

LINUX

- ◆ 内容全面：详细讲解412个常用指令和760个执行范例
- ◆ 查询方便：提供功能索引和字母索引
- ◆ 适用面广：适用于Red hat、Fedora Core、Debian、SuSE、Ubuntu、Gentoo等版本



指令 速查手册

吴恒奎 编著



LINUX



人民邮电出版社
POSTS & TELECOM PRESS

LINUX

指令 速查手册

吴恒奎 编著



LINUX

人民邮电出版社
北京

图书在版编目（CIP）数据

Linux 指令速查手册 / 吴恒奎编著. —北京：人民邮电出版社，2007.12
ISBN 978-7-115-16895-5

I . L… II . 吴… III . Linux 操作系统—技术手册
IV . TP316.89-62

中国版本图书馆 CIP 数据核字（2007）第 148225 号

内 容 提 要

Linux 是一款开源的操作系统，得到了广大开发者的青睐。掌握 Linux 系统的指令及其用法是学习 Linux 系统的基础。本书详细地介绍了常用 Linux 指令的功能、语法、参数说明以及具体的操作实例。全书共讲解指令 412 个，共有实例 760 个。

本书中所讲指令按照字母顺序和功能顺序进行索引，读者既可以按照字母顺序进行查找，也可以按照功能顺序进行学习。本书不仅适合于 Linux 初学者学习使用，而且可以作为 Linux 系统管理员的参考手册。

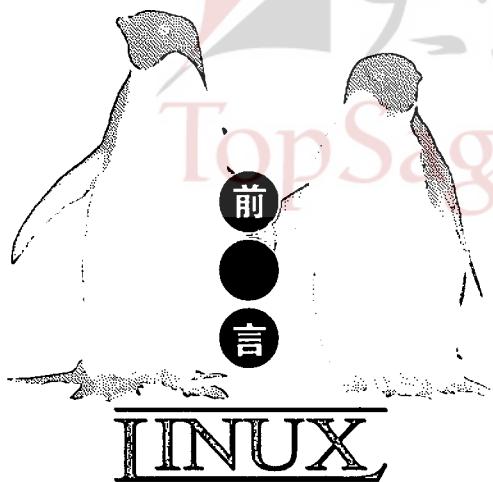
Linux 指令速查手册

-
- ◆ 编 著 吴恒奎
 - 责任编辑 黄 磊
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
 - 邮编 100061 电子函件 315@ptpress.com.cn
 - 网址 <http://www.ptpress.com.cn>
 - 北京顺义振华印刷厂印刷
 - 新华书店总店北京发行所经销
 - ◆ 开本：880×1230 1/32
 - 印张：22
 - 字数：656 千字 2007 年 12 月第 1 版
 - 印数：1—5 000 册 2007 年 12 月北京第 1 次印刷

ISBN 978-7-115-16895-5/TP

定价：45.00 元

读者服务热线：(010)67132692 印装质量热线：(010)67129223



LINUX

Linux 是开源的类 UNIX 操作系统，世界上任何一个程序员都可以参与到 Linux 的设计和实现中来，Linux 自身具有的特点吸引了无数的 Linux 开发者。首先，它是开源的，开发者可以免费获得 Linux 操作系统的源代码；其次，它具有 UNIX 的所有功能；再次，用户或者开发者可以根据自己的需要选择模块，构建自己的操作系统，还可以加载用户自己设计的模块，它是一种为用户量身定做的操作系统。

对于刚开始接触 Linux 的用户，特别是用惯了 Windows 系统图形界面的用户来说，一本 Linux 指令查询手册是必需的。笔者编写了本书，希望对 Linux 爱好者和使用者有所帮助。

本书特点

1. 命令详尽

本书详细讲解了 412 个 Linux 指令，覆盖了 Linux 应用中所有的常用命令。

2. 支持各种环境

本书支持多种 Linux 操作系统，包括 Red hat、Fedora Core、Debian、SuSE、Ubuntu、Gentoo 等，但是在不同发行版本所得的输出结果可能与本



书的例子不完全相同。

3. 实例丰富

书中的指令大都配有详细的实例说明，全书共有实例 760 个，读者可以根据实例说明进行操作，加深对指令的理解。

4. 索引方便

本书提供的命令按照两种方式进行索引，一是按照命令的首字母顺序进行索引；二是按照命令的功能进行索引，方便用户查找使用。

使用说明

本书命令的介绍分为 6 个部分：指令名和功能、语法、功能说明、参数说明、执行范例、相关指令。

指令名和功能：简要、概括地介绍指令的名称和功能。

语法：介绍使用此功能的具体语法要求。

功能说明：详细介绍指令的功能。

参数说明：详细介绍当前指令可能会使用的参数以及相应参数对应的具体功能。参数又分为必要参数和选择性参数，必要参数是实现某项功能必须要使用的参数，选择性参数是辅助性的参数，比如“-help”表示帮助信息。

常用的参数如下。

- -help: 帮助信息。
- -version: 版本信息。
- -v: 运行时显示详细的处理信息。
- -i: 交互模式。
- -d: 调试模式。
- -q: 运行时不显示处理信息。

执行范例：当前命令的执行实例详细地介绍指令的使用方法。执行范例均是在命令行下的操作，在每一步的操作之后均有详细的解释。比如：

- ```
root@localhost:~# useradd kk //添加用户 kk
➤ root@localhost:~#: 执行环境，显示当前的用户和目录。用户为 root，目录为根目录。
➤ useradd kk: 指令的操作，也就是用户输入的命令。
```

➤ //添加用户 kk：对输入命令的功能的解释，介绍操作目的。

**相关指令：**介绍与当前指令功能类似或者存在调用关系的指令。

## 适合的读者

- Linux 初学者；
- Linux 系统管理员；
- Linux 爱好者；
- 大中专院校的学生。

本书由吴恒奎组织编写，同时参与编写、资料整理和代码编写的有刘丹、刘冠军、罗思红、孙飞、王朋章、王石磊、王新平、文奇、吴琪、席国庆、谢超文、臧勇、张国强、张家春、郭玉敏、贺道权、胡斯登、江成海、姜海峰、李峥、利建昌、栗菊民、刘波等，在此一并表示感谢。

由于时间仓促，加之水平有限，书中不足之处在所难免，敬请读者批评指正。本书责任编辑的联系方式是 [huangyan@ptpress.com.cn](mailto:huangyan@ptpress.com.cn)，欢迎来信交流。

编者

2007 年 10 月



大智网是全球领先的中文知识门户，为您提供最全面、最权威的行业知识、管理经验、专业技能、实用工具等。大智网拥有强大的搜索功能，能够快速找到您需要的信息。同时，我们还提供丰富的互动社区，让您能够与众多行业专家和同行交流、分享经验。无论您是企业决策者、管理者、还是专业人士，大智网都是您不可或缺的知识伙伴。

大智网致力于打造一个开放、共享、创新的知识平台。我们相信，通过不断积累和分享知识，可以帮助企业和个人实现更大的成功。如果您有任何问题或建议，请随时联系我们，我们将竭诚为您服务。

大智网期待您的加入，共同构建一个更加美好的知识世界！

大智网，让知识更智慧！

大智网，让知识更智慧！

大智网，让知识更智慧！

## 目 录

第1章 系统管理..... 1

|                                      |    |
|--------------------------------------|----|
| 1.1 adduser 指令: 增加一个系统用户.....        | 2  |
| 1.2 apm 指令: 高级电源管理.....              | 3  |
| 1.3 apmsleep 指令: APM 进入备用或者休眠状态..... | 4  |
| 1.4 apropos 指令: 查找使用手册的名字和相关描述.....  | 5  |
| 1.5 arch 指令: 输出主机的体系结构.....          | 8  |
| 1.6 batch 指令: 执行批处理指令.....           | 8  |
| 1.7 bg 指令: 将程序放在后台执行.....            | 9  |
| 1.8 cd 指令: 切换目录.....                 | 9  |
| 1.9 chfn 指令: 设置 finger 信息.....       | 10 |
| 1.10 chsh 指令: 改变登录系统时的 shell.....    | 12 |
| 1.11 clear 指令: 清除终端屏幕.....           | 13 |
| 1.12 date 指令: 显示或设置系统时间.....         | 13 |
| 1.13 echo 指令: 显示文本行.....             | 16 |
| 1.14 exec 指令: 执行完指令后, 交出控制权.....     | 18 |
| 1.15 exit 指令: shell 退出.....          | 18 |
| 1.16 fc 指令: 修改或者执行指令.....            | 19 |
| 1.17 fg 指令: 将后台任务拉到前台执行.....         | 21 |
| 1.18 fgconsole 指令: 打印虚拟终端的数目.....    | 22 |
| 1.19 finger 指令: 查找并显示用户信息.....       | 22 |
| 1.20 free 指令: 显示内存信息.....            | 24 |
| 1.21 fuser 指令: 用文件或者套接口表示进程.....     | 25 |
| 1.22 fwhois 指令: 显示用户的信息.....         | 27 |
| 1.23 gcov 指令: coverage 测试工具.....     | 28 |

## 目录

|      |                                |    |
|------|--------------------------------|----|
| 1.24 | gdialog 指令：从 shell 显示文本信息      | 30 |
| 1.25 | gitps 指令：显示程序情况                | 32 |
| 1.26 | groupadd 指令：创建一个新的群组           | 33 |
| 1.27 | groupdel 指令：删除一个群组             | 33 |
| 1.28 | groupmod 指令：改变系统群组的属性          | 34 |
| 1.29 | halt 指令：关闭系统                   | 35 |
| 1.30 | help 指令：显示 shell 的内建指令的帮助信息    | 36 |
| 1.31 | history 指令：显示历史指令              | 38 |
| 1.32 | htpasswd 指令：创建和更新用户的认证文件       | 40 |
| 1.33 | id 指令：显示用户的 ID 以及所属群组的 ID      | 42 |
| 1.34 | info 指令：读取目录信息                 | 43 |
| 1.35 | init 指令：开关机设置                  | 46 |
| 1.36 | ipcs 指令：显示进程间通信的信息             | 47 |
| 1.37 | ipcrm 指令：删除消息队列、旗语设置或者共享内存的 ID | 49 |
| 1.38 | jobs 指令：列出所有后台程序               | 50 |
| 1.39 | kill 指令：杀死执行中的进程               | 51 |
| 1.40 | killall 指令：杀死同名的所有进程           | 52 |
| 1.41 | last 指令：列出目前与过去登录系统的用户相关信息     | 55 |
| 1.42 | lastb 指令：列出登录系统失败的用户相关信息       | 57 |
| 1.43 | login 指令：登录系统                  | 58 |
| 1.44 | logname 指令：显示登录账号的信息           | 59 |
| 1.45 | logrotate 指令：处理 log 文件         | 59 |
| 1.46 | logout 指令：退出系统                 | 60 |
| 1.47 | lsmod 指令：显示 Linux 内核的模块信息      | 61 |
| 1.48 | man 指令：格式化和显示在线手册              | 62 |
| 1.49 | manpath 指令：设置 man 手册的查询路径      | 65 |
| 1.50 | mkfontdir 指令：创建字体文件目录          | 66 |
| 1.51 | mount 指令：挂载文件系统                | 67 |
| 1.52 | mpost 指令：系统的绘画工具               | 69 |
| 1.53 | msgcat 指令：合并消息目录               | 70 |

|                                      |     |
|--------------------------------------|-----|
| 1.54 msgcomm 指令：匹配两个消息目录.....        | 71  |
| 1.55 msgen 指令：创建英语消息目录.....          | 73  |
| 1.56 msginit 指令：初始化消息目录.....         | 74  |
| 1.57 newgrp 指令：登录另一个群组.....          | 75  |
| 1.58 nohup 指令：退出系统继续执行指令.....        | 76  |
| 1.59 ntsysv 指令：设置系统的各种服务.....        | 76  |
| 1.60 open 指令：开启虚拟终端.....             | 77  |
| 1.61 pgrep 指令：基于名字和其他属性的查找或信号处理..... | 79  |
| 1.62 pidof 指令：找到运行程序的进程 ID.....      | 85  |
| 1.63 pkill 指令：杀死进程.....              | 86  |
| 1.64 pmap 指令：显示程序的内存信息.....          | 89  |
| 1.65 procinfo 指令：显示系统状态.....         | 94  |
| 1.66 ps 指令：报告程序状况.....               | 95  |
| 1.67 pstree 指令：以树状图显示程序.....         | 99  |
| 1.68 pwck 指令：检查密码文件.....             | 104 |
| 1.69 pwd 指令：显示工作目录.....              | 105 |
| 1.70 reboot 指令：重新启动.....             | 106 |
| 1.71 renice 指令：调整优先级.....            | 106 |
| 1.72 rlogin 指令：远端登录.....             | 108 |
| 1.73 rmmod 指令：删除模块.....              | 111 |
| 1.74 rsh 指令：远端登录的 shell.....         | 115 |
| 1.75 rwho 指令：查看系统用户.....             | 118 |
| 1.76 screen 指令：多重视窗管理程序.....         | 119 |
| 1.77 shutdown 指令：系统关机指令.....         | 122 |
| 1.78 sleep 指令：休眠.....                | 123 |
| 1.79 su 指令：变更用户身份.....               | 124 |
| 1.80 sudo 指令：以其他身份来执行指令.....         | 125 |
| 1.81 suspend 指令：暂停执行 shell.....      | 127 |
| 1.82 swatch 指令：系统监控程序.....           | 128 |
| 1.83 symlinks 指令：维护符号链接的工具程序.....    | 129 |

## 目录

|                             |            |
|-----------------------------|------------|
| 1.84 tload 指令：显示系统负载        | 129        |
| 1.85 top 指令：显示进程信息          | 130        |
| 1.86 uname 指令：显示系统信息        | 137        |
| 1.87 useradd 指令：建立用户账号      | 139        |
| 1.88 userconf 指令：用户账号设置     | 140        |
| 1.89 usermod 指令：修改用户账号      | 141        |
| 1.90 userdel 指令：删除用户账号      | 142        |
| 1.91 users 指令：显示用户          | 143        |
| 1.92 vlock 指令：锁定终端          | 143        |
| 1.93 w 指令：显示登录系统的用户信息       | 145        |
| 1.94 wait 指令：等待程序返回状态       | 146        |
| 1.95 watch 指令：将结果输出到标准输出设备  | 147        |
| 1.96 whereis 指令：查找文件        | 149        |
| 1.97 which 指令：查找文件          | 150        |
| 1.98 who 指令：显示系统用户信息        | 151        |
| 1.99 whoami 指令：显示用户名        | 153        |
| 1.100 whois 指令：显示指定用户信息     | 154        |
| 1.101 &指令：将任务放到后台执行         | 154        |
| <b>第2章 系统设置</b>             | <b>157</b> |
| 2.1 alias 指令：设置指令的别名        | 158        |
| 2.2 apmd 指令：高级电源管理          | 158        |
| 2.3 at 指令：系统将执行的任务排队        | 159        |
| 2.4 atd 指令：执行已经排队的任务        | 162        |
| 2.5 atq 指令：检查排队的任务          | 163        |
| 2.6 atrm 指令：删除已经排队的任务       | 164        |
| 2.7 atrun 指令：执行已排队的任务       | 164        |
| 2.8 aumix 指令：设置音效设备         | 165        |
| 2.9 authconfig 指令：配置系统的认证信息 | 167        |
| 2.10 bind 指令：显示或者设置按键组合     | 168        |

|      |                                   |     |
|------|-----------------------------------|-----|
| 2.11 | chkconfig 指令：设置系统的应用程序.....       | 171 |
| 2.12 | chroot 指令：改变根目录.....              | 175 |
| 2.13 | clock 指令：系统 RTC 时间设置.....         | 176 |
| 2.14 | crontab 指令：设置计时器.....             | 177 |
| 2.15 | declare 指令：显示或者设定 shell 变量.....   | 178 |
| 2.16 | depmod 指令：模块关系.....               | 180 |
| 2.17 | dircolors 指令：ls 指令对应的显示颜色设置.....  | 181 |
| 2.18 | dmesg 指令：显示开机信息.....              | 182 |
| 2.19 | enable 指令：可用的 shell 内置指令.....     | 184 |
| 2.20 | eval 指令：连接多个指令.....               | 186 |
| 2.21 | export 指令：设置或者显示环境变量.....         | 187 |
| 2.22 | false 指令：不做任何事情，表示失败.....         | 191 |
| 2.23 | fbset 指令：设置帧缓冲区.....              | 192 |
| 2.24 | hash 指令：显示和清除哈希表.....             | 193 |
| 2.25 | hostid 指令：打印出当前主机的标识.....         | 194 |
| 2.26 | hostname 指令：显示或者设置当前系统的主机名.....   | 195 |
| 2.27 | hwclock 指令：显示与设定硬件时钟（rtc）.....    | 197 |
| 2.28 | insmod 指令：载入模块.....               | 199 |
| 2.29 | isosize 指令：显示 iso9660 文件系统信息..... | 200 |
| 2.30 | kbdconfig 指令：设置键盘类型.....          | 200 |
| 2.31 | ldconfig 指令：配置动态链接绑定.....         | 201 |
| 2.32 | ldd 指令：打印共享库文件的相互依赖关系.....        | 202 |
| 2.33 | lilo 指令：引导安装程序.....               | 203 |
| 2.34 | liloconfig 指令：设置程序的载入.....        | 204 |
| 2.35 | losetup 指令：设置循环.....              | 205 |
| 2.36 | mv 指令：监视鼠标情况.....                 | 206 |
| 2.37 | minfo 指令：显示 MS-DOS 文件系统的各项参数..... | 207 |
| 2.38 | mkkickstart 指令：建立安装的组态文件.....     | 207 |
| 2.39 | mkraid 指令：初始化/升级 RAID 设备阵列.....   | 208 |
| 2.40 | modinfo 指令：显示内核信息 .....           | 209 |



## 目录

|                                                   |            |
|---------------------------------------------------|------------|
| 2.41 modprobe 指令: 自动处理可载入模块.....                  | 210        |
| 2.42 mouseconfig 指令: 设置鼠标相关参数.....                | 211        |
| 2.43 nice 指令: 设置优先权.....                          | 213        |
| 2.44 passwd 指令: 设置密码.....                         | 214        |
| 2.45 pwconv 指令: 开启用户的投影密码.....                    | 215        |
| 2.46 pwunconv 指令: 关闭用户的投影密码.....                  | 216        |
| 2.47 resize 指令: 设置终端机视窗的大小.....                   | 216        |
| 2.48 rpm 指令: 管理 RPM 包.....                        | 217        |
| 2.49 runlevel 指令: 显示执行等级.....                     | 219        |
| 2.50 set 指令: 设置 shell .....                       | 220        |
| 2.51 setconsole 指令: 设置系统终端.....                   | 222        |
| 2.52 setenv 指令: 查询或显示环境变量.....                    | 223        |
| 2.53 setserial 指令: 设置或显示串口的相关信息.....              | 224        |
| 2.54 setup 指令: 设置公用程序.....                        | 224        |
| 2.55 sliplogin 指令: 将终端机之间的连接变为 sliplogin 连接 ..... | 228        |
| 2.56 swapoff 指令: 关闭系统交换分区.....                    | 228        |
| 2.57 swapon 指令: 启动系统交换分区.....                     | 230        |
| 2.58 sysctl 指令: 设置系统核心参数.....                     | 231        |
| 2.59 telinit 指令: 设置系统的执行级别 .....                  | 233        |
| 2.60 timeconfig 指令: 设置时区 .....                    | 234        |
| 2.61 ulimit 指令: 控制 shell 程序的资源.....               | 235        |
| 2.62 unalias 指令: 删除别名.....                        | 237        |
| 2.63 unset 指令: 删除变量或函数.....                       | 238        |
| 2.64 up2date 指令: 更新 Linux 系统.....                 | 240        |
| 2.65 vmstat 指令: 显示虚拟内存信息.....                     | 242        |
| 2.66 yes 指令: 持续输出给定的字符串, 每行显示一个字符串.....           | 245        |
| <b>第3章 文本编辑.....</b>                              | <b>247</b> |
| 3.1 awk 指令: 模式匹配语言 .....                          | 248        |
| 3.2 col 指令: 过滤控制字符 .....                          | 250        |

|                                          |            |
|------------------------------------------|------------|
| 3.3 colm 指令: 删除指定的列.....                 | 251        |
| 3.4 comm 指令: 比较排序文件.....                 | 252        |
| 3.5 ed 指令: 文本编辑器.....                    | 254        |
| 3.6 egrep 指令: 输出匹配某种模式的行.....            | 256        |
| 3.7 ext2ed 指令: ext2 文件系统编辑.....          | 256        |
| 3.8 fgrep 指令: 匹配字符串.....                 | 256        |
| 3.9 fmt 指令: 编排文本文件.....                  | 259        |
| 3.10 gedit 指令: gnome 的文本编辑器.....         | 260        |
| 3.11 head 指令: 输出文件开头的部分信息.....           | 261        |
| 3.12 ispell 指令: 拼字检查程序.....              | 263        |
| 3.13 jed 指令: 编辑文本文件.....                 | 266        |
| 3.14 joe 指令: 编辑文本文件.....                 | 267        |
| 3.15 join 指令: 将两个文件中与指定栏位内容相同的行连接起来..... | 272        |
| 3.16 less 指令: 一次显示一页文本.....              | 275        |
| 3.17 more 指令: 显示文件信息.....                | 277        |
| 3.18 nano 指令: 文本编辑器.....                 | 278        |
| 3.19 pg 指令: 浏览文件.....                    | 280        |
| 3.20 pico 指令: 编辑文字文件.....                | 283        |
| 3.21 sed 指令: 利用 script 命令处理文本文件.....     | 285        |
| 3.22 sort 指令: 将文本文件内容加以排序.....           | 291        |
| 3.23 spell 指令: 拼字检查程序.....               | 297        |
| 3.24 tr 指令: 转换文件中的字符.....                | 297        |
| 3.25 uniq 指令: 检查文件中重复出现的行.....           | 299        |
| 3.26 vi 指令: 文字编辑器.....                   | 301        |
| 3.27 vim 指令: 增强型 vi 编辑器.....             | 306        |
| 3.28 view 指令: 文字编辑器.....                 | 307        |
| 3.29 wc 指令: 计算字数.....                    | 308        |
| <b>第4章 文件传输.....</b>                     | <b>311</b> |
| 4.1 bye 指令: 中断FTP连接.....                 | 312        |



## 目录

|                                     |            |
|-------------------------------------|------------|
| 4.2 fold 指令: 限制文件列宽.....            | 312        |
| 4.3 ftp 指令: 文件传输协议.....             | 314        |
| 4.4 ftptcount 指令: 显示 ftp 登录用户数..... | 321        |
| 4.5 ftptshut 指令: 定时关闭 ftp 服务器.....  | 321        |
| 4.6 ftptwho 指令: 显示 ftp 登录用户信息.....  | 322        |
| 4.7 ncftp 指令: 传输文件.....             | 322        |
| 4.8 ncftppget 指令: 下载文件.....         | 327        |
| 4.9 ncftpls 指令: 显示文件目录.....         | 328        |
| 4.10 ncftpput 指令: 上传文件.....         | 329        |
| 4.11 tftp 指令: 传输文件.....             | 330        |
| 4.12 uucico 指令: uucp 文件传输.....      | 332        |
| 4.13 uucp 指令: 在 UNIX 系统之间传送文件.....  | 333        |
| 4.14 uupick 指令: 处理文件.....           | 334        |
| 4.15 uuto 指令: 文件传送到远端主机.....        | 335        |
| <b>第5章 文件管理.....</b>                | <b>337</b> |
| 5.1 aspell 指令: 检查文件的错误.....         | 338        |
| 5.2 attr 指令: XFS 文件系统对象的扩展属性.....   | 340        |
| 5.3 basename 指令: 显示文件或者目录的基本名称..... | 341        |
| 5.4 chattr 指令: 改变文件的属性.....         | 341        |
| 5.5 chgrp 指令: 改变文件或者目录所属的群组.....    | 343        |
| 5.6 chmod 指令: 设置文件或者目录的权限.....      | 345        |
| 5.7 chown 指令: 改变文件的拥有者或者群组.....     | 347        |
| 5.8 cksum 指令: 文件的 CRC 校验.....       | 349        |
| 5.9 cmp 指令: 比较文件差异.....             | 350        |
| 5.10 cp 指令: 复制.....                 | 352        |
| 5.11 csplit 指令: 分割文件.....           | 355        |
| 5.12 diff 指令: 生成差异信息.....           | 356        |
| 5.13 diffstat 指令: diff 结果的统计信息..... | 362        |
| 5.14 dirname 指令: 显示文件的除名字外的路径.....  | 363        |

|                                                                        |     |
|------------------------------------------------------------------------|-----|
| 5.15 file 指令: 辨识文件类型.....                                              | 364 |
| 5.16 filterdiff 指令: 从 diff 文件中提取不同.....                                | 365 |
| 5.17 find 指令: 查找目录或者文件.....                                            | 366 |
| 5.18 findfs 指令: 通过列表或者用户 ID 查找文件系统.....                                | 374 |
| 5.19 git 指令: 在文字模式下管理文件.....                                           | 375 |
| 5.20 indent 指令: 调整 C 原始代码文件的格式.....                                    | 375 |
| 5.21 ln 指令: 链接文件或目录.....                                               | 378 |
| 5.22 lndir 指令: 连接目录内容.....                                             | 380 |
| 5.23 locate 指令: 查找文件.....                                              | 381 |
| 5.24 lsattr 指令: 显示文件属性.....                                            | 382 |
| 5.25 mattrib 指令: 变更或显示 MS-DOS 文件的属性.....                               | 384 |
| 5.26 mc 指令: 交互式文件管理程序.....                                             | 385 |
| 5.27 mcopy 指令: 复制 MS-DOS 文件到 UNIX 系统, 或者<br>从 UNIX 系统复制 MS-DOS 文件..... | 387 |
| 5.28 md5sum 指令: 检验文件.....                                              | 387 |
| 5.29 mdel 指令: MS-DOS 文件删除.....                                         | 389 |
| 5.30 mdir 指令: 显示 MS-DOS 目录.....                                        | 390 |
| 5.31 mkdir 指令: 建立目录.....                                               | 390 |
| 5.32 mktemp 指令: 建立暂存文件.....                                            | 391 |
| 5.33 mmmove 指令: 移动 MS-DOS 文件.....                                      | 392 |
| 5.34 mread 指令: 复制 MS-DOS 文件.....                                       | 393 |
| 5.35 mren 指令: 更改 MS-DOS 文件名称.....                                      | 393 |
| 5.36 mshowfat 指令: 显示 MS-DOS 文件的记录.....                                 | 393 |
| 5.37 mtools 指令: 显示 mtools 支持的指令.....                                   | 394 |
| 5.38 mtoolstest 指令: 测试并显示 mtools 的相关设置.....                            | 394 |
| 5.39 mv 指令: 移动或更名现有的文件或目录.....                                         | 397 |
| 5.40 od 指令: 输出文件内容.....                                                | 398 |
| 5.41 paste 指令: 合并文件的列.....                                             | 402 |
| 5.42 patch 指令: 修补文件.....                                               | 404 |
| 5.43 rcp 指令: 复制远端主机的文件或目录.....                                         | 407 |

## 目录



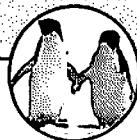
|                                        |            |
|----------------------------------------|------------|
| 5.44 rmask 指令: 产生加密文件.....             | 408        |
| 5.45 rm 指令: 删除文件或目录.....               | 408        |
| 5.46 rmdir 指令: 删除目录.....               | 413        |
| 5.47 scp 指令: 远程复制文件.....               | 415        |
| 5.48 slocate 指令: 查找文件或目录.....          | 417        |
| 5.49 split 指令: 切割文件.....               | 419        |
| 5.50 stat 指令: 显示 inode 内容.....         | 421        |
| 5.51 sum 指令: 计算文件的校验码.....             | 423        |
| 5.52 tee 指令: 读取标准输入到标准输出并可保存为文件.....   | 424        |
| 5.53 tmpwatch 指令: 删除临时文件.....          | 426        |
| 5.54 touch 指令: 更新文件或目录时间.....          | 427        |
| 5.55 tree 指令: 以树状图显示目录内容.....          | 428        |
| 5.56 umask 指令: 指定在建立文件时预设的权限掩码.....    | 434        |
| <b>第6章 备份压缩.....</b>                   | <b>437</b> |
| 6.1 ar 指令: 建立、修改或从档案文件中提取文件.....       | 438        |
| 6.2 bunzip2 指令: 解压缩.bz2 类型的文件.....     | 439        |
| 6.3 bzip2 指令: 解压缩.bz2 文件.....          | 441        |
| 6.4 bzip2recover 指令: 损坏.bz2 文件的修复..... | 442        |
| 6.5 compress 指令: 压缩解压文件.....           | 443        |
| 6.6 cpio 指令: 备份文件.....                 | 444        |
| 6.7 dump 指令: 文件系统备份.....               | 450        |
| 6.8 fdisk 指令: Linux 分区表控制.....         | 451        |
| 6.9 fsck 指令: 检查文件系统并尝试修复错误.....        | 454        |
| 6.10 fsck.ext2 指令: 检查 ext2 文件系统.....   | 456        |
| 6.11 fsck.ext3 指令: 检查 ext3 文件系统.....   | 457        |
| 6.12 fsck.minix 指令: 检查文件系统并尝试修复错误..... | 458        |
| 6.13 gunzip 指令: 解压缩文件.....             | 459        |
| 6.14 gzip 指令: 压缩可执行文件.....             | 460        |
| 6.15 gzip 指令: 压缩文件.....                | 460        |

|                                            |            |
|--------------------------------------------|------------|
| 6.16 hparm 指令：显示和设定硬盘参数.....               | 462        |
| 6.17 lha 指令：压缩或解压缩文件.....                  | 464        |
| 6.18 tar 指令：压缩/解压缩文件.....                  | 465        |
| 6.29 umount 指令：卸除文件系统.....                 | 467        |
| 6.20 unrar 指令：解压缩文件.....                   | 468        |
| 6.21 uncompress 指令：解压缩.Z 文件.....           | 469        |
| 6.22 unzip 指令：解压缩.zip 文件.....              | 470        |
| 6.23 zip 指令：压缩文件.....                      | 472        |
| 6.24 zipinfo 指令：显示压缩文件的信息.....             | 476        |
| <b>第7章 网络通信.....</b>                       | <b>481</b> |
| 7.1 apachectl 指令：apache HTTP 服务器控制接口 ..... | 482        |
| 7.2 arp 指令：系统 ARP 缓存.....                  | 483        |
| 7.3 arpwatch 指令：监听 ARP 记录.....             | 484        |
| 7.4 arping 指令：向邻居主机发送 ARP 请求.....          | 485        |
| 7.5 cu 指令：主机间通信.....                       | 488        |
| 7.6 dip 指令：IP 拨号连接.....                    | 489        |
| 7.7 gaim 指令：即时信息传输.....                    | 490        |
| 7.8 getty 指令：设置终端配置.....                   | 490        |
| 7.9 host 指令：dns 查询.....                    | 491        |
| 7.10 httpd 指令：apache http 服务器程序.....       | 495        |
| 7.11 ifconfig 指令：显示或者配置网络设备.....           | 497        |
| 7.12 iptables 指令：IPv4 的包过滤和 nat 的管理.....   | 500        |
| 7.13 iptables-save 指令：IP 列表存储.....         | 504        |
| 7.14 iwconfig 指令：配置无线网络设备.....             | 507        |
| 7.15 mesg 指令：控制终端的写入.....                  | 508        |
| 7.16 mingetty 指令：精简版的 getty.....           | 509        |
| 7.17 minicom 指令：调制解调器通信程序.....             | 510        |
| 7.18 mkfifo 指令：创建管道.....                   | 511        |
| 7.19 mtr 指令：网络诊断工具.....                    | 512        |

## 目录

|                                 |     |
|---------------------------------|-----|
| 7.20 nc 指令：设置路由器                | 513 |
| 7.21 netconfig 指令：设置各项网络功能      | 516 |
| 7.22 netstat 指令：显示网络状态          | 517 |
| 7.23 nslookup 指令：dns 查找         | 523 |
| 7.24 ping 指令：检测主机 (IPv4)        | 525 |
| 7.25 ping6 指令：检测主机 (IPv6)       | 527 |
| 7.26 pppd 指令：ppp 连线的守护进程        | 529 |
| 7.27 ppp-off 指令：关闭 ppp 连线       | 530 |
| 7.28 pppsetup 指令：设置 ppp 连线      | 530 |
| 7.29 pppstats 指令：显示 ppp 连线状态    | 530 |
| 7.30 rdate 指令：显示其他主机的日期与时间      | 531 |
| 7.31 route 指令：显示并设置路由           | 532 |
| 7.32 samba 指令：控制 Samba 服务端      | 535 |
| 7.33 smbd 指令：Samba 服务端          | 536 |
| 7.34 ssh 指令：远程登录                | 537 |
| 7.35 statserial 指令：samba 服务器程序  | 538 |
| 7.36 talk 指令：与其他用户交谈            | 540 |
| 7.37 tcpdump 指令：倾倒网络传输数据        | 540 |
| 7.38 telnet 指令：远程登录             | 545 |
| 7.39 testparm 指令：测试 Samba 配置    | 547 |
| 7.40 tracepath 指令：追踪路径          | 549 |
| 7.41 traceroute 指令：显示数据包到主机间的路径 | 550 |
| 7.42 tty 指令：显示标准输入设备名称          | 551 |
| 7.43 uulog 指令：显示 uucp 记录信息      | 552 |
| 7.44 uname 指令：显示 uucp 远端主机      | 553 |
| 7.45 uustat 指令：显示 uucp 状态       | 554 |
| 7.46 uux 指令：在远端的 uucp 主机上运行指令   | 555 |
| 7.47 wall 指令：发送信息               | 556 |
| 7.48 wget 指令：从互联网下载资源           | 556 |
| 7.49 write 指令：传送信息              | 558 |

|                                          |            |
|------------------------------------------|------------|
| 7.50 ytalk 指令：与其他用户交谈 .....              | 559        |
| <b>第8章 磁盘管理.....</b>                     | <b>561</b> |
| 8.1 automount 指令：为 auto 文件系统配置挂载点 .....  | 562        |
| 8.2 df 指令：磁盘信息.....                      | 563        |
| 8.3 dirs 指令：显示目录信息 .....                 | 565        |
| 8.4 du 指令：显示目录或者文件所占的磁盘空间 .....          | 566        |
| 8.5 mt 指令：磁带驱动器操作.....                   | 569        |
| 8.6 mzip 指令：zip/jaz 磁盘驱动器控制指令.....       | 571        |
| 8.7 quota 指令：显示磁盘已使用的空间与限制.....          | 571        |
| 8.8 quotacheck 指令：检查磁盘的使用空间与限制.....      | 572        |
| 8.9 quotaoff 指令：关闭磁盘空间限制.....            | 573        |
| 8.10 quotaon 指令：开启磁盘空间限制 .....           | 574        |
| 8.11 quotastats 指令：显示磁盘空间的限制.....        | 575        |
| 8.12 raidstop 指令：关闭软件控制的磁盘阵列 .....       | 576        |
| 8.13 repquota 指令：检查磁盘空间限制的状态.....        | 576        |
| 8.14 rmt 指令：远端磁带传输协议模块.....              | 577        |
| 8.15 sfdisk 指令：硬盘分区工具程序.....             | 577        |
| 8.16 sync 指令：将内存缓冲区内的数据写入磁盘.....         | 579        |
| <b>第9章 磁盘维护.....</b>                     | <b>581</b> |
| 9.1 badblocks 指令：检查磁盘坏道.....             | 582        |
| 9.2 cfdisk 指令：磁盘分区.....                  | 583        |
| 9.3 dd 指令：转换复制文件.....                    | 584        |
| 9.4 e2fsck 指令：检查 ext2 文件系统.....          | 586        |
| 9.5 mkbootdisk 指令：建立目前系统的启动盘.....        | 587        |
| 9.6 mke2fs 指令：建立 ext2 文件系统.....          | 588        |
| 9.7 mkfs 指令：建立各种文件系统.....                | 589        |
| 9.8 mkfs.minix 指令：建立 minix 文件系统.....     | 590        |
| 9.9 mkinitrd 指令：建立要载入 ramdisk 的映像文件..... | 591        |



## 目录

|                                              |            |
|----------------------------------------------|------------|
| 9.10 mkisofs 指令: 建立 iso 9660 映像文件.....       | 592        |
| 9.11 mkswap 指令: 设置交换区(swap area).....        | 593        |
| 9.12 restore 指令: 还原 dump 操作备份的文件.....        | 594        |
| <b>第 10 章 程序编译.....</b>                      | <b>597</b> |
| 10.1 as 指令: 标准 GNU 汇编程序.....                 | 598        |
| 10.2 autoconf 指令: 产生配置脚本.....                | 601        |
| 10.3 autoheader 指令: 为 configure 产生模板头文件..... | 603        |
| 10.4 autoreconf 指令: 更新已经生成的配置文件.....         | 604        |
| 10.5 autoscan 指令: 产生初步的 configure.in 文件..... | 605        |
| 10.6 autoupdate 指令: 更新 configure.in 文件.....  | 606        |
| 10.7 gcc 指令: GNU 的 C 和 C++ 编译器.....          | 607        |
| 10.8 gdb 指令: GNU 调试器.....                    | 608        |
| 10.9 gdbserver 指令: 远端 GNU 服务器.....           | 610        |
| 10.10 make 指令: 编译内核或模块.....                  | 611        |
| <b>第 11 章 打印作业.....</b>                      | <b>613</b> |
| 11.1 cat 指令: 输出文件内容.....                     | 614        |
| 11.2 cut 指令: 剪切文件.....                       | 615        |
| 11.3 pr 指令: 编排文件格式.....                      | 617        |
| 11.4 tac 指令: 反序输出文件.....                     | 619        |
| 11.5 tail 指令: 显示文件的末尾内容.....                 | 621        |
| 11.6 zcat 指令: 显示压缩文件的内容.....                 | 624        |
| <b>第 12 章 电子邮件新闻组.....</b>                   | <b>627</b> |
| 12.1 fetchmail 指令: 获得邮件.....                 | 628        |
| 12.2 getlist 指令: 下载新闻.....                   | 631        |
| 12.3 mail 指令: 收发邮件.....                      | 632        |
| 12.4 mailq 指令: 显示发件箱的邮件.....                 | 635        |
| 12.5 mutt 指令: e-mail 管理.....                 | 636        |

|                                                              |            |
|--------------------------------------------------------------|------------|
| 12.6 nntpget 指令：从新闻服务器下载文章.....                              | 637        |
| 12.7 pine 指令：收发邮件.....                                       | 638        |
| 12.8 slm 指令：新闻阅读程序.....                                      | 639        |
| <b>第 13 章 X Windows System.....</b>                          | <b>641</b> |
| 13.1 startx 指令：启动图形界面.....                                   | 642        |
| 13.2 xset 指令：设置 X windows.....                               | 642        |
| <b>第 14 章 格式转换.....</b>                                      | <b>645</b> |
| 14.1 dvips 指令：将 DVI 文件转换为 Postscript 文件.....                 | 646        |
| 14.2 fiascotopnm 指令：将压缩的 fiasco 镜像文件转换为<br>pgm 或 ppm 格式..... | 647        |
| 14.3 find2perl 指令：将 find 指令行转换为 perl 代码.....                 | 648        |
| 14.4 gemtopbm 指令：转换图形文件.....                                 | 648        |
| 14.5 giftopnm 指令：文件转换.....                                   | 649        |
| 14.6 iconv 指令：字符集转换.....                                     | 650        |
| 14.7 pcxtoppm 指令：转换图形文件.....                                 | 652        |
| 14.8 pfbtops 指令：转换字体文件.....                                  | 652        |
| 14.9 pictoppm 指令：文件转换.....                                   | 653        |
| 14.10 piltoppm 指令：转换图形文件.....                                | 654        |
| 14.11 pjtoppm 指令：转换打印文件.....                                 | 655        |
| 14.12 qrttoppm 指令：转换 qrt 文件.....                             | 655        |
| 14.13 sox 指令：音频文件转换.....                                     | 656        |
| <b>第 15 章 系统软件工具.....</b>                                    | <b>659</b> |
| 15.1 dc 指令：一个任意精度的计算器.....                                   | 660        |
| 15.2 expr 指令：简单计算器.....                                      | 662        |
| <b>附录 指令索引（按首字母排序）.....</b>                                  | <b>665</b> |



TopSage.com is a comprehensive online platform that offers a wide range of services and resources for individuals and businesses.

Our website features a user-friendly interface and a variety of tools and resources designed to help you succeed in your personal and professional endeavors.

Whether you're looking for information on a specific topic, or need to connect with other users, TopSage.com has something for everyone.

We invite you to explore our website and discover the many benefits of being part of the TopSage community.

Thank you for visiting TopSage.com, and we hope to see you soon!

Best regards,

The TopSage Team

TopSage.com

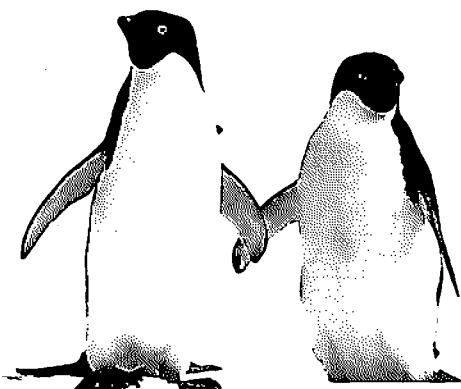


TopSage.com

# 第1章

## 系统管理

在 Linux 系统中，系统管理指令的主要功能是管理系统的进程和性能。常见的指令包括设置系统的历史记录、密码、系统任务、登录信息、退出信息的指令，以及创建用户、创建群组、开机关机、进程信息、重启、休眠等指令。本章将详细介绍系统管理的相关指令。





## 1.1

# adduser 指令：增加一个系统用户



**adduser[必要参数][选择性参数]用户名**



**功能说明** adduser 指令根据参数给系统添加用户，它的配置信息保存在/etc/adduser.conf 文件中。如果调用时没有指定“-system”或“-group”参数，adduser 指令在默认的情况下将添加一个一般用户；如果要添加一个系统用户，可以用“-system”参数；如果要添加一个用户组，可以用“-group”参数。



必要参数的具体说明如下：

| 参数              | 功 能                   |
|-----------------|-----------------------|
| -t system       | 创建一个系统用户              |
| -no-create-home | 不创建用户的/home 目录，即使它不存在 |
| -q quiet        | 只显示警告信息和错误信息，不显示一般信息  |
| -debug          | 用于调试                  |

选择性参数的具体说明如下：

| 参数                 | 功 能                          |
|--------------------|------------------------------|
| -d home<目录>        | 指定目录作为用户的/home 目录            |
| -s shell s<SHELL>  | 用指定的 SHELL 作为用户登录时的 shell    |
| -u uid<用户 ID>      | 用指定的用户 ID 作为用户的 ID，而不是默认分配   |
| -firstuid<用户 ID>   | 忽略指定的第一个用户                   |
| -lastid<用户 ID>     | 忽略指定的最后一个用户                  |
| -group<用户组>        | 当创建用户时，用户属于指定的用户组            |
| -disabled-password | 不设定密码，用户可以登录                 |
| -disable-login     | 不设定密码，但用户不能登录，直到密码被设定        |
| -ingroup<用户组>      | 将增加的用户归入指定的用户组，而不是创建一个用户组    |
| -add_extra_groups  | 将用户归入 adduser.conf 指定的额外的用户组 |

续表

| 参    数           | 功    能                    |
|------------------|---------------------------|
| -help-h          | 帮助信息                      |
| -version         | 版本信息                      |
| -conf<文件>        | 用指定的文件代替/etc/adduser.conf |
| -p -password<密码> | 为添加的用户指定密码                |

**◆ 参数使用说明**

-r|-system：表示“-r”或者“--system”。

-home<目录>：表示“--home”参数后面加一个指定目录作为参数，比如/home/temp/为目录名称，则参数表示为“-home/home/temp”。而是实际的目录路径。

**◆ 操作范例**

◎ 范例 1 ◎ 添加个一般用户。

```
root@localhost:~# useradd kk //添加用户 kk
```

◎ 范例 2 ◎ 为添加的用户指定相应的用户组。

```
root@localhost:~# useradd -g root kk //添加用户 kk，并指定用户所在的组为 root 用户组
```

◎ 范例 3 ◎ 创建一个系统用户。

```
root@localhost:~# useradd -r kk //创建一个系统用户 kk
```

◎ 范例 4 ◎ 为新添加的用户指定/home 目录。

```
root@localhost:~# useradd -d /home/myf kk //新添加用户 kk，其 home 目录为/home/myf
```

当用用户名 kk 登录主机时，系统进入的默认目录为/home/myf。

**◆ 相关指令**

deluser、groupadd、usermod

**1.2 apm 指令：高级电源管理****◆ 基础** apm[参数]

**◆ 功能说明** apm 指令用于高级电源管理。当不输入参数直接输



入 `apm` 指令时，系统将以用户可读的方式直接输出 `/proc/apm` 文件的内容。

### ◎ 功能说明

参数具体说明如下：

| 参 数                     | 功 能            |
|-------------------------|----------------|
| <code>-V version</code> | 版本信息           |
| <code>-v verbose</code> | APM 子系统的版本信息   |
| <code>-m minutes</code> | 以分钟数的形式输出剩余的时间 |
| <code>-s standby</code> | 进入 APM 的备用状态   |
| <code>-s suspend</code> | 进入 APM 的休眠状态   |
| <code>-d debug</code>   | 调试输出信息         |

### ◎ 案例说明

◎ 范例 5 ◎ 显示剩余电量。

```
[root@localhost ~]# apm -m //显示剩余电量
```

本例使用指令 “`apm -m`” 以分钟数的形式显示剩余的电量。

### ◎ 相关指令

`apmd`、`apmsleep`

## 1.3

## apmsleep 指令：APM 进入备用或者休眠状态

### ◎ 功能说明

`apmsleep[必要选项][选择性选项][+时间]`

◎ 功能说明 有些计算机，特别是笔记本电脑，能使用 RTC（实时锁）从低功耗的休眠状态进入 DRAM 模式。`apmsleep` 指令用于在 RTC 中设置警告时间，并进入休眠模式或者备用模式，默认情况下进入休眠模式。

### ◎ 功能说明

必要参数的具体说明如下：

| 参 数             | 功 能                               |
|-----------------|-----------------------------------|
| <code>-s</code> | 进入休眠模式，对于笔记本来说，休眠模式将关闭除内存以外所有的程序  |
| <code>-S</code> | 进入备用模式，对于笔记本来说，备用模式将关闭显示器、硬盘和 CPU |

## 1.4 apropos 指令：查找使用手册的名字和相关描述

续表

| 参数 | 功能                                   |
|----|--------------------------------------|
| -w | 等待时间跳                                |
| -p | 等待警告时间与实际时间相匹配                       |
| -n | 不调用 APM BIOS 来休眠电脑，只是设置警告时钟，等待时间跳的到来 |
| -d | 输出正在进行的任务的信息                         |

选择性参数的具体说明如下：

| 参数          | 功能   |
|-------------|------|
| -V -version | 版本信息 |
| -help       | 帮助信息 |

### ◎ 技术范例

◎ 范例 6 ◎ 进入休眠状态一段时间。

```
[root@localhost ~]# apmsleep +1:50 //进入休眠状态
```

进入休眠状态，休眠的时间为 1 小时 50 分钟。

```
[root@localhost ~]# apmsleep 9:00 //进入休眠状态
```

直到第二天早晨 9 点，计算机一直处于休眠状态。

◎ 范例 7 ◎ 进入备用模式一段时间。

```
[root@localhost ~]# apmsleep -s +1:50 //进入备用模式
```

进入备用模式，备用模式的时间为 1 小时 50 分钟。

```
[root@localhost ~]# apmsleep -s 9:00 //进入备用模式
```

直到第二天早晨 9 点，计算机一直处于备用模式。

### ◎ 相关指令

apmd、apm

## 1.4

## apropos 指令：查找使用手册的名字和相关描述

### ◎ 语法

apropos[必要参数][选择性参数]关键词

### ◎ 功能说明

每个指令的使用手册都有一个简短的描述，通过



**apropos** 指令可以查找相关的关键字的描述。若退出状态为 0，则说明程序成功执行；若退出状态为 1，则说明有用法、语法或者配置错误；若退出状态为 2，则说明操作错误；若退出状态为 16，则说明没有找到与指定的关键词相匹配的内容。

#### 【参数说明】

必要参数的具体说明如下：

| 参 数 | 功 能               |
|-----|-------------------|
| -d  | 输出调试信息            |
| -v  | 输出详细的警告信息         |
| -r  | 认为每一个关键词是一个通常的表达式 |
| -w  | 认为每一个关键词是一种统配符的信息 |
| -e  | 对每一个关键词进行精确匹配     |

选择性参数的具体说明如下：

| 参 数    | 功 能                                 |
|--------|-------------------------------------|
| -s     | 只在给定的部分进行搜索                         |
| -m     | 如果当前系统拥有其他系统的使用手册权限，即可以使用此选项选择相应的系统 |
| -M     | 指定一个可选的路径进行查询                       |
| -C<文件> | 使用指定的文件作为配置文件，而不是使用默认的文件～/.manpath  |
| -h     | 帮助信息                                |
| -V     | 版本信息                                |

#### 【环境变量】

- 系统参数：如果\$SYSTEM 已经指定，那么它将与使用“-m”参数具有相同的效果。
- 路径参数：如果\$MANPATH 已经指定，那么它将作为执行搜索的默认路径。
- 匹配参数：如果\$POSIX\_CORRECT 已经指定，即使没有给它赋

## 1.4 apropos 指令：查找使用手册的名字和相关描述

值，那么搜索的匹配方式将按照通常的方式进行，和使用“-r”参数具有相同的效果。

### 执行范例

#### 范例 8 以通常方式进行关键词匹配。

```
root@localhost:~# apropos -r ls //以通常方式进行关键词ls
_llseek (2) - reposition read/write file offset
add-shell (8) - add shells to the list of valid login shells
afs_syscall (2) - unimplemented system calls
Apache2::PerlSections (3pm) - write Apache configuration files in Perl
Apache2::RequestUtil (3pm) - Perl API for Apache request record utils
Apache2::ServerUtil (3pm) - Perl API for Apache server record utils
APR::Pool (3pm) - Perl API for APR pools
assert (3) - abort the program if assertion is false
auth_destroy (3) - library routines for remote procedure calls
authnone_create (3) - library routines for remote procedure calls
authunix_create (3) - library routines for remote procedure calls
authunix_create_default (3) - library routines for remote procedure calls
blockdev (8) - call block device ioctl from the command line
break (2) - unimplemented system calls
c++filt (1) - Demangle C++ and Java symbols.
```

运行以上指令之后，系统列出了所有与 ls 相关的指令信息及其描述，列出的信息很长，这里只截取了前面部分信息进行说明。此指令与指令“apropos ls”等效。

#### 范例 9 进行关键词的精确匹配。

```
root@localhost:~# apropos -e ls //以精确的方式查找关键词ls
dircolors (1) - color setup for ls
ls (1) - list directory contents
```

#### 范例 10 以通配符方式进行关键词匹配。

```
root@localhost:~# apropos -w ls //以通配符方式进行关键词ls
ls (1) - list directory contents
```

### 相关指令

whatis、man



## 1.5 arch 指令：输出主机的体系结构

命令 arch

功能说明 arch 指令等同于指令“uname -m”。在当前的 Linux 系统下，arch 指令的输出结果有：i386、i486、i586、alpha、sparc、arm、m68k、mips、ppc 和 i686 等。

执行范例

● 范例 11 ● 输出当前系统的体系结构。

```
root@localhost:~# arch //输出当前系统的体系结构
i686
root@localhost:~#
```

相关指令

uname

## 1.6 batch 指令：执行批处理指令

命令 batch[必要参数][选择性参数]

功能说明 batch 指令用来进行批处理。

特殊说明

必要参数的具体说明如下：

| 参 数 | 功 能      |
|-----|----------|
| -m  | email 设置 |

选择性参数的具体说明如下：

| 参 数    | 功 能     |
|--------|---------|
| -f<文件> | 从文件读取配置 |
| -q<队列> | 指定队列    |
| -V     | 版本信息    |

**● 执行范例****● 范例 12 ● 批处理。**

```
[root@localhost temp]# batch
at> rm desfile
at> rm jiamofile
at> <EOT>
job 10 at 2007-06-26 14:54
[root@localhost temp]# ls
srcfile
[root@localhost temp]#
```

//批处理

//删除文件

//删除文件

//结束输入

本例使用指令 batch 进行批处理。

**● 相关指令**

cron、nice、sh、umask、atd

**1.7****bg 指令：将程序放在后台执行****● 命令 bg[工作编号]****● 功能说明** bg 指令用来将程序放到后台执行。**● 执行范例****● 范例 13 ● 将程序放到后台执行。**

```
[root@localhost temp]# ed
```

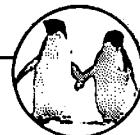
```
[6]+ Stopped ed
[root@localhost temp]# bg 6 //将程序放到后台执行
[6]+ ed &
[root@localhost temp]#
```

本例使用指令 bg 6 将 6 号程序放到后台执行。

**● 相关指令**

&

**1.8****cd 指令：切换目录****● 命令 cd[目录]**



**功能说明** cd 指令改变当前目录到指定目录。变量 HOME 是默认的目录路径。

**执行范例**

◎ 范例 14 ◎ 改变当前目录。

```
[root@localhost root]# pwd //显示当前目录
/root
[root@localhost root]# cd /home/kk //改变目录到/home/kk
[root@localhost kk]# pwd //显示当前目录
/home/kk
[root@localhost kk]#
```

第一次显示当前目录时，显示的是/root。使用指令 cd /home/kk，改变当前路径到/home/kk，再利用 pwd 指令显示当前目录时，显示的是 /home/kk。

**相关指令**

pwd

## 1.9 chfn 指令：设置 finger 信息

**基础** chfn[选择性参数][账号名称]

**功能说明** chfn 指令用于改变 finger 信息。finger 信息保存在 /etc/passwd 文件中，并将被 finger 程序显示。Linux 系统提供的 finger 指令显示的 4 种信息为姓名、工作地址、办公电话和家庭电话，它们可以被 chfn 指令改变。

**功能说明**

选择性参数的具体说明如下：

| 参数       | 功能        |
|----------|-----------|
| -f<姓名>   | 设置账号的真实姓名 |
| -h<电话号码> | 设置账号家庭电话  |
| -o<地址>   | 设置账号的办公地址 |
| -p<电话号码> | 设置账号的办公电话 |

## 1.9 chfn 指令：设置 finger 信息

续表

| 参数       | 功能     |
|----------|--------|
| -help    | 显示帮助信息 |
| -version | 显示版本信息 |

 线行范例

## ○ 范例 15 ○ 改变 finger 信息。

```
[root@localhost root]# chfn //改变 finger 信息
Changing finger information for root.
Name [root]: kk
Office []: bjtu iplab604
Office Phone []: 88888888
Home Phone []: 66666666
Finger information changed.
[root@localhost root]#
```

输入 chfn 指令，进入用户交互模式。按照系统提示依次输入账号名称、工作地址、办公电话和家庭电话。

## ○ 范例 16 ○ 改变账号的真实姓名。

```
[root@localhost root]# chfn -f zhizunbao //改变账号的真实姓名
Changing finger information for root.
Finger information changed.
[root@localhost root]#
```

## ○ 范例 17 ○ 改变账号的家庭电话。

```
[root@localhost root]# chfn -h 44444444 //改变账号的家庭电话
Changing finger information for root.
Finger information changed.
[root@localhost root]#
```

## ○ 范例 18 ○ 改变账号的办公地址。

```
[root@localhost root]# chfn -o bjtuiplab604 //改变账号的办公地址
Changing finger information for root.
Finger information changed.
[root@localhost root]#
```

## ○ 范例 19 ○ 改变账号的办公电话。

```
[root@localhost root]# chfn -p 33333333 //改变账号的办公电话
Changing finger information for root.
```



Finger information changed.  
[root@localhost root]#

### 相关指令

finger、passwd

## 1.10

## chsh 指令：改变登录系统时的 shell

### 语法 chsh[必要参数][选择性参数][用户名]

**功能说明** chsh 指令用来改变用户登录时的 shell。如果指令行没有指定一个 shell，那么 chsh 将提示指定一个 shell。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -s | 指定系统登录时的 shell |
| -l | 列出所有的 shell    |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -h--help | 显示帮助信息 |
| -v       | 显示版本信息 |

### 执行范例

◎ 范例 20 ◎ 列出当前所有的 shell。

```
[root@localhost root]# chsh -l //列出所有的 shell
/bin/sh
/bin/bash
/sbin/nologin
/bin/bash2
/bin/ash
/bin/bsh
/bin/tcsh
/bin/csh
[root@localhost root]#
```

## 1.12 date 指令：显示或设置系统时间

## ◎ 范例 21 ◎ 改变当前的 shell。

```
[root@localhost root]# chsh //shell 设置
Changing shell for root.
New shell [/bin/bash]: /bin/csh //输入新的指定的 shell
Shell changed.
[root@localhost root]#
```

当前的 shell 设置为/bin/bash，通过 chsh 指令，改变 shell 的设置为 /bin/csh。

## ◎ 范例 22 ◎ 通过“-s”参数改变当前的 shell 设置。

```
[root@localhost root]# chsh -s /bin/csh // 改变当前的 shell 设置为
/biin/csh
Changing shell for root.
Shell changed.
[root@localhost root]#
```

## ○○ 相关指令

login、passwd

## 1.11 clear 指令：清除终端屏幕

## ○○ 基本概念

clear

○○ 功能说明

clear 指令用来清除终端屏幕。

## ○○ 运行范例

## ◎ 范例 23 ◎ 清除终端屏幕。

```
[root@localhost root]# clear //清除终端屏幕
```

本例使用指令 clear 清除终端屏幕。

## ○○ 相关指令

tput、terminfo

## 1.12 date 指令：显示或设置系统时间

## ○○ 基本概念

date[选择性参数][必要参数]



**功能说明** 根据给定的格式显示当前的时间或者设置系统时间。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能                   |
|-----|-----------------------|
| %H  | 小时 (00~23)            |
| %I  | 小时 (01~12)            |
| %K  | 小时 (0~23)             |
| %l  | 小时 (0~12)             |
| %M  | 分钟 (00~59)            |
| %P  | AM 或者 PM              |
| %r  | 包含时、分、秒的时间            |
| %s  | UTC 时间，总秒数            |
| %S  | 秒数                    |
| %T  | 包含时、分、秒的时间，小时 (00~23) |
| %X  | 时间，本地时区               |
| %Z  | 时区                    |
| %a  | 星期使用缩写                |
| %A  | 星期使用完整的写法             |
| %b  | 月份缩写，英文               |
| %B  | 月份使用完整的写法，英文          |
| %c  | 包含年、月、日和时、分、秒的时间      |
| %d  | 包含日的日期                |
| %D  | 包含年、月、日的日期            |
| %j  | 天数，从本年年初开始算起          |
| %m  | 包含月的日期                |
| %U  | 周数，从年初开始算起            |
| %w  | 天数，从本周周日开始算起，周日为0     |

## 1.12 date 指令：显示或设置系统时间

续表

| 参 数 | 功 能           |
|-----|---------------|
| %x  | 本地时区的日期       |
| %y  | 包含年的日期(00~99) |
| %Y  | 包含年的日期,用4位数表示 |
| %n  | 输出时,插入【回车】键   |
| %t  | 输出时,插入【tab】键  |
| MM  | 月份            |
| DD  | 日期            |
| hh  | 小时            |
| mm  | 分钟            |
| CC  | 年的前两位         |
| YY  | 年的后两位         |
| ss  | 秒             |
| -U  | GMT 设置        |

选择性参数的具体说明如下：

| 参 数       | 功 能        |
|-----------|------------|
| -d<“字符串”> | 显示字符串代表的时间 |
| -s<“字符串”> | 设置字符串代表的时间 |
| --help    | 帮助信息       |
| --version | 版本信息       |

### ○ 执行范例

#### ◎ 范例 24 ◎ 显示当前时间。

```
[root@localhost ~]# date //显示当前时间
- 5月 14 16:11:10 CST 2007
[root@localhost ~]#
```

本例使用指令 date, 显示当前时间。

#### ◎ 范例 25 ◎ 按指定格式显示时间。



[root@localhost ~]# date '+%c' //显示当前时间

2007年05月14日 星期一 16时11分35秒

[root@localhost ~]#

通过使用指令“date '+%c'”，显示完整的时间。

[root@localhost ~]# date '+%D' //显示当前时间

05/14/07

[root@localhost ~]#

使用指令“date '+%D'”，显示数字日期，年份用两位数字表示。

[root@localhost ~]# date '+%x' //显示当前时间

2007年05月14日

[root@localhost ~]#

使用指令“date '+%x'”，显示日期，年份用四位数字表示。

[root@localhost ~]# date '+%T' //显示当前时间

16:24:11

[root@localhost ~]#

使用指令“date '+%T'”，显示24小时制的时间。

[root@localhost ~]# date '+%X' //显示当前时间

16时23分13秒

[root@localhost ~]#

使用指令“date '+%X'”，显示时、分、秒时间。

◎范例 26 ◎ 在时间输出中加入自己的格式。

[root@localhost ~]# date '+beautiful afternoon %l:%M %p let us go swimming' //显示时间

beautiful afternoon 4:30 下午 let us go swimming

[root@localhost ~]#

本例使用指令“date '+beautiful afternoon %l:%M %p let us go swimming'”，按照用户指定的格式输出时间。

### 相关命令

cal

## 1.13 echo 指令：显示文本行

### 命令功能

echo[必要参数][选择性参数][字符串]

### 功能说明

每一个 shell 支持的 echo 指令可能都不太相同。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能         |
|----|------------|
| -n | 不输出最后的“\n” |
| -e | 解释转义字符     |
| -E | 不解释转义字符    |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

**【转义字符】**

- \a: 警告。
- \b: 删除。
- \c: 相当于“-n”参数，不输出末尾的“\n”。
- \f: 换页。
- \n: 换行。
- \r: 回车。
- \t: 水平方向上的 tab。
- \v: 垂直方向上的 tab。
- \\: 反斜杠。
- '': 单引号。
- "": 双引号。
- \nnn: 字符 nnn。

**执行范例****◎ 范例 27 ◎ 输出指定的字符。**

```
[root@localhost other]# echo i am a prince and i am my kingdom
//输出指定的字符
```

```
i am a prince and i am my kingdom
```

```
[root@localhost other]#
```



## 常用命令

yes

## 1.14

## exec 指令：执行完指令后，交出控制权

## 语法

exec[指令]

## 功能说明

exec 指令用来在执行完指定的指令后交出控制权。

## 执行范例

- 范例 28 执行完指令后，退出。

```
[root@localhost root]# exec ls //执行完ls指令后退出
```

## 常用命令

echo

## 1.15

## exit 指令：shell 退出

## 语法

exit

## 功能说明

exit 指令用来退出当前的 shell。

## 执行范例

- 范例 29 远程登录主机。

```
[root@localhost ~]# telnet 192.168.88.2 //远程登录主机
Trying 192.168.88.2...
Connected to 192.168.88.2 (192.168.88.2).
Escape character is '^].
Debian GNU/Linux 4.0
iplab604 login: iplab604
Password:
Last login: Mon Jun 18 18:22:42 2007 from 192.168.88.7 on pts/0
iplab604@localhost:~$
```

本例使用指令“telnet 192.168.88.2”，远程登录主机 192.168.88.2。

- 范例 30 退出 shell 指令。

## 1.16 fc 指令：修改或者执行指令

```
iplab604@localhost:~$ exit
logout
```

//退出

```
Connection closed by foreign host.
[root@localhost ~]#
```

本例使用指令 exit，退出 shell 指令。

 相关指令

quit

## 1.16 fc 指令：修改或者执行指令

 语法 fc[选择性参数][必要参数][第一个指令][最后一个指令]

 功能说明 fc 指令可以编辑曾经执行过的指令，并以批处理的形式让这些指令重新得以执行。

 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能                   |
|-----|-----------------------|
| -l  | 列出第一个指令和最后一个指令之间的所有指令 |
| -n  | 不显示编号                 |
| -r  | 反向排序                  |

选择性参数的具体说明如下：

| 参 数      | 功 能            |
|----------|----------------|
| -<文本编辑器> | 指定文本编辑器，默认是 vi |
| -s<指令>   | 选择指令执行         |

 执行范例

◎ 范例 31 ◎ 列出当前执行过的指令。

```
root@localhost:~# fc -l //列出当前执行过的指令
```

```
331 clear
332 chkconfig
333 chkconfig --list
```

```

334 ls
335 compress
336 exit
337 who
338 wall
339 wall hello
340 cpio --help
341 ls
342 vi iptables.sh
343 ls
344 clear
345 man evolution
346 exit
root@localhost:~#

```

本例使用指令“fc -l”，列出当前执行过的指令。

○ 范例 32 ○ 执行已经执行过的指令。

```

root@localhost:~# fc 345 339
man evolution
No manual entry for evolution
clear
ls
gao iproute_20061002-2.diff.gz vim-7.0
ipip-1.1.6 iproute_20061002-2.dsc vim_7.0-122+1.diff.gz
ipip_1.1.6.dsc iproute_20061002.orig.tar.gz vim_7.0-122+1.dsc
ipip_1.1.6.tar.gz iptables.sh vim_7.0.orig.tar.gz
iproute-20061002 poweron
vi iptables.sh
ls
gao iproute_20061002-2.diff.gz vim-7.0
ipip-1.1.6 iproute_20061002-2.dsc vim_7.0-122+1.diff.gz
ipip_1.1.6.dsc iproute_20061002.orig.tar.gz vim_7.0-122+1.dsc
ipip_1.1.6.tar.gz iptables.sh vim_7.0.orig.tar.gz
iproute-20061002 poweron
cpio --help
Usage: cpio [OPTION...] [destination-directory]
GNU 'cpio' copies files to and from archives
Examples:

```

## 1.17 fg 指令：将后台任务拉到前台执行

```
Copy files named in name-list to the archive
cpio -o < name-list [> archive]
Extract files from the archive
cpio -i [< archive]
Copy files named in name-list to destination-directory
```

**Informative options:**

|            |                            |
|------------|----------------------------|
| -?, --help | Give this help list        |
| --license  | Print license and exit     |
| --usage    | Give a short usage message |
| --version  | Print program version      |

Mandatory or optional arguments to long options are also mandatory or optional

for any corresponding short options.

Report bugs to <[bug-cpio@gnu.org](mailto:bug-cpio@gnu.org)>.

wall hello

wall: can't read hello.

root@localhost:~#

本例使用指令“fc 345 339”，依次执行编号为345～339的指令，执行结果如上所示。

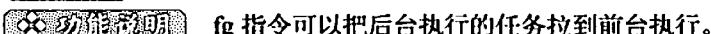


history

## 1.17 fg 指令：将后台任务拉到前台执行



fg[任务号]



fg 指令可以把后台执行的任务拉到前台执行。



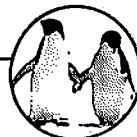
◎ 范例 33 ◎ 将后台的程序拉到前台执行。

[root@localhost ~]# ftp 192.168.88.2 & //程序在后台执行

[1] 9860

[root@localhost ~]# Connected to 192.168.88.2.

