地址。

(3) 客户系统的解析过程调用发送一个查询给本地的 DNS，等待域名服务器返回其 IP 地址。

(4) 本地域名服务器查询本机缓存的信息，看该域名最近是否被查询过，如果是，它就返回一个应答给客户。

(5) 如果在本地缓存中没有相应的记录，那么本地的 DNS 会询问其他 DNS，或直接向根域名服务器发送一个递归查询。在本例中，假设缓存中没有相应的记录，那么就访问一台根域名服务器。

(6) 根域名服务器返回它所知道的有关“.cn”的域名服务器和它们的 IP 地址。

(7) 本地域名服务器和一台“edu.cn”的域名服务器建立联系，发送和以前一样的查询请求。

(8) “edu.cn”域名服务器返回“szpt.edu.cn”域名服务器的地址。

(9) 本地域名服务器和一台“szpt.edu.cn”域名服务器建立连接，发送和以前一样的查询请求。

(10) “szpt.edu.cn”域名服务器返回“ftp.szpt.edu.cn”的 IP 地址，由本地域名服务器将该查询结果返回给客户。

这样，客户系统获得相应的 IP 后，ftp 命令继续执行。

11.3 DNS 配置

11.3.1 BIND 及其主要配置文件

DNS 最常用的版本是 BIND——伯克利 Internet 域名服务器，该服务器称作 named。BIND 是免费软件，它的版本处于不断更新和完善过程中，从 <http://www.isc.org/products/bind> 站点可以获得它的最新的源程序和相关的文档。大部分的 Linux 发行套件都包括 named。

BIND 的主要配置包括几个相关的文件，本节将结合一个具体的实例来说明 DNS 服务器的配置过程。

我们假定用户建立的 DNS 服务器所管辖的域名为 redflag.com，对应的子网 IP 地址是 10.1.14.0，域名服务器的 IP 地址为 10.1.14.61，配置过程如下。

1. 配置启动文件/etc/named.conf

该文件是域名服务器守护进程 named 启动时读取到内存的第一个文件。在该文件中定

义了域名服务器的类型、所授权管理的域以及相应数据库文件和其所在的目录。该文件我们分为两个部分讲述，第一部分的内容如下：

```
//file /etc/named.conf
options {
    directory "/var/named";
    // query-source address * port 53;
    notify no;
    forwarders{
        202.96.134.133;
    };
};
zone "." IN {
    type hint;
    file "named.ca";
};
zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.local";
    allow-update { none; };
};
```

【说明】这一部分是在安装了 bind 软件后系统自动产生的 named.conf 文件的原始内容。

“directory”指定了 DNS 记录文件的存放目录是/var/named。

“//”开头的文字是说明文字。

第一个“zone”语句定义了 DNS 服务器的根域名服务器的信息是从“name.ca”获得。这个记录文件是系统自带的，不用去改动它。

第二个“zone”语句是“回送地址”的数据库文件，文件名是“named.local”。这个记录文件也是系统自带的，不用去改动它。

named.conf 文件的这一部分内容，不用去改动它。

但我們可以在“diretory”下添加一行：forwarders {202.96.134.133;}，其中 202.96.134.133 是电信的 DNS 服务器的地址，forwarders 参数指明了其后的 IP 所在的服务器作为备选的 DNS 服务器。也就是说，把本机的 DNS 不能解析的主机发送到这个备选的 DNS 服务器上，让它来进行解析。

第二部分的内容如下：

```
zone "redflag.com" IN {
    type master;
    file "named.hosts";
    allow-update { none; };
};
```

```
zone "14.1.10.in-addr.arpa" IN {
type master;
file "named.10.1.14 ";
allow-update { none; };
};
```

【说明】 这一部分是手工添加的。

定义了“redflag.com”这个域和对应的反向查询域。

“type master”说明本机是“redflag.com”和“1.14.10.in-addr.arpa”这两个域的主 DNS 服务器(Primary Name Server)。

“named.hosts”和“named.10.1.14”是域的记录文件。这两个文件也是自己创建的。

在 named.conf 文件中涉及到的文件名当然也可以自己命名，但是要和/var/named 目录下的文件名保持一致。

在行“zone “14.1.10.in-addr.arpa” IN { ”行中，网络号一定反序写出。

2. 创建/var/named/named.ca

在 Linux 系统上，通常在/var/named 目录下已经提供了一个 named.ca 文件，该文件中包含了 Internet 的顶层域名服务器，但这个文件通常会有变化，所以建议最好从 InterNIC 下载最新的版本。该文件可以通过匿名 FTP 下载。该文件的内容如下：

```
;      This file holds the information on root name servers needed to
;      initialize cache of Internet domain name servers
;      (e.g. reference this file in the "cache . <file>"
;      configuration file of BIND domain name servers).
;
;      This file is made available by InterNIC registration services
;      under anonymous FTP as
;          file           /domain/named.root
;          on server      FTP.RS.INTERNIC.NET
;      -OR- under Gopher at RS.INTERNIC.NET
;          under menu     InterNIC Registration Services (NSI)
;          submenu        InterNIC Registration Archives
;          file           named.root
;
;      last update:      Aug 22, 1997
;      related version of root zone:  1997082200
;
;
;      formerly NS.INTERNIC.NET
;
.          3600000   IN   NS       A.ROOT-SERVERS.NET.
```

```

A.ROOT-SERVERS.NET.      3600000      A      198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                          3600000      NS      B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A      128.9.0.107
;
; formerly C.PSI.NET
;
.                          3600000      NS      C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A      192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                          3600000      NS      D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A      128.8.10.90
;
; formerly NS.NASA.GOV
;
.                          3600000      NS      E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A      192.203.230.10
;
; formerly NS.ISC.ORG
;
.                          3600000      NS      F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A      192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.                          3600000      NS      G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A      192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                          3600000      NS      H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A      128.63.2.53
;
; formerly NIC.NORDU.NET
;

```



```
.                3600000      NS      I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.  3600000      A       192.36.148.17
;
; temporarily housed at NSI (InterNIC)
;
.                3600000      NS      J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.  3600000      A       198.41.0.10
;
; housed in LINX, operated by RIPE NCC
;
.                3600000      NS      K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.  3600000      A       193.0.14.129
;
; temporarily housed at ISI (IANA)
;
.                3600000      NS      L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.  3600000      A       198.32.64.12
;
; housed in Japan, operated by WIDE
;
.                3600000      NS      M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.  3600000      A       202.12.27.33
; End of File
```

【说明】 以 “ ; ” 开始的行都是注释行。

其他每两行都和某个域名服务器有关，分别是 “ NS ” 和 “ A ” 资源记录。

行 “ .3600000 IN NS A.ROOT-SERVERS.NET. ” 的含义是：“ . ” 表示根域；“ 3600000 ” 是存活期；“ IN ” 是资源记录的网络类型，表示 Internet 类型；“ NS ” 表示该资源记录的类型；最后是主机域名，为 “ A.ROOT-SERVERS.NET. ”。

行 “ A.ROOT-SERVERS.NET.3600000 A 198.41.0.4 ” 的含义是：“ A ” 资源记录用来指定根域名服务器对应的 IP 地址，“ A.ROOT-SERVERS.NET. ” 为主机名；“ 3600000 ” 是存活期；“ A ” 表示资源记录的类型；最后对应的是 IP 地址。

其他行的含义基本同 和 。

3. 创建/var/named/named.hosts

该文件指定了域中主机域名同 IP 地址的映射。该文件的内容如下：

```
;file named.hosts
$TTL 86400
@      IN  SOA a100.redflag.com.  root.redflag.com. (
                2001110600 ; serial
                28800 ; refresh
```

```
14400 ; retry
3600000 ; expire
86400 ; minimum
)
IN   NS   a100.redflag.com.
IN   MX   10   a100.redflag.com.
```

```
localhost. IN   A   127.0.0.1
a100      IN   A   10.1.14.61
a101      IN   A   10.1.14.62
a102      IN   A   10.1.16.63
a103      IN   A   10.1.14.64
a104      IN   A   10.1.14.65
www       IN   CNAME a100
ftp       IN   CNAME a100
```

【说明】 文件中所有的记录行(本文件从第 10 行开始)都要顶行写,前面不能有空格。

“ ; ”引导的是注释行。

第一个有效的行是“SOA”记录,这是每个数据库文件都必须有的。该记录的各项含义如下:

? “@”表示该域的替代符,在该文件中表示“redflag.com”。

? “IN”表示网络类型。

? “SOA”表示该记录的类型。

? 最后是该数据的数据域。第一项是主域名服务器的完全标识域名,这个域名不能是别名;第二项是管理员的电子邮件地址,它是正常的 E-mail 地址的变通,将“@”改为“.”;第三项是序列号,该数据是与辅助域名服务器和主域名服务器进行时间同步的。每次修改数据库文件后,都要将该序号更新。习惯上用“yymmddnn”,年月日后加两位数字,表示一日之中的第几次修改;第四项是更新时间。辅助域名服务器根据该时间间隔周期性地检查主域名服务器的序列号是否已经改变,如果改变,需要取得改变后的数据库文件;第五项是重试时间间隔。当辅助域名服务器没有能够成功从主域名服务器获得数据库文件后,在定义的时间间隔后重新尝试;第六项是过期时间。如果辅助域名服务器在所定义的时间间隔内没有能够与主域名服务器或另一台辅助域名服务器取得联系,那么该域名服务器上的所有数据库文件都认为无效,不再响应查询请求;第七项是最小时间周期。这是缺省的资源记录的存活期,对于那些没有特别指定存活周期的资源记录,该值就成为它们的存活期。

行“IN NS a100.redflag.com.”说明该域的域名服务器,至少应该定义一个。

行“IN MX 10 a100.redflag.com.”是 MX 记录,该程序专门处理邮件地址的主机部分为“@redflag.com”的邮件,“10”表示优先级别。

类似行“a100 IN A 10.1.14.61”是一系列的“A”记录,表示主机名和 IP 地址的对应关系建立起来。“a100”是主机名,“10.1.14.61”是它的 IP 地址。

行“www IN CNAME a100”表示一条定义别名的记录。即“www.redflag.com”和“a100.redflag.com”表示同一台主机。

相对域名和绝对域名都可以在记录中使用。比如,“a100”和“a100.redflag.com.”的含义是相同的。如果写绝对域名,一定在最后边加上“.”。

4. 创建/var/named/named.10.1.14

该文件主要定义了 IP 地址到主机名的转换。IP 地址到主机名的转换是非常重要的,Internet 上很多应用,例如 NFS、Web 服务等都要用到该功能。该文件的内容如下:

```
; file named.10.1.14
$TTL 86400
@      IN  SOA a100.redflag.com.    root.a100.redflag.com. (
                                2001110600 ; serial
                                28800 ; refresh
                                14400 ; retry
                                3600000 ; expire
                                86400 ; minimum
                                )
      IN  NS  a100.redflag.com.
61     IN  PTR a100.redflag.com.
62     IN  PTR a101.redflag.com.
63     IN  PTR a102.redflag.com.
64     IN  PTR a103.redflag.com.
65     IN  PTR a104.redflag.com.
```

【说明】 除了注释行外,第一个仍然是 SOA 记录,这里的“@”代表 14.1.10.in-addr.arpa 域。

PTR 记录用于定义 IP 地址名到主机域名的映射。

PTR 使用的仍是相对域名,如“61”表示“61.14.1.10.in-addr.arpa”。

PTR 记录的最后一项必须是一个完整的标识域名,以“.”结束。

5. 创建/var/named/named.local

该文件用来说明“回送地址”的 IP 地址到主机名的映射。该文件的内容如下:

```
; file named.local
$TTL 86400
@      IN  SOA a100.redflag.com.    root.a100.redflag.com. (
                                2001110600 ; serial
                                28800 ; refresh
                                14400 ; retry
                                3600000 ; expire
                                86400 ; minimum
                                )
      IN  NS  a100.redflag.com.
1      IN  PTR localhost.
```

【说明】 此文件的内容是特定的，在不同的域的域名服务器上，所要修改的只是 SOA 记录和 NS 记录。

PTR 记录的最后域名为完全标识域名，以 “.” 结束。

11.3.2 相关配置文件

和域名服务器配置相关的文件主要包括如下两个文件。

1. /etc/resolv.conf

该文件用来告诉解析器调用的本地域名、域名查找的顺序以及要访问域名服务器的 IP 地址。该文件的内容如下：

```
domain redflag.com
nameserver 10.1.14.61
search redflag.com
```

【说明】 “domain” 项指出本地域名服务器。

“nameserver” 项指定了域名服务器的 IP 地址。

“search” 项说明当给出一个相对域名时，如 “a100”，附加在其后的域名，使之成为完全标识域名，即 “a100.redflag.com”。

2. /etc/nsswitch.conf

操作系统使用了很多关于主机、用户、组等信息的数据库。这些数据库的数据存在于不同的文件中。例如，主机名和主机 IP 地址可能存在于 /etc/hosts、NIS、NIS+ 或 DNS 中。对于每个数据库，可以使用多个源文件，这些源文件和它们的查询顺序就是在 /etc/nsswitch.conf 文件中指定的。该文件中 and 域名服务有关的一项是 “hosts”。该文件的内容如下：

```
#
# /etc/nsswitch.conf

passwd:      files nisplus nis
shadow:      files nisplus nis
group:       files nisplus nis

#hosts:      db files nisplus nis dns
hosts:       files dns nisplus nis
bootparams:  nisplus [NOTFOUND=return] files
ethers:      files
netmasks:   files
networks:    files
protocols:   files nisplus nis
rpc:         files
services:    files nisplus nis
netgroup:    files nisplus nis
```

```
publickey:      nisplus
automount:      files nisplus nis
aliases:        files nisplus
```

【说明】 “ hosts: files dns nisplus nis ” 行规定了当主机上的应用程序进行域名解析时的顺序。

解析时的顺序为：先查/etc/hosts 文件，再查 “ dns ” 文件，最后再查看 “ NIS+ ” 和 “ NIS ” 数据库文件。

当然用户可以根据自己的实际需要调整查找顺序。例如，如果主机上没有 “ NIS+ ” 和 “ NIS ” 信息服务，可以将该项改为：

```
hosts: files dns
```

11.3.3 DNS 的安全管理

由于 DNS 域名服务器处于 Internet 上，所以有可能受到入侵或非授权的访问。

1. 查询请求限制

在/etc/named.conf 中，可以用 “ allow-query ” 语句来限制提供所服务的 IP 地址范围。请看下面的这个例子：

```
options{
    allow-query{ 10.1.14.0/24;10.1.20.0/24};
}
```

在文件中加入上面的语句，就只允许位于子网 10.1.14 和 10.1.20 上的机器访问本域名服务器。“ 24 ” 表示子网掩码为 24 位。

2. 对特殊的域进行限制

可以通过在 “ zone ” 语句中使用 “ allow-query ” 语句对所定义的域进行限制。例如：

```
zone "redflag.com" IN {
    type master;
    file " named.hosts";
    allow-query { "10.1.14.0/24"; };
};
```

在这个例子中，只允许在 10.1.14.0/24 网络中的计算机对该域名服务器进行查询。

11.4 DNS 的启动、停止和测试

11.4.1 DNS 的启动和停止

按照上节所说的步骤设置好/etc/named.conf 文件中定义的几个文件后，就可以启动 DNS 进程。DNS 启动方法有自动和手工两种。如果系统启动是存在/etc/named.conf 文件，系统会自动启动域名服务器进程 named。

下面主要介绍手工方法，具体的命令如下：

(1) 启动。

```
[root@redflag /root]# /etc/rc.d/init.d/named start 或
```

```
[root@redflag /root]#service named start
```

(2) 重启。

```
[root@redflag /root]# /etc/rc.d/init.d/named restart 或
```

```
[root@redflag /root]#service named restart
```

(3) 停止。

```
[root@redflag /root]# /etc/rc.d/init.d/named stop 或
```

```
[root@redflag /root]#service named stop
```

(4) 查看状态。

```
[root@redflag /root]# service named status
```

【注意】 named 进程启动和运行的情况可以在/var/log/messages 文件中看到。

如果 DNS 启动成功的话，我们可以用“ping”命令来检查，操作如下：

```
[root@redflag /root]#ping -c 4 a100.redflag.com
```

```
PING a100.redflag.com (10.1.14.61) from 10.1.14.61 : 56(84) bytes of data.
```

```
64 bytes from hosta.redflag.com (10.1.14.61): icmp_seq=0 ttl=255 time=203 usec
```

```
64 bytes from hosta.redflag.com (10.1.14.61): icmp_seq=1 ttl=255 time=77 usec
```

```
64 bytes from hosta.redflag.com (10.1.14.61): icmp_seq=2 ttl=255 time=42 usec
```

```
64 bytes from hosta.redflag.com (10.1.14.61): icmp_seq=3 ttl=255 time=113 usec
```

```
--- a100.redflag.com ping statistics ---
```

```
4 packets transmitted, 4 packets received, 0% packet loss
```

```
round-trip min/avg/max/mdev = 0.042/0.108/0.203/0.061 ms
```

如果出现错误的话，则结果如下：

```
ping: unknown host a100.redflag.com
```

11.4.2 DNS 测试

配置好 DNS 并启动 named 进程后，应该对 DNS 进行测试。BIND 软件包提供了三个工具：nslookup、dig 和 host。其中最常用的是 nslookup。下面将分别使用这三个命令对 DNS 进行测试。

1. nslookup 查找命令

nslookup 命令用来向 Internet 域名服务器发出查询信息。它有两种模式：交互式和非交互式。交互模式允许使用者从名字服务器查询不同主机或域的信息，或者打印出一个域内的主机列表；非交互模式用于只打印一个主机或域的名字和所请求的信息。

当没有指定参数(使用缺省的域名服务器)或第一个参数是“_”，第二个参数为一个域名服务器的主机名或 IP 地址时，nslookup 为交互模式；当第一个参数是待查询的主机的域名

或 IP 地址时，nslookup 为非交互模式。这时，任选的第二个参数指定了一个域名服务器的主机名或 IP 地址。

下面通过实例使用交互模式对 DNS 进行测试。

【实例 11.5】

1) 检查正向 DNS

查找主机。

```
[root@redflag /root]#nslookup
>a100.redflag.com
Server:      10.1.14.61
Address:    10.1.14.61#53
```

```
Name:      a100.redflag.com
Address: 10.1.14.61
```

该命令用来查找主机 a100.redflag.com 的 IP 地址。

如果只输入：

```
>a100
```

nslookup 应该能够根据文件/etc/resolv.conf 中的定义，自动添加 redflag.com 的域名，然后显示与上面相同的结果。

查找域名信息。

```
[root@redflag /root]#nslookup
>set type=ns
>redflag.com
Server:      10.1.14.61
Address:    10.1.14.61#53
redflag.com    nameserver = a100.redflag.com.
```

【说明】“set type”表示设置查找的类型。

2) 检查反向 DNS

假如要查找 IP 地址为 10.1.14.61 的域名，输入：

```
[root@redflag /root]#nslookup
>set type=ptr
>10.1.14.61
Server:      10.1.14.61
Address:    10.1.14.61#53
61.14.1.10.in-addr.arpaname = a100.redflag.com.
```

3) 检查 MX 邮件记录

要查找 redflag.com 域的邮件记录地址，输入：

```
[root@redflag /root]#nslookup
>set type=mx
>redflag.com
```

```

Server:      10.1.14.61
Address:     10.1.14.61#53
redflag.com  mail exchanger = 10 a100.redflag.com.

```

4) 检查 CNAME 别名记录

要查找 www.redflag.com 主机的别名，输入：

```

[root@redflag /root]#nslookup
>set type=cname
>www.redflag.com
Server:      10.1.14.61
Address:     10.1.14.61#53
www.redflag.com  canonical name = a100.redflag.com.

```

2. dig 命令

dig(domain information groper)是一个灵活的命令行方式的域名查询命令，常用于从域名服务器获取特定的信息。

【实例 11.6】

通过 dig 可以查看域名 “a100.redflag.com” 的信息。

```

[root@redflag /root]#dig @redflag.com
; <<>> DiG 9.1.3 <<>> /a a100.redflag.com ns
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 35689
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
/a.                IN      A

;; AUTHORITY SECTION:
.                  10680   IN      SOA  A.ROOT-SERVERS.NET.  NSTLD.VERISIGN-GRS.COM.
2003031900 1800 900 604800 86400

;; Query time: 2 msec
;; SERVER: 10.1.14.61#53(10.1.14.61)
;; WHEN: Thu Mar 20 09:47:50 2003
;; MSG SIZE  rcvd: 95

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24283
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

```



```
;; QUESTION SECTION:
;a100.redflag.com.          IN      NS

;; AUTHORITY SECTION:
redflag.com.                86400   IN      SOA a100.redflag.com. root.redflag.com. 2001110600 28800
14400 3600000 86400

;; Query time: 1 msec
;; SERVER: 10.1.14.61#53(10.1.14.61)
;; WHEN: Thu Mar 20 09:47:50 2003
;; MSG SIZE rcvd: 75
```

dig 的命令的其他用法请参考 man pages，具体命令如下：

```
[root@redflag /root]#man dig
```

3. host 命令

host 命令用来做简单的主机名的信息查询，在缺省情况下，host 只在主机名和 IP 地址之间进行转换。

【实例 11.7】

查找 a101.redflag.com 主机的信息，操作如下：

```
[root@redflag /root]#host a101.redflag.com
a101.redflag.com has address 10.1.14.62
```

11.5 DNS 故障排除

由于前面所述的域名配置文件和数据库文件都是手工编辑输入的，所以有可能出现错误。下面是一些常见的错误列表：

- (1) 配置文件名写错。在这种情况下，运行 nslookup 命令不会出现提示符“>”。
- (2) 主机域名后没有一个小点(.)。这是最常见的错误。在不同的地方漏掉所出现的现象是不同的。如果是在一个完全标识域名后漏掉，那么查询到的主机名或部分的域名会出现两个。
- (3) /etc/resolv.conf 文件中域名服务器 IP 地址不正确。在这种情况下，nslookup 命令不会出现提示符。

(4) 回送地址数据库文件有问题。同样，nslookup 命令不会出现提示符。

(5) 在/etc/named.conf 文件中定义的文件名与/var/named 目录下的文件名一致。

【技巧】 如何快速查找 DNS 存在的问题？

```
[root@redflag /root]#tail -f /var/log/messages
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.10.1.14:8: using RFC 1035 TTL semantics
```

```
Mar 20 14:26:37 redflag 3月 20 14:26:37 named: named startup succeeded
Mar 20 14:26:37 redflag named[885]: master.c:1197: unexpected error:
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.local:1: no TTL specified. THIS
ZONE WILL NO LONGER WORK IN FUTURE VERSIONS. Add a TTL.
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.local:8: using RFC 1035 TTL
semantics
Mar 20 14:26:37 redflag named[885]: master.c:1197: unexpected error:
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.hosts:1: no TTL specified. THIS
ZONE WILL NO LONGER WORK IN FUTURE VERSIONS. Add a TTL.
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.hosts:8: using RFC 1035 TTL
semantics
Mar 20 14:26:37 redflag named[885]: dns_master_load: named.hosts:10: ignoring out-of-zone data
(localhost)
Mar 20 14:26:37 redflag named[885]: running
```

通过日志上面的提示信息我们可以非常迅速的查找问题出在哪里。

本章小结

域名系统(DNS)是一种用于 TCP/IP 应用程序的分布式数据库,它提供主机名和 IP 地址之间的转换以及有关电子邮件的路由信息。本章主要介绍了 DNS 的基本原理、DNS 域名解析过程、DNS 配置与测试以及 DNS 故障排除等内容。DNS 配置是网络管理的重点和难点,应该好好理解和掌握。

习 题

1. 指定域名服务器位置的文件是_____。
A. /etc/hosts B. /etc/networks C. /etc/resolv.conf D. /.profile
2. DNS 提供了一个_____命名方案。
A. 分级 B. 分层 C. 多级 D. 多层
3. Internet 管理结构最高层域划分中表示商业组织的是_____。
A. com B. gov C. mil D. org
4. _____表示别名的资源记录。
A. MX B. SOA C. cname D. PTR
5. DNS 最常见的版本是_____。
A. BIND B. named C. Samba D. COMBA
6. 主域名服务器在 DNS 配置文件/etc/named.conf 中由“zone”语句中的“type master”变量来定义。这种说法_____。
A. 正确 B. 错误

7. 在配置/etc/nsswitch.conf 文件时，我们发现在 hosts 行选项的次序是“files nis dns”，这表明在查找主机名时，将首先在/etc/hosts 文件查找，然后通过 NIS 查找，最后通过 DNS 服务器来查找。

- A. 正确 B. 错误

8. DNS 资源记录通常包含 4 项，格式为：Domain Record type Class Record data。这种说法_____。

- A. 正确 B. 错误

9. 常用的 DNS 测试的命令包括_____。

- A. nslookup B. host C. dig D. trace

10. 常见的域名服务器包括_____。

- A. 主域名服务器 B. 辅助域名服务器
C. 根域名服务器 D. 转发域名服务器

第 12 章 Apache

当今是信息时代,很多的信息是利用 WWW 网站传递的,而 Apache 是目前世界上使用最为广泛的 WWW 服务器,特别对于 Linux 用户来说,Apache 是最容易使用的 Web 服务器,而且是免费使用的。

12.1 Apache 简介

12.1.1 Apache 的地位和功能

2003 年 3 月,根据 Netcraft(www.netcraft.com)对 1995 年 9 月到 2003 年 3 月 Web 服务器使用情况的调查结果显示如图 12-1 所示,Apache 是世界排名第一的 Web 服务器,而且这种良好的势头还在上升。

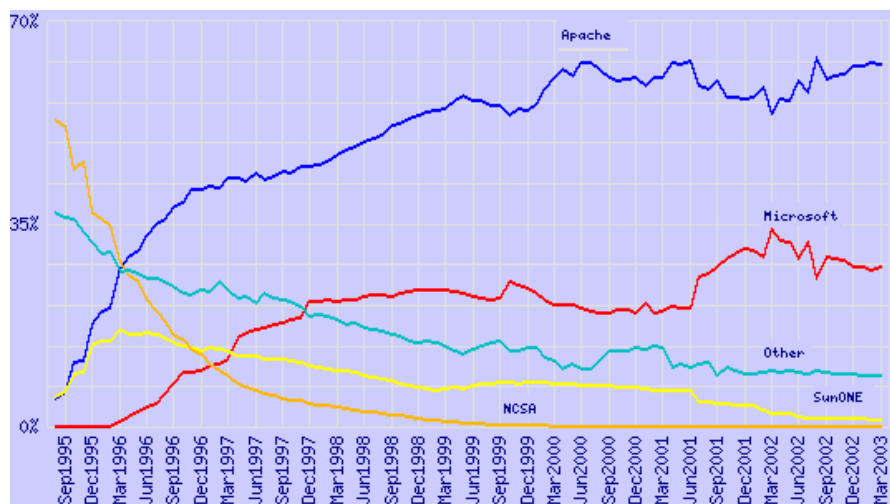


图 12-1 1995.9 ~ 2003.3 各种 Web 服务器的使用情况

Apache 之所以有今天的好成绩,是因为它有一个强大的 Apache Group。Apache Group 是一个完全通过 Internet 进行运作的非盈利机构。由它来决定 Apache Web 服务器的标准发行版中应该包含的内容,并且准许任何人修改排错,提供新的特征和将它移植到新的平台上等。当新的代码被提交给 Apache Group 时,该团体审核它的具体内容并进行测试,如果满意,该代码就会被集成到 Apache 的主要发行版中。Apache 的主要特性包括:

- (1) 几乎可以运行在所有的计算机平台上；
- (2) 支持最新的 HTTP/1.1 协议；
- (3) 简单而强有力的基于文件的配置，即 httpd.conf；
- (4) 支持通用网关接口(CGI)；
- (5) 支持虚拟主机；
- (6) 支持 HTTP 认证；
- (7) 集成 PERL；
- (8) 集成的代理服务器；
- (9) 可以通过 Web 浏览器监视服务器的状态，可以自定义日志；
- (10) 支持服务器端包含命令(SSl)；
- (11) 支持安全 SOCKET 层(SSL)；
- (12) 具有用户会话过程的跟踪能力；
- (13) 支持 FASTCGI；
- (14) 支持 JAVA SERVLETS。

12.1.2 Apache 的下载和安装

通常用户在安装 Linux 时可以选择安装 Apache，然后再在启动时启动 httpd 进程，这样打开浏览器，输入 localhost 或本机的 IP 地址，就会访问到 Apache 的默认页面。

用户可以按照下列步骤检查系统上的 Apache 服务器软件的安装情况。

- (1) 用 “rpm -q” 命令检查是否安装了 Apache 软件包。输入如下命令：

```
[root@redflag /root]#rpm -q apache
```

如果输出显示了 Apache 软件包名称，就说明已经安装了软件。

- (2) 输入如下命令，检查是否运行了 httpd 进程。

```
[root@redflag /root]#ps ax | grep httpd
955 ?      S        0:00 /usr/sbin/httpd
958 ?      S        0:00 /usr/sbin/httpd
959 ?      S        0:00 /usr/sbin/httpd
960 ?      S        0:00 /usr/sbin/httpd
961 ?      S        0:00 /usr/sbin/httpd
962 ?      S        0:00 /usr/sbin/httpd
963 ?      S        0:00 /usr/sbin/httpd
964 ?      S        0:00 /usr/sbin/httpd
965 ?      S        0:00 /usr/sbin/httpd
966 ?      S        0:00 /usr/sbin/httpd
967 ?      S        0:00 /usr/sbin/httpd
971 pts/1  S        0:00 grep httpd
```

Apache 版本的更新一般要快于 Linux 内核的更新，要下载新的 Apache 版本，可以到 <http://www.apache.org> 网站下载。Apache 网站的下载画面如图 12-2 所示。

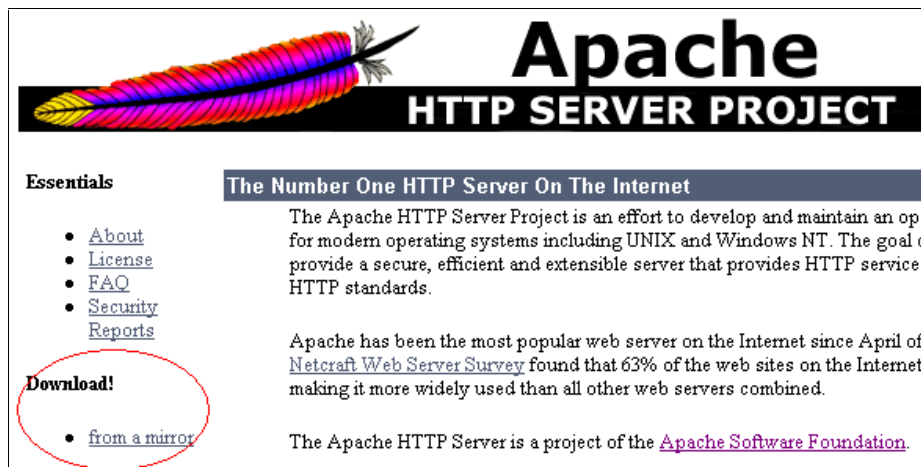


图 12-2 Apache 网站下载画面

很多免费的 Linux 应用软件在网上都可以找到，但基本上均为“*.tgz”或“*.tar.gz”格式，需要编译才能使用。本章将从网上下载一个新的 Apache 版本，通过其下载、编译、再到配置和使用的完整过程让大家学习和体会。

【实例 12.1】

我们下载的是 apache_1.3.27.tar.gz 文件，存放在 /root 目录下，通过下面的步骤可以完成安装和编译等工作。

(1) 解压缩。可用以下命令：

```
[root@redflag /root]#tar zxvf apache_1.3.27.tar.gz -C /root/apache
```

tar 命令是在 Linux 中应用最为广泛的，zxvf 和 -C 参数说明如下：

z：用 gzip 来压缩和解压的文件，还原时一定也要使用该参数。

x：从档案文件中释放文件。

v：报告 tar 命令处理的详细信息。

f：使用文件时这个参数是必须的。

-C：将压缩文件解压到指定的目录中。本例中将 Apache 源代码解压到 /root/apache 目录中。

(2) 进入到解压的目录。用以下命令：

```
[root@redflag /root]#cd apache/apache_1.3.27
```

进入到该目录后，用户会找到一个 configure 的脚本，该脚本是使用 APACI 配置所必须的。用户可以在该脚本后加“--help”来获取帮助，执行的命令如下：

```
[root@redflag apache_1.3.27]#./configure --help
```

(3) 确定安装 Apache 的路径。首先确定 Apache 的安装路径。Apache 默认的安装路径是 /usr/local/apache。当然用户可以根据自己的需要进行指定，本例中我们将 Apache 安装到 /usr/apache 下。执行如下的命令：

```
[root@redflag apache_1.3.27]#./configure --prefix=/usr/apache
```

执行结果如下：

```
Configuring for Apache , Version 1.3.27
+ using installation path layout: Apache (config.layout)
Creating Makefile
Creating Configuration.apaci in src
Creating Makefile in src
:
Creating Makefile in src/ap
Creating Makefile in src/main
Creating Makefile in src/lib/expat-lite
Creating Makefile in src/modules/standard
```

【说明】 Apache 的安装目录和存放源程序的目录是不同的概念，请大家区分开来。在本例中，Apache 的安装目录是/usr/apache，而放置源代码的目录是/root/apache。

(4) 编译 Apache。在成功的运行了上面的脚本文件后，用户就可以编译 Apache 了。从 Apache 源代码目录的根目录执行如下的指令来编译 Apache 源程序：

```
[root@redflag apache_1.3.27]#make
```

如果编译成功，将产生 WWW 服务器可执行的二进制文件，否则，请查看错误信息并重新按配置步骤运行一遍。

(5) 安装 Apache。make 命令成功编译以后，在没有任何错误信息的情况下，可以运行如下的指令来安装 Apache。

```
[root@redflag apache_1.3.27]#make install
```

安装完成后的画面如图 12-3 所示。

```
+-----+
| You now have successfully built and installed the          |
| Apache 1.3 HTTP server. To verify that Apache actually    |
| works correctly you now should first check the            |
| (initially created or preserved) configuration files      |
|                                                          |
| /usr/apache/conf/httpd.conf                               |
|                                                          |
| and then you should be able to immediately fire up      |
| Apache the first time by running:                         |
|                                                          |
| /usr/apache/bin/apachectl start                           |
|                                                          |
| Thanks for using Apache.                                  |
|                                                          |
| The Apache Group                                         |
| http://www.apache.org/                                   |
+-----+
```

图 12-3 Apache 安装成功画面

(6) 清除对象文件。只要已经编译并安装了 Apache，就可以运行“make clean”来删除任何在编译时创建的对象文件，执行的命令如下：

```
[root@redflag apache_1.3.27]#make clean
```

在正确地完成上述步骤之后，新版本 Apache 就已经安装好了。

12.1.3 Apache 的启动与关闭

Apache 安装好之后，会在安装目录下生成一个 bin 的目录，Apache 的主要运行脚本都在该目录下。在运行这些命令时一定要进到该目录里，或者输入绝对路径，否则不能正确执行。

(1) 启动 Apache。

```
[root@redflag /root]#cd /usr/apache/bin
```

```
[root@redflag bin]#./apachectl start
```

启动成功后，在浏览器中输入本机的 IP 地址，将会看到 Apache 的默认页面，如图 12-4 所示。

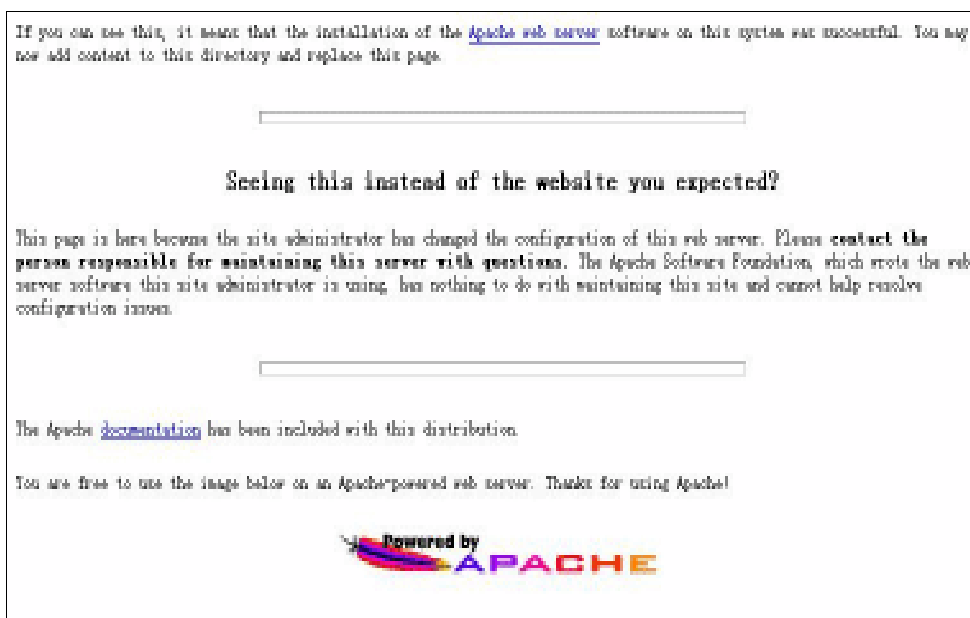


图 12-4 Apache 的默认首页

(2) 重新启动 Apache。

```
[root@redflag bin]#./apachectl restart 或
```

```
[root@redflag bin]#./apachectl graceful
```

(3) 关闭 Apache。

```
[root@redflag bin]#./apachectl stop
```

12.2 Apache 的 配 置

Apache 服务器的设置文件位于 /usr/apache/conf/ 目录下，传统上使用三个配置文件 httpd.conf、access.conf 和 srm.conf 来配置 Apache 服务器。事实上，当前版本的 Apache 将原来 httpd.conf、srm.conf 与 access.conf 中的所有配置参数均放在了一个配置文件 httpd.conf 中，只是为了与以前的版本兼容(使用这三个设置文件的方式来源于 NCSA-httpd)，才使用

三个配置文件，而提供的 `access.conf` 和 `srm.conf` 文件中没有具体的设置。由于在新版本的 Apache 中，所有的设置都被放在了 `httpd.conf` 中，因此只需要调整这个文件中的设置。在 `httpd.conf` 文件中分为三个部分，分别是全局参数、服务器的主要设置和虚拟主机。本节先介绍全局参数和服务器主要设置，虚拟主机将在 12.3 节详细介绍。

我们首先来介绍 Apache 配置文件的格式，第一个字符为“#”符号的是注释行，服务器在进行语法分析时会忽略掉所有的注释行，除了注释和空行外，服务器把其他的行认为是完整的或部分的指令。指令又分成与 shell 命令类似的命令和伪 HTML 标记。例如，伪 HTML 标记的用法如下：

```
<Directory />
    Options FollowSymLinks
    AllowOverride All
</Directory>
```

与 HTML 不同，伪 HTML 标记必须各占一行，我们可以像上面那样把命令组成一组放在某个伪 HTML 标记中。其实在 Apache 配置文件中有很多类似这样的模块。

12.2.1 文件 `httpd.conf` 的全局参数

下面我们通过实际应用中一个具体的例子来详细讲述 `httpd.conf` 的全局参数的配置。

1. `ServerType standalone`

`ServerType` 定义服务器的启动方式，缺省值为独立方式 `standalone`，`httpd` 服务器将由其本身启动并驻留在主机中监视连接请求。

2. `ServerRoot "/usr/apache"`

`ServerRoot` 用来存放服务器的配置、出错和记录文件的最底层目录。如果使用 APACI 接口编译和安装 Apache，那么缺省 `ServerRoot` 为在配置脚本中提供的 `prefix` 值；否则需要修改缺省值为一个合适的目录。

由于 `httpd` 会经常进行并发的文件操作，就需要使用加锁的方式来保证文件操作不冲突，由于 NFS 文件系统在文件加锁方面能力有限，因此这个目录应该是本地磁盘文件系统，而尽量避免使用 NFS 文件系统。

3. `#LockFile /usr/apache/logs/httpd.lock`

`LockFile` 参数指定了 `httpd` 守护进程的加锁文件，一般不需要设置这个参数，Apache 服务器将自动在 `ServerRoot` 下面的路径中进行操作，但如果 `ServerRoot` 为 NFS 文件系统，便需要使用这个参数指定本地文件系统的路径。

4. `PidFile /usr/apache/logs/httpd.pid`

`PidFile` 指定的文件将记录 `httpd` 守护进程的进程号，由于 `httpd` 能自动复制其自身，因此系统中有多多个 `httpd` 进程，但只有一个进程为最初启动的进程，它与其他进程的父进程，对这个进程发送信号将影响所有的 `httpd` 进程。`PidFile` 定义的文件中就记录 `httpd` 父进程的进程号。

5. ScoreBoardFile /usr/apache/logs/httpd.scoreboard

httpd 使用 ScoreBoardFile 来维护进程的內部数据，因此通常不需要改变这个参数，除非管理员想在一台计算机上运行几个 Apache 服务器，这时每个 Apache 服务器都需要独立的设置文件 httpd.conf，并使用不同的 ScoreBoardFile。

6. #ResourceConfig conf/srm.conf 和 #AccessConfig conf/access.conf

这两个参数 ResourceConfig 和 AccessConfig 是为了与使用 srm.conf 和 access.conf 设置文件的老版本 Apache 兼容。如果没有兼容的需要，可以将对应的设置文件指定为/dev/null，这将表示不存在其他设置文件，而仅使用 httpd.conf 一个文件来保存所有的设置选项。

7. Timeout 300

Timeout 定义客户程序和服务器连接的超时间隔，超过这个时间间隔(秒)后服务器将断开与客户机的连接。

8. KeepAlive On

在 HTTP 1.0 中，一次连接只能传输一次 HTTP 请求，而 KeepAlive 参数用于支持 HTTP 1.1 版本的一次连接、多次传输功能，这样就可以在一次连接中传递多个 HTTP 请求。只有较新的浏览器才支持这个功能。

9. MaxKeepAliveRequests 100

MaxKeepAliveRequests 为一次连接可以进行的 HTTP 请求的最大请求次数。将其值设为 0，将支持在一次连接内进行无限次的传输请求。事实上没有客户程序在一次连接中请求太多的页面，通常达不到这个上限就完成连接了。

10. KeepAliveTimeout 15

KeepAliveTimeout 测试一次连接中的多次请求传输之间的时间，如果服务器已经完成了一次请求，但一直没有接收到客户程序的下一次请求，在间隔超过了这个参数设置的值之后，服务器就断开连接。

11. MinSpareServers 5 和 MaxSpareServers 10

在使用子进程处理 HTTP 请求的 Web 服务器上，由于要首先生成子进程才能处理客户的请求，因此反应时间就有一点延迟。但是，Apache 服务器使用了一个特殊技术来摆脱这个问题，这就是预先生成多个空余的子进程驻留在系统中，一旦有请求出现，就立即使用这些空余的子进程进行处理，这样就不存在生成子进程造成的延迟了。在运行中随着客户请求的增多，启动的子进程会随之增多，但这些服务器副本在处理完一次 HTTP 请求之后并不立即退出，而是停留在计算机中等待下次请求。但是空余的子进程副本不能光增加不减少，太多的空余子进程没有处理任务，也占用服务器的处理能力，因此也要限制空余副本的数量，使其保持一个合适的数量，以便既能及时回应客户请求，又能减少不必要的进程数量。

因此就可以使用参数 MinSpareServers 来设置最少的空余子进程数量，以及使用参数 MaxSpareServers 来限制最多的空闲子进程数量，多余的服务器进程副本就会退出。根据服

服务器的实际情况来进行设置，如果服务器性能较高，并且也被频繁访问，就应该增大这两个参数的设置。对于高负载的专业网站，这两个值应该大致相同，并且等同于系统支持的最多服务器副本数量，也减少了不必要的副本退出。

12. StartServers 5

StartServers 参数就是用来设置 httpd 启动时启动的子进程副本数量，这个参数与上面定义的 MinSpareServers 和 MaxSpareServers 参数相关，都是用于启动空闲子进程以提高服务器的反应速度的。这个参数应该设置为前两个值之间的一个数值，小于 MinSpareServers 或大于 MaxSpareServers 都没有意义。

13. MaxClients 150

服务器的能力毕竟是有限的，不可能同时处理无限多的连接请求，因此参数 MaxClients 就是用于规定服务器支持的最多并发访问的客户数，如果这个值设置得过大，系统在繁忙时不得不在过多的进程之间进行切换来为太多的客户进行服务，这样对每个客户的反应就会减慢，并降低了整体的效率。如果这个值设置得较小，那么系统繁忙时就会拒绝一些客户的连接请求。当服务器性能较高时，就可以适当增加这个值的设置。对于专业网站，应该使用提高服务器效率的策略，因此这个参数不能超过硬件本身的限制，如果频繁出现拒绝访问现象，就说明需要升级服务器硬件了。对于非专业网站，不太在意客户浏览器的反应速度，或者认为反应速度较慢也比拒绝连接好，也就可以略微超过硬件条件来设置这个参数。