220 (vsFTPd 2.0.5)



530 Please login with USER and PASS.

530 Please login with USER and PASS.

KERBEROS\_V4 rejected as an authentication type

Name (192.168.88.2:root): iplab604

-bash: iplab604: command not found

[1]+ Stopped ftp 192.168.88.2

[root@localhost ~]# fg %1 //将后台程序拉到前台执行

ftp 192.168.88.2

本例使用指令“`ftp 192.168.88.2 &`”，使得程序在后台执行，然后使用指令“`fg %1`”，将 `ftp` 进程拉到前台执行。

#### 相关指令

`bg`、`jobs`

## 1.18

### fgconsole 指令：打印虚拟终端的数目

#### 语法

`fgconsole`

#### 功能说明

`fgconsole` 指令用来打印虚拟终端的数目。

#### 执行范例

○ 范例 34 ○ 显示虚拟终端的数目。

[root@localhost ~]# `fgconsole` //显示虚拟终端的数目

7

[root@localhost ~]#

本例使用指令 `fgconsole`，显示虚拟终端的数目。

#### 相关指令

`chvt`

## 1.19

### finger 指令：查找并显示用户信息

#### .. 语法

`finger[必要参数][账号名称]`

#### 功能说明

`finger` 指令显示系统用户的相关信息。

## 1.19 finger 指令：查找并显示用户信息

 参数说明

必要参数的具体说明如下：

| 参数 | 功能                        |
|----|---------------------------|
| -l | 列出用户的账号信息                 |
| -m | 忽略用户的真实姓名                 |
| -s | 列出用户的账号信息                 |
| -p | 列出用户的账号信息，但不显示计划文件和方案文件内容 |

 执行范例

- 范例 35 ○ 列出当前登录用户的相关信息。

```
[root@localhost ~]# finger -l //显示用户信息
Login: root Name: root
Directory: /root Shell: /bin/bash
On since Mon May 21 10:59 (CST) on pts/1 from 192.168.88.7
On since Mon May 21 14:13 (CST) on :0 (messages off)
On since Mon May 21 14:13 (CST) on pts/2 from :0.0
 37 minutes 12 seconds idle
New mail received Mon May 21 11:11 2007 (CST)
 Unread since Fri Dec 22 17:28 2006 (CST)
No Plan.

[root@localhost ~]#
```

本例使用指令“finger -l”，列出本地机器上当前用户的相关信息。

- 范例 36 ○ 显示指定用户信息。

```
[root@localhost ~]# finger -m kk //显示用户 kk 的信息
Login: kk Name: kk
Directory: /home/kk Shell: /bin/bash
Last login Fri Dec 22 16:04 (CST) on :0
No mail.
No Plan.

[root@localhost ~]#
```

本例使用指令“finger -m kk”，显示指定用户 kk 的信息。

- 范例 37 ○ 显示远端用户信息。

```
[root@localhost ~]# finger -m root@192.168.88.2 //显示用户 kk 的信息
```



本例使用指令“`finger -m root@192.168.88.2`”，显示远端用户即主机 IP 地址为 192.168.88.2 的 root 的用户信息。

### 相关指令

`chfn`、`passwd`、`w`、`who`

## 1.20

### free 指令：显示内存信息

#### 语法

`free[必要参数][选择性参数]`

**功能说明** `free` 指令显示系统使用和空闲的内存情况，包括物理内存、交换区内存和内核缓冲区内存。共享内存将被忽略。

#### 参数说明

必要参数的具体说明如下：

| 参数              | 功能          |
|-----------------|-------------|
| <code>-b</code> | 显示内存的单位为字节  |
| <code>-k</code> | 显示内存的单位为 KB |
| <code>-m</code> | 显示内存的单位为 MB |
| <code>-o</code> | 忽略缓冲区调节列    |
| <code>-t</code> | 总和信息        |

选择性参数的具体说明如下：

| 参数                        | 功能                  |
|---------------------------|---------------------|
| <code>-s&lt;时间&gt;</code> | 每间隔指定时间执行一次指令，单位为 s |
| <code>-V</code>           | 版本信息                |

#### 执行范例

◎ 范例 38 ◎ 显示内存的使用情况。

```
[root@localhost ~]# free //显示内存的使用信息
 total used free shared buffers cached
Mem: 515292 499556 15736 0 136404 76736
-/+ buffers/cache: 286416 228876
```

## 1.21 fuser 指令：用文件或者套接口表示进程

```
Swap: 522104 868 521236
[root@localhost ~]#
```

本例使用指令 free，显示内存的使用信息。

◎ 范例 39 ◎ 以总和的形式显示内存的使用信息。

```
[root@localhost ~]# free -t //以总和的形式显示内存的使用信息
total used free shared buffers cached
Mem: 515292 499548 15744 0 136404 76736
-/+ buffers/cache: 286408 228884
Swap: 522104 868 521236
Total: 1037396 500416 536980
[root@localhost ~]#
```

本例使用指令“free -t”，以总和的形式显示内存的使用信息。

◎ 范例 40 ◎ 周期性地查询内存的使用信息。

```
[root@localhost ~]# free -s 5 //每5s执行一次free指令
total used free shared buffers cached
Mem: 515292 499548 15744 0 136456 76736
-/+ buffers/cache: 286356 228936
Swap: 522104 868 521236
total used free shared buffers cached
Mem: 515292 499548 15744 0 13646 76736
-/+ buffers/cache: 286352 228940
Swap: 522104 868 521236
total used free shared buffers cached
Mem: 515292 499548 15744 0 136460 76736
-/+ buffers/cache: 286352 228940
Swap: 522104 868 521236
```

```
[root@localhost ~]#
```

本例使用指令“free -s 5”，每5s执行一次 free 指令，来周期性地检查主机内存的使用和空闲情况。



ps、slabtop、vmstat、top

## 1.21

## fuser 指令：用文件或者套接口表示进程



fuser[必要参数][选择性参数][程序]



**功能说明** fuser 指令用指定的文件或者文件系统显示进程的进程号，默认情况下每一个文件名后会跟着一个字母来表示类型。

#### 【文件类型说明】

- c: 当前目录。
- e: 扩展。
- f: 打开的文件。
- r: 根目录。
- m: 共享库。

#### 【参数说明】

必要参数的具体说明如下：

| 参 数     | 功 能           |
|---------|---------------|
| -l      | 显示信号          |
| -4      | IPv4          |
| -6      | IPv6          |
| -a      | 与程序相关的所有文件    |
| -k      | 删除程序相关的所有文件   |
| -I      | 交互模式          |
| -s      | 不显示处理信息       |
| -u      | PID 显示用户名     |
| -v      | 运行时，显示详细的处理信息 |
| -       | 清零            |
| -m      | 挂载文件系统        |
| -signal | 发送信号          |

选择性参数的具体说明如下：

| 参 数     | 功 能                    |
|---------|------------------------|
| -V      | 版本信息                   |
| -n<关键字> | 指定关键字进行搜索，比如文件、UDP、TCP |

## 1.22 fwwhois 指令：显示用户的信息

 执行范围

## ◎ 范例 41 ◎ 显示信号。

```
[root@localhost ~]# fuser -l //显示信号
HUP INT QUIT ILL TRAP ABRT IOT BUS FPE KILL USR1 SEGV USR2 PIPE ALRM TERM
STKFLT CHLD CONT STOP TSTP TTIN TTOU URG XCPU XFSZ VTALRM PROF WINCH IO
PWR SYS
```

## UNUSED

```
[root@localhost ~]#
```

本例使用指令“fuser -l”显示信号。

## ◎ 范例 42 ◎ 显示进程。

```
[root@localhost littleProg]# fuser -a /root //显示与/root 目录相关的进程
```

```
/root: 4253c 4278c 4282c 4284c 4322c 4326c 4337
4365c 4372c 4374c 4376c 4385c 4389c 4391c 4411c 4414c 4415c
```

```
[root@localhost littleProg]#
```

本例使用指令“fuser -a /root”，显示所有与/root 目录相关的进程。

## ◎ 范例 43 ◎ 杀死进程。

```
[root@localhost littleProg]# fuser -k /root //杀死所有与/root 目录相关的进程
```

```
/root: 4253c 4278c 4282c 4284c 4322c 4326c 4337 4365c 4372c
4374c 4376c 4385c 4389c 4391c 4411c 4414c 4415c
```

```
[root@localhost littleProg]#
```

本例使用指令“fuser -k /root”杀死所有与/root 目录相关的进程。

 功能说明

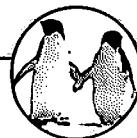
kill、killall、lsof、ps

**1.22****fwwhois 指令：显示用户的信息** 基础

fwwhois[用户名]

 功能说明

fwwhois 指令用来显示用户的信息。


● **执行范例**

- 范例 44 ○ 显示的用户信息。

```
[root@localhost root]# fwhois kk //显示的用户信息
```

本例使用指令“fwhois kk”，显示用户 kk 的信息。

● **相关指令**

whois

## 1.23

### gcov 指令：coverage 测试工具

● **语法**

gcov[必要参数][选择性参数]

● **功能说明**

gcov 指令是一个 coverage 测试工具。

● **参数说明**

必要参数的具体说明如下：

| 参 数       | 功 能                   |
|-----------|-----------------------|
| -coverage | 为 gcov 准备镜像           |
| -b        | 查看程序的分支数据             |
| -c        | 打印分支计数而不是分支频度         |
| -f        | 打印每一个函数的概要            |
| -n        | 创建长文件名                |
| -o        | .bb、.bbg 和.da 文件存放的目录 |

选择性参数的具体说明如下：

| 参 数       | 功 能  |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |

● **执行范例**

- 范例 45 ○ 显示文件的内容。

```
lifeng@lifang:~/cat bubblesort.c
```

```
#include <stdio.h>
```

## 1.23 gcov 指令：coverage 测试工具

```

void bubbleSort(int list[], int size)
{
 int i, j, temp, swap = 1;
 while (swap)
 {
 swap = 0;
 for (i = (size-1) ; i >= 0 ; i--) {
 for (j = 1 ; j <= i ; j++) {
 if (list[j-1] > list[j]) {
 temp = list[j-1];
 list[j-1] = list[j];
 list[j] = temp;
 swap = 1;
 }
 }
 }
 }
}

int main()
{
 int theList[10]={10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
 int i;
 /* Invoke the bubble sort algorithm */
 bubbleSort(theList, 10);
 /* Print out the final list */
 for (i = 0 ; i < 10 ; i++) {
 printf("%d\n", theList[i]);
 }
}

```

## ◎ 范例 46 ◎ 为 gcov 准备镜像。

lifeng@lifeng: ~ \$ gcc bubblesort.c -o bubblesort -fprofile-coverage  
 -fprofile-arcs //产生可执行文件

本例使用指令“gcc bubblesort.c -o bubblesort -fprofile-coverage -fprofile-arcs”产生镜像。

## ◎ 范例 47 ◎ 使用 gcov 程序。

lifeng@lifeng: ~ \$ ./bubblesort //运行 bubblesort.c  
 lifeng@lifeng: ~ \$ gcov bubblesort.c



本例使用指令**/bubblesort** 执行指令。

- 范例 48 ● 查看程序的分枝数据。

`lifeng@lifeng:~$ gcov -b bubblesort.c`

结果仍然在 `bubblesort.c.gcov` 文件中。



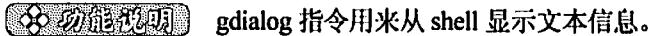
`gcc`

## 1.24

## gdialog 指令：从 shell 显示文本信息



`gdialog`



`gdialog` 指令用来从 shell 显示文本信息。



- 范例 49 ● 显示信息。

```
[root@localhost littleProg]# gdialog
```

```
cdialog (ComeOn Dialog!) version 1.0-20050306
```

```
Copyright (C) 2004 Thomas E. Dickey
```

```
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
* Display dialog boxes from shell scripts *
```

```
Usage: dialog <options> { --and-widget <options> }
```

```
where options are "common" options, followed by "box" options
```

```
Special options:
```

```
{--create-rc "file"}
```

```
Common options:
```

```
{--aspect <ratio>} [--backtitle <backtitle>} [--begin <y> <x>]}
```

```
{--cancel-label <str>} [--clear] [--colors] [--cr-wrap]}
```

```
{--default-item <str>} [--defaultno] [--exit-label <str>]}
```

```
{--extra-button} [--extra-label <str>} [--help-button]}
```

```
{--help-label <str>} [--help-status] [--ignore] [--input-fd <fd>]}
```

```
{--insecure} [--item-help] [--keep-window] [--max-input <n>]}
```

## 1.24 gdialog 指令：从 shell 显示文本信息

```
[--no-cancel] [--no-collapse] [--no-kill] [--no-label <str>]
[--no-shadow] [--ok-label <str>] [--output-fd <fd>] [--print-maxsize]
[--print-size] [--print-version] [--separate-output]
[--separate-widget <str>] [--shadow] [--single-quoted] [--size-err]
[--sleep <secs>] [--stderr] [--stdout] [--tab-correct] [--tab-len <n>]
[--timeout <secs>] [--title <title>] [--trim] [--visit-items]
[--version] [--yes-label <str>]
```

Box options:

```
--calendar <text> <height> <width> <day> <month> <year>
--checklist <text> <height> <width> <list height> <tag1> <item1>
<status1>...
--form <text> <height> <width> <form height> <label1> <l_y1>
<l_x1> <item1> <i_y1> <i_x1> <flen1> <ilen1>...
--fselect <filepath> <height> <width>
--gauge <text> <height> <width> [<percent>]
--infobox <text> <height> <width>
--inputbox <text> <height> <width> [<init>]
--inputmenu <text> <height> <width> <menu height> <tag1> <item1>...
--menu <text> <height> <width> <menu height> <tag1> <item1>...
--msgbox <text> <height> <width>
--passwordbox <text> <height> <width> [<init>]
--pause <text> <height> <width> <seconds>
--radiolist <text> <height> <width> <list height> <tag1> <item1>
```

<status1>...

```
--tailbox <file> <height> <width>
--tailboxbg <file> <height> <width>
--textbox <file> <height> <width>
--timebox <text> <height> <width> <hour> <minute> <second>
--yesno <text> <height> <width>
```

Auto-size with height and width = 0. Maximize with height and width = -1.

Global-auto-size if also menu\_height/list\_height = 0.

[root@localhost littleProg]#

本例使用指令 gdialog 从 shell 显示文本信息。


**相关指令**

kdialog、Xdialog、tk、wish

**1.25**
**gitps 指令：显示程序情况**
**语法** gitps[必要参数][选择性参数][终端][用户]

**功能说明** gitps 指令用来显示进程的状况，具有可视化界面，更有利于操作。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功 能        |
|----|------------|
| a  | 显示所有的进程信息  |
| c  | 只显示进程的真实名称 |
| e  | 显示环境变量     |
| f  | 显示进程间的关系   |
| n  | 用数字表示      |
| v  | 用虚拟内存的形式显示 |
| x  | 不区分终端      |

选择性参数的具体说明如下：

| 参数     | 功 能  |
|--------|------|
| U<用户>  | 指定用户 |
| p<进程号> | 指定进程 |
| t<终端>  | 指定终端 |

**执行范例**
**范例 50** 显示指定用户的进程信息。

```
[root@localhost littleProg]#gitps kk //显示进程信息
```

本例使用指令“gitps kk”显示用户 kk 下的进程信息。



ps

## 1.26

### groupadd 指令：创建一个新的群组



groupadd[选择性参数][必要参数]群组



groupadd 指令用来创建一个新的群组。



必要参数的具体说明如下：

| 参数 | 功能          |
|----|-------------|
| -f | 强制创建        |
| -o | 可以使用重复的 GID |

选择性参数的具体说明如下：

| 参数          | 功能      |
|-------------|---------|
| -gid<群组号>   | 指定群组号   |
| -key<关键字=值> | 指定关键字的值 |



◎ 范例 51 ◎ 创建一个群组。

```
[root@localhost ~]# groupadd kdjflk //创建一个群组
[root@localhost ~]#
```

本例使用指令“groupadd kdjflk”创建一个群组 kdjflk。



adduser、groupdel

## 1.27

### groupdel 指令：删除一个群组



groupdel[群组]



**功能说明** groupdel 指令用来改变系统的账户文件，删除指定用户的入口。指定的群组必须存在。

**执行范例**

- ◎ 范例 52 ◎ 删 除群组。

```
[root@localhost ~]# groupdel kdjflk //删除群组
[root@localhost ~]#
```

本例使用指令“groupdel kdjflk”删除群组 kdjflk。

**相关指令**

groupadd

## 1.28

## groupmod 指令：改变系统群组的属性

**语法**

groupmod[选择性参数][必要参数][群组]

**功能说明**

groupmod 指令用来改变指定群组的属性。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能        |
|-----|------------|
| -o  | 可以使用相同的群组名 |

选择性参数的具体说明如下：

| 参 数       | 功 能   |
|-----------|-------|
| -gid<群组号> | 指定群组号 |
| -n<群组名>   | 指定群组名 |

**执行范例**

- ◎ 范例 53 ◎ 改变群组名的属性。

```
root@localhost:~# groupmod -n jj kk //改变群组名
root@localhost:~#
```

本例使用指令“groupmod -n jj kk”将群组名 kk 改为 jj。

**相关指令**

groupadd、groupdel

**1.29****halt 指令：关闭系统****语法** halt[必要参数]

**功能说明** halt 指令用来关闭系统，然后再通知内核是停止、重启还是关闭电源。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能                    |
|----|-----------------------|
| -d | 不在 wtmp 中记录           |
| -f | 强制关机或者重启，不调用 shutdown |
| -h | 让硬件保持 standby 状态      |
| -i | 关闭系统前先关闭网络系统          |
| -n | 不执行 sync              |
| -p | halt 后执行 poweroff     |
| -w | 在 wtmp 中记录，不关闭系统      |

**执行范例**

## ◎ 范例 54 ◎ 关闭系统。

```
[root@localhost ~]# halt //关闭系统
```

本例使用指令 halt，关闭系统。

## ◎ 范例 55 ◎ 关闭系统并关闭电源。

```
[root@localhost ~]# halt -p //关闭系统并关闭电源
```

本例使用指令 halt 关闭系统并关闭电源。

**相关命令**

poweroff、shutdown



## 1.30

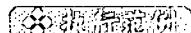
# help 指令：显示 shell 的内建指令的帮助信息



**help [指令]**



help 指令用于显示 shell 的内建指令的帮助信息。



○ 范例 56 ○ 显示指令帮助。

```
[root@localhost shal]# help hash //显示hash指令的帮助信息
```

```
hash: hash [-lr] [-p pathname] [-dt] {name ...}
```

For each NAME, the full pathname of the command is determined and remembered. If the -p option is supplied, PATHNAME is used as the full pathname of NAME, and no path search is performed. The -r option causes the shell to forget all remembered locations. The -d option causes the shell to forget the remembered location of each NAME. If the -t option is supplied the full pathname to which each NAME corresponds is printed. If multiple NAME arguments are supplied with -t, the NAME is printed before the hashed full pathname. The -l option causes output to be displayed in a format that may be reused as input. If no arguments are given, information about remembered commands is displayed.

```
[root@localhost shal]#
```

本例使用指令“help hash”显示 hash 指令的帮助信息。

○ 范例 57 ○ 显示所有支持的帮助信息。

```
[root@localhost shal]# help //显示所有支持的帮助信息
```

```
GNU bash, version 3.00.16(1)-release (i386-redhat-linux-gnu)
```

```
These shell commands are defined internally. Type 'help' to see this list.
```

```
Type 'help name' to find out more about the function 'name'.
```

```
Use 'info bash' to find out more about the shell in general.
```

```
Use 'man -k' or 'info' to find out more about commands not in this list.
```

```
A star (*) next to a name means that the command is disabled.
```

## 1.30 help 指令：显示 shell 的内建指令的帮助信息

```

%[DIGITS | WORD] [&] ((expression))
. filename [arguments] :
[arg...] [[expression]]
alias [-p] [name[=value] ...] bg [job_spec]
bind [-lpvsPVS] [-m keymap] [-f fi break [n]]
builtin [shell-builtin [arg ...]] caller [EXPR]
case WORD in [PATTERN [| PATTERN]. cd [-L|-P] [dir]
command [-pVv] command [arg ...] compgen [-abcdefgjksuv] [-o option
complete [-abcdefgjksuv] [-pr] [-o continue [n]
declare [-afFirstx] [-p] [name[=val dirs [-clpv] [+N] [-N]
disown [-h] [-ar] [jobspec ...] echo [-neE] [arg ...]
enable [-pnds] [-a] [-f filename] eval [arg ...]
exec [-cl] [-a name] file {redirec exit [n]
export [-nf] [name[=value] ...] or false
fc [-e ename] [-nir] {first} {last fg [job_spec]
for NAME [in WORDS ... ;] do COMMA for ((expl; exp2; exp3)); do COM
function NAME { COMMANDS ; } or NA getopt optstring name [arg]
hash [-lr] [-p pathname] [-dt] [na help [-s] [pattern ...]
history [-c] [-d offset] [n] or hi if COMMANDS; then COMMANDS; [elif
jobs [-lnprs] [jobspec ...] or job kill [-s sigspec | -n signum | -si
let arg [arg ...] local name[=value] ...
logout popd [+N | -N] [-n]
printf format [arguments] pushd [dir | +N | -N] [-n]
pwd [-PL] read [-ers] [-u fd] [-t timeout] [
readonly [-af] [name[=value] ...] return [n]
select NAME [in WORDS ... ;] do CO set [--abefhkmnptuvxBCHP] [-o opti
shift [n] shopt [-pqsu] [-o long-option] opt
source filename [arguments] suspend [-f]
test [expr] time [-p] PIPELINE
times trap [-lp] [arg signal_spec ...]
true type [-afptP] name [name ...]
typeset [-affFirstx] [-p] name[=valu ulimit (-SHacdflmnptuv) [limit]

```



```
umask [-p] [-S] [mode] unalias [-a] name [name ...]
unset [-f] [-v] [name ...] until COMMANDS; do COMMANDS; done
variables - Some variable names an wait [n]
while COMMANDS; do COMMANDS; done { COMMANDS ; }
[root@localhost shal]#
```

本例使用指令 help 显示所有支持的帮助信息。



man、info

## 1.31 history 指令：显示历史指令



**语法** history[必要参数][选择性参数][文件名]

**功能说明** 使用 history 指令可以显示曾经执行过的指令，也可以根据显示的指令重新执行需要的指令。



必要参数的具体说明如下：

| 参 数       | 功 能              |
|-----------|------------------|
| <b>-n</b> | 显示 n 个最近的指令记录    |
| <b>-a</b> | 添加记录             |
| <b>-r</b> | 读取记录，但不会添加内容到记录  |
| <b>-w</b> | 覆盖原有的 history 文件 |
| <b>-c</b> | 清除记录             |

选择性参数的具体说明如下：

| 参 数                 | 功 能      |
|---------------------|----------|
| <b>-d&lt;编号&gt;</b> | 删除指定编号指令 |
| <b>-n&lt;文件&gt;</b> | 读取指定文件   |
| <b>-r&lt;文件&gt;</b> | 读取文件但不记录 |
| <b>-w&lt;文件&gt;</b> | 覆盖原有文件   |

**○执行范例**

- ◎范例 58 ○ 显示最近执行的 n 个指令。

```
[root@localhost shal]# history 20 //显示最近执行的 20 个指令
1073 ls
1074 vi sha.cpp
1075 head sha.cpp
1076 head -n 20 sha.cpp
1077 head -c 100 sha.cpp
1078 head -c 1000 sha.cpp
1079 head -c 300 sha.cpp
1080 clear
1081 head -c 300 sha.cpp
1082 man help
1083 help --help
1084 help ls
1085 help exit
1086 help head
1087 help hash
1088 helpo
1089 help
1090 man history
1091 history
1092 history 20
[root@localhost shal]#
```

本例使用指令 history 显示最近执行的 20 个指令。

- ◎范例 59 ○ 执行 history 显示的指令。

```
[root@localhost shal]# !1087 //执行 1087 号指令
help hash
hash: hash [-lr] [-p pathname] [-dt] [name ...]
For each NAME, the full pathname of the command is determined and
remembered. If the -p option is supplied, PATHNAME is used as the
full pathname of NAME, and no path search is performed. The -r
option causes the shell to forget all remembered locations. The -d
```

option causes the shell to forget the remembered location of each NAME. If the -t option is supplied the full pathname to which each NAME corresponds is printed. If multiple NAME arguments are supplied with -t, the NAME is printed before the hashed full pathname. The -l option causes output to be displayed in a format that may be reused as input. If no arguments are given, information about remembered commands is displayed.

[root@localhost shal]#

本例使用指令“!1087”，执行 history 中的 1087 号指令，即“help hash”，执行情况如上所示。

#### ◎ 范例 60 ◎ 清除指令历史记录。

```
[root@localhost shal]# history -c //清除指令历史记录
[root@localhost shal]# history
95 history
[root@localhost shal]#
```

本例使用指令“history -c”清除指令历史记录。



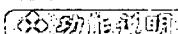
/root/.bash\_history 文件记录了 root 用户的历史记录。

## 1.32

## htpasswd 指令：创建和更新用户的认证文件



**htpasswd[必要参数][选择性参数][密码文件][用户名]**



htpasswd 指令用来创建和更新保存 HTTP 用户的用户名和密码的文件，如果 htpasswd 不能读取输出的文件，它将返回错误信息。



必要参数的具体说明如下：

| 参 数 | 功 能             |
|-----|-----------------|
| -c  | 创建一个文件          |
| -n  | 不更新文件，显示结果到标准输出 |

## 1.32 htpasswd 指令：创建和更新用户的认证文件

续表

| 参数 | 功能                      |
|----|-------------------------|
| -m | 强制执行密码的 MD5 加密          |
| -d | 强制执行密码 CRYPT 的加密，这是默认设置 |
| -p | 不加密密码                   |
| -s | 强制执行密码的 SHA 加密          |
| -b | 从指令行输入密码                |
| -D | 删除指定用户                  |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -h | 帮助信息 |


**技巧范例**

◎ 范例 61 ◎ 创建一个 apache 登录用户 kk。

```
[root@localhost shal]# htpasswd -c /home/kk/.htpasswd kk //创建一个新的
用户
```

```
New password:
Re-type new password:
Adding password for user kk
[root@localhost shal]#
```

本例使用指令“htpasswd -c /home/kk/.htpasswd kk”创建一个新的 apache 登录用户。

◎ 范例 62 ◎ 对密码进行 MD5 加密。

```
[root@localhost shal]# htpasswd -m /home/kk/.htpasswd kk //加密密码
New password:
Re-type new password:
Updating password for user kk
[root@localhost shal]#
```

本例使用指令“htpasswd -m /home/kk/.htpasswd kk”对密码进行 MD5 加密。


相关指令

apachectl

## 1.33

# id 指令：显示用户的 ID 以及所属群组的 ID

语法 id[必要参数][选择性参数][用户名]

功能说明 id 指令用来打印出指定用户名的信息或者当前用户名的信息。

参数说明

必要参数的具体说明如下：

| 参数 | 功能               |
|----|------------------|
| -a | 为了兼容，忽略信息        |
| -Z | 仅输出文本            |
| -g | 输出有效的群组 ID       |
| -G | 输出所有的群组 ID       |
| -n | 输出名字而不是数字        |
| -r | 输出真实 ID 而不是有效 ID |
| -u | 输出有效用户 ID        |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

执行范例

◎ 范例 63 ◎ 显示当前用户信息。

```
[root@localhost ~]# id //显示当前用户信息
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),
4(adm),6(disk),10(wheel) context=root:system_r:unconfined_t
[root@localhost ~]#
```

## 1.34 info 指令：读取目录信息

本例使用指令 id 显示当前用户的信息。

◎ 范例 64 ◎ 显示用户群组 ID。

```
[root@localhost ~]# id -g //显示用户群组 ID
```

0

```
[root@localhost ~]#
```

本例使用指令“id -g”显示用户群组 ID。

◎ 范例 65 ◎ 显示所有群组 ID。

```
[root@localhost ~]# id -G //显示所有群组 ID
```

0 1 2 3 4 6 10

```
[root@localhost ~]#
```

本例使用指令“id -G”显示所有群组的 ID。

◎ 范例 66 ◎ 显示指定用户的信息。

```
[root@localhost ~]# id kk //显示用户 kk 的信息
```

uid=500(kk) gid=500(kk) groups=500(kk)

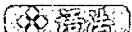
```
[root@localhost ~]#
```

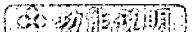
本例使用指令“id kk”显示用户 kk 的信息。

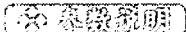
 相关指令

hostname、uname

## 1.34 info 指令：读取目录信息

 备注 info[必要参数][选择性参数]

 功能说明 info 指令将以 info 的格式读取文档。

 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能       |
|-----|-----------|
| -d  | 指定搜索路径    |
| -f  | 指定访问的帮助文件 |
| -n  | 指定节点      |

选择性参数的具体说明如下：



| 参数       | 功能    |
|----------|-------|
| -o<文件>   | 输出到文件 |
| -help    | 帮助信息  |
| -version | 版本信息  |

### ◎ 实例与操作

#### ○ 范例 67 ○ 以 info 的格式查看帮助信息。

```
[root@localhost ~]# info init >1.log //显示init的帮助信息
[root@localhost ~]# cat 1.log //显示文件1.log
File: libc.info, Node: Setting Groups, Next: Enable/Disable Setuid,
Prev: Setting User ID, Up: Users and Groups
29.7 Setting the Group IDs

```

This section describes the functions for altering the group IDs (real and effective) of a process. To use these facilities, you must include the header files 'sys/types.h' and 'unistd.h'.

-- Function: int setegid (gid\_t NEWGID)

This function sets the effective group ID of the process to NEWGID, provided that the process is allowed to change its group ID. Just as with 'seteuid', if the process is privileged it may change its effective group ID to any value; if it isn't, but it has a file group ID, then it may change to its real group ID or file group ID; otherwise it may not change its effective group ID. Note that a process is only privileged if its effective \_user\_ ID is zero. The effective group ID only affects access permissions. The return values and error conditions for 'setegid' are the same as those for 'seteuid'.

This function is only present if '\_POSIX\_SAVED\_IDS' is defined.

The calling process is not privileged.

-- Function: int initgroups (const char \*USER, gid\_t GROUP)

The 'initgroups' function sets the process's supplementary group IDs to be the normal default for the user name USER. The group GROUP is automatically included.

This function works by scanning the group database for all the groups USER belongs to. It then calls 'setgroups' with the list it

## 1.34 info 指令：读取目录信息

has constructed.

The return values and error conditions are the same as for 'setgroups'.

If you are interested in the groups a particular user belongs to, but do not want to change the process's supplementary group IDs, you can use 'getgrouplist'. To use 'getgrouplist', your programs should include the header file 'grp.h'.

```
-- Function: int getgrouplist (const char *USER, gid_t GROUP, gid_t
 *GROUPS, int *NGROUPS)
```

The 'getgrouplist' function scans the group database for all the groups USER belongs to. Up to \*NGROUPS group IDs corresponding to these groups are stored in the array GROUPS; the return value from the function is the number of group IDs actually stored. If \*NGROUPS is smaller than the total number of groups found, then 'getgrouplist' returns a value of '-1' and stores the actual number of groups in \*NGROUPS. The group GROUP is automatically included in the list of groups returned by 'getgrouplist'. Here's how to use 'getgrouplist' to read all supplementary groups for USER:

```
gid_t *
supplementary_groups (char *user)
{
 int ngroups = 16;
 gid_t *groups
 = (gid_t *) xmalloc (ngroups * sizeof (gid_t));
 struct passwd *pw = getpwnam (user);
 if (pw == NULL)
 return NULL;
 if (getgrouplist (pw->pw_name, pw->pw_gid, groups, &ngroups) < 0)
 {
 groups = xrealloc (ngroups * sizeof (gid_t));
 getgrouplist (pw->pw_name, pw->pw_gid, groups, &ngroups);
 }
 return groups;
}
[root@localhost ~]#
```

本例使用指令“info init”，显示指令 init 的帮助信息。中间内容较多，



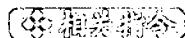
部分已删除，不影响本例的阅读。

◎ 范例 68 ◎ 查询 info 的具体使用方法。

```
[root@localhost ~]#info info //显示info的具体使用方法
```

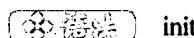
【info 文件的操作说明】

- h: info 帮助信息。
- 空格: 下一页。
- [Del] 键: 上一页。
- b: 开始处。
- e: 结束处。
- p: 上一个主题。
- n: 下一个主题。
- m: 菜单。
- Ctrl+L: 重新显示画面。
- q: 退出。
- ?: 帮助。



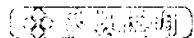
man

## 1.35 ·init 指令：开关机设置



【全功能】 init[必要参数][选择性参数]

【功能说明】 init 是所有进程的父进程，它的主要任务是根据存储在/etc/inittab 文件中的脚本创建进程。



必要参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| 0  | 停止系统 |
| 6  | 重启系统 |

## 1.36 ipcs 指令：显示进程间通信的信息

续表

| 参数 | 功能                  |
|----|---------------------|
| -l | 关闭系统进入单用户模式         |
| -S | 单用户模式               |
| -b | 直接进入单用户模式而不运行任何启动脚本 |
| -a | 自动设置                |

选择性参数的具体说明如下：

| 参数     | 功能             |
|--------|----------------|
| -z XXX | 设置给“-z”的参数将被忽略 |

### ◎ 技术案例

○ 范例 69 ○ 关机。

```
[root@localhost ~]#init 0 //关机
```

本例使用指令“init 0”关闭主机。关闭计算机的其他指令实际上也是通过调用 init 指令来实现的。

○ 范例 70 ○ 重启。

```
[root@localhost ~]#init 6 //重启
```

本例使用指令“init 6”重启计算机。

### ◎ 相关命令

getty、login、sh、runlevel、shutdown、kill、inittab、initscript、utmp、reboot、shutdown、poweroff

## 1.36 ipcs 指令：显示进程间通信的信息

### ◎ 基本 ipcs[源参数][输出参数]

○ 功能说明 ○ ipcs 指令将对进程通信进行描述。如果不指定参数，那么输出的是当前系统中活跃的消息队列、共享内存段和标志信息设置。

### ○ 参数说明

源参数的具体说明如下：



| 参数 | 功能   |
|----|------|
| -m | 共享内存 |
| -q | 消息队列 |
| -s | 旗语   |
| -a | 全部   |

输出参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -t | 时间   |
| -p | 进程ID |
| -c | 创建   |
| -l | 限制   |
| -u | 总结   |

### 操作范例

#### ◎ 范例 71 ◎ 显示进程间通信的状态。

```
[root@localhost drivers]# ipcs //显示进程间通信的状态
----- Shared Memory Segments -----
key shmid owner perms bytes nattch status
0x00000000 32768 gdm 600 393216 2 dest
----- Semaphore Arrays -----
key semid owner perms nsems
----- Message Queues -----
key msqid owner perms used-bytes messages
[root@localhost drivers]#
```

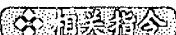
本例使用指令 ipcs 显示进程间通信的状态。

#### ◎ 范例 72 ◎ 显示共享内存的时间信息。

```
[root@localhost drivers]# ipcs -m -t //显示共享内存的时间信息
----- Shared Memory Attach/Detach/Change Times -----
shmid owner attached detached changed
32768 gdm May 23 14:11:23 Not set May 23 14:11:23
[root@localhost drivers]#
```

## 1.37 ipcrm 指令：删除消息队列、旗语设置或者共享内存的 ID

本例使用指令“ipcs -m -t”显示共享内存的时间信息。



ipcrm

1.37

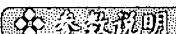
## ipcrm 指令：删除消息队列、旗语设置或者共享内存的 ID



ipcrm[选择性参数]



ipcrm 指令用来删除消息队列、旗语设置或者共享内存段。需要删除的进程号将用参数指定。



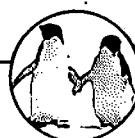
选择性参数的具体说明如下：

| 参数       | 功能           |
|----------|--------------|
| -m<进程号>  | 删除指定进程号的共享内存 |
| msg<进程号> | 删除指定进程号的消息队列 |
| -q<进程号>  | 删除指定进程号的消息队列 |
| -s<进程号>  | 删除指定进程号的旗语   |
| shm<进程号> | 删除指定进程号的共享内存 |
| sem<进程号> | 删除指定进程号的旗语   |



◎ 范例 73 ◎ 显示进程间的通信信息。

```
[root@localhost drivers]# ipcs //显示进程间的通信信息
----- Shared Memory Segments -----
key shmid owner perms bytes nattch status
0x00000000 32768 gdm 600 393216 2 dest
----- Semaphore Arrays -----
key semid owner perms nsems
----- Message Queues -----
key msgid owner perms used-bytes messages
[root@localhost drivers]#
```

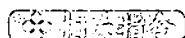


本例使用指令 ipcs 显示进程间的通信信息。

○ 范例 74 ○ 删 除 指 定 进 程 号 的 共 享 内 存。

```
[root@localhost drivers]# ipcrm -m 32768 //删除进程32768的共享内存
[root@localhost drivers]#
```

本例使用指令“ipcrm -m 32768”删除进程 32768 的共享内存。



ipcs

## 1.38 jobs 指令：列出所有后台程序

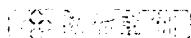
jobs[必要参数][任务]

jobs 指令显示当前正在后台执行的任务，得到相关的信息之后，可以对任务进行进一步操作，如使用 fg 指令将任务调到前台运行，或者使用 kill 指令结束任务。



必要参数的具体说明如下：

| 参 数 | 功 能          |
|-----|--------------|
| -p  | 列出进程 ID      |
| -n  | 列出发生变化的进程 ID |
| -l  | 列出后台进程的所有信息  |



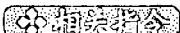
○ 范例 75 ○ 列出后台进程。

```
root@localhost:~# ftp & //后台运行
[1] 29163
root@localhost:~# jobs //列出后台进程
[1]+ Stopped ftp
root@localhost:~# jobs -l //列出后台进程的详细信息
[1]+ 29163 Stopped (tty output) ftp
root@localhost:~# jobs -p //列出后台进程 ID
29163
```

## 1.39 kill 指令：杀死执行中的进程

```
root@localhost:~# jobs -n //列出后台进程的变化信息
root@localhost:~#
```

本例使用指令 `jobs` 列出后台运行的进程，然后使用指令“`jobs -l`”列出后台运行的进程的详细信息。使用指令“`jobs -l`”，仅列出后台运行的进程的 ID，使用指令“`jobs -n`”可以列出后台进程的变化信息。



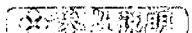
`kill`、`fg`

## 1.39

### kill 指令：杀死执行中的进程

`kill`[必要参数][选择性参数][进程号]

`kill` 指令发送指定的信号到相应进程。不指定信号时将发送 SIGTERM(15)终止指定进程。若仍无法终止该程序可用“-KILL”参数，其发送的信号为 SIGKILL(9)，将强制结束进程。使用 `ps` 指令或 `jobs` 指令可以查看进程号。`root` 用户可以影响任何用户的进程，非 `root` 用户只能影响自己的进程。



必要参数的具体说明如下：

| 参 数 | 功 能                                  |
|-----|--------------------------------------|
| -l  | <信号编号>，若不加信号编号参数，则使用“-l”参数会列出全部的信号名称 |

选择性参数的具体说明如下：

| 参 数 | 功 能                                         |
|-----|---------------------------------------------|
| -a  | 当处理当前进程时，不限制指令名和进程号的对应关系                    |
| -p  | 指定 <code>kill</code> 指令只打印相关进程的进程号，而不发送任何信号 |
| -s  | 指定发送信号                                      |



◎ 范例 76 ◎ 杀死进程 12345



\$ kill 12345

- 范例 77 ○ 当用 kill 指令不能杀死相关进程时, 可以使用“-KILL”参数强制杀死相应进程。

\$ kill -KILL 12345

上述指令将强制杀死进程, 但系统中被进程占用的某些资源可能尚未释放, 或者可能会留下文件。

- 范例 78 ○ 发送 SIGHUP 信号, 可以用下面的指令:

\$ kill -HUP pid

- 范例 79 ○ “kill -l”命令列出 kill 所能传送的所有信号。

\$ kill -l

|                 |                 |                 |                 |
|-----------------|-----------------|-----------------|-----------------|
| 1) SIGHUP       | 2) SIGINT       | 3) SIGQUIT      | 4) SIGILL       |
| 5) SIGTRAP      | 6) SIGABRT      | 7) SIGBUS       | 8) SIGFPE       |
| 9) SIGKILL      | 10) SIGUSR1     | 11) SIGSEGV     | 12) SIGUSR2     |
| 13) SIGPIPE     | 14) SIGALRM     | 15) SIGTERM     | 17) SIGCHLD     |
| 18) SIGCONT     | 19) SIGSTOP     | 20) SIGTSTP     | 21) SIGTTIN     |
| 22) SIGTTOU     | 23) SIGURG      | 24) SIGXCPU     | 25) SIGXFSZ     |
| 26) SIGVTALRM   | 27) SIGPROF     | 28) SIGWINCH    | 29) SIGIO       |
| 30) SIGPWR      | 31) SIGSYS      | 33) SIGRTMIN    | 34) SIGRTMIN+1  |
| 35) SIGRTMIN+2  | 36) SIGRTMIN+3  | 37) SIGRTMIN+4  | 38) SIGRTMIN+5  |
| 39) SIGRTMIN+6  | 40) SIGRTMIN+7  | 41) SIGRTMIN+8  | 42) SIGRTMIN+9  |
| 43) SIGRTMIN+10 | 44) SIGRTMIN+11 | 45) SIGRTMIN+12 | 46) SIGRTMIN+13 |
| 47) SIGRTMIN+14 | 48) SIGRTMIN+15 | 49) SIGRTMAX-14 | 50) SIGRTMAX-13 |
| 51) SIGRTMAX-12 | 52) SIGRTMAX-11 | 53) SIGRTMAX-10 | 54) SIGRTMAX-9  |
| 55) SIGRTMAX-8  | 56) SIGRTMAX-7  | 57) SIGRTMAX-6  | 58) SIGRTMAX-5  |
| 59) SIGRTMAX-4  | 60) SIGRTMAX-3  | 61) SIGRTMAX-2  | 62) SIGRTMAX-1  |
| 63) SIGRTMAX    |                 |                 |                 |

signal、killall

## 1.40

### killall 指令: 杀死同名的所有进程

killall[必要参数][选择性参数][指令名]

killall 指令用来结束同名的所有进程。

## 1.40 killall 指令：杀死同名的所有进程



必要参数的具体说明如下：

| 参 数 | 功 能                |
|-----|--------------------|
| -Z  | 只杀死拥有 scontext 的进程 |
| -e  | 要求精确匹配进程名称         |
| -I  | 忽略大小写              |
| -g  | 杀死进程组而不是进程         |
| -i  | 交互模式，杀死进程前先询问用户    |
| -l  | 列出所有的已知信号名称        |
| -q  | 不输出警告信息            |
| -s  | 发送指定的信号            |
| -v  | 报告信号是否成功发送         |
| -w  | 等待进程死亡             |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -version | 版本信息 |
| -help    | 帮助信息 |



◎ 范例 80 ◎ 杀死同名的所有进程。

```
[root@localhost ~]# ps -A //显示所有的进程
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:00 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
```

//此处省略了部分结果



```

3355 ? 00:00:17 gnome-terminal
3357 ? 00:00:00 gnome-pty-help
3358 pts/2 00:00:00 bash
3377 pts/2 00:00:03 joe
6734 pts/1 00:00:00 ps
[root@localhost ~]# killall joe //杀死同名的所有进程
[root@localhost ~]# killall joe //杀死同名的所有进程
joe: no process killed
[root@localhost ~]# ps -A //显示进程
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:00 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
 61 ? 00:00:00 kblockd/0
 //此处省略了部分结果
3345 ? 00:00:00 notification-ar
3347 ? 00:00:03 clock-applet
3349 ? 00:00:36 mixer_applet2
3355 ? 00:00:17 gnome-terminal
3357 ? 00:00:00 gnome-pty-help
3358 pts/2 00:00:00 bash
6739 pts/1 00:00:00 ps
[root@localhost ~]#

```

本例使用指令“ps -A”显示当前系统正在运行的所有进程，其中有一个正在运行的进程的名称为“3377 pts/2 00:00:03 joe”。本例使用指令“killall joe”，杀死所有的同名进程，然后在运行指令“ps -A”，发现joe指令已被杀死。

#### ◎ 范例 81 ◎ 向进程发送指定信号。

```

[root@localhost ~]# kill -l //显示所有的信号
 1) SIGHUP 2) SIGINT 3) SIGQUIT 4) SIGILL
 5) SIGTRAP 6) SIGABRT 7) SIGBUS 8) SIGFPE
 9) SIGKILL 10) SIGUSR1 11) SIGSEGV 12) SIGUSR2
13) SIGPIPE 14) SIGALRM 15) SIGTERM 17) SIGCHLD

```

## 1.41 last 指令：列出目前与过去登录系统的用户相关信息

```

18) SIGCONT 19) SIGSTOP 20) SIGSTP 21) SIGTTIN
22) SIGTTOU 23) SIGURG 24) SIGXCPU 25) SIGXFSZ
26) SIGVTALRM 27) SIGPROF 28) SIGWINCH 29) SIGIO
30) SIGPWR 31) SIGSYS 34) SIGRTMIN 35) SIGRTMIN+1
36) SIGRTMIN+2 37) SIGRTMIN+3 38) SIGRTMIN+4 39) SIGRTMIN+5
40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8 43) SIGRTMIN+9
44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13
52) SIGRTMAX-12 53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9
56) SIGRTMAX-8 57) SIGRTMAX-7 58) SIGRTMAX-6 59) SIGRTMAX-5
60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2 63) SIGRTMAX-1
64) SIGRTMAX

```

```

[root@localhost ~]# joe & //后台运行 joe 指令
[2] 6806
[root@localhost ~]# Processing '/etc/joe/joerc'...done
Processing '/etc/joe/joerc'...done

```

```

[2]+ Stopped joe
[root@localhost ~]# killall -TERM joe //杀死 joe 指令
[root@localhost ~]#

```

本例使用指令“kill -l”显示所有的信号信息，使用指令“killall -TERM joe”杀死所有名为 joe 的指令。

**杀**

## 1.41

## last 指令：列出目前与过去登录系统的用户相关信息

**杀**

last[必要参数]

**杀** last 指令查找 /var/log/wtmp 文件，并显示在此文件创建后的所有登录用户。可以指定数字或者时间来显示用户的登录情况。

**杀**

必要参数的具体说明如下：



| 参数 | 功能                        |
|----|---------------------------|
| -n | 显示最近登录信息的行数               |
| -t | 根据登录的时间显示指定用户的登录状态        |
| -R | 简略显示主机名区域                 |
| -a | 在最后一列显示主机名称               |
| -d | 对于非本地登录的用户，将显示的IP地址转换为主机名 |
| -i | 类似于“-d”参数，将显示的主机名转换为IP地址  |
| -o | 读取旧的类型的 wtmp 文件           |
| -x | 显示系统的登录等级                 |

选择性参数的具体说明如下：

| 参数        | 功能   |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |

### 执行范例

#### ◎ 范例 82 ◎ 一般的显示方法。

```
[root@wjc root]# last
root pts/4 :0.0 Tue May 22 05:12 still logged in
root pts/3 :0.0 Tue May 22 05:02 still logged in
root pts/2 :0.0 Tue May 22 04:51 still logged in
root pts/0 :0.0 Tue May 22 04:41 still logged in
root pts/1 :0.0 Tue May 22 04:13 still logged in
root pts/0 :0.0 Tue May 22 04:11 - 04:33 (00:22)
root :0 Tue May 22 04:10 still logged in
reboot system boot 2.4.20-8 Tue May 22 04:07 (19:57)
root pts/1 :0.0 Tue May 22 01:09 - down (02:56)
root pts/0 :0.0 Tue May 22 00:58 - down (03:07)
root :0 Tue May 22 00:58 - down (03:07)
reboot system boot 2.4.20-8 Tue May 22 00:52 (03:13)
root pts/4 :0.0 Tue May 22 00:18 - down (00:33)
root pts/3 :0.0 Mon May 21 23:58 - down (00:53)
root pts/2 :0.0 Mon May 21 23:24 - down (01:27)
```

## 1.42 lastb 指令：列出登录系统失败的用户相关信息

```
root pts/1 :0.0 Mon May 21 23:16 - down (01:34)
root pts/0 :0.0 Mon May 21 23:13 - down (01:37)
root :0 Mon May 21 23:13 - down (01:38)
reboot system boot 2.4.20-8 Mon May 21 06:30 (18:20)
root pts/3 :0.0 Mon May 21 05:47 - down (00:42)
wtmp begins Mon May 21 05:47:02 2007
```

◎ 范例 83 ◎ 简略显示，并指定显示的个数。

```
[root@wjc root]# last -n 5 -R
root pts/5 Tue May 22 23:32 still logged in
root pts/4 Tue May 22 05:12 still logged in
root pts/3 Tue May 22 05:02 still logged in
root pts/2 Tue May 22 04:51 still logged in
root pts/0 Tue May 22 04:41 still logged in
wtmp begins Mon May 21 05:47:02 2007
```

◎ 范例 84 ◎ 最后一列显示主机 IP 地址。

```
[root@wjc root]# last -n 5 -a -i
root pts/5 Tue May 22 23:32 still logged in 192.168.0.1
root pts/4 Tue May 22 05:12 still logged in 0.0.0.0
root pts/3 Tue May 22 05:02 still logged in 0.0.0.0
root pts/2 Tue May 22 04:51 still logged in 0.0.0.0
root pts/0 Tue May 22 04:41 still logged in 0.0.0.0
```



shutdown、login、init

## 1.42

## lastb 指令：列出登录系统失败的用户相关信息



**lastb [必要参数]**



与 last 指令类似，lastb 指令查找/var/log/btmp 文件，此文件列出了所有登录系统失败的用户的信息。



必要参数的具体说明如下：

| 参 数 | 功 能         |
|-----|-------------|
| -n  | 显示最近登录信息的行数 |



续表

| 参数 | 功能                        |
|----|---------------------------|
| -t | 根据登录的时间显示指定用户的登录状态        |
| -R | 简略显示主机名区域                 |
| -a | 在最后一列显示主机名称               |
| -d | 对于非本地登录的用户，将显示的IP地址转换为主机名 |
| -i | 类似于“-d”选项，将显示的主机名转换为IP地址  |
| -o | 读取旧的类型的 wtmp 文件           |
| -x | 显示系统的登录等级                 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### (☆热启动)

○范例 85 ○ 显示登录失败用户。

```
[root@wjc root]# lastb
lastb: /var/log/btmp: No such file or directory
Perhaps this file was removed by the operator to prevent logging lastb
info.
```

### (☆系统命令)

last

# 1.43

## login 指令：登录系统

(☆热启动) login

(☆热启动) login 指令使用户放弃现在使用的身份，重新登录系统。也可在后面加入用户名，再直接输入密码，登录系统。

## 1.45 logrotate 指令：处理 log 文件



- ◎ 范例 86 ◎ 用户使用新的身份重新登录系统。

[root@wjc backup]# login



logout

## 1.44

## logname 指令：显示登录账号的信息



logname[选择性参数]



logname 指令用来显示登录账号的信息。



选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



- ◎ 范例 87 ◎ 显示登录账号的信息。

root@localhost:~# logname //显示登录账号的信息

root

root@localhost:~#

本例使用指令 logname 显示登录账号的信息。



login

## 1.45

## logrotate 指令：处理 log 文件



logrotate[必要参数][选择性参数][文件]



logrotate 指令用来管理系统的 log 文件。



## 第1章 系统管理

### 功能说明

必要参数的具体说明如下：

| 参数 | 功 能       |
|----|-----------|
| -d | 显示调试信息    |
| -f | 强制显示      |
| -v | 显示详细的处理过程 |

选择性参数的具体说明如下：

| 参数     | 功 能           |
|--------|---------------|
| -usage | 显示用法          |
| -s<文件> | 使用指定的文件作为状态文件 |
| -usage | 显示用法          |

### 执行范例

- 范例 88 ○ 指定记录文件。

```
root@localhost:/home# logrotate /home/log.config
```

```
root@localhost:/home#
```

本例使用指令“logrotate /home/log.config”指定配置文件。

### 相关指令

gzip

## 1.46 logout 指令：退出系统

### 语法

logout

○ 功能说明 ○ 使用户退出系统，其功能和 login 指令相对应。

### 执行范例

- 范例 89 ○ 执行 logout 指令退出系统

```
[root@wjc backup]# logout
```

## 1.47 lsmod 指令：显示 Linux 内核的模块信息



login

**1.47****lsmod 指令：显示 Linux 内核的模块信息**

lsmod



**功能说明** lsmod 指令是一个小程序，用来显示文件 /proc/modules 的信息，也就是显示当前内核装载的模块。



◎ 范例 90 ◎ 显示当前系统下的模块信息。

```
[root@localhost ~]# lsmod //显示模块信息
Module Size Used by
parport_pc 28933 1
lp 13001 0
parport 40585 2 parport_pc,lp
autofs4 29253 2
rfcomm 42333 0
l2cap 30661 5 rfcomm
bluetooth 56133 4 rfcomm,l2cap
sunrpc 167813 1
dm_mod 58101 0
video 15941 0
button 6609 0
battery 9413 0
ac 4805 0
md5 4033 1
ipv6 268097 10
uhci_hcd 35152 0
shpchp 94405 0
i2c_piix4 8657 0
i2c_core 21569 1 i2c_piix4
snd_ens1371 31649 0
gameport 18633 1 snd_ens1371
snd_rawmidi 30305 1 snd_ens1371
snd_ac97_codec 75961 1 snd_ens1371
```



大赢家  
TOPSAGE.COM

```

snd_seq_dummy 3653 0
snd_seq_oss 37057 0
snd_seq_midi_event 9153 1 snd_seq_oss
snd_seq 62289 5 snd_seq_dummy,snd_seq_oss,snd_seq_midi_event
snd_seq_device 8781 4 snd_rawmidi,snd_seq_dummy,snd_seq_oss,snd_seq
snd_pcm_oss 51185 0
snd_mixer_oss 17857 1 snd_pcm_oss
snd_pcm 100169 3 snd_ens1371,snd_ac97_codec,snd_pcm_oss
snd_timer 33605 2 snd_seq,snd_pcm
snd 57157 10 snd_ens1371,snd_rawmidi,snd_
ac97_codec,snd_seq_oss,snd_seq,snd_seq_device,snd_pcm_oss,snd_mixer_oss,sn
d_pcm,snd_timer
soundcore 10913 1 snd
snd_page_alloc 9669 1 snd_pcm
pcnet32 36165 0
mii 5441 1 pcnet32
floppy 65269 0
ext3 132553 2
jbd 86233 1 ext3
BusLogic 79445 3
sd_mod 20545 4
scsi_mod 148105 2 BusLogic, sd_mod
[root@localhost ~]#

```

本例使用指令 lsmod 显示当前系统装载的模块。



modprobe、lsmod.old

## 1.48 man 指令：格式化和显示在线手册



**man[必要参数][选择性参数]**



如果指定章节，那么 man 指令将只在指定的章节中查找指定帮助信息。一般指定一个手册页的名字作为 man 命令的选择性参数，然后将显示这个手册页。可以通过参数指定和/etc/man.config 配置文件

指定 man 命令的搜索路径。

### ◎ 参数说明

必要参数的具体说明如下：

| 参 数      | 功 能                          |
|----------|------------------------------|
| -a       | 找出所有符合条件的对象                  |
| -C<文件名>  | 指定使用的配置文件，默认为/etc/man.config |
| -M<path> | 指定搜索手册页的目录列表，中间用冒号分隔         |
| -c       | 不使用 cat 文件                   |
| -S<名称>   | 在指定的章节搜索指定的手册页并显示            |
| -f<名称>   | 相当于 whatis 指令                |
| -k       | 相当于 apropos 指令               |
| -K<字符串>  | 在整个手册里寻找指定字符串                |
| -m<系统名>  | 基于指定的系统查找                    |
| -t       | 使用 troff 格式化手册页打印            |

选择性参数的具体说明如下：

| 参 数      | 功 能       |
|----------|-----------|
| -d       | 列出 bug 信息 |
| -help    | 列出帮助信息    |
| -version | 列出版本信息    |

### ◎ 实际范例

- ◎ 范例 91 ◎ 列出指令 clock 的手册页。

```
[root@localhost root]# man clock
Formatting page, please wait...
CLOCK(3) Linux Programmer's Manual CLOCK(3)
NAME
 clock - Determine processor time
SYNOPSIS
 #include <time.h>
```



`clock_t clock(void);`  
DESCRIPTION

The `clock()` function returns an approximation of processor time used by the program.

#### RETURN VALUE

The value returned is the CPU time used so far as a `clock_t`; to get the number of seconds used, divide by `CLOCKS_PER_SEC`. If the processor

time used is not available or its value cannot be represented, the function returns the value (`clock_t`)-1.

#### CONFORMING TO

ANSI C. POSIX requires that `CLOCKS_PER_SEC` equals 1000000 independent of the actual resolution.

:

#### ○ 范例 92 ○ 列出手册中的 netlink 字符串。

```
[root@localhost root]# man -K netlink
/usr/share/man/man2/bind.2.gz? [ynq] y
BIND(2) Linux Programmer's Manual BIND(2)
NAME
 bind - bind a name to a socket
SYNOPSIS
```

```
#include <sys/types.h>
#include <sys/socket.h>
int bind(int sockfd, struct sockaddr *my_addr, socklen_t addrlen);
```

#### DESCRIPTION

`bind` gives the socket `sockfd` the local address `my_addr`. `my_addr` is

`addrlen` bytes long. Traditionally, this is called "assigning a name to

a socket.`socket(2)`, it exists in a

name space (address family) but has no name assigned.

It is normally necessary to assign a local address using `bind` before a `SOCK_STREAM` socket may receive connections (see `accept(2)`).

## 1.49 manpath 指令：设置 man 手册的查询路径

The rules used in name binding vary between address families. Consult the manual entries in Section 7 for detailed information. For AF\_INET

see ip(7), for AF\_UNIX see unix(7), for AF\_APPLETALK see ddp(7), for

AF\_PACKET see packet(7), for AF\_X25 see x25(7) and for AF\_NETLINK see

netlink(7).

### RETURN VALUE

On success, zero is returned. On error, -1 is returned, and errno is

set appropriately.

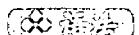
:



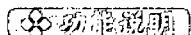
apropos、whatis、less、groff、man.conf

## 1.49

## manpath 指令：设置 man 手册的查询路径



**manpath[必要参数][选择性参数]**



**manpath** 指令用来设置 man 指令查找时的路径。



必要参数的具体说明如下：

| 参数      | 功能                |
|---------|-------------------|
| -M<路径名> | 指定手册页查询路径，中间用冒号隔开 |
| .       | 列出目前的查询路径         |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



### 执行范例

◎ 范例 93 ◎ 列出目前的查询路径。

```
[root@localhost root]# manpath
```

```
/usr/kerberos/man:/usr/local/share/man:/usr/share/man/en:/usr/share/man:/usr/X11R6/man:/usr/local/man
```

◎ 范例 94 ◎ 指定查询路径。

```
[root@localhost root]# manpath -M /home/wangyi
```

```
/home/wangyi
```

### 相关指令

man

## 1.50

### mkfontdir 指令：创建字体文件目录

#### 功能说明

**mkfontdir[选择性参数][必要参数]**

**功能说明** 对每一个指定的目录，搜寻所有的字体文件，生成文件 fonts.dir，供 X 服务器和字体服务器使用，以找到对应的字体文件。fonts.dir 也将被放置在指定的目录中。

#### 参数说明

必要参数的具体说明如下：

| 参数        | 功能           |
|-----------|--------------|
| <指定的搜寻目录> | 指定搜寻和生成文件的目录 |

选择性参数的具体说明如下：

| 参数 | 功能                        |
|----|---------------------------|
| -c | 指定一个含有编码文件的目录             |
| -n | 不搜寻字体，也不写入字体目录文件，用于生成编码目录 |
| -p | 指定编码文件生成路径的前缀             |
| -x | 搜寻字体文件时忽略 suffix 类型的文件    |
| -  | 结束点                       |

## 1.51 mount 指令：挂载文件系统

**执行范例**

- ◎ 范例 95 ◎ 在一个目录中通过搜寻字体文件创建文件 fonts.dir。

```
[root@yingzheng 100dpi]# mkdir fonts.dir
[root@yingzheng 100dpi]# ls fonts.dir
fonts.dir
```

**相关命令**

X、Xserver、xfs、xset

## 1.51 mount 指令：挂载文件系统

**命令格式** mount[选择性参数][必要参数]

**功能说明** 将指定设备上已存在的文件系统按照指定类型进行装载。这并不是真正地装载了设备，而是装载了已有设备上的文件系统。除了打印帮助和列表之类的指令外，一般装载都需要指定设备和挂载点。设备名称类似于/dev/hda1，挂载点就是一个目录。挂载完成后可以到挂载点目录去打开挂载的设备。

**参数说明**

必要参数的具体说明如下：

| 参数    | 功能       |
|-------|----------|
| <设备>  | 指定要挂载的设备 |
| <挂载点> | 指定的挂载目录  |

选择性参数的具体说明如下：

| 参数     | 功能               |
|--------|------------------|
| -V     | 打印版本信息           |
| -h     | 打印帮助信息           |
| -l     | 列出所有已挂载的系统，包括卷标签 |
| -t<类型> | 指定要挂载或者列出的文件系统类型 |



续表

| 参数       | 功能                              |
|----------|---------------------------------|
| -v       | 打印程序详细信息                        |
| -a       | 挂载 fstab 中提到的所有文件系统             |
| -F       | 和 “-a” 联用，以并行顺序为每个设备挂载          |
| -f       | 完成除了实际系统调用之外所有的挂载工作，但是并不实际挂载设备  |
| -i       | 不调用/sbin/mount.<文件系统>的帮助，即使文件存在 |
| -n       | 在挂载过程中不向/etc/mtab 中写入           |
| -p<数字>   | 在加密挂载的情况下，从数字指定的文件描述符中读取口令      |
| -s       | 容忍在挂载过程中的一些错误                   |
| -r       | 以只读的模式挂载设备，与 “-o ro” 相同         |
| -w       | 以读写的模式挂载设备，这是默认配置，与 “-o wo” 相同  |
| -L<标签>   | 只挂载符合指定标签的分区                    |
| -U<uuid> | 只挂载符合指定 uuid 的分区                |
| -t<类型>   | 指定要挂载的文件类型                      |
| -O       | 和 “-a” 连用，指定不希望挂载的设备            |
| -o       | 指定一些操作                          |
| -bind    | 重新挂载一个设备，在两个挂载点下设备都可以使用         |
| -move    | 把一个挂载树移到其他地方                    |

### ◎ 操作范例

#### ○ 范例 96 ○ 列出所有挂载的系统。

```
[root@yingzheng ~]# mount -l
/dev/sda3 on / type ext3 (rw) [/]
/dev/proc on /proc type proc (rw)
/dev/sys on /sys type sysfs (rw)
/dev/devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sdal on /boot type ext3 (rw) [/boot]
/dev/shm on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
```

## 1.52 mpost 指令：系统的绘图工具

```
automount (pid1963) on /misc type autofs (rw,fd=4,pgrp=1963,minproto=2,maxproto=4)
```

```
automount (pid2016) on /net type autofs (rw,fd=4,pgrp=2016,minproto=2,maxproto=4)
```

## ◎ 范例 97 ◎ 挂载一个 U 盘。

```
[root@yingzheng ~]# mount /dev/sdal /mnt/usb
```

```
[root@yingzheng ~]# cd /mnt/usb
```

```
[root@yingzheng usb]# ls
```

```
config-2.6.11-1.1369_FC4 System.map-2.6.11-1.1369_FC4
```

```
grub vmlinuz-2.6.11-1.1369_FC4
```

```
initrd-2.6.11-1.1369_FC4.img xen.gz
```

```
lost+found xen-syms
```

```
memtest86+-1.55.1
```

## ◎ 范例 98 ◎ 将一个已存在的挂载设备目录重新挂载到另一个目录。

```
[root@yingzheng usb2]# mount --bind /mnt/usb /mnt/usb2
```

```
[root@yingzheng mnt]# cd usb2
```

```
[root@yingzheng usb2]# ls
```

```
config-2.6.11-1.1369_FC4 System.map-2.6.11-1.1369_FC4
```

```
grub vmlinuz-2.6.11-1.1369_FC4
```

```
initrd-2.6.11-1.1369_FC4.img xen.gz
```

```
lost+found xen-syms
```

```
memtest86+-1.55.1
```

## 相关命令

umount、fstab、umount、swapon、nfs、xfs、e2label、xfs\_admin、mountd、nfsd、mke2fs、tune2fs、losetup

## 1.52

## mpost 指令：系统的绘图工具

## 基本

mpost[选择性参数]

## 功能说明

mpost 指令、inimpost 指令和 virmpost 指令组成了 MetaPost 的绘图工具系统。


● 选择性参数

选择性参数的具体说明如下：

| 参数                                  | 功能                                                         |
|-------------------------------------|------------------------------------------------------------|
| <b>[no]-file-line-error</b>         | 禁止或打开“文件：行：错误”格式的信息                                        |
| <b>-halt-on-error</b>               | 在第一个错误处停止处理                                                |
| <b>-ini</b>                         | ini/post 模式                                                |
| <b>-interaction=&lt;模式&gt;</b>      | 设置交互模式，模式包括 batchmode、nonstopmode、scrollmode、errorstopmode |
| <b>-jobname=&lt;字符串&gt;</b>         | 设置工作的名字                                                    |
| <b>-kpathsea-debug=&lt;数字&gt;</b>   | 根据数字设置路径搜寻标志                                               |
| <b>-mem=&lt;名字&gt;</b>              | 使用指定的名字代替程序名或者“%&”字符行                                      |
| <b>-output-directory=&lt;目录&gt;</b> | 以指定目录作为写入文件的目录                                             |
| <b>[no]-parse-first-line</b>        | 禁止或使用解析输入文件的第一行                                            |
| <b>-program=&lt;字符串&gt;</b>         | 设置程序名称为指定字符串                                               |
| <b>-recorder</b>                    | 打开文件名记录                                                    |
| <b>-tex=&lt;字符串&gt;</b>             | 以指定字符作为文本标签                                                |
| <b>-translate-file=&lt;文件&gt;</b>   | 使用指定的 TCX 文件                                               |
| <b>-8bit</b>                        | 使所有的字符默认可以打印                                               |
| <b>-help</b>                        | 打印帮助信息                                                     |
| <b>-version</b>                     | 输出版本信息                                                     |

● 相关命令

tex、mf、dvips

# 1.53

## msgcat 指令：合并消息目录

● 基本

msgcat[选项][输入文件]

● 功能说明 连接并合并指定的 PO (Portable Object, 可跨平台对象) 文件。在两个或多个指定的 PO 文件中找出共同的翻译消息。利用

## 1.54 msgcomm 指令：匹配两个消息目录

“–more-than”参数，可以使消息在不同文件出现的次数较多时才打印，使用“–less-than”参数则会使消息在不同文件出现次数较少时才打印（例如“–less-than=2”表示打印只出现一次的消息）。



必要参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -u | 要求只列出出现一次的消息   |
| -s | 输出前排序          |
| -F | 按文件位置排序输出      |
| -e | 不在输出中使用 C 转码序列 |

选择性参数的具体说明如下：

| 参数     | 功能           |
|--------|--------------|
| -f<文件> | 由文件读入输入文件的列表 |
| -D<目录> | 在目录中查找输入文件   |
| -o<文件> | 将输出写入指定文件    |



msgcomm

## 1.54 msgcomm 指令：匹配两个消息目录



msgcomm[必要参数][选择性参数][输入文件]



在两个或多个指定 PO 文件中找出共同的翻译信息。利用“–more-than”参数，可以使消息在不同文件出现的次数较多时才会打印；使用“–less-than”参数则会使消息在不同文件出现的次数较少时才打印。



必要参数的具体说明如下：



## 第1章 系统管理

| 参数                  | 功能                                 |
|---------------------|------------------------------------|
| -f<文件>              | 由指定文件读入输入文件的列表                     |
| -D<目录>              | 在指定目录中查找输入文件                       |
| -o<文件>              | 将输出写入指定文件                          |
| -><数字>              | 只打印出现次数多于指定次数的消息，若没有指定次数则默认值为1     |
| -<<数字>              | 只打印出现次数少于指定次数的消息，若没有指定次数则默认值为1     |
| -u                  | 只列出出现一次的消息                         |
| -P                  | 输入文件以 Java.properties 语法给出         |
| -stringtable-input  | 输入文件以 NeXTstep/GNUstep.string 语法给出 |
| -e                  | 不在输出中使用 C 转码序列                     |
| -E                  | 在输出中使用 C 转码序列，没有扩展字符               |
| -force-po           | 就算为空也写入 PO 文件                      |
| -i                  | 使用缩进风格写入 PO 文件                     |
| -no-location        | 不写入“#: 文件名：行号位置行”                  |
| -n                  | 生成“#: 文件名：行号位置行”                   |
| -strict             | 写入极为严格的 Uniform 使 PO 文件保持一致        |
| -p                  | 写出 Java.properties 文件              |
| -stringtable-output | 写出 NeXTstep/GNUstep.strings 文件     |
| -w=<数字>             | 指定输出页面宽度                           |
| -no-wrap            | 不将超过页面输出宽度的长消息断为多行                 |
| -s                  | 输出前排列                              |
| -F                  | 按文件位置排序输出                          |
| -omit-header        | 不写入带有 msgid 项的文件头                  |

选择性参数的具体说明如下：

| 参数        | 功能   |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |



msgcat

## 1.55

## msgen 指令：创建英语消息目录

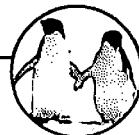
**msgen[必要参数][选择性参数][输入文件]**

创建一个英文翻译库。输入文件名是最近创建的英文 PO 文件或 POT 模板文件（通常由 xgettext 创建）。未翻译的条目会填上和 msgid 一样的字符串，并标记为模糊。



必要参数的具体说明如下：

| 参数                  | 功能                                 |
|---------------------|------------------------------------|
| -D=<目录>             | 在目录中查找输入文件。如果输入文件为“-”，则将读取标准输入     |
| -o=<文件>             | 将输出写至指定文件                          |
| -P                  | 输入文件以 Java.properties 语法给出         |
| -stringtable-input  | 输入文件以 NeXTstep/GNUstep.string 语法给出 |
| -e                  | 不在输出中使用 C 转码序列                     |
| -E                  | 在输出中使用 C 转码序列，没有扩展字符               |
| --force-po          | 就算为空也写入 PO 文件                      |
| -i                  | 使用缩进风格写入 PO 文件                     |
| --no-location       | 不写入“#: 文件名: 行号位置行”                 |
| --add-location      | 生成“#: 文件名: 行号位置行”                  |
| --strict            | 写入极为严格的 Uniform 使 PO 文件保持一致        |
| -p                  | 写出 Java.properties 文件              |
| -stringtable-output | 写出 NeXTstep/GNUstep.strings 文件     |
| -w=<数字>             | 指定输出页面宽度                           |
| --no-wrap           | 不将超过页面输出宽度的长消息断为多行                 |



续表

| 参数 | 功能        |
|----|-----------|
| -s | 输出前排列     |
| -F | 按文件位置排序输出 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



**msginit**

## 1.56 msginit 指令：初始化消息目录



**msginit[必要参数][选择性参数]**

**功能说明** 创建新的 PO 文件，即根据用户环境变量取值初始化文件头。



必要参数的具体说明如下：

| 参数                 | 功能                                 |
|--------------------|------------------------------------|
| -i=<文件>            | 指定输入的 POT 文件                       |
| -o=<文件>            | 指定输出文件                             |
| -P                 | 输入文件以 Java.properties 语法给出         |
| -stringtable-input | 输入文件以 NeXTstep/GNUstep.string 语法给出 |
| -I                 | 设置目标语系                             |
| --no-translator    | 假定 PO 文件是自动生成的                     |
| -p                 | 写出 Java.properties 文件              |

## 1.57 newgrp 指令：登录另一个群组

续表

| 参数                  | 功能                             |
|---------------------|--------------------------------|
| -stringtable-output | 写出 NeXTstep/GNUstep.strings 文件 |
| -w=<数字>             | 指定输出页面宽度                       |
| -no-wrap            | 不将超过页面输出宽度的长消息断为多行             |
| -h                  | 显示帮助并退出                        |
| -V                  | 输出版本信息并退出                      |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



msgen

## 1.57

## newgrp 指令：登录另一个群组



newgrp[群组名称]

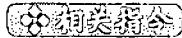
**功能说明** newgrp 指令可以使得用户以相同的用户名、不同的群组重新登录系统，但是用户名必须同时输入这两个群组。



○ 范例 99 ○ 改变群组。

```
[root@localhost ~]# newgrp root //改变群组
```

本例使用指令“newgrp root”改变群组重新登录系统。假设未改变群组之前时用户所属的群组为 kk 群组，现在改变群组为 root 群组，那么当前的用户名必须同时输入 kk 群组和 root 群组才能进行现在的操作。

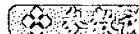


login、su



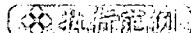
## 1.58

# nohup 指令：退出系统继续执行指令

**nohup[选择性参数][指令]****功能说明** nohup 指令可以忽略挂断信号，继续执行指令。

选择性参数的具体说明如下：

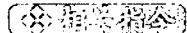
| 参数       | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



○范例 100 ○ 退出系统后继续在后台执行指令。

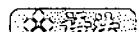
```
-sh-2.05b# nohup ncftpget -u other -p 02211025 192.168.88.7 /home
/kk ./111 & //退出系统后继续执行指令
[1] 2349
nohup: appending output to 'nohup.out'
-sh-2.05b#
```

本例使用指令“nohup ncftpget -u other -p 02211025 192.168.88.7 /home/kk ./111”，使得用户在退出系统之后仍然可以从指定 ftp 服务器上下载文件。

**ps**

## 1.59

# ntsysv 指令：设置系统的各种服务

**ntsysv[必要参数][选择性参数]**

**功能说明** ntsysv 指令用来设置系统的各种服务，执行指令后进入图形化界面，与 chkconfig 功能类似，但后者不提供图形界面。

## 1.60 open 指令：开启虚拟终端



必要参数的具体说明如下：

| 参数    | 功能      |
|-------|---------|
| -back | 显示 back |

选择性参数的具体说明如下：

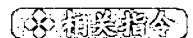
| 参数         | 功能   |
|------------|------|
| -level<等级> | 设置等级 |



- ◎ 范例 101 ◎ 设置操作系统启动时执行的程序。

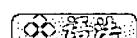


本例使用指令 ntsysv 设置系统开机时启动的各种服务。输入 ntsysv 指令后，按 [回车] 键，会进入如上所示的图形界面，然后进行相应的设置。

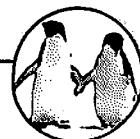


chkconfig

## 1.60 open 指令：开启虚拟终端



open[必要参数][选择性参数][-][指令]



**功能说明** open 指令用来开启虚拟终端，然后执行指定的指令，指定的指令可以包含参数。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -l | 登录环境         |
| -s | 切换到终端        |
| -u | 开启新的终端       |
| -v | 运行时显示详细的处理过程 |
| -w | 进入等待状态       |

选择性参数的具体说明如下：

| 参数     | 功能    |
|--------|-------|
| -c<数字> | 指定终端号 |

**执行范例**

○ 范例 102 ○ 开启新的终端执行指令。

[root@localhost ~]# open -c 8 - nslookup //开启终端 8，执行 nslookup  
指令

[root@localhost ~]# ps -A //显示当前进程

| PID TTY | TIME     | CMD         |
|---------|----------|-------------|
| 1 ?     | 00:00:01 | init        |
| 2 ?     | 00:00:00 | ksoftirqd/0 |
| 3 ?     | 00:00:00 | watchdog/0  |
| 4 ?     | 00:00:00 | events/0    |
| 5 ?     | 00:00:00 | khelper     |
| 6 ?     | 00:00:00 | kthread     |
| 8 ?     | 00:00:00 | kacpid      |

//此处省略了部分结果

|            |          |             |
|------------|----------|-------------|
| 2553 ?     | 00:00:08 | X           |
| 2586 ?     | 00:00:03 | gdm greeter |
| 2748 ?     | 00:00:00 | sshd        |
| 2752 pts/1 | 00:00:00 | bash        |

## 1.61 pgrep 指令：基于名字和其他属性的查找或信号处理

```
2946 tty9 00:00:00 bash
3065 tty8 00:00:00 nslookup
3069 pts/1 00:00:00 ps
[root@localhost ~]#
```

本例使用指令“open -c 8 -nslookup”开启终端 8，执行 nslookup 指令，然后运行“ps -A”指令时将看到“3065 tty8 00:00:00 nslookup”，进程号为 3065 的进程即为新开启的终端执行的指令。

- 范例 103 ○ 开启新的终端执行指令，并等待执行完后才返回。

```
[root@localhost ~]# open -c 9 -sw -- tcpdump . //开启终端 9，执行
tcpdump 指令
```

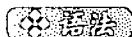
本例使用指令“open -c 9 -sw -tcpdump”执行指令 tcpdump，并等待，直到指令 tcpdump 执行完成之后才返回。



login、doshell、switchto

## 1.61

## pgrep 指令：基于名字和其他属性的查找或信号处理



**pgrep[必要参数][选择性参数][程序]**



pgrep 指令可以依照进程 ID 来浏览程序。



必要参数的具体说明如下：

| 参 数 | 功 能       |
|-----|-----------|
| -f  | 显示完整程序    |
| -l  | 显示源代码     |
| -n  | 显示新程序     |
| -o  | 显示旧程序     |
| -v  | 与条件不符合的程序 |
| -x  | 与条件符合的程序  |

选择性参数的具体说明如下：



| 参数      | 功能              |
|---------|-----------------|
| -P<进程号> | 列出父进程为指定进程的进程信息 |
| -t<终端>  | 指定终端下的所有程序      |
| -u<用户>  | 指定用户的程序         |

### ◎ 执行范例

◎ 范例 104 ◎ 显示与指定终端相联系的程序。

```
root@localhost:~# ps -A //显示所有进程信息
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 migration/0
 3 ? 00:00:01 ksoftirqd/0
 4 ? 00:00:00 migration/1
 5 ? 00:00:00 ksoftirqd/1
 6 ? 00:00:00 events/0
 7 ? 00:00:00 events/1
 8 ? 00:00:00 khelper
 9 ? 00:00:00 kthread
 //此处省略了部分结果
 32576 ? 00:00:00 lpd
 670 ? 00:00:00 sshd
 672 pts/0 00:00:00 bash
 706 pts/0 00:00:00 ps
root@localhost:~# pgrep -t tty2 //显示终端 2 下的所有程序
3952
8557
8587
8588
8779
root@localhost:~#
```

本例使用指令“pgrep -t tty2”显示终端 2 下运行的所有程序。pgrep 指令将终端 2 下运行的进程的进程号输出到标准输出，与运行“ps -A”指令

## 1.61 pgrep 指令：基于名字和其他属性的查找或信号处理

时显示的结果是相吻合的。与“ps -A”指令显示结果的吻合情况如下所示。

```
3951 tty1 00:00:00 login
3952 tty2 00:00:00 login
3953 tty3 00:00:00 login
3974 tty1 00:00:00 bash
8557 tty2 00:00:00 sh
8587 tty2 00:00:00 su
8588 tty2 00:00:00 bash
8606 tty3 00:00:00 bash
8779 tty2 00:00:00 vi
```

### ◎ 范例 105 ◎ 显示与某字符串相关的指令。

```
root@localhost:~# pgrep postmaster //显示与指定字符串相关的字符
```

```
3723
3726
3727
```

```
root@localhost:~#
```

本例使用指令“pgrep postmaster”显示与指定字符串 postmaster 相关的进程号。

### ◎ 范例 106 ◎ 列出父进程为 init 进程的所有进程号。

```
root@localhost:~# pgrep -P 1 //列出父进程为 1 的所有进程
```

```
2
3
4
5
6
7
8
9
119
738
2100
3201
3473
```



3479

3723

3739

3814

3832

3854

3857

3863

3891

3901

3908

3951

3952

3953

13786

26541

32406

32576

root@localhost:~#

本例使用指令“pgrep -P 1”列出父进程的进程号为 1 的进程的所有进程，进程号为 1 的进程为 init 进程。

○ 范例 107 ○ 反向选择。

root@localhost:~# pgrep -v -P 1 //反向选择

1

12

13

14

84

117

118

120

121

1532

## 1.61 pgrep 指令：基于名字和其他属性的查找或信号处理

```
1535
1607
1921
2872
3115
3726
3727
3974
8557
8587
8588
8606
8779
32417
32418
32420
32421
32422
670
672
```

```
root@localhost:~#
```

本例使用指令“pgrep -v -P 1”显示父进程不为 1 的所有进程的信息。

○ 范例 108 ○ 显示指定用户的所有进程。

```
root@localhost:~# pgrep -u root //显示指定用户的所有进程
```

```
1
2
3
4
5
6
7
8
9
```



12  
13  
14  
84  
117  
118  
119  
120  
121  
738  
1532  
1535  
1607  
1921  
2100  
2872  
3115  
3473  
3479  
3814  
3832  
3854  
3863  
3908  
3951  
3952  
3953  
3974  
8587  
8588  
8606  
8779  
13786



## 1.62 pidof 指令：找到运行程序的进程 ID

26541

32406

670

672

root@localhost:~#

本例使用指令“pgrep -u root”显示指定用户 root 的所有进程。



ps、kill

**1.62****pidof 指令：找到运行程序的进程 ID**

pidof[必要参数][选择性参数][程序]



pidof 指令可以显示正在执行的进程 ID。



必要参数的具体说明如下：

| 参 数 | 功 能      |
|-----|----------|
| -s  | 单次选中     |
| -c  | 只返回进程 ID |
| -x  | 同时返回脚本   |

选择性参数的具体说明如下：

| 参 数     | 功 能      |
|---------|----------|
| -o<进程号> | 忽略指定的进程号 |



○范例 109 ○ 显示指令所对应的进程号。

```
[root@localhost ~]# ps -A //显示所有进程
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 ksoftirqd/0
```



```

3 ? 00:00:00 watchdog/0
4 ? 00:00:00 events/0
5 ? 00:00:00 khelper
6 ? 00:00:00 kthread
8 ? 00:00:00 kacpid
 //此处省略了部分结果

2593 ? 00:00:01 sshd
2599 pts/1 00:00:00 bash
3655 ? 00:00:00 cupsd
2948 pts/1 00:00:00 ps
[root@localhost ~]# pidof mingetty //显示与 mingetty 相关的进程号
2333 2332 2331 2330 2329 2328
[root@localhost ~]#

```

本例使用指令“pidof mingetty”显示与 mingetty 相关的进程号，与指令“ps -A”显示结果相吻合，“ps -A”的对应部分如下所示：

```

2328 tty1 00:00:00 mingetty
2329 tty2 00:00:00 mingetty
2330 tty3 00:00:00 mingetty
2331 tty4 00:00:00 mingetty
2332 tty5 00:00:00 mingetty
2333 tty6 00:00:00 mingetty

```

○ 范例 110 ○ 每次只返回一个进程号。

```

[root@localhost ~]# pidof mingetty -s //每次只返回一个进程号
2333
[root@localhost ~]#

```

本例使用指令“pidof mingetty -s”显示与 mingetty 相关的指令，同时通过指定“-s”参数，每次只返回一个进程 ID。

**相关命令**

ps

## 1.63 pkill 指令：杀死进程

**语法**

pkill[必要参数][选择性参数][字符串]

**功能说明** pkill 指令是 ps 指令和 kill 指令的结合,用来杀死指定的进程。

**参数说明**

必要参数的具体说明如下:

| 参数 | 功 能      |
|----|----------|
| -f | 显示完整程序   |
| -l | 显示源代码    |
| -n | 显示新程序    |
| -o | 显示旧程序    |
| -v | 与条件不符的程序 |
| -x | 与条件符合的程序 |

选择性参数的具体说明如下:

| 参数      | 功 能             |
|---------|-----------------|
| -P<进程号> | 列出父进程为指定进程的进程信息 |
| -t<终端>  | 指定终端下的所有程序      |
| -u<用户>  | 指定用户的程序         |

**执行效果**

◎ 范例 111 ◎ 杀死指定进程。

```
[root@localhost ~]# ps -A //显示当前的进程
 PID TTY TIME CMD
 1 ? 00:00:02 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:01 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
 61 ? 00:00:00 kblockd/0
```



```

64 ? 00:00:00 khubd
//此处省略了部分结果

2586 ? 00:00:29 gdmgreeter
2593 ? 00:00:02 sshd
2599 pts/1 00:00:00 bash
3655 ? 00:00:00 cupsd
5817 pts/1 00:00:00 ftp
5825 pts/1 00:00:00 ps

[root@localhost ~]# pkill -9 ftp //杀死进程 ftp
[root@localhost ~]# ps -A //显示当前的进程

 PID TTY TIME CMD
 1 ? 00:00:02 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:01 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
//此处省略了部分结果

2586 ? 00:00:29 gdmgreeter
2593 ? 00:00:02 sshd
2599 pts/1 00:00:00 bash
3655 ? 00:00:00 cupsd
5829 pts/1 00:00:00 ps

[root@localhost ~]#

```

本例使用指令“`pkill -9 ftp`”杀死进程 `ftp`，然后通过“`ps -A`”，显示当前的进程，发现不再有进程 `ftp`。

○ 范例 112 ○ 杀死同一终端的所有进程。

```

[root@localhost ~]# pkill -t tty1 //杀死所有终端 1 下的所有进程
[root@localhost ~]# ps -A //显示所有的进程

 PID TTY TIME CMD
 1 ? 00:00:02 init
 2 ? 00:00:00 ksoftirqd/0

```

## 1.64 pmap 指令：显示程序的内存信息

第1章

```

3 ? 00:00:01 watchdog/0
4 ? 00:00:00 events/0
//此处省略了部分结果
2546 ? 00:00:00 gdm-binary
2554 ? 00:01:01 X
2586 ? 00:00:29 gdmgreeter
2593 ? 00:00:02 sshd
2599 pts/1 00:00:00 bash
3655 ? 00:00:00 cupsd
5866 ttym1 00:00:00 mingetty
5905 pts/1 00:00:00 ps
[root@localhost ~]#

```

本例使用指令“`pkill -t tty1`”杀死终端1的所有的进程，使用指令“`ps -A`”发现终端1下的所有进程都已经被杀死。

◎ 范例 113 ◎ 杀死指定用户的所有进程。

```
[root@localhost ~]# pkill -u kk //杀死用户 kk 的所有进程
[root@localhost ~]#

```

本例使用指令“`pkill -u kk`”杀死用户 kk 的所有进程。

◎ 范例 114 ◎ 反向选择。

```
[root@localhost ~]# pkill -vu kk //杀死不属于用户 kk 的所有进程
[root@localhost ~]#

```

本例使用指令“`pkill -vu kk`”杀死不属于用户 kk 的所有进程。

 **相关命令**

`kill`、`pgrep`

## 1.64 pmap 指令：显示程序的内存信息

 **操作说明** `pmap[必要参数][选择性参数][程序]`

 **功能说明** `pmap` 指令用来显示指定程序的内存信息。

 **参数说明**

必要参数的具体说明如下：



| 参数 | 功能       |
|----|----------|
| -x | 显示细节     |
| -d | 显示偏移和设备号 |
| -q | 不显示处理信息  |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -V | 版本信息 |

### ○执行范例○

○范例 115 ○ 显示程序的内存信息。

```
[root@localhost ~]# ps -A //显示进程信息
 PID TTY TIME CMD
 1 ? 00:00:02 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:01 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
 61 ? 00:00:00 kblockd/0
 64 ? 00:00:00 khubd
 //此处省略了部分结果
3655 ? 00:00:00 cupsd
5866 ttyl 00:00:00 mingetty
6014 pts/1 00:00:00 ps
[root@localhost ~]# pmap 2253 //显示进程 2253 的内存信息
2253: xfs -droppriv -daemon
007f0000 36K r-x-- /lib/libnss_files-2.3.5.so
007f9000 4K r-x-- /lib/libnss_files-2.3.5.so
007fa000 4K rwx-- /lib/libnss_files-2.3.5.so
00a84000 104K r-x-- /lib/ld-2.3.5.so
```

## 1.64 pmap 指令：显示程序的内存信息

```

00a9e000 4K r-x-- /lib/ld-2.3.5.so
00a9f000 4K rwx-- /lib/ld-2.3.5.so
00aa6000 1168K r-x-- /lib/libc-2.3.5.so
00bca000 8K r-x-- /lib/libc-2.3.5.so
00bcc000 8K rwx-- /lib/libc-2.3.5.so
00bce000 8K rwx-- [anon]
00bd2000 136K r-x-- /lib/libm-2.3.5.so
00bf4000 4K r-x-- /lib/libm-2.3.5.so
00bf5000 4K rwx-- /lib/libm-2.3.5.so
00bfe000 72K r-x-- /usr/lib/libz.so.1.2.2.2
00c10000 4K rwx-- /usr/lib/libz.so.1.2.2.2
00c13000 380K r-x-- /usr/X11R6/lib/libXfont.so.1.5
00c72000 32K rwx-- /usr/X11R6/lib/libXfont.so.1.5
00c7a000 36K rwx-- [anon]
00d6b000 388K r-x-- /usr/lib/libfreetype.so.6.3.7
00dcc000 28K rwx-- /usr/lib/libfreetype.so.6.3.7
00e03000 4K r-x-- [anon]
08047000 80K r-x-- /usr/X11R6/bin/xfs
0805b000 4K rw--- /usr/X11R6/bin/xfs
0805c000 16K rw--- [anon]
08c8e000 2364K rw--- [anon]
b7f48000 144K rw--- [anon]
b7f6c000 144K rw--- [anon]
b7f90000 8K rw--- [anon]
b7fb5000 4K rw--- [anon]
bfba1000 84K rw--- [stack]
total 5284K
[root@localhost ~]#

```

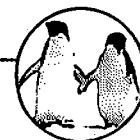
本例使用指令“pmap 2253”显示进程 2253 的内存使用信息。

○ 范例 116 ○ 显示进程使用内存的详细信息。

```
[root@localhost ~]# pmap -x 2253 //显示进程使用内存的详细信息
```

```
2253: xfs -droppriv -daemon
```

| Address | Kbytes | RSS | Anon | Locked | Mode | Mapping |
|---------|--------|-----|------|--------|------|---------|
|---------|--------|-----|------|--------|------|---------|



|          |      |   |   |         |                       |
|----------|------|---|---|---------|-----------------------|
| 007f0000 | 36   | - | - | - r-x-- | libnss_files-2.3.5.so |
| 007f9000 | 4    | - | - | - r-x-- | libnss_files-2.3.5.so |
| 007fa000 | 4    | - | - | - rwx-- | libnss_files-2.3.5.so |
| 00a84000 | 104  | - | - | - r-x-- | ld-2.3.5.so           |
| 00a9e000 | 4    | - | - | - r-x-- | ld-2.3.5.so           |
| 00a9f000 | 4    | - | - | - rwx-- | ld-2.3.5.so           |
| 00aa6000 | 1168 | - | - | - r-x-- | libc-2.3.5.so         |
| 00bca000 | 8    | - | - | - r-x-- | libc-2.3.5.so         |
| 00bcc000 | 8    | - | - | - rwx-- | libc-2.3.5.so         |
| 00bce000 | 8    | - | - | - rwx-- | [ anon ]              |
| 00bd2000 | 136  | - | - | - r-x-- | libm-2.3.5.so         |
| 00bf4000 | 4    | - | - | - r-x-- | libm-2.3.5.so         |
| 00bf5000 | 4    | - | - | - rwx-- | libm-2.3.5.so         |
| 00bfe000 | 72   | - | - | - r-x-- | libz.so.1.2.2.2       |
| 00c10000 | 4    | - | - | - rwx-- | libz.so.1.2.2.2       |
| 00c13000 | 380  | - | - | - r-x-- | libXfont.so.1.5       |
| 00c72000 | 32   | - | - | - rwx-- | libXfont.so.1.5       |
| 00c7a000 | 36   | - | - | - rwx-- | [ anon ]              |
| 00d6b000 | 388  | - | - | - r-x-- | libfreetype.so.6.3.7  |
| 00dcc000 | 28   | - | - | - rwx-- | libfreetype.so.6.3.7  |
| 00e03000 | 4    | - | - | - r-x-- | [ anon ]              |
| 08047000 | 80   | - | - | - r-x-- | xfs                   |
| 0805b000 | 4    | - | - | - rw--- | xfs                   |
| 0805c000 | 16   | - | - | - rw--- | [ anon ]              |
| 08c8e000 | 2364 | - | - | - rw--- | [ anon ]              |
| b7f48000 | 144  | - | - | - rw--- | [ anon ]              |
| b7f6c000 | 144  | - | - | - rw--- | [ anon ]              |
| b7f90000 | 8    | - | - | - rw--- | [ anon ]              |
| b7fb5000 | 4    | - | - | - rw--- | [ anon ]              |
| bfba1000 | 84   | - | - | - rw--- | [ stack ]             |
| <hr/>    |      |   |   |         |                       |
| total kB | 5284 | - | - | -       | -                     |

本例使用指令“pmap -x 2253”显示进程 2253 使用内存的详细信息。

## 1.64 pmap 指令：显示程序的内存信息

◎ 范例 117 ◎ 显示内存对应时同时显示设备号。

```
[root@localhost ~]# pmap -d 2253 //显示内存对应时同时显示设备号
2253: xfs -droppriv -daemon
Address Kbytes Mode Offset Device Mapping
007f0000 36 r-x-- 0000000000000000 008:00003 libnss_files-2.3.5.so
007f9000 4 r-x-- 000000000008000 008:00003 libnss_files-2.3.5.so
007fa000 4 rwx-- 000000000009000 008:00003 libnss_files-2.3.5.so
00a84000 104 r-x-- 0000000000000000 008:00003 ld-2.3.5.so
00a9e000 4 r-x-- 0000000000019000 008:00003 ld-2.3.5.so
00a9f000 4 rwx-- 000000000001a000 008:00003 ld-2.3.5.so
00aa6000 1168 r-x-- 0000000000000000 008:00003 libc-2.3.5.so
00bca000 8 r-x-- 0000000000124000 008:00003 libc-2.3.5.so
00bcc000 8 rwx-- 00000000000126000 008:00003 libc-2.3.5.so
00bce000 8 rwx-- 00000000000bce000 000:00000 [anon]
00bd2000 136 r-x-- 0000000000000000 008:00003 libm-2.3.5.so
00bf4000 4 r-x-- 0000000000021000 008:00003 libm-2.3.5.so
00bf5000 4 rwx-- 0000000000022000 008:00003 libm-2.3.5.so
00bfe000 72 r-x-- 0000000000000000 008:00003 libz.so.1.2.2.2
00c10000 4 rwx-- 0000000000011000 008:00003 libz.so.1.2.2.2
00c13000 380 r-x-- 0000000000000000 008:00003 libXfont.so.1.5
00c72000 32 rwx-- 000000000005e000 008:00003 libXfont.so.1.5
00c7a000 36 rwx-- 00000000000c7a000 000:00000 [anon]
00d6b000 388 r-x-- 0000000000000000 008:00003 libfreetype.so.6.3.7
00dcc000 28 rwx-- 0000000000061000 008:00003 libfreetype.so.6.3.7
00e03000 4 r-x-- 0000000000e03000 000:00000 [anon]
08047000 80 r-x-- 0000000000000000 008:00003 xfs
0805b000 4 rwx-- 0000000000013000 008:00003 xfs
0805c000 16 rw--- 0000000000805c000 000:00000 [anon]
08c8e000 2364 rw--- 0000000008c8e000 000:00000 [anon]
b7f48000 144 rw--- 00000000b7f93000 000:00000 [anon]
b7f6c000 144 rw--- 00000000b7f93000 000:00000 [anon]
b7f90000 8 rw--- 00000000b7f90000 000:00000 [anon]
b7fb5000 4 rw--- 00000000b7fb5000 000:00000 [anon]
```



```
bfba1000 84 rw--- 00000000bfba1000 000:00000 [stack]
mapped: 5284K writeable/private: 2896K shared: OK
[root@localhost ~]#
```

本例使用指令“pmap -d 2253”显示进程 2253 的内存对应，同时显示设备号。

### ◆ 常用命令

pgrep、pkill

## 1.65 procinfo 指令：显示系统状态

**procinfo[必要参数][选择性参数]**

procinfo 指令用来显示系统的参数，并将结果输出到标准输出。

### ◆ 功能说明

必要参数的具体说明如下：

| 参 数 | 功 能    |
|-----|--------|
| -a  | 显示所有信息 |
| -f  | 交互模式   |

选择性参数的具体说明如下：

| 参 数    | 功 能       |
|--------|-----------|
| -F<文件> | 结果输出到指定文件 |
| -h     | 帮助信息      |
| -n<时间> | 周期性更新的时间  |
| -v     | 版本信息      |

### ◆ 执行范例

◎ 范例 118 ◎ 显示系统状态。

```
[root@localhost ~]# procinfo //显示系统状态
```

**1.66 ps 指令：报告程序状况**

Linux 2.6.11-1.1369\_FC4 (bhcompile@decompose) (gcc 4.0.0 20050525 ) #1  
Thu Jun 2 22:55:56 EDT 2005 1CPU [localhost]

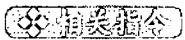
| Memory: | Total  | Used   | Free   | Shared | Buffers |
|---------|--------|--------|--------|--------|---------|
| Mem:    | 515292 | 341188 | 174104 | 0      | 29880   |
| Swap:   | 522104 | 0      | 522104 |        |         |

Bootup: Sun May 27 08:32:41 2007 Load average: 0.00 0.00 0.00 1/68 6585

| user :  | 0:01:17.91     | 0.2%  | page in : | 0                      |
|---------|----------------|-------|-----------|------------------------|
| nice :  | 0:01:14.23     | 0.2%  | page out: | 0                      |
| system: | 0:11:29.32     | 2.1%  | swap in : | 0                      |
| idle :  | 8:48:58.49     | 97.1% | swap out: | 0                      |
| uptime: | 9:04:39.93     |       | context : | 1203951                |
| irq 0:  | 32584931 timer |       | irq 8:    | 1 rtc                  |
| irq 1:  | 10 i8042       |       | irq 9:    | 0 acpi, Ensoniq AudioP |
| irq 2:  | 0 cascade [4]  |       | irq 10:   | 510820 eth0            |
| irq 3:  | 1              |       | irq 11:   | 33973 BusLogic BT-958  |
| irq 4:  | 1              |       | irq 12:   | 101 i8042              |
| irq 6:  | 4              |       | irq 15:   | 290946 idel            |
| irq 7:  | 0 parport0     |       |           |                        |

[root@localhost ~]#

本例使用指令 procinfo 显示系统状态。

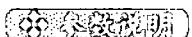


ps

## **1.66 ps 指令：报告程序状况**

ps[必要参数][选择性参数]

ps 指令用来显示当前进程的状态。



必要参数的具体说明如下：



| 参数 | 功能           |
|----|--------------|
| a  | 显示所有的进程      |
| -a | 显示同一终端下的所有程序 |
| -A | 显示所有的进程      |
| c  | 显示进程的真实名称    |
| -N | 反向选择         |
| -e | 等于“-A”       |
| e  | 显示环境变量       |
| f  | 显示程序间的关系     |
| -H | 显示树状结构       |
| r  | 显示当前终端的进程    |
| T  | 显示当前终端的所有进程  |
| u  | 指定用户的所有进程    |

选择性参数的具体说明如下：

| 参数          | 功能        |
|-------------|-----------|
| -C<指令>      | 列出指定指令的状况 |
| -help       | 帮助信息      |
| --version   | 版本信息      |
| -lines<行数>  | 每页显示的行数   |
| -width<字符数> | 每行显示的字符数  |

### ◎ 执行范例

◎ 范例 119 ◎ 显示所有的进程信息。

```
[root@localhost ~]# ps -A //显示进程信息
 PID TTY TIME CMD
 1 ? 00:00:02 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:01 watchdog/0
 4 ? 00:00:00 events/0
```

```

5 ? 00:00:00 khelper
6 ? 00:00:00 kthread
8 ? 00:00:00 kacpid
61 ? 00:00:00 kblockd/0
64 ? 00:00:00 khubd
110 ? 00:00:00 pdflush
 //此处省略了部分结果

```

```

2599 pts/1 00:00:01 bash
3655 ? 00:00:00 cupsd
5866 ttyl 00:00:00 mingetty
6686 pts/1 00:00:00 ps
[root@localhost ~]#

```

本例使用指令“ps -A”显示所有的进程信息。

○ 范例 120 ○ 显示指定用户的进程信息。

```
[root@localhost ~]# ps -u root //显示指定用户的所有进程信息
```

| PID  | TTY | TIME     | CMD         |
|------|-----|----------|-------------|
| 1 ?  |     | 00:00:02 | init        |
| 2 ?  |     | 00:00:00 | ksoftirqd/0 |
| 3 ?  |     | 00:00:01 | watchdog/0  |
| 4 ?  |     | 00:00:00 | events/0    |
| 5 ?  |     | 00:00:00 | khelper     |
| 6 ?  |     | 00:00:00 | kthread     |
| 8 ?  |     | 00:00:00 | kacpid      |
| 61 ? |     | 00:00:00 | kblockd/0   |

//此处省略了部分结果

```

2593 ? 00:00:03 sshd
2599 pts/1 00:00:01 bash
3655 ? 00:00:00 cupsd
5866 ttyl 00:00:00 mingetty
7621 pts/1 00:00:00 ps
[root@localhost ~]#

```

本例使用指令“ps -u root”显示指定用户 root 的所有进程信息。

○ 范例 121 ○ 显示指定终端的所有进程。

```
[root@localhost ~]# ps -t ttym //显示指定终端的所有进程
```

| PID  | TTY   | TIME     | CMD      |
|------|-------|----------|----------|
| 5866 | ttym1 | 00:00:00 | mingetty |

本例使用指令“ps -t ttym1”显示指定终端 ttym1 的所有进程。

## ◎ 范例 122 ◎ 显示所有进程，连同命令行。

```
[root@localhost ~]# ps -ef //显示所有指令，连同命令行
UID PID PPID C STIME TTY TIME CMD
root 1 0 08:32 ? 00:00:02 init [5]
root 2 1 08:32 ? 00:00:00 [ksoftirqd/0]
root 3 1 08:32 ? 00:00:01 [watchdog/0]
root 4 1 08:32 ? 00:00:00 [events/0]
root 5 1 08:32 ? 00:00:00 [khelper]
root 6 1 08:32 ? 00:00:00 [kthreadd]
root 8 6 08:32 ? 00:00:00 [kacpid]
root 61 6 08:32 ? 00:00:00 [kblockd/0]
 //此处省略了部分结果
root 2554 2546 0 08:34 ? 00:01:25 /usr/X11R6/bin/X :0 -audit
0 -auth /var/gdm/:0.Xauth -nolisten tcp vt7
gdm 2586 2546 0 08:34 ? 00:00:40 /usr/bin/gdmgreeter
root 2593 2128 0 08:35 ? 00:00:03 sshd: root@pts/1
root 2599 2593 0 08:35 pts/1 00:00:01 -bash
root 3655 1 09:39 ? 00:00:00 cupsd
root 5866 1 16:31 ttyl 00:00:00 /sbin/mingetty ttyl
root 7667 2599 0 19:38 pts/1 00:00:00 ps -ef
[root@localhost ~]#
```

本例使用指令“ps -ef”显示当前系统下的所有的进程信息，连同命令行。

## ◎ 范例 123 ◎ 显示指令的完整命令行。

```
[root@localhost ~]# ps -efww //显示进程信息，连同完整的命令行
UID PID PPID C STIME TTY TIME CMD
root 1 0 08:32 ? 00:00:02 init [5]
root 2 1 08:32 ? 00:00:00 [ksoftirqd/0]
root 3 1 08:32 ? 00:00:01 [watchdog/0]
 //此处省略了部分结果
root 2593 2128 0 08:35 ? 00:00:03 sshd: root@pts/1
root 2599 2593 0 08:35 pts/1 00:00:01 -bash
```

## 1.67 pstree 指令：以树状图显示程序

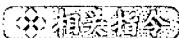
```
root 3655 1 0 09:39 ? 00:00:00 cupsd
root 5866 1 0 16:31 ttyl 00:00:00 /sbin/mingetty ttyl
root 7695 2599 0 19:41 pts/1 00:00:00 ps -efww
[root@localhost ~]#
```

本例使用命令“ps -efww”显示当前系统的所有进程信息，并且显示的进程是完整的命令行。

### ○ 范例 124 ○ 显示进程间的关系。

```
[root@localhost ~]# ps -efH //显示进程间的关系
UID PID PPID C STIME TTY TIME CMD
root 1 0 0 08:32 ? 00:00:02 init [5]
root 2 1 0 08:32 ? 00:00:00 [ksoftirqd/0]
root 3 1 0 08:32 ? 00:00:01 [watchdog/0]
root 4 1 0 08:32 ? 00:00:00 [events/0]
root 5 1 0 08:32 ? 00:00:00 [khelper]
 //此处省略了部分结果
gdm 2586 2546 0 08:34 ? 00:00:41 /usr/bin/gdmreeter
root 3655 1 0 09:39 ? 00:00:00 cupsd
root 5866 1 0 16:31 ttyl 00:00:00 /sbin/mingetty ttyl
[root@localhost ~]#
```

本例使用指令“ps -efH”显示进程间的关系，子进程以缩排的形式显示。



top、kill

## 1.67 pstree 指令：以树状图显示程序

**pstree[必要参数][选择性参数][程序]**

**pstree** 指令用来显示程序间的关系。

必要参数的具体说明如下：



| 参数 | 功能             |
|----|----------------|
| -a | 显示完整的指令        |
| -c | 不进行精简          |
| -h | 对现在执行的程序进行特别标注 |
| -p | 显示程序识别码        |
| -u | 显示用户名称         |
| -l | 显示长列表          |
| -n | 以进程ID进行排序      |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -version | 版本信息 |

### ☆范例与实训

#### ◎ 范例 125 ◎ 显示当前进程间的关系。

```
-sh-2.05b# pstree //显示进程间的关系
init+-+apmd
 |-atd
 |-bdflush
 |-crond
 |-cupsd
 |-gdm-binary---gdm-binary--+
 |-gdmgreeter
 |-gpm
 |-kapmd
 |-keventd
 |-khubd
 |-2*[kjournald]
 |-klogd
 |-kscand/DMA
 |-kscand/HighMem
```

## 1.67 pstree 指令：以树状图显示程序

```

|-kscand/Normal
|-ksoftirqd_CPU0
|-kswapd
|-kupdated
|-mdrecoveryd
|-6*[mingetty]
|-portmap
|-rpc.statd
|-2*[sendmail]
|-sshd---sshd---sh---pstree
|-syslogd
|-xfs
`-xinetd

```

-sh-2.05b#

本例使用指令 pstree 显示当前系统下运行的进程间的关系。

- 范例 126 ○ 特别表明正在运行的进程。

```
-sh-2.05b# pstree -apnh //显示进程间的关系
```

```

init,1
|-{keventd,2)
|-{kapmd,3)
|-{ksoftirqd_CPU0,4)
|-{kswapd,5)
|-{kscand/DMA,6)
|-{kscand/Normal,7)
|-{kscand/HighMem,8)
|-{bdfflush,9)
|-{kupdated,10)
|-{mdrecoveryd,11)
|-{kjournald,19)
|-{khubd,77)
|-{kjournald,1168)
|-syslogd,1479 -m 0
|-klogd,1483 -x

```



```

|- (portmap, 1493)
|- (rpc.statd, 1512)
|- apmd, 1579 -p 10 -w 5 -W -P /etc/sysconfig/apm-scripts/ apmscript
|- sshd, 1617
| '- sshd, 2228
| '- sh, 2230
| '- ptree, 2255 -apnh
|- xinetd, 1631 -stayalive -reuse -pidfile /var/run/xinetd.pid
|- (sendmail, 1650)
|- (sendmail, 1659)
|- gpm, 1669 -t imps2 -m /dev/mouse
|- crond, 1678
|- (xfs, 1753)
|- (atd, 1771)
|- mingetty, 1779 tty1
|- mingetty, 1780 tty2
|- mingetty, 1781 tty3
|- mingetty, 1782 tty4
|- mingetty, 1783 tty5
|- mingetty, 1784 tty6
|- (gdm-binary, 1785)
| '- (gdm-binary, 1812)
| |-X, 1813 :0 -auth /var/gdm/:0.Xauth vt7
| '- gdmreeter, 1821
'- cupsd, 1931
-sh-2.05b#

```

本例使用指令“`pstree -apnh`”显示当前系统下的所有进程间的关系。通过指定“`-h`”参数，可以标注正在进行中的进程；通过指定“`-a`”参数，可以显示完整的信息；通过指定“`-p`”参数，可以显示进程 ID；通过指定“`-n`”参数，可以使得输出结果以按照进程 ID 排序。

- 范例 127 ○ 同时显示用户名称。

## 1.67 pstree 指令：以树状图显示程序

```
[kk@localhost root]$ pstree -u //显示用户名
init--apmd
|_atd(daemon)
|_bdflush
|_crond
|_cupsd
|-gdm-binary---gdm-binary--X
| '-gdmreeter(gdm)
|-gpm
|-kapmd
|-keventd
|-khubd
|-2*[kjournald]
|-klogd
|-kscand/DMA
|-kscand/HighMem
|-kscand/Normal
|-ksoftirqd_CPU0
|-kswapd
|-kupdated
|-mdrecoveryd
|-6{mingetty}
|-portmap(rpc)
|-rpc.statd(rpcuser)
|-sendmail
|-sendmail(smmsp)
|-sshd---sshd---sh---su---sh---su---bash(kk)--+ftp
| '-pstree
|-syslogd
|-xfs(xfs)
'-xineted
[kk@localhost root]$
```

本例使用指令“pstree -cu”显示当前系统下所有进程的关系，同时显



示用户的名称。

相关指令

ps

# 1.68

## pwck 指令：检查密码文件

**pwck[必要参数][文件]**

**功能说明** pwck 指令可以检测密码文件的正确性。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功 能  |
|----|------|
| -r | 只读模式 |
| -s | 分类模式 |

**执行范例**

○ 范例 128 ○ 检查密码文件。

```
root@localhost:~# pwck /etc/passwd //检测密码文件
user news: directory /var/spool/news does not exist
user uucp: directory /var/spool/uucp does not exist
user list: directory /var/list does not exist
user irc: directory /var/run/ircd does not exist
user gnats: directory /var/lib/gnats does not exist
user nobody: directory /nonexistent does not exist
user wang: directory /home/wang does not exist
user telnetd: directory /nonexistent does not exist
user lchx: directory /home/lchx does not exist
user xgh: directory /home/xgh does not exist
user niu: directory /home/niu does not exist
user hq: directory /home/hq does not exist
user x2y: directory /home/x2y does not exist
```

## 1.69 pwd 指令：显示工作目录

```

user liang: directory /home/liang does not exist
user uml-net: directory /home/uml-net does not exist
user huohw: directory /home/huohw does not exist
user sensor: directory /home/sensor does not exist
user 973: directory /home/973 does not exist
user iplab: directory /home/iplab does not exist
user server: directory /home/server does not exist
pwck: no changes
root@localhost:~#

```

本例使用指令“`pwck /etc/passwd`”检测密码文件的正确性。

**命令相关指令**

`group`、`passwd`、`shadow`、`usermod`

## 1.69 pwd 指令：显示工作目录

**命令功能** `pwd`[选择性参数]

**功能说明** `pwd` 指令可以使用户获得目前的路径信息。

**参数说明**

选择性参数的具体说明如下：

| 参数                     | 功能   |
|------------------------|------|
| <code>-help</code>     | 帮助信息 |
| <code>--version</code> | 版本信息 |

**命令实例**

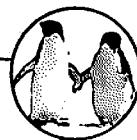
- 范例 129 显示当前的路径。

```
[root@localhost ~]# pwd //显示当前的路径
/root
[root@localhost ~]#
```

本例使用指令 `pwd` 显示用户当前所在的目录的绝对路径。

**命令相关指令**

`ls`

**1.70****reboot 指令：重新启动**

reboot[必要参数]



reboot 指令用来重新启动计算机。



必要参数的具体说明如下：

| 参 数 | 功 能                      |
|-----|--------------------------|
| -d  | 重新启动时不把数据写入/var/tmp/wtmp |
| -f  | 强制执行指令                   |
| -i  | 关闭网络设置之后再重新启动            |
| -n  | 保存数据后再重新启动               |
| -w  | 测试                       |



◎ 范例 130 ◎ 重新启动主机。

-sh-2.05b# reboot //重新启动

Broadcast message from root (pts/0) (Sun May 27 22:47:48 2007):

The system is going down for reboot NOW!

-sh-2.05b#

本例使用指令 reboot 重新启动主机。



init

**1.71****renice 指令：调整优先级**

renice[优先级][选择性参数]



renice 指令用来重新设置优先级，优先级的数值设

置为-20~19，只有系统管理员才可以设置。

### ◎ 参数说明

选择性参数的具体说明如下：

| 参    数  | 功    能        |
|---------|---------------|
| -g<群组>  | 修改指定群组的程序的优先级 |
| -p<进程号> | 改变指定进程的优先级    |
| -u<用户>  | 改变指定用户的优先级    |

### ◎ 实际范例

◎ 范例 131 ◎ 改变同一群组的程序的优先级。

```
[root@localhost ~]# renice 1 -g kk //改变同一群组的程序的优先级
0: old priority 0, new priority 1
[root@localhost ~]#
```

本例使用指令“renice 1 -g kk”，将同属于 kk 群组的所有程序的优先级改变为 1。

◎ 范例 132 ◎ 改变同一用户的程序的优先级。

```
[root@localhost ~]# renice 1 -u root //改变同一用户的程序的优先级
0: old priority -10, new priority 1
[root@localhost ~]#
```

本例使用指令“renice 1 -u root”，将同属于用户 root 的所有程序的优先级改变为 1。

◎ 范例 133 ◎ 改变指定进程的优先级。

```
[root@localhost ~]# ps -A //显示所有进程
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:00 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
```



```

61 ? 00:00:00 kblockd/0
2464 ? 00:00:06 X
2494 ? 00:00:02 gdmgreeter
2660 ? 00:00:00 sshd
2665 pts/1 00:00:00 bash
2796 pts/1 00:00:00 ps
[root@localhost ~]# renice 2 -p 2242 //改变进程的优先级
2242: old priority 1, new priority 2
[root@localhost ~]#

```

本例使用指令“renice 2 -p 2242”，将进程号为 2242 的进程的优先级改为 2。

#### 相关指令

nice

## 1.72 rlogin 指令：远端登录

#### 功能

rlogin[远端主机][必要参数][选择性参数]

功能说明 rlogin 指令用来连接当前本地主机系统到远端的主机系统。

#### 重要说明

必要参数的具体说明如下：

| 参数  | 功能                            |
|-----|-------------------------------|
| -E  | 忽略 escape 字符                  |
| -8  | 只识别 8 位的字符                    |
| -L  | 允许 rlogin 会话运行在 litout 模式     |
| -cc | 设置 escape 字符为 c               |
| -c  | 断开链接前要求确认                     |
| -a  | 强制要求远端主机在发送完一个空的本地用户名之后请求一个密码 |

续表

| 参数 | 功能                      |
|----|-------------------------|
| -f | 向远端主机发送一个本地的认证          |
| -F | 向远端主机发送一个可转寄的本地的认证      |
| -7 | 强制执行 7 位的传输             |
| -d | 打开用于与远端主机通信的 TCP 套接口的调试 |
| -k | 要求包含远端主机的 tickets 信息    |
| -x | 启动数据传输的 DES 加密          |
| -4 | 只使用 Kerberos 的版本 4 的认证  |

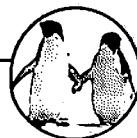
选择性参数的具体说明如下：

| 参数       | 功能      |
|----------|---------|
| -e<字符>   | 设置退出字符  |
| -l<用户>   | 指定登录的用户 |
| -t<终端类型> | 设置终端类型  |

### ☆ 执行范例

#### ◎ 范例 134 ◎ 显示 rlogin 服务是否开启。

```
-sh-2.05b# chkconfig --list //检测 rlogin 服务是否开启
kudzu 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
syslog 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
 //此处省略了部分结果
postgresql 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
pxe 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
lisa 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
vsftpd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
yppasswdd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
ypserv 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
ypfrd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
基于 xinetd 的服务
chargen-udp: 关闭
```



```

rsync: 关闭
chargen: 关闭
daytime-udp: 关闭
daytime: 关闭
echo-udp: 关闭
echo: 关闭
services: 关闭
servers: 关闭
time-udp: 关闭
time: 关闭
cups-lpd: 关闭
sgi_fam: 启用
finger: 关闭
imap: 关闭
imaps: 关闭
ipop2: 关闭
ipop3: 关闭
pop3s: 关闭
ktalk: 关闭
rexec: 关闭
rlogin: 关闭
rsh: 关闭
ntalk: 关闭
talk: 关闭
telnet: 启用

```

-sh-2.05b#

本例使用指令“`chkconfig -list`”检测 `rlogin` 服务是否启用，发现“`rlogin`:  
关闭”。

- 范例 135 ○ 开启 `rlogin` 服务。

-sh-2.05b# `chkconfig rlogin on` //开启 `rlogin` 服务

-sh-2.05b#

本例使用指令 “`chkconfig rlogin on`”，开启 `rlogin` 服务。

- 范例 136 ○ 登录远端主机。

## 1.73 rmmod 指令：删除模块

```
-sh-2.05b# rlogin 192.168.88.59 //登录主机192.168.88.59
Password:
Password:
Login incorrect
login: root
Password:
Login incorrect
login: kk
Password:
[kk@localhost kk]$
```

本例使用指令“rlogin 192.168.88.59”登录远端主机 192.168.88.59。由于系统不支持匿名登录，所以前两次的密码输入是错误的；由于系统默认不支持 root 用户登录，所以 root 用户登录时密码输入之后显示错误；最后用户名 kk 登录 192.168.88.59 成功。

## ◎ 范例 137 ◎ 指定用户名登录远端主机。

```
-sh-2.05b# rlogin 192.168.88.59 -l kk //登录远端主机
Password:
Last login: Mon May 28 15:30:25 from 192.168.88.59
[kk@localhost kk]$
```

本例使用指令“rlogin 192.168.88.59 -l kk”以用户名 kk 登录远端主机 192.168.88.59，输入密码后，登录成功。

## ◆ 相关命令

rsh、kerberos、krb\_sendauth、krb\_realmofhost、rlogin、klogind

## 1.73 rmmod 指令：删除模块

 rmmod[必要参数][选择性参数][模块]

 rmmod 指令从运行的内核中卸载可安装的模块。



必要参数的具体说明如下：



| 参数 | 功能             |
|----|----------------|
| -a | 删除所有不使用的模块     |
| -c | 保存当前的数据，不卸载模块  |
| -r | 删除栈，从命名的模块开始   |
| -s | 使用syslog记录错误信息 |
| -v | 运行时，显示详细的处理信息  |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◎执行范例

#### ◎ 范例 138 ◎ 显示已经安装的模块。

```
-sh-2.05b# lsmod //显示已经安装的模块
Module Size Used by Not tainted
parport_pc 19076 1 (autoclean)
lp 8996 0 (autoclean)
parport 37056 1 (autoclean) [parport_pc lp]
autofs 13268 0 (autoclean) (unused)
pcnet32 18240 1
mii 3976 0 [pcnet32]
keybdev 2944 0 (unused)
mousedev 5492 1
hid 22148 0 (unused)
input 5856 0 [keybdev mousedev hid]
usb-uhci 26348 0 (unused)
usbcore 78784 1 [hid usb-uhci]
ext3 70784 2 [ext3]
jbd 51892 2 [ext3]
BusLogic 100796 3
```

## 1.73 rmmod 指令：删除模块

```
sd_mod 13452 6
scsi_mod 107128 2 [BusLogic sd_mod]
-sh-2.05b#
```

本例使用指令 lsmod 显示当前系统下已经安装的模块，输出结果如上所示。

## ◎ 范例 139 ◎ 卸载模块。

```
-sh-2.05b# rmmod -v hid //卸载模块hid
Checking hid for persistent data
-sh-2.05b# lsmod //显示当前系统的模块
```

| Module     | Size   | Used by | Not tainted                 |
|------------|--------|---------|-----------------------------|
| parport_pc | 19076  | 1       | (autoclean)                 |
| lp         | 8996   | 0       | (autoclean)                 |
| parport    | 37056  | 1       | (autoclean) [parport_pc lp] |
| autofs     | 13268  | 0       | (autoclean) (unused)        |
| pcnet32    | 18240  | 1       |                             |
| mii        | 3976   | 0       | [pcnet32]                   |
| keybdev    | 2944   | 0       | (unused)                    |
| mousedev   | 5492   | 1       |                             |
| input      | 5856   | 0       | [keybdev mousedev]          |
| usb-uhci   | 26348  | 0       | (unused)                    |
| usbcore    | 78784  | 1       | [usb-uhci]                  |
| ext3       | 70784  | 2       |                             |
| jbd        | 51892  | 2       | [ext3]                      |
| BusLogic   | 100796 | 3       |                             |
| sd_mod     | 13452  | 6       |                             |
| scsi_mod   | 107128 | 2       | [BusLogic sd_mod]           |

本例使用指令“rmmod -v hid”删除模块 hid。

## ◎ 范例 140 ◎ 安装模块。

```
-sh-2.05b# insmod -v hid >1.log //安装模块
-sh-2.05b# tail -n 30 1.log //显示文件信息
user function /lib/modules/2.4.20-8/kernel/drivers/sound/uart6850.o
user function /lib/modules/2.4.20-8/kernel/drivers/sound/v_midi.o
```



```

user function /lib/modules/2.4.20-8/kernel/drivers/sound/via82cx0x_audio.o
user function /lib/modules/2.4.20-8/kernel/drivers/sound/wavefront.o
user function /lib/modules/2.4.20-8/kernel/drivers/sound/ymfpci.o
type 2 /lib/modules/2.4.20-8/kernel/drivers/telephony
xftw_readdir /lib/modules/2.4.20-8/kernel/drivers/telephony
user function /lib/modules/2.4.20-8/kernel/drivers/telephony
user function /lib/modules/2.4.20-8/kernel/drivers/telephony/ixj.o
user function /lib/modules/2.4.20-8/kernel/drivers/telephony/ixj_pcmcia.o
user function /lib/modules/2.4.20-8/kernel/drivers/telephony/phonedev.o
type 2 /lib/modules/2.4.20-8/kernel/drivers/usb
xftw_readdir /lib/modules/2.4.20-8/kernel/drivers/usb
user function /lib/modules/2.4.20-8/kernel/drivers/usb
user function /lib/modules/2.4.20-8/kernel/drivers/usb/CDCEther.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/acm.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/aiptek.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/audio.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/auerswald.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/brlvger.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/catc.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/dabusb.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/dsbr100.o
type 2 /lib/modules/2.4.20-8/kernel/drivers/usb/hcd
xftw_readdir /lib/modules/2.4.20-8/kernel/drivers/usb/hcd
user function /lib/modules/2.4.20-8/kernel/drivers/usb/hcd
user function /lib/modules/2.4.20-8/kernel/drivers/usb/hcd/ehci-hcd.o
user function /lib/modules/2.4.20-8/kernel/drivers/usb/hid.o
Using /lib/modules/2.4.20-8/kernel/drivers/usb/hid.o
Symbol version prefix ''
-sh-2.05b# lsmod //显示系统当前安装的模块
Module Size Used by Not tainted
hid 22148 0 (unused)

```

## 1.74 rsh 指令：远端登录的 shell

|            |        |   |                             |
|------------|--------|---|-----------------------------|
| parport_pc | 19076  | 1 | (autoclean)                 |
| lp         | 8996   | 0 | (autoclean)                 |
| parport    | 37056  | 1 | (autoclean) [parport_pc lp] |
| autofs     | 13268  | 0 | (autoclean) (unused)        |
| pcnet32    | 18240  | 1 |                             |
| mii        | 3976   | 0 | [pcnet32]                   |
| keybdev    | 2944   | 0 | (unused)                    |
| mousedev   | 5492   | 1 |                             |
| input      | 5856   | 0 | [hid keybdev mousedev]      |
| usb-uhci   | 26348  | 0 | (unused)                    |
| usbcore    | 78784  | 1 | [hid usb-uhci]              |
| ext3       | 70784  | 2 |                             |
| jbd        | 51892  | 2 | [ext3]                      |
| BusLogic   | 100796 | 3 |                             |
| sd_mod     | 13452  | 6 |                             |
| scsi_mod   | 107128 | 2 | [BusLogic sd_mod]           |

本例使用指令“insmod -v hid>1.log”安装模块 hid。



lsmod、insmod

## 1.74 rsh 指令：远端登录的 shell



rsh[必要参数][选择性参数][主机][指令]



rsh 指令复制标准输入到远端主机指令行，远端主机的输出被输出到本地主机的标准输出，远端输出的错误也输出到本地主机的标准输出。



必要参数的具体说明如下：



| 参数 | 功能   |
|----|------|
| -d | 调试模式 |
| -n | 指令定位 |

选择性参数的具体说明如下：

| 参数      | 功能    |
|---------|-------|
| -l<用户名> | 指定用户名 |

### ◎ 扩展范围

#### ◎ 范例 141 ◎ 开启 rsh 服务。

```
-sh-2.05b# chkconfig --list //检测开启的服务
kudzu 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
syslog 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
netfs 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
 //省略了部分结果
ypserv 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
ypxfrd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
```

#### 基于 xinetd 的服务

```
chargen-udp: 关闭
rsync: 关闭
chargen: 关闭
daytime-udp: 关闭
daytime: 关闭
echo-udp: 关闭
echo: 关闭
services: 关闭
servers: 关闭
time-udp: 关闭
time: 关闭
cups-lpd: 关闭
sgi_lam: 启用
finger: 关闭
```

```

imap: 关闭
imaps: 关闭
ipop2: 关闭
ipop3: 关闭
pop3s: 关闭
ktalk: 关闭
rexec: 关闭
rlogin: 启用
rsh: 关闭
ntalk: 关闭
talk: 关闭
telnet: 启用

-sh-2.05b# chkconfig rsh on //开启 rsh 服务
-sh-2.05b# chkconfig -list //检测开启的服务

kudzu 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
syslog 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
netfs 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
network 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
random 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
 //此处省略了部分结果

yppasswdd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
ypserv 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
ypxfrd 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭

基于 xinetd 的服务

chargen-udp: 关闭
rsync: 关闭
chargen: 关闭
daytime-udp: 关闭
daytime: 关闭
echo-udp: 关闭
echo: 关闭
services: 关闭
servers: 关闭

```



```

time-udp: 关闭
time: 关闭
cups-lpd: 关闭
sgi_fam: 启用
finger: 关闭
imap: 关闭
imaps: 关闭
ipop2: 关闭
ipop3: 关闭
pop3s: 关闭
ktalk: 关闭
rexec: 关闭
rlogin: 启用
rsh: 启用
ntalk: 关闭
talk: 关闭
telnet: 启用

```

-sh-2.05b#

本例使用指令“chkconfig rsh on”开启 rsh 服务。

○范例 142 ○ 远程指令执行。

[root@localhost ~]# rsh -l kk 192.168.88.59 /bin/ls//远程指令执行

本例使用指令“rsh -l kk 192.168.88.59 /bin/ls”以账号 kk 登录远程主机 192.168.88.59，执行指令/bin/ls。

#### 相关指令

rlogin、kerberos、krb\_sendauth、krb\_realmofhost、kshd

## 1.75

### rwho 指令：查看系统用户

#### 语法

rwho[必要参数]

#### 功能说明

rwho 指令显示局域网中的主机的用户信息。

## 1.76 screen 指令：多重视窗管理程序

第1章

**功能说明**

必要参数的具体说明如下：

| 参数 | 功 能     |
|----|---------|
| -a | 显示所有的用户 |

**运行范例**

◎ 范例 143 ◎ 显示本地局域网内的所有用户。

```
[root@localhost ~]# rwho -a //显示局域网内的所有用户
```

本例使用指令“rwho -a”显示本地局域网内的所有的用户。

**相关指令**

finger、rup、ruptime、rusers、who、rwhod

**1.76****screen 指令：多重视窗管理程序****语法** screen[必要参数][选择性参数]

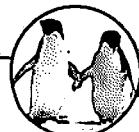
◆ 功能说明 ◆ screen 指令用来管理视窗程序。

**参数说明**

必要参数的具体说明如下：

| 参数    | 功 能               |
|-------|-------------------|
| -A    | 所有视窗调整为当前视窗的大小    |
| -m    | 强制建立作业            |
| -R    | 恢复作业，若不能恢复则建立新的作业 |
| -x    | 显示离线的作业           |
| -ls   | 显示当前的作业           |
| -wipe | 检查作业              |

选择性参数的具体说明如下：



| 参数        | 功能       |
|-----------|----------|
| -h<行数>    | 设置缓冲区行数  |
| -r<作业>    | 恢复指定的作业  |
| -s<shell> | 指定 shell |
| -v        | 显示版本信息   |
| -x        |          |

### ○○○执行范例

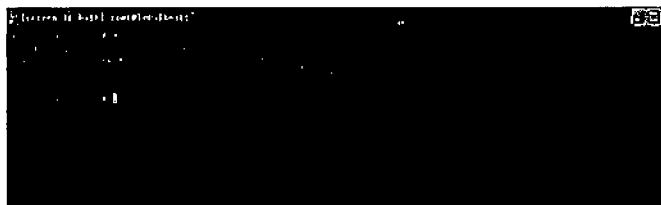
○范例 144 ○ 执行 screen 命令。

[root@localhost ~]# screen //打开窗口



执行 screen 指令之后会出现以上的窗口。

本例使用指令“Ctrl+a+c”，再打开一个窗口，如下所示。



然后运行指令 w，看到当前的用户“192.168.88.7:s.0”和“192.168.88.7:s.1”分别显示了两个 screen。

通过“Ctrl+a+a”，可以在不同的 screen 之间进行切换，如下所示。



## 1.76 screen 指令：多重视窗管理程序

以上是 screen0，通过屏幕左上角可以看出。



以上是 screen1，通过屏幕左上角可以看出。

由以上可以得到结论：在一个终端上可以通过 screen 指令来仿真多个终端。

- 范例 145 ○ 显示当前的 screen 作业。

```
[root@localhost ~]# screen -ls //显示当前的screen作业
```

There is a screen on:

```
 10473.pts-1.localhost (Attached)
```

```
1 Socket in /var/run/screen/S-root.
```

```
[root@localhost ~]#
```

本例使用指令“screen -ls”，显示当前的 screen 作业。

- 范例 146 ○ 让 screen 作业离线。

```
[root@localhost ~]# screen -d 10473.pts-1.localhost
```

//让screen作业离线

```
[10473.pts-1.localhost detached.]
```

```
[root@localhost ~]# screen -ls //显示screen作业
```

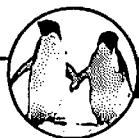
There is a screen on:

```
 10473.pts-1.localhost (Detached)
```

```
1 Socket in /var/run/screen/S-root.
```

```
[root@localhost ~]#
```

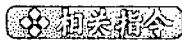
本例使用指令“screen -d 10473.pts-1.localhost”使指定的 screen 作业“10473.pts-1.localhost”离线，然后运行指令“screen -ls”，显示 screen 作业的信息，发现 screen 作业“10473.pts-1.localhost”的状态由 Attached 变为 Detached。



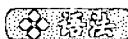
## ◎ 范例 147 ◎ 恢复 screen 作业。

```
[root@localhost ~]# screen -r 10473 pts-1.localhost //恢复 screen 作业
[remote detached]
[root@localhost ~]# screen -ls //显示 screen 作业
There is a screen on:
 10473.pts-1.localhost (Attached)
1 Socket in /var/run/screen/S-root.
```

本例使用指令“screen -r 10473.pts-1.localhost”恢复 screen 作业“10473.pts-1.localhost”。



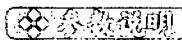
termcap、utmp、vi、captoinfo、tic

**1.77****shutdown 指令：系统关机指令**

**.shutdown** shutdown[必要参数][选择性参数][时间][警告信息]



**功能说明** shutdown 指令可以进行关机或者重新启动。



必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -a | 权限设置         |
| -c | 中断关机         |
| -f | 重新启动时不进行磁盘检测 |
| -F | 重新启动时进行磁盘检测  |
| -h | 关机           |
| -n | 不调用 init 关机  |
| -p | 关机后关闭电源      |
| -t | 测试           |
| -r | 重新启动         |

选择性参数的具体说明如下：

| 参数     | 功能        |
|--------|-----------|
| -<时间>  | 延迟指定的时间   |
| [时间]   | 指定时间后关机   |
| [警告信息] | 关机时显示警告信息 |

### ○ 热门范例

#### ◎ 范例 148 ◎ 关闭计算机。

```
-sh-2.05# shutdown 5 "i am going to sleep,byebye ^_~" //关闭计算机
Broadcast message from root (pts/0) (Tue May 29 11:17:39 2007):
i am going to sleep,byebye ^_~
The system is going DOWN to maintenance mode in 5 minutes!
Shutdown cancelled.

-sh-2.05#
```

本例使用指令“shutdown 5 “i am going to sleep,byebye ^\_~””，在5min之后关闭计算机，并在屏幕显示字符串“i am going to sleep,byebye”。关机尚未真正执行时，通过“Ctrl+C”，可以随时终止指令。

### ○ 相关指令

init、poweroff

## 1.78 sleep 指令：休眠

### ○ 热门

#### sleep[选择性参数][时间]

○ 功能说明

sleep 指令可以使系统进入休眠状态。

### ○ 参数说明

必要参数的具体说明如下：

| 参数   | 功能    |
|------|-------|
| [时间] | 休眠的时间 |



选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### 执行范例

- 范例 149 ○ 休眠 1min。

```
-sh-2.05b# sleep 1m //休眠
```

本例使用指令“sleep 1m”，系统将进入休眠时间 1min。

#### 相关指令

init

## 1.79

## su 指令：变更用户身份

#### 功能说明

su[必要参数][选择性参数][用户名]

功能说明 su 指令可以切换系统的用户账号。不输入用户名时，默认的用户为 root 用户。

#### 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能     |
|-----|---------|
| -m  | 不改变环境变量 |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### 执行范例

- 范例 150 ○ 切换用户。

## 1.80 sudo 指令：以其他身份来执行指令

```
-sh-2.05b# whoami //显示当前用户名
root
-sh-2.05b# pwd //当前目录
/root
-sh-2.05b# su kk //切换到用户 kk
[kk@localhost root]$ whoami //显示当前用户
kk
[kk@localhost root]$ pwd //显示当前目录
/root
[kk@localhost root]$
```

本例使用指令“su kk”切换当前的用户到 kk 用户。

○ 范例 151 ○ 切换至 root 用户。

```
[kk@localhost root]$ whoami //显示当前用户名
kk
[kk@localhost root]$ su //切换到 root 用户
Password:
sh-2.05b# whoami //显示当前用户名
root
sh-2.05b#
```

本例使用指令 su 切换到 root 用户，输入密码后，进入 root 用户模式。



passwd

## 1.80 sudo 指令：以其他身份来执行指令

**语法** sudo[必要参数][选择性参数][指令]

**功能说明** sudo 指令可以以系统的其他用户的身份执行指定的指令。

必要参数的具体说明如下：



| 参数 | 功能          |
|----|-------------|
| -b | 后台执行指定的指令   |
| -k | 每次执行都要输入密码  |
| -p | 变更密码提示符     |
| -L | 列出 sudo 的设置 |
| -v | 延长密码有效时间    |

选择性参数的具体说明如下：

| 参数     | 功能            |
|--------|---------------|
| -h     | 帮助信息          |
| -H<目录> | 使用指定的目录作为用户目录 |
| -u<用户> | 指定用户          |
| -V     | 版本信息          |

### ◎ 拓展范例

#### ◎ 范例 152 ◎ 显示 sudo 指令的设置。

```
sh-2.05b# sudo -L //显示 sudo 命令的设置
Available options in a sudoers "Defaults" line:
 syslog: Syslog facility if syslog is being used for logging
 syslog_goodpri: Syslog priority to use when user authenticates
successfully
 syslog_badpri: Syslog priority to use when user authenticates
unsuccessfully
 long_otp_prompt: Put OTP prompt on its own line
 ignore_dot: Ignore '.' in $PATH
 mail_always: Always send mail when sudo is run
 mail_badpass: Send mail if user authentication fails
 mail_no_user: Send mail if the user is not in sudoers
 mail_no_host: Send mail if the user is not in sudoers for this host
 mail_no_perms: Send mail if the user is not allowed to run a command
 tty_tickets: Use a separate timestamp for each user/tty combo
```

## 1.81 suspend 指令：暂停执行 shell

lecture: Lecture user the first time they run sudo

//此处省略了部分结果

badpass\_message: Incorrect password message

timestampdir: Path to authentication timestamp dir

exempt\_group: Users in this group are exempt from password and PATH requirements

passprompt: Default password prompt

runas\_default: Default user to run commands as

editor: Path to the editor for use by visudo

env\_check: Environment variables to check for sanity

env\_delete: Environment variables to remove

env\_keep: Environment variables to preserve

listpw: When to require a password for 'list' pseudocommand

verifypw: When to require a password for 'verify' pseudocommand

sh-2.05b#

本例使用指令“sudo -L”，显示 sudo 指令的设置。

◎ 范例 153 ◎ 以其他用户执行指令。

sh-2.05b# sudo -u kk ls -l /home/kk/ //以其他用户执行指令

总用量 8

-rw-r--r-- 1 kk kk 220 5月 28 22:03 123

-rwxrwxrwx 1 root root 12 4月 23 15:30 123.c

sh-2.05b#

本例使用指令“sudo -u kk ls -l /home/kk/”，以用户 kk 的名义执行指令“ls -l /home/kk”。

 相关命令

su

## 1.81 suspend 指令：暂停执行 shell

 悬浮按钮 suspend[必要参数]