14. MaxRequestsPerChild 0

使用子进程的方式提供服务的 Web 服务，常用的方式是一个子进程为一次连接服务，这样造成的问题就是每次连接都需要生成、退出子进程的系统操作，使得这些额外的处理过程占据了计算机的大量处理能力。因此，最好的方式是一个子进程可以为多次连接请求服务，这样就不需要这些生成、退出进程的系统消耗。Apache 就采用了这样的方式，一次连接结束后，子进程并不退出，而是停留在系统中等待下一次服务请求，这样就极大地提高了性能。但由于在处理过程中子进程要不断地申请和释放内存，次数多了就会造成一些内存垃圾，就会影响系统的稳定性，并且影响系统资源的有效利用。因此在一个副本处理过一定次数的请求之后，就可以让这个子进程副本退出，再从原始的 httpd 进程中重新复制一个干净的副本，这样就能提高系统的稳定性。由此，每个子进程处理服务请求的次数由 MaxRequestPerChild 定义。缺省的设置值为 30，这个值对于具备高稳定性特点的 Linux 系统来讲是过于保守的设置，实际上可以设置为 1000 甚至更高，设置为 0 时支持每个副本进行无限次的服务处理。

15. #Listen 3000 和 #Listen 12.34.56.78:80

Listen 参数可以指定服务器除了监视标准的 80 端口之外，还监视其他端口的 HTTP 请求。

12.2.2 文件 httpd.conf 的服务器的主要设置

同上节一样，我们将用实际应用中一个具体的例子来讲述 Apache 服务器的主要设置。

1. Port 80

Port 定义了 Standalone 模式下 httpd 守护进程使用的端口，标准端口是 80。这个选项只对于以独立方式启动的服务器有效。

2. User nobody 和 Group nobody

User 和 Group 配置是 Apache 的安全保证，Apache 在打开端口之后，就将其本身设置为这两个选项所设置的用户和组的权限然后进行运行，这样就降低了服务器的危险性。这个选项也只用于 Standalone 模式。缺省设置为 nobody 和 nogroup，这个用户和组在系统中不拥有文件，保证了服务器本身和由它启动的 CGI 进程没有权限更改文件系统。

3. ServerAdmin webmaster@redflag.com

配置文件中应该改变的也许只有 ServerAdmin，这一项用于配置 WWW 服务器的管理员的 E-mail 地址，这将在 HTTP 服务出现错误的条件下返回给浏览器，以便让 Web 使用者和管理员联系，报告错误。习惯上使用服务器上的 webmaster 作为 WWW 服务器的管理员，通过邮件服务器的别名机制，将发送到 webmaster 的电子邮件发送给真正的 Web 管理员。

4. ServerName 10.1.14.61

缺省情况下，并不需要指定这个 ServerName 参数，服务器将自动通过名字解析过程来获得自己的名字，但如果服务器的名字解析有问题(通常为反向解析不正确)，或者没有正式的 DNS 名字，也可以在这里指定 IP 地址。当 ServerName 设置不正确的时候，服务器不能正常启动。

5. DocumentRoot "/usr/apache/htdocs"

DocumentRoot 定义服务器对外发布的网页文档存放的路径，客户程序请求的 URL 就被映射为这个目录下的网页文件。这个目录下的子目录，以及使用符号连接指出的文件和目录都能被浏览器访问，只是要在 URL 上使用同样的相对目录名。当然用户可以根据自己的实际情况来决定把网页放在哪个目录。

6. UserDir public_html

当在一台 Linux 上运行 Apache 服务器时，这台计算机上的所有用户都可以有自己的网页路径，形如 `http://www.linux.com/~user`，使用波浪符号加上用户名就可以映射到用户自己的网页目录上。映射目录为用户个人主目录下的一个子目录，其名字就用 UserDir 这个参数进行定义，缺省为“public_html”。

7. DirectoryIndex index.html

很多情况下，URL 中并没有指定文档的名字，而只是给出了一个目录名，Apache 服务器就自动返回这个目录下由 DirectoryIndex 定义的文件，当然可以指定多个文件名字，系统会在这个目录下顺序搜索。当所有由 DirectoryIndex 指定的文件都不存在时，Apache 服务器可以根据系统设置，生成这个目录下的所有文件列表，提供用户选择。此时该目录的访问控制选项中的 Indexes 选项(Options Indexes)必须打开，以使得服务器能够生成目录列表，否则 Apache 将拒绝访问。

8. ErrorLog /usr/apache/logs/error_log

用来存放 Web Server 的出错信息的文件。

9. ServerSignature On

在一些情况下,例如当客户请求的网页并不存在时,服务器将产生错误文档,缺省情况下由于打开了 ServerSignature 选项,错误文档的最后一行将包含服务器的名字、Apache 的版本等信息。有的管理员更倾向于不对外显示这些信息,就可以将这个参数设置为 Off,或者设置为 E-mail,最后一行将替换为对 ServerAdmin 的 E-mail 提示。

10. Alias /icons/ "/usr/apache/icons/"

Alias 参数用于将 URL 与服务器文件系统中的真实位置进行直接映射,一般的文档将在 DocumentRoot 中进行查询,然而使用 Alias 定义的路径将直接映射到相应目录下,而不再到 DocumentRoot 下面进行查询。因此 Alias 可以用来映射一些公用文件的路径,例如,保存了各种常用图标的 icons 路径。这样使得除了使用符号连接之外,文档根目录(DocumentRoot)外的目录也可以通过使用 Alias 映射,以提供给浏览器访问。

11. ScriptAlias /cgi-bin/ "/usr/apache/cgi-bin/"

ScriptAlias 也是用于 URL 路径的映射,但与 Alias 的不同在于,ScriptAlias 是用于映射 CGI 程序的路径,这个路径下的文件都被定义为 CGI 程序,通过执行它们来获得结果,而非由服务器直接返回其内容。缺省情况下 CGI 程序使用“cgi-bin”目录作为虚拟路径。

12.3 Apache 的各种服务

正如前面介绍的,Apache 是功能非常强大的 Web 服务器,除了提供最基本的 Web 服务之外,还可以实现个人主页服务、虚拟主机和代理服务。

12.3.1 用户个人主页

现在许多网站(如 www.163.com 等)都允许用户有自己的主页空间,而用户可以很容易地管理自己的主页目录。Apache 服务器中同样可以实现用户的个人主页。用户个人主页的 URL 的格式一般如下所示:

`http://www.mydomain.com/~username`

其中“~username”是 Linux 合法用户的名字。

用户的主页存放的目录由文件 httpd.conf 服务器的主要设置参数 UserDir 设定,一般情况下 UserDir 的值是“public_html”,当然用户可以根据自己的需要来设定。下面我们通过具体的实例来讲述如何实现用户的个人主页。

【实例 12.2】

在本实例中 UserDir 的值设为 public_html,用户名为 test,实现步骤如下:

(1) 修改用户的主目录的权限。执行如下的指令:

```
[root@redflag /root]#chmod 705 /home/test
```

- (2) 创建存放用户主页的目录。执行如下指令：

```
[root@redflag /root]#mkdir /home/test/public_html
```

- (3) 创建索引文件。执行如下指令：

```
[root@redflag /root]#cd /home/test/public_html
```

```
[root@redflag public_html]#vi index.html
```

- (4) 测试。执行如下指令：

```
[root@redflag /root]#lynx 127.0.0.1/~test
```

如果能够看到 index.html 文件的内容，则证明用户个人主页成功实现。如果用户不想让他人访问自己的主页，则可以通过把用户的主目录的权限设为 0700 来完成。

【技巧】 如何关闭个人主页的服务？

如果不想为正式的用户提供网页服务，使用 DISABLED 作 UserDir 的参数即可。

12.3.2 虚拟主机

Apache 的配置文件 httpd.conf 的第三部分是关于实现虚拟主机的。虚拟主机是在一台 Web 服务器上，可以为多个单独域名提供 Web 服务，并且每个域名都完全独立，包括具有完全独立的文档目录结构及设置，这样域名之间完全独立，不但使每个域名访问到的内容完全独立，并且使另一个域名无法访问其他域名所提供的网页内容。

虚拟主机的概念对于 ISP 来讲非常有用，虽然一个组织可以将自己的网页挂在具备其他域名的服务器上的下级网址上，但使用独立的域名和根网址更为正式，易为众人接受。传统上，必须自己设立一台服务器才能达到单独域名的目的，然而这需要维护一个单独的服务器，很多小单位缺乏足够的维护能力，所以更为合适的方式是租用别人维护的服务器。ISP 也没有必要为一个机构提供一个单独的服务器，完全可以使用虚拟主机，使服务器为多个域名提供 Web 服务，而且不同的服务互不干扰，对外就表现为多个不同的服务器。

有两种设定虚拟主机的方式，分别是基于 IP 的虚拟主机和基于名称的虚拟主机。基于 IP 的虚拟主机基于 HTTP 1.0 标准，它需要一个具备多 IP 地址的服务器，再配置 DNS 服务器，给每个 IP 地址以不同的域名，最后才能配置 Apache 的配置文件，使服务器对不同域名返回不同的 Web 文档。由于这需要使用额外的 IP 地址，对每个要提供服务的域名都要使用单独的 IP 地址，因此这种方式实现起来问题较多。可以在一个网卡上绑定多个 IP 地址，Linux 下需要使用 ifconfig 的 Alias 参数来进行这个配置，但此时会影响网络性能。而基于名称的虚拟主机基于 HTTP 1.1 标准，在 HTTP 1.1 协议中规定了对浏览器和服务器通信时，服务器能够跟踪浏览器请求的主机名字，因此可以利用这个新特性，使用更轻松的方式设定虚拟主机。这种方式不需要额外的 IP 地址，但需要新版本的浏览器支持。这种方式已经成为建立虚拟主机的标准方式。要建立非 IP 基础的虚拟主机，多个域名是不可少的配置，因为每个域名就对应一个要服务的虚拟主机。本书中主要讨论基于名称的虚拟主机的实现。下面我们通过一个具体的实例来讲述虚拟主机的配置。

虚拟主机的 IP 地址为 10.1.14.61，两个虚拟主机的域名分别为 hosta.redflag.com 和 hostb.redflag.com。实现步骤如下。

1. 创建虚拟主机目录

我们在/usr/apache/htdocs 目录下创建 hosta 和 hostb 两个目录，分别用来存放两主机的网页。操作如下：

```
[root@redflag /root]#mkdir /usr/apache/htdocs/hosta
[root@redflag /root]#mkdir /usr/apache/htdocs/hostb
[root@redflag /root]#chmod 705 /usr/apache/htdocs/hosta
[root@redflag /root]#chmod 705 /usr/apache/htdocs/hostb
```

2. 域名解析

实现域名解析可以有两种方法：

(1) 在客户机上通过修改/etc/hosts 文件实现。这是一种比较简单的方法，只需在/etc/hosts 文件中加入下面两行：

```
10.1.14.61 hosta.redflag.com
10.1.14.61 hostb.redflag.com
```

(2) 在 DNS 服务器上通过配置 DNS 实现。虽然基于名称的虚拟主机不需要惟一的 IP 地址，但需要给每台虚拟主机创建一个 CNAME。我们在第 11 章的实例中已经配置了 DNS，现在我们只需要在/var/named/named.hosts 文件中加入如下两行：

```
hosta.redflag.com      IN      CNAME      a100.redflag.com.
hostb.redflag.com      IN      CNAME      a100.redflag.com.
```

重新启动 DNS 后，可以用 nslookup 和 ping 命令来测试，命令如下：

```
[root@redflag /root]#nslookup
>set type=cname
>hosta.redflag.com
[root@redflag /root]#ping hosta.redflag.com
```

3. 修改 httpd.conf 文件

在 httpd.conf 文件中修改如下：

```
NameVirtualHost 10.1.14.61
```

NameVirtualHost 用来指定虚拟主机使用的 IP 地址，这个 IP 地址将对应多个 DNS 名字，如果 Apache 使用 Listen 参数控制了多个端口，那么就可以在这里加上端口号以进一步区分对不同端口的不同连接请求。接下来使用 VirtualHost 语句，并且使用 NameVirtualHost 指定的 IP 地址作参数，对每个名字都定义对应的虚拟主机设置。

```
<VirtualHost 10.1.14.61>
```

#每个虚拟主机的定义以<VirtualHost IP>开始，以</VirtualHost>结束，成对出现，构成伪 HTML 代码。

```
ServerAdmin webmaster@redflag.com
```

#管理员的 E-mail 地址。

```
DocumentRoot /usr/apache/htdocs/hosta
```

#虚拟主机 Hosta 的目录。

```
ServerName hosta.redflag.com
```

```
#虚拟主机 Hosta 的域名。
ErrorLog logs/ hostb.redflag.com-error_log
#指定错误信息记录文件。
</VirtualHost>
<VirtualHost 10.1.14.61>
    ServerAdmin webmaster@redflag.com
    DocumentRoot /usr/apache/htdocs/hostb
    ServerName hostb.redflag.com
    ErrorLog logs/ hostb.redflag.com-error_log
</VirtualHost>
```

4. 测试

重新启动 Apache 后就可以进行测试了。

执行命令如下：

```
[root@redflag /root]#cd /usr/apache/bin
[root@redflag bin]#./httpd -S
```

结果如下：

```
virtualhost configuration:
10.1.14.61:80          is a NameVirtualHost
default server hosta.redflag.com (/usr/apache/conf/httpd.conf:980)
port 80 namevhost hosta.redflag.com (/usr/apache/conf/httpd.conf:980)
port 80 namevhost hostb.redflag.com (/usr/apache/conf/httpd.conf:987)
```

另外，在浏览器中输入 hosta.redflag.com 和 hostb.redflag.com，应该可以看到属于不同主机的网页。

【说明】 VirtualHost 的参数地址一定要和 NameVirtualHost 定义的地址相一致，必须保证所有的值严格一致，Apache 服务器才承认这些定义是为这个 IP 地址定义的虚拟主机。

12.3.3 代理服务

代理服务器的英文全称是 Proxy Server，其功能就是代理网络用户去取得网络信息。形象地说，它是网络信息的中转站。在一般情况下，我们使用网络浏览器直接去连接其他 Internet 站点并取得网络信息时，须送出 Request 信号来得到回答，然后对方再把信息以 bit 方式传送回来。代理服务器是介于浏览器和 Web 服务器之间的一台服务器，有了它之后，浏览器不是直接到 Web 服务器去取回网页而是向代理服务器发出请求，Request 信号会先送到代理服务器，由代理服务器来取回浏览器所需要的信息并传送给用户的浏览器。而且，大部分代理服务器都具有缓冲的功能，就好像一个大的 cache，它有很大的存储空间，不断将新取得数据储存到它本机的存储器上，如果浏览器所请求的数据在它本机的存储器上已经存在而且是最新的，那么它就不重新从 Web 服务器取数据，而直接将存储器上的数据传送给用户的浏览器，这样就能显著提高浏览速度和效率。而且它建立起了内部网和外部网之间的一道屏障，对内部网的用户起到了保护作用。

Apache 同样具有代理服务的功能，它的 proxy 支持来自于 mod_proxy 模块，在缺省条件下，它不能被编译生成，需要添加 mod_proxy 模块，然后进行编译和安装，最后进行配置。接下来我们通过具体的实例来讲述。

【实例 12.3】

本实例将讲述 Apache 代理模块的编译、安装、确认和配置过程。

(1) 加入代理模块。

在用户的 Apache 源代码目录下执行如下的操作：

```
[root@redflag apache_1.3.27]# ./configure --prefix=/usr/apache
--enable-module=proxy
```

结果如下：

```
Configuring for Apache , Version 1.3.27
+ using installation path layout: Apache (config.layout)
Creating Makefile
:
Creating Makefile in src/lib/expat-lite
Creating Makefile in src/modules/standard
Creating Makefile in src/modules/proxy
```

(2) 编译。执行如下指令：

```
[root@redflag apache_1.3.27]#make
```

(3) 安装。执行如下指令：

```
[root@redflag apache_1.3.27]#make install
```

(4) 确认。在执行完上述三条命令后，可以用下面的命令来确认 mod_proxy 是否在输出中显示：

```
[root@redflag apache_1.3.27]#cd /usr/apache/bin
[root@redflag bin]#./httpd -l
```

结果如下：

```
Compiled-in modules:
http_core.c
mod_env.c
mod_log_config.c
:
mod_proxy.c
:
```

(5) 配置代理服务器。在 httpd.conf 中我们配置如下：

```
<IfModule mod_proxy.c>
# IfModule 判断加载模块是否成功 ,如果已经成功地加载了 mod_proxy.c 的模块 ,我们定义 proxy
 的相关信息从这里开始。
ProxyRequests On
```

#打开代理支持。

```
<Directory proxy:*>
```

```
    Order deny,allow
```

```
    Deny from all
```

```
    Allow from .redflag.com
```

```
</Directory>
```

#<Directory proxy:*>与</Directory>之间的代码表示只允许 redflag.com 域使用该代理。

```
    ProxyVia On
```

#此设置用于处理 HTTP/1.1 的 via:文件头的功能。

```
CacheRoot "/usr/apache/proxy"
```

#指定缓存要放置的目录。

```
CacheSize 1024
```

#指定缓存的大小，默认是 5 KB，可以自行调整，不能设得过小。

```
CacheGcInterval 4
```

#设置间隔多少时间去检查缓冲的大小，默认是 4 小时。

```
CacheMaxExpire 24
```

#设置一份文件最久可留在缓存的时间，默认是 24 小时。

```
CacheLastModifiedFactor 0.1
```

该参数用于计算缓存的文件自上次更新后多久需要删除的系数。例如，一个网页已在缓存中保留了 12 小时，则 $12 \times 0.1 = 1.2$ 小时，表示 1.2 小时后此文件要被删除。若计算出来的值大于 CacheDefaultExpire，则以 CacheDefaultExpire 为准。

```
CacheDefaultExpire 1
```

#对于使用非 HTTP 的传输协议，可由此值来判断何时将该文件从缓存中删除。

```
NoCache a-domain.com another-domain.edu joes.garage-sale.com
```

设置缓存之后，不一定每一个网站的文件都要存入缓存，可用 NoCache 指定某个网站的文件不要放入缓存。

```
</IfModule>
```

#该模块结束，与<IfModule mod_proxy.c>对应组成伪 HTML 代码。

(6) 代理服务器的测试。

在完成第(5)步后，重新启动 Apache，就可以测试了。首先需要将客户端的浏览器设置为通过代理服务器上网。接下来在浏览器中输入想要访问的 URL，如果访问成功，则证明代理服务器配置成功。

12.4 Apache 访问控制

假设有一些敏感的信息要放在 Intranet/Internet 上，你首先可能会想到自己开发一个用户身份认证的系统来保护自己的 Web 页面。其实 Apache 本身就自带了很多访问控制的机制，实现起来也不复杂。

12.4.1 Apache 访问控制指令

在 httpd.conf 文件中，有很多类似<Directory "目录">...</Directory>区域，在每个模块中有 Options、AllowOverride、Limit 等指令，它们都是和访问控制相关的。

1. Options

Options 用来设置模块的功能。可以设置的特性如表 12-1 所示。

表 12-1 Options 指令参数表

Options 指令参数	功 能 说 明
All	所有的目录特性都有效，这是缺省状态
ExecCGI	允许这个目录下可以执行 CGI 程序
FollowSymLinks	允许使用符号连接，这将使浏览器有可能访问文档根目录(DocumentRoot)之外的文档
Includes	提供 SSI 功能
IncludesNOExec	执行 SSI 功能，但不执行 CGI 程序中的#exec 与#include 命令
Indexes	允许浏览器生成这个目录下所有文件的索引，使得在这个目录下没有 index.html(或其他索引文件)时，能向浏览器发送这个目录下的文件列表
MultiViews	使用内容协商功能，经过服务器和网页浏览器相互沟通后，决定网页传送的性质
None	不允许访问目录
SymLinksIfOwnerMatch	只有符号连接的目的与符号连接本身为同一用户所拥有时，才允许访问，这个设置将增加一些安全性

可以使用“+”和“-”号在 Options 指令里打开或取消某选项。如果不使用这两个符号，那么在模块内的 Options 值将完全覆盖以前的 Options 指令里的值。

【实例 12.4】

本实例允许在目录/usr/apache/htdocs 中执行 CGI 和 SSI，配置如下：

```
<Directory "/usr/apache/htdocs">  
    Options +ExecCGI +Includes  
</Directory>
```

【说明】在使用多个 Options 命令时，应注意小环境里使用的 Options 比大环境里使用的 Options 具有优先权。

2. AllowOverride

AllowOverride 决定是否可废除之前所设置的访问权限，而在此处另设权限。AllowOverride 会读取“.htaccess”文件，以决定是否改变原来的设置。如果 AllowOverride 设置为 None，服务器将不去读取 AccessFileName 指定的文件，这样可以加快服务器的响应时间，因为服务器不必对每一个请求去找 AccessFileName 指定的文件。AllowOverride 可以设置的参数如表 12-2 所示。

表 12-2 AllowOverride 指令参数表

AllowOverride 指令参数	功 能 说 明
AuthConfig	允许访问控制文件使用 AuthName、AuthType 等针对每个用户的认证机制，这使目录属主能用口令和用户名来保护目录
FileInfo	允许访问控制文件中可以使用 AddType 等参数设置
Indexes	允许使用控制目录检索的指令
Limit	允许使用控制主机访问的命令
All	允许忽略任何目录命令
None	停止忽略任何目录命令
Options	允许使用控制特定文件类型的指令

3. Limit

Limit 指令的语法如下：

```
<Limit mothhod> .....</Limit>
```

这一模块包含了一组访问控制指令，这些指令只针对所指定的 HTTP 方式。方式的名称列表可以用下面的一个或多个：GET、POST、PUT、CONNECT。如果使用 GET，它还会影响到 HEAD 的请求。如果想限制所有的方式，就不要在<Limit>指令里包含任何的方式的名称。对于一般的情况，可以对大部分客户打开 GET、POST 和 HEAD 请求，而关闭 PUT 等其他更复杂且不常用的请求。

12.4.2 基于主机的访问控制

下面三条指令用于实现基于主机的访问控制。这里的主机可以是一个完整的域名，也可以是一个 IP 地址。

1. Allow

用户可以通过该指令指定一个关于主机的列表，列表中可包含一个或多个主机名或 IP 地址，列表中的主机被允许访问某一特定的目录。当指定了多个主机名或 IP 地址时，它们必须用空格来分隔。Allow 指令的常见参数如下：

(1) ALL：允许所有主机访问站点。示例如下：

```
Allow from ALL
```

(2) 某主机的绝对域名：允许合格的域名访问站点。示例如下：

```
Allow from www.redflag.com
```

(3) 某主机的 IP 地址：仅被指定的 IP 地址才允许访问站点。示例如下：

```
Allow from 10.1.14.61
```

(4) 网络号/掩码：指定的网络可以访问站点。示例如下：

```
Allow from 10.1.14.0/255.255.255.0
```

2. Deny

该指令的功能与 Allow 指令相反。用户可以通过该指令指定一个关于主机的列表，列表中可包含一个或多个主机名或 IP 地址，列表中的主机被拒绝访问某一特定的目录。在

Allow 指令中参数和示例同样也适用于 Deny 指令。

3. Order

该指令定义处理 Allow 和 Deny 的顺序，其参数是以逗号分隔的列表，列表中指定了哪一条指令先执行。特别注意的是，影响所有主机的命令被授予最低的优先级。

下面通过一个具体的实例来讲述基于主机的 Apache 的访问控制。

【实例 12.5】

本实例是拒绝 IP 地址为 10.1.14.125 和 10.1.14.66 的主机访问/usr/apache/htdocs 目录。在 httpd.conf 中的模块如下：

```
<Directory "/usr/apache/htdocs">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All

    Order deny ,allow
    #定义处理顺序。
    Deny from 10.1.14.125 10.1.14.66
    #拒绝两台主机访问/usr/apache/htdocs 目录。
    Allow from all
    #允许其他主机的访问。
</Directory>
```

重新启动 Apache 服务器后，在 IP 地址为 10.1.14.125 的主机访问时的结果如图 12-5 所示。

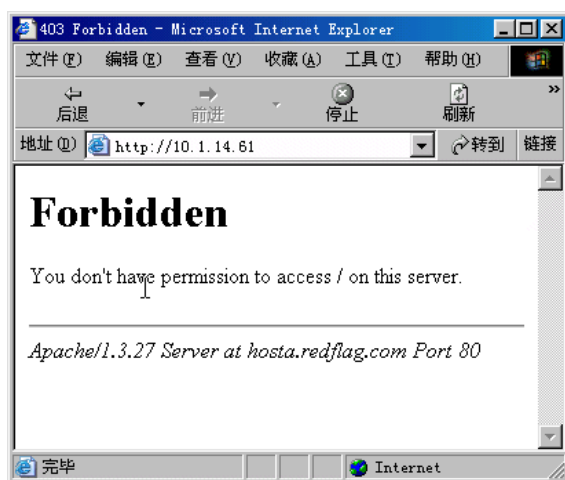


图 12-5 主机 10.1.14.125 访问被拒绝

12.4.3 基于用户名的访问控制

相信大多数读者都有这样的经历，当访问某些网站或网页时需要输入用户名和密码。身份认证是防止非法用户使用资源的有效手段，也是管理注册用户的有效方法，现在很多网站都使用身份认证来管理用户资源，对用户的访问权限进行严格的限制。Apache 服务器

允许在全局访问配置文件或用户的 “.htaccess” 文件中对目录进行强制口令保护。对于口令保护的目录, 必须为四个不同的命令指定相应的值, 即 AuthName、AuthType、AuthUserFile 和 AuthGroupFile 命令。下面分别来介绍。

(1) AuthName。AuthName 命令是一个短的字符串, 告诉用户他们所被询问的口令, 如: “Please input you valid username and password:”。

(2) AuthType。AuthType 命令标识了服务器使用的鉴别方法。AuthType 可以选择两个设置之一: Basic 和 Digest。如果将 AuthType 设为 Basic, 使用的就是标准的和基于 Unix 加密方式的口令系统, 还需使用 AuthUserFile 命令。如果将其设为 Digest, 使用的就是 MD5 加密方式, 这是一种更为安全的编码系统。在大多数站点上不应该使用 Digest, 因为它还没有得到大多数浏览器的支持, 但是在 Intranet 和可以决定所有用户使用的浏览器软件的小规模的应用中, 它是可以使用的。

(3) AuthUserFile。AuthUserFile 命令指定了目录中 Apache 服务器用户口令文件的全路径名。可以使用 htpasswd 程序来创建口令文件。

(4) AuthDigestFile。如果使用的是 Digest 鉴别方式, 就要使用 AuthDigestFile 作为口令名单。与 AuthUserFile 相同, 只要将 AuthDigestFile 设置为口令文件的路径名和文件名即可。要创建文摘式的口令文件, 可以使用 htdigest 程序。

(5) require。require 命令指定需要什么条件才能被授权访问。可以使用这个命令强制对一目录的口令保护。这个命令后面应该跟一份“实体”名单。这些实体可以是 AuthUserFile 或 AuthGroupFile 命令所定义的用户或组的名称, 也可以使用“valid-user”的关键字, 告诉服务器在 AuthUserFile 中的任何用户只要能够提供有效的口令就允许进行访问。它可以只列出可能连接的指定用户、指定可能连接的用户的一个组或多个组的清单, 如:

```
require user user1 user2
#只有 user1 和 user2 可以访问。

require group test
#只有 test 组可以访问。
```

下面通过一个具体的实例来讲述基于主机的 Apache 的访问控制。

【实例 12.6】

本实例是对 /usr/apache/htdocs 目录进行保护, 需要合法的用户名和口令才能访问。操作过程如下:

(1) 配置 httpd.conf 文件。在 httpd.conf 文件中的配置模块代码如下:

```
<Directory /usr/apache/htdocs/>
#该模块的作用目录为 /usr/apache/htdocs/.
AuthName Protected
#提示用户的信息为 “Protected”。
AuthType basic
# AuthType 鉴别方法是 basic。
AuthUserFile /usr/apache/conf/users
# 这一行很重要, 它指定了验证用户名和口令的路径和文件名。
<Limit GET POST>
```

```
# 限制 HTTP 协议中的 GET 和 POST 方法。
require valid-user
# 需要合法的用户，即在/usr/apache/conf/users 文件中的用户。
</Limit>
</Directory>
```

(2) 生成用户密码文件。命令 `htpasswd` 可以帮助我们完成这项任务。

比如说我们想要生成 `lgm` 用户密码文件，操作如下：

```
[root@redflag /root]# htpasswd -c /usr/apache/conf/users lgm 123456
```

其中 123456 是用户 `lgm` 的验证密码。

如果想要生成多个用户，则需要使用 `htpasswd` 命令的 “-b” 参数，操作如下：

```
[root@redflag /root]# htpasswd -b /usr/apache/conf/users test 123456
```

执行完上述的命令后，可以查看 `users` 文件的内容，操作如下：

```
[root@redflag /root]# cat /usr/apache/conf/users
```

内容如下：

```
lgm:bBdPD1.jOo3tQ
```

```
test:3PIo6y6wDBuI2
```

(3) 测试。重新启动 Apache 服务器后，在浏览器中输入本机 IP，会出现如图 12-6 所示的画面，要求输入用户名和密码才能访问。

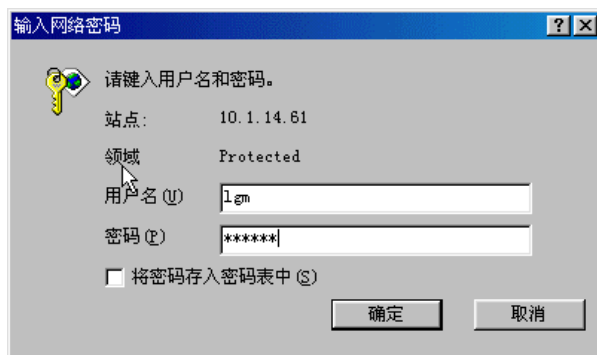


图 12-6 访问需要输入用户名和密码

12.5 Apache 常见故障排除

由于 Apache 服务器的配置大部分都是手工完成的，所以不可避免地会出现错误，下面介绍几种 Apache 常见故障的排除方法。

1. 检查服务器的状态

在得到服务器的完整状态之前，用户需要确信自己已经将 Lynx 基于文本的 Web 浏览器安装在自己的系统上。如果已安装了 Lynx，就可以通过运行下面的 `apachectl` 脚本来获得所运行的服务器的完整状态：

```
[root@redflag /root]# /usr/apache/bin/apachectl fullstatus
```

这样用户就会得到一个充满了各种服务器状态数据的信息页。最好是将这个状态信息重定向到一个文件中，因为文件可以将数据完好地保存下来。也可以通过下面的指令将数据重定向到一个叫做/tmp/status 的文件中：

```
[root@redflag /root]# /usr/apache/bin/apachectl fullstatus > /tmp/status
```

2. 查看服务器配置文件

要想检查服务器配置文件有没有语法错误，你可以运行下面的 apachectl 脚本：

```
[root@redflag /root]# /usr/apache/bin/apachectl configtest
```

3. 检查日志文件

在/var/log/httpd 目录下提供了访问记录文件 access_log 以及出错文件 error_log，所以当出现故障时，可以首先检查这些记录文件，一般故障都会在文件中指出。常出现的错误有文档路径错误、CGI 脚本程序没有执行权限、CGI 脚本程序本身有错误等。检查的文件如下：

(1) 检查访问日志。要想查看访问日志文件，可以进入日志文件目录并运行下面的命令：

```
tail -f [/path /to /access_log]
```

命令中的 tail 是一个 Linux 工具，它使你可以查看一个不断增长的文件(当使用了 -f 选项时)。现在，用 Web 浏览器来访问该站点。如果你已经进入了该站点，那么只需刷新一下浏览器，就会看到一个输入被添加到屏幕上的清单里。多次点击刷新按钮，看看访问文件是否被正确更新。如果更新正确，表示访问日志文件工作正常。按【Ctrl-C】退出 tail 命令。如果没有在日志文件中看到新记录，则应该检查一下日志文件和其所保存在的目录的许可权设置。

(2) 检查错误日志。要想查看错误日志文件，可以进入日志文件目录并运行下面的命令：

```
tail -f [/path /to /error_log]
```

(3) 检查服务器故障。如果故障属于服务器本身故障，可以使用下面的目录检查配置文件的设置：

```
检查配置文件句法    /usr/sbin/httpd -t
```

```
检查虚拟主机的配置  /usr/sbin/httpd -S
```

4. 检查端口设置

因为在安装 Linux 时已经安装了 Apache 软件，而且在系统启动的时候也启动了 Apache，那么它将使用默认的端口，当启动新版本的 Apache 时，会发生冲突，而且在浏览的时候肯定看到的是旧版本的 Apache 服务器提供的服务，这时我们需要关掉旧版本的 Apache 服务器，执行的指令如下：

```
[root@redflag /root]# /etc/rc.d/init.d/httpd stop
```

然后再重新启动新版本的 Apache 服务器即可。

本章小结

Apache 服务器以功能强大、配置简单、使用代价小而博得了大量用户的青睐，很快成为使用最广泛的 Web 服务器。本文系统而全面地介绍了 Apache 服务器及其相关知识。

用户可以在安装 Linux 时选择安装 Web Server，这样 Apache 服务器将自动安装；否则，

可以使用 configure 脚本来安装。

每个希望能对 Apache 服务器进行熟练配置的技术人员都必须深入了解三个配置文件：access.conf、httpd.conf、srm.conf。你可以通过在配置文件中加入命令来实现各种奇妙的功能，例如，设置服务器本身的信息，设定某个目录的功能和访问的控制信息、告诉服务器你想从你的 Web 服务器提供何种资源，以及从哪里、如何提供这些资源等。

Apache 服务器提供了很好的访问安全机制。对访问安全控制的讨论是本文的重点。你可以通过使用 Allow、Deny、Option 这三条指令将你的站点设置为需要根据客户主机名或 IP 地址限制访问。用户认证可以用来限制某些文档的访问权限，当我们将资源放在用户认证下面时，就可以要求用户输入名字和口令来获得对资源的访问。Apache 服务器提供两种类型的虚拟主机：基于 IP 的虚拟主机和基于名称的虚拟主机。如同其他的 Web 服务器，Apache 也提供了 Proxy 服务。相信随着大家对 Apache 服务器的熟练应用，更加能够体会它的强大的功能。

习 题

1. 最新版本的 Apache 的配置文件是_____。
A. access.conf B. srm.conf C. httpd.conf D. http.conf
2. 世界上排名第一的 Web 服务器是_____。
A. Apache B. Microsoft C. SunONE D. NCSA
3. 检查是否安装了 Apache 软件包的指令是_____。
A. rpm -x apache B. rpm -r apache C. rpm -t apache D. rpm -q apache
4. Apache 服务器的默认工作方式是_____。
A. inetd B. xinetd C. standby D. Standalone
5. 用来设定当服务器产生错误时，显示在浏览器上的管理员的 E-mail 地址的指令是_____。
A. ServerName B. ServerAdmin C. ServerRoot D. DocumentRoot
6. 我们设置 Apache 服务器时，一般将服务的端口绑定到系统的_____端口上。
A. 10 000 B. 23 C. 80 D. 53
7. 用户的主页存放的目录由文件 httpd.conf 的参数_____设定。
A. UserDir B. Directory C. pubic_html D. DirectoryIndex
8. 检查 Apache 配置文件句法是./httpd 的_____参数。
A. -t B. -S C. -l D. -c
9. 下面_____不是 Apache 基于主机的访问控制指令。
A. Allow B. Deny C. Order D. All
10. 在 Apache 基于用户名的访问控制中，生成用户密码文件的命令是_____。
A. smbpasswd B. htpasswd C. passwd D. password
11. 检查服务器的状态的命令是_____。
A. configtest B. apachectl fullstatus
C. apachectl graceful D. netstatus

第 13 章 FTP

FTP 是(File Transfer Protocol)文件传输协议的缩写,FTP 服务器能够在网络上提供文件传输服务。FTP 最初与 WWW 服务和邮件服务一起被列为因特网的三大应用,可见其在网络应用中的地位举足轻重。

13.1 FTP 简介

13.1.1 文件传输协议

文件传输协议是 Internet 上使用得最广泛的文件传送工具,服务器端使用 21 端口。FTP 提供交互式的访问,用来在远程主机与本地主机之间或两台远程主机之间传输文件。FTP 不仅可从远程主机上获取文件,而且可以将文件从本地主机传送到远程主机,如图 13-1 所示。

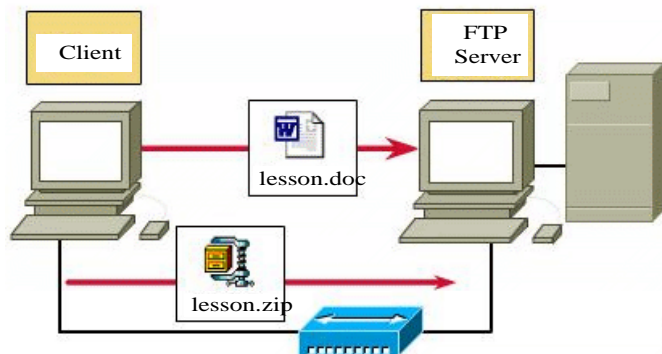


图 13-1 文件传输

在 Internet 上有两类 FTP 服务器。一类是普通的 FTP 服务器,连接到这种 FTP 服务器上时,用户必须具有合法的用户名和口令;另一类是匿名 FTP 服务器。所谓匿名 FTP,是指在访问远程计算机时,不需要账号或口令就能访问许多文件信息资源。用户不需要经过注册就可以与它连接并且进行下载和上传文件的操作,通常这种访问限制在公共目录下。

FTP 提供的命令十分丰富,涉及文件传输、文件管理、目录管理、连接管理等。目前世界上有很多文件服务系统,为用户提供公用软件、技术通报、论文研究报告,这就使 Internet 成为目前世界上最大的软件和信息流通渠道。Internet 是一个资源宝库,有很多共享软件、免费程序、学术文献、影像资料、图片、文字、动画,它们都允许用户用 FTP 下载。人们可以直接使用 WWW 浏览器去搜索所需要的文件,然后利用 WWW 浏览器所支持的 FTP 功能下载文件。

目前在 Unix 和 Linux 下常用的免费 FTP 服务器软件主要是 wu-ftp 和 proftpd 这两种。wu-ftp 广泛应用在众多的 Unix 和 Linux 系统中,是很多 Linux 版本默认的 FTP 服务器软件。

13.1.2 FTP 命令

FTP 命令在前面的章节已简单介绍过,这里将作进一步介绍。文件传输软件的使用格式为:

FTP(网址或 IP 地址)

若连接成功,系统将提示用户输入用户名及口令。

LOGIN: 输入合法的用户名或者“anonymous”。

PASSWORD: 输入合法的口令,若以“anonymous”方式登录,一般不用口令。

进入想要连接的 FTP 站点后,用户就可以进行相应的文件传输操作了,其中一些重要的命令及相似的命令如下。

1. help、?、rhelp

(1) help 显示 LOCAL 端的命令说明,若不接受则显示所有可用命令;

(2) ?相当于 help,例如:“?cd”;

(3) rhelp 同 help,只是它用来显示 REMOTE 端的命令说明。

2. ascii、binary、image、type

(1) ascii 切换传输模式为文字模式;

(2) binary 切换传输模式为二进制模式;

(3) image 相当于 binary;

(4) type 可更改或显示目前传输模式。

3. bye、quit

(1) bye 表示退出 FTP 服务器;

(2) quit 相当于 bye。

4. cd、cdup、lcd、pwd、!

(1) cd 改变当前工作目录;

(2) cdup 回到上一层目录,相当于“cd..”;

(3) lcd 可更改或显示 LOCAL 端的工作目录;

(4) pwd 显示目前的工作目录(REMOTE 端);

(5) !可执行外壳命令,例如:“!ls”。

5. delete、mdelete、rename

(1) delete 删除 REMOTE 端的文件;

(2) mdelete 批量删除文件;

(3) rename 更改 REMOTE 端的文件名。

6. get、mget、put、mput、recv、send

(1) get 下载文件;

- (2) mget 批量下载文件；
- (3) put 上传文件；
- (4) mput 批量上传文件；
- (5) recv 相当于 get；
- (6) send 相当于 put。

7. hash、verbose、status、bell

- (1) hash 当有数据传送时，显示“#”号，每一个#号表示传送了 1024 / 8192 bytes/bits；
- (2) verbose 切换所有文件传输过程的显示；
- (3) status 显示目前的一些参数；
- (4) bell 当指令执行完时会发出叫声。

8. ls、dir、mls、mdir、mkdir、rmdir

- (1) ls 有点像 Unix 下的 ls(list)命令；
- (2) dir 相当于“ls -l”；
- (3) mls 只是将远端某目录下的文件存于 LOCAL 端的某文件里；
- (4) mdir 相当于 mls；
- (5) mkdir 像 DOS 下的 MD(创建子目录)一样；
- (6) rmdir 像 DOS 下的 RD(删除子目录)一样。

9. open、close、disconnect、user

- (1) open 连接某个远端 FTP 服务器；
- (2) close 关闭目前的连接；
- (3) disconnect 相当于 close；
- (4) user 再输入一次用户名和口令(有点像 Linux 下的 su)。

当我们执行不同命令的时候，会发现 FTP 服务器返回一组数字，每组数字代表不同的信息，常见的数字及表示的信息如表 13-1 所示。

表 13-1 访问 FTP 服务器命令的返回值及含义

数字	含 义	数字	含 义
125	打开数据连接，传输开始	230	用户成功登录
200	命令被接受	331	用户名被接受，需要密码
211	系统状态，或者系统返回的帮助	421	服务不可用
212	目录状态	425	不能打开数据连接
213	文件状态	426	连接关闭，传输失败
214	帮助信息	452	写文件出错
220	服务就绪	500	语法错误，不可识别的命令
221	控制连接关闭	501	命令参数错误
225	打开数据连接，当前没有传输进程	502	命令不能执行
226	关闭数据连接	503	命令顺序错误
227	进入被动传输状态	530	登录不成功

13.1.3 wu-ftp

1. wu-ftp 简介

wu-ftp 全称是 Washington University FTP, 现在因特网上 FTP 服务器的安装软件大多使用的都是 wu-ftp 软件, 因为它不仅支持多个操作系统平台, 而且还具有如下的功能:

- (1) 让用户在下载文件的同时可以对文件做自动的压缩或解压缩操作;
- (2) 可以对不同网络的机器做相应的存取限制和存取时间设定;
- (3) 可以记录文件上传和下载的时间;
- (4) 可以显示传输时的相关信息, 以便让用户知道目前的传输状态;
- (5) 可以设定连接的数量限制, 以提高工作效率。

2. wu-ftp 的安装

如果你在安装 Linux 系统的过程中, 在选择启动进程的时候选择了 “xinetd” 这一项的话, 安装完 Linux 系统后, 就已经将一个默认的 FTP 服务器安装到系统中去了, 我们就可以利用它来实现系统 FTP 服务器的功能了。在此基础上我们只需根据需要进行一些个性化设定就可以了, 用 “rpm -q” 命令检查是否安装了 wu-ftp 软件包:

```
[root@redflag /root]#rpm -q wu-ftpd
```

当然你可以到 www.wu-ftp.org 去下载 wu-ftp 的最新版本, 或者在 Linux 的安装光盘上找到 wu-ftp 的 RPM 包, 运行如下的命令进行安装:

```
[root@redflag RPMS]#rpm -ivh wu-ftpd-2.6.2-7.rpm
```

安装了 wu-ftp 后, 将在 /usr/bin 目录下可以看到以下四个可执行文件:

- (1) ftpshut: 用于关闭 FTP 服务器程序;
- (2) ftpcount: 显示目前登录到 FTP 服务器的在线人数;
- (3) ftpwho: 查看目前 FTP 服务器的连接情况;
- (4) ckconfig: 检查 FTP 服务器的设置是否正确。

除了这些可执行文件以外, 它还在 /etc 和 /var 目录下生成了六个相关文件:

- (1) /etc/ftpusers: 设定哪些账号不能用于访问 FTP 服务器;
- (2) /etc/ftpaccess: wu-ftp 的主要配置文件;
- (3) /etc/ftpconversions: 配置文件压缩/解压缩转换;
- (4) /var/log/xferlog: 存储日志文件;
- (5) /etc/ftpgroups: 设定 FTP 自己定义的群组;
- (6) /etc/ftphosts: 设定个别的用户权限。