 功能说明 suspend 指令可以暂停 shell 的执行。


功能说明

必要参数的具体说明如下：

| 参 数 | 功 能  |
|-----|------|
| -f  | 强制执行 |

执行范例

◎ 范例 154 ◎ 暂停 shell。

```
[root@localhost ~]# suspend //暂停 shell
-bash: suspend: cannot suspend a login shell
[root@localhost ~]# suspend -f //强制暂停 shell
```

本例使用指令“suspend -f”强制暂停 shell 的执行。

功能说明

kill

## 1.82

## swatch 指令：系统监控程序

语法

swatch[选择性参数]

功能说明

swatch 指令可以用来监控 syslog 文件。

参数说明

选择性参数的具体说明如下：

| 参 数      | 功 能       |
|----------|-----------|
| -c<配置文件> | 指定配置文件    |
| -f<文件>   | 检查记录文件    |
| -r<时间>   | 在指定时间重新启动 |

执行范例

◎ 范例 155 ◎ 开启系统监视。

```
[root@localhost ~]# swatch //开启系统监视
```

本例使用指令 swatch 开启系统监视。



syslogd

## 1.83

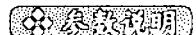
### symlinks 指令：维护符号链接的工具程序



symlinks[必要参数][目录]



symlinks 指令将检查目录中的符号链接情况。



必要参数具体说明如下：

| 参    数 | 功    能   |
|--------|----------|
| -c     | 显示相对路径   |
| -d     | 删除符号链接   |
| -r     | 递归检查子目录  |
| -v     | 显示所有符号链接 |



◎ 范例 156 ◎ 显示符号链接。

```
[root@localhost ~]# symlinks -v /root //显示符号链接
absolute: /root/.DCOPserver_localhost.localdomain_0 -> /root/./
DCOPserver_localhost.localdomain_0
[root@localhost ~]#
```

本例使用指令“symlinks -v /root”显示目录/root 下的符号链接。



ld

## 1.84

### tload 指令：显示系统负载



tload[选择性参数][终端]


**功能说明**

tload 指令用于以文本的方式显示系统的负载信息。

**参数说明**

选择性参数的具体说明如下：

| 参 数    | 功 能  |
|--------|------|
| -d<时间> | 间隔时间 |
| -V     | 版本信息 |

**执行范例**

○ 范例 157 ○ 显示系统负载。

```
[root@localhost ~]# tload //显示系统负载
0.36, 0.44, 0.19* * * * *
```

本例使用指令 tload 显示当前系统的负载。

**相关指令**

ps、top、uptime、w

# 1.85

## top 指令：显示进程信息

**语法**

top[必要参数][选择性参数]

**功能说明** top 指令可以显示当前系统正在执行的进程的相关信息，包括进程 ID、内存占用率、CPU 占用率等。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能    |
|-----|--------|
| -b  | 批处理    |
| -c  | 显示完整指令 |
| -l  | 忽略失效进程 |
| -s  | 保密模式   |
| -S  | 累积模式   |

## 1.85 top 指令：显示进程信息

选择性参数的具体说明如下：

| 参 数     | 功 能     |
|---------|---------|
| -i<时间>  | 设置间隔时间  |
| -u<用户名> | 指定用户名   |
| -p<进程号> | 指定进程    |
| -n<次数>  | 循环显示的次数 |

## ◎ 执行范例

## ◎ 范例 158 ◎ 显示进程信息。

```
[root@localhost ~]# top //显示进程信息
top - 16:56:57 up 21 min, 1 user, load average: 0.10, 0.03, 0.05
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.3% us, 0.7% sy, 0.0% ni, 99.0% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 515292k total, 170556k used, 344736k free, 16400k buffers
Swap: 522104k total, 0k used, 522104k free, 84912k cached

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2651 xfs 16 0 5284 3420 868 S 0.3 0.7 0:00.09 xfs
2978 root 15 0 29216 8688 2488 S 0.3 1.7 0:05.55 X
3010 gdm 16 0 41588 10m 7628 S 0.3 2.0 0:01.40 gdmgreeter
 1 root 16 0 1744 572 492 S 0.0 0.1 0:01.25 init
 2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
 3 root RT 0 0 0 0 S 0.0 0.0 0:00.02 watchdog/0
 4 root 10 -5 0 0 0 S 0.0 0.0 0:00.03 events/0
 5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
 6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread
 8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
 61 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 kblockd/0
 110 root 20 0 0 0 S 0.0 0.0 0:00.00 pdfflush
 111 root 15 0 0 0 S 0.0 0.0 0:00.05 pdfflush
 113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 aio/0
 112 root 23 0 0 0 S 0.0 0.0 0:00.00 kswapd0
```



## 第1章 系统管理

本例使用指令 top 显示当前系统的进程信息，包括进程 ID、内存占用率、CPU 占用率等。

### ○范例 159 ○ 显示完整指令。

```
[root@localhost ~]# top -c //显示程序信息
top - 17:00:03 up 24 min, 1 user, load average: 0.00, 0.01, 0.04
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 10.4% us, 5.7% sy, 0.0% ni, 83.9% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 515292k total, 170928k used, 344364k free, 16400k buffers
Swap: 522104k total, 0k used, 522104k free, 84912k cached
 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 1 root 16 0 1744 572 492 S 0.3 0.1 0:01.26 init [5]
 370 root 15 0 0 0 S 0.3 0.0 0:00.17 [kjournald]
 2978 root 15 0 29216 8688 2488 S 0.3 1.7 0:05.89
/usr/X11R6/bin/X :0 -audit 0 -auth /var/gdm/:0.Xauth -nolisten
 2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 [ksoftirqd/0]
 3 root RT 0 0 0 S 0.0 0.0 0:00.02 [watchdog/0]
 4 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 [events/0]
 5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 [khelper]
 6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 [kthread]
 8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 [kacpid]
 61 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 [kblockd/0]
 110 root 20 0 0 0 S 0.0 0.0 0:00.00 [pdflush]
 111 root 15 0 0 0 S 0.0 0.0 0:00.05 [pdflush]
 113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 [aio/0]
 112 root 23 0 0 0 S 0.0 0.0 0:00.00 [kswapd0]
 64 root 15 0 0 0 S 0.0 0.0 0:00.00 [khubd]
```

本例使用指令“top -c”显示当前系统的进程信息，包括进程 ID、内存占用率、CPU 占用率等；同时指定“-c”参数，用来显示指令的完整名称。

### ○范例 160 ○ 以批处理模式显示程序信息。

```
[root@localhost ~]# top -b //以批处理模式显示程序信息
top - 17:02:13 up 26 min, 1 user, load average: 0.00, 0.00, 0.02
```

## 1.85 top 指令：显示进程信息

```

Tasks: 65 total, 1 running, 64 sleeping, 0 stopped, 0 zombie
Cpu(s): 1.1% us, 2.6% sy, 0.0% ni, 94.8% id, 1.4% wa, 0.1% hi, 0.0% si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84980k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
3301 root 15 0 2016 860 692 R 2.0 0.2 0:00.01 top
 1 root 16 0 1744 572 492 S 0.0 0.1 0:01.26 init
 2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
 3 root RT 0 0 0 0 S 0.0 0.0 0:00.03 watchdog/0
 4 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 events/0
 5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
 6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread
 8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
 61 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 kblockd/0
 //此处省略了部分结果
2756 root 18 0 1556 432 372 S 0.0 0.1 0:00.00 mingetty
2757 root 18 0 1556 432 372 S 0.0 0.1 0:00.00 mingetty
2758 root 21 0 4592 1232 1060 S 0.0 0.2 0:00.01 prefdm
2943 root 16 0 11488 2304 1908 S 0.0 0.4 0:00.05 gdm-binary
2970 root 16 0 12080 2888 2384 S 0.0 0.6 0:00.03 gdm-binary
2978 root 15 0 29216 8688 2488 S 0.0 1.7 0:06.14 X
3010 gdm 15 0 41588 10m 7628 S 0.0 2.0 0:01.69 gdmreeter

```

本例使用指令“top -b”以批处理模式显示当前系统的进程信息，包括进程ID、内存占用率、CPU占用率等。

## ◎ 范例 161 ◎ 以累积模式显示程序信息。

```

[root@localhost ~]# top -b //显示程序信息
top - 17:04:07 up 28 min, 1 user, load average: 0.00, 0.00, 0.01
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0% us, 0.3% sy, 0.0% ni, 99.7% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84980k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2978 root 15 0 29216 8688 2488 S 0.3 1.7 0:06.65 X

```



```

1 root 16 0 1744 572 492 S 0.0 0.1 0:29.00 init
2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
3 root RT 0 0 0 0 S 0.0 0.0 0:00.03 watchdog/0
4 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 events/0
5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread
8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
61 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 kblockd/0
110 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pdflush
111 root 15 0 0 0 0 S 0.0 0.0 0:00.06 pdflush
113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 aio/0
112 root 23 0 0 0 0 S 0.0 0.0 0:00.00 kswapd0
64 root 15 0 0 0 0 S 0.0 0.0 0:00.00 khubd
200 root 15 0 0 0 0 S 0.0 0.0 0:00.00 ksericd

```

本例使用指令“top -S”以累积模式显示当前系统的进程信息，包括进程ID、内存占用率、CPU占用率等。

○范例 162 ○ 设置信息更新次数。

```

[root@localhost ~]# top -n 2 //显示进程信息
top - 17:05:40 up 29 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0% us, 0.7% sy, 0.0% ni, 99.3% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84984k cached

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2978 root 15 0 29216 8688 2488 S 0.7 1.7 0:06.57 X
2737 root 17 0 2148 720 632 S 0.3 0.1 0:00.44 hald-addon-stor
3010 gdm 15 0 41588 10m 7628 S 0.3 2.0 0:01.90 gdmreearer
1 root 16 0 1744 572 492 S 0.0 0.1 0:01.26 init
2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
3 root RT 0 0 0 0 S 0.0 0.0 0:00.03 watchdog/0
4 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 events/0
5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread

```

## 1.85 top 指令：显示进程信息

```
8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
61 root 10 -5 0 0 0 S 0.0 0.0 0:00.05 kblockd/0
110 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pdflush
111 root 15 0 0 0 0 S 0.0 0.0 0:00.06 pdflush
113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 aio/0
112 root 23 0 0 0 0 S 0.0 0.0 0:00.00 kswapd0
```

[root@localhost ~]#

本例使用指令“top -n 2”显示当前系统的进程信息，包括进程 ID、内存占用率、CPU 占用率等，并指定信息更新的次数为 2，更新信息两次之后，程序终止。

◎ 范例 163 ◎ 设置更新时间。

```
[root@localhost ~]# top -d 3 //设置更新时间
top - 17:08:19 up 32 min, 1 user, load average: 0.08, 0.02, 0.01
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.7%sy, 0.0%ni, 99.3%id, 0.0%wa, 0.0%hi, 0.0%si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84984k cached

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2978 root 15 0 29216 8688 2488 S 0.3 1.7 0:06.86 X
3010 gdm 15 0 41588 10m 7628 S 0.3 2.0 0:02.04 gdmrepeater
3352 root 16 0 2020 968 788 R 0.3 0.2 0:00.03 top
 1 root 16 0 1744 572 492 S 0.0 0.1 0:01.26 init
 2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
 3 root RT 0 0 0 0 S 0.0 0.0 0:00.04 watchdog/0
 4 root 10 -5 0 0 0 S 0.0 0.0 0:00.04 events/0
 5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
 6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread
 8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
61 root 10 -5 0 0 0 S 0.0 0.0 0:00.05 kblockd/0
110 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pdflush
111 root 15 0 0 0 0 S 0.0 0.0 0:00.07 pdflush
113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 aio/0
112 root 23 0 0 0 0 S 0.0 0.0 0:00.00 kswapd0
```



本例使用指令“top -d 3”显示当前系统的进程信息，包括进程ID、内存占用率、CPU占用率等，并指定信息更新的周期为3s，每3s更新一次数据。

◎范例 164 ◎ 显示指定用户的信息。

```
[root@localhost ~]# top -u root //显示进程信息
top - 17:10:35 up 34 min, 1 user, load average: 0.06, 0.02, 0.00
Tasks: 65 total, 2 running, 63 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.3% us, 0.3% sy, 0.0% ni, 99.3% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84988k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2978 root 15 0 29216 8688 2488 S 0.3 1.7 0:07.12 X
 1 root 16 0 1744 572 492 S 0.0 0.1 0:01.27 init
 2 root 34 19 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/0
 3 root RT 0 0 0 0 S 0.0 0.0 0:00.04 watchdog/0
 4 root 10 -5 0 0 0 S 0.0 0.0 0:00.05 events/0
 5 root 11 -5 0 0 0 S 0.0 0.0 0:00.02 khelper
 6 root 10 -5 0 0 0 S 0.0 0.0 0:00.00 kthread
 8 root 20 -5 0 0 0 S 0.0 0.0 0:00.00 kacpid
 61 root 10 -5 0 0 0 S 0.0 0.0 0:00.05 kblockd/0
 110 root 20 0 0 0 0 S 0.0 0.0 0:00.00 pdflush
 111 root 15 0 0 0 0 S 0.0 0.0 0:00.07 pdflush
 113 root 11 -5 0 0 0 S 0.0 0.0 0:00.00 aio/0
 112 root 23 0 0 0 0 S 0.0 0.0 0:00.00 kswapd0
 64 root 15 0 0 0 0 S 0.0 0.0 0:00.00 khubd
 200 root 15 0 0 0 0 S 0.0 0.0 0:00.00 ksericod
```

本例使用指令“top -u root”显示指定用户下运行的进程的信息，包括CPU占用率、内存占用率等。

◎范例 165 ◎ 显示指定进程信息。

```
[root@localhost ~]# top -p 200 //显示指定进程信息
top - 17:13:00 up 37 min, 1 user, load average: 0.00, 0.01, 0.00
```

## 1.86 uname 指令：显示系统信息

```
Tasks: 1 total, 0 running, 1 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.3% us, 0.7% sy, 0.0% ni, 98.7% id, 0.0% wa, 0.3% hi, 0.0% si
Mem: 515292k total, 170804k used, 344488k free, 16420k buffers
Swap: 522104k total, 0k used, 522104k free, 84988k cached
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
200 root 15 0 0 0 0 S 0.0 0.0 0:00.00 ksericd
```

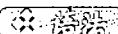
本例使用指令“top -p 200”显示进程号为 200 的进程的 CPU 占用率、内存占用率等信息。



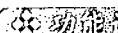
free、ps、uptime、atop、slabtop、vmstat、w

## 1.86

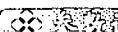
## uname 指令：显示系统信息



**uname[必要参数][选择性参数]**



uname 指令用来显示系统的相关信息。

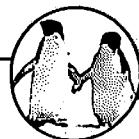


必要参数的具体说明如下：

| 参 数 | 功 能       |
|-----|-----------|
| -a  | 显示所有信息    |
| -m  | 计算机类型     |
| -n  | 计算机名      |
| -r  | release 号 |
| -s  | 操作系统名     |
| -v  | 显示系统时间    |

选择性参数的具体说明如下：

| 参 数       | 功 能  |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |



### ◎ 范例范例

#### ◎ 范例 166 ◎ 显示系统信息。

```
[root@localhost ~]# uname -a //显示系统信息
Linux localhost.localdomain 2.6.11-1.1369_FC4 #1 Thu Jun 2 22:55:56 EDT
2005 i686 i686 i386 GNU/Linux
[root@localhost ~]#
```

本例使用指令“uname -a”显示系统信息。

#### ◎ 范例 167 ◎ 显示计算机类型。

```
[root@localhost ~]# uname -m //显示计算机类型
i686
[root@localhost ~]#
```

本例使用指令“uname -m”显示计算机类型。

#### ◎ 范例 168 ◎ 显示计算机名。

```
[root@localhost ~]# uname -n //显示计算机名
localhost.localdomain
[root@localhost ~]#
```

本例使用指令“uname -n”显示计算机在网络上的名称。

#### ◎ 范例 169 ◎ 显示操作系统发行编号。

```
[root@localhost ~]# uname -r //显示操作系统发行编号
2.6.11-1.1369_FC4
[root@localhost ~]#
```

本例使用指令“uname -r”显示操作系统发行编号。

#### ◎ 范例 170 ◎ 显示操作系统名称。

```
[root@localhost ~]# uname -s //显示操作系统名称
Linux
[root@localhost ~]#
```

本例使用指令“uname -s”显示操作系统名称。

#### ◎ 范例 171 ◎ 显示系统时间。

```
[root@localhost ~]# uname -v //显示系统时间
#1 Thu Jun 2 22:55:56 EDT 2005
[root@localhost ~]#
```

本例使用指令“uname -v”显示系统时间。

**相关指令**

make、patch

**1.87****useradd 指令：建立用户账号****语法**

useradd[必要参数][选择性参数]

**功能说明**

useradd 指令用来建立账号。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能                |
|----|-------------------|
| -D | 修改默认值             |
| -m | 建立用户的根目录          |
| -o | 两个新建的用户的 uid 可以相同 |

选择性参数的具体说明如下：

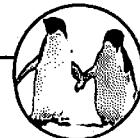
| 参数        | 功能       |
|-----------|----------|
| -p<密码>    | 指定密码     |
| -s<shell> | 指定 shell |
| -c<备注>    | 加上备注     |
| -d<目录>    | 指定根目录    |
| -e<时间>    | 指定有效期    |
| -f<时间>    | 指定用户缓冲区  |
| -g<群组>    | 指定用户群组   |
| -G<群组>    | 指定用户的次群组 |

**执行范例**

◎ 范例 172 ◎ 添加一般的用户。

root@localhost:~# useradd kk

//添加用户 kk



- 范例 173 ○ 为添加的用户指定相应的用户组。

`root@localhost:~# useradd -g root kk`

//添加用户 kk，并指定用户所

在的组为 root 用户组

- 范例 174 ○ 创建一个系统用户。

`root@localhost:~# useradd -r kk`

//创建一个系统用户 kk

- 范例 175 ○ 为新添加的用户指定 home 目录。

`root@localhost:~# useradd -d /home/myf kk` //新添加用户 kk，其 home 目录为/home/myf

当用用户名 kk 登录主机时，系统进入的默认目录为/home/myf。



deluser、groupadd、usermod

## 1.88 userconf 指令：用户账号设置



userconf[选择性参数]



userconf 指令用来设置用户配置，当不输入任何参数时，将进入图形界面进行设置。



选择性参数的具体说明如下：

| 参 数                        | 功 能  |
|----------------------------|------|
| <code>-addgroup[群组]</code> | 增加群组 |
| <code>-adduser[用户]</code>  | 增加用户 |
| <code>-delgroup[群组]</code> | 删除群组 |
| <code>-deluser[用户]</code>  | 删除用户 |
| <code>-help</code>         | 帮助信息 |



- 范例 176 ○ 新增用户。

`[root@localhost temp]# userconf --adduser 666 kk lord /bin/bash`

## 1.89 usermod 指令：修改用户账号

//新增用户账号

[root@localhost temp]#

本例使用指令“userconf --adduser 666 kk lord /bin/bash”新增用户账号 lord。



useradd

## 1.89

## usermod 指令：修改用户账号



usermod[必要参数][选择性参数][用户账号]



usermod 指令用来修改用户账号的设置。

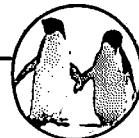


必要参数的具体说明如下：

| 参 数 | 功 能              |
|-----|------------------|
| -D  | 修改默认值            |
| -m  | 建立用户的根目录         |
| -o  | 两个新建用户的 uid 可以相同 |

选择性参数的具体说明如下：

| 参 数       | 功 能      |
|-----------|----------|
| -p<密码>    | 指定密码     |
| -s<shell> | 指定 shell |
| -c<备注>    | 加上备注     |
| -d<目录>    | 指定根目录    |
| -e<时间>    | 指定有效期    |
| -f<时间>    | 指定用户缓冲区  |
| -g<群组>    | 指定用户群组   |
| -G<群组>    | 指定用户的次群组 |



### 执行范例

◎ 范例 177 ◎ 改变登录目录。

```
[root@localhost temp]# usermod -d /home/temp/ kk //改变登录目录
```

```
[root@localhost temp]#
```

本例使用指令“usermod -d /home/temp/ kk”改变用户 kk 的登录目录为/home/temp。

◎ 范例 178 ◎ 改变用户的 uid。

```
[root@localhost temp]# usermod -u 777 kk //改变用户的uid
```

```
[root@localhost temp]#
```

本例使用指令“usermod -u 777 kk”改变用户 kk 的 uid 为 777。

### 相关指令

useradd、userdel

## 1.90 userdel 指令：删除用户账号

### 命令

**userdel[必要参数][用户]**

**功能说明** userdel 指令用来删除用户账号。还可以通过加入参数，删除与指定用户相关的文件。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能                      |
|----|-------------------------|
| -r | 删除用户的同时，删除用户登录目录及目录下的文件 |

### 执行范例

◎ 范例 179 ◎ 删除用户账号。

```
[root@localhost temp]# userdel kk //删除用户 kk
```

```
[root@localhost temp]#
```

本例使用指令“userdel kk”删除用户 kk。

### 相关指令

useradd

## 1.91 users 指令：显示用户

 **帮助** users[选择性参数]

 **功能说明** users 指令用来显示所有的用户。

 **参数说明**

选择性参数的具体说明如下：

| 参数                    | 功能   |
|-----------------------|------|
| <code>-help</code>    | 帮助信息 |
| <code>-version</code> | 版本信息 |

 **执行范例**

◎ 范例 180 ◎ 显示所有的用户。

```
[root@localhost temp]# users //显示所有的用户
```

```
root root
```

```
[root@localhost temp]#
```

本例使用指令 users 显示所有的用户。

 **相关指令**

`useradd`、`userdel`

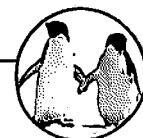
## 1.92 vlock 指令：锁定终端

 **帮助** vlock[必要参数][选择性参数]

 **功能说明** vlock 指令用来锁定虚拟终端。

 **参数说明**

必要参数的具体说明如下：



| 参数 | 功能       |
|----|----------|
| -a | 锁定所有虚拟终端 |
| -c | 锁定当前虚拟终端 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◎ 范例 181 ◎ 锁定虚拟终端

```
[root@localhost ~]# vlock //锁定虚拟终端
*** This tty is not a VC (virtual console). ***
*** It may not be securely locked. ***
This TTY is now locked.
Please enter the password to unlock.
```

本例使用指令 vlock 锁定虚拟终端。

### ○ 范例 182 ○ 通过输入密码解锁。

```
[root@localhost ~]# vlock //锁定虚拟终端
*** This tty is not a VC (virtual console). ***
*** It may not be securely locked. ***
This TTY is now locked.
Please enter the password to unlock.
root's Password: //输入密码
root's Password:
```

锁定虚拟终端之后，可以通过输入 root 用户的密码进行解锁。

## 1.93

### w 指令：显示登录系统的用户信息



w[必要参数][用户名]



w 指令用来显示当前登录到系统的用户信息。



必要参数的具体说明如下：

| 参 数 | 功 能              |
|-----|------------------|
| -f  | 显示或者不显示登录位置      |
| -h  | 不显示标题            |
| -l  | 显示详细信息           |
| -s  | 显示简洁信息           |
| -u  | 忽略程序名和占用 CPU 的时间 |

选择性参数的具体说明如下：

| 参 数 | 功 能  |
|-----|------|
| -V  | 版本信息 |



◎ 范例 183 ◎ 显示当前用户。

```
[root@localhost ~]# w //显示当前用户，不显示登录位置
11:13:19 up 2:05, 3 users, load average: 0.00, 0.03, 0.04
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root tty1 - 11:02 10:26 0.10s 0.10s -bash
root pts/1 192.168.88.7 09:13 0.00s 0.28s 0.02s w
root :0 - 11:01 ?xdm? 52.72s 0.50s
/usr/bin/gnome-session
[root@localhost ~]#
```

本例使用指令 w 显示当前用户。



## ○范例 184 ○ 不显示登录位置。

```
[root@localhost ~]# w -f //显示登录用户
11:14:13 up 2:06, 3 users, load average: 0.00, 0.03, 0.04
USER TTY LOGINE IDLE JCPU PCPU WHAT
root tty1 11:02 11:20 0.10s 0.10s -bash
root pts/1 09:13 0.00s 0.28s 0.01s w -f
root :0 11:01 ?xdm? 53.00s 0.50s /usr/bin/gnome-session
[root@localhost ~]#
```

本例使用指令“w -f”显示登录用户，但不显示登录位置。

## ○范例 185 ○ 以精简模式显示。

```
[root@localhost ~]# w -s //显示登录用户
11:15:44 up 2:07, 3 users, load average: 0.07, 0.04, 0.04
USER TTY FROM IDLE WHAT
root tty1 - 12:51 -bash
root pts/1 192.168.88.7 0.00s w -s
root :0 - ?xdm? /usr/bin/gnome-session
[root@localhost ~]#
```

本例使用指令“w -s”以精简模式显示登录用户的信息。

## ○范例 186 ○ 不显示标题栏。

```
[root@localhost ~]# w -h //显示登录用户但不显示标题栏
root tty1 - 11:02 13:54 0.10s 0.10s -bash
root pts/1 192.168.88.7 09:13 0.00s 0.29s 0.01s w -h
root :0 - 11:01 ?xdm? 53.69s 0.50s
/usr/bin/gnome-session
[root@localhost ~]#
```

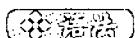
本例使用指令“w -h”显示登录用户但不显示标题栏。



free、ps、top、uptime、utmp、who

# 1.94

## wait 指令：等待程序返回状态



wait[程序]

## 1.95 watch 指令：将结果输出到标准输出



**功能说明** wait 指令用来等待指定的程序返回状态。



◎ 范例 187 ◎ 等待程序返回状态。

```
[root@localhost ~]# ps -A //显示当前进程
 PID TTY TIME CMD
 1 ? 00:00:01 init
 2 ? 00:00:00 ksoftirqd/0
 3 ? 00:00:00 watchdog/0
 4 ? 00:00:00 events/0
 5 ? 00:00:00 khelper
 6 ? 00:00:00 kthread
 8 ? 00:00:00 kacpid
 61 ? 00:00:00 kblockd/0
 64 ? 00:00:00 khubd
 110 ? 00:00:00 pdfflush
 111 ? 00:00:00 pdfflush
 //此处省略了部分结果
 4344 ? 00:00:00 mapping-daemon
 4345 ? 00:00:00 pam_timestamp_c
 4347 ? 00:00:00 notification-ar
 4349 ? 00:00:00 clock-applet
 4569 pts/1 00:00:00 top
 4570 pts/1 00:00:00 ps
[1]+ Stopped top
[root@localhost ~]# wait 4569 //等待程序返回状态
```

本例使用指令“wait 4569”等待进程 4569，也就是 top 程序，返回状态值。



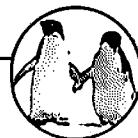
ps

# 1.95

## watch 指令：将结果输出到标准输出设备



watch[必要参数][选择性参数][指令]



**功能说明** watch 指令可以将指定指令的输出结果输出到标准输出设备。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -d | 显示差异 |

选择性参数的具体说明如下：

| 参数        | 功能         |
|-----------|------------|
| -n<时间>    | 周期性执行指令的间隔 |
| -h        | 帮助信息       |
| --version | 版本信息       |

**执行范例**

○ 范例 188 ○ 每隔 10s 执行一次指令。

```
[root@localhost ~]# watch -n 10 who //每隔 10s 执行一次 who 指令
```

```
Every 10.0s: who Thu May 31 11:37:31 2007
```

```
root ttys1 May 31 11:02
```

```
root pts/1 May 31 09:13 (192.168.88.7)
```

```
root :0 May 31 11:01
```

本例使用指令“watch -n 10 who”每隔 10s 执行一次 who 指令。

○ 范例 189 ○ 高亮字符标识信息的差异。

```
[root@localhost ~]# watch -d ls -l //将“ls -l”的结果输出到标准输出
```

```
Every 2.0s: ls -l Thu May 31 11:40:06 2007
```

```
5988
```

```
-rw-r--r-- 1 root root 220 5 27 17:18 123
```

```
-rw-r--r-- 1 root root 2 5 27 09:27 123.pcx
```

```
-rw-r--r-- 1 root root 0 5 30 10:21 1.log
```

## 1.96 whereis 指令：查找文件

第  
1  
章

```
-rw-r--r-- 1 root root 0 5 23 21:33 1.log.new
-rw-r--r-- 1 root root 0 5 30 09:33 2.log
-rw-r--r-- 1 root root 211 5 27 17:26 456
-rw-r--r-- 1 root root 0 5 30 09:36 7.log
-rw-r--r-- 1 root root 76 5 31 10:19 abc
-rw----- 1 root root 975 12 22 11:16 anaconda-ks.cfg
-rw-r--r-- 1 root root 56991 2005-06-03 config-2.6.11-1.1369_FC4
-rw-r--r-- 1 root root 348 5 27 17:21 ddd
-rw----- 1 root root 104 5 24 15:11 DEADJOE
drwxr-xr-x 2 root root 4096 4 19 17:32 Desktop
-rw-r--r-- 1 root root 180 5 28 23:08 examplesed
-rw-r--r-- 1 root root 126 5 24 10:53 file1
-rw-r--r-- 1 root root 120 5 24 10:53 file2
-rw-r--r-- 1 root root 483 3 30 14:37 gmon.out
drwxr-xr-x 2 root root 4096 12 22 11:16 grub
-rw-r--r-- 1 root root 1224449 12 22 09:52 initrd-2.6.11-1.1369_FC4.img
-rw-r--r-- 1 root root 64471 12 22 11:16 install.log
```

本例使用指令“watch -d ls -l”将“ls -l”的结果输出到标准输出，并在出现不同的信息时，用高亮字符标识。



cat

## 1.96

## whereis 指令：查找文件

**whereis[必要参数][选择性参数][文件]**

whereis 指令用来查找符合条件的文件，并将结果列出到标准输出。



必要参数的具体说明如下：



## 第1章 系统管理

| 参数 | 功能         |
|----|------------|
| -b | 只匹配二进制文件   |
| -f | 只显示文件的真实名称 |
| -m | 只匹配说明文件    |
| -s | 只匹配原文件     |

选择性参数的具体说明如下：

| 参数     | 功能     |
|--------|--------|
| -S<目录> | 设置查找目录 |
| -M<目录> | 设置查找目录 |
| -S<目录> | 设置查找目录 |

### 命令行简介

- 范例 190 ○ 查找文件。

```
[root@localhost ~]# whereis linux //查找文件
linux: /usr/include/linux
[root@localhost ~]#
```

本例使用指令“whereis linux”查找文件 linux。

### 命令大全

locate、which

# 1.97

## which 指令：查找文件

### 命令语法

which[文件]

**功能说明** which 指令用来查找文件，它的查找范围由环境变量“\$PATH”设置。

### 命令行简介

- 范例 191 ○ 查找文件。

## 1.98 who 指令：显示系统用户信息

```
[root@localhost ~]# which linux //查找文件
/usr/bin/which: no linux in (/usr/kerberos/sbin:/usr/kerberos/
bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R
6/bin:/root/bin)
```

```
[root@localhost ~]#
```

本例使用指令“which linux”查找文件 linux，但是在指定的路径下没有找到文件 linux。

```
[root@localhost ~]# which who //查找文件
/usr/bin/who
```

```
[root@localhost ~]#
```

本例使用指令“which who”查找文件 who。



whereis、locate

## 1.98 who 指令：显示系统用户信息



**who[必要参数][选择性参数][记录文件]**



who 指令用来显示当前登录系统的用户的信息。



必要参数的具体说明如下：

| 参数 | 功    能    |
|----|-----------|
| -H | 多输出一行标题栏  |
| -l | 显示用户登录地   |
| -u | 显示用户空闲的时间 |
| -T | 显示终端属性    |
| -m | 显示当前用户信息  |
| -q | 精简模式      |

选择性参数的具体说明如下：



| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◇◇◇ 范例范例

◎ 范例 192 ◎ 显示当前登录系统的用户。

```
[root@localhost ~]# who //显示当前登录系统的用户
root ttym1 May 31 15:15
root ttym2 May 31 15:15
root pts/1 May 31 15:07 (192.168.88.7)
root :0 May 31 15:15
[root@localhost ~]#
```

本例使用指令 who 显示当前登录系统的用户的信息。

◎ 范例 193 ◎ 显示标题栏。

```
[root@localhost ~]# who -H //显示标题栏
NAME LINE TIME COMMENT
root ttym1 May 31 15:15
root ttym2 May 31 15:15
root pts/1 May 31 15:07 (192.168.88.7)
root :0 May 31 15:15
[root@localhost ~]#
```

本例使用指令“who -H”显示登录系统的用户的信息，同时显示标题栏。

◎ 范例 194 ◎ 显示用户登录来源。

```
[root@localhost ~]# who -l -H //显示用户登录来源
NAME LINE TIME IDLE PID COMMENT
LOGIN ttym5 May 31 15:07 2337 id=5
LOGIN ttym6 May 31 15:07 2338 id=6
LOGIN ttym3 May 31 15:16 2931 id=3
LOGIN ttym4 May 31 15:16 2974 id=4
[root@localhost ~]#
```

本例使用指令“who -l -H”显示登录系统的用户的信息，同时显示用

户登录来源。

◎ 范例 195 ◎ 显示终端属性。

```
[root@localhost ~]# who -T -H
NAME LINE TIME COMMENT
root + tty1 May 31 15:15
root + tty2 May 31 15:15
root + pts/1 May 31 15:07 (192.168.88.7)
root ? :0 May 31 15:15
[root@localhost ~]#
```

本例使用指令“who -T -H”显示登录系统的用户的信息，同时显示终端属性。

◎ 范例 196 ◎ 只显示当前用户。

```
[root@localhost ~]# who -m -H //显示当前用户的信息
NAME LINE TIME COMMENT
root pts/1 May 31 15:07 (192.168.88.7)
[root@localhost ~]#
```

本例使用指令“who -m -H”显示当前登录用户的信息。

◎ 范例 197 ◎ 精简模式显示。

```
[root@localhost ~]# who -q //显示系统用户信息
root root root root
users=4
[root@localhost ~]#
```

本例使用指令“who -q”以精简模式显示系统用户信息。



users

## 1.99 whoami 指令：显示用户名



whoami[选择性参数]



whoami 指令用来显示当前用户名。



### ◆ 功能说明

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◆ 执行范例

- 范例 198 ● 显示用户名。

```
[root@localhost ~]# whoami //显示用户名
root
[root@localhost ~]#
```

本例使用指令 whoami 显示用户名。

### ◆ 相关指令

who

## 1.100

## whois 指令：显示指定用户信息

### ◆ 语法 whois[用户]

◆ 功能说明 whois 指令用来查找并显示指定用户账号的信息。

### ◆ 执行范例

- 范例 199 ● 显示指定用户信息。

```
[root@localhost ~]# whois root //查找 root 用户信息
```

本例使用指令“whois root”查找 root 用户的信息。

### ◆ 相关指令

who

## 1.101

## &指令：将任务放到后台执行

### ◆ 语法 指令&

## 1.101 &amp;指令：将任务放到后台执行

**功能说明** 在要执行的指令后加上&, 可以使得进程在后台执行。

**执行范例**

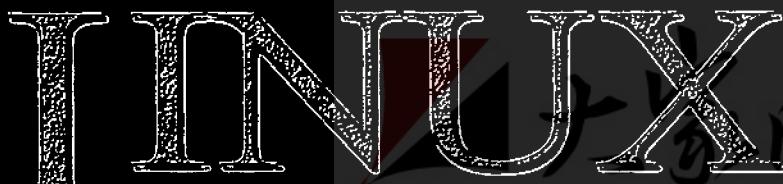
◎ 范例 200 ◎ 让进程在后台执行。

```
[root@localhost ~]# ftp 192.168.88.2 & //让进程在后台执行
[1] 9934
[root@localhost ~]# Connected to 192.168.88.2.
220 (vsFTPd 2.0.5)
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (192.168.88.2:root): iplab604
-bash: iplab604: command not found
[1]+ Stopped ftp 192.168.88.2
[root@localhost ~]#
```

本例使用指令“`ftp 192.168.88.2 &`”使得 `ftp` 进程在后台运行。

<http://www.TopSage.com>

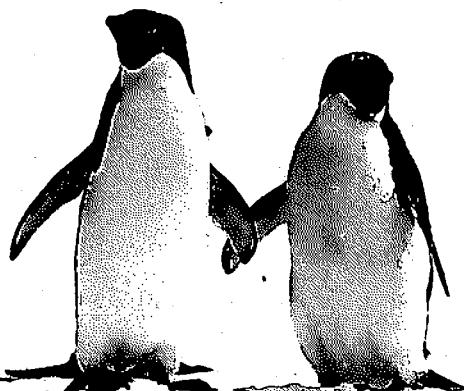




## 第2章

### 系统设置

系统设置指令主要用来设置系统的相关参数。常见的参数包括音效装置、定时器、系统时间、开机信息、色彩和环境变量等。系统设置指令是 Linux 系统中最常使用的一种指令，本章将详细讲解系统设置的相关指令。





## 2.1 alias 指令：设置指令的别名

**语法** alias[指令的别名]=[指令名]

**功能说明** alias 指令为已经存在的指令设置别名。设置别名之后，用户可以运行自己设置的别名来实现与已经存在指令相同的功能。这样就可以实现用较短的字符串来实现较长的而又常用的字符串的指令的功能。

系统重启之后，本次设置的别名将不再生效。要让设置永久生效，请查看并修改.profile 或.cshrc 文件。

**参数说明**

参数为系统中已有的相应的指令的名称或自己为指令取的别名。

若不输入任何参数，只输入 alias，则显示当前系统下已经设置的指令别名的情况。

**执行范例**

◎ 范例 201 ◎ 列出当前系统下已经设置的别名的情况。

```
root@localhost:~# alias //列出已设置的别名
alias ll='ls -l'
alias ls='ls --color=auto'
```

根据当前系统下别名的设置，指令“ll”相当于指令“ls -l”，指令“ls”相当于指令“ls -color=auto”。

◎ 范例 202 ◎ 设置别名。

```
root@localhost:~# alias a=cd //设置别名
root@localhost:~# alias //列出当前已经设置的别名
alias a='cd'
alias ll='ls -l'
alias ls='ls --color=auto'
```

经过设置，指令“a”相当于指令“cd”。

## 2.2 apmd 指令：高级电源管理

**语法** apmd[必要参数][选择性参数]

## 2.3 at 指令：系统将执行的任务排队

**功能说明** apmd 指令是高级电源管理子系统的守护进程，apm 子系统主要由电源管理硬件、APM BIOS 和内核中的驱动器组成。当电源不足时将发出警告信息。

**参数说明**

必要参数的具体说明如下：

| 参数         | 功 能             |
|------------|-----------------|
| -u utc     | 设置 BIOS 时钟为 UTC |
| -v verbose | 记录所有的电源管理事件     |
| -W wall    | 警告信息            |

选择性参数的具体说明如下：

| 参数         | 功 能                              |
|------------|----------------------------------|
| -p<百分比>    | 控制电池状态信息记录的频繁程度                  |
| -w<百分比>    | 当电池不是在充电时，电量下降速度超过指定的速度时，将产生警告信息 |
| -V version | 版本信息                             |
| -h help    | 帮助信息                             |

**执行范例**

● 范例 203 ● 记录所有的电源管理事件。

```
[root@localhost ~]# apmd -v //记录所有的电源管理事件
```

● 范例 204 ● 设置 BIOS 时钟。

```
[root@localhost ~]# apmd -u //设置 BIOS 时钟为 UTC
```

**相关命令**

apm、apmsleep

## 2.3 at 指令：系统将执行的任务排队

**语法** at[选择性参数][必要参数]时间

**功能说明** at 指令和 batch 指令从标准输入或者指定文件读取



指令，这些指令将在一段时间之后执行。

### ❖ 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能                           |
|-----|-------------------------------|
| -m  | 当指定的任务被完成之后，将给用户发送邮件，即使没有标准输出 |
| -l  | atq 的别名                       |
| -d  | atrm 的别名                      |
| -v  | 显示任务将被执行的时间                   |
| -c  | 打印任务的内容到标准输出                  |

选择性参数的具体说明如下：

| 参 数      | 功 能                 |
|----------|---------------------|
| -V       | 版本信息                |
| -q<队列>   | 使用指定的队列             |
| -f<文件>   | 从指定文件读入任务而不是从标准输入读入 |
| -t<时间参数> | 以时间参数的形式提交要运行的任务    |

### ❖ 执行范例

#### ◎ 范例 205 ◎ 设定任务在特定时间执行。

```
[root@localhost ~]# at 5pm tomorrow //设定任务在明天下午 5 点执行
at> echo Go for dinner with rich //打印 "Go for dinner with rich"
at> <EOT>
job 3 at 2007-05-06 17:00
[root@localhost ~]#
```

运行“at 5pm tomorrow”指令，设定任务在第二天下午 5 点执行，然后进入 at 指令的指令行，这样就可以设定要执行的任务了。此处输入指令“echo Go for dinner with rich”表示打印“Go for dinner with rich”。任务设定完成之后，可以输入“Ctrl+D”完成任务的设定。这样，明天下午 5 点，系统将打印信息提醒要与 rich 一块吃饭的事。

```
[root@localhost ~]# at 8am+3days //设定任务在 3 天后早上 8 点执行
at> /bin/date //执行 date 指令
```

## 2.3 at 指令：系统将执行的任务排队

```
at> <EOT>
job 4 at 2007-05-08 08:00
[root@localhost ~]#
```

设置三天后早上 8 点执行 date 指令。

◎ 范例 206 ◎ 查询已经设置的任务。

```
[root@localhost ~]# atq //查询已经设置的任务
3 2007-05-06 17:00 a root
4 2007-05-08 08:00 a root
```

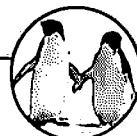
◎ 范例 207 ◎ 删 除已经设置的任务。

```
[root@localhost ~]# atm 3 //删除已经设置的任务
[root@localhost ~]#
```

删除已经设置的任务 3。

◎ 范例 208 ◎ 显示已经设置的任务的内容。

```
[root@localhost ~]# at -c 4 //显示已经设置的任务 4 的内容
#!/bin/sh
atrun uid=0 gid=0
mail root 0
umask 22
HOSTNAME=localhost.localdomain; export HOSTNAME
SHELL=/bin/bash; export SHELL
HISTSIZE=1000; export HISTSIZE
SSH_CLIENT=:ffff:192.168.88.7\ 1295\ 22; export SSH_CLIENT
QTDIR=/usr/lib/qt-3.3; export QTDIR
SSH_TTY=/dev/pts/1; export SSH_TTY
USER=root; export USER
LS_COLORS=no=00:fi=00:di=00\:;34:ln=00\:;36:pi=40\:;33:so=00\:;35:bd=40\:;
33\:;01:cd=40\:;33\:;01:or=01\:05\:;37\:;41:md=01\:05\:;37\:;41:ex=00\:;32:*.cmd=
00\:;32:*.exe=00\:;32:*.com=00\:;32:*.btm=00\:;32:*.bat=00\:;32:*.sh=00\:;3
2:*.csh=00\:;32:*.tar=00\:;31:*.tgz=00\:;31:*.arj=00\:;31:*.taz=00\:;31:*
.1zh=00\:;31:*.zip=00\:;31:*.z=00\:;31:*.2=00\:;31:*.gz=00\:;31:*.bz2=00\:;
31:*.bz=00\:;31:*.tz=00\:;31:*.rpm=00\:;31:*.cpio=00\:;31:*.jpg=00\:;35:*
.gif=00\:;35:*.bmp=00\:;35:*.xbm=00\:;35:*.xpm=00\:;35:*.png=00\:;35:*.tif
=00\:;35\:; export LS_COLORS
KDEDIR=/usr; export KDEDIR
MAIL=/var/spool/mail/root; export MAIL
PATH=/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/
bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root/bin; export PATH
```



```

INPUTRC=/etc/inputrc; export INPUTRC
PWD=/root; export PWD
LANG=zh_CN.UTF-8; export LANG
SSH_ASKPASS=/usr/libexec.openssh/gnome-ssh-askpass;export SSH_ASKPASS
SHLVL=1; export SHLVL
HOME=/root; export HOME
LOGNAME=root; export LOGNAME
SSH_CONNECTION=:ffff:192.168.88.7\ 1295\ ::ffff:192.168.88.76\ 22;
export SSH_CONNECTION
LESSOPEN=\\!/usr/bin/lesspipe.sh\\ ts; export LESSOPEN
G_BROKEN_FILERAMES=1; export G_BROKEN_FILERAMES
cd /root || (
 echo 'Execution directory inaccessible' >&2
 exit 1
)
${SHELL:-/bin/sh} << '(dd if=/dev/urandom count=200 bs=1 2>/dev/
null|LC_ALL=C tr -d -c [:alnum:])'
/bin/date
[root@localhost ~]#
```

显示已经设置的指令 4 的指令，也就是执行 date 指令。



batch、atq、atrm、cron、nice、sh、umaskatd

## 2.4

## atd 指令：执行已经排队的任务



atd[选择性参数][必要参数]



atd 指令执行被 at 指令排队的任务。



必要参数的具体说明如下：

| 参数 | 功能          |
|----|-------------|
| -d | 输出调试信息      |
| -s | 只执行已排队的任务一次 |

## 2.5 atq 指令：检查排队的任务

选择性参数的具体说明如下：

| 参 数      | 功 能                           |
|----------|-------------------------------|
| -l<负载参数> | 指定一个限制的负载参数，超过这个参数的任务将不被执行    |
| -b<时间>   | 指定两个 batch 任务之间的最小的间隔，默认是 60s |

## ☆ 执行范例

- ◎ 范例 209 ◎ 启动 atd。

```
[root@localhost ~]# atd -d //启动 atd
```

以输出调试信息的方式启动 atd。

## ☆ 相关指令

at、atrun、cron、crontab、syslog、at.deny、at.allow

## 2.5 atq 指令：检查排队的任务

## ☆ 功能 atq[选择性参数]

☆ 功能说明 atq 指令列出用户已设置排队的任务。如果用户是超级用户，那么所有的任务将被列出。标准输出的格式为：任务号，天数，小时数，工作类型。

## ☆ 功能说明

选择性参数的具体说明如下：

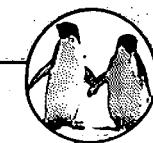
| 参 数    | 功 能  |
|--------|------|
| -V     | 版本信息 |
| -h     | 帮助信息 |
| -q<队列> | 指定队列 |

## ☆ 执行范例

- ◎ 范例 210 ◎ 查询已经设置的队列任务。

```
[root@localhost ~]# atq //显示已经设置的队列任务
```

```
6 2007-05-08 05:00 a root
4 2007-05-08 08:00 a root
```



## 第2章 系统设置

7 2007-05-08 08:00 a root

[root@localhost ~]#



at、atd、atrm、atrun

### 2.6

### atrm 指令：删除已经排队的任务



atrm[选择性参数]任务



atrm 指令用来删除已经排队的任务，任务将以任务号为标识。



选择性参数的具体说明如下：

| 参数     | 功能   |
|--------|------|
| -V     | 版本信息 |
| -h     | 帮助信息 |
| -q<任务> | 指定任务 |



○ 范例 211 ○ 删 除已经排队的任务。

[root@localhost ~]# atq

//显示当前已经设置的任务

6 2007-05-08 05:00 a root

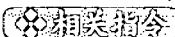
4 2007-05-08 08:00 a root

7 2007-05-08 08:00 a root

[root@localhost ~]# atrm 4

//删除任务 4

[root@localhost ~]#



at、atq、atrun、atd

### 2.7

### atrun 指令：执行已排队的任务



atrun[选择性参数]

## 2.8 aumix 指令：设置音效设备

**功能说明** atrun 指令运行 at 指令设置的任务。它是一个 shell 脚本，调用 /usr/sbin/atd 指令使用 “-s” 参数的情况。它向后兼容老版本的指令。

**参数说明**

选择性参数的具体说明如下：

| 参 数     | 功 能           |
|---------|---------------|
| -<负载参数> | 设置负载参数，超过将不执行 |

**执行范例**

◎ 范例 212 ◎ 执行已排队的任务。

```
[root@localhost ~]# atrun //执行已排队的任务
[root@localhost ~]#
```

**相关指令**

at、atd

## 2.8 aumix 指令：设置音效设备

**语法** aumix[必要参数][选择性参数]

**功能说明** aumix 指令用来调整音频设备的设置。它可以应用在命令行、脚本，可以与键盘或者鼠标进行交互。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能  |
|-----|------|
| -v  | 主音量  |
| -b  | 低音   |
| -c  | CD   |
| -i  | 进入排队 |
| -m  | 麦克风  |
| -o  | 跳出排队 |



续表

| 参数 | 功能      |
|----|---------|
| -p | PC 话筒   |
| -s | 合成器     |
| -t | 3 倍设置   |
| -w | PCM 设置  |
| -x | imix 设置 |
| -1 | 队列 1    |
| -2 | 队列 2    |
| -3 | 队列 3    |

选择性参数的具体说明如下：

| 参数     | 功能            |
|--------|---------------|
| -C<文件> | 指定包含颜色设置机制的文件 |
| -d<文件> | 指定音频设备名称      |
| -f<文件> | 指定保存和载入设置的文件  |

### 执行范例

◎ 范例 213 ◎ 设置音效设备。

```
[root@localhost root]# aumix //设置音效设备
```



在命令行下输入指令 aumix 后，将会出现上面的图形界面，用鼠标直接拖动上面的白色圆圈，将可以直接设置相应的选项。

### 相关指令

gpm、moused、sb、xaumix

## 2.9 authconfig 指令：配置系统的认证信息

**2.9 authconfig 指令：配置系统的认证信息**

authconfig[必要参数][选择性参数]



authconfig 指令提供了一个简单的方法设置 /etc/sysconfig/network 来处理 NIS、/etc/passwd 和 /etc/shadow。



必要参数的具体说明如下：

| 参数              | 功能         |
|-----------------|------------|
| -test           | 测试信息       |
| -enablecache    | 开启缓存设置     |
| -disablecache   | 关闭缓存设置     |
| -enablenis      | 开启 NIS 设置  |
| -disablenis     | 关闭 NIS 设置  |
| -enableldap     | 开启 LDAP 设置 |
| -enableldapauth | 开启 LDAP 认证 |
| -disableldap    | 关闭 LDAP 设置 |
| -enablekrb5     | 开启 KRB5 设置 |
| -disablekrb5    | 关闭 KRB5 设置 |
| -kickstart      | 不显示用户界面    |
| -probe          | 探测网络默认值    |

选择性参数的具体说明如下：

| 参数               | 功能          |
|------------------|-------------|
| -ldapserver<服务器> | 设置 LDAP 服务器 |
| -smbservers<服务器> | 设置 SMB 服务器  |
| -help            | 帮助信息        |
| -usage           | 显示主要用法      |

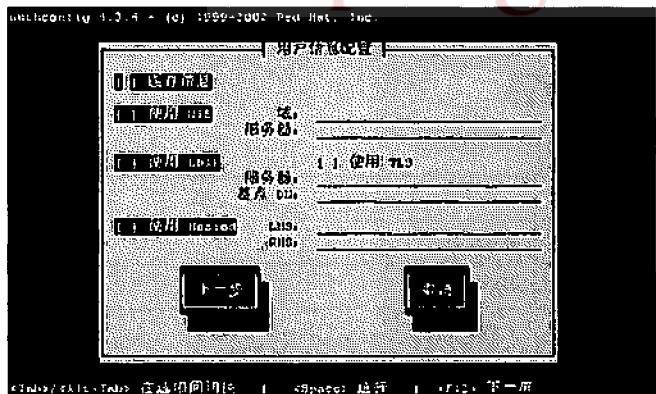


### ☆☆范例说明

◎ 范例 214 ◎ 设置系统的认证信息。

```
[root@localhost root]# authconfig //设置系统的认证信息
```

在命令行下输入指令 aumix 后，将会出现上面的图形界面，用鼠标直接拖动上面的白色圆圈，将可以直接设置相应的参数。



### ☆☆相关指令

passwd、shadow、pwconv、domainname、ypbind

## 2.10 bind 指令：显示或者设置按键组合

### ☆☆语法

**bind[必要参数][选择性参数]**

☆☆功能说明 bind 指令可以显示按键组合代表的功能，也可以设置使得某些按键组合代表某些特殊的功能。

### ☆☆参数说明

必要参数的具体说明如下：

| 参数 | 功    能           |
|----|------------------|
| -d | 显示按键组合的内容        |
| -l | 列出所有的功能          |
| -v | 列出当前设置的按键组合代表的功能 |

## 2.10 bind 指令：显示或者设置按键组合

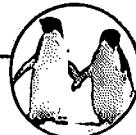
选择性参数的具体说明如下：

| 参 数      | 功 能             |
|----------|-----------------|
| -f<配置文件> | 读入特定的配置文件       |
| -m<按键组合> | 设定特殊的按键组合       |
| -q<功能>   | 列出指定功能的按键或者按键组合 |

 操作范例

## ◎ 范例 215 ◎ 显示按键组合的所有功能。

```
[root@localhost root]# bind -l //显示按键组合的内容
abort
accept-line
alias-expand-line
arrow-key-prefix
backward-byte
backward-char
backward-delete-char
backward-kill-line
backward-kill-word
backward-word
beginning-of-history
beginning-of-line
call-last-kbd-macro
capitalize-word
character-search
character-search-backward
clear-screen
complete
complete-command
complete-filename
complete-hostname
complete-into-braces
complete-username
.....
vi-yank-to
yank
yank-last-arg
```



yank-nth-arg

yank-pop

[root@localhost root]#

显示内容很多，中间部分用省略号省略掉了。

◎ 范例 216 ◎ 显示当前按键组合的设置。

```
[root@localhost root]# bind -v //显示当前按键组合的设置
set blink-matching-paren on
set byte-oriented off
set completion-ignore-case off
set convert-meta off
set disable-completion off
set enable-keypad off
set expand-tilde off
set history-preserve-point off
set horizontal-scroll-mode off
set input-meta on
set mark-directories on
set mark-modified-lines off
set mark-symlinked-directories on
set match-hidden-files on
set meta-flag on
set output-meta on
set page-completions on
set prefer-visible-bell on
set print-completions-horizontally off
set show-all-if-ambiguous off
set visible-stats off
set bell-style audible
set comment-begin #
set completion-query-items 100
set editing-mode emacs
set keymap emacs
[root@localhost root]#
```

◎ 范例 217 ◎ 列出指定功能的按键或按键组合。

[root@localhost root]# bind -q abort //列出功能 abort 的按键及按键组合

abort can be invoked via "\C-g", "\C-x\C-g", "\M-\C-g".

[root@localhost root]#

## 2.11 chkconfig 指令：设置系统的应用程序

列出功能 abort 的按键以及按键组合。

```
[root@localhost root]# bind -q accept-line //列出功能“accept-line”按键
以及按键组合
```

```
accept-line can be invoked via "\C-j", "\C-m".
[root@localhost root]#
```

列出功能 accept-line 的按键以及按键组合。



ls

## 2.11 chkconfig 指令：设置系统的应用程序

**语法**      `chkconfig[必要参数][选择性参数][系统程序][on/off/reset]`

**功能说明**    `chkconfig` 指令提供了一个维护/etc/rc[0~6].d 文件夹的命令行工具，它减轻了系统直接管理这些文件夹中的符号连接的负担。

`chkconfig` 主要包括 5 个原始功能：为系统管理增加新的服务、为系统管理移除服务、列出当前服务的启动信息、改变服务的启动信息和检查特殊服务的启动状态。当单独运行 `chkconfig` 指令而不加任何参数时，它将显示服务的使用信息。



必要参数的具体说明如下：

| 参数                 | 功 能                                |
|--------------------|------------------------------------|
| <code>-add</code>  | 开启指定的服务程序                          |
| <code>-del</code>  | 关闭指定的服务程序                          |
| <code>-list</code> | 列出 <code>chkconfig</code> 所知道的所有服务 |

选择性参数的具体说明如下：

| 参数                            | 功 能                                                                  |
|-------------------------------|----------------------------------------------------------------------|
| <code>-level&lt;代号&gt;</code> | 设置服务程序的等级代号，它是一串 0~7 的数字，如 “ <code>-level35</code> ” 代表指定运行等级为 3 和 5 |
| <code>-version</code>         | 版本信息                                                                 |
| <code>-help</code>            | 显示帮助信息                                                               |



### ◎ 执行范例

- ◎ 范例 218 ◎ 列出 chkconfig 所知道的所有的指令。

```
[root@localhost root]# chkconfig -list //列出chkconfig 所知道的所有服务
NetworkManager 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
NetworkManagerDispatcher 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
acpid 0:关闭 1:关闭 2:关闭 3:启用 4:启用 5:启用 6:关闭
 //此处省略了部分结果
isicom 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
kadmin 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
```

### 基于 xinetd 的服务

|              |    |
|--------------|----|
| amanda:      | 关闭 |
| amandaidx:   | 关闭 |
| amidxtape:   | 关闭 |
| auth:        | 关闭 |
| chargen:     | 关闭 |
| chargen-udp: | 关闭 |
| cups-lpd:    | 关闭 |
| cvs:         | 关闭 |
| daytime:     | 关闭 |
| daytime-udp: | 关闭 |
| echo:        | 关闭 |
| echo-udp:    | 关闭 |
| eklogin:     | 关闭 |
| finger:      | 关闭 |
| gssftp:      | 关闭 |
| klogin:      | 关闭 |
| krb5-telnet: | 关闭 |
| kshell:      | 关闭 |
| ktalk:       | 关闭 |
| ntalk:       | 关闭 |
| rexec:       | 关闭 |
| rlogin:      | 关闭 |
| rsh:         | 关闭 |
| rsync:       | 关闭 |
| swat:        | 关闭 |
| talk:        | 关闭 |

## 2.11 chkconfig 指令：设置系统的应用程序

|           |    |
|-----------|----|
| telnet:   | 关闭 |
| tftp:     | 关闭 |
| time:     | 关闭 |
| time-udp: | 关闭 |
| uucp:     | 关闭 |

现在可以看到系统中的服务的开启和关闭的情况。

### ◎ 范例 219 ◎ 开启服务。

```
[root@localhost root]# chkconfig telnet on //开启 Telnet 服务
[root@localhost root]# chkconfig -list //列出 chkconfig 所知的所有
```

的服务的情况

#### 基于 xinetd 的服务

|              |    |
|--------------|----|
| amanda:      | 关闭 |
| amandaidx:   | 关闭 |
| amidxtape:   | 关闭 |
| auth:        | 关闭 |
| chargen:     | 关闭 |
| chargen-udp: | 关闭 |
| cups-lpd:    | 关闭 |
| cvs:         | 关闭 |
| daytime:     | 关闭 |
| daytime-udp: | 关闭 |
| echo:        | 关闭 |
| echo-udp:    | 关闭 |
| eklogin:     | 关闭 |
| finger:      | 关闭 |
| gssftp:      | 关闭 |
| klogin:      | 关闭 |
| krb5-telnet: | 关闭 |
| kshell:      | 关闭 |
| ktalk:       | 关闭 |
| ntalk:       | 关闭 |
| rexec:       | 关闭 |
| rlogin:      | 关闭 |
| rsh:         | 关闭 |
| rsync:       | 关闭 |
| swat:        | 关闭 |



|           |    |
|-----------|----|
| talk:     | 关闭 |
| telnet:   | 开启 |
| tftp:     | 关闭 |
| time:     | 关闭 |
| time-udp: | 关闭 |
| uucp:     | 关闭 |

使用指令“`chkconfig telnet on`”开启 Telnet 服务后，可以使用指令“`chkconfig -list`”查看 Telnet 服务的开启、关闭情况。

◎ 范例 220 ◎ 关闭服务。

```
[root@localhost root]# chkconfig telnet off //关闭Telnet服务
[root@localhost root]# chkconfig -list //列出chkconfig所知的所有
的服务的情况
```

基于 xinetd 的服务

|              |    |
|--------------|----|
| amanda:      | 关闭 |
| amandaidx:   | 关闭 |
| amidxtape:   | 关闭 |
| auth:        | 关闭 |
| chargen:     | 关闭 |
| chargen-udp: | 关闭 |
| cups-lpd:    | 关闭 |
| cvs:         | 关闭 |
| daytime:     | 关闭 |
| daytime-udp: | 关闭 |
| echo:        | 关闭 |
| echo-udp:    | 关闭 |
| eklogin:     | 关闭 |
| finger:      | 关闭 |
| gssftp:      | 关闭 |
| klogin:      | 关闭 |
| krb5-telnet: | 关闭 |
| kshell:      | 关闭 |
| ktalk:       | 关闭 |
| ntalk:       | 关闭 |
| rexec:       | 关闭 |
| rlogin:      | 关闭 |

|           |    |
|-----------|----|
| rsh:      | 关闭 |
| rsync:    | 关闭 |
| swat:     | 关闭 |
| talk:     | 关闭 |
| telnet:   | 关闭 |
| tftp:     | 关闭 |
| time:     | 关闭 |
| time-udp: | 关闭 |
| uucp:     | 关闭 |

使用指令“chkconfig telnet off”关闭 Telnet 服务后，可以使用指令“chkconfig –list”查看 Telnet 服务的开启、关闭情况。

### ○○相关指令

init、ntsysv、serviceconf

## 2.12 chroot 指令：改变根目录

**○○语法** chroot[选择性参数][目的目录][执行指令]

**○○功能说明** chroot 指令用来改变根目录信息。执行 chroot 指令能增加系统的安全性，限制登录用户所做的事情。

### ○○参数说明

选择性参数的具体说明如下：

| 参数       | 功 能    |
|----------|--------|
| -help    | 显示帮助信息 |
| -version | 显示版本信息 |

### ○○范例与技巧

● 范例 221 ● 改变根目录。

[root@localhost root]#chroot /mnt/ls //改变根目录

改变用户的根目录，成功之后，用户登录时，默认的目录将是/mnt/ls。

### ○○相关指令

chmod



## 2.13 clock 指令：系统 RTC 时间设置

**语法** `clock[必要参数][选择性参数]`

**功能说明** `clock` 指令返回处理器时间的近似值。

**概念说明**

必要参数的具体说明如下：

| 参数                      | 功 能                         |
|-------------------------|-----------------------------|
| <code>-adjust</code>    | 自动调整硬件时钟时间                  |
| <code>-debug</code>     | 运行时，输出详细的处理过程               |
| <code>-directisa</code> | 不通过设备文件/etc/rtc，直接对硬件时钟进行存取 |
| <code>-getepoch</code>  | 输出硬件时钟的数值到标准输出              |
| <code>-hctosys</code>   | 时钟同步，使系统时钟和硬件时钟同步           |
| <code>-se--date</code>  | 设置硬件时间                      |
| <code>-show</code>      | 显示硬件时钟到标准输出                 |
| <code>-systohc</code>   | 系统时间写入硬件时钟                  |
| <code>-test</code>      | 进行测试                        |
| <code>-utc</code>       | 将硬件时钟的时间设置为 UTC             |

选择性参数的具体说明如下：

| 参数                                        | 功 能       |
|-------------------------------------------|-----------|
| <code>-stepoch-epoch&gt;&lt;年份&gt;</code> | 设置硬件时钟的年份 |
| <code>-version</code>                     | 版本信息      |

### 【概念说明】

UTC 时间：UTC 为 Universal Time Coordinated 的缩写，GPS 系统中有两种时间区分，一种为 UTC，另一种为 LT（地方时）两者的区别为时区不同。UTC 就是 0 时区的时间，地方时为本地时间，如北京为早上 8 点（东

八区)，UTC 时间就为 0 点，比北京晚 8h，以此计算即可。

### ◎ 实例范例

- ◎ 范例 222 ◎ 获取当前的时间。

```
[root@localhost root]# clock //获取当前的时间
2007年05月09日 星期三 22时11分50秒 -0.912898 seconds
```

- ◎ 范例 223 ◎ 显示 UTC 时间。

```
[root@localhost root]# clock -utc //显示 UTC 时间
2007年05月10日00星期四 06时17分03秒 -0.740993 seconds
[root@localhost root]#
```

### ◎ 命令指令

getusage、times

## 2.14 crontab 指令：设置计时器

 crontab[选择性参数][配置文件][必要参数]

 **功能说明** crontab 指令是用来安装、卸载或者列出驱动 cron 程序的表。每一个用户可以拥有自己的 crontab，配置文件存在/var 下面，不能被直接编辑。

### ◎ 功能说明

必要参数的具体说明如下：

| 参数 | 功能                 |
|----|--------------------|
| -e | 设置计时器              |
| -l | 列出当前计时器的设置         |
| -r | 删除计时器设置            |
| -i | 交互式模式，删除计时器设置时先要询问 |

选择性参数的具体说明如下：

| 参数    | 功能        |
|-------|-----------|
| <用户名> | 指定计时器的用户名 |



大家网

### ◆ 执行范例

- 范例 224 ◎ 列出当前的 crontab 文件的设置。

```
[root@localhost kk]#crontab -l //列出当前的 crontab 设置
```

本例使用指令“`crontab -l`”列出当前的 crontab 文件的内容。

- 范例 225 ◎ 编辑当前 crontab 文件的内容。

```
[root@localhost kk]#crontab -e //编辑当前的 crontab 设置
```

本例使用指令“`crontab -e`”编辑当前的 crontab 文件的内容。

- 范例 226 ◎ 删除当前的 crontab 设置。

```
[root@localhost kk]#crontab -r //编辑当前的 crontab 设置
```

本例使用指令“`crontab -r`”删除当前的 crontab 设置。

### 【指令中时间段的设置】

用户建立的文件 crontab 将存储于 /var/spool/cron 中, 格式为: 时间段 时间段 时间段 时间段 时间段 指令。时间格式如下:

第一段 代表分钟 取值 0~59;

第二段 代表小时 取值 0~23;

第三段 代表日期 取值 1~31;

第四段 代表月份 取值 1~12;

第五段 代表星期 取值 0~6。

### ◆ 相关指令

`crontab`、`cron`

## 2.15 declare 指令: 显示或者设定 shell 变量

### ◆ 命令

`declare[+/-][必要参数][选择性参数]`

### ◆ 功能说明

declare 指令用来设定 shell 变量。

### ◆ 参数说明

必要参数的具体说明如下:

| 参数 | 功能     |
|----|--------|
| +  | 设置指定属性 |
| -  | 删除指定属性 |

## 2.15 declare 指令：显示或者设定 shell 变量

续表

| 参数 | 功能     |
|----|--------|
| r  | 只读模式   |
| -f | 只显示函数  |
| x  | 设置环境变量 |

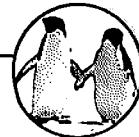
选择性参数的具体说明如下：

| 参数        | 功能   |
|-----------|------|
| -help     | 帮助信息 |
| --version | 版本信息 |

### 范例 227

#### ○ 范例 227 ○ 显示 shell 变量。

```
[root@localhost temp]# declare //显示 shell 变量设置
BASH=/bin/bash
BASH_ENV=/root/.bashrc
BASH_VERSINFO=([0]="2" [1]="05b" [2]="0" [3]="1" [4]="release"
[5]="i386-redhat-linux-gnu")
BASH_VERSION='2.05b.0(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=124
DIRSTACK=()
EUID=0
GROUPS=()
G_BROKEN_FILERAMES=1
HISTFILE=/root/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/root
HOSTNAME=localhost.localdomain
HOSTTYPE=i386
IFS=$' \t\n' //此处省略了部分结果
TERM=xterm
UID=0
```



```
USER=root
USERNAME=root
_=cu
[root@localhost temp]#
```