3. wu-ftpd 的启动

在登录 FTP 服务器之前, 必须先检查 /etc/xinetd.d/wu-ftpd 文件的内容, 其内容如下:

```
#default on
# description: The wu-ftpd FTP server serves FTP connections. It uses \
#             normal, unencrypted usernames and passwords for authentication.
service ftp
{
```

```
disable = no
socket_type      = stream
wait             = no
user             = root
server           = /usr/sbin/in.ftpd
server_args      = -l -a
log_on_success   += DURATION USERID
log_on_failure   += USERID
nice             = 10
}
```

【说明】 Disable 的值必须设为 “no”。

第六行 “server_args = -l -a” 是 wu-ftpd 的执行参数，-l 和 -a 也是默认的设置，常见的参数及其含义如表 13-2 所示。

表 13-2 wu-ftpd 的执行参数及含义

参数	含 义
-d	当 FTP 服务器发生错误时，将错误写入系统的 syslog 文件中
-l	将每次 FTP 客户端进行的连接写入系统的 syslog 文件中
-t	设置 FTP 客户端连接几分钟无操作就切断连接
-a	使 wu-ftp 使用/etc/ftpaccess 的设定
-A	使 wu-ftp 不使用/etc/ftpaccess 的设定
-L	将 FTP 客户端连线后所执行的程序记录在系统的 syslog 文件中
-I	将 FTP 客户端上传的文件记录在日志文件中
-o	将 FTP 客户端下载的文件记录在日志文件中

【注意】 syslog 提供了一种可以发送错误、状态和调试信息到控制台或日志文件中的机制。通常日志文件的位置会随着 Linux 的版本的不同而不同。在红旗 Linux 中，可以在 /var/log 目录下找到 messages、maillog 等日志文件。

修改完/etc/xinetd.d/wu-ftpd 文件的内容后，执行下面的指令来启动 FTP 服务器。

```
[root@redflag /root]#/etc/rc.d/init.d/xinetd restart 或
```

```
[root@redflag /root]#service xinetd restart
```

13.2 配置 wu-ftp 服务器

虽然 FTP 服务器只要安装好了就可以使用，但是在/etc 目录下还有一些关于 FTP 服务器的配置文件，如 ftpaccess、ftpusers 等文件，可以帮助我们完成诸如访问权限等任务。

13.2.1 ftpaccess 文件

ftpaccess 是 FTP 最主要的配置文件，而且修改后不需要重新启动 wu-ftp 就可以生效。下面我们以实例来详细讲述该文件的内容。

wu-ftp 服务器的 IP 地址为 10.1.14.61, ftpaccess 文件的内容涉及到的选项如下(为了让大家很好的对应和方便的学习, 我们仍然采用逐句逐行的解释)。

1. deny-gid 与 deny-uid

deny-gid 与 deny-uid 主要定义系统中有哪些组和用户不能登录 FTP 服务器。下面两项是常用的的设置。

```
deny-uid %-99 %65534-
```

```
deny-gid %-99 %65534-
```

以 deny-gid 为例, 在 deny-uid 中定义了两个范围:

(1) %-99: 表示 gid 编号在 99 之前。

(2) %65534-: 表示 gid 编号在 65 534 之后。

【说明】 定义范围的正确写法是 %x-y。如果省略 x, 也就是只注明 %-y, 表示编号的 y 之前的所有组或用户。如果省略 y, 也就是只注明 %x-, 表示编号在 x 之后的所有组 and 用户。

因此如果只要组编号, 那么这两个范围的组的用户都不能登录 FTP 服务器。此外也可以用组名的方式来限制特定的组不允许登录 FTP 服务器, 如下面一行:

```
deny-gid nobody
```

表示 nobody 的组的用户将无法登录。

由此可见, deny-gid 后面接的是组的名称或者组的 gid。如果是组编号范围, 则可以同时指定多组, 每组范围之间只要用空格隔开即可; 但是如果是组名称, 则只能有一个。deny-uid 与 deny-gid 的规则相同, 这里就不详细讲述了。

2. allow-uid 与 allow-gid

如果要想让 deny-uid 或 deny-gid 范围中所限制的特定的用户也能登录服务器, 则可以用 allow-uid 或 allow-gid 进行设置。从 deny-gid %-99 来看, 组编号在 99 之前的用户将无法登录服务器, 但是在这个范围中, 包含了 FTP 服务器的重要组——FTP 组, 因此为了使 FTP 组能够正常登录, 默认有如下两行:

```
allow-uid ftp
```

```
allow-gid ftp
```

这样 FTP 组的用户就不会受到 deny-gid %-99 或 deny-uid %-99 的限制, 可以正常登录 FTP 服务器。

3. guestgroup、restricted-uid 和 guestuser

guestgroup、restricted-uid 和 guestuser 都是和 FTP 服务器的安全相关的, 它们的作用都是在用户登录成功后, 把用户限制在自己的主目录下, 不让用户到处乱跑, 对服务器起到安全保护作用。

(1) guestgroup 使用 guestgroup 可以用来指定 FTP 服务器中属于 guest 类的组。这些组的用户登录后的权限会和 guest 相同, 且只能在自己的主目录中活动, 内容如下面一行:

```
guestgroup ftpchgroup
```

其中 ftpchgroup 组只是一个例子, 不是固定的, 当然用户可以根据自己的实际情况来定义, 然后把需要设定 guest 权限的用户加入到该组即可。

(2) `restricted-uid` 使用 `guestgroup` 大家可能觉得麻烦, `restricted-uid` 同样可以实现防止用户到处乱跑, 而且只需要如下一行命令即可:

```
restricted-uid *
```

(3) `guestuser` 使用 `guestuser` 同样可以实现将用户限制在自己的主目录, 具体实现如下:

```
guestuser *  
real user user1, user2
```

其中 `real` 可以实现用户 `user1` 和 `user2` 突破 “`guestuser *`” 的限制。

当 `guestgroup`、`restricted-uid` 或 `guestuser` 选项生效时, 如果用户企图进入 FTP 服务器的其他目录时, 如 `/etc` 目录, 就会出现下面的信息:

```
[lgm@redflag lgm]$ ftp 10.1.14.61  
Connected to 10.1.14.61.  
220 redflag.com FTP server (Version wu-2.6.1-16) ready.  
530 Please login with USER and PASS.  
530 Please login with USER and PASS.  
KERBEROS_V4 rejected as an authentication type  
Name (10.1.14.61:lgm): lgm  
331 Password required for lgm.  
Password:  
230 User lgm logged in.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> cd /etc  
550 /etc: No such file or directory.  
ftp>quit
```

4. class

`class` 的功能设定 FTP 服务器上用户的类别, 并可对客户端的 IP 地址进行限制, 允许某部分的 IP 地址或全部的 IP 地址访问。格式如下:

```
class [类名] [用户类别] [IP 地址]
```

在 FTP 服务器上的用户基本可以分为以下三类。

(1) `real`: 在该 FTP 服务器有合法账号的用户。此类用户使用 FTP 登录时, 默认登录后的目录是其主目录。例如, `lgm` 用户登录后, 进入的目录为 `/home/lgm`, 但是只要他拥有某目录的访问权限, 即使该目录不在 `/home/lgm` 下, 也能切换到该目录下。

(2) `guest`: 某些情况下, 管理者可能希望某些拥有账号的用户登录后, 只能访问其主目录的文件, 而不允许到处乱跑。这时, 管理者就可以使用 `guestgroup` 或 `restricted-uid` 来进行设定。

(3) `anonymous`: 在 FTP 服务器上没有账号的匿名用户登录时可以使用 “`anonymous`”

和 E-mail 地址作为账号和密码。登录的目录是/var/ftp。匿名用户只能在这个目录下活动，而不能进入到其他任何目录。

下面三行定义了 all、local 和 remote 三个不同类的用户。

(1) class all real, anonymous , guest * : 本行定义了一个名为 all 的类。它包含了在任何地方登录(“*”代表所有 IP 地址，当然在这里也可以是某些主机具体的 IP 地址)的所有用户，即包括 real 用户、anonymous 用户和 guest 用户。

(2) class local real * : 本行定义了一个名为 local 的类，它包含了在任何地方登录的 real 用户。

(3) class remote anonymous , guest * : 本行定义了一个名为 remote 的类，它包含了在任何地方登录的 anonymous 用户和 guest 用户。

5. email

email 的功能是指定 FTP 服务器管理员的 E-mail 地址。在服务器运行过程中，如果需要显示便会列出此处的设置值，如下面一行：

```
email webmaster@redflag.com
```

6. loginfails

loginfails 的功能是设定当用户登录到 FTP 服务器时，允许用户输错密码的次数。格式如下：

```
loginfails 3
```

其中“3”表示允许用户输错密码的次数。下面的操作是验证的过程及显示的相应的信息：

```
[lgm@redflag lgm]$ ftp 10.1.14.61
Connected to 10.1.14.61.
220 redflag.com FTP server (Version wu-2.6.1-16) ready.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (10.1.14.61:root): lgm
331 Password required for lgm.
Password:
530 Login incorrect.
Login failed.
ftp> user lgm
331 Password required for lgm.
Password:
530 Login incorrect.
Login failed.
ftp> user lgm
331 Password required for lgm.
```

```
Password:
530 Login incorrect.
Login failed.
ftp> user lgm
421 Service not available, remote server has closed connection
Login failed.
ftp> quit
```

7. readme

readme 的功能是用来提醒用户读取说明文件的。格式如下：

```
readme [路径和文件名] [时机] [组名称]
```

下面两行是具体的配置：

```
readme  README*      login
readme  README*      cwd=*
```

上面两行分别表示用户在登录系统和改变目录时提醒用户读取位于用户当前目录下的 readme 所用到的文件。cwd 是 change working directory 的英文缩写。

8. message

message 的功能是用来显示欢迎的信息的。格式如下：

```
message [路径和文件名] [时机] [组名称]
```

下面两行是具体的配置：

```
message /welcome.msg      login
message .message          cwd=*
```

登录时显示 “welcome.msg”，改变目录时显示该目录的 “.message”。我们可以为每个目录建立一个 “.message” 文件，在 message 的信息文件中还可以使用表 13-3 所示的变量，使信息更加丰富一些。当然文件所在的目录由用户根据自己的需要来定义。

表 13-3 message 经常使用的变量

变量	说 明
%T	FTP 服务器主机的时间
%C	当前所在的目录名
%E	FTP 服务器管理员的 E-mail 地址
%R	用户所在的远程主机名
%L	FTP 服务器主机名
%U	用户登录时给定的名称
%M	同一组可允许最多几个用户登录
%N	同一组当前已经登录的用户数目

接下来我们就可以用上表中的变量来编辑个性化的 welcome.msg 文件，其内容如下：

```
*****
*You are from %R.
*The address of my host is %L
*Time is %T
*Any problem mail to %E
*****
```

用 FTP 命令登录的结果显示如下：

```
[lgm@redflag lgm]$ ftp 10.1.14.61
Connected to 10.1.14.61.
220 redflag.com FTP server (Version wu-2.6.1-16) ready.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (10.1.14.61:lgm): lgm
331 Password required for lgm.
Password:
230- *****
230- *You are from hosta.redflag.com.
230- *The address of my host is redflag.com
230- *Time is Thu Apr 17 17:21:36 2003
230- *Any problem mail to root@redflag.com
230- *****
230-
230 User lgm logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

9. 压缩命令

常用的压缩命令有 compress 和 tar。

(1) compress compress 的功能是设置哪一个类别的用户可以使用 compress(压缩)功能。格式如下：

```
compress [yes/no] [类别]
```

具体配置如下：

```
compress yes local remote
```

允许 local 和 remote 两个类别的用户都能使用 compress(压缩)功能。

(2) tar tar 的功能是设置哪一个类别的用户可以使用 tar(归档)功能。格式如下：

```
tar [yes/no] [类别]
```

具体配置如下：

```
tar yes all
```

允许 all 类的用户使用 tar 功能。

10. 执行命令的权限

在 FTP 服务器下“执行命令”通常包括 chmod、delete、overwrite 和 rename，格式如下：

```
命令 [yes/no] [real/anonymous/guest]
```

具体配置如下：

```
chmod      no      guest,anonymous
delete     no      anonymous
overwrite no      anonymous
rename     no      anonymous
```

上面四行是设置用户具有的权限。例如在第一行中，设置 guest 和 anonymous 组用户没有执行 chmod 命令的权限，只有 real 组有执行 chmod 命令的权限。为了更好地管理 FTP 服务器，一般情况下，我们不允许匿名用户和 guest 用户执行 chmod 等命令。其余三行含义类似，这里就不一一解释了。

11. log

log 是用来生成日志文件的，这对服务器的维护和管理非常重要，因为管理员可以很容易地知道诸如用户执行了哪些命令、上传和下载了哪些文件等。通常和 log 相关的操作有 commands 和 transfers 两项。

(1) commands commands 的功能是设置哪些用户登录后的操作记录在文件 /var/log/xferlog 中。格式如下：

```
log commands [real/guest/anonymous]
```

具体配置如下：

```
log commands real
```

当 real 用户登录后，将他的操作记录下来。由于其他用户权限较低，所以操作不会引起太大的安全隐患，所以一般只需记下 real 用户的操作就可以了。

(2) transfers transfers 的功能是设置对哪些用户的上传(inbound)和下载(outbound)操作做日志。格式如下：

```
log transfers [real/guest/anonymous] [inbound/outbound]
```

具体配置如下：

```
log transfers anonymous , guest inbound , outbound
```

对于匿名用户和 guest 用户要更加关注它们的文件操作，所以无论上传、下载都要进行记录。

12. shutdown

shutdown 的功能是设定 FTP 服务器关闭的时间。格式如下：

```
shutdown [文件名]
```

配置如下：

```
shutdown /etc/shutmsg
```

当 FTP 服务器启动时会检查 shutdown 的设置 然后到指定的/etc/shutmsg 文件中去检查关机的时间的设置。若是找不到 shutmsg 文件，则会忽略此项目。shutmsg 文件是有具体格式的，下面显示的内容是 shutmsg 文件的内容。

```
2003 4 15 23 30 20 5
```

其中：

2003—表示年；

4 —表示月份，可选范围为 0~11(1 月为 0，二月为 1.....)；

15 —表示日期，可选范围为 1~31；

23 —表示小时，可选范围为 0~23；

30 —表示分钟，可选范围为 0~59；

20 —表示在关闭之前 20 分钟(即 23:10)禁止用户登录服务器；

5 —表示在关闭之前 5 分钟(即 23:25)禁止已经登录服务器的用户。

FTP 服务器关闭的时间一到，便无法登录 FTP 服务器了，要恢复的话只有将 shutmsg 这个文件删除掉。否则 FTP 服务器发现 shutmsg 中的关机时间已经过去，就不会允许登录。而 shutmsg 这个文件必须由指令 ftpshut 来生成，详细内容请参阅 13.3.3 节。

13. passwd-check

passwd-check 功能是设定对匿名用户 anonymous 的密码使用方式。格式如下：

```
passwd-check [通用标准] [系统回应]
```

通用标准有三种，分别为以下三种。

(1) none：表示不做密码验证，任何密码都可以登录；

(2) trival：表示只要输入的密码中含有字符“@”就可以登录；

(3) rfc822：表示密码一定要符合 RFC822 中所规定的 E-mail 格式才能登录。

系统回应有两种，分别为以下两种。

(1) enforce：表示输入的密码不符合以上指定的格式就不让登录；

(2) warn：表示密码不符合规定时只出现警告信息，但仍然能够登录。

具体配置如下：

```
passwd-check rfc822 warn
```

希望能够得到符合规定的 E-mail 作为密码，但如果不是，也允许登录。

14. deny

deny 的功能是限制某些 IP 地址或域名的用户登入 FTP 服务器。格式如下：

```
deny [IP 地址/域名] [说明文件]
```

说明文件要自己创建，当用户要从被拒绝的 IP 地址连接 FTP 服务器时，就会出现说明文件的内容而谢绝登录。

具体配置如下：

```
deny 10.1.14.125 /etc/deny.msg
```

拒绝 IP 地址为 10.1.14.125 的主机进行 FTP 访问。

15. limit

limit 的功能是设置在指定的时间段内允许多少用户登录本机。格式如下：

```
limit [类别] [人数] [时间] [文件名]
```

当达到人数上限的时候，显示指定文件的内容。

具体配置如下：

```
limit remote 200 SaSu|Any1800-0600 /etc/toomany.msg
```

表示星期六(Sa)和星期日(Su)允许 200 人 FTP 远程登录，其他任何时间(星期一至星期五)只有在晚上 6 点到凌晨 6 点允许 200 人远程 FTP 登录。当然 remote 类别一定要在前面定义才可以。

```
limit all 100 Any /etc/toomany.msg
```

表示在任何时间内，all 类的访问用户达到 100 人时，将不再允许产生新的连接，当第 101 位客户要连接时，连接将失败，并向用户显示文件/etc/ftpd/toomany.msg 的内容。

16. upload

upload 的功能是对可以上传的目录进行更加详细的权限设置。格式如下：

```
upload [根目录] [子目录] [是否可以上传] [拥有者] [拥有组] [文件权限]  
[能否创建子目录]
```

说明如下：

- (1) 根目录：指的是 ftp 的根目录，即登录时所见到的根目录；
- (2) 子目录：指可供上传的子目录；
- (3) 是否可以上传：设置此目录是否可以上传文件，“yes”表示可以，“no”表示不可以；
- (4) 拥有者/拥有组：文件上传后赋予它的拥有者和拥有组的名称；
- (5) 文件权限：文件上传后的权限；
- (6) 能否创建子目录：“dirs”表示可以，“nodirs”表示不可以。

具体配置如下：

```
upload /var/ftp * no
```

表示在子目录/var/ftp 下不允许上传。

```
upload /var/ftp /bin no
```

表示在子目录/var/ftp/bin 下不允许上传。

```
upload /var/ftp /etc no
```

表示在子目录/var/ftp/etc 下不允许上传。

```
upload /var/ftp /pub yes ftp ftp 0644 dirs
```

允许用服务器上的合法用户在子目录/var/ftp/pub 目录下上传权限为 0644(也就是 -rw-r--r--)的文件，而且在这个目录下可以新建子目录。

17. alias

alias 的功能是给指定目录设置一个别名，这样在切换目录时就可以使用较短的目录别名。格式如下：

```
alias [目录别名] [目录名]
```

具体配置如下：

```
alias R pub
```

表示给予目录 pub 设置一个别名 R。

18. cdfpath

cdfpath 的功能是提供直接进入某个搜索路径。格式如下：

```
cdfpath [目录名]
```

具体配置如下：

```
cdfpath /var/ftp
```

```
cdfpath /var/ftp/pub
```

```
cdfpath /var/ftp/etc
```

假设用户输入“ cd linux ”命令 ,系统就会依次搜索是否有/var/ftp/linux、/var/ftp/pub/linux、/var/ftp/etc/linux 目录存在 , 如果找到就进入该目录。

19. path-filter

path-filter 的功能是限制上传的文件名可以包括哪些字符。

具体配置如下：

```
path-filter anonymous /etc/badfilename_msg ^[A-Za-z0-9]*$ ^\.\^-
```

```
path-filter guest /etc/badfilename_msg ^[A-Za-z0-9]*$ ^\.\^-
```

表示 anonymous 和 guest 类的用户上传的文件名可以包括 A-Z、a-z 和 0-9 , 且不可以用 “ . ” 或 “ - ” 开头。如果文件名不在规定的范围之内 , 则会显示文件 badfilename_msg 的内容。

20. file-limit

file-limit 的功能是限制某 class 只能传几个文件。格式如下：

```
file-limit [raw] [in|out|total] [count] [class]
```

对某个 class 限制存取文件的数目 , 包含了 in(上传)、out(下载)、total , raw 代表不光是数据文件 , 而是整个传输的结果。具体配置如下：

```
file-limit out 20 remote
```

表示 remote 类的用户最多能下载 20 个文件 , 如果超过了这个限制 , 系统会返回“ Transfer limits exceeded. ” 信息。

21. limit-time

limit-time 的功能是限制用户的访问保持时间。格式如下：

```
limit-time <typelist> <minutes>
```

typelist 可以是 anonymous 或者 guest , 正式用户(real 类)不会受到限制。

具体配置如下：

```
limit-time anonymous 10
```

```
limit-time guest 10
```

上面两行表示对 anonymous 和 guest 用户的访问时间不能超过 10 分钟 , 否则系统会返回 “ 221 Time limit reached. Good bye. ” 信息。

【说明】 ftpaccess 可设置的选项很多 , 详细说明可以执行 man ftpaccess 命令查询。

13.2.2 ftpusers

有时我们需要禁止一些用户使用 FTP 服务，其设置是十分简单的，只需将要禁止的用户账号写入文件/etc/ftpusers 中。基于系统的安全考虑，一般我们不希望权限过大的用户和一些与命令名相同的用户进入 FTP 服务器。所以在缺省的配置中，一般以下用户已经被列入了“黑名单”：

```
root uucp news bin adm nobody lp sync shutdown halt mail
```

当然在 ftpusers 文件中，每个用户名占一行。

【实例 13.1】

本实例用来实现拒绝 lgm 用户进行 FTP 连接。

- (1) 将 lgm 加入 ftpusers 文件中。
- (2) 重新启动 xinetd 进程。
- (3) 进行测试，操作步骤如下：

```
[lgm@redflag lgm]$ ftp 10.1.14.61
Connected to 10.1.14.61.
220 redflag.com FTP server (Version wu-2.6.1-16) ready.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (10.1.14.61:lgm): lgm
331 Password required for lgm.
Password:
530 Login incorrect.
Login failed.
```

13.2.3 ftphosts

ftphosts 文件的功能是禁止某些来自指定机器上的登录。如果需要拒绝来自某些主机的登录，一种方法就是在/etc/ftpaccess 中设置“deny”命令，另一种更加简单的方法就是在/etc/ftphosts 中写入要禁止的主机的 IP 地址或域名。

【实例 13.2】

本实例用来验证 ftphosts 文件的功能。

ftphosts 文件的内容如下：

```
# Example host access file
#
# Everything after a '#' is treated as comment,
# empty lines are ignored
deny lgm 10.1.14.125
#表示拒绝 lgm 用户从 10.1.14.125 主机上登录。
allow test 10.1.14.66
#表示允许 test 用户从 10.1.14.66 主机上登录。
```



```
deny user1 10.1.14.0/255.255.255.0
```

#表示拒绝 user1 用户从网络 10.1.14.0/255.255.255.0 登录。

我们用 lgm 用户从主机 10.1.14.125 上登录，执行结果如图 13-2 所示：

```
C:\WINDOWS>ftp 10.1.14.61
Connected to 10.1.14.61.
220 redflag.com FTP server (Version wu-2.6.1-16) ready.
User (10.1.14.61:(none)): lgm
331 Password required for lgm.
Password:
530 Login incorrect.
Login failed.
ftp>
```

图 13-2 从主机 10.1.14.125 登录被拒绝

13.3 wu-ftp 的相关应用

13.3.1 连接数统计命令 ftpcount

我们可以使用 ftpcount 命令十分清楚地统计出当前连接到 FTP 服务器上的用户数目，并且同时列出上限。

【实例 13.3】

本实例用来统计连接到正在运行的 wu-ftp 服务器的用户的个数，操作如下：

```
[root@redflag /root]# ftpcount
Service class all - 3 users (no maximum)
```

13.3.2 在线用户查看命令 ftpwho

我们可以使用 ftpwho 命令十分清楚地列出当前连接的用户的详细情况。

【实例 13.4】

本实例用来显示连接到正在运行的 wu-ftp 服务器的用户的信息，操作如下：

```
[root@redflag /root]# ftpwho
Service class all:
1743 ? SN 0:00 ftpd: hosta.redflag.com: test1: IDLE
1738 ? SN 0:00 ftpd: 10.1.14.125: connected: IDLE
1714 ? SN 0:00 ftpd: hosta.redflag.com: lgm: IDLE
- 3 users (no maximum)
```

13.3.3 FTP 关闭文件生成命令 ftpshut

我们可以使用 ftpshut 命令生成一个在/etc/ftppaccess 中设置的 shut.msg 文件，用于关机设定。ftpshut 命令的格式为：

```
ftpshut [-l min] [-d min] time [说明]
```

-l：设定在关闭 FTP 服务器功能前多少分钟时停止用户的连接；

-d：设定在关闭 FTP 服务器功能前多少分钟时切断用户连接；

time :指定关闭 FTP 服务器的时间 ,例如希望在晚上 10 点关闭 FTP 服务器 ,则为 2200 ;
说明 :断线以前显示给用户的告警信息。

【实例 13.5】

```
[root@redflag /root]#ftpshut -l15 -d5 1800 "ftp server will shutdown."  
[root@redflag /root]# less shutmsg  
2003 04 07 18 00 0015 0005  
ftp server will shutdown.
```

13.3.4 用脚本实现自动 FTP

对于经常使用的 FTP 操作 ,我们可以编写成脚本 ,这样就不用每次反复做重复的工作 ,只需简单地运行一下脚本即可。下面我们用一个具体的实例来讲述如何用脚本来实现 FTP。

【实例 13.6】

本实例用来实现从服务器下载文件 a.txt ,然后把本地的 b.txt 文件上传到 FTP 服务器上。实现步骤如下 :

- (1) 创建文本文件 ftp.txt。内容如下 :

```
open 10.1.14.61  
#用 open 连接远程服务器 10.1.14.61。  
user test 123456  
#test 是用户名 , 123456 是密码。  
binary  
#以二进制传送。  
hash  
#当有数据传送时 , 显示 “ # ” 号。  
cd pub  
#进入远程目标路径 pub。  
get a.txt /home/test/a.txt  
#把远程文件 a.txt 下载到本地/home/test/目录下。  
put /root/b.txt b.txt  
#将本地文件 b.txt 上传到 FTP 服务器。  
bye  
#退出 FTP 应用。
```

- (2) 执行脚本。执行如下的命令即可执行上面的脚本 :

```
[root@redflag /root]#cat ftp.txt | ftp -n
```

13.4 wu-ftp 常见故障排除

由于 wu-ftp 服务器的配置大部分都是手工完成的 ,所以不可避免地会出现错误 ,下面介绍 wu-ftp 几种常见故障的排除方法。

13.4.1 检查 ftp 的配置文件

在 wu-ftp 安装目录下的 bin 子目录下执行 ckconfig，如下所示：

```
[root@redflag bin]#./ckconfig
Checking _PATH_FTPUSERS :: /etc/ftpusers
ok.
Checking _PATH_FTPSERVERS :: /etc/ftpservers
I can't find it... look in doc/examples for an example.
Checking _PATH_FTPACCESS :: /etc/ftpaccess
ok.
Checking _PATH_PIDNAMES :: /var/run/ftp.pids-%s
ok.
Checking _PATH_CVT :: /etc/ftpconversions
ok.
Checking _PATH_XFERLOG :: /var/log/xferlog
ok.
Checking _PATH_PRIVATE :: /etc/ftpgroups
ok.
Checking _PATH_FTPHOSTS :: /etc/ftphosts
ok.
```

这个操作是用来查看 FTP 的配置文件是否都在正确的位置，通常的配置文件有以下八个：

```
/etc/ftpusers /etc/ftpaccess /var/run/ftp.pids /etc/ftpconversions
/var/log/xferlog /etc/ftpgroups /etc/ftphosts /etc/ftpservers
```

如果执行 ckconfig 时，出现信息说找不到某些文件，你可以在 doc/examples/这个目录下找到相应的例子。就上面的例子来说，它出现的信息是找不到/etc/ftpservers 这个文件，该文件是用来依据客户的 IP 作相应访问控制的，基本上不影响 FTP 的正常工作。

13.4.2 查看 log 文件

log 日志文件对服务器的维护和管理非常重要，因为管理员可以很容易地知道诸如用户执行了哪些命令，上传和下载了哪些文件等信息。wu-ftp 的日志文件一般保存在 /var/log/xferlog 中，可以通过如下命令来查看：

```
[root@redflag /root]#tail -f /var/log/xferlog
```

本章小结

在众多的网络应用中，FTP(File Transfer Protocol)有着非常重要的地位。在 Internet 中一个十分重要的资源就是软件资源，而各种各样的软件资源大多数都是放在 FTP 服务器中的。在绝大多数的 Linux 发行版本中都选用的是 wu-ftp，它是一个著名的 FTP 服务器软件，其

功能强大，能够很好地运行于众多的 Unix 操作系统中。本章主要讲述了 FTP 基本原理、基本命令以及 wu-ftp 的配置和故障排除等内容。目前在 Internet 上的 FTP 服务器，一大半以上采用了 wu-ftp，所以我们应该掌握它的配置、管理和使用。

习 题

1. FTP 服务使用的是_____端口。
A. 21 B. 23 C. 25 D. 53
2. 我们从 Internet 上获得软件最常采用的是_____。
A. WWW B. Telnet C. FTP D. DNS
3. 在 Unix 和 Linux 下默认采用的免费的 FTP 软件是_____。
A. cute-ftp B. flash-ftp C. wu-ftp D. pro-ftp
4. 一次下载多个文件用_____命令。
A. mget B. get C. put D. send
5. 在 FTP 操作过程中，“530”表示_____。
A. 登录成功 B. 登录不成功 C. 服务就绪 D. 写文件出错
6. 用于关闭 FTP 服务器的程序是_____。
A. ftpwho B. ftpshut C. ftpcount D. ftpd
7. wu-ftp 最重要的配置文件是_____。
A. ftpaccess B. ftpusers C. ftphosts D. ftpservers
8. 下面_____不是 FTP 用户的类别。
A. real B. anonymous C. guest D. users
9. 下面_____不是 passwd-check 的通用标准。
A. none B. no C. trivial D. rfc822
10. 在 ftpaccess 文件中，_____对可以上传的目录进行更加详细的权限设置。
A. up B. upload C. put D. send

第 14 章 防 火 墙

随着网络的普及，人们对网络的安全已经越来越重视，而谈到网络的安全又会自然而然地想到“防火墙”(Firewall)。但是这里首先要强调的是，安全应是一个完整的策略，而不是系统中某一部分的安全。无论数据在网络上传输多么安全，但是如果用户的密码不设置，系统就谈不上丝毫的安全。最终系统的安全程度取决于系统的最薄弱环节，这和水桶所能盛的水取决于最短木板的高度的道理是一样的。我们这里所谈到的防火墙是整个安全策略中的一个部分而不是全部，只不过该部分在网络安全中占有比较重要的地位而已。

14.1 防火墙简介

14.1.1 防火墙的分类和基本工作原理

防火墙一般分为两大类，一类是“包过滤”型防火墙，另一类是“代理服务器”型防火墙。

1. 包过滤型防火墙(Packet Filter)

包过滤型防火墙内置于 Linux 内核，它和日常生活中门卫的作用是类似的。门卫把守着大门，根据上级的指示允许或不允许某些人员进出大楼。包过滤型防火墙根据预先的设置允许或不允许数据包通过防火墙。门卫是根据来往人员是什么样的人、从什么地方来、到什么地方去、想干什么等信息来决定是否放行，那么防火墙是根据什么来决定是否放行数据的呢？我们先来看看数据包中都包含了什么信息。

图 14-1 是一个以太网数据帧的格式，在帧内封装着 IP 数据包，IP 数据包又封装着传输层的数据报。

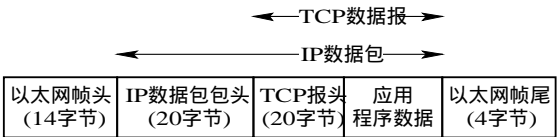


图 14-1 带全部 TCP 数据报的以太网数据帧格式

我们再来看一个 IP 数据包的格式(见图 14-2)。IP 数据包包头含有源 IP 地址、目的 IP 地址，如果数据包是一个 TCP 或 UDP 数据包，还会有协议类型。紧接着我们看一个 TCP 报文的格式(见图 14-3)。报文中包含有源端口号、目的端口号。有了 IP 地址，我们就可以

知道数据包是从什么主机发出的、到达哪个主机；而端口号通常表示什么服务，即是干什么的。

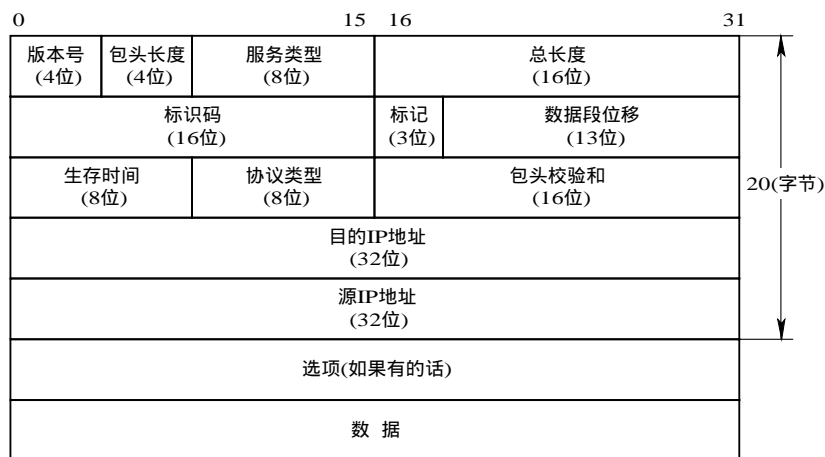


图 14-2 IP 数据包格式

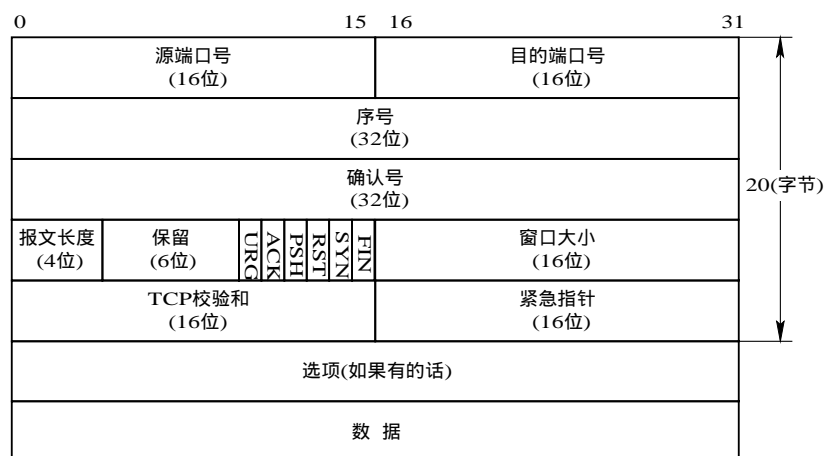


图 14-3 TCP 数据报文格式

包过滤型防火墙就是通过检查数据包中的源 IP 地址、目的 IP 地址、源端口号、目的端口号、协议类型等信息来决定是放行数据还是拒绝数据的。由于防火墙要能读得懂这些信息才能工作，而这些信息分别在 TCP/IP 协议的网络层、传输层等层中，所以说包过滤型防火墙工作在网络层或传输层中。图 14-4 是包过滤型防火墙的一种常用的模式，它用来阻隔来自外部网络对内部网络的威胁。

从以上的讨论中可以看到，只要是合法的数据就会被防火墙正常转发，应用程序感觉不到防火墙的存在，因此包过滤型防火墙有着较好的透明性，即应用程序无须做任何更改。同时由于包过滤型防火墙是内置于 Linux 内核的，也就具有较好的网络性能。

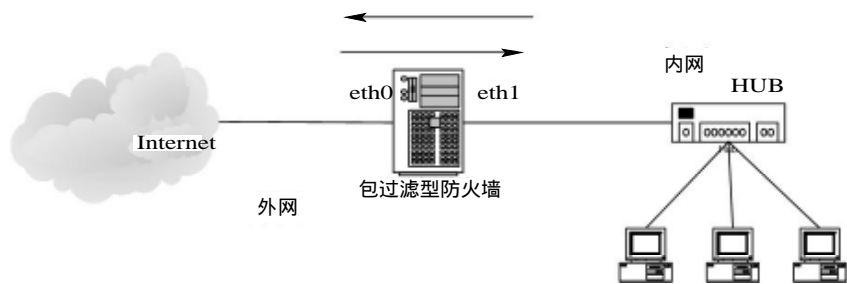


图 14-4 包过滤型防火墙的一种常用结构

2. 代理服务器型防火墙(Proxy Server)

代理服务器型防火墙是应用网关型防火墙，通常工作在应用层，图 14-5 是应用网关型防火墙的常用应用模式。

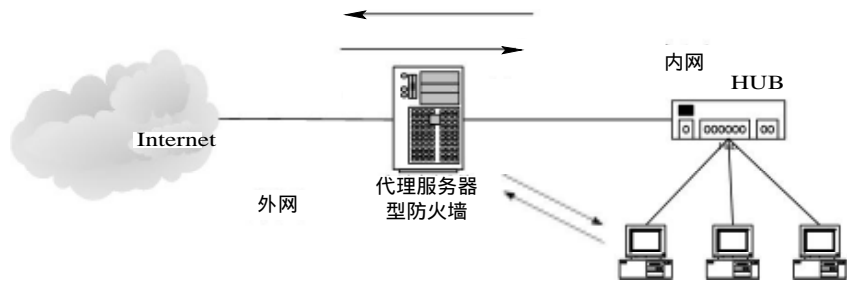


图 14-5 代理服务器型防火墙的一种常用结构

代理服务器(Proxy)实际上是运行在防火墙上的一种服务器程序，防火墙通常是具有两个网卡(或接口)的主机。服务器监听着客户的请求，如：申请浏览网页等。当内网的客户机请求与外网的真实服务器连接时，客户先连接代理服务器，然后再由代理服务器与真实的服务器进行连接，取得客户想要的信息，代理服务器再把信息返回给客户。此处的代理服务器是一个中间点，其角色和我们平常所说的中介十分相似，客户机和外界的主机没有发生直接联系，这是代理服务器型防火墙和包过滤型防火墙的本质差别所在。我们只要控制好这个中介所能代理的服务，就可以控制信息的进出，从而实现防火墙的目的。代理服务器型防火墙中数据的流通完全依赖于代理服务器所能代理的服务，因此透明性差。如果我们已经有了房地产的中介，而现在要进行的是外汇买卖，除了再找一个外汇买卖代理外，我们别无办法。使用代理服务器时，通常要把客户机的应用程序进行改动(即设置应用网关，例如：使用 WIN Proxy 代理服务器时，要在客户端的 IE 浏览器上进行代理服务器的有关配置)，如果应用程序没有代理版本，则常常造成服务器无法使用。另外，一个代理服务器软件通常只代理一种或几种服务，因此往往要在主机上安装多个代理服务器软件。一般来说，各种互联网服务的代理软件总是滞后于其标准版本，常常出现没有可靠的代理版本的情况。

使用代理服务器型防火墙的好处是可彻底地与真实网络隔离开来。一般说来，系统的安全性较好，缺点是透明性差，每一项服务都要有相应的代理软件。

14.1.2 包过滤型防火墙的两种策略

设计防火墙有两种基本的策略：一种是一开始就禁止所有的数据包通过，然后根据需

要打开相应的服务端口；另一种是一开始就开放所有的服务，然后对不可靠的服务端口进行封锁。原则上，第一种方法比较可靠，但是对于不熟悉 TCP/IP 的用户，可能会导致一些其他困扰。

14.2 用 ipchains 过滤数据包

14.2.1 什么是 ipchains

ipchains 是 Linux 的包过滤防火墙，是 Linux IPV4 防火墙的重写和 ipfwadm 的重写。ipchains 是内核支持的，实现速度较快。它分析一个数据包，检查源 IP 地址、目的 IP 地址、源端口号、目的端口号、协议的类型等信息。Linux 的 ipchains 命令可以建立规则，为进入、离开、穿越防火墙的数据提供可控制的限制，从而提供比较安全的防火墙功能。

14.2.2 使用 ipchains 的准备工作

在 Linux 中使用 ipchains 的前提满足以下五点：

(1) Linux 的内核是 Linux 2.1.102 以后的版本(红旗 Linux 3.0 的内核是 Linux 2.4.0 以后的，因此满足条件)。

(2) 若安装红旗 Linux 没有安装 ipchains，则安装之前要确保当前的内核支持 ipchains，请需要/usr/src/linux-2.4 目录下用命令“make menuconfig”对内核进行配置，在“Network options”选项下确保：

[*]Network packet filtering(replaces ipchains)

IP: Netfilter Configuration --->下的所有选项为“M”

重新编译内核，使用新的内核启动系统。

【提示】缺省时，红旗 Linux 3.0 内核支持 ipchains，无需重新编译。有关内核的重新编译，请参见有关资料。

(3) 打开 IP 转发功能，为了完成伪装等功能，必须打开系统的 IP 转发功能，使得 Linux 变成路由器。有两种方法：用命令“[root @redflag /root]#echo 1 >/proc/sys/net/ipv4/ip_forward”或修改/etc/sysconfig/network 文件，将 FORWARD_IPV4 的值改为“true”，重新启动系统。

(4) 安装 ipchains 软件包。如果没有安装软件包，请把红旗 Linux 光盘 mount 上，找到相应的目录，用“rpm -ivh ipchains-1.3.10-7.i386.rpm”进行安装。

(5) 用命令“service ipchains start”启动 ipchains 服务，也可以用“ntsysv”配置 ipchains 服务在系统启动时自动启动，目的是装入 ipchains 模块。这里也可以使用命令“modprobe ipchains”来装入 ipchains 模块。

14.2.3 ipchains 的工作流程

ipchains 在内核插入规则，使得系统根据这些规则进行数据包的过滤。规则是以链(ipchains)的形式组织的，ipchains 定义的四类规则链分别是：输入链(input)、输出链(output)、

转发链(forward)和用户自定义链(user defined chains), 其中前三类是永久规则链。输入链的规则作用于进入系统的数据; 输出链的规则作用于离开系统的数据; 转发链的规则作用于从系统的一个接口进来然后从另一个接口离开的数据(即穿越系统的数据)。永久规则链不能被删除; 用户规则链是由用户定义的, 可以被删除, 它可以由永久规则链中的规则来激活。

我们首先来讨论什么是数据的输入、输出和转发。它们都是从防火墙的角度来定义的, 如图 14-6(a)所示, 数据从主机 A 出发到达主机 B: 是先进入防火墙的 eth0 网卡, 然后穿越防火墙, 从网卡 eth1 转发出去, 最终到达主机 B。以上数据先后经过了输入、转发和输出的过程。再来看图 14-6(b), 数据从主机 A 出发, 目的地就是防火墙所在的服务器, 则数据从网卡 eth0 进入就到达目的地, 数据没有被转发到别处, 因此只有输入的过程。在图 14-6(c)中, 数据从服务器出发到达主机 A, 数据只有输出的过程。搞清楚数据的方向对于确定哪些规则链会被采用是至关重要的。

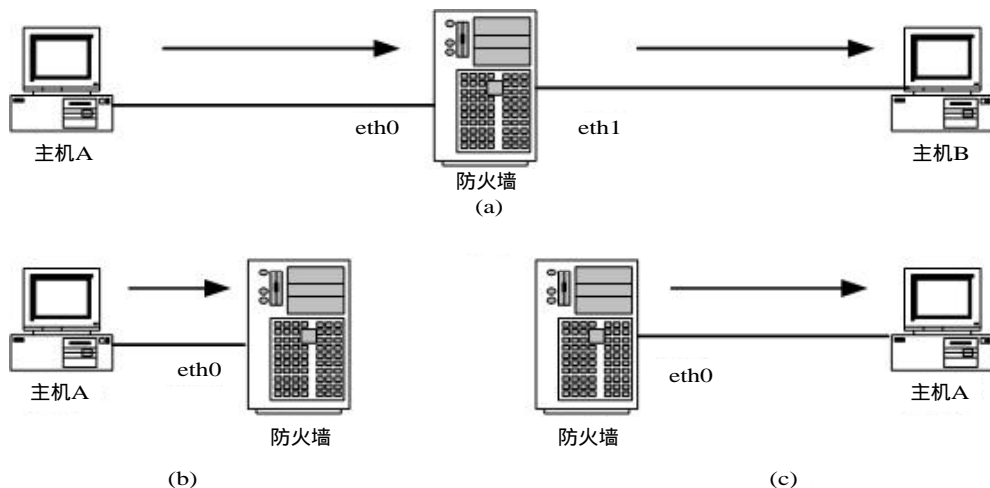


图 14-6 数据流动的方向

我们可以在数据输入、转发、输出的过程中定义检查规则和动作, 从而对数据进行控制。在图 14-6(a)中从主机 A 发出到达主机 B 的数据, 要先后经过 ipchains 中的 input、forward、output 规则链的检查, 而图 14-6(b)的数据只经过 input 规则链的检查, 图 14-6(c)的数据也只经过 output 规则链的检查。

一个规则链可以由一个或多个规则按顺序排列而成。所谓采用规则链过滤数据包, 就是数据包中的源 IP 地址、目的 IP 地址、源端口号、目的端口号等信息与规则链中的每一条规则按顺序进行匹配。如果有一条规则满足, 就执行规则中的动作(例如: 拒绝数据、接受数据、转到用户定义的规则链等); 如果没有任何规则满足, 则采用缺省动作(称为策略的默认规则)。图 14-7 描述了这个过程。

【提示】在 IP 数据包匹配一个规则链时, 是按顺序从上而下一个一个规则进行匹配的, 如果有一个规则匹配就立即采取规则中相应的动作, 然后停止以后规则的匹配, 因此规则的顺序是至关重要的, 通常把条件更具体的规则放在前面。

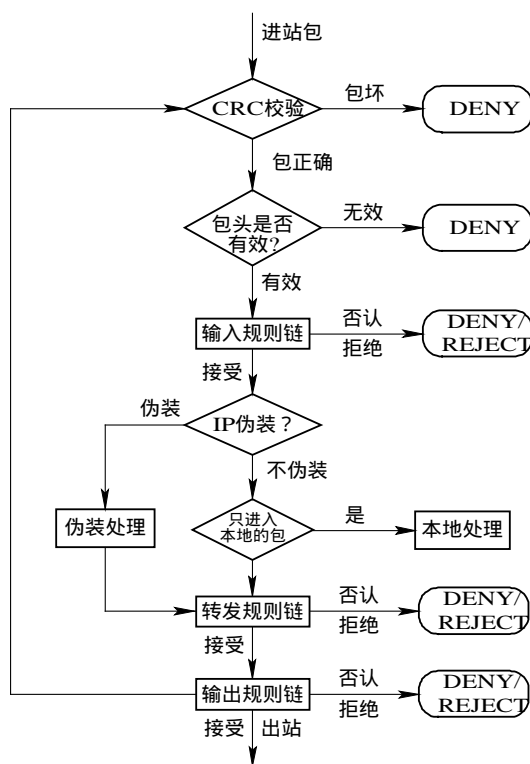


图 14-7 ipchains 事件流

【实例 14.1】

我们还是以 input 规则链为例介绍以上数据包的匹配方式。假如主机 192.168.0.15 上的 input 规则链定义了如下几条的规则：

#以下对所有的 ICMP 包全部拒绝。

Rules1: -icmp -j REJECT

#以下对所有的 TCP 协议数据包，转到用户自定义的规则链 userchain。

Rules2: -p tcp -j userchain

#以下拒绝所有的 UDP 协议数据包。

Rules3: -p udp -j DENY

其中用户自定义规则链 userchain 的规则集如下：

#以下接受来自 192.168.0.1 主机的数据。

Rules1: -s 192.168.0.1/32 -j ACCEPT

#拒绝所有的 TCP 协议数据。

Rules2: -p tcp -j DENY

现在我们从主机 192.168.0.1 上 Telnet 主机 192.168.0.15，这个数据是如何通过防火墙的呢？

(1) 从 192.168.0.1 主机发出的数据到达 192.168.0.15 主机时，因为数据要进入主机，所以要经过 input 规则链的检查。从规则链的第一条规则开始进行匹配，由于 Telnet 应用程序

使用的是 TCP 数据包，不是 ICMP 数据包，所以第一条规则不符，接着匹配下一条规则；

(2) 数据包符合第二条规则，所以根据定义的动作，跳转到用户自定义的规则链中，匹配 userchain 中的规则；

(3) 由于数据是从 192.168.0.1 主机出发的，因此数据包和 userchain 中的第一条规则匹配，执行规定的动作：接受数据，数据包顺利通过，交给主机 192.168.0.15 处理，因此 Telnet 能够成功；

(4) 由于已经有规则得到匹配，就停止以后规则的匹配，即用户自定义的 userchain 规则链的第二条规则以及 input 规则链中的第三条规则并不起作用。

我们用图 14-8 来表示以上过程。

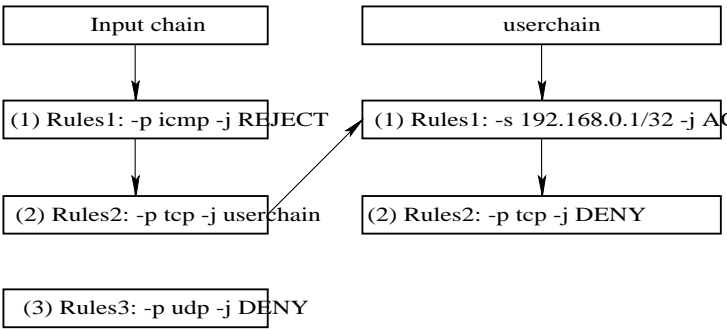


图 14-8 规则的匹配过程示例

14.2.4 ipchains 命令

Linux 中的防火墙具有强大的功能，而命令却只有一个 ipchains。这个命令实现防火墙的所有包过滤功能，因此该命令十分复杂。我们这里先把常用的选项、参数、动作等用表格列出，分别见表 14-1、表 14-2 和表 14-3，在下一节，我们将逐一用小的实例进行解析。

ipchains 的命令格式如下：

ipchains [选项] <命令参数> <目标动作>

表 14-1 ipchains 的命令选项

选项	含 义
-A	在规则链尾增加一条规则
-D	从规则链中删除规则
-R	替换规则链中的规则
-I	在给定的规则号前插入规则
-L	列出规则链中的规则，如未指明规则链，则列出所有规则链中的规则
-F	清除规则链中的规则，如未指明规则链，则清除所有规则链中的规则
-N	创建用户自己的规则链
-X	删除用户自定义的规则链
-P	改变永久规则链的默认规则(即默认动作)

表 14-2 ipchains 的命令参数

参 数	含 义
-h	显示帮助
-p [!] protocol	指明要检查的协议，可以列出/etc/protocols 列出的名字，如：TCP、UDP、ICMP、ALL，可以用“！”来否定它，表示去除协议
-s [!] address [/mask] [!] [port[:port]]	指明源地址。源地址可以是主机名(www.263.net)、IP 地址(192.168.0.1)、网络地址(192.168.0.0/24 或 192.168.0.0/255.255.255.0)、所有的地址(0.0.0.0/0)，对于 TCP 和 UDP 协议，还可以指定端口号和端口范围，6000 表示这一单一端口，6000:6009 包含 6000 ~ 6009 的 10 个端口，没有下限时默认是 0，没有上限默认是 65 535，端口也可以用名字表示，如：www
-d [!] address [/mask] [!] [port[:port]]	指明目的地址，用法和指明源地址一样
-b	设置双向模式。用该参数创建的一条规则分别作用于源、目的地址，相当于为源和目的各创建了一条规则
-i [!] name	指明单独的接口或某种类型的接口设置过滤规则，不指明时默认是所有的接口；可以指定一个不激活的接口，如 PPP0
-j target	指定规则中条件匹配时要执行的动作，target 可以是 ACCEPT、DENY、REJECT、MASQ 和用户的 chains
-v	显示出对用户的命令，ipchains 是如何响应的；详细的输出信息包括：接口地址、规则选项、包和字节计数器等

表 14-3 ipchains 的目标动作

动 作	含 义
ACCEPT	允许包通过
DENY	丢弃包，不向源主机发送 ICMP 信息
REJECT	丢弃包，向源主机发送 ICMP 信息，如果包为 ICMP 类型，与 DENY 一样不返回 ICMP 信息
MASQ	数据包的 IP 地址伪装成本机地址发送，回应的数据包自动地在被转发时解伪装
REDIRECT	数据被重定向到本地，即使它是要发往远端主机的，该规则只能用于输入链和用户链
RETURN	在用户链中，表示将包送到当前链中的下一条规则；在永久链中则表示将包送到默认规则
custom_chain	激活名为 custom_chain 的用户自定义规则链

14.2.5 ipchains 的使用

1. 规则链

ipchains 定义了三个默认的基本规则链：

(1) input：由网络接口进入本机。

(2) output：由网络接口输出。

(3) forward：在各网络接口之间转发，即从一个接口进从另一接口出。

用户也可以建立自己的规则链。建立自己的规则链使用“ipchains -N”指令，要删除用户规则链使用“ipchains -X”命令，基本规则链是不能删除的。

【实例 14.2】