本例使用指令 declare，显示 shell 变量设置。

### 图标指令

export

## 2.16 depmod 指令：模块关系

**功能说明** depmod[必要参数][选择性参数][模块]

**功能说明** depmod 指令用来显示模块间的关系。

**功能说明**

必要参数的具体说明如下：

| 参数 | 功 能           |
|----|---------------|
| -a | 包含所有模块的关系     |
| -d | 输出调试信息        |
| -e | 输出符号表不能识别的符号  |
| -i | 忽略符号表的版本      |
| -s | 错误信息记录        |
| -v | 运行时，报告详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功 能         |
|----------|-------------|
| -m<文件>   | 指定文件做为符号表文件 |
| -help    | 帮助信息        |
| -version | 版本信息        |

### 执行范例

- 范例 228 ○ 显示可用模块。

## 2.17 dircolors 指令：ls 指令对应的显示颜色设置

[root@localhost temp]# depmod -a

//显示可用模块

本例使用指令“depmod -a”显示可用模块。



lsmod

## 2.17

### dircolors 指令：ls 指令对应的显示颜色设置



dircolors[必要参数][选择性参数]



dircolors 指令用来设置 ls 指令对应的显示颜色。



必要参数的具体说明如下：

| 参数 | 功 能     |
|----|---------|
| -p | 显示预设值信息 |

选择性参数的具体说明如下：

| 参数       | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



◎ 范例 229 ◎ 显示默认值。

```
[root@localhost ~]# dircolors -p //显示默认值
Configuration file for dircolors, a utility to help you set the
LS_COLORS environment variable used by GNU ls with the --color option.
The keywords COLOR, OPTIONS, and EIGHTBIT (honored by the
slackware version of dircolors) are recognized but ignored.
Below, there should be one TERM entry for each termtype that is colorizable
TERM linux
TERM linux-c
```



```

TERM mach-color
TERM console
TERM con132x25

.mpeg 01;35 //此处省略了部分结果
.avi 01;35
.fli 01;35
.g1 01;35
.d1 01;35
.xcf 01;35
.xwd 01;35
audio formats
.ogg 01;35
.mp3 01;35
.wav 01;35
[root@localhost ~]#

```

本例使用指令“`dircolors -p`”显示颜色设置的默认值。

搜索命令

dir

## 2.18 dmesg 指令：显示开机信息

`dmesg[必要参数][选择性参数]`

`dmesg` 指令用来检测和控制内核 ring 缓冲区。它帮助用户输出开机信息。

必要参数的具体说明如下：

| 参    数          | 功    能         |
|-----------------|----------------|
| <code>-c</code> | 显示信息后清空缓冲区中的信息 |

选择性参数的具体说明如下：

## 2.18 dmesg 指令：显示开机信息

| 参数      | 功能                    |
|---------|-----------------------|
| -s<字节数> | 缓冲区大小的设置，默认是 16392 字节 |
| -n<层级>  | 记录层级设置                |

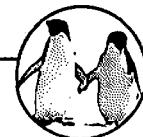
**◎ 实例范例****◎ 范例 230 ◎ 显示开机信息。**

```
[root@localhost root]# dmesg //显示开机信息
Linux version 2.4.20-8 (bhcompile@porky.devel.redhat.com) (gcc version
3.2.2 20030222 (Red Hat Linux 3.2.2-5)) #1 Thu Mar 13 17:54:28 EST 2003
BIOS-provided physical RAM map:
BIOS-e820: 0000000000000000 - 000000000009f800 (usable)
BIOS-e820: 000000000009f800 - 00000000000a0000 (reserved)
BIOS-e820: 00000000000ca000 - 00000000000cc000 (reserved)
BIOS-e820: 00000000000dc000 - 0000000000100000 (reserved)
BIOS-e820: 0000000000100000 - 0000000000fef0000 (usable)
BIOS-e820: 0000000000fef0000 - 0000000000fefc000 (ACPI data)
BIOS-e820: 0000000000fefc000 - 000000000ff00000 (ACPI NVS)
BIOS-e820: 000000000ff00000 - 0000000010000000 (usable)
BIOS-e820: 000000000fec00000 - 000000000fec10000 (reserved)
BIOS-e820: 000000000fee00000 - 000000000fee01000 (reserved)
BIOS-e820: 000000000fffe0000 - 0000000100000000 (reserved)

0MB HIGHMEM available.
256MB LOWMEM available.

On node 0 totalpages: 65536
zone(0): 4096 pages.
zone(1): 61440 pages.
zone(2): 0 pages.

Kernel command line: ro root=LABEL=/
Initializing CPU#0
Detected 3389.095 MHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 6763.31 BogoMIPS
Memory: 252832k/262144k available (1347k kernel code, 6812k reserved, 999k
data, 132k init, 0k highmem)
Dentry cache hash table entries: 32768 (order: 6, 262144 bytes)
Inode cache hash table entries: 16384 (order: 5, 131072 bytes)
```



Mount cache hash table entries: 512 (order: 0, 4096 bytes)  
 Buffer-cache hash table entries: 16384 (order: 4, 65536 bytes)

lp0: using parport0 (polling).  
 lp0: console ready  
 mttr: your processor doesn't support write-combining  
 [root@localhost root]#

syslogd

## 2.19

### enable 指令：可用的 shell 内置指令

**◎ 活学活用** enable[必要参数][选择性参数]

**◎ 功能说明** enable 指令用来开启或者关闭 shell 所内置的指令。

**◎ 专家提醒**

必要参数的具体说明如下：

| 参 数 | 功 能                    |
|-----|------------------------|
| -a  | 显示所有的已经开启的 shell 内置的指令 |

选择性参数的具体说明如下：

| 参 数    | 功 能     |
|--------|---------|
| -n<指令> | 关闭指定的指令 |

**◎ 操作范例**

◎ 范例 231 ◎ 显示 shell 内置指令。

```
root@localhost:~# enable //显示shell指令
enable .
enable :
enable [
enable alias
enable bg
enable bind
```

## 2.19 enable 指令：可用的 shell 内置指令

```
enable break
enable builtin
enable caller
enable cd
enable command
enable compgen
enable complete
enable continue
enable declare
enable dirs
enable disown
enable echo
enable enable
enable eval
enable exec
enable exit
enable export
enable false
enable fc
enable fg
enable getopt
enable hash
enable help
enable history
enable jobs
enable kill
enable let
enable local
enable logout
enable popd
enable printf
enable pushd
enable pwd
enable read
enable readonly
enable return
enable set
enable shift
enable shopt
```



```
enable source
enable suspend
enable test
enable times
enable trap
enable true
enable type
enable typeset
enable ulimit
enable umask
enable unalias
enable unset
enable wait
root@localhost:~#
```

本例使用指令 enable 显示已经开启的 shell 的内置指令。

#### 相关指令

exec

## 2.20

## eval 指令：连接多个指令

#### 语法

eval[指令]

功能说明 eval 指令用来连接多个指令，连接的多个指令将同时执行。

#### 执行范例

◎ 范例 232 ◎ 连接多个指令。

```
[root@localhost ~]# eval enable;ls //连接多个指令
enable .
enable :
enable [
enable alias
enable bg
enable bind
enable break
enable builtin
enable caller
```



## 2.21 export 指令：设置或者显示环境变量

//此处省略了部分结果

```

enable wait
123 inter
123.pcx inter_backup1
1.log inter_backup2
1.log.new lib
2.log lib.a
456 lib.new
7.log lost+found
abc mbox
anaconda-ks.cfg memtest86+-1.55.1
config-2.6.11-1.1369_FC4 nc
ddd nohup.out
DEADJOE restoresymtable
Desktop sushe
exampleseed System.map-2.6.11-1.1369_FC4
file1 temp
file2 test
gmon.out uhvbo8ac.pfa
grub uhvbo8ac.pfb
index.html vmlinuz-2.6.11-1.1369_FC4
initrd-2.6.11-1.1369_FC4.img workspace
install.log xen.gz
install.log.new xen-syms
install.log.syslog xuehao
[root@localhost ~]#

```

本例使用指令“eval enable;ls”连接enable指令和ls指令，使其同时执行，指令之间用分号隔开。



enable

## 2.21 export 指令：设置或者显示环境变量



export[必要参数][环境变量]=[变量值]



通过export指令可以设置环境变量值。



### 3.3 参数说明

必要参数的具体说明如下：

| 参数 | 功能          |
|----|-------------|
| -f | 代表环境变量为函数名称 |
| -n | 删除指定的环境变量   |
| -p | 列出所有的环境变量   |

### 3.4 执行范例

○ 范例 233 ○ 列出当前所有的环境变量。

```
[root@localhost ~]# export -p //列出当前的环境变量值
declare -x G_BROKEN_FILERAMES="1"
declare -x HISTSIZE="1000"
declare -x HOME="/root"
declare -x HOSTNAME="localhost.localdomain"
declare -x INPUTRC="/etc/inputrc"
declare -x KDEDIR="/usr"
declare -x LANG="zh_CN.UTF-8"
declare -x LESSOPEN="|/usr/bin/lesspipe.sh %s"
declare -x LOGNAME="root"
declare -x LS_COLORS="no=00:fi=00:di=00;34:ln=00;36:pi=40;33: so=00;
35:bd=40;33:01:cd=40;33:01:or=01;05:37:41:mi=01;05:37:41:ex=00;32:*.cmd=00
;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32
:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*
z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*
.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.pn
g=00;35:*.tif=00;35:*
declare -x MAIL="/var/spool/mail/root"
declare -x OLDPWD
declare -x PATH="/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:
/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root/bin"
declare -x PWD="/root"
declare -x QTDIR="/usr/lib/qt-3.3"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_ASKPASS="/usr/libexec.openssh/gnome-ssh-askpass"
```

## 2.21 export 指令：设置或者显示环境变量

```
declare -x SSH_CLIENT="::ffff:192.168.88.7 4976 22"
declare -x SSH_CONNECTION="::ffff:192.168.88.7 4976 ::ffff:192.
168.88.76 22"
declare -x SSH_TTY="/dev/pts/1"
declare -x TERM="xterm"
declare -x USER="root"
[root@localhost ~]#
```

本例使用指令“`export -p`”列出所有的环境变量。

○范例 234 ○ 定义环境变量。

```
[root@localhost ~]# export MYENV //定义环境变量
[root@localhost ~]# export -p //列出当前的环境变量
declare -x G_BROKEN_FILERAMES="1"
declare -x HISTSIZE="1000"
declare -x HOME="/root"
declare -x HOSTNAME="localhost.localdomain"
declare -x INPUTRC="/etc/inputrc"
declare -x KDEDIR="/usr"
declare -x LANG="zh_CN.UTF-8"
declare -x LESSOPEN="|/usr/bin/lesspipe.sh %s"
declare -x LOGNAME="root"
declare -x LS_COLORS="no=00:fi=00:di=00;34:ln=00;36:pi=40;33: so=00;35:
bd=40;33:01:cd=40;33:01:or=01;05;37;41:mi=01;05;37;41:ex=00;32:*.cmd=00;32
:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.2=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:*
declare -x MAIL="/var/spool/mail/root"
declare -x MYENV
declare -x OLDPWD
declare -x PATH="/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root/bin"
declare -x PWD="/root"
declare -x QTDIR="/usr/lib/qt-3.3"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
```



```

declare -x SSH_ASKPASS="/usr/libexec/openssh/gnome-ssh-askpass"
declare -x SSH_CLIENT="::ffff:192.168.88.7 4976 22"
declare -x SSH_CONNECTION="::ffff:192.168.88.7 4976 ::ffff:192.
168.88.76 22"
declare -x SSH_TTY="/dev/pts/1"
declare -x TERM="xterm"
declare -x USER="root"
[root@localhost ~]#

```

本例使用指令“`export MYENV`”定义了新的环境变量 `MYENV`，然后运行指令“`export -p`”，发现当前的环境变量列表中多了一个环境变量 `MYENV`。

#### ◎ 范例 235 ◎ 定义环境变量赋值。

```

[root@localhost ~]# export MYENV=7 //定义环境变量并赋值
[root@localhost ~]# export -p
declare -x G_BROKEN_FILERAMES="1"
declare -x HISTSIZE="1000"
declare -x HOME="/root"
declare -x HOSTNAME="localhost.localdomain"
declare -x INPUTRC="/etc/inputrc"
declare -x KDEDIR="/usr"
declare -x LANG="zh_CN.UTF-8"
declare -x LESSOPEN="| /usr/bin/lesspipe.sh %s"
declare -x LOGNAME="root"
declare -x LS_COLORS="no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so= 00;35:
bd=40;33:01:cd=40;33:01:or=01;05;37:41:mi=01;05;37:41:ex=00;32:*.cmd=00;32
:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.t
ar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;
31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio
=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;
35:*.tif=00;35:*
declare -x MAIL="/var/spool/mail/root"
declare -x MYENV="7"
declare -x OLDPWD
declare -x PATH="/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/
local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root/bin"
declare -x PWD="/root"
declare -x QTDIR="/usr/lib/qt-3.3"

```

## 2.22 false 指令：不做任何事情，表示失败

```

declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_ASKPASS="/usr/libexec.openssh/gnome-ssh-askpass"
declare -x SSH_CLIENT="::ffff:192.168.88.7 4976 22"
declare -x SSH_CONNECTION="::ffff:192.168.88.7 4976 ::ffff:192.
168.88.7 22"
declare -x SSH_TTY="/dev/pts/1"
declare -x TERM="xterm"
declare -x USER="root"
[root@localhost ~]#

```

本例使用指令“`export MYENV=7`”定义了新的环境变量 `MYENV`，并且给它赋值为 7，然后运行指令“`export -p`”，发现当前的环境变量列表中多了一个环境变量 `MYENV`。



dc

## 2.22

## false 指令：不做任何事情，表示失败

`false[选择性参数]``false` 指令将以一个状态码结束，状态码表示失败。

选择性参数的具体说明如下：

| 参 数                   | 功 能  |
|-----------------------|------|
| <code>-help</code>    | 帮助信息 |
| <code>-version</code> | 版本信息 |



◎ 范例 236 ◎ 表示失败。

|                                           |         |
|-------------------------------------------|---------|
| <code>[root@localhost ~]# true</code>     | //设置状态码 |
| <code>[root@localhost ~]# echo \$?</code> | //显示状态码 |
| 0                                         |         |
| <code>[root@localhost ~]# false</code>    | //设置状态码 |



```
[root@localhost ~]# echo $? //显示状态码
1
```

[root@localhost ~]#

本例使用指令 false 和 true 来设置状态码，表示失败或者成功。



true

## 2.23 fbset 指令：设置帧缓冲区



**fbset[必要参数][选择性参数]**

 **功能说明** fbset 指令是一个系统工具，用来显示或者更改帧缓冲设备的设置。帧缓冲设备提供一个简单和惟一的接口，来实现不同种类的图像显示。



必要参数的具体说明如下：

| 参数    | 功能             |
|-------|----------------|
| -a    | 改变所有设备的显示终端    |
| -i    | 显示当前帧缓冲区的详细设置  |
| -n    | 改变显示模式         |
| -s    | 列出当前的设置        |
| -v    | 程序运行时显示详细的处理信息 |
| -test | 测试而不改变模式       |

选择性参数的具体说明如下：

| 参数         | 功能                  |
|------------|---------------------|
| -db<文件>    | 读取指定的文件作为配置         |
| -ifb<设备代号> | 以指定设备的设置为参考，设置当前的设备 |
| -ofb<设备代号> | 等于“fb”              |
| -version   | 版本信息                |
| -help      | 帮助信息                |

## 2.24 hash 指令：显示和清除哈希表

## 技术范例

- ◎ 范例 237 ◎ 设置画面的分辨率和桌面的分辨率。

```
[root@localhost ~]# fbset -g 800 600 1024 768 //设置分辨率
```

本例使用指令“`fbset -g 800 600 1025 768`”设置画面的分辨率为`800*600`，桌面的分辨率为`1024*768`。

- ◎ 范例 238 ◎ 启动硬件文本加速。

```
[root@localhost ~]# fbset -accel true //启动硬件文本加速
```

本例使用指令“`fbset -accel true`”启动硬件文本加速。

- ◎ 范例 239 ◎ 启动广播功能。

```
[root@localhost ~]# fbset -bcast true //启动广播功能
```

## 相关命令

`fbdev`

## 2.24 hash 指令：显示和清除哈希表

## 命令

`hash[必要参数][选择性参数]`

功能说明 `hash` 指令用来显示和清除哈希表，执行指令的时候，系统将先查询哈希表。

## 大众点评

必要参数的具体说明如下：

| 参数 | 功    能 |
|----|--------|
| -l | 显示哈希表  |
| -r | 清除哈希表  |

选择性参数的具体说明如下：

| 参数     | 功    能    |
|--------|-----------|
| -d<名称> | 清除哈希表     |
| -p<路径> | 向哈希表中增加内容 |
| -t<指令> | 显示指令的完整路径 |



### 执行范例

◎ 范例 240 ◎ 显示哈希表。

```
[root@localhost ~]# hash -l //显示哈希表
builtin hash -p /sbin/halt halt
builtin hash -p /usr/bin/man man
[root@localhost ~]#
```

本例使用指令“hash -l”显示哈希表。

◎ 范例 241 ◎ 显示指令的完整路径。

```
[root@localhost ~]# hash -t halt //显示halt指令的完整路径
/sbin/halt
[root@localhost ~]#
```

本例使用指令“hash -t halt”显示 halt 指令的完整路径。

◎ 范例 242 ◎ 向哈希表中增加内容。

```
[root@localhost ~]# hash -p /usr/bin/aclocal aclocal //向哈希表中增加
aclocal条目
[root@localhost ~]# hash -l //显示哈希表
builtin hash -p /sbin/halt halt
builtin hash -p /usr/bin/aclocal aclocal
builtin hash -p /usr/bin/man man
builtin hash -p /bin/ls ls
[root@localhost ~]#
```

本例使用指令“hash -p /usr/bin/aclocal aclocal”向哈希表中添加条目 aclocal。

◎ 范例 243 ◎ 删除哈希表内容。

```
[root@localhost ~]# hash -r //删除哈希表内容
[root@localhost ~]# hash -l //显示哈希表内容
hash: hash table empty
[root@localhost ~]#
```

本例使用指令“hash -r”删除哈希表的内容。

### 相关指令

alias

## 2.25

## hostid 指令：打印出当前主机的标识

### 备注

hostid[选择性参数]

## 2.26 hostname 指令：显示或者设置当前系统的主机名

**功能说明** hostid 指令用来打印当前主机的数字化标识。

**参数说明**

选择性参数的具体说明如下：

| 参 数       | 功 能  |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |

**执行范围**

◎ 范例 244 ◎ 显示主机标识。

```
[root@localhost shal]# hostid //显示主机标识
7f0100
```

[root@localhost shal]#

本例使用指令 hostid 显示主机标识。

**相关指令**

host

## 2.26

## hostname 指令：显示或者设置当前系统的主机名

**语法** hostname[必要参数][选择性参数]

**功能说明** hostname 指令用来显示或者设置当前系统的主机名，主机名被许多网络程序使用，用来标识主机。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能    |
|-----|--------|
| -s  | 短主机名   |
| -a  | 别名     |
| -i  | IP 地址  |
| -f  | 长主机名   |
| -d  | DNS 域名 |



续表

| 参数 | 功能           |
|----|--------------|
| -y | NIS/YP 域名    |
| -v | 运行时显示详细的处理过程 |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -F<文件>   | 读取指定文件 |
| -help    | 帮助信息   |
| -version | 版本信息   |

#### ○○○执行范例

○ 范例 245 ○ 显示主机名。

```
[root@localhost shal]# hostname
localhost.localdomain
[root@localhost shal]#
```

本例使用指令 `hostname` 显示主机名称。

○ 范例 246 ○ 显示短主机名。

```
[root@localhost shal]# hostname -s
localhost
[root@localhost shal]#
```

本例使用指令 “`hostname -s`” 显示短主机名称。

○ 范例 247 ○ 显示主机别名。

```
[root@localhost shal]# hostname -a
localhost
[root@localhost shal]#
```

本例使用指令 “`hostname -a`” 显示主机别名。

○ 范例 248 ○ 显示主机 IP 地址。

```
[root@localhost shal]# hostname -i
127.0.0.1
[root@localhost shal]#
```

本例使用指令 “`hostname -i`” 显示主机 IP 地址。

○ 范例 249 ○ 显示主机域名。

## 2.27 hwclock 指令：显示与设定硬件时钟 ( rtc )

```
[root@localhost shal]# hostname -d //显示主机域名
localdomain
```

```
[root@localhost shal]#
```

本例使用指令“hostname -d”显示主机域名。

- ◎ 范例 250 ◎ 显示 NIS/YP 域名。

```
[root@localhost shal]# hostname -y //显示NIS/YP域名
(none)
```

```
[root@localhost shal]#
```

本例使用指令“hostname -y”显示主机的 NIS/YP 域名。

- ◎ 范例 251 ◎ 设置主机名称。

```
[root@localhost shal]# hostname apple //设置主机名称
[root@localhost shal]# hostname
apple
```

```
[root@localhost shal]#
```

本例使用指令“hostname apple”设置主机名称。

### 相关命令

host、uname、ifconfig

## 2.27

## hwclock 指令：显示与设定硬件时钟 ( rtc )

 **语法**      hwclock[必要参数][选择性参数]

 **功能说明**    hwclock 是一个管理硬件时钟的工具，可以显示当前的系统时间，可以设定指定时间到硬件时钟。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功    能  |
|----|---------|
| -a | 校正时间    |
| -D | 显示详细的信息 |
| -r | 显示硬件时间  |
| -s | 设置时间    |



续表

| 参数         | 功能     |
|------------|--------|
| -u         | 格尼威治时间 |
| -w         | 调整硬件时间 |
| --debug    | 输出调试时间 |
| --frectisa | 存取硬件时间 |
| --test     | 测试     |

选择性参数的具体说明如下：

| 参数              | 功能   |
|-----------------|------|
| -v              | 版本信息 |
| --set-date=<时间> | 设置时间 |

### ◆ 实例与范例 ◆

- 范例 252 ○ 显示当前时间。

```
root@localhost:~# hwclock //显示当前时间
Wed 23 May 2007 03:37:48 AM CST -0.588086 seconds
root@localhost:~#
```

本例使用指令 hwclock 显示当前时间。

- 范例 253 ○ 显示格林尼治时间。

```
root@localhost:~# hwclock -u //显示格林尼治时间
Wed 23 May 2007 03:38:44 AM CST -0.763379 seconds
root@localhost:~#
```

本例使用指令“hwclock -u”显示格林尼治时间。

- 范例 254 ○ 显示版本信息。

```
root@localhost:~# hwclock -v //显示版本信息
hwclock from util-linux-2.12r.
root@localhost:~#
```

本例使用指令“hwclock -v”显示 hwclock 的版本信息。

### ◆ 相关命令 ◆

adjtimex、date、gettimeofday、settimeofday、crontab、tzset

**2.28****insmod 指令：载入模块**

 **语法**    `insmod[必要参数][选择性参数][模块文件]`

 **功能说明**    `insmod` 指令用来向内核中添加模块。大部分用户可能会使用 `modprobe` 指令，它使用起来更方便一些。

 **参数说明**

必要参数的具体说明如下：

| 参数              | 功能            |
|-----------------|---------------|
| <code>-f</code> | 强制加载模块        |
| <code>-k</code> | 模块的自动卸载设置     |
| <code>-m</code> | 模块加载信息        |
| <code>-p</code> | 输出模块加载结果      |
| <code>-s</code> | 记录 log 信息     |
| <code>-v</code> | 运行时，显示详细的处理信息 |
| <code>-X</code> | 模块符号          |

选择性参数的具体说明如下：

| 参数                        | 功能     |
|---------------------------|--------|
| <code>&lt;模块名称&gt;</code> | 指定模块名称 |

 **执行范例**

● 范例 255 ● 加载模块。

`[root@localhost drivers]#insmod abc.o //加载模块`

本例使用指令 “`insmod abc.o`” 可以向内核加载模块。

 **相关指令**

`modprobe`

**2.29****isosize 指令：显示 iso9660 文件系统信息**

**◎语法** isosize[必要参数][选择性参数][文件]

**◎功能说明** isosize 指令输出 iso9660 文件系统中的文件的大小。文件可能是正常的文件或者是一个块装置，如/dev/hdd 或者/dev/sr0。

**◎参数说明**

必要参数的具体说明如下：

| 参数 | 功 能         |
|----|-------------|
| -x | 以用户易识别的方式显示 |

选择性参数的具体说明如下：

| 参数     | 功 能                          |
|--------|------------------------------|
| -d[数目] | 与“-x”不同时使用，显示的大小是实际大小除以指定的数目 |

**◎范例说明**

○范例 256 ○ 显示 iso9660 文件的大小。

`root@localhost:~#isosize /dev/hdd //显示文件大小`

本例使用指令“isosize /dev/hdd”显示文件/dev/hdd 的大小。

**◎相关指令**

du

**2.30****kbdconfig 指令：设置键盘类型**

**◎语法** kbdconfig[必要参数]

**◎功能说明** kbdconfig 指令进行相应的键盘设置，提供图形化的操作界面。

**◎参数说明**

必要参数的具体说明如下：

## 2.31 ldconfig 指令：配置动态链接绑定

| 参数     | 功能       |
|--------|----------|
| -test  | 测试       |
| --back | back 键设置 |

## 指令说明

○范例 257 ○ 键盘设置。

`[root@localhost root]# kdbconfig` //设置键盘

本例使用指令 kdbconfig 设置键盘。

## 相关指令

timeconfig

## 2.31 ldconfig 指令：配置动态链接绑定

## 基本命令

ldconfig[必要参数][选择性参数]

功能说明 ldconfig 指令根据在命令行中指定的文件库，产生与其相对应的链接，包含在/etc/ld.so.conf 中。当要确定链接的版本号时，ldconfig 指令将验证存在于函数库中的头文件。查找函数库时，ldconfig 忽略符号链接。

## 参数说明

必要参数的具体说明如下：

| 参数              | 功能                          |
|-----------------|-----------------------------|
| -v -verbose     | 当查找和产生符号链接时，打印出每个文件的名字和符号链接 |
| -n              | 只处理在命令行中指定的文件               |
| -N              | 不更新缓存。除非同时给定“-X”，否则链接仍旧被更新  |
| -X              | 不更新链接。除非同时给定“-N”，否则缓存仍旧被更新  |
| -l              | 函数库模式。根据用户指定来链接个库文件         |
| -p -print-cache | 打印缓冲区中的目录列表和函数库             |

选择性的参数的具体说明如下：



| 参数                | 功能                        |
|-------------------|---------------------------|
| -f <配置文件>         | 使用指定配置文件代替/etc/ld.so.conf |
| -C <缓存>           | 使用指定缓存代替/etc/ld.so.conf   |
| -r root           | 更新并使用 root 文件夹            |
| -c -format=FORMAT | 使用 FORMAT 作为缓冲区文件         |
| -? -help -usage   | 打印使用信息                    |
| -v -version       | 打印版本信息                    |

### 命令行进阶

◎ 范例 258 ◎ 建立对于共享库的正确链接，并更新缓冲区。

```
[root@wjc root]# ldconfig -v|more
/lib:
libcap.so.1 -> libcap.so.1.10
libnss_wins.so -> libnss_wins.so.2
libiw.so.25 -> libiw.so.25
libnss_winbind.so -> libnss_winbind.so.2
libnss_ldap.so.2 -> libnss_ldap-2.3.1.so
libpam_misc.so.0 -> libpam_misc.so.0.75
libpam.so.0 -> libpam.so.0.75
libproc.so.2.0.11 -> libproc.so.2.0.11
libpamc.so.0 -> libpamc.so.0.75
liblvm-10.so.1 -> liblvm-10.so.1.0
libtermcap.so.2 -> libtermcap.so.2.0.8
libpcre.so.0 -> libpcre.so.0.0.1
libgcc_s.so.1 -> libgcc_s-3.2.2-20030225.so.1
```

◎ 范例 259 ◎ 只更新在指令行中指定的库 /lib

```
[root@wjc root]# ldconfig -n /lib
```

### 相关命令

ldd、ld.so

## 2.32

### ldd 指令：打印共享库文件的相互依赖关系

### 命令行进阶

ldd[选择性参数]文件

## 2.33 lilo 指令：引导安装程序

**功能说明** ldd 指令打印指令行指定的程序或者库文件所使用的共享库。

**参数说明**

必要参数的具体说明如下：

| 参数                  | 功 能                        |
|---------------------|----------------------------|
| -v -verbose         | 打印所有的信息                    |
| -d -data-relocs     | 执行重定位并报告缺失对象的信息            |
| -f -function-relocs | 对于数据对象和函数都实行重定位，并报告缺失对象的信息 |

选择性参数的具体说明如下：

| 参数       | 功 能         |
|----------|-------------|
| -version | 打印 ldd 的版本号 |
| -help    | 打印使用信息      |

**执行范例**

◎ 范例 260 ◎ 显示/bin/mkdir 所使用的共享函数库。

```
[root@wjc root]# ldd /bin/mkdir
 libc.so.6 => /lib/tls/libc.so.6 (0x42000000)
 /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

**相关指令**

ldconfig

## 2.33

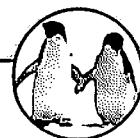
### lilo 指令：引导安装程序

**语法** lilo [必要性参数] [选择性参数]

**功能说明** lilo 指令安装一个启动装载程序，在下次启动时会运行。单独执行会根据/etc 目录下的 lilo.conf 配置文件安装 lilo。

**参数说明**

必要参数的具体说明如下：



| 参数 | 功能                          |
|----|-----------------------------|
| -c | 将系统启动时所需要的文件放在连续的磁区里，增加读取效率 |
| -l | 采用线性磁区地址                    |
| -q | 将系统核心文件的名称及存放的地址记录在映射文件中    |
| -V | 显示版本信息                      |

选择性参数的具体说明如下：

| 参数          | 功能                                             |
|-------------|------------------------------------------------|
| -b <外围设备代号> | 指定安装 lilo 处的外围设备代号                             |
| -C <配置文件>   | 指定 lilo 的配置文件，预设文件为 /etc/lilo.conf             |
| -d <延迟时间>   | 改变开机的启动时间，预设为 5s                               |
| -f <几何参数文件> | 指定开机几何参数文件，预设为 /etc 里的 disktab 文件              |
| -i <启动激活文件> | 指定开机所使用的磁区文件，预设文件为 /boot 目录里的 boot.b 文件        |
| -m <映射文件>   | 通过参数指定映射文件，预设文件为 /boot 的 map                   |
| -r <根目录>    | 当系统启动时指定想挂载成为根目录的目录                            |
| -s <备份文件>   | 指定备份文件，预设为 /boot 目录里的文件 bootxxxx，其中 xxxx 为数字代码 |

#### ● 执行范例

- ◎ 范例 261 ◎ 安装 lilo 到第一台 SCSI 硬盘的第 3 个主要分区，采用 3 级模式。

```
[root@wjc linuxzhiling]# lilo -b /dev/sda3 -v -v -v
```

- ◎ 范例 262 ◎ 指定安装 lilo 的配置文件和备份文件。

```
[root@wjc linuxzhiling]# lilo -C /etc/lilo.conf2 -s /boot/boot. Backup
```

#### ● 相关指令

liloconfig

2.34

**liloconfig 指令：设置程序的载入**

#### ● 相关

liloconfig

## 2.35 losetup 指令：设置循环

**功能说明** liloconfig 指令是 Shareware 发行版用来调整 lilo 设置的程序。用户通过互动的界面，轻松地操作 lilo 的安装和设置，不需要编写繁琐的配置文件。

- 范例 263 执行 liloconfig 指令。

```
[root@wjc linuxzhiling]# liloconfig
```

**相关指令**

lilo

## 2.35 losetup 指令：设置循环

**语法** losetup[必要参数][选择性参数][文件]

**功能说明** losetup 指令用来设置循环设备。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能 |
|----|----|
| -d | 卸载 |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| <<加密方式>> | 设置加密方式 |

**执行范例**

- 范例 264 显示未使用的设备。

```
[root@localhost chord]# losetup -f //显示未使用的设备
/dev/loop0
```

[root@localhost chord]#

本例使用指令“losetup -f”显示未使用的设备。

**相关指令**

rmmod

**2.36****mev 指令：监视鼠标情况**

**语法** mev[必要参数][选择性参数]

**功能说明** gpm 包的一部分，将鼠标控制口的事件打印到标准控制台上。

**参数说明**

必要参数的具体说明如下：

| 参数     | 功能                          |
|--------|-----------------------------|
| -C<数字> | 选择从数字标识的控制口上获取事件            |
| -d<数字> | 选择默认事件掩码，所有不属于事件掩码的事件都不会被打印 |
| -e<数字> | 选择事件掩码                      |
| -E     | Emacs 模式                    |
| -i     | 接收从标准终端输入的指令                |
| -f     | 适应屏幕内的拖曳行为                  |
| -m<数字> | 最小修正掩码，低于这个修正掩码的事件将不被上报     |
| -M<数字> | 最大修正掩码，高于这个修正掩码的事件将不被上报     |
| -p     | 在拖曳期间描绘出指针的轨迹               |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -u | 用户模式 |

**执行范例**

◎ 范例 265 ◎ 打印鼠标事件。

[yingzheng@yingzheng ~]\$ mev -C

mouse event 0x01 , at 40, 12 (delta 34,43), buttons 0, modifiers 0x56

**相关指令**

gpm、gpm-root

## 2.38 mkkickstart 指令：建立安装的组态文件

## 2.37

**minfo 指令：显示 MS-DOS 文件系统的各项参数**

minfo[选择性参数][必要参数]



显示 MS-DOS 文件系统的各项参数。



必要参数的具体说明如下：

| 参数    | 功能     |
|-------|--------|
| <驱动器> | 指定的驱动器 |

选择性参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -v | 打印根分区的 hexdump |



◎ 范例 266 ◎ 显示 DOS 系统参数。

[root@localhost littleProg]# minfo -v C: //显示系统参数

本例使用指令“minfo -v C:”显示 C 盘的系统参数。



mattrib

## 2.38

**mkkickstart 指令：建立安装的组态文件**

mkkickstart[选择性参数]



根据目前系统的设置来建立组态文件。这个文件可以供其他电脑在安装时使用。



必要参数的具体说明如下：



| 参数      | 功能                 |
|---------|--------------------|
| --nonet | 在没有网络环境的情况下不进行网络安装 |

选择性参数的具体说明如下：

| 参数        | 功能               |
|-----------|------------------|
| --version | 显示版本信息           |
| --nox     | 不进行 X Windows 配置 |
| --bootp   | 使用 bootp 安装或者开机  |
| --dhcp    | 使用 DHCP          |

#### 执行范例

- 范例 267 构建一个安装组态文件。

```
[root@yingzheng yingzheng]#mkkick --nonet --bootp
```

#### 相关指令

mkdir

## 2.39

### mkraid 指令：初始化/升级 RAID 设备阵列

#### 语法

mkraid[选择性参数][必要参数]

**功能说明** 可以将一组块设备设置为一个 RAID 设备阵列。注意初始化 RAID 设备阵列时将毁坏所有原数据。

#### 参数说明

必要参数的具体说明如下：

| 参数   | 功能                           |
|------|------------------------------|
| <设备> | 指定要设置为 RAID 设备阵列的设备，多个用“+”分开 |

选择性参数的具体说明如下：

## 2.40 modinfo 指令：显示内核信息

| 参数               | 功能                   |
|------------------|----------------------|
| --configfile<文件> | 以指定文件作为配置文件          |
| -force           | 强制初始化设备，不管设备原有的数据    |
| -help            | 显示帮助信息               |
| -upgrade         | 将旧的 RAID 阵列升级为当前内核版本 |
| --version        | 显示版本信息               |

**执行范例**

◎ 范例 268 ◎ 将几个设备组成 RAID 阵列。

```
[root@yingzheng yingzheng]# mkraid -f /dev/hda1+/dev/fd0
```

**相关指令**

raidtab、raidstart、raid0run、raidstop

**2.40****modinfo 指令：显示内核信息****语法** modinfo[选择性参数][必要参数]

**功能说明** modinfo 指令用来显示内核信息。

**参数说明**

必要参数的具体说明如下：

| 参数   | 功能        |
|------|-----------|
| <模块> | 指定输出信息的模块 |

选择性参数的具体说明如下：

| 参数          | 功能                      |
|-------------|-------------------------|
| -V          | 打印版本信息                  |
| -F<域>       | 仅打印指定域的信息               |
| -0          | 使用 ASCII 码的零字符作为域信息的分割符 |
| -a/-d/-l/-p | 分别指定作者、描述、许可证、参数域信息     |



### ◎ 案例范例

- ◎ 范例 269 ◎ 显示 sg 模块的信息。

```
[root@localhost root]# modinfo sg
author: Douglas Gilbert
description: SCSI generic (sg) driver
license: GPL
version: 3.5.31 9C7E7F0416AD01EF83DD507
parm: def_reserved_size:size of buffer reserved for each fd
parm: allow_dio:allow direct I/O (default: 0 (disallow))
alias: char-major-21-
vermagic: 2.6.9 686 REGPARM gcc-3.3
depends: scsi_mod
```

### ◎ 相关指令

modprobe、modinfo.old

## 2.41

### modprobe 指令：自动处理可载入模块

#### ◎ 案例范例

modprobe[选择性参数][必要参数]

**◎ 功能说明** modprobe 指令会根据模块内部的依赖关系自动安装所有依赖的模块，如果发生错误将卸载所有相关模块。这都是根据模块声称的依赖关系来决定的。modprobe 并不会对模块本身产生任何影响。

#### ◎ 参数说明

必要参数的具体说明如下：

| 参 数  | 功 能        |
|------|------------|
| <模块> | 指定要安装的模块文件 |

选择性参数的具体说明如下：

| 参 数    | 功 能                    |
|--------|------------------------|
| -v     | 打印程序执行信息。一般只有当出错时才打印信息 |
| -C<文件> | 使用指定的文件作为配置而不是使用默认配置   |
| -c     | 输出配置文件                 |

## 2.42 mouseconfig 指令：设置鼠标相关参数

续表

| 参 数                 | 功 能                  |
|---------------------|----------------------|
| -i                  | 忽略任何配置文件中的安装和移除指令    |
| -q                  | 不输出错误信息              |
| -r                  | 卸载指定的模块，其依赖的模块也将一起卸载 |
| -V                  | 输出版本信息               |
| -f                  | 强制安装                 |
| -l                  | 列出所有匹配的模块            |
| -a                  | 安装所有匹配的模块            |
| -o<名称>              | 试图重命名一个已安装的模块        |
| --show-dependencies | 列出模块的依赖关系            |

## 技巧范例

## ◎ 范例 270 ◎ 安装软驱模块。

```
[root@yingzheng yingzheng]# modprobe -v floppy
insmod /lib/modules/2.6.11-1.1369_FC4/kernel/drivers/block/floppy.ko
```

## ◎ 范例 271 ◎ 卸载软驱模块。

```
[root@yingzheng yingzheng]# modprobe -v -r floppy
rmmod /lib/modules/2.6.11-1.1369_FC4/kernel/drivers/block/floppy.ko
```

modprobe.conf、lsmod、modprobe.old

## 2.42

## mouseconfig 指令：设置鼠标相关参数

## 技巧 mouseconfig[选择性参数]

功能说明 mouseconfig 是 Red Hat Linux 下的鼠标配置工具。有两种模式，一种是交互式，一种是自动设置。

## 必要说明

必要参数的具体说明如下：



| 参数                 | 功能         |
|--------------------|------------|
| <code>-text</code> | 在文本界面模式中运行 |

选择性参数的具体说明如下：

| 参数                              | 功能                                                                                 |
|---------------------------------|------------------------------------------------------------------------------------|
| <code>--help</code>             | 显示帮助信息                                                                             |
| <code>--noui</code>             | 在命令行模式中运行                                                                          |
| <code>--emulthree</code>        | 模拟三键鼠标                                                                             |
| <code>--device&lt;设备&gt;</code> | 指定鼠标所在的端口，设备号一般为 <code>ttyS0~ttyS3</code> ，位于 PS/2 端口的 PS/2 鼠标为 <code>psaux</code> |
| <code>&lt;鼠标类型&gt;</code>       | 指定鼠标类型                                                                             |

### 执行范例

- 范例 272 以交互式的文本方式配置鼠标。

```
[root@yingzheng ~]# mouseconfig -text //下面打开配置界面
```



- 范例 273 以文本方式指定鼠标类型。

```
[root@yingzheng ~]# mouseconfig -noui morosoft
```

没有指定串口设备。假定为 `ttyS0`

关闭控制台鼠标服务。 [失败] [确定]

启动控制台鼠标服务: `inputattach`, can't set line discipline [确定]

### 相关指令

`system-config-mouse`

## 2.43

### nice 指令：设置优先权



nice[必要参数][选择性参数][指令]



nice 指令用来设置优先级，优先级的数值为-20~19，-20 的优先级最高，19 的优先级最低。



必要参数的具体说明如下：

| 参数 | 功能    |
|----|-------|
| -n | 指定优先级 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



◎ 范例 274 ◎ 设置程序运行时的优先级。

```
[root@localhost ~]# vi & //后台运行
[1] 4250
[root@localhost ~]# nice vi & //设置默认优先级
[2] 4252
[1]+ Stopped vi
[root@localhost ~]# nice -n 19 vi & //设置优先级为19
[3] 4257
[2]+ Stopped nice vi
[root@localhost ~]# nice -n -20 vi & //设置优先级为-20
[4] 4260
[3]+ Stopped nice -n 19 vi
[root@localhost ~]# ps -l //显示进程
F S UID PID PPID C PRI NI ADDR SZ WCHAN TTY TIME CMD
4 S 0 2740 2738 0 75 0 - 1148 wait pts/1 00:00:00 bash
```



## 第2章 系统设置

```

0 T 0 4250 2740 0 77 0 - 1115 finish pts/1 00:00:00 vi
0 T 0 4252 2740 0 86 10 - 1115 finish pts/1 00:00:00 vi
0 T 0 4257 2740 0 99 19 - 1116 finish pts/1 00:00:00 vi
4 T 0 4260 2740 0 60 -20 - 1116 finish pts/1 00:00:00 vi
4 R 0 4261 2740 0 77 0 - 1163 - pts/1 00:00:00 ps
[4]+ Stopped nice -n -20 vi
[root@localhost ~]#

```

本例使用指令“nice -n<数值> vi &”设置 vi 指令运行的优先级，默认时，设置的优先级为 10；然后运行“ps -l”指令，可以看到，在 NI 列显示了程序运行的优先级。上边运行的 4 个 vi 程序的优先级分别为 0、10、19、-20。

### 相关指令

ps

## 2.44 passwd 指令：设置密码

### 功能说明

passwd 指令用来设置用户账号的密码，超级用户可以设置所有用户的密码。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能          |
|----|-------------|
| -d | 删除密码        |
| -f | 强制执行        |
| -k | 更新只能发生在过期之后 |
| -l | 停止账号的使用     |
| -s | 显示密码信息      |
| -u | 启用已被停止的账号   |
| -x | 设置密码的有效期    |
| -g | 修改群组密码      |
| -i | 过期后停止用户账号   |

## 2.45 pwconv 指令：开启用户的投影密码

选择性参数的具体说明如下：

| 参数     | 功能   |
|--------|------|
| -help  | 帮助信息 |
| -usage | 主要用法 |

## ○○ 执行范例

## ◎ 范例 275 ◎ 修改用户密码。

```
[root@localhost ~]# passwd kk //设置用户 kk 的密码
Changing password for user kk.
New UNIX password:
BAD PASSWORD: it does not contain enough DIFFERENT characters
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@localhost ~]#
```

本例使用指令“passwd kk”设置账号 kk 的密码。

## ◎ 范例 276 ◎ 显示账号的密码信息。

```
[root@localhost ~]# passwd -S kk //显示账号的密码信息
Password set, MD5 crypt.
[root@localhost ~]#
```

本例使用指令“passwd -S kk”显示账号 kk 的密码信息。

## ◎ 范例 277 ◎ 删除用户密码。

```
[root@localhost ~]# passwd -d kk //删除用户密码
Removing password for user kk.
passwd: Success
[root@localhost ~]#
```

本例使用指令“passwd -d kk”删除用户 kk 的密码。

## ○○ 相关指令

useradd

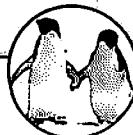
## 2.45 pwconv 指令：开启用户的投影密码

## ○○ 命令

pwconv

## ○○ 功能说明

pwconv 指令使得系统下用户的密码得到更安全的



保护。

#### 命令执行范例

- 范例 278 ○ 开启用户的投影密码。

```
[root@localhost ~]#pwconv //开启用户投影密码
```

本例使用指令 pwconv 开启用户的投影密码，使得密码得到更好的保护，不容易被别的用户得到。

#### 命令相关指令

`pwunconv`

## 2.46

### pwunconv 指令：关闭用户的投影密码

#### 命令语法

`pwunconv`

#### 功能说明

pwunconv 指令用来关闭用户的投影密码。

#### 命令执行范例

- 范例 279 ○ 关闭用户的投影密码。

```
[root@localhost ~]#pwunconv //关闭用户投影密码
```

本例使用指令 pwunconv 关闭用户的投影密码。

#### 命令相关指令

`pwconv`

## 2.47

### resize 指令：设置终端机视窗的大小

#### 命令语法

`resize[必要参数][选择性参数]`

#### 功能说明

resize 指令用来设置虚拟终端的屏幕显示大小。

#### 参数说明

必要参数的具体说明如下：

| 参数 | 功能                   |
|----|----------------------|
| -c | 用 C shell 来实现指令      |
| -u | 用 Bourne shell 来实现指令 |

选择性参数的具体说明如下：

| 参数         | 功能     |
|------------|--------|
| -s<列数><行数> | 指定屏幕大小 |

### ○ 行行范围

○ 范例 280 ○ 使用 C shell。

```
[root@localhost ~]# resize -c //使用 C shell
set noglob;
setenv COLUMNS '80';
setenv LINES '21';
unset noglob;
[root@localhost ~]#
```

本例使用指令“resize -c”，使用 C shell 来改便虚拟终端的大小。

○ 范例 281 ○ 使用 Bourne shell。

```
[root@localhost ~]# resize -u //使用 Bourne shell
COLUMNS=80;
LINES=21;
export COLUMNS LINES;
[root@localhost ~]#
```

本例使用指令“resize -u”，使用 Bourne shell 来使用改变虚拟终端的屏幕大小。

○ 范例 282 ○ 设置屏幕大小。

```
[root@localhost ~]# resize -s 60 120 //设置屏幕大小
COLUMNS=120;
LINES=45;
export COLUMNS LINES;
[root@localhost ~]#
```

本例使用指令“resize -s 60 120”设置虚拟终端的大小为高 60、长 120。

### ○ 相关指令

csh、tset、xterm

## 2.48 rpm 指令：管理 RPM 包

### ○ 命令

rpm[必要参数][选择性参数]



## 第2章 系统设置

**◆ 功能说明** rpm 指令是一个功能强大的包管理器，它可以用来建立、安装、请求、确认、更新和卸载软件包。

**◆ 参数说明**

必要参数的具体说明如下：

| 参数        | 功 能          |
|-----------|--------------|
| -c        | 显示所有的配置文件    |
| -d        | 列出所有的文档文件    |
| -dump     | 输出基本的文件信息    |
| -l        | 列出包里的所有文件    |
| -s        | 显示文件的状态      |
| -a        | 请求或者校验所有的数据包 |
| -f        | 请求或者校验软件包的文件 |
| -g        | 请求或者校验群组的软件包 |
| -p        | 请求或者校验软件包的文件 |
| -specfile | 请求指定文件       |
| -h        | 显示安装进度       |
| -v        | 运行时显示详细的处理过程 |

选择性参数的具体说明如下：

| 参数                 | 功 能      |
|--------------------|----------|
| -queryformat<请求格式> | 设置请求格式   |
| -e<RPM 包>          | 删除 RPM 包 |
| -f<RPM 包>          | 查询 RPM 包 |
| -i<RPM 包>          | 安装 RPM 包 |
| -p<RPM 包>          | 查询 RPM 包 |
| -help              | 帮助信息     |
| -version           | 版本信息     |

**◆ 执行范例**

(◎ 范例 283 ◎) 安装软件。

## 2.49 runlevel 指令：显示执行等级

```
-sh-2.05b# rpm -hvi RealPlayer10GOLD.rpm //安装 RPM 包
Preparing... [100%]
 1:RealPlayer [100%]
-sh-2.05b#
```

本例使用指令“`rpm -hvi RealPlayer10GOLD.rpm`”安装 RPM 包 `RealPlayer10GOLD.rpm`, 安装完成之后, 系统就可以使用 RealPlayer 软件了。

## ◎ 范例 284 ◎ 显示软件的安装信息。

```
-sh-2.05b# rpm -qi RealPlayer //显示软件的安装信息
Name : RealPlayer Relocations: /usr/local
Version : 10.0.8.805 Vendor: RealNetworks, Inc
Release : 20060718 Build Date: 2006年07月19日 09时17分36秒
Install Date: 2007年05月28日 09时38分03秒 Build Host:marlin
Group: Applications/Multimedia Source RPM:RealPlayer-10.0.8.805-
20060718.src.rpm
Size : 15265081 License: RPSL, EULA
Signature : (none)
Packager : RealNetworks, Inc
URL : http://www.real.com/
Summary : RealPlayer
Description :
RealPlayer is a media player with the aim of providing solid media
playback locally and via streaming. It plays RealAudio, RealVideo, MP3,
3GPP Video, Flash, SMIL 2.0, JPEG, GIF, PNG, RealPix and RealText and
more. The RealPlayer 10 for Linux builds on top of the popular
open-source Helix Player.
```

-sh-2.05b#

## ◎ 范例 285 ◎ 卸载软件。

```
-sh-2.05b# rpm -ev RealPlayer //卸载软件 RealPlayer
-sh-2.05b#
```

本例使用指令“`rpm -ev RealPlayer`”卸载软件 RealPlayer。

## ◎ 相关命令

`popt`、`rpm2cpio`、`rpmbuild`

## 2.49

## runlevel 指令：显示执行等级



runlevel



 **功能说明** runlevel 指令读取系统的 utmp 文件来定位系统的运行级别记录，然后打印当前系统的执行等级到标准输出。

 **执行范例**

○ 范例 286 ○ 显示执行等级。

```
[root@localhost ~]# runlevel //显示执行等级
N 5
```

```
[root@localhost ~]#
```

本例使用指令 runlevel 显示当前系统的执行等级。

 **相关命令**

init、utmp

## 2.50 set 指令：设置 shell

 **语法**

set[必要参数][选择性参数][指令]

 **功能说明** set 指令可以设置 shell 的执行方式，同时也可使用指令 set 设置环境变量。

 **参数说明**

必要参数的具体说明如下：

| 参数 | 功 能           |
|----|---------------|
| -a | 显示修改的变量       |
| -b | 显示程序执行情况      |
| -d | 取消杂姿记忆        |
| -e | 出现错误时退出 shell |
| -f | 取消通配符设置       |
| -h | 记录函数信息        |
| -k | 环境变量设置        |
| -l | for 循环设置      |
| -m | monitor 模式    |
| -n | 测试模式          |

## 2.50 set 指令：设置 shell

续表

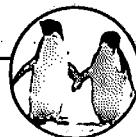
| 参数 | 功能                 |
|----|--------------------|
| -P | 优先模式               |
| -P | 用符号链接指向的实际文件代替符号链接 |
| -t | 指令执行完毕就退出 shell    |
| -u | 不执行未定义的变量          |
| -v | 运行时显示详细的处理信息       |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -H Shell | 执行历史指令 |
| +<参数>    | 取消参数设置 |


**执行范例**
**◎ 范例 287 ◎ 显示当前的环境变量的设置。**

```
[root@localhost chord]# set //显示环境变量的设置
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="00" [2]="16" [3]="1" [4]="release"
[5]="i386-redhat-linux-gnu")
BASH_VERSION='3.00.16(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=126
DIRSTACK=()
EUID=0 //此处省略了部分结果
SSH_CONNECTION='::ffff:192.168.88.7 2229::ffff:192.168.88.76 22'
SSH_TTY=/dev/pts/1
SUPPORTED=zh_CN.UTF-8:zh_CN:zh
TERM=xterm
UID=0
```



```
USER=root
_=clear
[root@localhost chord]#
```

本例使用指令 set 显示当前系统下环境变量的设置。

◎ 范例 288 ◎ 设置环境变量。

```
[root@localhost chord]# set SHELL "/bin/sh" //设置环境变量
```

本例使用指令 “set SHELL "/bin/sh"” 设置环境变量 SHELL 的值为 /bin/sh。

◎ 范例 289 ◎ 取消通配符设置。

```
-sh-2.05b# ll //显示文件列表
总用量 24
-rw-r--r-- 1 root root 17 5月 29 09:45 123
-rw-r--r-- 1 root root 17 5月 29 09:45 124
-rw-r--r-- 1 root root 17 5月 29 09:45 125
-rw-r--r-- 1 root root 17 5月 29 09:45 126
-rw-r--r-- 1 root root 17 5月 29 09:46 127
-rw-r--r-- 1 root root 17 5月 29 09:46 128
-sh-2.05b# set -f //取消通配符设置
-sh-2.05b# rm 12*
rm: lstat:@12*失败: 没有那个文件或者目录
-sh-2.05b# set +f //设置通配符
-sh-2.05b# rm 12* //删除文件
-sh-2.05b# ll
总用量 0
-sh-2.05b#
```

本例使用指令 “set -f” 取消通配符的设置；本例使用指令 “set +f” 恢复通配符的设置。



unset

## 2.51

## setconsole 指令：设置系统终端



setconsole[必要参数]



setconsole 指令用来设置系统的终端。

## 2.52 setenv 指令：查询或显示环境变量

## 功能说明

必要参数的具体说明如下：

| 参 数    | 功 能        |
|--------|------------|
| serial | PROM 终端    |
| ttyS0  | 终端设置为第一个串口 |
| ttyS1  | 终端设置为第二个串口 |
| video  | 终端设置为显卡    |

## 执行范例

## ◎ 范例 290 ◎ 设置终端。

```
[root@localhost ~]#setconsole ttyS0 //设置第1个串口为终端
```

本例使用指令“setconsole ttyS0”设置第1个串口为终端。

## 相关指令

set

## 2.52

## setenv 指令：查询或显示环境变量

## 语法 setenv[环境变量][环境变量值]

功能说明 指令 setenv 可以显示当前系统的环境变量的值，也可以用来设置环境变量的值。

## 执行范例

## ◎ 范例 291 ◎ 显示环境变量。

```
[root@localhost ~]#setenv //显示环境变量
```

本例使用指令 setenv 显示当前系统设置的环境变量。

## ◎ 范例 292 ◎ 设置环境变量。

```
[root@localhost ~]#setenv USER kk //设置环境变量
```

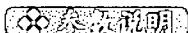
本例使用指令“setenv USER kk”设置环境变量。

## 相关指令

set

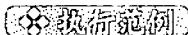
**2.53****setserial 指令：设置或显示串口的相关信息****命令** setserial[必要参数][设备][其他参数]

**功能说明** setserial 指令可以显示当前系统的串口的设置，同时也可以设置当前串口的参数。



必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -a | 显示详细的信息      |
| -v | 运行时显示详细的处理信息 |
| -g | 显示串口信息       |



○ 范例 293 ○ 设置串口。

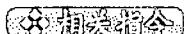
```
[root@localhost temp]# setserial -v /dev/ttyS2 //设置串口
/dev/ttyS2, UART: unknown, Port: 0x03e8, IRQ: 4
[root@localhost temp]#
```

本例使用指令“setserial -v /dev/ttyS2”设置串口 2。

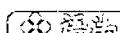
○ 范例 294 ○ 显示串口信息。

```
[root@localhost temp]# setserial -g /dev/ttyS2 //显示串口信息
/dev/ttyS2, UART: unknown, Port: 0x03e8, IRQ: 4
[root@localhost temp]#
```

本例使用指令“setserial -g /dev/ttyS2”显示串口 2 的信息。



set

**2.54****setup 指令：设置公用程序****命令** setup

**功能说明** setup 指令用来设置认证、防火墙、鼠标、网络配置、打印机配置、系统服务、时区配置等。

**执行范例**

- 范例 295 ○ 使用 setup 指令进行设置。

```
-sh-2.05b# setup //设置
```

运行指令 setup 后，将出现以下的界面。



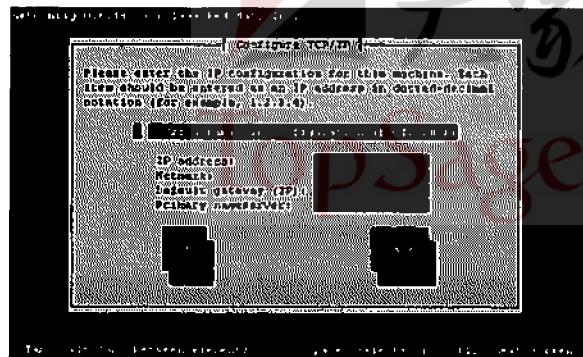
然后选择相应的参数进行配置。如，选择 Network configuration，会出现以下界面。



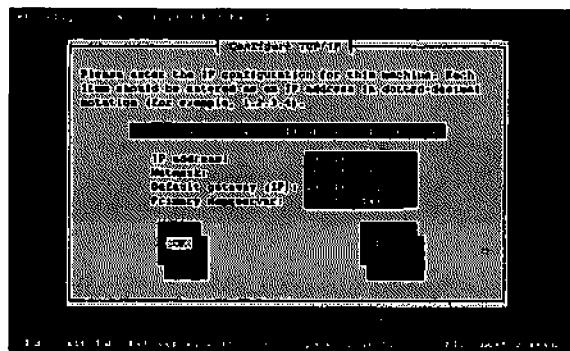
单击“Yes”按钮，进入下一步，出现网络配置界面。



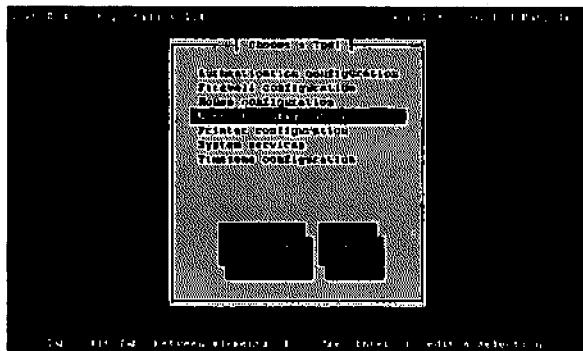
## 第2章 系统设置



配置相应的选项。



单击“OK”按钮，按下“Enter”键，完成配置，然后返回初始的配置界面。



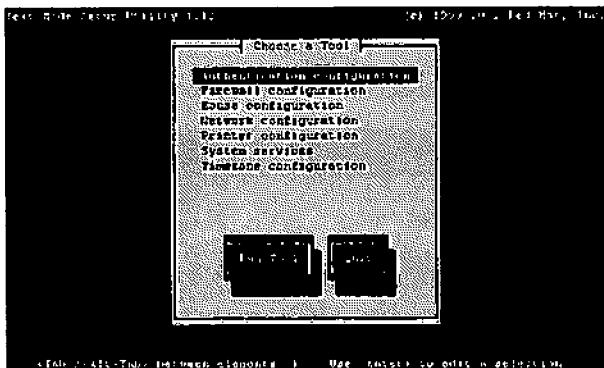
## 2.54 setup 指令：设置公用程序

这时，网络配置已经完成，若不再进行其他的配置，则单击“Quit”按钮，退出 setup 设置。

### ◎范例 296 ◎ 用 setup 指令进行系统服务设置。

```
-sh-2.05b# setup //设置
```

运行指令 setup 后，将出现以下的界面。



选择“System services”选项，按“Enter”键，进入系统服务配置界面。



选择要启用的服务，前面的[]内有“\*”字符的代表该服务已启动，光标移到相应的服务，反复按空格键可以完成服务的选中或者取消选中。按“F1”可以显示服务的具体信息。



选择完成之后，一直按“确定”按钮，然后退出，即可完成配置。

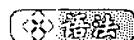
同样地可以进行其他的配置，比如认证配置、防火墙配置、鼠标配置、打印机配置、时区配置等。



set

## 2.55

**sliplogin 指令：将终端机之间的连接变为 sliplogin 连接**



**sliplogin[用户名]**



sliplogin 指令将终端机之间的连接变为 sliplogin 连接。



◎ 范例 297 ◎ 改变用户的连接方式。

```
[root@localhost ~]# sliplogin kk //改变用户的连接方式
```

本例使用指令“sliplogin kk”将用户 kk 的连接方式改变为 SLIP 方式。

## 2.56

**swapoff 指令：关闭系统交换分区**



**swapoff[必要参数][选择性参数][设备]**

## 2.56 swapoff 指令：关闭系统交换分区

**功能说明** swapoff 指令可以关闭交换分区，swapon 指令可以打开交换分区。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能       |
|----|----------|
| -a | 关闭所有交换设备 |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -h | 帮助信息 |
| -v | 版本信息 |

**执行范例**

○ 范例 298 ○ 显示分区信息。

```
[root@localhost ~]# sfdisk -l //显示分区信息
Disk /dev/sda: 1305 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0
 Device Boot Start End #cyls #blocks Id System
/dev/sdal * 0+ 12 13- 104391 83 Linux
/dev/sda2 13 77 65 522112+ 82 Linux swap / Solaris
/dev/sda3 78 1304 1227 9855877+ 83 Linux
/dev/sda4 0 - 0 0 0 Empty
[root@localhost ~]#
```

本例使用指令“sfdisk -l”显示系统的分区信息，可以看到/dev/sda2为交换分区。

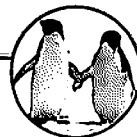
○ 范例 299 ○ 关闭交换分区。

```
[root@localhost ~]# swapoff /dev/sda2 //关闭交换分区
[root@localhost ~]#
```

本例使用指令“swapoff /dev/sda2”关闭交换分区/dev/sda2。

**相关指令**

swapon



## 2.57 swapon 指令：启动系统交换分区



**swapon**[必要参数][选择性参数][设备]

**功能说明** swapon 指令用来启动交换分区，swapoff 指令用来关闭交换分区。



必要参数的具体说明如下：

| 参数 | 功 能        |
|----|------------|
| -a | 开启所有交换分区   |
| -s | 显示 swap 信息 |

选择性参数的具体说明如下：

| 参数 | 功 能  |
|----|------|
| -h | 帮助信息 |
| -V | 版本信息 |



**范例 300** 显示分区信息。

```
[root@localhost ~]# sfdisk -l //显示分区信息
Disk /dev/sda: 1305 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0
 Device Boot Start End #cyls #blocks Id System
 /dev/sda1 * 0+ 12 13- 104391 83 Linux
 /dev/sda2 13 77 65 522112+ 82 Linux swap / Solaris
 /dev/sda3 78 1304 1227 9855877+ 83 Linux
 /dev/sda4 0 - 0 0 0 Empty
[root@localhost ~]#
```

本例使用指令“`sdisk -l`”显示系统的分区信息，可以看到`/dev/sda2`为交换分区。

## 2.58 sysctl 指令：设置系统核心参数

## ◎ 范例 301 ◎ 开启交换分区。

```
[root@localhost ~]# swapon /dev/sda2 //开启交换分区
[root@localhost ~]#
```

本例使用指令“swapon /dev/sda2”开启交换分区/dev/sda2。

## ◎ 范例 302 ◎ 显示交换分区信息。

```
[root@localhost ~]# swapon -s /dev/sda2 //显示交换分区信息
Filename Type Size Used Priority
/dev/sda2 partition 522104 0 -2
[root@localhost ~]#
```

本例使用指令“swapon -s /dev/sda2”显示交换分区/dev/sda2 的信息。



**swapoff**

## 2.58

### sysctl 指令：设置系统核心参数

**语法** sysctl[必要参数][选择性参数][变量]

**功能说明** sysctl 指令用来设置系统的核心参数。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功 能       |
|----|-----------|
| -a | 显示当前所有的设置 |
| -A | 以表格的形式显示  |
| -e | 模糊模式      |
| -n | 忽略关键词     |

选择性参数的具体说明如下：

| 参数               | 功 能    |
|------------------|--------|
| -w<variable=key> | 设置变量值  |
| -p<文件>           | 指定配置文件 |
| -h               | 帮助信息   |
| -V               | 版本信息   |



### ◎ 范例 303

显示系统核心设置。

```
[root@localhost ~]# sysctl -a >1.log //显示系统设置
```

```
[root@localhost ~]# tail -n 50 1.log
```

//显示文件信息

```
kernel.msgmnb = 16384
```

```
kernel.msgmni = 16
```

```
kernel.msgmax = 8192
```

```
kernel.shmmni = 4096
```

```
kernel.shmall = 2097152
```

```
kernel.shmmax = 33554432
```

```
kernel.acct = 4 2 30
```

```
kernel.hotplug = /sbin/udevsend
```

```
kernel.modprobe = /sbin/modprobe
```

```
kernel.printk = 6 4 1 7
```

```
kernel.ctrl-alt-del = 0
```

```
kernel.real-root-dev = 0
```

```
kernel.cap-bound = -257
```

```
kernel.tainted = 0
```

```
kernel.core_pattern = core
```

```
kernel.core_uses_pid = 1
```

```
kernel.vdso = 1
```

```
kernel.print-fatal-signals = 0
```

```
kernel.exec-shield = 1
```

```
kernel.panic = 0
```

```
kernel.domainname = (none)
```

```
kernel.hostname = localhost.localdomain
```

```
kernel.version = #1 Thu Jun 2 22:55:56 EDT 2005
```

```
kernel.osrelease = 2.6.11-1.1369_FC4
```

```
kernel.ostype = Linux
```

```
fs.mqueue.msgsize_max = 8192
```

```
fs.mqueue.msg_max = 10
```

```
fs.mqueue.queues_max = 256
```

```
fs.quota.warnings = 1
```

```
fs.quota.syncs = 19
```

```
fs.quota.free_dquots = 0
```

```
fs.quota.allocated_dquots = 0
```

```
fs.quota.cache_hits = 0
```

## 2.59 telinit 指令：设置系统的执行级别

```

fs.quota.writes = 0
fs.quota.reads = 0
fs.quota.drops = 0
fs.quota.lookups = 0
fs.aio-max-nr = 65536
fs.aio-nr = 0
fs.lease-break-time = 45
fs.dir-notify-enable = 1
fs.leases-enable = 1
fs.overflowgid = 65534
fs.overflowuid = 65534
fs.dentry-state = 27450 25570 45 0 0 0
fs.file-max = 50823
fs.file-nr = 870 0 50823
fs.inode-state = 18904 178 0 0 0 0
fs.inode-nr = 18904 178
fs.binfmt_misc.status = enabled[root@localhost ~]#

```

本例使用指令“`sysctl -a >1.log`”显示系统设置，并将结果重定向到文件 `1.log`。

### ○ 范例 304 ○ 设置变量。

```

[root@localhost ~]# sysctl -w fs.leases-enable=1 //设置变量
fs.leases-enable = 1
[root@localhost ~]#

```

本例使用指令“`sysctl -w fs.leases-enable=1`”设置变量 `fs.leases-enable` 的值为 1。



`set`

## 2.59

## telinit 指令：设置系统的执行级别



`telinit[必要参数][选择性参数]`



`telinit` 指令可以设置当前系统的执行级别。



必要参数的具体说明如下：



## 第2章 系统设置

| 参数  | 功能           |
|-----|--------------|
| 0~6 | 执行等级         |
| q   | 重新执行 telinit |
| s   | 单人模式         |
| u   | 保持当前状态       |

选择性参数的具体说明如下：

| 参数    | 功能    |
|-------|-------|
| -<时间> | 等候的时间 |

### ◆ 技巧范例 ◆

#### ● 范例 305 ● 设置当前的执行级别。

```
[root@localhost ~]# runlevel //显示当前的执行级别
N 5
[root@localhost ~]# telinit 2 //设置执行级别
[root@localhost ~]# runlevel //显示执行等级
5 2
[root@localhost ~]#
```

本例使用指令“telinit 2”设置当前的执行级别为 2。

### ◆ 相关指令 ◆

runlevel

## 2.60

## timeconfig 指令：设置时区

### ◆ 技巧 ◆

timeconfig[必要参数][时区]

### ◆ 功能说明 ◆

timeconfig 指令用来设置系统时间所属的时区。

### ◆ 参数说明 ◆

必要参数的具体说明如下：

| 参数     | 功能       |
|--------|----------|
| --arc  | Alpha 时间 |
| --test | 测试       |
| --utc  | utc 时间   |

## 2.61 ulimit 指令：控制 shell 程序的资源

## ❖ 执行范例

## ◎ 范例 306 ◎ 设置时区。