```
[root @redflag /root]#ipchains -N mychain
```

建立自己的规则链，名为 mychain，随后就可以往规则链中添加规则。

```
[root @redflag /root]#ipchains -X mychain
```

删除自己建立的规则链 mychain，不指明删除规则链时删除所有用户自定义的规则链。

规则链由一条或多条规则组成，而每个规则由两个部分组成：匹配条件和动作(目标)。

往规则链添加规则使用“ipchains -A”命令；删除规则使用“ipchains -D”命令。

【实例 14.3】

```
[root @redflag /root]#ipchains -A input -p icmp -j DENY
```

往 input 规则链中添加一条规则“-p icmp -j DENY”，该规则的含义放在稍后介绍。

```
[root @redflag /root]#ipchains -D input -p icmp -j DENY
```

把规则从 input 规则链中删除，如果想删除规则链中全部规则，可使用“ipchains -F”命令。

```
[root @redflag /root]#ipchains -F input
```

删除 input 规则链中的全部规则，不指明规则链时删除所有规则链中的所有规则，注意规则链并不被删除。

可以使用“ipchains -L”命令显示规则链中的全部规则。

【实例 14.4】

```
[root @redflag /root]#ipchains -L input
```

Chain input (policy ACCEPT):

target	prot	opt	source	destination	ports
DENY	icmp	-----	anywhere	anywhere	any -> any

显示的是 input 规则链中全部规则，至于规则的意义稍后介绍。

我们在前面说过，规则链中规则的顺序是至关重要的。因此要有往规则链中指定位置插入规则的命令：“ipchains -I”。

【实例 14.5】

```
[root @redflag /root]#ipchains -I input 2 -p tcp -j DENY
```

在 input 规则链中的第二条规则前插入规则，如果没有指明插入的位置，则在规则链的第一条规则前插入。

2. 匹配条件

每个规则由匹配条件和动作(目标)组成，匹配条件就是对数据包的源 IP 地址、目的地址、源端口、目的端口、协议类型等进行判断，“-s”表示源，“-d”表示目的，“-p”表示协议类型。

【实例 14.6】

```
[root @redflag /root]#ipchains -A input -s 192.168.0.1 -d 192.168.0.15 -j ACCEPT
```

如果数据是从 192.168.0.1 主机出发，到达 192.168.0.15 则满足条件。

```
[root @redflag /root]#ipchains -A input -s 172.16.0.0/16 -d 202.96.134.0/24 -j ACCEPT
```

如果数据是从 172.16.0.0/255.255.0.0 子网上的主机发出，到达 202.96.134.0/255.255.255.0 子网上的主机则满足条件，这里采用网络而不是单一的主机。

【提示】如果在匹配中没有标出源地址或目的地址，那么默认就是全部主机，即 0.0.0.0/0。

【实例 14.7】

```
[root @redflag /root]#ipchains -A input -s 192.168.0.1 -p tcp -j ACCEPT
```

如果数据包是从 192.168.0.1 主机发出的 TCP 协议数据包，则满足匹配条件，目的地可以是任意。

【实例 14.8】

可以在源 IP 地址、目的 IP 地址后加上数字表示端口号，但要注意端口号必须和协议类型一起使用。

```
[root @redflag /root]#ipchains -A input -s 192.168.0.1 -d 192.168.0.15 23 -p tcp -j ACCEPT
```

匹配从 192.168.0.1 主机出发，到 192.168.0.15 的 Telnet 连接(23 端口)，也可以使用名字代替端口号，如下的命令和该命令等效：

```
[root @redflag /root]#ipchains -A input -s 192.168.0.1 -d 192.168.0.15 telnet -p tcp -j ACCEPT
```

端口号可以使用区间，方法是[最小端口号]:[最大端口号]。

```
[root @redflag /root]#ipchains -A input -s 192.168.0.1 1024 : 4096 -d 192.168.0.15 telnet -p tcp -j ACCEPT
```

表示从 192.168.0.1 出发所有端口在 1024 ~ 4096 之间的 Telnet 请求。上下界可以省略，“1024:”代表 1024 以上的所有端口。

3. 否定

“!”符号用于对匹配条件进行否定。

【实例 14.9】

```
[root @redflag /root]#ipchains -A input -s ! 192.168.0.1 -d 192.168.0.15 -j ACCEPT
```

如果数据到达 192.168.0.15，但不是从 192.168.0.1 主机出发，则满足条件。

4. 动作

规则中的动作，用于指明如果 IP 数据和规则中的条件匹配，则防火墙将要采用的动作，用“-j”指示要进行的动作，动作的类型见表 14-3。

【实例 14.10】

```
[root @redflag /root]#ipchains -A input -p icmp -j DENY
```

不允许(DENY)ICMP 数据包(ping 使用的)通过 input 规则链。这样实际上就是不允许别的主机 ping 本机了。

5. 规则链的默认动作

当数据开始匹配规则链的规则时，从上到下一个一个规则地匹配，如果没有一个规则匹配，防火墙将采取默认的动作。我们可以用“ipchains -P”命令更改规则链的默认动作。

【实例 14.11】

```
[root @redflag /root]#ipchains -P input DENY
```

表示把 input 规则链的默认动作改为 DENY。

```
[root @redflag /root]#ipchains -L input
```

Chain input (policy DENY):

target	prot	opt	source	destination	ports
DENY	icmp	-----	anywhere	anywhere	any -> any

在输出的第一行显示的是 input 规则链的默认动作 DENY，缺省时所有规则链的默认动作是 ACCEPT。

6. 接口

在指示规则链中的匹配条件时，可以使用“-i”指明接口，当它用于 input 和 output 规则链时其意义是显然的，如果用于 forward 规则链时，那么它给出的是转发的目标接口。

14.2.6 实例

1. 实例一

如图 14-9 所示，为了 Web 服务器的安全，只允许它为同一子网上的计算机提供 www 服务，同时为了一定程度的方便，也允许子网上的主机用 ping 命令测试它是否在线。

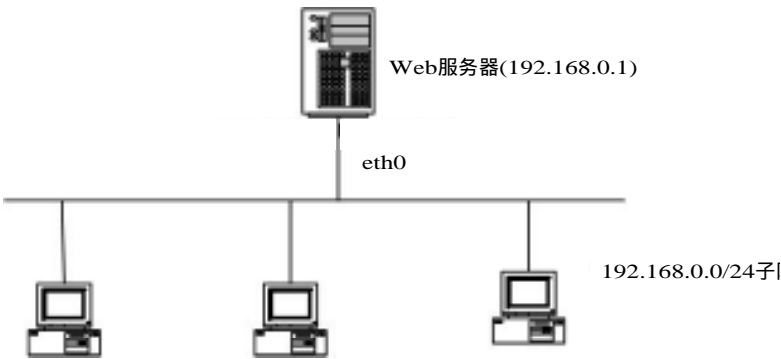


图 14-9 实例一示意图

在 192.168.0.1 主机上进行以下的配置：

```
[root @redflag /root]#ipchains -F (删除所有规则链的所有规则)
[root @redflag /root]#ipchains -A input -p tcp -s 192.168.0.0/24 -d 192.168.0.1 www -j ACCEPT
(允许来自本子网上主机的 WWW 服务请求)
[root @redflag /root]#ipchains -A input -p icmp -s 192.168.0.0/24 -d 192.168.0.1 -j ACCEPT
(允许来自本子网上主机的 ICMP 数据包进入)
[root @redflag /root]#ipchains -P input DENY
```

(其余的任何数据不允许进入本主机，我们这里采用的策略是先关闭所有的服务，再一一开放指定的服务)

```
[root @redflag /root]#ipchains -L (列出全部规则链进行检查)
```

```
Chain input (policy DENY):
```

target	prot	opt	source	destination	ports
ACCEPT	tcp	-----	192.168.0.0/24	192.168.0.1	any -> http
ACCEPT	icmp	-----	192.168.0.0/24	192.168.0.1	any -> any

```
Chain forward (policy ACCEPT):
```

```
Chain output (policy ACCEPT):
```

在以上输出中,第3和第4行是 input 规则链中的规则,每一列依次是动作、协议类型、选项、源 IP 地址、目的 IP 地址、源端口和目的端口。

这样子网上的主机只能 ping 通 Web 服务器和浏览服务器上的网页了,其他服务如 FTP、Telnet 等均不能成功。

2. 实例二

如图 14-10 所示,两个部门在各自的子网上,我们除了允许 192.168.0.0 /255.255.255.0 子网的主机可以访问 192.168.0.1 的邮件服务器外,两个子网间不允许有别的通信。可以在防火墙上进行:

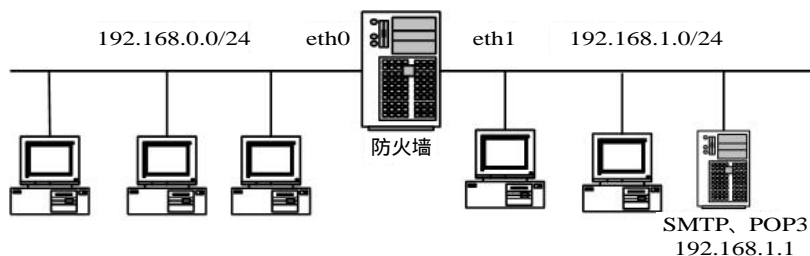


图 14-10 实例二示意图

```
[root @redflag /root]#ipchains -F (删除所有规则链的所有规则)
```

```
[root @redflag /root]#ipchains -X (删除所有用户自定义的规则链)
```

```
[root @redflag /root]#ipchains -A forward -p tcp -s 192.168.0.0/24 -d 192.168.1.1 smtp -i eth1 -j ACCEPT
```

(允许转发来自 192.168.0.0/24 子网主机,从 eth1 接口出去的 SMTP 数据包)

```
[root @redflag /root]#ipchains -A forward -p tcp -s 192.168.0.0/24 -d 192.168.1.1 pop3 -i eth1 -j ACCEPT
```

(允许转发来自 192.168.0.0/24 子网主机,从 eth1 接口出去的 POP3 数据包)

```
[root @redflag /root]#ipchains -A forward -p tcp -s 192.168.1.1 smtp -d 192.168.0.0/24 -i eth0 -j ACCEPT
```

(允许转发邮件服务器 192.168.1.1 应答 192.168.0.0/24 子网主机 SMTP 请求的数据报,从 eth0 接口出去)

```
[root @redflag /root]#ipchains -A forward -p tcp -s 192.168.1.1 pop3 -d 192.168.0.0/24 -i eth0 -j ACCEPT
```


(允许转发邮件服务器 192.168.1.1 应答 192.168.0.0/24 子网主机 POP3 请求的数据报, 从 eth0 接口出去)

```
[root @redflag /root]#ipchains -P forward DENY (拒绝其他数据包的转发)
```

【提示】 实现相同的防火墙功能常常可以有多种方案, 在本例中也可以把规则放在 input 和 output 规则链中实现, 请读者思考这时候应使用什么样的规则。

14.2.7 让建立的规则在系统启动时生效

有两种方法让 ipchains 中的规则在系统启动时生效。

1. 建立脚本并自动运行

可以直接用 ipchains 命令编写一个规则脚本, 并在系统启动时执行脚本。例如, 建立脚本名为 /etc/myfirewall, 脚本要是可执行的, 并在启动脚本 /etc/rc.d/rc.local 中加入语句:

```
if [-x /etc/myfirewall]; then /etc/myfirewall ; fi;
```

2. 用 ipchains 恢复

首先要测试通过防火墙的配置, 再用命令把内存中的规则转储在文件上。

```
[root @redflag /root]#/etc/rc.d/init.d/ipchains save
```

该命令会把所有的规则保存在文件 /etc/sysconfig/ipchains 中。系统启动时会自动执行 /etc/rc.d/init.d/ipchains, 而它会自动从 /etc/sysconfig/ipchains 文件中恢复已经保存的规则。

14.2.8 IP 伪装

防火墙的使用场合大多是如图 14-4 所示的模式, 为了减少占用互联网上真实的 IP 地址, 我们在内网(eth1 接口所接的网段)使用的是私有 IP 地址, 如 192.168.0.0/24, 而只有 eth0 接口才具有一个真正互联网上的 IP 地址, 如 202.199.148.120。由于私有 IP 地址不能和互联网上的主机直接通信, 因此要采用网络地址转换(NAT), 也称 IP 代理。处于私有网络的主机将自己的请求传递给代理机, 由代理机重写源 IP 地址和端口, 使得互联网上的主机接收到的数据包看起来像是从代理机发出来的, 代理机收到应答数据后, 将会把数据转给原来的内网上的主机, 这样从内网的主机看来就像是和外部主机直接通信。要使用 IP 伪装功能, 简单地使用 “-j MASQ” 动作即可。当然首先要打开 Linux 内核的 IP 转发功能:

```
[root @redflag /root]#echo 1 > /proc/sys/net/ipv4/ip_forward
```

然后设置防火墙对数据包进行伪装:

```
ipchains -A forward -s 192.168.0.0/24 -d ! 192.168.0.0/24 -j MASQ
```

最后还要在内网的客户机上把网关指向防火墙。这实际上做到了网络上的主机共享线路(例如 eth0 接 ADSL)上网了。

但是这里还存在一个问题, 在内网的主机如果使用 CuteFtp 从外网服务器下载文件没有问题, 而使用 Windows 2000 下的 ftp 命令行工具下载文件会出现如下错误:

```
C:\>ftp ftp.szptt.net.cn
Connected to ftp.szptt.net.cn.
220 (vsFTPd 1.1.3)
User (ftp.szptt.net.cn:(none)): anonymous
```

```
331 Please specify the password.
```

```
Password:
```

```
230 Login successful. Have fun.
```

```
ftp> ls
```

```
500 Illegal PORT command.
```

```
425 Use PORT or PASV first.
```

```
ftp>
```

这是一个特殊的问题，FTP 依赖两个连接，一个连接是用于控制的，它是客户端发起的；而另一个连接是用于传输数据的，通常是服务器端发起的，这种模式叫被动模式。可是服务器根本不知道关于内网(私有网络)的事情，因此连接不会成功。一个解决办法是和 CuteFtp 相同，强行让数据传输的这个连接也由客户端发起，叫主动模式；另一种方法就是采取补救方法，我们下一节介绍。

14.3 iptables

在 Linux 2.2 内核中采用了我们前面介绍的 ipchains 来控制内核包过滤规则。现在的内核已是 Linux 2.4 的了，该内核常常采用全新的内核包过滤管理工具——iptables。这个全新的内核包过滤工具将使用户更易于理解其工作原理，更容易被使用，当然也将具有更为强大的功能。iptables 只是一个管理内核包过滤的工具，实际上真正来执行这些过滤规则的是 netfilter (Linux 核心中一个通用架构)及其相关模块(如 iptables 模块和 nat 模块)。

14.3.1 iptables 的原理

netfilter 是 Linux 核心中一个通用架构，它提供了一系列的表(tables)，每个表由若干链(chains)组成，而每条链中可以由一条或数条规则(rule)组成。我们可以这样来理解，netfilter 是表的容器，表是链的容器，而链又是规则的容器。系统有 filter、nat、mangle 三个表，缺省的表为 filter，该表中包含了 input、forward 和 output 三个链，nat 表有 prerouting、postrouting、output 三个链；mangle 有 prerouting、output 两个链。这里链的概念和 ipchains 中链的概念是一样的，每一条链中可以有一条或数条规则。当一个数据包到达一个链时，系统就会从第一条规则开始检查，看是否符合该规则所定义的条件。如果满足，系统将根据该条规则所定义的方法处理该数据包；如果不满足则继续检查下一条规则。最后，如果该数据包不符合该链中任一条规则的话，系统就会根据该链预先定义的策略(Policy)来处理该数据包。数据包在 filter 表中的流程如图 14-11 所示。有数据包进入系统时，系统首先根据路由表决定将数据包发给哪一条链，则可能有以下三种情况：

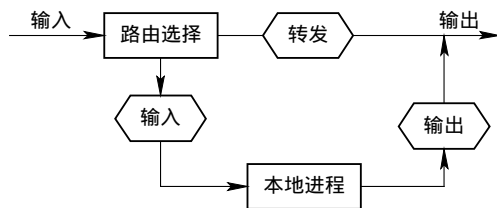


图 14-11 iptables 的工作流程

(1) 如果数据包的目的地址是本机，则系统将数据包送往 INPUT 链，如果通过规则检查，则该包被发给相应的本地进程处理；如果没通过规则检查，系统就会将这个包丢掉。

(2) 如果数据包的目的地址不是本机。也就是说，这个包将被转发，则系统将数据包送往 forward 链，如果通过规则检查，则该包被发给相应的本地进程处理；如果没通过规则检查，系统就会将这个包丢掉。

(3) 如果数据包是由本地系统进程产生的，则系统将其送往 output 链，如果通过规则检查，则该包被发给相应的本地进程处理；如果没通过规则检查，系统就会将这个包丢掉。

【提示】和 ipchains 的重要差别是，数据包不再经过 input、forward、output 全部三个规则链了，而是经过其中的一个链。

14.3.2 使用 iptables 准备工作

1. 系统需求

检查内核是否支持 netfilter，如果不支持要求重新编译。这些项目通常都是位于“Networking options”子项下的。以 Linux 2.4.17 内核为例，我们应该选中的项目有：

```
.....  
[*] Network packet filtering (replaces ipchains)  
.....
```

然后，在“IP: Netfilter Configuration ---->”选中所有的模块为“M”。

【提示】缺省时，红旗 Linux 3.0 支持 iptables，所以也无需重新编译。

2. 载入模块

ipchains 和 iptables 是相对立的，在使用 iptables 时就不能同时使用 ipchains。可以使用“rmmod ipchains”命令卸载 ipchains 模块。要使用 iptables，必须载入相关模块。可以使用以下命令载入相关模块：

```
[root@redflag /root]#modprobe ip_tables 或 insmod ip_tables
```

modprobe 命令会自动载入指定模块及其相关模块。iptable_filter 模块会在运行时自动载入。

14.3.3 iptables 命令

一个 iptables 命令基本上包含如下五部分：

- (1) 希望工作在哪个表上；
- (2) 希望使用该表的哪个链；
- (3) 进行的操作(插入、添加、删除和修改)；
- (4) 对特定规则的目标动作；
- (5) 匹配数据报条件。

基本的语法为：

```
iptables -t table -Operation chain -j target match(es)
```

例如，希望添加一个规则，允许所有从任何地方到本地 smtp 端口的连接：

```
iptables -t filter -A INPUT -p tcp --dport smtp -j ACCEPT
```

当然，还有其他的对规则进行操作的命令，如：清空链表，设置链缺省策略，添加一个用户自定义的链.....

基本操作：

-A：在链尾添加一条规则。

-I：插入规则。

-D：删除规则。

-R：替代一条规则。

-L：列出规则。

基本目标动作，适用于所有的链：

ACCEPT：接收该数据报。

DROP：丢弃该数据报。

QUEUE：排队该数据报到用户空间。

RETURN：返回到前面调用的链。

Customchain：用户自定义链。

基本匹配条件，适用于所有的链：

-p：指定协议(tcp/icmp/udp/...)。

-s：源地址(ip address/masklen)。

-d：目的地址(ip address/masklen)。

-i：数据报输入接口。

-o：数据报输出接口。

14.3.4 iptables 使用实例

假设某一单位租用 DDN 专线上网，网络拓扑如图 14-12 所示，eth0: 198.199.37.254，eth1: 198.168.80.254。以上的 IP 地址都是 Internet 上真实的 IP，故没有用到 IP 伪装。并且，我们假设在内部网中存在以下服务器：

WWW 服务器：198.168.80.11

FTP 服务器：198.168.80.12

E-mail 服务器：198.168.80.13

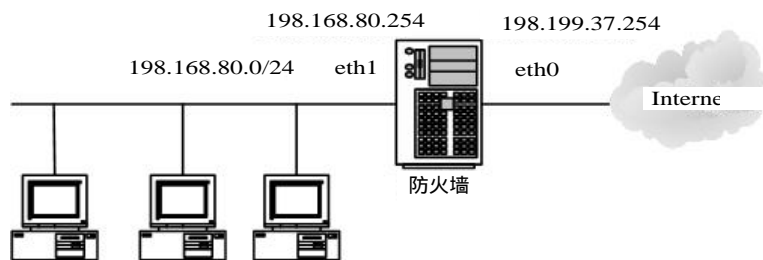


图 14-12 iptables 实例图示

下面我们将用 iptables 一步一步地来建立我们的包过滤型防火墙，需要说明的是，在这个例子中，我们主要是对内部的各种服务器提供保护。如下：

(刷新所有的链的规则)

```
iptables -F
```

(首先禁止转发任何包，然后再一步步设置允许通过的包)

```
iptables -P FORWARD DROP
```

(数据包是双向的，我们不仅仅要设置数据包出去的规则，还要设置数据包返回的规则，我们先建立针对来自 Internet 数据包的过滤规则，允许来自 Internet 的用户的 WWW、FTP 请求数据包到内网)

```
iptables -A FORWARD -p tcp -d 198.168.80.12 --dport www -i eth0 -j ACCEPT
```

```
iptables -A FORWARD -p tcp -d 198.168.80.12 --dport ftp -i eth0 -j ACCEPT
```

```
iptables -A FORWARD -p tcp -d 198.168.80.12 --dport ftp-data -i eth0 -j ACCEPT
```

(允许来自 Internet 的用户的 SMTP、POP3 请求数据包到内网)

```
iptables -A FORWARD -p tcp -d 198.168.80.13 --dport smtp -i eth0 -j ACCEPT
```

```
iptables -A FORWARD -p tcp -d 198.168.80.13 --dport pop3 -i eth0 -j ACCEPT
```

(接受来自整个内网的数据包过滤)

```
ACCEPTiptables -A FORWARD -s 198.168.80.0/24 -i eth1 -j ACCEPT
```

(对不管来自哪里的 ICMP 包都进行限制，允许每秒通过一个包，该限制触发的条件是 10 个包。)

```
iptables -A FORWARD -p icmp -m limit --limit 1/s --limit-burst 10 -j ACCEPT
```

以下是完整的脚本文件内容，希望通过这个实例使读者能对 iptables 的用法有所了解：

```
#!/bin/sh

/sbin/iptables -F

/sbin/iptables -A FORWARD -p tcp -d 198.168.80.11 --dport www -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d 198.168.80.12 --dport ftp -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d 198.168.80.12 --dport ftp-data -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d 198.168.80.13 --dport smtp -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d 198.168.80.13 --dport pop3 -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -s 198.168.80.0/24 -i eth1 -j ACCEPT
/sbin/iptables -A FORWARD -p icmp -m limit --limit 1/s --limit-burst 10 -j ACCEPT
```

14.3.5 iptables 与 ipchains 的区别

iptables 与 ipchains 的具体区别如下：

(1) iptables 的缺省链的名称从小写换成大写，并且意义不再相同。input 和 output 分别放置对目的地址是本机以及本机发出的数据包的过滤规则；

(2) “-I”选项现在只代表输入网络接口，输出网络接口则使用“-o”选项；

(3) TCP 和 UDP 端口现在需要用“--source-port”或“--sport”(或--destination-port/--dport)选项拼写出来并且必须置于“-p tcp”或“-p udp”选项之后，因为它们分别是载入 TCP 和 UDP 扩展的；

(4) 以前 TCP 的“-y”标志现在改为“--syn”，并且必须置于“-p tcp”之后；

(5) 原来的 DENY 目标最后改为了 DROP；

(6) MASQ 现在改为 MASQUERADE，并且使用不同的语法。REDIRECT 保留原名称，但也改变了所使用的语法。

14.3.6 iptables 中的 IP 伪装

用 iptables 来进行 IP 伪装也是相当的简单，网络结构见图 14-4 中，使用的命令如下：

```
[root @redflag /root]#iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

命令中“-t nat”说明对 NAT 表操作；“-A POSTROUTING”说明是添加 POSTROUTING 链中的规则；“-o eth0 -j MASQUERADE”表示对所有通过 eth0(ADSL 连接)接口向外发送的包进行 IP 伪装。但是仍然解决不了 14.2.8 所提到的 FTP 数据连接问题，我们要有补救措施：还要加载几个模块。如下：

```
insmod ip_tables
insmod ip_conntrack
insmod iptable_nat
insmod ipt_MASQUERADE
insmod ip_conntrack_ftp
insmod ip_nat_ftp
```

“ip_conntrack”是连接跟踪模块，是后面两个模块所需要的；“iptable_nat”是实现 NAT 功能的模块；ipt_MASQUERADE 则是实现 IP 伪装的。在红旗 Linux 3.0 中，前面 4 个模块会自动加载。当然路由功能也是一定要启用的：

```
[root @redflag /root]#echo 1 > /proc/sys/net/ipv4/ip_forward
```

这样即使内网中 FTP 客户端不主动发起数据连接，也能正常和外网的 FTP 服务器正常通信了。

本章小结

本章首先介绍了防火墙的两个类型：包过滤型和代理服务器型防火墙的工作原理及它们的差别；接着主要介绍了 ipchains 的工作流程和命令的各种参数选项以及命令的使用方法，为了便于理解，还通过三个实际例子对之进行了阐述，特别是 IP 伪装技术实际上就是 NAT 技术；最后介绍了替代 ipchains、当前较新的 iptables 的使用和解决 FTP 客户存在的问题的方法。

习 题

1. 防火墙可以分成_____和_____两大类。
2. 假设内部网络地址是 202.204.19.0/24，要拒绝来自外部网络的 ping 命令，请写出 ipchains 命令。
3. 防火墙进行 IP 伪装时，在把数据发送到目的主机前，要改写数据包的什么内容？
4. 在图 14-9 中，如果只允许服务器为本子网的主机提供 WWW 和 Telnet 服务，其余的服务一律拒绝，应如何用 ipchains 命令实现？

附录 A 习题参考答案

第 1 章答案：

1. D 2. A 3. A 4. D 5. B 6. D
7. ABCD 8. AD 9. AB 10. ABCD

第 2 章答案：

1. chap1.txt 文件为 longkey 用户所有，文件所属组为 longkey 用户组；文件的拥有者、所属组、其他用户对文件的权限分别为：读写、读写、读；文件大小为 16 字节；修改时间为：16 20A 24 22:23（乱码）。
2. 文件的拥有者、所属组、其他用户对文件的权限分别为：读写、读写、读。
3. chmod a+r test.txt
4. find 用于根据指定条件查找文件；grep 用于在文本文件中查找指定的字符串。
5. 软链接的文件有“->”字符。
6. 00 12 * * 1 echo hello|mail root
7. 执行 tar xzf soft.tar.gz 命令。
8. C
9. 依次执行命令：sync; sync; sync shutdown 或其他关机命令。
10. 用 ps 命令查出 user1 的登录进程号；再用 kill -9 命令终止进程。
11. rpm -q tcp_wrappers

第 3 章答案：

1. D 2. B 3. C 4. D 5. C

第 4 章答案：

1. 用户名为 test，密码在 shadow 文件中，UID 和 GID 都为 500，个人主目录为/home/test；Shell 为 Bash Shell。
2. 用户组名为 group1；GID 为 1000；密码在 shadow 文件中；无组成员。
3. 安装选项为：/dev/hda3 /mnt/disk1 ext2 defaults,usrquota,grpquota 1 2；还要重启系统后建立 aquota.user 和 aquota.grp 文件、执行命令 quotacheck -avug、执行命令 quotaon -avug。

第 5 章答案：

1. A 2. B 3. B 4. ABCD 5. ABCD 6. AB
7. ACD

第 6 章答案：

1. 不同文件系统安装在目录树中的不同目录下，共同构成了一个完整的目录，整个目录树包含了多个文件系统；Linux 系统屏蔽了访问不同文件的差别，对于使用者来说感觉不到它们之间的差别，所以目录树是无缝的。

2. 要经过分区、建立文件系统、安装文件系统三个步骤；分别使用 fdisk、mkfs、mount 命令。

3. 把分区/dev/hda1 安装在/目录下；分区类型为 reiserfs；安装选项为 defaults,notail；系统启动时第一个检查该文件系统。

第 7 章答案：

- | | | | | | |
|----------|--------|--------|-------|---------|--------|
| 1. A | 2. B | 3. B | 4. D | 5. C | 6. D |
| 7. D | 8. B | 9. A | 10. B | 11. BCD | 12. AB |
| 13. ABCD | 14. BC | 15. AB | | | |

第 8 章答案：

1. /etc/rc.d/init.d/nfs start 或 service nfs start
2. 把/home 目录导出，为所有用户只读，即等效为：*(ro)。
3. 自动安装：在/etc/fstab 文件中进行配置；手动安装：用 mount 命令安装。

第 9 章答案：

1. 动态 IP 方案优点是：可以减少网络管理员管理 IP 地址的工作量；提高 IP 地址的使用率，节约 IP 地址。缺点是：主机获得的 IP 地址不固定，对于提供网络服务的主机不适用；需要 DHCP 服务器。

2. dhcpd.conf 书写格式如下：

```
subnet 192.168.0.0 netmask 255.255.255.0 {  
    option domain-name-servers 192.168.0.250;  
    option router 192.168.0.251;  
    range 192.168.0.1 192.168.0.199;  
}
```

3. 当 DHCP 客户机所在的局域网无 DHCP 服务器时，DHCP 客户机将无法获得 IP 地址。DHCP 代理可以从另一网络上的 DHCP 服务器上获得 IP 地址，然后送给 DHCP 客户。

4. 31 536 000 秒

第 10 章答案：

- | | | | | | |
|------|------|------|-------|-------|-------|
| 1. A | 2. C | 3. D | 4. B | 5. A | 6. C |
| 7. B | 8. D | 9. B | 10. A | 11. A | 12. A |

第 11 章答案：

- | | | | | | |
|------|------|--------|----------|------|------|
| 1. C | 2. A | 3. A | 4. C | 5. A | 6. A |
| 7. A | 8. B | 9. ABC | 10. ABCD | | |

第 12 章答案：

- | | | | | | |
|------|------|------|-------|-------|------|
| 1. C | 2. A | 3. D | 4. D | 5. B | 6. C |
| 7. A | 8. B | 9. D | 10. B | 11. B | |

第 13 章答案：

- | | | | | | |
|------|------|------|-------|------|------|
| 1. A | 2. C | 3. C | 4. A | 5. A | 6. B |
| 7. A | 8. D | 9. B | 10. B | | |

第 14 章答案：

1. 包过滤型和代理服务器型
2. `ipchains -A input -p icmp -d 202.204.19.0/24 -j DENY`
3. 源 IP 地址
4. `ipchains -F`
`ipchains -A input -p tcp -s 192.168.0.0/24 -d 192.168.0.1 www -j ACCEPT`
`ipchains -A input -p tcp -s 192.168.0.0/24 -d 192.168.0.1 telnet -j ACCEPT`
`ipchains -P input DENY`

附录 B 命令说明

Linux 中常用命令及说明如下：

名 称	说 明
&	后台执行命令
>	将结果输出到指定的文件
>>	将结果附加到指定的文件
	管道命令
adduser	创建新账号
adsl-start	ADSL 上网
adsl-status	查询 ADSL 状态
adsl-stop	将 ADSL 断线
alias	设置、显示命令的别名
at	在设定的时间执行作业
atq	列出用户排在队列中的作业
atrm	删除队列中的作业
badblocks	检查磁盘坏块
bc	Linux 下的计算器
bg	后台执行命令
cal	显示年历或月历
cat	显示或连接文件
cd	切换当前所在的目录
chgrp	改变文件或目录的所属组
chmod	改变文件或目录的权限
chown	改变文件或目录的拥有者
chsh	更改登录系统所使用的 Shell
clear	清屏
clock	显示及调整时间
cmp	比较两个文件内容的不同处
cnfn	修改用户的个人信息
compress	压缩.Z 的命令
cp	复制文件或目录

createdb	创建数据库
createuser	创建数据库服务器用户
crond	任务高度的常驻命令
crontab	设置任务调度工作
csch	启动 C Shell
date	显示当前的日期与时间
dd	转换或复制文件
df	查阅分区大小、使用率与剩余空间
diff	比较两个文件内容的不同处
dircolors	设置 ls 命令在显示时所用的颜色
dmesg	显示开机信息
du	查阅每个目录占用的磁盘空间
echo	显示信息
edquota	编辑磁盘空间限制
exit	退出当前的 Shell
export	输出环境变量
exportfs	导出文件系统
fdisk	磁盘分区
fg	将程序或命令切换至当前台执行
file	显示文件类型
find	查找文件或目录
finger	查看用户信息
fsck	维护文件系统
ftp	传输文件
ftpcount	查看有多少人登录 FTP 服务器
ftpsht	关闭 FTP 服务器
ftpwho	查看登录 FTP 服务器的人数
gcc	编译 C 程序
gpasswd	管理组内的用户
grep	查找文件中符合条件的字串
groupadd	增加组
groupdel	删除组
grpconv	打开组投影密码功能
grpunconv	关闭组投影密码功能
gunzip	解压缩.gz 文件的命令
gzip	压缩.gz 文件的命令

halt	关机
head	显示文件的前几行
help	用于查看 Linux 内置命令的帮助
history	列出最近使用过的命令
host	查找主机 IP 地址
hostname	显示或设置系统的主机名
httpd	Apache 服务器启动程序
ifconfig	查询网络接口
init	改变系统的运行等级
insmod	载入内核模块
jobs	显示正在后台执行的工作
kill	删除运行中的程序
killall	根据进程名来发送信号
last	查看曾登录此系统的用户
ldd	查看一个程序使用哪些共享库
less	显示文件内容
lilo	内核载入及 LILO 启动管理程序
ln	创建文件或目录链接
locate	查找绝对路径中包含指定字符串的文件
login	登录系统
logout	注销系统
lpq	显示打印任务
lpr	打印文件
lprm	删除打印任务
ls	列出目录内容
lsmod	列出当前载入的模块
mail	发送邮件
make	编译内核模块或程序
man	在线查询命令
mcopv	复制文件
mesg	控制他人往自己的终端发送消息的能力
mkbootdisk	创建引导盘
mkdir	创建目录
mke2fs	创建 ext2 或 ext3 文件系统
mkfs	创建文件系统
mkraid	初始化磁盘阵列

mkreiserfs	创建 ReiserFS 文件系统
mkswap	创建交换分区
modprobe	处理可载入模块
more	使文件一页一页显示
mount	挂载文件系统
mtr	查看经过路由表
mv	移动或重命名现有的文件或目录
netstat	监视网络状况
nice	设置程序运行的等级
nohup	后台继续运行程序
ntsysv	设置系统服务程序
passwd	设置密码
pdadduser	增加大量用户
pico	Pico 文本编辑器
ping	检测主机网络状况
poweroff	立即停止系统，并且关闭电源
printtool	打印机配置程序
ps	报告程序运行的情况
pwconv	打开投影密码功能
pwd	显示工作目录
pwunconv	关闭投影密码功能
quota	显示磁盘使用的空间与限制
quotacheck	检查磁盘空间限制
quotaoff	停用磁盘空间限制
quotaon	启用磁盘空间限制
raidstart	启动磁盘阵列
raidstop	停止运行中的磁盘阵列
reboot	重新开机
renice	重新设置程序运行的等级
repquota	检查磁盘空间限制的状态
rlogin	远程登录
rm	删除文件或目录
rmdir	删除目录
rpm	管理软件包
sendmail	Sendmail 邮件程序
set	查询与设置系统环境变量

shutdown	系统关机命令
smbclient	Samba 服务器查看程序
smbmount	挂载 Samba 共享的目录
smbpasswd	设置 Samba 服务器的用户密码
smbstatus	检查 Samba 服务器资源的使用情形
sndconfig	声卡配置程序
sort	将文件中的内容排序输出
swapoff	关闭交换文件或分区
swapon	激活交换文件或分区
startx	启动 X Window
stat	显示文件或目录的各种信息
su	改变用户的 ID
tail	显示文件的末尾几行
tar	打包文件
telnet	过程登录
test	测试真假值
testparm	检查 Samba 服务器的配置文件
testprns	检查打印机配置文件
top	监视系统资源
touch	改变文件或目录时间，也可产生新文件
traceroute	显示本机到达目标主机的路由路径
tree	以树的形式显示指定目录下的内容
umask	改变默认权限掩码
umount	卸载文件系统
unalias	取消命令的别名设置
uname	显示系统信息
uncompress	解压缩.Z 的命令
uniq	比较相邻的行，显示不重复的行
unzip	解压缩 zip 文件
useradd	创建新账号
userdel	删除账号
usermod	改变用户的属性
vi,vim	Vim 文本编辑器
w	观察登录系统者的举动
wall	向任何用户终端发送字符消息
wc	统计一个文件字节数、单词数、行数

whereis	寻找文件
which	寻找文件
who	显示当前登录系统的用户信息
whoami	显示用户名
write	向用户发送字符消息
Xconfigurator	设置 X Window
Xhost	管理访问 X Server 的主机

参 考 文 献

1. 中科红旗软件技术有限公司. 红旗 Linux 用户基础教程. 北京: 电子工业出版社, 2001
2. 中科红旗软件技术有限公司. 红旗 Linux 网络管理教程. 北京: 电子工业出版社, 2001
3. 中科红旗软件技术有限公司. 红旗 Linux 系统管理教程. 北京: 电子工业出版社, 2001