```
[root@localhost ~]#timeconfig //设置时区
```

本例使用指令 timeconfig 进入可视化界面，设置时区。

## ❖ 相关指令

**date**

## 2.61 ulimit 指令：控制 shell 程序的资源

## ❖ 活学 ulimit[必要参数][选择性参数]

❖ 功能说明 ulimit 指令用来控制系统资源。

## ❖ 参数说明

必要参数的具体说明如下：

| 参数 | 功能        |
|----|-----------|
| -a | 显示目前资源的设置 |
| -H | 资源限制      |
| -S | 弹性限制      |

选择性参数的具体说明如下：

| 参数         | 功能        |
|------------|-----------|
| -c<文件上限>   | 设定文件上限    |
| -d<数据节区大小> | 程序数据节区的大小 |
| -f<文件大小>   | 文件大小上限    |
| -m<内存大小>   | 内存上限      |
| -n<文件数目>   | 文件数目上限    |
| -p<缓冲区大小>  | 缓冲区大小上限   |
| -t<时间>     | CPU 时间上限  |
| -u<程序数目>   | 程序数目上限    |
| -v<虚拟内存大小> | 虚拟内存上限    |



### 范例范例

#### ○ 范例 307 ○ 显示系统资源的设置。

```
[root@localhost ~]# ulimit -a //显示系统资源的设置
core file size (blocks, -c) 0
data seg size (kbytes, -d) unlimited
file size (blocks, -f) unlimited
pending signals (-i) 8192
max locked memory (kbytes, -l) 32
max memory size (kbytes, -m) unlimited
open files (-n) 1024
pipe size (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
stack size (kbytes, -s) 10240
cpu time (seconds, -t) unlimited
max user processes (-u) 8192
virtual memory (kbytes, -v) unlimited
file locks (-x) unlimited
[root@localhost ~]#
```

本例使用指令“ulimit -a”显示当前系统的设置。

#### ○ 范例 308 ○ 设置单一用户程序数目上限。

```
[root@localhost ~]# ulimit -u 700 //设置单一用户程序上限
[root@localhost ~]# ulimit -a //显示系统资源配置
core file size (blocks, -c) 0
data seg size (kbytes, -d) unlimited
file size (blocks, -f) unlimited
pending signals (-i) 8192
max locked memory (kbytes, -l) 32
max memory size (kbytes, -m) unlimited
open files (-n) 1024
pipe size (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
stack size (kbytes, -s) 10240
cpu time (seconds, -t) unlimited
max user processes (-u) 700
virtual memory (kbytes, -v) unlimited
file locks (-x) unlimited
[root@localhost ~]#
```

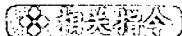
## 2.62 unalias 指令：删除别名

本例使用指令“ulimit -u 700”设置单一用户程序上限为 700；使用指令“ulimit -list”显示系统资源配置，可以看到“max user processes (-u) 700”，支持的单一用户的程序上限为 700。

## ◎ 范例 309 ◎ 设置同一时间可打开的文件数目。

```
[root@localhost ~]# ulimit -n 1000 //设置同一时间可打开的文件数目
[root@localhost ~]# ulimit -a //显示系统资源配置
core file size (blocks, -c) 0
data seg size (kbytes, -d) unlimited
file size (blocks, -f) unlimited
pending signals (-i) 8192
max locked memory (kbytes, -l) 32
max memory size (kbytes, -m) unlimited
open files (-n) 1000
pipe size (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
stack size (kbytes, -s) 10240
cpu time (seconds, -t) unlimited
max user processes (-u) 700
virtual memory (kbytes, -v) unlimited
file locks (-x) unlimited
[root@localhost ~]#
```

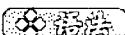
本例使用指令“ulimit -n 1000”设置同一时间可打开的文件数目的上限为 1000；使用 ulimit 指令显示系统资源配置，可以看到“open files (-n) 1000”。



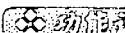
sysctl

## 2.62

## unalias 指令：删除别名



unalias[比要参数][别名]



unalias 指令用来删除别名设置。



必要参数的具体说明如下：



| 参数 | 功能     |
|----|--------|
| -a | 删除所有别名 |

### ❖ 执行范例

#### ◎ 范例 310 ◎ 删除别名。

```
[root@localhost ~]# alias DP=tcpdump //设置别名
[root@localhost ~]# alias DP //显示别名信息
alias DP='tcpdump'
[root@localhost ~]# unalias DP //删除别名
[root@localhost ~]# alias DP //显示别名
-bash: alias: DP: not found
[root@localhost ~]#
```

本例使用指令“unalias DP”删除别名 DP 的设置。

### ❖ 相关指令

alias

## 2.63

## unset 指令：删除变量或函数

### ❖ 命令

unset[必要参数][名称]

❖ 功能说明

unset 指令用来删除指定的变量或者函数。

### ❖ 例题说明

必要参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -f | 删除函数 |
| -v | 删除变量 |

### ❖ 执行范例

#### ◎ 范例 311 ◎ 删除环境变量。

```
[root@localhost ~]# set //显示环境变量
BASH=/bin/bash
BASH_ARGC=()
```

## 2.63 unset 指令：删除变量或函数

```

BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="00" [2]="16" [3]="1" [4]="release"
[5]="i386-redhat-linux-gnu")
BASH_VERSION='3.00.16(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=125
DIRSTACK=()
EUID=0
GROUPS=()
G_BROKEN_FILERNAME=1
HISTFILE=/root/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/root
HOSTNAME=localhost.localdomain
HOSTTYPE=i386
IFS=$' \t\n'
INPUTRC=/etc/inputrc
KDEDIR=/usr
LANG=zh_CN.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=22
LOGNAME=root
 //此处省略部分结果
TERM=xterm
UID=0
USER=root
=test
[root@localhost ~]# unset FWD //删除环境变量
[root@localhost ~]# set //显示环境变量
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="00" [2]="16" [3]="1" [4]="release"
[5]="i386-redhat-linux-gnu")

```



```
BASH_VERSION='3.00.16(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=125
DIRSTACK=()
EUID=0
GROUPS=()
//此处省略了部分结果

SUPPORTED=zh_CN.UTF-8:zh_CN:zh
TERM=xterm
UID=0
USER=root
_=PWD
[root@localhost ~]#
```

本例使用指令“unset PWD”删除环境变量 PWD。



set

## 2.64

## up2date 指令：更新 Linux 系统



up2date[必要参数][选择性参数]



up2date 指令可以保持系统的最新状态。



必要参数的具体说明如下：

| 参 数         | 功 能         |
|-------------|-------------|
| -configure  | 配置更新代理参数    |
| -register   | 注册一个系统      |
| -d          | 只下载更新包，而不安装 |
| -nodownload | 不下载更新包      |
| -f          | 强制执行包的安装    |
| -i          | 下载更新包后安装    |
| -l          | 列出已经更新的更新包  |

## 2.64 up2date 指令：更新 Linux 系统

续表

| 参数       | 功能                |
|----------|-------------------|
| -dry-run | 输出可以更新的数据包        |
| -showall | 显示更新包             |
| -k       | 下载之前复制要查找的更新包的路径  |
| --nosig  | 不使用 GPG 来检查更新包信号  |
| --src    | 下载源 RPM 和二进制的 RPM |
| -u       | 完全更新系统            |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### 执行范例

#### ◎ 范例 312 ◎ 更新系统。

```
[root@localhost ~]# up2date //更新系统
0. debug No
1. useRhn No
2. rhnuuid 3df31014-915f-11db-8781-af09d047da8a
3. isatty Yes
4. showAvailablePacks No
5. useNoSSLForPackage No
6. storageDir /var/spool/up2date
7. pkgSkipList ['kernel*']
8. retrieveOnly No
9. noSSLServerURL http://xmlrpc.rhn.redhat.com/XMLRPC
10. networkRetries 5
11. pkgsToInstallNotUp ['kernel', 'kernel-modules', 'kernel-devel']
12. enableProxy No
13. noBootLoader No
14. proxyPassword
15. updateUp2date Yes
16. keepAfterInstall No
```

17. useGPG Yes  
 18. headerCacheSize 40  
 19. forceInstall No  
 20. systemIdPath /etc/sysconfig/rhn/systemid  
 21. retrieveSource No  
 22. enableRollbacks No  
 23. gpgKeyRing /etc/sysconfig/rhn/up2date-keyring.gpg  
 24. adminAddress ['root@localhost']  
 25. serverURL https://xmlrpc.rhn.redhat.com/XMLRPC  
 26. fileSkipList []  
 27. versionOverride  
 28. sslCACert /usr/share/rhn/RHNS-CA-CERT  
 29. noReplaceConfig Yes  
 30. enableProxyAuth No  
 31. disallowConfChange ['noReboot', 'sslCACert', 'useNoSSLForPackages',  
 'noSSLse'  
 32. headerFetchCount 10  
 33. proxyUser  
 34. removeSkipList ['kernel\*']  
 35. httpProxy  
 36. noReboot No

Enter number of item to edit <return to exit, q to quit without saving>:

本例使用指令 up2date 更新 Linux 系统。

● 范例 313 ● 列出所有可更新的更新包。

[root@localhost ~]# up2date -l

本例使用指令“up2date -l”列出所有可供更新的数据包。



rpm

## 2.65

### vmstat 指令：显示虚拟内存信息



vmstat[必要参数][选择性参数]



vmstat 指令用来显示虚拟内存的信息。

## 2.65 vmstat 指令：显示虚拟内存信息

**◆ 参数说明**

必要参数的具体说明如下：

| 参数 | 功能          |
|----|-------------|
| -a | 显示所有虚拟内存的信息 |
| -d | 磁盘信息        |
| -f | ford 信息     |

选择性参数的具体说明如下：

| 参数     | 功能   |
|--------|------|
| -S<单位> | 指定单位 |
| -V     | 版本信息 |

**◆ 执行范例****◎ 范例 314 ◎ 显示虚拟内存使用信息。**

```
[root@localhost ~]# vmstat //显示虚拟内存的使用信息
procs -----memory----- swap-----io-----system-----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa
 0 0 0 252240 18724 162312 0 0 32 8 1032 35 1 1 97 1
[root@localhost ~]#
```

本例使用指令 vmstat 显示虚拟内存的使用信息。

**◎ 范例 315 ◎ 显示磁盘信息。**

```
[root@localhost ~]# vmstat -d //显示磁盘信息
disk-----reads-----writes-----IO-----
 total merged sectors ms total merged sectors ms cur sec
ram0 0 0 0 0 0 0 0 0 0 0 0
ram1 0 0 0 0 0 0 0 0 0 0 0
ram2 0 0 0 0 0 0 0 0 0 0 0
ram3 0 0 0 0 0 0 0 0 0 0 0
ram4 0 0 0 0 0 0 0 0 0 0 0
ram5 0 0 0 0 0 0 0 0 0 0 0
ram6 0 0 0 0 0 0 0 0 0 0 0
ram7 0 0 0 0 0 0 0 0 0 0 0
ram8 0 0 0 0 0 0 0 0 0 0 0
```



## 第2章 系统设置

```

ram9 0 0 0 0 0 0 0 0 0 0 0
ram10 0 0 0 0 0 0 0 0 0 0 0
ram11 0 0 0 0 0 0 0 0 0 0 0
ram12 0 0 0 0 0 0 0 0 0 0 0
ram13 0 0 0 0 0 0 0 0 0 0 0
ram14 0 0 0 0 0 0 0 0 0 0 0
ram15 0 0 0 0 0 0 0 0 0 0 0
hdc 0 0 0 0 0 0 0 0 0 0 0
sda 8133 1666 42254 78489 3999 6622 84878 50072 0 53
fd0 0 0 0 0 0 0 0 0 0 0 0
md0 0 0 0 0 0 0 0 0 0 0 0
[root@localhost ~]#

```

本例使用指令“vmstat -d”显示磁盘信息。

- 范例 316 ○ 显示内存的完整信息。

```
[root@localhost ~]# vmstat -s //显示内存的完整信息
```

```

515292 total memory
263052 used memory
101084 active memory
120264 inactive memory
252240 free memory
18756 buffer memory
162312 swap cache
522104 total swap
 0 used swap
522104 free swap
2669 non-nice user cpu ticks
 359 nice user cpu ticks
 7025 system cpu ticks
534443 idle cpu ticks
 4197 IO-wait cpu ticks
 886 IRQ cpu ticks
 0 softirq cpu ticks
171127 pages paged in
 42603 pages paged out
 0 pages swapped in
 0 pages swapped out
5669476 interrupts
193227 CPU context switches

```

## 2.66 yes 指令：持续输出给定的字符串，每行显示一个字符串

```
1180573686 boot time
3954 forks
[root@localhost ~]#
```

本例使用指令“vmstat -s”显示内存的完整信息。



ps、top

## 2.66 yes 指令：持续输出给定的字符串，每行显示一个字符串

**范例** yes[选择性参数][字符串]

**功能说明** yes 指令用来回应字符串。

**参数说明**

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

**执行范例**

○ 范例 317 ○ 持续输出字符串。

```
[root@localhost ~]# yes good good study day day up //持续输出字符串
good good study day day up
```

本例使用指令“yes good good study day day up”持续输出字符串“good good study day day up”。

○ 范例 318 ○ 回答使用交互式指令时出现的问题。

```
[root@localhost temp]# ls //显示文件列表
libacl.so libcap.so.1.10 libext2fs.so.2
```



## 第2章 系统设置

```

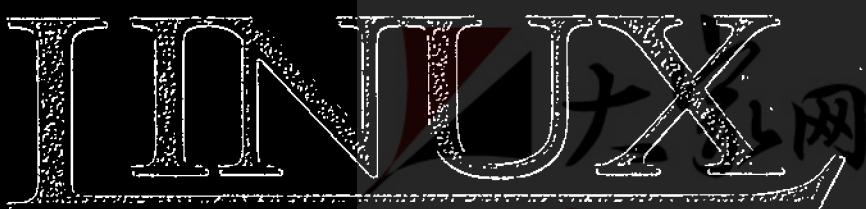
libacl.so.1 libcom_err.so.2 libext2fs.so.2.4
libacl.so.1.0.3 libcom_err.so.2.0 libgcc_s-3.2.2-20030225.so.1
libanl-2.3.2.so libcrypt-2.3.2.so libgcc_s.so.1
libanl.so.1 libcrypto.so.0.9.7a libiw.so.25
libattr.so libcrypto.so.4 liblvm-10.so
libattr.so.1 libcrypt.so.1 liblvm-10.so.1
libattr.so.1.0.1 libc.so.6 liblvm-10.so.1.0
libBrokenLocale-2.3.2.so libdb-4.0.so libssl.so.0.9.7a
libBrokenLocale.so.1 libdl-2.3.2.solibssl.so.4
libc-2.3.2.so libdl.so.2 libss.so.2
libcap.so libe2p.so.2 libss.so.2.0
libcap.so.1 libe2p.so.2.3
[root@localhost temp]# yes |rm * //回答交互式问题
rm: 是否删除一般文件'libacl.so'? rm: 是否删除一般文件'libacl.so.1'? rm: 是否删除一般文件'libacl.so.1.0.3'? rm: 是否删除一般文件'libanl-2.3.2.so'? rm: 是否删除一般文件'libanl.so.1'? rm: 是否删除一般文件'libattr.so'? rm: 是否删除一般文件'libattr.so.1'? rm: 是否删除一般文件'libattr.so.1.0.1'? rm: 是否删除一般文件'libBrokenLocale-2.3.2.so'? rm: 是否删除一般文件'libBrokenLocale.so.1'? rm: 是否删除一般文件'libc-2.3.2.so'? rm: 是否删除一般文件'libcap.so'? rm: 是否删除一般文件'libcap.so.1'? rm: 是否删除一般文件'libcap.so.1.10'? rm: 是否删除一般文件'libcom_err.so.2'? rm: 是否删除一般文件'libcom_err.so.2.0'? rm: 是否删除一般文件'libcrypt-2.3.2.so'? rm: 是否删除一般文件'libcrypto.so.0.9.7a'? rm: 是否删除一般文件'libcrypto.so.4'? rm: 是否删除一般文件'libcrypt.so.1'? rm: 是否删除一般文件'libc.so.6'? rm: 是否删除一般文件'libdb-4.0.so'? rm: 是否删除一般文件'libdl-2.3.2.so'? rm: 是否删除一般文件'libdl.so.2'? rm: 是否删除一般文件'libe2p.so.2'? rm: 是否删除一般文件'libe2p.so.2.3'? rm: 是否删除一般文件'libext2fs.so.2'? rm: 是否删除一般文件'libext2fs.so.2.4'? rm: 是否删除一般文件'libgcc_s-3.2.2-20030225.so.1'? rm: 是否删除一般文件'libgcc_s.so.1'? rm: 是否删除一般文件'libiw.so.25'? rm: 是否删除一般文件'liblvm-10.so'? rm: 是否删除一般文件'liblvm-10.so.1'? rm: 是否删除一般文件'liblvm-10.so.1.0'? rm: 是否删除一般文件'libssl.so.0.9.7a'? rm: 是否删除一般文件'libssl.so.4'? rm: 是否删除一般文件'libss.so.2'? rm: 是否删除一般文件'libss.so.2.0'? [root@localhost temp]# ls

```

本例使用指令“yes |rm \*”，在删除文件过程中，需要用户选择“yes”或“no”时，均选择“yes”。



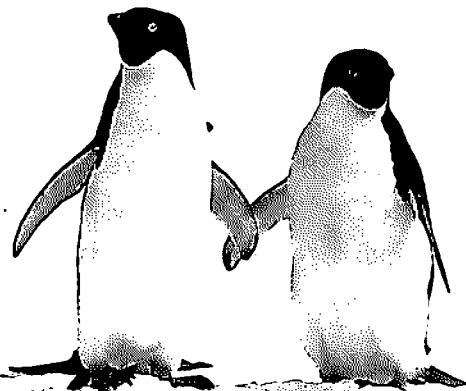
grep



## 第3章

### 文本编辑

文本编辑指令的主要功能是编辑 Linux 系统下的文本文件，主要包括文本文件的显示、修改、删除、添加，也包括文件内容的重新编排、命令行下的文本的编辑和图形界面下的文本的编辑等。





## 3.1 awk 指令：模式匹配语言

**【必读说明】** awk[必要参数][选择性参数]文件

**【必读说明】** awk 指令是一种模式匹配语言，它针对正则表达式对文件进行相应的操作，比如将文件中符合某种规则的信息输出。

**【必读说明】**

必要参数的具体说明如下：

| 参    数 | 功    能   |
|--------|----------|
| -F     | 指定输入分离器  |
| -f     | 指定程序的源文件 |

选择性参数的具体说明如下：

| 参    数    | 功    能 |
|-----------|--------|
| --help    | 帮助信息   |
| --version | 版本信息   |

**【执行案例】**

○ 范例 319 ○ 输出指定文件指定格式的内容。

```
[root@localhost root]# cat abc
i am a student
i am from bj
[root@localhost root]# awk '{print $2,$4}' abc //显示文件 abc 的每一行的第二
二个和第四个单词
am student
am bj
[root@localhost root]#
```

当前目录下有一个文件文件 abc，文件第一行的内容为“i am a student”，第二行的指令为“i am from bj”，此内容已经在本例使用指令 cat 在屏幕上输出，然后，运行指令“awk '{print \$2,\$4}' abc”将文件 abc 的每一行的第二个和第四个单词输出，得到结果第一行的输出为“am student”，第二行

的输出为“am bj”，如上所示。

○范例 320 ○ 输出指定文件中小于指定长度的文本行。

```
[root@localhost root]# awk 'length($0<60)' abc //显示文件abc中小于60个字符的文本行
```

```
i am a student
i am from bj
[root@localhost root]#
```

运行“awk 'length(\$0<60)' abc”指令，显示文件abc中文本行小于60个字符的文本行，得到的结果第一行为“i am a student”，第二行为“i am from bj”。

○范例 321 ○ 输出指定文件的某种格式的文本信息。

```
[root@localhost root]# awk '{print $1}' /etc/passwd //输出/etc/passwd文件中每一行第一个字符的信息
```

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP
nobody:x:99:99:Nobody:/sbin/nologin
rpm:x:37:37:/var/lib/rpm:/bin/bash
vcsa:x:69:69:virtual
nscd:x:28:28:NSCD
sshd:x:74:74:Privilege-separated
rpc:x:32:32:Portmapper
rpcuser:x:29:29:RPC
nfsnobody:x:65534:65534:Anonymous
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
pcap:x:77:77::/var/arpwatch:/sbin/nologin
```



```
apache:x:48:48:Apache:/var/www:/sbin/nologin
squid:x:23:23::/var/spool/squid:/sbin/nologin
webalizer:x:67:67:Webalizer:/var/www/html/usage:/sbin/nologin
xfs:x:43:43:X
named:x:25:25:Named:/var/named:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
gdm:x:42:42::/var/gdm:/sbin/nologin
postgres:x:26:26:PostgreSQL
desktop:x:80:80:desktop:/var/lib/menu/kde:/sbin/nologin
kk:x:500:500:kk:/home/kk:/bin/bash
[root@localhost root]#
```

**相关指令**

gawk

## 3.2 col 指令：过滤控制字符

**语法** col[必要参数][选择性参数]

**功能说明** col 指令过滤器解析输出为正确的格式，可能的情况下它将用 tab 代替空格符，这对 nroff 和 tbl 的输出可能是有用的。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能                         |
|-----|-----------------------------|
| -b  | 过滤掉包括 RLF 和 HRLF 在内的所有的控制字符 |
| -f  | 过滤掉 RLF 字符                  |
| -x  | 使用 tab 代替多个空格               |
| -p  | 不改变未知序列                     |

选择性参数的具体说明如下：

| 参 数    | 功 能               |
|--------|-------------------|
| -l<列数> | 指定缓冲区列数，默认为 128 列 |

**执行范例**

○ 范例 322 ○ 过滤控制字符。

## 3.3 colrm 指令：删除指定的列

[root@localhost root]# man ls|col -b >ls.man.txt //将 ls 的帮助信息重定向，并且过滤掉 RLF 和 HRLF 字符

本例使用指令“man ls|col -b >ls.man.txt”将 ls 的 man 信息重定向到文件 ls.man.txt，并且输入到 ls.man.txt 文件的信息已经过滤掉 RLF 和 HRLF 等控制字符。

## ◎ 相关指令

expand、nroff、tbl

## 3.3 colrm 指令：删除指定的列

## ◎ 基础 colrm[开始列][结束列]

◎ 功能说明 colrm 指令从指定文件中过滤掉选择的列。若只指定一列，那么将删除指定列之后的所有列；若指定两个列数值，将删除两个值之间的所有的列。输入为标准输入，输出被发送到标准输出。

## ◎ 实际案例

## ◎ 范例 323 ◎ 删除指定的列。

```
[root@localhost root]# colrm 2 4 //删除指定的第 2 列和第 4 列
kdjflksjdlkf
klksjdlkf
1234567890
1567890 //输出
```

本例使用指令“colrm 2 4”删除指定的第 2 列和第 4 列，当从标准输入输入“kdjflksjdlkf”时，输出的结果为“klksjdlkf”，当从标准输入输入的结果为“1234567890”时，输出的结果为“1567890”。

## ◎ 范例 324 ◎ 删除文件中指定的列。

```
[root@localhost root]# cat abc //显示文件 abc 的内容
i am a student
i am from bj
[root@localhost root]# colrm 3 <abc //删除文件的每一行第 3 列之后的值
i
i
[root@localhost root]# colrm 3 5 <abc //删除文件第 3 列~第 5 列之间的值
i a student
```



```
i from bj
[root@localhost root]#
```

首先使用指令“cat abc”显示文件的内容，第一行为“i am a student”，第二行为“i am from bj”，本例使用指令“colrm 3<abc”删除文件的每一行第3列之后的值，输出的结果第一行为“i”，第二行为“i”。本例使用指令“colrm 3 5<abc”删除文件第3列～第5列之间的值，输出的结果第一行为“i a student”，第二行为“i from bj”。

#### 相关命令

awk、column、expend、paste

## 3.4 comm 指令：比较排序文件

#### 基础

comm[必要参数][选择性参数][文件1][文件2]

功能说明 逐行比较排序的文件。如果不指定参数，将输出三部分的内容。第一部分是文件1单独含有的行，第二部分是文件2单独含有的行，第三部分是两个文件共有的行。

#### 基本说明

必要参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -1 | 文件1独有的行不显示     |
| -2 | 文件2独有的行不显示     |
| -3 | 文件1和文件2共有的行不显示 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### 实际案例

- ◎ 范例 325 ◎ 比较排序文件。

## 3.4 comm 指令：比较排序文件

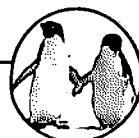
```
[root@localhost linux]# cat file1 //显示文件 file1 的内容
i am a student
i am from bj
i love my life
happy
everyday
the common part
[root@localhost linux]# cat file2 //显示文件 file2 的内容
11111111111111111111111111
2222222222222
333333333333
444444444444
555555555555
6666666666
the common part
[root@localhost linux]# comm file1 file2 //比较排序文件 file1 和 file2
11111111111111111111111111
2222222222222
333333333333
444444444444
555555555555
6666666666
i am a student
i am from bj
i love my life
happy
everyday
the common part
[root@localhost linux]#
```

本例使用指令“comm file1 file2”比较排序文件 file1 和 file2，当不指定参数时，将输出三部分的内容。第一部分是文件 1 单独含有的行，第二部分是文件 2 单独含有的行，第三部分是两个文件共有的行，如上所示。

◎ 范例 326 ◎ 比较排序的文件，文件 1 单独含有的行不显示。

[root@localhost linux]# comm -1 file1 file2 //比较排序文件，文件 file1 单独含有的行不显示

```
11111111111111111111111111
```



2222222222222

333333333333

444444444444

555555555555

6666666666

the common part

[root@localhost linux]#

◎ 范例 327 ◎ 比较排序文件，文件 2 独有的行不显示。

[root@localhost linux]# comm -2 file1 file2 //比较排序文件，文件 file2

独有的行不显示

i am a student

i am from bj

i love my life

happy

everyday

the common part

[root@localhost linux]#

◎ 范例 328 ◎ 比较排序文件，文件 1 和文件 2 共有的部分不显示。

[root@localhost linux]# comm -3 file1 file2 //比较排序文件，文件 file1

和文件 file2 共有的部分不显示

11111111111111111111111111

22222222222222

333333333333

444444444444

555555555555

6666666666

i am a student

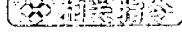
i am from bj

i love my life

happy

everyday

[root@localhost linux]#



diff

**3.5****ed 指令：文本编辑器**

ed[必要参数][选择性参数][文件]

**功能说明** ed 指令用来编辑文本，它是一个比较简单的文本编辑器。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能    |
|----|-------|
| -G | 支持兼容  |
| -s | 不进行检查 |

选择性参数的具体说明如下：

| 参数       | 功能    |
|----------|-------|
| -help    | 帮助信息  |
| -version | 版本信息  |
| -p<字符串>  | 指定提示符 |

**执行范围**

◎ 范例 329 ◎ 文本编辑。

```
[root@localhost ~]# ed //进入ed编辑
a
aaaaaaaaaaaaaa
bbbbbbbbbbbbbb
c
.
w file
32
q
[root@localhost ~]# cat file //查看文件信息
aaaaaaaaaaaaaa
bbbbbbbbbbbbbb
c
[root@localhost ~]#
```

本例使用指令 ed 进入文本编辑。

**相关指令**

vi



## 3.6 egrep 指令：输出匹配某种模式的行



**egrep[必要参数][选择性参数][文件]**

**功能说明** 使用方法和效果同“grep-e”，使用方法见 grep 的使用方法，此处不再详细描述。

## 3.7 ext2ed 指令：ext2 文件系统编辑



**ext2ed**



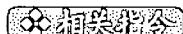
**ext2ed** 指令用来处理硬盘分区上的数据。



◎ 范例 330 ◎ ext2 文件系统编辑。

```
[root@localhost root]#ext2ed //文件系统编辑
```

本例使用指令 ext2ed 编辑文件系统。



e2fsck

## 3.8 fgrep 指令：匹配字符串



**fgrep[必要参数][选择性参数]样式[文件或者目录]**



在文件中查找符合条件的字符串，相当于“grep -f”。



必要参数的具体说明如下：

| 参数 | 功能              |
|----|-----------------|
| -a | 包含所有类型的数据，包括二进制 |
| -b | 显示位编号           |
| -c | 计算符号条件的行数       |

续表

| 参数 | 功能                 |
|----|--------------------|
| -i | 对大小写不敏感            |
| -l | 列出符合条件的文件名         |
| -q | 程序运行时不显示处理信息       |
| -s | 显示错误信息             |
| -v | 方向选择，匹配不包含指定字符串的内容 |
| -y | 等同于“-i”            |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -help    | 帮助信息   |
| -version | 版本信息   |
| -f<文件>   | 指定样式文件 |
| -e<样式>   | 指定样式   |

### ◎范例 331 ◎ 匹配字符串

```
[root@localhost ~]# fgrep bc * //匹配字符串“bc”
install.log:安装 glibc-common-2.3.5-10.i386.
install.log:安装 glibc-2.3.5-10.i686.
install.log:安装 bc-1.06-18.i386.
install.log:安装 jakarta-commons-dbc-1.2.1-3jpp_1fc.noarch.
install.log:安装 bcel-5.1-1jpp_4fc.noarch.
install.log:安装 glibc-kernheaders-2.4-9.1.94.i386.
install.log:安装 glibc-headers-2.3.5-10.i386.
install.log:安装 glibc-devel-2.3.5-10.i386.
install.log:安装 php-odbc-5.0.4-10.i386.
install.log:安装 libcap-1.10-22.i386.
install.log:安装 libcroco-0.6.0-5.i386.
install.log:安装 evolution-webcal-2.2.0-1.i386.
install.log:安装 libcap-devel-1.10-22.i386.
install.log:安装 libcroco-devel-0.6.0-5.i386.
```



```

install.log:安装 libc-client-2002e-9.i386.
install.log:安装 ant-apache-bcel-1.6.2-3jpp_8fc.i386.
install.log:安装 jakarta-commons-dbc- javadoc-1.2.1-3jpp_1fc.noarch.
install.log:安装 bcel-javadoc-5.1-1jpp_4fc.noarch.
install.log:安装 postgresql-jdbc-8.0.3-1.i386.
install.log:安装 postgresql-cdbc-08.00.0100-1.i386.
install.log:安装 aqhbc-1.0.2beta-2.i386.
install.log:安装 glibc-profile-2.3.5-10.i386.
install.log:安装 glibc-utils-2.3.5-10.i386.
install.log:安装 aqhbc-devel-1.0.2beta-2.i386.
install.log:安装 libc-client-devel-2002e-9.i386.
install.log:安装 php-bcmath-5.0.4-10.i386.
install.log:glibc-2.3.5-10.i386.rpm
kk.diff:--- abc 2007-05-15 10:05:28.000000000 +0800
[root@localhost ~]#

```

本例使用指令“`fgrep bc *`”查找当前文件夹下含有字符串“bc”的文件，并将得到的文件信息输出到标准输出。

◎ 范例 332 ◎ 列出符合条件的文件名。

```

[root@localhost ~]# fgrep -l bc * //列出含有字符串“bc”的文件
install.log
kk.diff
[root@localhost ~]#

```

本例使用指令“`fgrep -l bc *`”列出当前文件夹下符合条件即含有字符串“bc”的所有文件。

◎ 范例 333 ◎ 字符匹配的反向选择。

```

[root@localhost ~]# cat abc //显示文件abc的内容
i am a student
i am from bj
i like sports
espacially basketball and football
my favorite football club is intermilan
i will love it for ever
oh,abc
[root@localhost ~]# fgrep -v bc abc. //显示文件abc中不含“abc”的行
i am a student
i am from bj

```

```
i like sports
especially basketball and football
my favorite football club is intermilan
i will love it for ever
[root@localhost ~]# fgrep bc abc//显示文件abc中含有“bc”的行
oh,abc
[root@localhost ~]#
```

本例首先使用指令“fgrep -v bc abc”进行字符串的反向匹配，输出文件abc中不含字符串“abc”的行，所以文件abc的最后一行“oh,abc”没有显示，然后使用指令“fgrep bc abc”显示文件abc中的含有字符串“abc”的行。

### ○ 基本语法

grep、egrep

## 3.9 fmt 指令：编排文本文件

**○ 基本语法** fmt[必要参数][选择性参数][文件]

**○ 功能说明** 指令 fmt 重排指定文件的每一个段落，并输出到标准输出。如果不指定文件将从标准输入进行输入。

### ○ 重要说明

必要参数的具体说明如下：

| 参 数 | 功 能                 |
|-----|---------------------|
| -c  | 每段的前两排缩排            |
| -s  | 拆开超过限制的列            |
| -t  | 缩排，但是第一列和第二列的缩排格式不同 |
| -u  | 句子间空两个空格，字符间空一个空格   |

选择性参数的具体说明如下：

| 参 数    | 功 能            |
|--------|----------------|
| -p<字符> | 仅合并起始字符为指定字符的列 |
| -w<字符> | 指定每列的字符数       |



续表

| 参数        | 功能   |
|-----------|------|
| -help     | 帮助信息 |
| --version | 版本信息 |

### ◎ 执行范例

◎ 范例 334 ◎ 重排文件。

```
[root@localhost ~]# cat abc
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
abcdefhijk
abcdefhijkl
[root@localhost ~]# fmt abc. //重排文件 abc
a ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefhij
abcdefghijkl abcdefghijkl
[root@localhost ~]#
```

本例使用指令“fmt abc”重排文件 abc。

### ◎ 相关指令

fold

## 3.10 gedit 指令：gnome 的文本编辑器

◎ 基本用法 gedit[必要参数][选择性参数][文件]

◎ 功能说明 gedit 指令用来编辑文本文件。

### ◎ 参数说明

必要参数的具体说明如下：

### 3.11 head 指令：输出文件开头的部分信息

| 参数            | 功能       |
|---------------|----------|
| -new-document | 创建一个新的文档 |
| -quit         | 退出 gedit |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### ◎ 操作范例

##### ◎ 范例 335 ◎ 编辑文件。

```
[root@localhost littleProg]# gedit abc //编辑文件
```

本例使用指令“gedit abc”编辑文件。

#### ◎ 相关命令

vi

## 3.11 head 指令：输出文件开头的部分信息

 **语句** head[必要参数][选择性参数][文件名]

 **功能说明** head 指令将打印文件的前 10 行到标准输出。如果不指定文件，将从标准输入中输入。

#### ◎ 语句说明

必要参数的具体说明如下：

| 参数 | 功能    |
|----|-------|
| -q | 隐藏文件名 |
| -v | 显示文件名 |

选择性参数的具体说明如下：



| 参数        | 功能     |
|-----------|--------|
| -c<字节>    | 显示的字节数 |
| -n<行数>    | 显示的行数  |
| -help     | 帮助信息   |
| --version | 版本信息   |

### ○范例 336 ○ 显示文件的前 10 行。

```
[root@localhost sha1]# head sha.cpp //显示文件的前10行
/*
 * sha.cpp
 *
 * Copyright (C) 1998
 * Paul E. Jones <paulej@arid.us>
 * All Rights Reserved
 *

 * $Id: sha.cpp,v 1.5 2004/03/27 18:02:15 paulej Exp $

[root@localhost sha1]#
```

本例使用指令“head sha.cpp”显示文件 sha.cpp 的前 10 行。

### ○范例 337 ○ 显示文件的前 n 行。

```
[root@localhost sha1]# head -n 20 sha.cpp //显示文件的前20行
/*
 * sha.cpp
 *
 * Copyright (C) 1998
 * Paul E. Jones <paulej@arid.us>
 * All Rights Reserved
 *

 * $Id: sha.cpp,v 1.5 2004/03/27 18:02:15 paulej Exp $

*
* Description:
```

## 3.12 ispell 指令：拼字检查程序

第3章

```

* This utility will display the message digest (fingerprint) for
* the specified file(s).
*
* Portability Issues:
* None.
*/

```

```
#include <stdio.h>
[root@localhost shal]#

```

本例使用指令“head -n 20 sha.cpp”显示文件的前 20 行。

◎ 范例 338 ◎ 显示文件的前 n 个字节。

```
[root@localhost shal]# head -c 300 sha.cpp//显示文件的前 300 个字节
/*
```

```

* sha.cpp
*
* Copyright (C) 1998
* Paul E. Jones <paulej@arid.us>
* All Rights Reserved
*

* $Id: sha.cpp,v 1.5 2004/03/27 18:02:15 paulej Exp $

```

```
[root@localhost shal]#

```

本例使用指令“head -c 300 sha.cpp”显示文件 sha.cpp 的前 300 个字节。



tail

## 3.12 ispell 指令：拼字检查程序



**ispell[必要参数][选择性参数][文件]**



ispell 指令将使用预设的字典检查指定的文件。



必要参数的具体说明如下：



| 参数 | 功能               |
|----|------------------|
| -a | 管道时使用            |
| -A | 检查文件内容中指定的文件     |
| -b | 产生.bak 备份文件      |
| -B | 检查连字错误           |
| -C | 不检查连字错误          |
| -l | 从标准输入读入字符        |
| -m | 检查词尾变化           |
| -M | 菜单               |
| -n | nroff 或 troff 格式 |
| -N | 不显示菜单            |
| -P | 不考虑词尾变化          |
| -t | 文本文件             |
| -x | 不备份              |

选择性参数的具体说明如下：

| 参数     | 功能     |
|--------|--------|
| -d<文件> | 指定字典文件 |
| -p<文件> | 指定字典文件 |

### 执行案例

#### ◎ 范例 339 ◎ 检查文件。

```
root@localhost:~# ispell iptables.sh //检查文件
IP File: iptables.sh
51
echo "Enable IP Forwarding..."
00: DIP 16: LIP
01: HIP 17: NIP
02: HP 18: NP
03: I 19: P
04: ID 20: PI
05: IF 21: PIP
```

## 3.12 ispell 指令：拼字检查程序

第3章

```

06: II 22: RIP
07: IMP 23: SIP
08: IN 24: TIP
09: IO 25: UP
10: IR 26: YIP
11: IS 27: ZIP
12: IT
13: IV
14: IX
15: KIP
[SP] <number> R)epl A)ccept I)nsert L)ookup U)ncap Q)uit e(X)it or ?
for help

```

本例使用指令“ispell iptables.sh”检查文件 iptables.sh。

○范例 340 ○ 检查文件，如果有修改则产生备份文件。

```
root@localhost:~# ispell -b iptables.sh //检查文件，如有修改则产生备份文件
```

```
IP File: iptables.sh
```

```
#!
```

```
echo "Enable IP Forwarding..."
```

```
00: DIP 16: LIP
```

```
01: HIP 17: NIP
```

```
02: HP 18: NP
```

```
03: I 19: P
```

```
04: ID 20: PI
```

```
05: IF 21: PIP
```

```
06: II 22: RIP
```

```
07: IMP 23: SIP
```

```
08: IN 24: TIP
```

```
09: IO 25: UP
```

```
10: IR 26: YIP
```

```
11: IS 27: ZIP
```

```
12: IT
```

```
13: IV
```

```
14: IX
```

```
15: KIP
```

```
[SP] <number> R)epl A)ccept I)nsert L)ookup U)ncap Q)uit e(X)it or ?
```

for help

本例使用指令“ispell -b iptables.sh”检查文件，如果有修改则产生后



缀为.bak 的备份文件。

◎ 范例 341 ◎ 指定词典进行检查。

```
root@localhost:~# ispell -d american iptables.sh //指定词典进行检查
```

```
IP File: iptables.sh
#! echo "Enable IP Forwarding..."
00: DIP 16: LIP
01: HIP 17: NIP
02: HP 18: NP
03: I 19: P
04: ID 20: PI
05: IF 21: PIP
06: II 22: RIP
07: IMP 23: SIP
08: IN 24: TIP
09: IO 25: UP
10: IR 26: YIP
11: IS 27: ZIP
12: IT
13: IV
14: IX
15: KIP
[SP] <number> R)epl A)ccept I)nsert L)ookup U)ncap Q)uit e(X)it or ?
for help
```

本例使用指令“spell -d american iptables.sh”指定词典为美语词典进行检查。

相关命令

spell、egrep、look、join、sort、sq、tib

### 3.13

### jed 指令：编辑文本文件

概述 jed[必要参数][选择性参数][文件]

功能说明 jed 指令用来编辑文本文件。

大致说明

必要参数的具体说明如下：

## 3.14 joe 指令：编辑文本文件

第  
3  
章

| 参数     | 功能      |
|--------|---------|
| -2     | 显示两个编辑区 |
| -batch | 批处理     |
| -n     | 忽略配置文件  |

选择性参数的具体说明如下：

| 参数     | 功能          |
|--------|-------------|
| -f<函数> | 执行 Slang 函数 |
| -g<行数> | 处理指定行数      |

## ◆ 执行范例

◎ 范例 342 ◎ 编辑文件。

[root@localhost ~]jed -2 gettimeofday.c //编辑文件

本例使用指令“jed -2 gettimeofday.c”以两个编辑区编辑文件 gettimeofday.c。

## ◆ 相关指令

vi

## 3.14 joe 指令：编辑文本文件

◆ 语法 ◆ joe[必要参数][选择性参数][文件]

◆ 功能说明 ◆ joe 指令是一个功能强大的 ASCII 文本编辑器。它操作比较复杂，但是功能非常强大。

【模拟环境】

- jstar：模拟 WordStar 环境。
- jmacs：模拟 Emacs 环境。
- jpico：模拟 Pico 环境。
- rjoe：简约版 joe。

## ◆ 重要说明

必要参数的具体说明如下：



### 第3章 文本编辑

| 参数          | 功 能           |
|-------------|---------------|
| -asis       | 只处理小于 127 的字符 |
| -beep       | 出错则发声         |
| -csmode     | 连续检查          |
| -force      | 最后一行加换行符      |
| -help       | 帮助菜单          |
| -keepup     | 状态菜单          |
| -lightoff   | 不反白显示         |
| -marking    | 反白显示          |
| -mid        | 卷页时光标移到中央     |
| -nobackups  | 不备份           |
| -nonotice   | 不显示版本信息       |
| -nosta      | 不显示状态列        |
| -autoindent | 自动缩排          |
| -crlf       | 兼容性设置         |
| -linum      | 显示行号          |
| -overwrite  | 覆盖模式          |
| -ronly      | 只读方式          |

选择性参数的具体说明如下：

| 参数            | 功 能    |
|---------------|--------|
| -backpath<目录> | 备份目录   |
| -columns<栏位>  | 设置栏数   |
| -lines<行数>    | 设置     |
| +<行数>         | 移到指定行数 |

#### ◆ 执行范例 ◆

◎ 范例 343 ◎ 编辑文件。

```
[root@localhost ~]# joe abc //编辑文件 abc
Processing '/etc/joe/joerc'...done
```

## 3.14 joe 指令：编辑文本文件

```
Processing '/etc/joe/joerc'...done
IW abc
Row 1 Col 1 9:59 Ctrl-K H for help
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
abcdefhijk
abcdefhijkl
```

\*\* Joe's Own Editor v3.1 \*\* (utf-8) \*\* Copyright © 2004 \*\*

本例使用指令“joe abc”编辑文件 abc。

◎ 范例 344 ◎ 打开文件时直接移到指定行数。

```
[root@localhost ~]# joe +5 abc //移到指定行数
Processing '/etc/joe/joerc'...done
Processing '/etc/joe/joerc'...done
IW abc
Row 5 Col 1 10:06 Ctrl-K H
for help
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
abcdefhijk
abcdefhijkl
```

\*\* Joe's Own Editor v3.1 \*\* (utf-8) \*\* Copyright © 2004 \*\*



本例使用指令“joe +5 abc”打开文件abc，并将光标移动到第5行，然后进行编辑。

◎范例345 ◎ 显示帮助菜单。

```
[root@localhost ~]# joe abc
Processing '/etc/joe/joerc'...done
Processing '/etc/joe/joerc'...done

 Help Screen turn off with ^KH
more help with ESC . (^[.])
 CURSOR GO TO BLOCK DELETE MISC EXIT
 ^B left ^F right ^U prev. screen ^KB begin ^D char.
^KJ reformat ^KX save
 ^P up ^N down ^V next screen ^KK end ^Y line
^T options ^C abort
 ^Z previous word ^A beg. of line ^KM move ^W >word
^R refresh ^KZ shell
 ^X next word ^E end of line ^KC copy ^O word<
^I insert FILE
 SEARCH ^KU top of file ^KW file ^J >line
SPELL ^KE edit
 ^KF find text ^KV end of file ^KY delete ^_ undo ^[N word
^KR insert
 ^L find next ^KL to line No. ^K/ filter ^^ redo ^[L file
^KD save
 IW abc
for help
 a
 ab
 abc
 abcd
 abcde
 abcdef
 abcdefg
 abcdefgh
 abcdefghi
 abcdefghij
 abcdefghijk
 abcdefghijkl
```

本例使用指令“joe abc”打开文件abc，然后执行组合键[Ctrl+K+H]，

## 3.14 joe 指令：编辑文本文件

在屏幕的上方显示帮助信息，方便进一步操作。

◎ 范例 346 ◎ 以只读的方式打开文件。

```
[root@localhost ~]# joe -rdonly abc //以只读的方式打开文件
Processing '/etc/joe/joerc'...done
Processing '/etc/joe/joerc'...done
IW abc (Read only) Row 1 Col 1 10:13 Ctrl-K
```

H for help

```
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefghij
abcdefghijk
abcdefghijkl
```

\*\* Joe's Own Editor v3.1 \*\* (utf-8) \*\* Copyright © 2004 \*\*

本例使用指令“joe -rdonly abc”以只读的方式打开文件 abc。

**【进入 joe 后的相关操作】**

- PgUp: 上一页。
- PgDn: 下一页。
- Ctrl+a: 该行开始。
- Ctrl+e: 该行结尾。
- Ctrl+k+u: 文件起始处。
- Ctrl+k+v: 文件结尾处。
- Ctrl+k+h: 帮助菜单。
- Ctrl+k+f: 查找字符串。
- Ctrl+k+c: 复制。
- Ctrl+k+y: 删除。
- Ctrl+k+d: 保存。
- Ctrl+c: 关闭文件。


◆ 相关指令 ◆

vi、emacs、vim

**3.15****join 指令：将两个文件中与指定栏位  
内容相同的行连接起来**
◆ 适当 ◆

join[必要参数][选择性参数][文件 1][文件 2]

◆ 功能说明 ◆

join 指令用于连接两个指定的文件。

◆ 参数说明 ◆

必要参数的具体说明如下：

| 参数 | 功能         |
|----|------------|
| -i | 不区别大小写     |
| -e | 用空白代替失去的栏位 |

选择性参数的具体说明如下：

| 参数       | 功能                |
|----------|-------------------|
| -help    | 帮助信息              |
| -version | 版本信息              |
| -a 1     | 显示文件 1 中不同的行      |
| -a 2     | 显示文件 2 中不同的行      |
| -j<栏位>   | 等同于“-1<栏位>-2<栏位>” |
| -1<栏位>   | 加入文件 1 的此栏        |
| -2<栏位>   | 加入文件 2 的此栏        |
| -o<格式>   | 指定格式              |

◆ 执行范例 ◆

- 范例 347 ○ 将两个文件中内容相同的行连接起来。

```
[root@localhost ~]# cat xuehao //显示文件 xuehao 的信息
```

```
A 001
```

```
B 002
```

### 3.15 join 指令：将两个文件中与指定栏位内容相同的行连接起来

```
C 003
D 004
E 005
F 006
G 007
H 008
I 009
J 010
K 011
L 012
M 013
N 014
O 015
P 016
Q 017
R 018
S 019
T 020

[root@localhost ~]# cat sushe
```

//显示文件sushe的信息

```
A xy101
B xy102
C xy103
D xy104
E xy105
F xy106
G xy107
H xy108
I xy109
J xy110
K xy111
L xy112
M xy113
N xy114
O xy115
P xy116
Q xy117
R xy118
S xy119
T xy120
```



### 第3章 文本编辑

```
[root@localhost ~]# join xuahao sushe //连接两个文件
```

```
A 001 xy101
B 002 xy102
C 003 xy103
D 004 xy104
E 005 xy105
F 006 xy106
G 007 xy107
H 008 xy108
I 009 xy109
J 010 xy110
K 011 xy111
L 012 xy112
M 013 xy113
N 014 xy114
O 015 xy115
P 016 xy116
Q 017 xy117
R 018 xy118
S 019 xy119
T 020 xy120
```

```
[root@localhost ~]# join sushe xuahao //连接两个文件
```

```
A xy101 001
B xy102 002
C xy103 003
D xy104 004
E xy105 005
F xy106 006
G xy107 007
H xy108 008
I xy109 009
J xy110 010
K xy111 011
L xy112 012
M xy113 013
N xy114 014
O xy115 015
P xy116 016
Q xy117 017
```

## 3.16 less 指令：一次显示一页文本

```
R xy118 018
S xy119 019
T xy120 020
[root@localhost ~]#
```

本例使用指令“join xuehao sushe”，将文件 xuehao 和文件 sushe 相同的部分连接起来，输出结果如上所示。同时看到，如果将文件 1 和文件 2 的位置调换，输出的结果是不同的，如上所示。

**相关指令**

cat

## 3.16 less 指令：一次显示一页文本

**语法** less[参数][文件名称]

**功能说明** less 指令与 more 指令类似，但可以通过翻页键查看上下页的内容。

**参数说明**

必要参数的具体说明如下：

| 参数         | 功能                          |
|------------|-----------------------------|
| -b <缓冲区大小> | 设置缓冲区的大小                    |
| -e         | 当文件显示结束后，自动离开               |
| -f         | 强迫打开特殊文件，例如外围设备代号、目录和二进制文件等 |
| -g         | 只标志最后搜索到的关键词                |
| -i         | 忽略搜索时的大小写                   |
| -m         | 显示类似 more 指令的百分比            |
| -N         | 显示每行的行号                     |
| -o <文件名>   | 将 less 输出的内容在指定文件中保存起来      |
| -Q         | 不使用警告音                      |
| -s         | 显示连续空行为一行                   |
| -S         | 行过长时将超出部分舍弃                 |
| -x <数字>    | 将“tab”键显示为规定数字的空格           |



执行 less 中参数的具体说明如下：

| 参 数 | 功 能        |
|-----|------------|
| b   | 向后翻一页      |
| d   | 向后翻半页      |
| h   | 显示帮助界面     |
| Q   | 退出 less 指令 |
| u   | 向前滚动半页     |
| y   | 向前滚动一行     |
| 回车键 | 滚动一行       |
| 空格键 | 滚动一页       |

### 执行范例

#### ◎ 范例 348 ◎ 显示文件

```
[root@wjc root]# less juddi.log
```

其显示如下。

```
2007-01-22 20:51:10,066 [main] INFO org.apache.juddi.registry.
RegistryServlet - Loading jUDDI configuration.
2007-01-22 20:51:10,083 [main] INFO org.apache.juddi.registry.
RegistryServlet - Resources loaded from: /WEB-INF/juddi.properties
2007-01-22 20:51:10,083 [main] INFO org.apache.juddi.registry.
RegistryServlet - Initializing jUDDI components.
2007-01-23 02:50:30,617 [main] INFO org.apache.juddi.registry.
RegistryServlet - Loading jUDDI configuration.
2007-01-23 02:50:30,650 [main] INFO org.apache.juddi.registry.
RegistryServlet - Resources loaded from: /WEB-INF/juddi.properties
2007-01-23 02:50:30,651 [main] INFO org.apache.juddi.registry.
RegistryServlet - Initializing jUDDI components.
2007-01-23 02:55:44,776 [main] INFO org.apache.juddi.registry.
RegistryServlet - jUDDI Stopping: Cleaning up existing resources
```

### 相关指令

more

## 3.17 more 指令：显示文件信息

 **语法** more[选择性参数][必要参数]

 **功能说明** 按屏幕输出指定文件的内容。More 命令类似于 cat，但是比 cat 功能强。按空格下翻一页，按“b”上翻一页。

 **参数说明**

必要参数的具体说明如下：

| 参数   | 功    能   |
|------|----------|
| <文件> | 指定要显示的文件 |

选择性参数的具体说明如下：

| 参数     | 功    能                   |
|--------|--------------------------|
| -d     | 当用户错误输入时产生提示符            |
| -l     | 取消遇见特殊字元“^L”(送纸字元)时的暂停功能 |
| -f     | 计算实际行数                   |
| -p     | 不捲页显示，而是消屏显示以后的内容        |
| +<字符串> | 搜寻指定字符串，并从此处开始显示         |
| +num   | 从指定行开始显示                 |

 **执行范例**

◎ 范例 349 ◎ 显示文件 yy，当错误时提示。

```
[root@yingzheng yingzheng]# more -d yy
Hello, this is file yy!
[Press 'h' for instructions.] //此处没有使用指定的键
```

 **相关指令**

vi, less



## 3.18 nano 指令：文本编辑器

### ❖ 语法

**nano[选择性参数][必要参数][文件]**

### ❖ 功能说明

**nano** 指令是一个以取代 **pico** 指令为目标的小的、免费的、用户友好的编辑器。不只是复制 **pico** 指令的界面，**nano** 指令同时执行一些在 **pico** 中没有的功能，例如查找、替换和跳转到指定行数的功能。

### ❖ 选择性说明

必要参数的具体说明如下：

| 参数 | 功能                      |
|----|-------------------------|
| -A | 启动 smart home key 功能    |
| -B | 支持已经存在的保存文件             |
| -F | 支持多文件缓存                 |
| -H | 历史记录                    |
| -I | 忽略 nanorc 文件            |
| -N | 不从 DOS/Mac 格式转换文件       |
| -O | 引用字符串                   |
| -R | 搜索                      |
| -c | 显示光标位置                  |
| -d | 解决空格键和“Del”键混淆的问题       |
| -k | 剪切                      |
| -l | 如果是符号链接，将不修改原文件而是产生新的文件 |

选择性参数的具体说明如下：

| 参数      | 功能          |
|---------|-------------|
| -<行数>   | 跳转到指定行数     |
| -h      | 帮助信息        |
| -E<文件夹> | 指定备份文件存放的目录 |

**【编辑指令】**

- Ctrl+G: 帮助菜单。
- Ctrl+X: 关闭文件。
- Ctrl+O: 保存。
- Ctrl+R: 插入文件。
- Ctrl+W: 搜索特定的文件。
- Ctrl+K: 删除一行。
- Ctrl+U: 回到上一步。
- Ctrl+A: 移到行的开始处。
- Ctrl+E: 行的末尾。
- Ctrl+D: 删除光标所在的字符。
- Ctrl+\: 替换。

**◎执行范例****○范例 350 ○ 编辑文件。**

```
[root@localhost ~]# nano 123 //编辑文件 123
GNU nano 1.3.5-cvs
a
ab
123
1234
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefg hij
abcdefg hijk
abcdefg hijkl
[Read 12 lines]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Our Pos
^K Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Txt ^T To Spell
```

本例使用指令“nano 123”编辑文件 123。

**○范例 351 ○ 直接跳转到指定行进行编辑。**

```
[root@localhost ~]# nano +5 123 //编辑文件 123, 光标跳转到第 5 行
GNU nano 1.3.5-cvs
File: 123
```



```
a
ab
123
1234
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
abcdefhijk
abcdefhijkl
```

[ Read 12 lines ]

```
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C
Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Txt
^T To Spell
```

本例使用指令“nano +5 123”编辑文件，并且打开文件时，直接跳转到第5行进行编辑，这时光标停留在文件的第5行。

### ◆ 相关指令 ◆

pico

## 3.19 pg 指令：浏览文件

**◆ 语法 ◆** pg[必要参数][选择性参数][文件]

**◆ 功能说明 ◆** pg 指令用来显示文件内容。

**◆ 大家说明 ◆**

必要参数的具体说明如下：

| 参 数 | 功 能        |
|-----|------------|
| -c  | 先清屏再显示文件内容 |
| -e  | 显示所有文件     |
| -f  | 过长的行不换行    |

续表

| 参数   | 功能        |
|------|-----------|
| +num | 从指定行数开始显示 |
| -num | 每页显示的行数   |

选择性参数的具体说明如下：

| 参数      | 功能         |
|---------|------------|
| -p<字符串> | 指定字符串作为提示符 |

### 执行范例

#### ◎ 范例 352 ◎ 显示文件的内容。

```
root@localhost:~/# pg iptables.sh //显示文件的内容
#
echo "Enable IP Forwarding..."
echo 1 >/proc/sys/net/ipv4/ip_forward
echo "Starting iptables rules..."
/sbin/modprobe iptable_filter
/sbin/modprobe ip6table_filter
/sbin/modprobe ip_tables
/sbin/modprobe ip6_tables
/sbin/modprobe iptable_nat
/sbin/modprobe ip_nat_ftp
/sbin/modprobe ip_conntrack_ftp
iptables -F INPUT
iptables -F FORWARD
iptables -F OUTPUT
iptables -F POSTROUTING -t nat
iptables -F PREROUTING -t nat
#iptables -P INPUT DROP
#iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
iptables -A FORWARD -d 192.168.88.88 -j ACCEPT
iptables -t nat -A PREROUTING -d 202.112.146.70 -p tcp --dport 3389 -j
DNAT --to-destination 192.168.88.88:3389
iptables -A INPUT -i lo -j ACCEPT
```



```

iptables -A INPUT -i eth0 -j ACCEPT
iptables -A INPUT -i eth1 -p icmp -j ACCEPT
iptables -A INPUT -i eth1 -p udp --dport 2304 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 21:23 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 8888 -j ACCEPT
#iptables -A INPUT -i eth1 -p udp --dport 177 -j ACCEPT
iptables -A INPUT -i eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A FORWARD -s 192.168.88.0/24 -d 0/0 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.88.0/24 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 192.168.88.88 -j SNAT --to-source
202.112.146.70

```

(EOF):

本例使用指令“pg iptables.sh”显示文件 iptables.sh 的内容。

◎范例 353 ◎ 设置提示符。

```

root@localhost:~# pg -p aisinile iptables.sh //显示文件内容并设置提示符
#
echo "Enable IP Forwarding..."
echo 1 >/proc/sys/net/ipv4/ip_forward
echo "Starting iptables rules..."
/sbin/modprobe iptable_filter
/sbin/modprobe ip6table_filter
/sbin/modprobe ip_tables
/sbin/modprobe ip6_tables
/sbin/modprobe iptable_nat
/sbin/modprobe ip_nat_ftp
/sbin/modprobe ip_conntrack_ftp
iptables -F INPUT
iptables -F FORWARD
iptables -F OUTPUT
iptables -F POSTROUTING -t nat
iptables -F PREROUTING -t nat
#iptables -P INPUT DROP
#iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
iptables -A FORWARD -d 192.168.88.88 -j ACCEPT

```

## 3.20 pico 指令：编辑文字文件

第3章

```

iptables -t nat -A PREROUTING -d 202.112.146.70 -p tcp --dport 3389 -j
DNAT --to-destination 192.168.88.88:3389
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -i eth0 -j ACCEPT
iptables -A INPUT -i eth1 -p icmp -j ACCEPT
iptables -A INPUT -i eth1 -p udp --dport 2304 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 21:23 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -i eth1 -p tcp --dport 8888 -j ACCEPT
#iptables -A INPUT -i eth1 -p udp --dport 177 -j ACCEPT
iptables -A INPUT -i eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A FORWARD -s 192.168.88.0/24 -d 0/0 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.88.0/24 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 192.168.88.88 -j SNAT --to-source
202.112.146.70
(EOF)aisinile

```

本例使用指令“`pg -p aisinile iptables.sh`”显示文件内容并设置提示符为“aisinile”，取代默认的提示符“：“。



more

## 3.20 pico 指令：编辑文字文件



pico[必要参数][选择性参数][文件]



pico 指令是一个编辑文字文件的工具，它提供互动式的操作界面，比较方便。



必要参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -b | 替换功能 |
| -d | 删除功能 |



续表

| 参数 | 功能      |
|----|---------|
| -e | 文件的完整名称 |
| -k | 剪切      |
| -m | 支持鼠标    |
| -t | 工具模式    |
| -v | 只读，无法编辑 |

选择性参数的具体说明如下：

| 参数       | 功能       |
|----------|----------|
| +<行数>    | 打开时跳到指定行 |
| -o<目录>   | 指定目录     |
| -help    | 帮助信息     |
| -version | 版本信息     |

#### 【进入指令后的操作】

- Ctrl+G：帮助菜单。
- Ctrl+X：关闭。
- Ctrl+O：保存。
- Ctrl+R：插入文件。
- Ctrl+W：搜索。
- Ctrl+K：删除行。
- Ctrl+U：回到上一步。
- Ctrl+A：该行的最前面。
- Ctrl+E：该行的最后面。
- Ctrl+D：删除行。
- Ctrl+\: 替换。

#### ● 执行范例

##### ◎ 范例 354 ◎ 编辑文件。

```
root@localhost:~# pico iptables.sh //编辑文件 iptables.sh
GNU nano 2.0.1 File: iptables.sh
```

### 3.21 sed 指令：利用 script 命令处理文本文件

```
#!/bin/sh
echo "Enable IP Forwarding..."
echo 1 >/proc/sys/net/ipv4/ip_forward
echo "Starting iptables rules..."
/sbin/modprobe iptable_filter
/sbin/modprobe ip6table_filter
/sbin/modprobe ip_tables
/sbin/modprobe ip6_tables
/sbin/modprobe iptable_nat
/sbin/modprobe ip_nat_ftp
/sbin/modprobe ip_conntrack_ftp
iptables -F INPUT
iptables -F FORWARD
iptables -F OUTPUT
iptables -F POSTROUTING -t nat
iptables -F PREROUTING -t nat
#iptables -P INPUT DROP
[Read 41 lines]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

本例使用指令“pico iptables.sh”编辑文件 iptables.sh。



vi、vim

## 3.21 sed 指令：利用 script 命令处理文本文件

**语法** sed[必要参数][选择性参数][文本文件]

**功能说明** sed 指令可以利用 script 指令来处理指定的文本文件。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能      |
|----|---------|
| -n | 不显示处理过程 |



选择性参数的具体说明如下：

| 参数            | 功能           |
|---------------|--------------|
| -e<script>    | 指定 script    |
| -f<script 文件> | 指定 script 文件 |
| -h            | 帮助信息         |
| --version     | 版本信息         |
| -l<字符数>       | 指定每行的字符数     |

### 【script 指令】

- [行号]: 指定行号。
- a\<字符串>: 追加字符串。
- c\<字符串>: 替代。
- d: 删除行。
- i\<字符串>: 行前追加字符串。
- p: 显示行。
- y/<查找字符串>/<替代字符串>; 用替代字符串替代查找的字符串。
- r<文本>; 处理文本。
- s/<查找字符串>/<替代字符串>/<替代方式>; 以指定的替代方式用替代字符串替代查找字符串。

### ◎ 实例范例

#### ● 范例 355 ● 字符的替代。

```
[root@localhost ~]# cat examplesed //显示文件内容
1 i love intermilan
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e s/ve/aisinile/ examplesed //替代字符
```

### 3.21 sed 指令：利用 script 命令处理文本文件

```
1 i loaisinile intermilan
2 i loaisinile intermilan
3 i loaisinile intermilan
4 i loaisinile intermilan
5 i loaisinile intermilan
6 i loaisinile intermilan
7 i loaisinile intermilan
8 i loaisinile intermilan
9 i loaisinile intermilan
[root@localhost ~]#
```

本例使用指令“`sed -e s/ve/aisinile/ examplesed`”替代文件 `examplesed` 中的字符串 “ve” 为 “aisinile”，并将结果输出到标准输出。

◎ 范例 356 ◎ 只替换指定行数的字符串。

```
[root@localhost ~]# cat examplesed //显示文件内容
1 i love intermilan
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e 1,2s/vo/aisinile/ examplesed //替换指定行的字符
1 i loaisinile intermilan
2 i loaisinile intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]#
```

本例使用指令“`sed -e 1,2s/ve/aisinile/ examplesed`”替换文件 `examplesed` 中第 1、2 行中的字符串 “ve” 为 “aisinile”。

◎ 范例 357 ◎ 在行后追加字符串。



### 第3章 文本编辑

```
[root@localhost ~]# cat examplesed //显示文件内容
1 i love intermilan
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e /1/a\lovely examplesed //追加字符串
1 i love intermilan
lovely
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e /2/a\lovely examplesed //追加字符串
1 i love intermilan
2 i love intermilan
lovely
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]#
```

本例使用指令“`sed -e /2/a\lovely examplesed`”在含有字符 2 的行后追加一行，追加行的字符串内容为 lovely。

◎ 范例 358 ◎ 在行前追加字符串。

```
[root@localhost ~]# cat examplesed //显示文件内容
1 i love intermilan
```

### 3.21 sed 指令：利用 script 命令处理文本文件

```

2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e /2/i\lovely examplesed //追加字符串
1 i love intermilan
lovely
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]#

```

本例使用指令“`sed -e /2/i\lovely examplesed`”在含有字符串 2 的行前追加一行，追加行的字符串内容为“`lovely`”。

#### ◎ 范例 359 ◎ 删除指定的行。

```

[root@localhost ~]# cat examplesed //显示文件内容
1 i love intermilan
2 i love intermilan
3 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]# sed -e 2,3d examplesed //删除指定的行
1 i love intermilan
4 i love intermilan
5 i love intermilan
6 i love intermilan

```



```
7 i love intermilan
8 i love intermilan
9 i love intermilan
[root@localhost ~]#
```

本例使用指令“`sed -e 2,3d examplesed`”显示文件 `examplesed` 中的第 2、3 行。

◎ 范例 360 ◎ 替换指定的字符串，并且将替代的行重新显示一遍。

```
[root@localhost ~]# sed -e s/ve/aisinile/g examplesed //显示文件内容
1 i loaisinile intermilan
2 i loaisinile intermilan
3.i loaisinile intermilan
4 i loaisinile intermilan
5 i loaisinile intermilan
6 i loaisinile intermilan
7 i loaisinile intermilan
8 i loaisinile intermilan
9 i loaisinile intermilan
[root@localhost ~]# sed -e s/ve/aisinile/p examplesed //替代字符,重
复显示替代的行
```

```
1 i loaisinile intermilan
1 i loaisinile intermilan
2 i loaisinile intermilan
2 i loaisinile intermilan
3 i loaisinile intermilan
3 i loaisinile intermilan
4 i loaisinile intermilan
4 i loaisinile intermilan
5 i loaisinile intermilan
5 i loaisinile intermilan
6 i loaisinile intermilan
6 i loaisinile intermilan
7 i loaisinile intermilan
7 i loaisinile intermilan
8 i loaisinile intermilan
8 i loaisinile intermilan
9 i loaisinile intermilan
9 i loaisinile intermilan
[root@localhost ~]#
```

## 3.22 sort 指令：将文本文件内容加以排序

本例使用指令“`sed -e s/ve/aisinile/p examplesed`”替换字符串“ve”为“aisinile”，并且将发生替换的行重新显示一遍。



`awk`、`ed`、`grep`、`tr`、`perlre`

## 3.22

## sort 指令：将文本文件内容加以排序



`sort[必要参数][选择性参数][文件]`



`sort` 指令可以对文本文件进行排序。

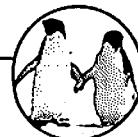


必要参数的具体说明如下：

| 参数              | 功能                          |
|-----------------|-----------------------------|
| <code>-b</code> | 忽略行前的空格                     |
| <code>-c</code> | 检查是否已排序过                    |
| <code>-d</code> | 排序时只识别英文数字及空格               |
| <code>-f</code> | 把小写字母看作大写字母                 |
| <code>-l</code> | 排序时只识别 040~176 范围内的 ASCII 码 |
| <code>-m</code> | 合并排序好的文件                    |
| <code>-n</code> | 按照数值排序                      |
| <code>-r</code> | 逆序排列                        |

选择性参数的具体说明如下：

| 参数                                    | 功能        |
|---------------------------------------|-----------|
| <code>-o&lt;文件&gt;</code>             | 结果输出到指定文件 |
| <code>-t&lt;分隔符&gt;</code>            | 指定分隔符     |
| <code>-help</code>                    | 帮助信息      |
| <code>--version</code>                | 版本信息      |
| <code>+&lt;起始行&gt;-&lt;结束行&gt;</code> | 指定范围      |

**执行范例**

○ 范例 361 ○ 按照指定规则排序。

```
-sh-2.05b# ll //显示文件
总用量 6800
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda- ks.cfg
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install. log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.
```

log.syslog

```
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick. miff
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 6649992 5月 28 20:37 Real
```

Player10GOLD.rpm

```
-sh-2.05b# ll|sort +4 //按照第5栏排序
```

总用量 6800

```
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda- ks.cfg
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick. miff
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install. log.bak
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.
```

log.syslog

```
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 6649992 5月 28 20:37 RealPlay
```

## 3.22 sort 指令：将文本文件内容加以排序

er10GOLD.rpm

```
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-sh-2.05b#
```

本例使用指令“ll|sort +4”显示当前文件夹下的文件信息，并以第 5 栏为参考进行排序，然后输出到标准输出。

以下是以其他栏为参考进行排序后输出的结果：

```
-sh-2.05b# ll | sort +3 //按照第 4 栏排序
总用量 6800
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda- ks.cfg
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 4138 4月 20 19:28 install. log.
```

syslog

```
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 6649992 5月 28 20:37 RealPlayer
```

10GOLD.rpm

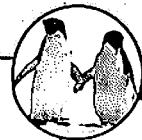
```
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-sh-2.05b# ll | sort +6 //按照第 7 栏排序
```

总用量 6800

```
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log. syslog
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks. cfg
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick. miff
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 6649992 5月 28 20:37 Real
```

Player10GOLD.rpm

```
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 27035 5月 3 22:46 install. log. bak
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
```



### 第3章 文本编辑

```

-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-sh-2.05b# 11 | sort +7 //按照第8栏进行排序
总用量 6800

-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.

syslog
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks. cfg
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 6649992 5月 28 20:37 RealPl
ayer10GOLD.rpm
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log. bak
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-sh-2.05b#
-sh-2.05b# 11 | sort +8 //按照第9栏进行排序
总用量 6800

-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks. cfg
-rw-r--r-- 1 root root 180 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log. bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.

syslog
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2

```

## 3.22 sort 指令：将文本文件内容加以排序

```
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 6649992 5月 28 20:37 Real Play
er10GOLD.rpm
```

-sh-2.05b#

以上是按照不同栏位进行排序的结果。

◎ 范例 362 ◎ 对文件内容进行排序。

-sh-2.05b# cat temp //显示文件内容

```
1 i am a dog 3
3 i am a dog 7
5 i am a dog 8
8 i am a dog 9
2 i am a dog 2
4 i am a dog 6
9 i am a dog 1
7 i am a dog 4
6 i am a dog 0
0 i am a dog 5
```

-sh-2.05b# sort temp //文件排序

```
0 i am a dog 5
1 i am a dog 3
2 i am a dog 2
3 i am a dog 7
4 i am a dog 6
5 i am a dog 8
6 i am a dog 0
7 i am a dog 4
8 i am a dog 9
9 i am a dog 1
```

-sh-2.05b# sort +5 temp //按照第 6 栏对文件进行排序

```
6 i am a dog 0
9 i am a dog 1
2 i am a dog 2
1 i am a dog 3
7 i am a dog 4
0 i am a dog 5
4 i am a dog 6
```



```
3 i am a dog 7
5 i am a dog 8
8 i am a dog 9
-sh-2.05b#
```

本例首先使用指令“sort temp”对文件temp排序后输出到标准输出，  
默认情况下按照第1栏进行排序；然后使用指令“sort +5 temp”以第6栏  
为依据对文件temp进行排序，然后将结果输出到标准输出。

◎ 范例 363 ◎ 多个文件混合排序。

```
-sh-2.05b# cat temp1 //显示文件内容
```

```
3 i am a dog 7
5 i am a dog 8
4 i am a dog 6
6 i am a dog 0
0 i am a dog 5
```

```
-sh-2.05b# cat temp2 //显示文件内容
```

```
1 i am a dog 3
8 i am a dog 9
7 i am a dog 4
```

```
-sh-2.05b# cat temp3 //显示文件内容
```

```
2 i am a dog 2
9 i am a dog 1
```

```
-sh-2.05b# sort temp1 temp2 temp3 //多个文件排序
```

```
0 i am a dog 5
1 i am a dog 3
2 i am a dog 2
3 i am a dog 7
4 i am a dog 6
5 i am a dog 8
6 i am a dog 0
7 i am a dog 4
8 i am a dog 9
9 i am a dog 1
```

```
-sh-2.05b#
```

本例使用指令“sort temp1 temp2 temp3”对多个文件temp1、temp2、  
temp3进行排序，并将结果输出到标准输出。



sed

**3.23****spell 指令：拼字检查程序**

**语法** spell

**功能说明** spell 指令检查输入的字符串是否出错。

**使用范围**

● 范例 364 ● 检查输入的字符串。

[root@localhost ~]# spell //检查字符串

本例使用指令 spell 检查输入的字符串的拼写错误。

**相关指令**

cat

**3.24****tr 指令：转换文件中的字符**

**语法** tr[必要参数][选择性参数][字符串 1][字符串 2]

**功能说明** tr 指令将标准输入输入的字符串转换为指定的字符串，然后输出到标准输出。

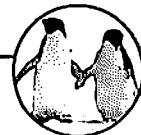
**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能                                         |
|-----|---------------------------------------------|
| -c  | 取代不输入字符串 1 的所有字符                            |
| -d  | 删除与字符串 1 不匹配的字符                             |
| -s  | 若有多个同样的字符，则只显示一个                            |
| -t  | 若字符串 2 的长度大于字符串 1 的长度，则删除字符串 2 中的字符使得两者长度相同 |

选择性参数的具体说明如下：

| 参 数       | 功 能  |
|-----------|------|
| -help     | 帮助信息 |
| --version | 版本信息 |


◆ **执行范例**
**◎ 范例 365 ◎ 将小写字母转换为大写字母。**

```
[root@localhost ~]# cat abc //显示文件内容
```

```
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefg hij
```

```
[root@localhost ~]# cat abc |tr a-z A-Z//将小写字母转换为大写字母
```

```
A
AB
ABC
ABCD
ABCDE
ABCDEF
ABCDEFG
ABCDEFGH
ABCDEFGHI
ABCDEFGHIJ
```

```
[root@localhost ~]#
```

本例使用指令“cat abc |tr a-z A-Z”将文件 abc 的小写字母转换为大写字母，并输出到标准输出。

**◎ 范例 366 ◎ 更改字符。**

```
[root@localhost ~]# cat abc //显示文件内容
```

```
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
```

## 3.25 uniq 指令：检查文件中重复出现的行

```
abcdefgij
[root@localhost ~]# cat abc | tr c 4 //将文件中的“c”替换为“4”
a
ab
ab4
ab4d
ab4de
ab4def
ab4defg
ab4defgh
ab4defghi
ab4defghi
[root@localhost ~]
```

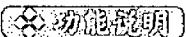
本例使用指令“cat abc | tr c 4”将文件 abc 中的所有的“c”都用“4”替换。

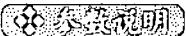
## ◆ 相关指令

awk、sed

## 3.25 uniq 指令：检查文件中重复出现的行

 **语义** uniq[必要参数][选择性参数][文件]

 **功能说明** uniq 指令用来检查文件中重复出现的行列，并将重复的部分删除。

 **参数说明**

必要参数的具体说明如下：

| 参    数 | 功    能   |
|--------|----------|
| -c     | 显示重复次数   |
| -d     | 显示重复的行   |
| -u     | 显示没有重复的行 |

选择性参数的具体说明如下：



| 参数       | 功能       |
|----------|----------|
| -f<列>    | 忽略指定的列   |
| -s<字符>   | 忽略指定的字符  |
| -w<字符>   | 指定要比较的字符 |
| -help    | 帮助信息     |
| -version | 版本信息     |

### 执行范例

#### ◎ 范例 367 ◎ 删 除重 复的行。

```
[root@localhost ~]# cat test //显示文件内容
aaaaaaaaaaaaaaaaaa
bbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbb
cccccccccccccccccc
cccccccccccccccccc
dddddddddcccccccc
[root@localhost ~]# uniq test //不显示重复的行
aaaaaaaaaaaaaaaaaa
bbbbbbbbbbbbbbbbbb
cccccccccccccccccc
dddddddddcccccccc
```

本例使用指令“uniq test”不显示文件 test 中的重复的行。

#### ◎ 范例 368 ◎ 显 示重 复的行。

```
[root@localhost ~]# cat test //显示文件内容
aaaaaaaaaaaaaaaaaa
bbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbb
cccccccccccccccccc
cccccccccccccccccc
dddddddddcccccccc
[root@localhost ~]# uniq -d test //仅显示重复的行
bbbbbbbbbbbbbbbbbb
cccccccccccccccccc
[root@localhost ~]#
```

本例使用指令“`uniq -d test`”显示文件 `test` 中重复的行。

### 相关指令

`sort`

## 3.26 vi 指令：文字编辑器

**语法** vi[必要参数][选择性参数][文件]

**功能说明** vi 指令是 Linux 默认的文字编辑器，在命令行界面上编辑文字。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -b | 二进制模式        |
| -d | 差异模式         |
| -m | 只读模式         |
| -n | 缓存不使用        |
| -r | 显示缓存信息       |
| -R | 只读模式         |
| -s | 运行时不显示处理信息   |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功能          |
|----------|-------------|
| +<行数>    | 从指定行数开始显示   |
| -C<指令>   | 设置编辑完成之后的指令 |
| -help    | 帮助信息        |
| -o<数字>   | 指定同时打开的文件数  |
| -version | 版本信息        |



### 【进入 vi 后的指令】

- Esc: 退出编辑。
- :: 输入指令。
- 回车: 显示模式。
- ZZ: 保存退出。

### 显示模式下

- Ctrl+d: 向下翻半页。
- Ctrl+u: 向上翻半页。
- Ctrl+f: 向下翻一页。
- Ctrl+b: 向上翻半页。
- : 数字: 跳到指定行。
- : \$: 转到文件最后一行。
- x: 删除行。
- X: 删除字符。
- dd: 删除行。
- ndd: 删除 n 行。
- Y: 复制行。
- nyy: 删除 n 行。
- p: 复制。
- P: 复制。
- /key: 向光标之后查找字符。
- ?name: 向光标之前查找字符。
- a,b s/F/T: a 行之后 b 行之前, 用 T 字符串代替 F 字符串。

### 编辑模式下

- a: 从光标之后插入。
- A: 从光标行的最后插入。
- i: 从光标处插入。
- I: 从光标后的非空字符插入。
- o: 从光标下方新建行。
- O: 从光标上方新建行。

## 操作模式下

- : w: 存盘。
- : q: 退出。
- : !: 强制执行。
- : n: 编辑下个文件。
- : f: 显示当前信息。
- : set nu: 显示行数。

**◎范例与例**

## ◎ 范例 369 ◎ 编辑文件。

```
[root@localhost ~]# vi abc //编辑文件 abc
```

```
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
"abc" 10L, 65C
```

本例使用指令“vi abc”编辑文件 abc，现在是显示模式。

通过输入“: set nu”显示行号。

```
1 a
2 ab
3 abc
4 abcd
5 abcde
6 abcdef
7 abcdefg
8 abcdefgh
9 abcdefghi
10 abcdefhij
:set nu
```

在显示模式下，按字母“o”键，在光标所在的行下加入一行。



```

1 a
2 a new ling
3 ab
4 abc
5 abcd
6 abcde
7 abcdef
8 abcdefg
9 abcdefgh
10 abcdefghi
11 abcdefghij
-- INSERT --

```

按退出键，返回显示模式，然后输入“: wq”确认，保存后退出。

```

1 a
2 a new ling
3 ab
4 abc
5 abcd
6 abcde
7 abcdef
8 abcdefg
9 abcdefgh
10 abcdefghi
11 abcdefghij

```

:wq

如果不保存更改，可以键入“: q!”，回车，退出。

```

a
a new ling
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefghij

```

:q!

## ○范例 370 ○ 显示版本信息。

```
[root@localhost ~]# vi --version //显示版本信息
VIM - Vi IMproved 6.3 (2004 June 7, compiled Apr 27 2005 02:24:01)
Included patches: 1-21, 23-24, 26, 28-34, 36-37, 39-40, 42-43, 45-46, 49-51,
54-71
Compiled by <bugzilla@redhat.com>
Tiny version without GUI. Features included (+) or not (-):
+arabic -autocmd +balloon_eval +browse +builtin_terms +byte_offset
+cindent
+clientserver +clipboard +cmdline_compl +cmdline_hist +cmdline_info
+comments
+cryptv +cscope +dialog +diff +digraphs +dnd +ebcdic +emacs_tags +eval
+ex_extra +extra_search +farsi +file_in_path +find_in_path +folding
+footer
+fork() +gettext +hangul_input +iconv +insert_expand +jumplist +keymap
+langmap
+libcall +linebreak +lispindent +listcmds +localmap +menu +mksession
+modify_fname +mouse +mouse_dec +mouse_gpm +mouse_jsbterm +mouse_netterm
+mouse_xterm +multi_byte +multi_lang +netbeans_intg +osfiletype
+path_extra
+perl +printer +python +quickfix +rightleft +ruby +scrollbind +signs
+smartindent +sniff +statusline +sun_workshop +syntax +tag_binary
+tag_old_static +tag_any_white +tcl +terminfo +termresponse +textobjects
+title
+toolbar +user_commands +vertsplit +virtualedit +visual +viminfo
+vreplace
+wildignore +wildmenu +windows +writebackup +X11 +xfontset +xim +xsmp
+xterm_clipboard +xterm_save
 system vimrc file: "/etc/vimrc"
 user vimrc file: "$HOME/.vimrc"
 user exrc file: "$HOME/.exrc"
 fall-back for $VIM: "/usr/share/vim"
Compilation: i386-redhat-linux-gcc -c -I . -Iproto -DHAVE_CONFIG_H -O2 -g
-pipe -Wp,-D_FORTIFY_SOURCE=2 -fexceptions -m32 -march=i386 -mtune=pentium4
-fasynchronous-unwind-tables -D_GNU_SOURCE -D_FILE_OFFSET_BITS=64
-D_FORTIFY_SOURCE=2
Linking: i386-redhat-linux-gcc -L/usr/local/lib -o vim -lselinux
-ltermcap -lacl
[root@localhost ~]#
```



本例使用指令“vi –version”显示 vi 的版本信息。

**◆ 相关指令**

vim

## 3.27

### vim 指令：增强型 vi 编辑器

**◆ 语法** vim[必要参数][选择性参数][文件]

**◆ 功能说明** vim 指令是 vi 指令的增强版。

**◆ 参数说明**

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -b | 二进制模式        |
| -d | 差异模式         |
| -m | 只读模式         |
| -n | 缓存不使用        |
| -r | 显示缓存信息       |
| -R | 只读模式         |
| -s | 运行时不显示处理信息   |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功能          |
|----------|-------------|
| +<行数>    | 从指定行数开始显示   |
| -C<指令>   | 设置编辑完成之后的指令 |
| -help    | 帮助信息        |
| -o<数字>   | 指定同时打开的文件数  |
| -version | 版本信息        |

**【进入 vi 后的指令】**

参见 vi 指令的说明。



vim 指令的执行范例参见 vi 指令。



vi

**3.28****view 指令：文字编辑器**

**view[必要参数][选择性参数][文件]**



view 指令用来编辑文字文件，它连接 vi 指令。



必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -b | 二进制模式        |
| -d | 差异模式         |
| -m | 只读模式         |
| -n | 缓存不使用        |
| -r | 显示缓存信息       |
| -R | 只读模式         |
| -s | 运行时不显示处理信息   |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数     | 功能          |
|--------|-------------|
| +<行数>  | 从指定行开始显示    |
| -C<指令> | 设置编辑完成之后的指令 |
| -help  | 帮助信息        |



续表

| 参数       | 功能         |
|----------|------------|
| -o<数字>   | 指定同时打开的文件数 |
| -version | 版本信息       |

**【进入 vi 后的指令】****参见 vi 指令的说明。****❖ 执行范例**

view 指令的执行范例参见 vi 指令的执行范例。

**❖ 相关指令****vi**

## 3.29 wc 指令：计算字数

**❖ 语法** **wc[必要参数][选择性参数][文件]****❖ 功能说明** **wc** 指令用来计算指定文件的字数，可以以不同的形式显示。**❖ 参数说明**

必要参数的具体说明如下：

| 参数 | 功能    |
|----|-------|
| -c | 显示字符数 |
| -l | 显示行数  |
| -w | 显示单词数 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

 执行范例

- ◎ 范例 371 ◎ 计算文件的字符数。

```
[root@localhost ~]# wc -c 123 //计算字符数
```

```
220 123
```

```
[root@localhost ~]#
```

本例使用指令“wc -c 123”计算文件 123 的字符数。

- ◎ 范例 372 ◎ 计算文件的行数。

```
[root@localhost ~]# wc -l 123 //计算文件的行数
```

```
10 123
```

```
[root@localhost ~]#
```

本例使用指令“wc -l 123”计算文件 123 的行数。

- ◎ 范例 373 ◎ 显示单词数。

```
[root@localhost ~]# wc -w 123 //显示单词数
```

```
20 123
```

```
[root@localhost ~]#
```

本例使用指令“wc -w 123”显示文件 123 的单词数。

- ◎ 范例 374 ◎ 显示以上所有信息。

```
[root@localhost ~]# wc 123 //显示字数
```

```
10 20 220 123
```

```
[root@localhost ~]#
```

本例使用指令“wc 123”显示文件字数，有 10 行、20 个单词、220 个字符。

 相关命令

cat



## 大家网·精英人才库

精英人才库 - 精英人才库

精英人才库是大家网为精英人才量身定做的求职平台，精英人才库的精英人才都是大家网通过各种途径筛选出来的精英人才。精英人才库的精英人才都是大家网通过各种途径筛选出来的精英人才。

精英人才库的精英人才都是大家网通过各种途径筛选出来的精英人才。

精英人才库的精英人才都是大家网通过各种途径筛选出来的精英人才。

# LINUX

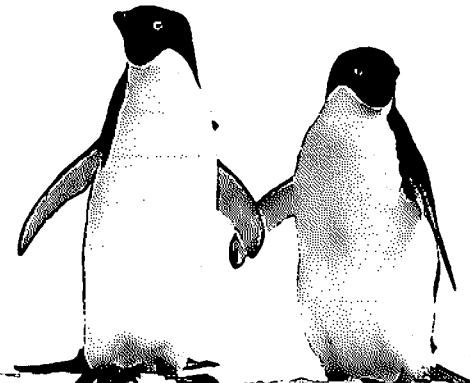
Linux

TopSage.com

## 第4章

### 文件传输

文件传输主要指 Linux 系统中文件的上传和下载，主要包括 ftp 文件的上传、下载、账户查询、图形化界面的 ftp 的文件的上传下载、uucp 文件传输等。本章将详细讲解相关的指令。





## 4.1 bye 指令：中断FTP连接

**语法** bye

**功能说明** 在FTP连接中使用bye指令将直接终止FTP连接，从FTP中正常退出，返回到Linux的命令行界面。

**相关指令**

exit

## 4.2 fold 指令：限制文件列宽

**语法** fold[必要参数][选择性参数][文件]

**功能说明** fold指令用来调整文件的每一行的宽度，写到标准输出。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能        |
|----|-----------|
| -b | 列宽的单位为字节数 |
| -s | 遇到空格换列    |

选择性参数的具体说明如下：

| 参数       | 功能      |
|----------|---------|
| -w<行数>   | 指定每列的行数 |
| -help    | 帮助信息    |
| -version | 版本信息    |

**执行范例**

◎ 范例 375 ◎ 调整文件的列宽。

## 4.2 fold 指令：限制文件列宽

```
[root@localhost ~]# cat abc
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefgij
abcdefgijk
abcdefghijkl
[root@localhost ~]# fold -w 5 abc //调整文件abc的列宽为5个字符
a
ab
abc
abcd
abcde
abcde
f
abcde
fg
abcde
fgh
abcde
fghi
abcde
fghij
abcde
fghij
k
abcde
fghij
kl
[root@localhost ~]#
```

本例使用指令“fold -w 5 abc”调整文件abc的列宽为每列5个字符。



fmt



## 4.3 ftp 指令：文件传输协议

**❖ 语法**    `ftp[必要参数][选择性参数][主机名称或者 IP 地址]`

**❖ 功能说明**    `ftp` 指令是 ARPANET 标准文件传输协议的用户接口。程序允许用户从远端网络站点下载文件或者传输文件到远端网络站点。

**❖ 参数说明**

必要参数的具体说明如下：

| 参    数          | 功    能         |
|-----------------|----------------|
| <code>-d</code> | 运行时显示详细的处理信息   |
| <code>-e</code> | 不记录历史指令        |
| <code>-i</code> | 关闭交互模式         |
| <code>-g</code> | 关闭 tab 功能      |
| <code>-n</code> | 关闭自动登录功能       |
| <code>-p</code> | 传输文件的模式为被动模式   |
| <code>-v</code> | 程序运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参    数                       | 功    能                            |
|------------------------------|-----------------------------------|
| <code>-k&lt;realm&gt;</code> | 当使用 Kerberos v4 认证时，从 realm 中得到信息 |

### 【ftp 内部指令】

- `!:` 执行本地机器的指令。
- `ascii:` 传输模式为 ascii。
- `binary:` 传输模式为二进制模式。
- `bye:` 结束 ftp。
- `cd:` 切换到远端 ftp 服务器上的目录。
- `cdup:` 上一层目录。
- `close:` 在不结束 ftp 进程的情况下，关闭与 ftp 服务器的链接。

- delete:** 删除远端 ftp 服务器上的文件。
- get:** 下载。
- hash:** 显示 “#”。
- lcd:** 更改本地目录。
- ls:** 列出远端 ftp 服务器上的文件列表。
- mdelete:** 删除文件，模糊匹配。
- mget:** 下载文件，模糊匹配。
- mkdir:** 在远端 ftp 服务器上建立文件夹。
- mput:** 上传文件，模糊匹配。
- newer:** 下载时检测是不是新文件。
- open:** 打开连线。
- prompt:** 关闭交互模式。
- put:** 上传。
- pwd:** 显示当前目录。
- quit:** 退出。

### ◎范例范例

#### ◎ 范例 376 ◎ 建立 ftp 连接。

```
[root@localhost ~]# ftp 192.168.88.2 //建立 ftp 连接
Connected to 192.168.88.2.
220 (vsFTPd 2.0.5)
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (192.168.88.2:root): iplab604 //输入用户名
331 Please specify the password.
Password: //输入密码
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls //列出当前目录
227 Entering Passive Mode (192,168,88,2,24,180)
150 Here comes the directory listing.
drwxr-xr-x 9 1020 1020 4096 Apr 04 14:42 863 标书
```



## 第4章 文件传输

```

drwxr-xr-x 14 1020 1020 4096 Apr 26 09:40 chordgroup
drwxr-xr-x 2 1020 1020 4096 Feb 05 08:32 dns
-rw-r--r-- 1 1020 1020 672 Apr 05 17:05 fcntl.c
drwxrwxrwx 10 0 0 4096 Apr 03 16:45 other
-rw-r--r-- 1 1020 1020 421898 Apr 28 08:20 putty.exe
drwxrwxrwx 6 0 0 4096 May 14 14:17 sip
drwxrwxrwx 4 0 0 4096 Mar 14 14:59 snmp
drwxrwxrwx 2 0 0 4096 Mar 28 17:24 transport
-rw-r--r-- 1 1020 1020 975 May 08 09:03 udpserv.c
drwxrwxrwx 3 0 0 4096 Apr 13 15:07 web
drwxr-xr-x 6 1020 1020 4096 Apr 25 18:55 yd&wb
drwxr-xr-x 2 1020 1020 4096 Apr 18 16:32 好书
drwxr-xr-x 2 1020 1020 4096 Apr 13 10:16 技术交流
drwxr-xr-x 5 1020 1020 4096 Mar 30 11:02 专利示例
226 Directory send OK.

```

ftp> cd chordgroup

//更改目录

250 Directory successfully changed.

ftp> pwd

//显示远端当前目录

257 "/home/iplab604/chordgroup"

ftp> cdup

//返回上级目录

250 Directory successfully changed.

ftp> pwd

//显示远端当前目录

257 "/home/iplab604"

ftp> lcd /home/kk

//改变本地目录

Local directory now /home/kk

ftp> !pwd

//显示本地目录

/home/kk

ftp> bye

//结束 ftp 连接

221 Goodbye.

[root@localhost ~]#

以上是执行 ftp 指令的基本操作。

◎ 范例 377 ◎ 上传、下载文件。

[root@localhost ~]# ftp 192.168.88.7 //建立 ftp 连接

Connected to 192.168.88.7.

220 Serv-U FTP Server v5.1 for WinSock ready..

500 'AUTH': command not understood.

500 'AUTH': command not understood.

KERBEROS\_V4 rejected as an authentication type

## 4.3 ftp 指令：文件传输协议

```

Name (192.168.88.7:root): other //输入用户名
331 User name okay, need password.
Password: //输入密码
230 User logged in, proceed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls //显示当前文件夹
227 Entering Passive Mode (192,168,88,7,5,142)
150 Opening ASCII mode data connection for /bin/ls.
drw-rw-rw- 1 user group 0 May 4 11:36 .
drw-rw-rw- 1 user group 0 May 4 11:36 ..
-rw-rw-rw- 1 user group 8997 May 9 09:46 111
drw-rw-rw- 1 user group 0 May 4 17:39 comm
-rw-rw-rw- 1 user group 1471488 May 3 16:10 unpv12e.tar
226 Transfer complete.
ftp> get 111 //下载文件 111
local: 111 remote: 111
227 Entering Passive Mode (192,168,88,7,5,143)
150 Opening BINARY mode data connection for 111 (8997 Bytes).
226 Transfer complete.
8997 bytes received in 0.00017 seconds (5.1e+04 Kbytes/s)
ftp> mget 1* //模糊方式下级文件
mget 111? y
227 Entering Passive Mode (192,168,88,7,5,145)
150 Opening BINARY mode data connection for 111 (8997 Bytes).
226 Transfer complete.
8997 bytes received in 0.00079 seconds (1.1e+04 Kbytes/s)
ftp> !ls //显示本地文件目录
111 anaconda-ks.cfg gmon.out install.log.new lib test
abc Desktop install.log install.log.syslog lib.a workspace
ftp> put test //上传文件 test
local: test remote: test
227 Entering Passive Mode (192,168,88,7,5,148)
150 Opening BINARY mode data connection for test.
226 Transfer complete.
1563 bytes sent in 0.041 seconds (37 Kbytes/s)
ftp> mput te* //以模糊方式上传文件
mput test? y
227 Entering Passive Mode (192,168,88,7,5,149)

```



```
150 Opening BINARY mode data connection for test.
226 Transfer complete.
1563 bytes sent in 0.00021 seconds (7.4e+03 Kbytes/s)
ftp> bye //结束 ftp 连接
221 Goodbye!
[root@localhost ~]#
```

以上为上传、下载文件的一些指令及运行情况。

◎ 范例 378 ◎ 断开连线但不退出 ftp 程序。

```
[root@localhost ~]# ftp 192.168.88.7 //ftp 连线
Connected to 192.168.88.7.
220 Serv-U FTP Server v5.1 for WinSock ready...
500 'AUTH': command not understood.
500 'AUTH': command not understood.
KERBEROS_V4 rejected as an authentication type
Name (192.168.88.7:root): other
331 User name okay, need password.
Password:
230 User logged in, proceed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (192,168,88,7,5,160)
150 Opening ASCII mode data connection for /bin/ls.
drw-rw-rw- 1 user group 0 May 22 10:08 .
drw-rw-rw- 1 user group 0 May 22 10:08 ..
-rw-rw-rw- 1 user group 8997 May 9 09:46 111
drw-rw-rw- 1 user group 0 May 4 17:39 comm
-rw-rw-rw- 1 user group 1563 May 22 10:09 test
-rw-rw-rw- 1 user group 1471488 May 3 16:10 unpv12e.tar
226 Transfer complete.
ftp> close //断开连线
221 Goodbye!
ftp> ls
Not connected.
ftp> open //重新开启连线
(to) 192.168.88.7
Connected to 192.168.88.7.
220 Serv-U FTP Server v5.1 for WinSock ready...
```

## 4.3 ftp 指令：文件传输协议

```

500 'AUTH': command not understood.
500 'AUTH': command not understood.
KERBEROS_V4 rejected as an authentication type
Name (192.168.88.7:root): other
331 User name okay, need password.
Password:
230 User logged in, proceed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (192,168,88,7,5,161)
150 Opening ASCII mode data connection for /bin/ls.
drw-rw-rw- 1 user group 0 May 22 10:08 .
drw-rw-rw- 1 user group 0 May 22 10:08 ..
-rw-rw-rw- 1 user group 8997 May 9 09:46 111
drw-rw-rw- 1 user group 0 May 4 17:39 comm
-rw-rw-rw- 1 user group 1563 May 22 10:09 test
-rw-rw-rw- 1 user group 1471488 May 3 16:10 unpv12e.tar
226 Transfer complete.
ftp> bye
221 Goodbye!
[root@localhost ~]#

```

以上为 ftp 程序里断开连线但不结束 ftp 进程和重新开启 ftp 连线的指令，以及运行过程中的情况。

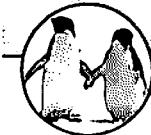
◎ 范例 379 ◎ 删 除 文件。

```

[root@localhost ~]# ftp 192.168.88.7 //建立 ftp 连线
Connected to 192.168.88.7.
220 Serv-U FTP Server v5.1 for WinSock ready...
500 'AUTH': command not understood.
500 'AUTH': command not understood.
KERBEROS_V4 rejected as an authentication type
Name (192.168.88.7:root): other
331 User name okay, need password.
Password:
230 User logged in, proceed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls

```

//显示当前目录



## 第4章 文件传输

```

227 Entering Passive Mode (192,168,88,7,5,172)
150 Opening ASCII mode data connection for /bin/ls.
drw-rw-rw- 1 user group 0 May 22 10:08 .
drw-rw-rw- 1 user group 0 May 22 10:08 ..
-rw-rw-rw- 1 user group 8997 May 9 09:46 111
drw-rw-rw- 1 user group 0 May 4 17:39 comm
-rw-rw-rw- 1 user group 1563 May 22 10:09 test
-rw-rw-rw- 1 user group 1471488 May 3 16:10 unpv12e.tar
226 Transfer complete.

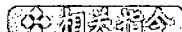
ftp> hash //显示“#”，以显示进度
Hash mark printing on (1024 bytes/hash mark).
ftp> mget 111
mget 111? y
227 Entering Passive Mode (192,168,88,7,5,176)
150 Opening BINARY mode data connection for 111 (8997 Bytes).
#####
226 Transfer complete.
8997 bytes received in 0.0013 seconds (7e+03 Kbytes/s)
ftp> delete 111 //删除文件
250 DELE command successful.

ftp> ls
227 Entering Passive Mode (192,168,88,7,5,177)
150 Opening ASCII mode data connection for /bin/ls.
drw-rw-rw- 1 user group 0 May 22 10:18 .
drw-rw-rw- 1 user group 0 May 22 10:18 ..
drw-rw-rw- 1 user group 0 May 4 17:39 comm
-rw-rw-rw- 1 user group 1563 May 22 10:09 test
-rw-rw-rw- 1 user group 1471488 May 3 16:10 unpv12e.tar
#
226 Transfer complete.

ftp> bye
221 Goodbye!
[root@localhost ~]#

```

以上是在 ftp 程序中删除文件的一些指令，以及执行过程中的一些输出信息。



ftpd

**4.4****ftpcount 指令：显示 ftp 登录用户数****◎语法**    **ftpcount****◎功能说明** 显示登录本地 ftp 的用户数、支持的用户上限。**◎执行范例**

- 范例 380 ○ 显示当前 ftp 的用户数。

```
[root@localhost ~]#ftpcount //显示 ftp 用户数
```

本例使用指令 **ftpcount** 显示当前登录的 ftp 用户数。

**◎相关指令**

**ftp**

**4.5****ftpshut 指令：定时关闭 ftp 服务器****◎语法**    **ftpshut[选择性参数][时间][字符串]****◎功能说明** **ftpshut** 指令用来在指定的时间关闭 ftp 服务器，并可以设置关闭前的字符串。**◎参数说明**

选择性参数的具体说明如下：

| 参    数 | 功    能  |
|--------|---------|
| -d<时间> | 关闭连线的时间 |
| -l<时间> | 停止登录的时间 |

**◎执行范例**

- 范例 381 ○ 关闭 ftp 服务器。

```
[root@localhost root]# ftpshut -d 10 -l 8 0000 "byebye"
//定时关闭 ftp
```

本例使用指令 “**ftpshut -d 10 -l 8 0000 "byebye"**” 在指定时间关闭 ftp。

**◎相关指令**

**ftp**



## 4.6 ftpwho 指令：显示 ftp 登录用户信息

**语法**      **ftpwho**

**功能说明**      **ftpwho** 指令用来显示 ftp 登录的用户信息。

**执行范例**

◎ 范例 382 ◎ 显示登录用户。

```
[root@localhost ~]# ftpwho //显示登录用户
```

本例使用指令 “[root@localhost ~]# ftpwho” 显示 ftp 登录的用户。

**相关指令**

**ftp**

## 4.7 ncftp 指令：传输文件

**语法**      **ncftp[必要参数][选择性参数][主机]**

**功能说明**      ncftp 指令为 Internet 标准文件传输协议提供了一个强大的可扩展的接口。它将取代 ftp 指令的位置。

**参选说明**

必要参数的具体说明如下：

| 参 数 | 功 能                               |
|-----|-----------------------------------|
| -F  | 输出 “\$HOME/.ncftp/firewall” 到标准输出 |

选择性参数的具体说明如下：

| 参 数     | 功 能   |
|---------|-------|
| -u<用户名> | 指定用户名 |
| -p<密码>  | 指定密码  |
| -P<端口号> | 指定端口号 |
| -j<账号>  | 指定账号  |

续表

| 参数 | 功能   |
|----|------|
| -h | 帮助信息 |
| -v | 版本信息 |

**【进入 ncftp 后的操作】**

- !: 执行本地机器的指令。
- ascii: 传输模式为 ascii。
- binary: 传输模式为二进制模式。
- bye: 结束 ftp。
- cd: 切换到远端 ftp 服务器上的目录。
- cdup: 上一层目录。
- close: 在不结束 ftp 进程的情况下, 关闭与 ftp 服务器的连接。
- delete: 删除远端 ftp 服务器上的文件。
- get: 下载。
- hash: 显示 “#”。
- lcd: 更改本地目录。
- ls: 列出远端 ftp 服务器上的文件列表。
- mdelete: 删除文件, 模糊匹配。
- mget: 下载文件, 模糊匹配。
- mkdir: 在远端 ftp 服务器上建立文件夹。
- mput: 上传文件, 模糊匹配。
- newer: 下载时检测是不是新文件。
- open: 打开连线。
- prompt: 关闭交互模式。
- put: 上传。
- pwd: 显示当前目录。
- quit: 退出。

**◆ 执行范例**

- ◎ 范例 383 ◎ 显示 ncftp 防火墙参数。



## 第4章 文件传输

```
[root@localhost ~]# ncftp -F
NcFTP firewall preferences
#

#
If you need to use a proxy for FTP, you can configure it below.
If you do not need one, leave the ``firewall-type'' variable set
to 0. Any line that does not begin with the ``#'' character is
considered a configuration command line.
#
NOTE: NcFTP does NOT support HTTP proxies that do FTP, such as "squid"
or Netscape Proxy Server. Why? Because you have to communicate
with
them using HTTP, and this is a FTP only program.
#

#
Types of firewalls:

#
type 1: Connect to firewall host, but send "USER user@real.host.name"
//此处省略了部分结果
#
NOTE: This file was created for you on Fri May 25 09:19:16 2007
by NcFTP 3.2.0. Removing this file will cause the next run of NcFTP
to generate a new one, possibly with more configurable options.
#
ALSO: A /usr/local/etc/ncftp.firewall file, if present, is processed
before this file,
and a /usr/local/etc/ncftp.firewall.fixed file, if present, is
processed after.
```

[root@localhost ~]#

本例使用指令“ncftp -F”显示当前主机的防火墙的设置，输出结果如上所示。

### ○ 范例 384 ○ 登录 ftp 服务器并下载文件。

```
[root@localhost ~]# ncftp -u other -p 02211025 192.168.88.7 //登录
NcFTP 3.2.0 (Aug 05, 2006) by Mike Gleason (http://www.NcFTP.com/contact/).
Connecting to 192.168.88.7...
```

## 4.7 ncftp 指令：传输文件

```

Serv-U FTP Server v5.1 for WinSock ready...
Logging in...
User logged in, proceed.
Logged in to 192.168.88.7.
ncftp / > ls
111 comm/ test unpv12e.tar
ncftp / > lcd /home/kk //本地改变目录
ncftp / > !pwd //显示当前路径
/home/kk
ncftp / > get 111 //下载文件
111: 8.79 kB 218.68 kB/s
ncftp / > mget test
test: 1.53 kB 37.71 kB/s
ncftp / > ls
111 comm/ test unpv12e.tar
ncftp / > mget un*
unpv12e.tar: //下载文件
1.40 MB 944.76 kB/s
ncftp / > bye

```

You have not saved a bookmark for this site.

Would you like to save a bookmark to:

```

ftp://other:PASSWORD@192.168.88.7
Save? (yes/no) n //是否保存标签
Not saved. (If you don't want to be asked this, "set confirm-close no")

```

Thank you for using NcFTP Client.

If you find it useful, please consider making a donation!

<http://www.ncftp.com/ncftp/donate.html>

[root@localhost ~]#

本例首先使用指令“ncftp -u other -p 02211025 192.168.88.7”登录到 ftp 服务器 192.168.88.7 上，用户名和密码通过“-u”和“-p”参数指定；然后使用指令 ls 显示服务器目录下的文件列表；然后使用指令 lcd 改变本地目录；然后使用指令 pwd 显示本地路径；然后使用指令 get 下载指定文件；



最后使用指令 mget 以模糊的方式下载文件，支持通配符。

◎ 范例 385 ◎ 删除文件和上传文件。

```
[root@localhost ~]# ncftp -u other -p 02211025 192.168.88.7
//登录服务器
NcFTP 3.2.0 (Aug 05, 2006) by Mike Gleason (http://www.NcFTP.com/contact/).

Connecting to 192.168.88.7...
Serv-U FTP Server v5.1 for WinSock ready...
Logging in...
User logged in, proceed.
Logged in to 192.168.88.7.

ncftp / > ls //显示文件列表
111 comm/ test unpv12e.tar
ncftp / > delete 111 //删除文件
ncftp / > ls //显示文件列表
comm/ test unpv12e.tar
ncftp / > lcd /home/kk //改变本地目录
ncftp / > put 111 //上传文件
111: 8.79 kB 216.53 kB/s
ncftp / > ls //显示文件列表
111 comm/ test unpv12e.tar
ncftp / > bye //退出

You have not saved a bookmark for this site.
Would you like to save a bookmark to:
 ftp://other:PASSWORD@192.168.88.7
Save? (yes/no) n //不保存标签
Not saved. (If you don't want to be asked this, "set confirm-close no")

[root@localhost ~]#
```

本例首先使用指令“ncftp -u other -p 02211025 192.168.88.7”登录到 ftp 服务器 192.168.88.7，用户名和密码通过“-u”和“-p”参数指定；然后使用指令 ls 显示当前文件夹下的文件列表；然后使用指令 put 上传文件；最后使用指令 bye 退出 ncftp 程序。

◎ 范例 386 ◎ 登录时必须输入用户名和密码，除非服务器允许匿名

## 4.8 ncftpget 指令：下载文件

登录。

```
[root@localhost ~]# ncftp 192.168.88.7 //登录服务器
NcFTP 3.2.0 (Aug 05, 2006) by Mike Gleason (http://www.NcFTP.
com/contact/).

Connecting to 192.168.88.7...
Serv-U FTP Server v5.1 for WinSock ready...
Sorry, no ANONYMOUS access allowed.
Redialing (try 1)...
Serv-U FTP Server v5.1 for WinSock ready...
Sorry, no ANONYMOUS access allowed.
Redialing (try 2)...
Serv-U FTP Server v5.1 for WinSock ready...
Sorry, no ANONYMOUS access allowed.
Sleeping 20 seconds...
[root@localhost ~]#
```

登录时必须输入用户名和密码，除非 ftp 服务器允许匿名登录，否则就会出现上面的结果。



ftp

## 4.8 ncftpget 指令：下载文件

ncftpget[必要参数][选择性参数][远端主机名][本地存储路径][远端文件]

ncftpget 指令用来直接获取远端 ftp 服务器上文件，是 ncftp 的公用程序，可以更方便地下载文件。



必要参数的具体说明如下：

| 参    数 | 功    能                          |
|--------|---------------------------------|
| -F     | 输出“\$HOME/.ncftp/firewall”到标准输出 |

选择性参数的具体说明如下：



| 参数      | 功能    |
|---------|-------|
| -u<用户名> | 指定用户名 |
| -p<密码>  | 指定密码  |
| -P<端口号> | 指定端口号 |
| -j<账号>  | 指定账号  |
| -h      | 帮助信息  |
| -v      | 版本信息  |

### ◎ 指令说明

◎ 范例 387 ◎ 下载文件。

```
[root@localhost ~]# ncftpget -u other -p 02211025 192.168.88.7
/home/kk ./111 //下载文件
/home/kk/111: 8.79 kB 219.55 kB/s
[root@localhost ~]#
```

本例使用指令 “ncftpget -u other -p 02211025 192.168.88.7 /home/kk./111” 下载远端主机 192.168.88.7 上的文件 111 到本地指定的目录/home/kk。

### ◎ 相关指令

ncftp、ncftppu

## 4.9 ncftpls 指令：显示文件目录

◎ 基础 ncftpls[必要参数][选择性参数][远端主机 URL]

◎ 功能说明 ncftpls 指令用来显示远端主机的文件列表。

### ◎ 参数说明

必要参数的具体说明如下：

| 参数 | 功能                                |
|----|-----------------------------------|
| -F | 输出 "\$HOME/.ncftp/firewall" 到标准输出 |

选择性参数的具体说明如下：

## 4.10 ncftpput 指令：上传文件

| 参数      | 功能    |
|---------|-------|
| -u<用户名> | 指定用户名 |
| -p<密码>  | 指定密码  |
| -P<端口号> | 指定端口号 |
| -j<账号>  | 指定账号  |
| -h      | 帮助信息  |
| -v      | 版本信息  |

## 技巧范例

- ◎ 范例 388 ◎ 显示远端 ftp 服务器的文件列表。

```
[root@localhost ~]# ncftpls -u other -p 02211025 ftp://192.168.88.7
 //显示文件列表
111 comm/ test unpv12e.tar
[root@localhost ~]#
```

本例使用指令 “ncftpls -u other -p 02211025 ftp://192.168.88.7” 显示远端 ftp 服务器上文件列表信息。

## 相关指令

ncftp、ncftpget、ncftpput

## 4.10 ncftpput 指令：上传文件

技巧 ncftpput[必要参数][选择性参数][远端目录][本地文件]

功能说明 ncftpput 指令用来直接上传文件到远端 ftp 服务器，是 ncftp 的公用程序，可以更方便地上传文件。

## 功能说明

必要参数的具体说明如下：

| 参数 | 功能                                |
|----|-----------------------------------|
| -F | 输出 "\$HOME/.ncftp/firewall" 到标准输出 |

选择性参数的具体说明如下：



| 参数      | 功能    |
|---------|-------|
| -u<用户名> | 指定用户名 |
| -p<密码>  | 指定密码  |
| -P<端口号> | 指定端口号 |
| -j<账号>  | 指定账号  |
| -h      | 帮助信息  |
| -v      | 版本信息  |

### ☆ 功能说明

◎ 范例 389 ◎ 上传文件。

```
[root@localhost ~]#ncftpput -u other -p 02211025 192.168.88.7 ./ /home/
kk/111 //上传文件
/home/kk/111: 8.79 kB 218.36 kB/s
```

[root@localhost ~]#

本例使用指令“ncftpput -u other -p 02211025 192.168.88.7 ./ /home/kk/111”上传本地文件/home/kk/111 到远端主机 192.168.88.7 上的当前目录。

### ☆ 相关指令

ncftp、ncftpget

## 4.11 tftp 指令：传输文件

### ☆ 基础

tftp[必要参数][选择性参数][主机]

### ☆ 功能说明

tftp 指令是简单的文字模式的 FTP 指令。

### ☆ 参数说明

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

## 4.11 tftp 指令：传输文件

| 参数     | 功能     |
|--------|--------|
| -c<指令> | 远程执行指令 |
| -m<模式> | 选择传输模式 |
| -V     | 版本信息   |

 实际范例

## ◎ 范例 390 ◎ 开启 tftp 程序。

```
[root@localhost ~]# tftp //开启 tftp 程序
(to) 192.168.88.2 //连接至 192.168.88.2
tftp> ? //显示帮助信息
tftp-hpa 0.40
Commands may be abbreviated. Commands are:
connect connect to remote tftp
mode set file transfer mode
put send file
get receive file
quit exit tftp
verbose toggle verbose mode
trace toggle packet tracing
status show current status
binary set mode to octet
ascii set mode to netascii
rexmt set per-packet transmission timeout
timeout set total retransmission timeout
? print help information
help print help information
tftp> status //当前状态
Connected to 192.168.88.2.
Mode: octet Verbose: off Tracing: off
Rexmt-interval: 5 seconds, Max-timeout: 25 seconds
tftp> binary //二进制模式
tftp> quit //退出
[root@localhost ~]#
```

本例使用指令 tftp 开启 tftp 程序，进入 tftp 之后的操作说明如下所示：

- connect：连接到远端 tftp 服务器。
- mode：文件传输模式。



- put: 上传文件。
- get: 下载文件。
- quit: 退出。
- verbose: 显示详细的处理信息。
- trace: 显示包路径。
- status: 显示当前状态信息。
- binary: 二进制传输模式。
- ascii: ascii 传递模式。
- rexmt: 设置包传输的超时时间。
- timeout: 设置重传的超时时间。
- ?: 帮助信息。
- help: 帮助信息。

 相关指令

ftp

## 4.12 uucico 指令：uucp 文件传输

 语法 uucico[必要参数][选择性参数]

 功能说明 uucico 指令用来处理 uucp 的文件传输。

 参数说明

必要参数的具体说明如下：

| 参数  | 功能         |
|-----|------------|
| -c  | 运行时不显示处理信息 |
| -D  | 始终保持与终端的连接 |
| -e  | 附属模式       |
| -f  | 强制执行       |
| -t0 | 附属方式       |
| -t1 | 主动方式       |

## 4.13 uucp 指令：在 UNIX 系统之间传送文件

选择性参数的具体说明如下：

| 参 数      | 功 能      |
|----------|----------|
| -i<类型>   | 指定连接端口类型 |
| -s<主机>   | 指定主机     |
| -u<用户>   | 指定用户     |
| -version | 版本信息     |
| -help    | 帮助信息     |

◆ ◆ ◆ 实际范例

◎ 范例 391 ◎ 启动 uucico。

```
[root@localhost ~]# uucico -rl //启动uucico
[root@localhost ~]#
```

本例使用指令“uucico -rl”以主动模式启动 uucico。

◆ ◆ ◆ 相关指令

uucp

## 4.13 uucp 指令：在 UNIX 系统之间传送文件

◆ ◆ ◆ 语法 uucp[必要参数][选择性参数][目的]

◆ ◆ ◆ 功能说明 uucp 指令用来在 UNIX 系统之间传送文件。

◆ ◆ ◆ 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能           |
|-----|---------------|
| -c  | 先将文件复制到缓冲区    |
| -C  | 不将文件复制到缓冲区    |
| -j  | 显示任务编号        |
| -m  | 启动 email 报告信息 |
| -d  | 自动建立目录        |
| -R  | 递归包含目录的子目录    |



选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### ❖ 执行范例

◎ 范例 392 ◎ 传送文件。

```
[root@localhost ~]# uucp -d -R /home/ kk!~/111 //传送文件
[root@localhost ~]#
```

本例使用指令“uucp -d -R /home/ kk!~/111”将本机的/home 目录下的文件传送到主机 kk 下 UUCP 公用目录下的 111 目录下。

#### ❖ 相关指令

uucico

## 4.14 uupick 指令：处理文件

#### ❖ 语法

uupick[必要参数][选择性参数]

#### ❖ 功能说明

uupick 指令用来处理 uucp 传送来的文件。

#### ❖ 参数说明

必要参数的具体说明如下：

| 参数     | 功能   |
|--------|------|
| -debug | 调试模式 |

选择性参数的具体说明如下：

| 参数         | 功能     |
|------------|--------|
| -debug<等级> | 调试等级   |
| -<文件>      | 指定配置文件 |
| -version   | 版本信息   |
| -help      | 帮助信息   |

## 4.15 uuto 指令：文件传送到远端主机

**执行范例**

- ◎ 范例 393 ◎ 处理文件。

```
[root@localhost ~]# uupick //处理文件
```

本例使用指令 uupick 处理 uucp 传送的文件。

**相关指令**

uucp

## 4.15 uuto 指令：文件传送到远端主机

**简介** uucp[必要参数][选择性参数][文件][主机]

**功能说明** uuto 指令是一个 script 指令，它传送文件到远端的 uucp 主机。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能            |
|----|---------------|
| -c | 先将文件复制到缓冲区    |
| -C | 不将文件复制到缓冲区    |
| -j | 显示任务编号        |
| -m | 启动 email 报告信息 |
| -d | 自动建立目录        |
| -R | 递归包含目录的子目录    |

选择性参数的具体说明如下：

| 参数        | 功能   |
|-----------|------|
| -help     | 帮助信息 |
| --version | 版本信息 |

**执行范例**

- ◎ 范例 394 ◎ 传送文件。



## 第4章 文件传输

```
[root@localhost ~]# uuto /home/kk/111 kk!111 //传送文件
[root@localhost ~]#
```

本例使用指令“uuto /home/kk/111 kk!111”传送本地文件/home/kk/111到远端主机kk下的111目录。



uucp



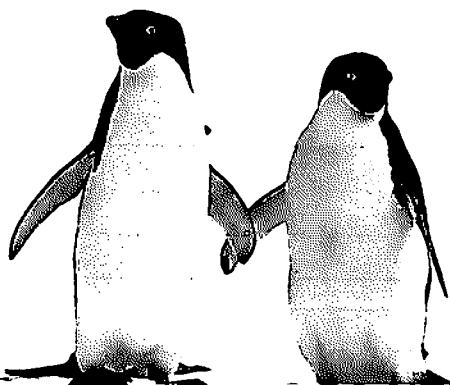
顶点网

TopSage.cn

## 第5章

### 文件管理

文件管理是管理和设置 Linux 系统的基础，这些指令主要用来于管理文件系统。实现的主要功能包括设置文件属性，文件的复制、查找、链接、创建、删除等。这些功能都是用户在使用 Linux 系统中经常使用的。





## 5.1 aspell 指令：检查文件的错误

 **帮助** aspell[必要参数][选择性参数][文件]

 **功能说明** aspell 指令用于检查文件的拼写。它有许多参数，是一个超级检查工具。

 **参数说明**

必要参数的具体说明如下：

| 参 数          | 功 能                 |
|--------------|---------------------|
| -c           | 检查文件                |
| -a           | 兼容模式                |
| -l           | 从标准输入产生一个拼写错误的单词的列表 |
| config <关键词> | 打印当前某一参数的值          |
| filter       | 过滤器，过滤掉某些单词         |

选择性参数的具体说明如下：

| 参 数                  | 功 能          |
|----------------------|--------------|
| -? [help]            | 帮助信息         |
| -v                   | 版本信息         |
| -conf=<字符串>          | 主配置文件        |
| -conf-dir=<字符串>      | 本地的主配置文件     |
| -data-dir=<字符串>      | 本地的语言数据文件    |
| -dict-dir=<字符串>      | 本地的主单词列表     |
| -addrem-filter=<字符串> | 增加或者移除一个过滤器  |
| -mode=<str>          | 过滤器模式        |
| -e                   | 电子邮件模式       |
| -H                   | HTML/SGML 模式 |
| -t                   | 文档模式         |

## 5.1 aspell 指令：检查文件的错误

## 88 执行范例

## ◎ 范例 395 ◎ 检查文件的错误。

```
[root@localhost root]# aspell -c a.c //检查文件a.c
jdlkfj1ksdjflksd
df1ksj1kdfjflkds
fj1ksdj1kfdj1kdsj
jdlkfjsdlkjflkdsjlfhdkljhfg1
jd1fjlsdkjflkjds
njdf1kdsj1k
jldkfjdlk
fkflj1ksd
i) Ignore I) Ignore all
r) Replace R) Replace all
a) Add x) Exit
```

运行“aspell -c a.c”指令之后，下面显示的内容为文件 a.c 的内容，并给出了 Ignore、Ignore all、Replace、Replace all、Add、Exit 等修改意见。

## ◎ 范例 396 ◎ 将文件中拼错的单词输出到标准输出。

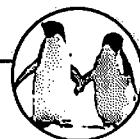
```
[root@localhost root]# aspell -l <a.c //将a.c中拼错的单词输出到标准输出
kdjslkfjsdlkjflkjflk
jdlkfj1ksdjflksd
df1ksj1kdfjflkds
fj1ksdj1kfdj1kdsj
jdlkfjsdlkjflkdsjlfhdkljhfg1
jd1fjlsdkjflkjds
njdf1kdsj1k
jldkfjdlk
fkflj1ksd
jdlkfj1kjd1ksjflksdjflkj
[root@localhost root]#
```

## ◎ 范例 397 ◎ 输出检错词典。

```
[root@localhost root]# aspell dump master //输出检错词典
```

## 88 相关指令

spell



## 5.2

## attr 指令：XFS文件系统对象的扩展属性



**attr [必要参数] [属性名称] [选择性参数] [路径名]**



**attr** 指令用来设置 XFS 文件系统的扩展属性。



必要参数的具体说明如下：

| 参 数 | 功 能                                                             |
|-----|-----------------------------------------------------------------|
| -s  | 设置属性值                                                           |
| -g  | 获得属性值                                                           |
| -r  | 删除相关属性                                                          |
| -L  | 如果指定“-L”参数并且对象是一个符号链接，那么操作的是引用符号链接的对象的属性。如果不指定这个参数，那么操作的是符号链接本身 |
| -R  | 如果“-R”参数指定并且处理过程拥有适当的特权，操作将发生在 root 用户的属性空间，而不是一般用户的属性空间        |
| -q  | 如果“-q”参数指定，属性将保持安静，将不打印状态信息                                     |

选择性参数的具体说明如下：

| 参 数     | 功 能   |
|---------|-------|
| -V<属性值> | 设置属性值 |



◎ 范例 398 ◎ 设置文件的属性。

```
[root@localhost ~]# attr -s attrname pathname //设置文件的属性
```

◎ 范例 399 ◎ 获得文件的属性。

```
[root@localhost ~]# attr -g attrname pathname //获得文件的属性
```

◎ 范例 400 ◎ 移除文件的属性。

```
[root@localhost ~]# attr -r attrname pathname //移除文件的属性
```

当指定“-s”参数时，将从标准输入中读取一个属性值，当设定“-g”参数时将读取一个属性值到标准输出。

**相关指令**

getfattr、setfattr、xfsdump

**5.3****basename 指令：显示文件或者目录的基本名称****语法**

basename[选择性参数][路径名]

**功能说明**

basename 指令用于打印目录或者文件的基本名称，它将剥除目录或者文件的路径的前缀以及文件的后缀信息。

**参数说明**

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

**执行范例****◎ 范例 401 ◎ 显示基本的文件或者目录名称。**

```
[root@localhost /]# basename /home/kk //显示目录的基本名称
kk
[root@localhost /]#
```

basename 显示指定完整路径的最后部分，也就是目录或者文件的真正的名称。例如上例中完整的路径信息是/home/kk，执行 basename 指令之后，得到目录的基本名称 kk。

**相关指令**

uname

**5.4****chattr 指令：改变文件的属性****语法**

chattr[必要参数][选择性参数][+/-&lt;属性&gt;][文件或者

目录]



**功能说明** chattr 指令改变 Linux 第二个扩展文件系统的属性。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能                   |
|----|----------------------|
| -R | 处理指定目录以及目录的子目录下的所有文件 |
| -V | 显示指令运行时的所有处理过程       |

选择性参数的具体说明如下：

| 参数       | 功能                 |
|----------|--------------------|
| -v<版本编号> | 设置文件或者目录的版本        |
| +<属性>    | 使文件或目录具有该项属性       |
| -<属性>    | 删除文件或者目录具有的该项属性    |
| =<属性>    | 设置文件或者目录的该项属性为相应的值 |

### 【属性设置】

- a: 文件或者目录的附加用途。
- b: 不更新文件或者目录的最后存取时间。
- c: 压缩文件或者目录。
- d: 不对文件或者目录进行倾倒操作。
- i: 不改动文件或者目录的属性。
- s: 安全删除文件或者目录。
- u: 进行意外删除的保护。

**执行范例**

#### ◎ 范例 402 ◎ 设置文件的属性。

```
[root@localhost root]# lsattr abc //显示文件 abc 的属性
-----abc
[root@localhost root]# chattr +a abc //使文件具有附加属性
[root@localhost root]# lsattr abc //显示文件 abc 的属性
-----a-----abc
[root@localhost root]#
```

第一次使用指令“lsattr abc”时显示文件 abc 的属性，发现文件 abc 的

## 5.5 chgrp 指令：改变文件或者目录所属的群组

各项属性为空白，然后使用指令“chattr +a abc”，使得文件 abc 具有附加的属性。现在再使用指令“lsattr abc”显示文件 abc 的属性时会发现文件具有的附加的属性。

### ◎ 范例 403 ◎ 删除文件具有的属性。

```
[root@localhost root]# chattr -a abc //删除文件abc的附加属性
[root@localhost root]# lsattr abc //显示文件abc的属性
----- abc
[root@localhost root]#
```

文件 abc 具有了附加的属性，使用指令“chattr -a abc”删除文件 abc 的附加属性。

### ◎ 范例 404 ◎ 设置文件的属性值。

```
[root@localhost root]# chattr =a abc //设置文件的附加属性
```

本例使用指令“chattr =a abc”设置文件的属性值。



lsattr

## 5.5 chgrp 指令：改变文件或者目录所属的群组

**语法** chgrp[必要参数][选择性参数][群组][文件或者目录]

**功能说明** chgrp 指令用来改变文件或者目录所属的群组。



必要参数的具体说明如下：

| 参 数              | 功 能                    |
|------------------|------------------------|
| -c               | 当发生改变时输出调试信息           |
| -f               | 不显示错误信息                |
| -R               | 处理指定目录以及其子目录下的所有文件     |
| -v               | 运行时显示详细的处理信息           |
| --dereference    | 作用于每个符号链接的指向，而不是符号链接本身 |
| --no-dereference | 作用于符号链接本身              |



选择性参数的具体说明如下：

| 参数                  | 功能                           |
|---------------------|------------------------------|
| -help               | 显示帮助信息                       |
| -reference=<文件或者目录> | 依据指定文件或者目录所属群组，设置文件或者目录的群组属性 |
| -version            | 显示版本信息                       |

### 执行范例

#### ◎ 范例 405 ◎ 改变文件的群组属性。

```
[root@localhost kk]# chgrp -v kk king //改变文件king的群组为kk群组
'king'的所属组已变为 kk
[root@localhost kk]#
```

使用“chgrp -v kk king”指令改变文件 king 的群组为 kk 群组。

#### ◎ 范例 406 ◎ 根据指定文件改变文件的群组属性。

```
[root@localhost kk]# ll //显示当前文件夹下的文件详细信息
总用量 8
-rw-r--r-- 1 root root 12 4月 23 15:30 123.c
-rw-rw-r-- 1 kk kk 51 5月 9 10:08 king
[root@localhost kk]# chgrp --reference=123.c king //改变文件king的群组
信息
```

```
[root@localhost kk]# ll //显示当前文件夹下的文件详细信息
总用量 8
-rw-r--r-- 1 root root 12 4月 23 15:30 123.c
-rw-rw-r-- 1 kk root 51 5月 9 10:08 king
[root@localhost kk]#
```

首先使用指令 ll 显示当前文件夹下的文件的详细信息，发现文件 king 所属的群组为 kk；然后使用指令“chgrp --reference=123.c king”改变文件 king 的群组属性，使得文件 king 的群组属性和参考文件 123.c 的群组属性相同；然后再使用指令 ll，列出文件夹下的所有文件的详细信息，发现文件 king 的群组变为 root 群组。

### 相关指令

chmod

## 5.6 chmod 指令：设置文件或者目录的权限

## 5.6 chmod 指令：设置文件或者目录的权限

**语法**      `chmod[必要参数][选择性参数]<权限范围>+/-<权限设置>][文件或者目录]`

**功能说明**    `chmod` 指令依据权限设置改变给定文件的权限。权限既可以是一个符号，也可以是一个代表性的数字。符号的模式可以是 `[ugoa...][[+=-][rwxXstugo...]...][...]`，可以给定多个符号，它们之间用逗号隔开。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能                 |
|----|--------------------|
| -c | 当发生改变时，报告处理信息      |
| -f | 错误信息不输出            |
| -R | 处理指定目录以及其子目录下的所有文件 |
| -v | 运行时显示详细处理信息        |

选择性参数的具体说明如下：

| 参数                 | 功能                     |
|--------------------|------------------------|
| -reference<目录或者文件> | 设置成具有与指定的目录或者文件具有相同的权限 |
| -version           | 显示版本信息                 |
| <权限范围><权限设置>       | 使权限范围内的目录或者文件具有指定的权限   |
| <权限范围><权限设置>       | 删除权限范围内的目录或者文件的指定权限    |
| <权限范围><权限设置>       | 设置权限范围内的目录或者文件的权限为指定的值 |

**【权限范围】**

- u：目录或者文件的当前的用户。
- g：目录或者文件的当前的群组。
- o：除了目录或者文件的当前用户或者群组之外的用户或者群组。
- a：所有的用户及群组。

**【权限代号】**

- r: 读取目录或者文件的权限, 用数字 4 表示。
- w: 写入目录或者文件的权限, 用数字 2 表示。
- x: 执行目录或者文件的权限, 用数字 1 表示。
- -: 删减目录或者文件的所有权限, 用数字 0 表示。
- s: 特殊权限, 改变目录或者文件的权限。

**○ 实例与范例****○ 范例 407 ○ 设置文件具有所有的权限。**

```
[root@localhost root]# ll //显示当前文件夹下的所有文件的信息
```

**总用量 264**

```
-rw-r--r-- 1 root root 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 83833 4月 23 15:59 magick.miff
```

```
[root@localhost root]# chmod 777 1.log //设置文件 1.log 的权限
```

```
[root@localhost root]# ll //显示当前文件夹下的所有文件的信息
```

**总用量 264**

```
-rwxrwxrwx 1 root root 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
```

```
[root@localhost root]#
```

本例使用指令 “chmod 777 1.log” 设置文件 1.log 具有所有的权限。

**○ 范例 408 ○ 设置文件的属性。**

```
[root@localhost root]# chmod ug+rw 1.log //设置文件 1.log 的属性
```

本例使用指令 “chmod ug+rw 1.log” 设置文件的用户和同群组内的用户具有指定文件 1.log 的可读可写权限。

**○ 范例 409 ○ 删减文件的权限。**

## 5.7 chown 指令：改变文件的拥有者或者群组

[root@localhost root]# chmod a-x 1.log //设置文件 1.log 的属性

本例使用指令“chmod a-x 1.log”删除文件 1.log 的可执行权限。

◎ 范例 410 ◎ 使用“=”设置权限。

[root@localhost root]# chmod u=r 1.log //设置文件 1.log 的属性

本例使用指令“chmod u=r 1.log”撤销原来的所有权限，然后使拥有者具有可读权限。



chown

## 5.7

## chown 指令：改变文件的拥有者或者群组

**语法** chown[必要参数][选择性参数][用户或者群组]

**功能说明** chown 指令改变每一个给定文件的用户和/或群组的拥有权限。如果只给定一个用户名（或者用户 ID 号），那么用户将成为指定文件的拥有者，文件的群组并不改变。如果同时指定用户名和群组名，那么指定文件的用户和群组都要改变。

如果用户名后面跟随一个冒号或者点号，而不出现群组名，那么指定文件的群组也要改变，变为指定用户所在的群组。如果冒号或者点号和群组名称指定，但是用户名忽略，那么只有文件的群组属性改变。在这种情况下，chown 指令的执行效果等同于指令 chgrp 的执行效果。



必要参数的具体说明如下：

| 参 数           | 功 能                  |
|---------------|----------------------|
| -c            | 显示更改部分的信息            |
| -f            | 忽略错误信息               |
| -h            | 修改符号链接               |
| -R            | 处理指定目录以及其子目录下的所有文件   |
| -v            | 运行时显示详细的处理信息         |
| --dereference | 作用于符号链接的指向，而不是符号链接本身 |



选择性参数的具体说明如下：

| 参 数                  | 功 能                                                     |
|----------------------|---------------------------------------------------------|
| -help                | 帮助信息                                                    |
| --reference=<目录或者文件> | 把指定的目录或者文件作为参考目录或者文件，把要操作的目录或者文件设置成跟参考目录或者文件相同的所属用户或者群组 |
| --version            | 显示版本信息                                                  |
| --from=<当前用户：当前群组>   | 只有当前用户和群组跟指定的用户和群组相同时才进行改变                              |

### ◎ 实际范例

○ 范例 411 ○ 改变文件的用户和群组。

```
[root@localhost root]# ll //列出当前的文件信息
总用量 264
-rwxrwxrwx 1 root root 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 2019:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
[root@localhost root]# chown kk:kk 1.log //改变文件的用户和群组
[root@localhost root]# ll //显示文件的信息
总用量 264
-rwxrwxrwx 1 kk kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
[root@localhost root]#
```

本例使用指令“chown kk:kk 1.log”改变文件和用户的群组信息，使得用户的群组为 kk 群组，用户为 kk 用户，而不是原来的用户和群组 root。

○ 范例 412 ○ 如果用户名后面跟随一个冒号或者点号，而不出现群

**5.8 cksum 指令：文件的 CRC 校验**

组名，那么指定文件的群组也要改变，变为指定用户名所在的群组。

```
[root@localhost root]# chown root: 1.log //改变文件的用户和群组
[root@localhost root]# ll //显示文件的详细信息
总用量 264
-rwxrwxrwx 1 root root 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
[root@localhost root]#
```

○ 范例 413 ○ 如果冒号或者点号和群组名称指定，但是用户名忽略，那么只有文件的群组属性改变。

```
[root@localhost root]# chown :kk 1.log //改变文件的群组信息
[root@localhost root]# ll //显示文件的详细信息
总用量 264
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
[root@localhost root]#
```

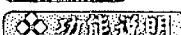


**chgrp**

## 5.8 cksum 指令：文件的 CRC 校验



**cksum[选择性参数][文件]**



cksum 指令将打印出文件的校验和。



大家网

### 功能说明

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### 执行范例

◎ 范例 414 ◎ 对文件进行校验。

```
[root@localhost root]# cksum 1.log //计算文件的校验和
1259339231 2398 1.log
[root@localhost root]#
```

### 相关指令

ckeygen

## 5.9 cmp 指令：比较文件差异

### 语法

cmp[必要参数][选择性参数][文件 1][文件 2]

◎ 功能说明 ◎ cmp 指令比较任意类型的两个文件，并且把结果输出到标准输出。默认情况下，如果文件是相同的，那么它们将不进行任何输出。如果两个文件不同，那么第一次出现不同的字符和行数将报告给用户。字符和行数将从 1 开始计数。

### 功能说明

必要参数的具体说明如下：

| 参 数 | 功 能       |
|-----|-----------|
| -c  | 显示不同的信息   |
| -l  | 列出所有的不同信息 |
| -s  | 错误信息不显示   |

选择性参数的具体说明如下：

## 5.9 cmp 指令：比较文件差异

| 参数      | 功能     |
|---------|--------|
| -i<字符数> | 指定字符数目 |
| -v      | 版本信息   |
| -help   | 帮助信息   |

## 【函数的返回信息】

- 0: 文件是同样的。
- 1: 文件是不同的。
- >1: 发生错误。

 执行范例

## ◎ 范例 415 ◎ 比较两个文件。

```
[root@localhost root]# cmp 1.log abc //比较两个文件 1.log 和 abc
1.log abc differ: byte 1, line 1
[root@localhost root]#
```

根据输出结果可以看到，文件 1.log 和文件 abc 不同，并且文件 1.log 和文件 abc 从第一行的第一个字节就不同了。

## ◎ 范例 416 ◎ 显示两个文件的所有不同的信息。

```
[root@localhost root]# cmp -l 1.log abc //显示文件 1.log 和 abc 的所有不同的信息
```

```
1 141 151
2 142 40
3 157 141
4 162 155
5 164 40
6 12 141
7 141 40
8 143 163
9 143 164
10 145 165
11 160 144
12 164 145
13 55 156
14 154 164
15 151 12
```



```

16 156 151
17 145 40
18 12 141
19 141 155
20 154 40
21 151 146
22 141 162
23 163 157
24 55 155
25 145 40
26 170 142
27 160 152
28 141 12
cmp: EOF on abc
[root@localhost root]#

```

本例使用指令“`cmp -l 1.log abc`”将显示文件 1.log 和文件 abc 的所有不同的信息。

#### 相关指令

`diff`、`diff3`

## 5.10 cp 指令：复制

**语法** `cp[必要参数][选择性参数][源目录或者文件][目标文件或者目录]`

**功能说明** 复制源文件到目标文件，或者复制多个源到目标文件夹。

#### **参数说明**

必要参数的具体说明如下：

| 参 数             | 功 能                                   |
|-----------------|---------------------------------------|
| <code>-a</code> | 等同于“ <code>-dpR</code> ”              |
| <code>-b</code> | 若删除或者覆盖目标文件，将对目标文件进行备份，备份的文件以备份的字符串结尾 |
| <code>-d</code> | 复制符号链接                                |

续表

| 参数 | 功能                    |
|----|-----------------------|
| -f | 强制复制                  |
| -i | 交互式模式，覆盖目标文件之前先要进行询问  |
| -l | 建立硬链接，而非复制            |
| -p | 源目录或者文件的属性保留          |
| -P | 源文件或者文件的路径保留          |
| -r | 处理指定目录以及目录的子目录下的所有文件  |
| -R | 处理指定目录以及目录的子目录下的所有文件  |
| -s | 不进行复制，而是建立符号链接        |
| -u | 只在源文件更新时进行复制          |
| -v | 运行时显示详细的处理信息          |
| -x | 只在源和目标的文件系统类型相同时才进行复制 |

选择性参数的具体说明如下：

| 参数                | 功能                             |
|-------------------|--------------------------------|
| -suffix=<备份字尾字符串> | 与“-b”参数同时使用，指定备份的目录或者文件的字尾的字符串 |
| -version=<备份方式>   | 与“-b”参数同时使用，选择备份的方式            |
| -help             | 帮助信息                           |
| -version          | 版本信息                           |

### ◎执行范例

#### ◎ 范例 417 ◎ 复制文件。

```
[root@localhost ~]# cp a.c /home/kk/ //复制文件到目录/home/kk
[root@localhost ~]# cd /home/kk/ //改变目录到/home/kk
[root@localhost kk]# ls
a.c
[root@localhost kk]#
```

本例使用指令“cp a.c /home/kk/”复制文件 a.c 到指定的目录/home/kk，将目录切换到/home/kk 下，可以看到当前的目录下多了文件 a.c。注意：指



定的文件夹必须存在，如此处的文件夹/home/kk 必须存在。

◎ 范例 418 ◎ 复制时，进行备份。

```
[root@localhost ~]# cp -b a.c /home/kk/ //复制文件，将已经存在的文件
进行备份
```

```
[root@localhost ~]# cd /home/kk/
[root@localhost kk]# ls
a.c a.c~
[root@localhost kk]#
```

本例使用指令“cp -b a.c /home/kk/”复制文件 a.c。由于文件 a.c 在 /home/kk/下已经存在，并且复制时指定了参数“-b”，所以原来的文件 a.c 将以备份的形式进行存储，存储的名称为“a.c~”。现在在文件夹下可以看到两个文件，如上所示。

◎ 范例 419 ◎ 复制备份时选择备份文件的字尾的字符串。

```
[root@localhost ~]# cp -bv --suffix=@ a.c /home/kk/ //备份时选择字
尾的字符串
```

```
[root@localhost ~]# cd /home/kk/
[root@localhost kk]# ls
a.c a.c~ a.c@
[root@localhost kk]#
```

本例使用指令“cp -bv --suffix=@ a.c /home/kk/”复制文件并且指定备份文件的字尾的字符串为“@”，这样在文件夹/home/kk/下将出现文件 a.c@，如上所示。

◎ 范例 420 ◎ 复制文件夹。

```
[root@localhost ~]# cp -fr Desktop/ /home/kk/ //复制文件夹到指定路径
[root@localhost ~]# cd /home/kk/
[root@localhost kk]# ls
a.c a.c~ a.c@ Desktop
[root@localhost kk]#
```

本例使用指令“cp -fr Desktop/ /home/kk/”复制文件夹 Desktop/到指定路径/home/kk/，现在看到在/home/kk/下多了一个文件夹 Desktop，如上所示。



mv、ld

## 5.11 csplit 指令：分割文件



**csplit[必要参数][选择性参数][文件][模式]**



使用 csplit 指令后，文件的输出将按模式进行分割，并将结果输出到标准输出。



必要参数的具体说明如下：

| 参数 | 功能      |
|----|---------|
| -k | 保留源文件   |
| -q | 不显示调试信息 |
| -z | 删除空文件   |

选择性参数的具体说明如下：

| 参数        | 功能              |
|-----------|-----------------|
| -b<输出格式>  | 指定输出格式，可改变输出文件名 |
| -f<首个字符串> | 指定输出的字符串首字符串    |
| -n<位数>    | 指定输出文件名的位数      |
| --help    | 帮助信息            |
| --version | 版本信息            |



◎ 范例 421 ◎ 分割文件。

```
[root@localhost temp]# ll //显示文件列表
总用量 28
-rw-r--r-- 1 root root 27032 5月 31 23:31 install.log
[root@localhost temp]# csplit install.log 200 //分割文件
6352
20680
[root@localhost temp]# ll //显示文件列表
总用量 60
```



```
-rw-r--r-- 1 root root 27032 5月 31 23:31 install.log
-rw-r--r-- 1 root root 6352 5月 31 23:32 xx00
-rw-r--r-- 1 root root 20680 5月 31 23:32 xx01
[root@localhost temp]#
```

本例使用指令“`csplit install.log 200`”分割文件 `install.log`, 前 200 行为一个文件, 剩下的为另一个文件。



`split`

## 5.12

### diff 指令：生成差异信息

`diff`[必要参数][选择性参数][文件 1 或者目录 1][文件 2 或者目录 2]

`diff` 指令比较两个指定文件的内容。如果指定的最后两个参数都是目录, 那么 `diff` 指令将比较两目录下的同名文件, 如果同时指定“`-r`”参数, 那么两个目录下的子目录也将进行递归比较。



必要参数的具体说明如下:

| 参 数                 | 功 能                                          |
|---------------------|----------------------------------------------|
| <code>-a</code>     | 把所有文件看成文本形式, 并对它们进行逐行比较, 即使它们看起来不是文本文件       |
| <code>-b</code>     | 忽略空格                                         |
| <code>-B</code>     | 忽略空白行                                        |
| <code>-i</code>     | 忽略大小写                                        |
| <code>-brief</code> | 只报告两个文件是否相同, 不报告详细的不同信息                      |
| <code>-c</code>     | 使用上下文输出格式                                    |
| <code>-d</code>     | 改变算法, 使得 <code>diff</code> 指令可能找到一个更小的文件间的差异 |
| <code>-e</code>     | 产生一个将文件 1 变成文件 2 的“ <code>ed script</code> ” |
| <code>-r</code>     | 比较目录时递归比较子目录                                 |

## 5.12 diff 指令：生成差异信息

续表

| 参数         | 功能                                            |
|------------|-----------------------------------------------|
| -h         | 支持 UNIX 的兼容性                                  |
| -n         | RCS 格式                                        |
| -N         | 在目录比较中，如果一个文件只出现在第一个目录中，那么认为它在另外一个目录中也出现，并且为空 |
| -p         | 显示不同的信息位于哪个 C 函数中                             |
| -P         | 在目录比较中，如果一个文件只出现在第二个目录中，那么认为它也出现在另外一个目录中，并且为空 |
| --paginate | 通过 pr 输出，并且标记页数                               |
| -q         | 不报告差异，只说明两者内容是否有差异                            |
| -s         | 当比较的两个文件相同时报告信息                               |
| -t         | 输出时，[tab] 键按照空格键来输出                           |
| -T         | 输出时，按照 [tab] 键输出而不是按照空格键输出                    |
| -u         | 使用统一的输出格式                                     |
| -y         | 输出时使用对比格式                                     |

选择性参数的具体说明如下：

| 参数                     | 功能                                       |
|------------------------|------------------------------------------|
| -D<宏名字>                | 引入 if-then-else 输出格式来预处理宏名称              |
| -new-group-format=<模式> | 用指定的模式，按照 if-then-else 方式，输出第二个文件中的一组行内容 |
| -s<文件>                 | 当比较文件时，从指定的文件进行比较                        |
| -from-file=<文件>        | 同所有的操作数比较指定的文件，可以是目录                     |
| -version               | 版本信息                                     |
| -help                  | 帮助信息                                     |

### ◎ 实际范例

- ◎ 范例 422 ◎ 比较两个文件的异同。

```
[root@localhost ~]# cat 123 //显示文件 123 的内容
i am from bj
i like sports
```



```

espacially basketball and football
my favorite football club is acmilan
i will love it for ever
[root@localhost ~]# cat abc //显示文件abc的内容
i am a student
i am from bj
i like sports
espacially basketball and football
my favorite football club is intermilan
i will love it for ever
[root@localhost ~]# diff 123 abc //比较两个文件的不同
0a1
> i am a student
4c5
< my favorite football club is acmilan

> my favorite football club is intermilan
[root@localhost ~]#

```

本例使用指令“diff 123 abc”，显示文件 123 和文件 abc 的不同，运行结果显示“i am a student”是新增的行，文件 123 的第 4 行和文件 abc 的第 5 行不同，一个是“my favorite football club is acmilan”，另外一个是“my favorite football club is intermilan”。

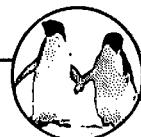
#### ◎ 范例 423 ◎ 比较文件夹的不同。

```

[root@localhost ~]# diff /home/kk/ /home/chord/ //比较文件夹的不同
Only in /home/kk/: abc.c
Only in /home/kk/: a.c
Only in /home/kk/: a.c~
Only in /home/kk/: a.c@
Only in /home/kk/: .bash_history
Only in /home/kk/: .bash_logout
Only in /home/kk/: .bash_profile
Only in /home/kk/: .bashrc
Only in /home/kk/: .canna
Only in /home/kk/: chch.c
Only in /home/kk/: cmh.c
Only in /home/kk/: D.c
Only in /home/kk/: Desktop
Only in /home/kk/: .dmrc

```

```
Only in /home/kk/: .eggcups
Only in /home/kk/: .emacs
Only in /home/kk/: .esd_auth
Only in /home/kk/: fcntl.c
Only in /home/kk/: file.c
Only in /home/kk/: findconf.c
Only in /home/kk/: .gconf
Only in /home/kk/: .gconfd
Only in /home/kk/: getcurtime.c
Only in /home/kk/: getsec.c
Only in /home/kk/: gettimeofday.c
Only in /home/kk/: .gettime.c.swp
Only in /home/kk/: getusec.c
Only in /home/kk/: .gnome
Only in /home/kk/: .gnome2
Only in /home/kk/: .gnome2_private
Only in /home/kk/: .gstreamer-0.8
Only in /home/kk/: .gtkrc
Only in /home/kk/: .gtkrc-1.2-gnome2
Only in /home/kk/: .ICEauthority
Only in /home/kk/: .kde
Only in /home/kk/: long.c
Only in /home/kk/: max.c
Only in /home/kk/: .metacity
Only in /home/chord/: multi-lsd
Only in /home/kk/: .nautilus
Only in /home/kk/: rand.c
Only in /home/kk/: .recently-used
Only in /home/kk/: .rhn-applet.conf
Only in /home/chord/: sfs1
Only in /home/chord/: sfs1.zip
Only in /home/chord/: sfs-build
Only in /home/chord/: sfsnet.zip
Only in /home/kk/: sprintf.c
Only in /home/kk/: strrchr.c
Only in /home/kk/: wenjian.c
Only in /home/kk/: .zshrc
Only in /home/kk/: zuobiao.c
[root@localhost ~]#
```



本例使用指令“`diff /home/kk/ /home/chord/`”，比较文件夹`/home/kk/`和文件夹`/home/chord/`的不同。

○范例 424 ○ 比较文件夹的不同，并递归比较文件夹下的子文件夹。

```
[root@localhost home]# diff -r test/ kk/ //比较两个文件夹的不同
```

```
Only in test/: 123.cpio
Only in kk/: a.c
Only in kk/: a.c~
Only in kk/: a.c@
Only in kk/: .bash_history
Only in kk/: .bash_logout
Only in kk/: .bash_profile
Only in kk/: .bashrc
Only in kk/: .canna
Only in kk/: Desktop
Only in kk/: .dmrc
Only in kk/: .eggcups
Only in kk/: .emacs
Only in kk/: .esd_auth
Only in kk/: .gconf
Only in kk/: .gconfd
Only in kk/: .gettime.c.swp
Only in kk/: .gnome
Only in kk/: .gnome2
Only in kk/: .gnome2_private
Only in kk/: .gstreamer-0.8
Only in kk/: .gtkrc
Only in kk/: .gtkrc-1.2-gnome2
Only in kk/: .ICEauthority
Only in kk/: .kde
Only in kk/: .metacity
Only in kk/: .nautilus
Only in kk/: .recently-used
Only in kk/: .rhn-applet.conf
Only in kk/: .zshrc
[root@localhost home]#
```

本例使用指令比较`kk`文件夹和`test`文件夹的不同，同时指定“`-r`”参数递归比较两个文件夹下的子文件夹的不同。

## 5.12 diff 指令：生成差异信息

- ◎ 范例 425 ◎ 比较两个文件的不同，并生成补丁。

```
[root@localhost ~]# diff -ruN abc 123 >kk.diff //比较文件abc和123,
```

生成差异文件

```
[root@localhost ~]# cat kk.diff
--- abc 2007-05-15 10:05:28.000000000 +0800
+++ 123 2007-05-15 10:06:08.000000000 +0800
@@ -1,6 +1,5 @@
-i am a student
 i am from bj
 i like sports
 espacially basketball and football
-my favorite football club is intermilan
+my favorite football club is acmilan
 i will love it for ever
[root@localhost ~]#
```

本例使用指令“diff -ruN abc 123 >kk.diff”比较文件abc 和文件123 的不同，并生成差异文件，此差异文件可以作为补丁。

- ◎ 范例 426 ◎ 打补丁。

```
[root@localhost ~]# patch -p1 <kk.diff //打补丁
missing header for unified diff at line 3 of patch
can't find file to patch at input line 3
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:
```

```

|--- abc 2007-05-15 10:05:28.000000000 +0800
+++ 123 2007-05-15 10:06:08.000000000 +0800
```

```
File to patch: 123 //123为输入的文件
patching file 123
Reversed (or previously applied) patch detected! Assume -R? [n] y
[root@localhost ~]# cat 123
i am a student
i am from bj
i like sports
espacially basketball and football
my favorite football club is intermilan
i will love it for ever
[root@localhost ~]#
```



本例使用指令“patch -p1 <kk.diff”进行打补丁。在运行中选择要打补丁的文件，此处选择的打补丁的文件是 123，如上所示。打完补丁之后，文件 123 的内容和文件 abc 的内容相同，这就是打补丁的过程。

### ◆ 相关指令

cmp、comm、diff3、ed、patch、pr、sdiff

## 5.13

### diffstat 指令：diff 结果的统计信息

#### ◆ 基本用法 diff[必要参数][选择性参数]

◆ 功能说明 diffstat 指令读取 diff 指令的输出，然后统计信息。

#### ◆ 举个说明

必要参数的具体说明如下：

| 参数        | 功能           |
|-----------|--------------|
| -n<文件名长度> | 设置文件名长度      |
| -p<文件名长度> | 设置包含路径的文件名长度 |
| -w        | 输出栏位宽度       |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -V | 版本信息 |

#### ◆ 实际范例

● 范例 427 ● 显示 diff 指令的统计信息。

```
[root@localhost temp]# diff/home/ /root/ |diffstat //统计diff信息
home//kk !only
root//.ICEauthority !only
root//.Xauthority !only
root//.Xresources !only
root//.bash_history !only
root//.bash_logout !only
```

## 5.14 dirname 指令：显示文件的除名字外的路径

```

root//.bash_profile |only
root//.bashrc |only
root//.cshrc |only
root//.gconf |only
root//.gconfd |only
root//.gnome |only
root//.gnome-desktop |only
root//.gnome2 |only
root//.gnome2_private |only
root//.gstreamer |only
root//.gtkrc |only
root//.gtkrc-1.2-gnome2 |only
root//.metacity |only
root//.nautilus |only
root//.pyinput |only
root//.recently-used |only
root//.rhn-applet.conf |only
root//.tcshrc |only
root//.xsession-errors |only
root//anaconda-ks.cfg |only
root//install.log |only
root//install.log.syslog |only
28 files changed
[root@localhost temp]#

```

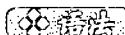
本例使用指令“diff /home/ /root/difffstat”统计 diff 指令的统计信息。

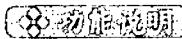
 **相关指令**

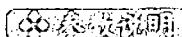
diff

## 5.14

## dirname 指令：显示文件的除名字外的路径

 **命令** dirname[选择性参数][路径]

 **功能说明** dirname 指令打印出指定路径的除了名字之外的路径前缀。

 **参数说明**

选择性参数的具体说明如下：



| 参数        | 功能   |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |

### 操作范例

◎ 范例 428 ◎ 显示文件指定路径除了名字之外的路径前缀。

```
[root@localhost ~]# dirname /home/kk/a.c //显示除了名字之外的路径前缀
/home/kk
[root@localhost ~]#
```

本例使用指令“dirname /home/kk/a.c”显示路径/home/kk/a.c 中除了文件名字 a.c 的路径前缀，也就是/home/kk，如上所示。

### 相关指令

rmdir、mkdir

## 5.15 file 指令：辨识文件类型

### 语法

file[必要参数][选择性参数]文件或者目录

◎ 功能说明 ◎ file 指令根据给定的参数给指定的文件分类，有 3 种类型的设定：文件系统测试、魔法数字测试和语言测试。如果第一个测试成功将输出文件的类型。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能                 |
|----|--------------------|
| -b | 文件名不显示             |
| -c | 详细显示处理过程           |
| -L | 操作对象为符号链接所指向的对象    |
| -z | 压缩文件的处理            |
| -C | 列出 magic.mgc 的输出文件 |
| -n | 将结果打印到标准输出         |

## 5.16 filterdiff 指令：从 diff 文件中提取不同

选择性参数的具体说明如下：

| 参数      | 功能         |
|---------|------------|
| -f<文件名> | 指定文件名      |
| -m<文件名> | 指定魔法数字的文件名 |
| -v      | 版本信息       |
| -help   | 帮助信息       |

## ◎ 范例与技巧

## ◎ 范例 429 ◎ 显示文件类型。

```
[root@localhost ~]# file * //显示当前文件夹下文件的类型
123: ASCII text
123.orig: ASCII text
25: empty
abc: ASCII text
anaconda-ks.cfg: ASCII English text
Desktop: directory
install.log: UTF-8 Unicode text
install.log.new: empty
install.log.syslog: UTF-8 Unicode text
kk.diff: ASCII text
test: ASCII English text
workspace: directory
[root@localhost ~]#
```

本例使用指令“file \*”显示当前文件夹下文件的类型，具体情况如上所示。

## ◎ 相关指令

magic、strings、od、hexdump

## 5.16

## filterdiff 指令：从 diff 文件中提取不同

## ◎ 命令

filterdiff[必要参数][选择性参数][文件]


◆ 功能说明

**filterdiff** 指令用来提取 diff 文件的不同。

◆ 参数说明

必要参数的具体说明如下：

| 参 数   | 功 能          |
|-------|--------------|
| -list | 等同于 lsdiff   |
| -grep | 等同于 grepdiff |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

◆ 举个范例

◎ 范例 430 ◎ 提取 patch 信息。

```
[root@localhost ~]# filterdiff message-with-diff-in-the-body >patch
//提取信息
```

本例使用指令“filterdiff message-with-diff-in-the-body >patch”提取 diff 信息。

◆ 相关指令

lsdiff、grepdiff

## 5.17 find 指令：查找目录或者文件

◆ 基本

**find[目录][选择性参数][必要参数]**

◆ 功能说明

**find** 指令用于在文件树中查找文件，并做出相应的处理。

◆ 参数说明

必要参数的具体说明如下：

## 5.17 find 指令：查找目录或者文件

| 参数        | 功能                           |
|-----------|------------------------------|
| -daystart | 时间基准从当日算起                    |
| -depth    | 指定搜索顺序，从文件树的最子层目录查起          |
| -empty    | 查找空文件或者空目录                   |
| -false    | 设置 find 指令的返回值为 false        |
| -follow   | 不查找符号链接                      |
| -mount    | 等同于 “-xdev”                  |
| -noleaf   | 不考虑硬链接限制                     |
| -nouser   | 没有用户的文件                      |
| -nogroup  | 没有群组的文件                      |
| -print    | 将查找结果列到标准输出，但是 find 的返回值必须为真 |
| -xdev     | 只查找当前目录下的文件                  |

选择性参数的具体说明如下：

| 参数              | 功能                                                                                                       |
|-----------------|----------------------------------------------------------------------------------------------------------|
| -amin<时间>       | 指定在指定时间内曾被存取过的文件，单位为 min                                                                                 |
| -anewer<文件或者目录> | 指定参考文件，查找的文件的存取时间比指定的参考文件的存取时间要晚                                                                         |
| -atime<24 小时数>  | 查找指定时间内存取过的文件。如果是“3”，表示 72 小时之前被存取过的文件或者目录；如果是“+3”，表示超过 72 小时之前被存取过的文件或者目录；如果是“-3”，表示 72 小时之内被存取过的文件或者目录 |
| -ctime<时间>      | 指定时间内被修改过的文件或者目录，单位是 min                                                                                 |
| -newer<文件或者目录>  | 等同于 “-anewer<文件或者目录>”                                                                                    |
| -ctime<24 小时数>  | 等同于 “-atime<24 小时数>”                                                                                     |
| -exec<指令>       | 执行指定的指令，但是 find 的返回值必须为真                                                                                 |
| -fls<文件>        | 将指令运行的结果保存到指定的文件                                                                                         |
| -fprintf<文件>    | 将输出结果输出到指定的文件                                                                                            |
| -ftype<文件类型>    | 只查找指定文件类型的文件或者目录                                                                                         |
| -group<群组>      | 只查找属于指定群组的文件或者目录                                                                                         |
| -help           | 帮助信息                                                                                                     |



续表

| 参数          | 功能              |
|-------------|-----------------|
| -version    | 版本信息            |
| -name<样式>   | 指定字符串作为搜索时的样式   |
| -ok<指令>     | 等同于“-exec<指令>”  |
| -perm<权限数值> | 查找符合权限数值的文件或者目录 |
| -print<样式>  | 指定输出样式          |
| -size<字节数>  | 查找指定字节数的文件      |
| -type<文件类型> | 查找指定类型的文件       |
| -user<用户>   | 查找指定用户的文件或者目录   |

### ❖ 执行范例

#### ◎ 范例 431 ◎ 查找指定时间内修改的文件。

```
[root@localhost ~]# find -atime -3 //查找 72 小时内修改过的文件
.
./.gnome2/panel2.d/default/launchers/larry-00c1672919.desktop
./.gnome2/panel2.d/default/launchers/eek-00c276ca7a.desktop
./.gnome2/panel2.d/default/launchers/larry-0093596ba3.desktop
./.gnome2/panel2.d/default/launchers/foo-007b291a28.desktop
./.gnome2/panel2.d/default/launchers/curlly-00b76df261.desktop
./.gnome2/nautilus-scripts
./.gnome2/keyrings
./.gnome2/share/fonts
./.gnome2/share/fonts/fonts.dir
./.gnome2/share/cursor-fonts
./.gnome2/share/cursor-fonts/fonts.dir
./install.log.syslog
./bash_logout
./gconfd/saved_state
./install.log.new
./bashrc
./gstreamer-0.8/registry-i386.xml
./metacity/sessions/1179728012-6412-55464834.ms
./kde/share/config/kdeglobals
```

## 5.17 find 指令：查找目录或者文件

第  
5  
章

```
./.recently-used
./.bash_profile
./.Trash
./123
./.rhn-applet.conf
./test
./123.orig
./abc
./.bash_history
./.Xauthority

./.nautilus/metafiles/file:%2F%2F%2Froot%2F.Trash.xml
./.nautilus/metafiles/x-nautilus-desktop:%2F%2F%2F.xml
./.nautilus/metafiles/file:%2F%2F%2Froot%2FDesktop.xml
./kk.diff
./install.log
./25
./anaconda-ks.cfg
[root@localhost ~]#
```

本例使用指令“find -atime -3”查找 72 小时之内修改过的文件。

◎ 范例 432 ◎ 按字符串格式查找文件。

```
[root@localhost ~]#find -name linux.* //查找指定字符串类型的文件
/usr/local/BerkeleyDB.4.4/docs/ref/build_unix/linux.html
/usr/X11R6/lib/Server/config/cf/linux.cf
/usr/X11R6/lib/X11/config/linux.cf
/usr/include/net-snmp/system/linux.h
/usr/include/parted/linux.h
/usr/include/pgsql/server/port/linux.h
/usr/include/boost/config/platform/linux.hpp
/usr/share/dict/linux.words
/usr/share/doc/db4-devel-4.3.27/ref/build_unix/linux.html
/usr/share/doc/kernel-doc-2.6.11/Documentation/usb/linux.inf
/usr/share/doc/HTML/en/kdevelop/reference/GRAPHICS/linux.gif
/usr/share/doc/IIIM/build/linux.html
/usr/share/doc/IIIM/test/linux.html
/usr/share/emacs/21.4/lisp/term/linux.el
/usr/share/sendmail-cf/ostype/linux.m4
```



```
/usr/share/apps/konsole/linux.desktop
/usr/share/apps/konsole/linux.keytab
/usr/share/pvm3/xpvm/src/xbm/linux.xbm
find: WARNING: Hard link count is wrong for /proc: this may be a bug in
your filesystem driver. Automatically turning on find's -noleaf option.
Earlier results may have failed to include directories that should have been
searched.
```

[root@localhost ~]#

本例使用指令“`find / -name linux.*`”查找整个文件系统中以“`linux.*`”打头的文件，并输出到标准输出。

◎ 范例 433 ◎ 按照目录或者文件的权限来查找文件。

```
[root@localhost ~]# find /home -perm 777 //查找指定权限数值的文件
/home/chord/sfs1/tests
/home/chord/sfs1/tests/CVS
/home/chord/sfs1/rex
/home/chord/sfs1/rex/CVS
/home/chord/sfs1/py
/home/chord/sfs1/py/CVS
/home/chord/sfs1/doc
/home/chord/sfs1/doc/CVS
/home/chord/sfs1/devel
/home/chord/sfs1/devel/rex
/home/chord/sfs1/async/CVS
/home/chord/sfs1/sfscd
/home/chord/sfs1/sfscd/CVS
/home/chord/sfs1/uvfs
/home/chord/sfs1/uvfs/common
/home/chord/sfs1/uvfs/common/CVS
/home/chord/sfs1/uvfs/acinclude.m4
/home/chord/sfs1/uvfs/openbsd
/home/chord/sfs1/uvfs/openbsd/CVS
/home/chord/sfs1/uvfs/arpogen
/home/chord/sfs1/uvfs/arpogen/CVS
/home/chord/sfs1/uvfs/CVS
/home/chord/sfs1/uvfs/linux
/home/chord/sfs1/uvfs/linux/CVS
```

## 5.17 find 指令：查找目录或者文件

```
/home/chord/sfs1/uvfs/server
/home/chord/sfs1/uvfs/server/CVS
/home/chord/sfs1/uvfs/setup
/home/chord/sfs1/sfsrosd
/home/chord/sfs1/sfsrosd/CVS
/home/chord/sfs1/sfssd
/home/chord/sfs1/sfssd/CVS
/home/chord/sfs1/setup
/home/chord/sfs1/sfsrocd
/home/chord/sfs1/sfsrocd/CVS
/home/chord/multi-lsd
/home/software/zhcon-0.2
/home/shal/Makefile
/home/shal/shatest.exe
/home/ftp/ncftp/ncftp-3.2.0/bin/ncftpspooler
[root@localhost ~]#
```

本例使用指令“`find /home -perm 777`”查找`/home`下权限值为 777 的文件，并将结果输出到标准输出。

## ◎ 范例 434 ◎ 查找不属于任何用户和群组的目录或者文件。

```
[root@localhost ~]# find / -nogroup -nouser //查找指定用户群组的文件
find: WARNING: Hard link count is wrong for /proc: this may be a bug in
your filesystem driver. Automatically turning on find's -noleaf option.
Earlier results may have failed to include directories that should have been
searched.
```

```
find: /proc/7718/task/7718/fd/4: 没有那个文件或者目录
find: /proc/7718/fd/4: 没有那个文件或者目录
[root@localhost ~]#
```

本例使用指令“`find / -nogroup -nouser`”查找所有的文件系统下不属于任何用户和群组的目录或者文件。

## ◎ 范例 435 ◎ 按照类型查找目录或者文件。

```
[root@localhost chord]# find / -type f -name linux -print
//按照类型查找文件
/home/linux/linux
/usr/share/doc/ntp-4.2.0.a.20040617/build/hints/linux
/usr/share/terminfo/l/linux
find: WARNING: Hard link count is wrong for /proc: this may be a bug in
```



your filesystem driver. Automatically turning on find's -noleaf option. Earlier results may have failed to include directories that should have been searched.

```
[root@localhost chord]# find / -type f -name linux -print
//按照类型查找文件
/home/chord/sfs1/uvfs/linux
/home/linux
/usr/src/kernels/2.6.11-1.1369_FC4-i686/include/linux
/usr/src/kernels/2.6.11-1.1369_FC4-i686/include/config/logo/linux
/usr/X11R6/lib/Server/include/linux
/usr/X11R6/lib/Server/modules/linux
/usr/X11R6/lib/modules/drivers/linux
/usr/X11R6/lib/modules/input/linux
/usr/X11R6/lib/modules/linux
/usr/lib/eclipse/plugins/org.eclipse.cdt.core.linux.x86_3.0.0/os/linux
/usr/lib/jvm/java-1.4.2-gcj-1.4.2.0/include/linux
/usr/lib/perl5/5.8.6/i386-linux-thread-multi/linux
/usr/include/linux
/usr/include/valgrind/linux
/usr/share/eclipse/configuration/org.eclipse.osgi/bundles/64/1/.cp/os/linux
/usr/share/eclipse/configuration/org.eclipse.osgi/bundles/65/1/.cp/os/linux
/usr/share/eclipse/configuration/org.eclipse.osgi/bundles/63/1/.cp/os/linux
/usr/share/eclipse/plugins/org.eclipse.update.core.linux_3.1.0/os/linux
/usr/share/eclipse/plugins/org.eclipse.core.resources.linux_3.0.0/os/linux
/usr/share/eclipse/plugins/org.eclipse.cdt.core.linux.x86_3.0.0/os/linux
find: WARNING: Hard link count is wrong for /proc: this may be a bug in
your filesystem driver. Automatically turning on find's -noleaf option.
Earlier results may have failed to include directories that should have been
searched.
```

```
[root@localhost chord]#
```

本例首先使用指令“`ind / -type f -name linux -print`”查找文件名为“linux”的文件，参数“`-type f`”指定了查找的类型为文件；然后使用指令“`find / -type d -name linux -print`”查找文件名“linux”的目录，参数“`-type f`”指定了查找的类型为目录。

`type` 的其他几个参数如下所示。

□ `f`: 普通文件。

- d: 目录。
- l: 符号链接。
- b: 块设备。
- c: 字符设备。
- p: 管道。
- s: socket。

◎ 范例 436 ◎ 查找到相关的指令之后，再进行相应的指令操作。

```
[root@localhost ~]# find . -name '*.zip' -exec unzip "{}" \; //找到文件
```

后执行指定的指令

```
Archive: ./.
inflating: 123.cpio
inflating: abc.mp3
inflating: abc.mp3.conf
inflating: a.o
inflating: b.o
inflating: chch
inflating: chch.c
inflating: cmh
inflating: cmh.c
inflating: cstring.cpp
inflating: D
inflating: D.c
inflating: fcntl
inflating: fcntl.c
inflating: file
inflating: file.c
inflating: findconf
inflating: findconf.c
inflating: getcurtime
inflating: getcurtime.c
inflating: getsec
inflating: getsec.c
inflating: gettimeofday
inflating: gettimeofday.c
inflating: getusec
inflating: getusec.c
inflating: gmon.out
```



```

extracting: kk
extracting: lib
inflating: lib.a
inflating: long
inflating: long.c
inflating: max
inflating: max.c
inflating: rand
inflating: rand.c
inflating: sprintf
inflating: sprintf.c
inflating: strrchr
inflating: strrchr.c
replace test? [y]es, [n]o, [A]ll, [N]one, [r]ename: NULL
(assuming [N]one)
inflating: wenjian
inflating: wenjian.c
inflating: zuobiao
inflating: zuobiao.c
[root@localhost ~]#

```

本例使用指令“`find . -name *.zip -exec unzip "{}" \;`”查找当前文件夹下的所有的.zip 文件，并对查到的文件进行解压缩。指令的最后必须以分号结尾，“{}”代表了查到的文件名称，注意指令中的空格。

#### 相关命令

`locate`、`locatedb`、`updatedb`、`xargs`、`regex`、`stat`、`lstat`、`ls`、`printf`、`strftime`、`ctime`、`Finding Files`

## 5.18

### findfs 指令：通过列表或者用户 ID 查找文件系统

**语义** `findfs` [LABEL=卷标][UUID=uuid][选择性参数]

**功能说明** 指令 `findfs` 通过卷标或者 uuid 查找文件系统，找到的设备名将输出到标准输出。

#### 选择性说明

选择性参数的具体说明如下：

## 5.20 indent 指令：调整 C 原始代码文件的格式

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ❖ 执行范例

- ① 范例 437 ② 通过卷标名称查找文件系统。

```
[root@localhost ~]# findfs LABEL=/ //通过卷标名称查找文件系统
/dev/sda3
[root@localhost ~]# findfs LABEL=/boot //通过卷标名称查找文件系统
/dev/sdal
[root@localhost ~]#
```

本例使用指令“findfs LABEL=/”查找卷标名为“/”的文件系统，然后使用指令“findfs LABEL=/boot”查找卷标名为“/boot”的文件系统。

### ❖ 相关指令

fsck

## 5.19 git 指令：在文字模式下管理文件

### ❖ 语法

git

### ❖ 功能说明

git 指令用来管理文件。

### ❖ 执行范例

- ① 范例 438 ② 管理文件与目录。

```
[root@localhost littleProg]#git //管理文件与目录
```

本例使用指令 git 管理文件与目录。

### ❖ 相关指令

dir、ls

## 5.20 indent 指令：调整 C 原始代码文件的格式

### ❖ 语法

indent[必要参数][选择性参数][源文件][-o 目标文件]



**功能说明** indent 指令可以使得代码更容易读懂。它可以转化一种类型的源代码到另一种类型。

**参数说明**

必要参数的具体说明如下：

| 参数    | 功能           |
|-------|--------------|
| -bad  | 声明后加空白行      |
| -bap  | 程序后加空白行      |
| -bbb  | 注释后加空白行      |
| -bc   | 逗号换行         |
| -bl   | if 格式设置      |
| -br   | if 格式设置      |
| -bs   | sizeof 后空一格  |
| -cdb  | 注释自成一行       |
| -ce   | else 设置      |
| -fca  | 设置所有的注释格式    |
| -gnu  | 设置 GNU 格式    |
| -orig | Berkeley 格式  |
| -st   | 结果输出到标准输出    |
| -v    | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -cli<格式> | {}缩排格数 |
| -c<栏位>   | 注释设置   |
| -cd<栏位>  | 注释设置   |

**执行范例**

**范例 439** 以 GNU 格式处理 C 文件。

```
[root@localhost littleProg]# cat file.c
```

```
#include <stdio.h>
```

## 5.20 indent 指令：调整 C 原始代码文件的格式

```

int main ()
{
 FILE *fp;
 int a, b;
 fp = fopen ("abc.mp3", "rb");
 a = ftell (fp);
 printf ("$$$$$$$$$$$$$$$$$$$$d\n", a);
 fseek (fp, 10, 0);
 b = ftell (fp);
 printf ("oooooooooooooooooqd\n", b);
 fclose (fp);
}
[root@localhost littleProg]# indent file.c //以GNU格式来处理C文件
[root@localhost littleProg]# cat file.c
#include <stdio.h>
int
main ()
{
 FILE *fp;
 int a, b;
 fp = fopen ("abc.mp3", "rb");
 a = ftell (fp);
 printf ("$$$$$$$$$$$$$$$$$$$$d\n", a);
 fseek (fp, 10, 0);
 b = ftell (fp);
 printf ("oooooooooooooooooqd\n", b);
 fclose (fp);
}

```

本例使用指令“indent file.c”以 GNU 的格式来处理 C 文件。

◎ 范例 440 ◎ 以 Berkeley 的格式来处理 C 文件。

```

[root@localhost littleProg]# cat file.c
#include <stdio.h>
int main ()
{
 FILE *fp;
 int a, b;
 fp = fopen ("abc.mp3", "rb");
 a = ftell (fp);
 printf ("$$$$$$$$$$$$$$$$$$$$d\n", a);

```



```
fseek (fp, 10, 0);
b = ftell (fp);
printf ("@@@@@@@@eeeeeeeeed\n", b);
fclose (fp);
[root@localhost littleProg]# indent -st -orig -v file.c //以 Berkeley 的
格式处理C文件
#include <stdio.h>
int
main()
{
 FILE *fp;
indent: file.c:5: Warning:Line broken 2
 int a,
 b;
 fp = fopen("abc.mp3", "rb");
 a = ftell(fp);
 printf("#####$$$$$$$$##@d\n", a);
 fseek(fp, 10, 0);
 b = ftell(fp);
 printf("@@@@eeeeeeeeeeee@d\n", b);
indent: file.c:12: Warning:Line broken 2
 fclose(fp);
}
There were 15 non-blank output lines and 0 comments
[root@localhost littleProg]#
```

本例使用指令“`indent -st -orig -v file.c`”以 Berkeley 的格式处理 C 文件 `file.c`，并将处理的结果输出到标准输出。



fmt

5.21

### In 指令：链接文件或目录



**ln[必要参数][选择性参数][源文件或目录][目标文件或目录]**



**ln** 指令用来链接文件或目录。当同时指定多个文件

或目录时，目的目录必须存在，若不存在则出错，若存在，则指定的多个文件和目录将被复制到目的目录下。当不指定任何参数时将创建硬链接，使用“-symbolic”参数时将创建符号链接。当目标不存在时，不能创建硬链接。

### 【参数说明】

必要参数的具体说明如下：

| 参 数 | 功 能                |
|-----|--------------------|
| -b  | 删除，覆盖以前建立的链接       |
| -d  | 允许超级用户制作目录的硬链接     |
| -f  | 强制模式，无条件予以覆盖       |
| -i  | 交互模式，覆盖文件前先征求用户的同意 |
| -n  | 把符号链接的目的目录视为一般文件   |
| -s  | 符号链接（默认方式是硬链接）     |
| -v  | 显示指令执行过程           |

选择性参数的具体说明如下：

| 参 数       | 功 能                                   |
|-----------|---------------------------------------|
| -S        | “-S<字尾备份字符串>”或“--suffix=<字尾备份字符串>”    |
| -V        | “-V<备份方式>”或“--version-control=<备份方式>” |
| --help    | 帮助信息                                  |
| --version | 版本信息                                  |

### 【概念说明】

- **链接：**新建一个文件，新建的文件和已经存在的文件之间存在着某种关系。
- **硬链接：**新建的文件是已经存在的文件的一个别名，当原文件删除时，新建的文件仍然可用。
- **符号链接：**新建的文件以“路径”的形式来表示另一个文件，类似于一种快捷方式。



### ◎ 功能说明

- 范例 441 ○ 建立硬链接。

```
$ ln myfile hardlink //建立硬链接
```

hardlink 为指定的链接文件，它和文件 myfile 指向物理磁盘中的同一 inode，它们的各种属性相同，一个发生变化将引起另一个的变化。只有超级用户并且使用 “-d” 参数，才能给目录建立硬链接。如果链接的源是文件而目标是已经存在的目录，那么则在目的目录中生成一个链接文件。

- 范例 442 ○ 建立符号链接。

```
$ ln -s myfile softlink //建立符号链接
```

softlink 为指定的链接文件，它以“路径”的形式表示已经存在的文件 myfile。如果文件位置发生变化，则相应的“路径”将发生变化，已建立的符号链接将失效，为了避免这种情况，一般会使用绝对路径建立符号链接。符号链接的目标可以是目录，其路径是相对路径。以“路径”形式表示的符号链接可以跨越磁盘分区，而硬链接不可以跨越磁盘分区。

### ◎ 相关命令

readlink

## 5.22 lndir 指令：连接目录内容

### ◎ 命令

**lndir [参数] [源目录] [目的目录]**

**◎ 功能说明** 执行 lndir 指令可以把源目录下的文件和子目录都建立起对应的符号连接到目的目录中去。如果没有目的目录，则在现行目录中建立符号连接。源目录必须用绝对路径。

### ◎ 选择性说明

选择性参数的具体说明如下：

| 参数           | 功能                                               |
|--------------|--------------------------------------------------|
| -ignorelinks | 直接建立各个符号连接之间的连接。不采用这个参数的指令遇到符号连接时，预设会维持该连接指向原始文件 |
| -silent      | 不显示指令的执行过程                                       |

**○范例范例**

- 范例 443 ○ 给目录 uddi-matchmaker 下的所有文件和子目录建立符号连接。

```
[root@wjc backup]# lndir /home/wjc/uddi-matchmaker abc
/home/wjc/uddi-matchmaker/dist:
/home/wjc/uddi-matchmaker/docs:
/home/wjc/uddi-matchmaker/docs/api:
/home/wjc/uddi-matchmaker/docs/api/EDU:
/home/wjc/uddi-matchmaker/docs/api/EDU/cmu:
/home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas:
/home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker:
/home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker/algorithm:
/home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker/algorithm
/advmgmt:
 /home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker/algorithm
/class-use:
 /home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker/algorithm
/classmgmt:
 /home/wjc/uddi-matchmaker/docs/api/EDU/cmu/Atlas/matchmaker/algorithm
/datastructure:
```

**○相关指令**

ln

**5.23****locate 指令：查找文件****○语法规则**

locate[选择性参数]范本样式

**○功能说明**

locate 指令用来查找符合范本样式的文件和目录。

在范本样式中可以采用正则表达式，例如，采用“\*”和“？”等符号。与 find 指令相比，locate 指令从数据库中查找文件的速度比 find 指令要快得多。但对于新增的或者已经删除的文件，locate 指令便无法自行察觉。

**○参数说明**

选择性参数的具体说明如下：



| 参数                | 功 能                               |
|-------------------|-----------------------------------|
| -d <数据库文件>        | 预设 locate 命令所使用的数据库。命令预设的数据库在     |
| -database=<数据库文件> | /var/lib/slocate 中，名称为 slocate.db |
| --help            | 在线帮助                              |
| --version         | 显示版本信息                            |

### ◎ 指令范例

#### ◎ 范例 444 ◎ 查找文件。

```
[root@localhost root]# locate *.linux //查找文件
/usr/share/doc/shadow-utils-4.0.3/README.linux
/usr/share/doc/gawk-3.1.1/README_d/README.linux
/usr/share/doc/sysklogd-1.4.1/README.linux
/usr/share/doc/ppp-2.4.1/README.linux
/usr/share/doc/star-1.5a08/README.linux
/usr/share/doc/cdrecord-2.0/README.linux
/usr/share/doc/sane-backends-1.0.9/README.linux
/usr/share/doc/ethereal-0.9.8/README.linux
[root@localhost root]#
```

本例使用指令“locate \*.linux”查找含有“linux”字符的文件。

### ◎ 相关指令

find

## 5.24

## lsattr 指令：显示文件属性

**◎ 简介** lsattr[必要参数][目录或者文件]

**◎ 功能说明** lsattr 指令用来显示文件的属性。

### ◎ 重点提示

必要参数的具体说明如下：

| 参 数 | 功 能           |
|-----|---------------|
| -a  | 显示所有的目录和文件    |
| -R  | 同时处理文件夹下的子文件夹 |

## 5.24 lsattr 指令：显示文件属性

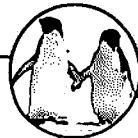
选择性参数的具体说明如下：

| 参 数 | 功 能    |
|-----|--------|
| -v  | 显示版本信息 |

## ◎ 操作范例

## ◎ 范例 445 ◎ 显示文件属性。

```
[root@localhost ~]# lsattr ----- ./install.log.syslog //显示文件属性
----- ./file1
----- ./workspace
----- ./mbox
----- ./xuehao
----- ./file2
----- ./inter_backup2
----- ./install.log.new
----- ./config-2.6.11-1.1369_FC4
----- ./lib.a
----- ./ddd
----- ./initrd-2.6.11-1.1369_FC4.img
----- ./123
----- ./lost+found
----- ./123.pcx
----- ./xen-syms
----- ./2.log
----- ./nc
----- ./test
----- ./abc
----- ./index.html
----- ./7.log
----- ./Desktop
----- ./lib
----- ./1.log
----- ./uhvbo8ac.pfa
----- ./nohup.out
----- ./inter
----- ./sushe
```



```
----- ./lib.new
----- ./memtest86+-1.55.1
----- ./temp
----- ./xen.gz
----- ./456
----- ./vmlinuz-2.6.11-1.1369_FC4
----- ./gmon.out
----- ./grub
----- ./exampleseed
----- ./System.map-2.6.11-1.1369_FC4
----- ./1.log.new
----- ./DEADJOE
----- ./install.log
----- ./anaconda-ks.cfg
----- ./restoresymtable
----- ./inter_backup1
----- ./uhvbo8ac.pfb
[root@localhost ~]#
```

本例使用指令 `lsattr` 显示当前文件夹下的文件属性。

**相关命令**

`chattr`

## 5.25

### mattrib 指令：变更或显示 MS-DOS 文件的属性

**语法**

`mattrib[必要参数][选择性参数][文件]`

**功能说明**

`mtools` 工具指令集中的指令模拟 MS-DOS 的 `attrib` 指令。

**参数说明**

必要参数的具体说明如下：

| 参数               | 功能                            |
|------------------|-------------------------------|
| <code>+/-</code> | “+”表示打开指定的文件属性，“-”表示关闭指令的文件属性 |
| <code>a</code>   | 存档位，被备份程序用来标识一个文件是不是新文件       |

## 5.26 mc 指令：交互式文件管理程序

续表

| 参数 | 功能                 |
|----|--------------------|
| r  | 只读位，用来标识一个只读文件     |
| s  | 系统位，用来标识一个操作系统文件   |
| h  | 隐藏位，用来使一个文件在目录中不可见 |
| /  | 列出所有子目录下的文件属性      |
| x  | 简明列出属性，不含任何空格      |
| p  | 专为 tar 指令配合使用的一个参数 |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -help    | 列出帮助信息 |
| -version | 列出版本信息 |

❖ 执行实例

- 范例 446 ◦ 列出 c 盘上所有文件的属性。

[root@localhost root]# mattrib c

❖ 功能简介

attrib

## 5.26

## mc 指令：交互式文件管理程序

❖ 语法

mc[必要参数][选择性参数][文件]

❖ 功能说明

mc 指令提供一个交互式的文件管理程序，具有图形化界面。

❖ 典型应用

必要参数的具体说明如下：



| 参数 | 功能          |
|----|-------------|
| -x | 使用 xterm 特性 |
| -d | 不支持鼠标       |
| -b | 黑白模式        |
| -c | 彩色模式        |

选择性参数的具体说明如下：

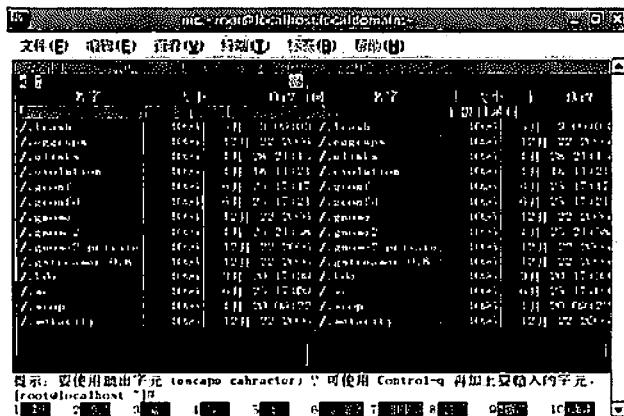
| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

执行范例

## ◎ 范例 447 ◎ 交互式文件处理界面。

[root@localhost root]#mc

本例使用指令 mc 进入交互式界面。



然后，根据菜单的提示，进行文件管理。

相关书籍

## lsattr

**5.27****mcopy 指令：复制 MS-DOS 文件到 UNIX 系统，或者从 UNIX 系统复制 MS-DOS 文件**

mcopy[必要参数][选择性参数][源文件][目标文件]



mcopy 指令用来复制 MS-DOS 文件到 UNIX 系统，或者从 UNIX 系统复制 MS-DOS 文件。



必要参数的具体说明如下：

| 参 数 | 功 能            |
|-----|----------------|
| -b  | batch 模式       |
| -s  | 递归复制目录下的子目录的文件 |
| -a  | 文本文件传输         |
| -v  | 显示详细的处理信息      |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



○ 范例 448 ○ 复制文件。

```
root@localhost:~# mcopy dosfile unixfile //复制文件
root@localhost:~#
```

本例使用指令“mcopy dosfile unixfile”复制 MS-DOS 文件。



mtype

**5.28****md5sum 指令：检验文件**

md5sum[选择性参数][必要参数][FILE]



## 第5章 文件管理

**功能说明** 计算文件的 md5 值，通过这个 md5 值检验文件。

**参数说明**

必要参数的具体说明如下：

| 参数     | 功能          |
|--------|-------------|
| -check | 检验哪个文件发生了变化 |

选择性参数的具体说明如下：

| 参数        | 功能            |
|-----------|---------------|
| -b        | 以二进制的模式读取文件   |
| -t        | 以文本的形式读取文件    |
| -help     | 帮助信息          |
| --version | 版本信息          |
| -w        | 不合适的哈希内容将产生警告 |

### 【概念说明】

**md5 值：**一种哈希值。哈希函数以文件内容为依据，生成一个统一位数的哈希值。

**重定向：**将指令执行结果输入到指定文件保存。

**操作示例**

○ 范例 449 ○ 计算当前文件夹下的文件 file1 的 md5 值。

```
[root@localhost other]# ls
file1 file2
[root@localhost other]# md5sum file1 //计算文件 file1 的 md5 值
681530dc5eef42e2b200511c64bcf819 file1
```

○ 范例 450 ○ 比较文件的不同。

```
[root@localhost other]# md5sum file1>sum1 //计算文件 file1 的 md5 值
[root@localhost other]# md5sum file2>sum2 //计算文件 file2 的 md5 值
[root@localhost other]# diff -q sum1 sum2 //比较两个文件的不同
Files sum1 and sum2 differ
```

哈希文件 file1 将结果重定向到文件 sum1，哈希文件 file2 将结果重定

## 5.29 mdel 指令：MS-DOS 文件删除

向到文件 sum2，用 diff 指令比较两个文件的不同，发现文件 sum1 和 sum2 不同，也就是文件 file1 和文件 file2 的哈希值不同，也就是文件 file1 和文件 file2 不同。diff 的用法可参见 diff 指令解释。

- ◎ 范例 451 ◎ 使用“--check”参数，检查一组文件中某些文件的改变。

```
[root@localhost other]# md5sum file1 file2 >sum //计算文件file1和file2的md5值
```

```
[root@localhost other]# md5sum --check sum //检验文件的内容是否发生变化
```

```
file1: OK
```

```
file2: OK
```

```
[root@localhost other]# echo "abc" >file2 //改变文件file2的内容
```

```
[root@localhost other]# md5sum --check sum //检验文件的内容是否发生变化
```

```
file1: OK
```

```
file2: FAILED
```

```
md5sum: WARNING: 1 of 2 computed checksums did NOT match
```

哈希文件 file1 和 file2 得到的哈希值存到文件 sum，利用“md5sum –check”指令检查文件 file1 和 file2 中的内容有没有变化，第一次检查时文件均没有发生变化，第二次利用 echo 指令改变文件 file2 的值，再次利用“md5sum –check”指令进行检查，发现文件 file2 发生变化。



sum、cksum

## 5.29 mdel 指令：MS-DOS 文件删除



mdel[文件……]



mdel 指令用来删除 MS-DOS 文件。



- ◎ 范例 452 ◎ 删除 MS-DOS 文件。

```
[root@localhost root]#mdel dosfile //删除MS-DOS文件
```

本例使用指令“mdel dosfile”删除 MS-DOS 文件 dosfile。


相关指令

mcopy

**5.30****mdir 指令：显示 MS-DOS 目录**
语法

mdir[必要参数][目录]

功能说明

mdir 指令模拟 MS-DOS 的 dir 指令，用来显示目录内容。

参数说明

必要参数的具体说明如下：

| 参 数 | 功 能       |
|-----|-----------|
| - / | 显示所有目录和文件 |
| -a  | 显示隐藏信息    |
| -w  | 横排模式      |
| -X  | 完整路径模式    |

执行命令

◎ 范例 453 ◎ 显示 MS-DOS 目录。

```
[root@localhost root]#mdir -X dosdir //显示MS-DOS目录
```

本例使用指令“mdir -X dosdir”以完整路径显示 MS-DOS 目录。

相关指令

mcopy、mdel

**5.31****mkdir 指令：建立目录**
语法

mkdir[选择性参数][必要参数]

功能说明

如果必要参数指定的目录不存在，则创建它。

参数说明

必要参数的具体说明如下：

## 5.32 mktemp 指令：建立暂存文件

| 参数    | 功能      |
|-------|---------|
| <目录名> | 创建指定的目录 |

选择性参数的具体说明如下：

| 参数            | 功能             |
|---------------|----------------|
| -m<目录属性>      | 设置创建的目录属性      |
| --mode=<目录属性> | 等同于“-m”        |
| -p            | 建立父目录          |
| --parents     | 建立父目录          |
| --help        | 显示帮助信息         |
| --version     | 输出版本信息         |
| -v            | 为每个创建的目录打印一条消息 |

### ◎ 实例范例

#### ◎ 范例 454 ◎ 创建一个新目录。

```
[yingzheng@yingzheng ~]$ ls
[yingzheng@yingzheng ~]$ mkdir new
[yingzheng@yingzheng ~]$ ls
new
```

#### ◎ 范例 455 ◎ 指定目录的属性。

```
[yingzheng@yingzheng ~]$ mkdir -m a=r new
[yingzheng@yingzheng ~]$ ls -l
总用量 8
dr--r---r-- 2 yingzheng yingzheng 4096 5月 25 20 02:15 new
```

### ◎ 相关指令

info coreutils mkdir

## 5.32

## mktemp 指令：建立暂存文件

### ◎ 语法

mktemp[选择性参数]



**功能说明** 这个程序根据指定的名字生成一个暂存文件，这个文件供 shell 脚本安全地使用临时文件。生成的暂存文件将打印出提示信息。

### 参数说明

选择性参数的具体说明如下：

| 参数     | 功    能              |
|--------|---------------------|
| -V     | 打印版本信息并退出           |
| -d     | 建立一个目录代替文件          |
| -p<目录> | 使用指定目录作为生成临时文件名时的前缀 |
| -q     | 发生错误时不打印信息          |

### 执行范例

- 范例 456 ○ 建立一个默认的暂存文件。

```
[root@yingzheng yingzheng]# mktemp
/tmp/tmp.DtVKGp6388
```

### 相关指令

mkdtemp、mkstemp、mktemp

## 5.33

## mmove 指令：移动 MS-DOS 文件

### 语法

mmove[源文件……][目标文件]

**功能说明** mmmove 指令用来移动 MS-DOS 文件或者给 MS-DOS 文件改名。

### 执行范例

- 范例 457 ○ 将 MS-DOS 文件改名。

```
[root@yingzheng yingzheng]# mmmove file1 file2 //将MS-DOS文件改名
```

本例使用指令“mmmove file1 file2”将 MS-DOS 文件 file1 改名为 file2。

### 相关指令

mren

## 5.36 mshowfat 指令：显示 MS-DOS 文件的记录

## 5.34 mread 指令：复制 MS-DOS 文件



mread[MS-DOS 文件……][Linux 文件]



mread 指令用来复制 MS-DOS 文件，是 mtools 工具指令。



- 范例 458 复制 MS-DOS 文件。

```
[root@yingzheng ~]#mread dosfile /home
```

本例使用指令 mread dosfile /home 复制 MS-DOS 文件 dosfile 到目录 /home 下。



mcopy

## 5.35 mren 指令：更改 MS-DOS 文件名称



mren[源文件……][目标文件]



mren 指令用来更改 MS-DOS 文件名称。



- 范例 459 更改 MS-DOS 文件名称。

```
[root@yingzheng ~]#mren dosfile1 dosfile2 //更改文件名称
```

本例使用指令 “mren dosfile1 dosfile2” 将 MS-DOS 文件 dosfile1 的名称更改为 dosfile2。



mcopy

## 5.36 mshowfat 指令：显示 MS-DOS 文件的记录



mshowfat[文件……]



 **功能说明** mshowfat 指令用来显示 MS-DOS 文件的记录。

 **执行范例**

◎ 范例 460 ◎ 显示 MS-DOS 文件记录。

[root@yingzheng ~]#mshowfat msdos.txt //显示MS-DOS文件的记录

本例使用指令“mshowfat msdos.txt”显示 MS-DOS 文件的记录。

 **相关指令**

mren

## 5.37

### mtools 指令：显示 mtools 支持的指令

 **语法** mtools

 **功能说明** 显示 mtools 支持的指令。

 **执行范例**

◎ 范例 461 ◎ 显示指令。

[root@yingzheng ~]# mtools

Supported commands:

mattrib, mbadblocks, mcat, mcd, mcopy, mdel, mdeltree, mdir  
mdoctorfat, mdu, mformat, minfo, mlabel, mmd, mmount, mpartition  
mrd, mread, mmove, mren, mshowfat, mtoolstest, mtype, mwrite  
mzip

 **相关指令**

mtoolstest

## 5.38

### mtoolstest 指令：测试并显示 mtools 的相关设置

 **语法** mtoolstest

 **功能说明** 测试并显示 mtools 的相关设置。

 **执行范例**

◎ 范例 462 ◎ 测试并显示。

## 5.38 mtoolstest 指令：测试并显示 mtools 的相关设置

```
[root@yingzheng ~]# mtoolstest
drive J:
 #fn=0 mode=0 builtin
 file="/dev/sdb4" fat_bits=16
 tracks=0 heads=0 sectors=0 hidden=0
 offset=0x0
 partition=0
 mformat_only

drive Z:
 #fn=0 mode=0 builtin
 file="/dev/sdb4" fat_bits=16
 tracks=0 heads=0 sectors=0 hidden=0
 offset=0x0
 partition=0
 mformat_only

drive X:
 #fn=0 mode=0 builtin
 file="$DISPLAY" fat_bits=0
 tracks=0 heads=0 sectors=0 hidden=0
 offset=0x0
 partition=0

drive A:
 #fn=2 mode=128 defined in /etc/mtools.conf
 file="/dev/fd0" fat_bits=0
 tracks=0 heads=0 sectors=0 hidden=0
 offset=0x0
 partition=0
 mformat_only
 exclusive

drive B:
 #fn=2 mode=128 defined in /etc/mtools.conf
 file="/dev/fd1" fat_bits=0
 tracks=0 heads=0 sectors=0 hidden=0
 offset=0x0
 partition=0
 mformat_only
 exclusive

drive N:
```



## 第5章！文件管理

```

#fn=2 mode=0 defined in /etc/mtools.conf
file="/var/lib/dosemu/hdimage" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x2280
partition=0

drive P:
#fn=2 mode=0 defined in /etc/mtools.conf
file=":0" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0

tounix:
0xc7 0xfc 0xe9 0xe2 0xe4 0xe0 0xe5 0xe7
0xea 0xeb 0xe8 0xef 0xee 0xec 0xc4 0xc5
0xc9 0xe6 0xc6 0xf4 0xf6 0xf2 0xfb 0xf9
0xff 0xd6 0xdc 0xf8 0xa3 0xd8 0xd7 0x5f
0xe1 0xed 0xf3 0xfa 0xf1 0xd1 0xaa 0xba
0xbf 0xae 0xac 0xbd 0xbc 0xa1 0xab 0xbb
0x5f 0x5f 0x5f 0x5f 0x5f 0xc1 0xc2 0xc0
0xa9 0x5f 0x5f 0x5f 0x5f 0xa2 0xa5 0xac
0x5f 0x5f 0x5f 0x5f 0x5f 0x5f 0xe3 0xc3
0x5f 0x5f 0x5f 0x5f 0x5f 0x5f 0x5f 0xa4
0xf0 0xd0 0xc9 0xcb 0xc8 0x69 0xcd 0xce
0xcf 0x5f 0x5f 0x5f 0x7c 0x49 0x5f
0xd3 0xdf 0xd4 0xd2 0xf5 0xd5 0xb5 0xfe
0xde 0xda 0xd9 0xfd 0xdd 0xde 0xaf 0xb4
0xad 0xb1 0x5f 0xbe 0xb6 0xa7 0xf7 0xb8
0xb0 0xa8 0xb7 0xb9 0xb3 0xb2 0x5f 0x5f

fucase:
0x80 0x9a 0x90 0xb6 0x8e 0xb7 0x8f 0x80
0xd2 0xd3 0xd4 0xd8 0xd7 0xde 0x8e 0x8f
0x90 0x92 0x92 0xe2 0x99 0xe3 0xea 0xeb
0x59 0x99 0x9a 0x9d 0x9c 0x9d 0x9e 0x9f
0xb5 0xd6 0xe0 0xe9 0xa5 0xa5 0xa6 0xa7
0xa8 0xa9 0xaa 0xab 0xac 0xad 0xae 0xaf
0xb0 0xb1 0xb2 0xb3 0xb4 0xb5 0xb6 0xb7
0xb8 0xb9 0xba 0xbb 0xbc 0xbd 0xbe 0xbf
0xc0 0xc1 0xc2 0xc3 0xc4 0xc5 0xc7 0xc7

```

## 5.39 mv 指令：移动或更名现有的文件或目录

```
0xc8 0xc9 0xca 0xcb 0xcc 0xcd 0xce 0xcf
0xd1 0xd1 0xd2 0xd3 0xd4 0x49 0xd6 0xd7
0xd8 0xd9 0xda 0xdb 0xdc 0xdd 0xde 0xdf
0xe0 0xe1 0xe2 0xe3 0xe5 0xe5 0xe6 0xe8
0xe8 0xe9 0xea 0xeb 0xed 0xed 0xee 0xef
0xf0 0xf1 0xf2 0xf3 0xf4 0xf5 0xf6 0xf7
0xf8 0xf9 0xfa 0xfb 0xfc 0xfd 0xfe 0xff
```

mtools\_fat\_compatibility=1

mtools\_skip\_check=1

mtools\_lower\_case=0

**相关指令**

mtools

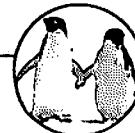
**5.39****mv 指令：移动或更名现有的文件或目录****功能说明** mv[选择性参数] [必要参数]**功能说明** 移动或者重命名指定文件。**功能说明**

必要参数的具体说明如下：

| 参数  | 功 能            |
|-----|----------------|
| 源文件 | 指定要操作的源文件      |
| 目标  | 目标目录或者重新命名后的文件 |

选择性参数的具体说明如下：

| 参数                       | 功 能            |
|--------------------------|----------------|
| -b                       | 为每个存在的目标文件制作备份 |
| -f                       | 覆盖时不作提示        |
| -i                       | 覆盖时提示          |
| --reply=<yes/no/query>   | 指定在遇到提示时的回答    |
| --strip-trailing-slashes | 删除所有源参数尾部的斜线   |



续表

| 参数                      | 功能                        |
|-------------------------|---------------------------|
| -s                      | 不理会通常的备份后缀                |
| --target-directory=<目录> | 将所有的源文件移入指定目录             |
| -u                      | 只有当源文件比目标文件新或者目标文件不存在时才移入 |
| -v                      | 输出执行信息                    |
| --help                  | 输出帮助信息                    |
| --version               | 输出版本信息                    |

### 操作实例

- 范例 463 ○ 将文件 newfifo 移入 yingzheng 文件夹内。

```
[root@yingzheng home]# ls
lqf new newfile test ws yingzheng
[root@yingzheng home]# mv newfile yingzheng
[root@yingzheng home]# ls
lqf new test ws yingzheng
[root@yingzheng home]# cd yingzheng/
[root@yingzheng yingzheng]# ls
fonts.dir newfifo newfile test testiso yy
[root@yingzheng yingzheng]#
```

- 范例 464 ○ 将 yy 文件更名为 xx 文件。

```
[root@yingzheng yingzheng]# ls
fonts.dir newfifo newfile test testiso yy
[root@yingzheng yingzheng]#
[root@yingzheng yingzheng]# mv yy xx
[root@yingzheng yingzheng]# ls
fonts.dir newfifo newfile test testiso xx
```

### 相关命令

info coreutils mv

## 5.40 od 指令：输出文件内容

### 命令行

od[必要参数][选择性参数][文件]

## 5.40 od 指令：输出文件内容

**功能说明** od 指令会显示指定文件的指令，显示的格式为八进制。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能         |
|-----|-------------|
| -v  | 如果有重复的数据也输出 |

选择性参数的具体说明如下：

| 参 数       | 功 能                                                                  |
|-----------|----------------------------------------------------------------------|
| -A<进制>    | 指定进制<br>a: 名称字符<br>c: ASCII<br>d: 十进制<br>o: 八进制<br>u: 十进制<br>x: 十六进制 |
| -t<进制>    | 输出格式<br>a: 名称字符<br>c: ASCII<br>d: 十进制<br>o: 八进制<br>u: 十进制<br>x: 十六进制 |
| -s<字符串数目> | 显示指定数目的字符串                                                           |
| -help     | 帮助信息                                                                 |
| -version  | 版本信息                                                                 |
| -w<字符数>   | 设置每列的最大字符数                                                           |

**执行范例**

◎ 范例 465 ◎ 以八进制的形式计算字码。

```
[root@localhost ~]# cat abc //显示文件内容
a
```



```

ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefgij
abcdefgijk
abcdefghijkl

[root@localhost ~]# cd abc //以八进制的形式显示文件内容
0000000 005141 061141 060412 061542 060412 061542 005144 061141
0000020 062143 005145 061141 062143 063145 060412 061542 062544
0000040 063546 060412 061542 062544 063546 005150 061141 062143
0000060 063145 064147 005151 061141 062143 063145 064147 065151
0000100 060412 061542 062544 063546 064550 065552 060412 061542
0000120 062544 063546 064550 065552 005154
0000132

[root@localhost ~]#

```

本例使用指令“od abc”以八进制的形式计算文件abc的字码。

◎ 范例 466 ◎ 按十进制计算字码。

```

[root@localhost ~]# cd -A d abc //以十进制计算字码
0000000 005141 061141 060412 061542 060412 061542 005144 061141
0000016 062143 005145 061141 062143 063145 060412 061542 062544
0000032 063546 060412 061542 062544 063546 005150 061141 062143
0000048 063145 064147 005151 061141 062143 063145 064147 065151
0000064 060412 061542 062544 063546 064550 065552 060412 061542
0000080 062544 063546 064550 065552 005154
0000090

[root@localhost ~]#

```

本例使用指令“od -A d abc”以十进制计算文件abc的字码。

◎ 范例 467 ◎ 以名称字符显示文件内容。

```

[root@localhost ~]# od -t a abc //以名称字符显示文件内容
0000000 a n l a b n l a b c n l a b c d n l a b
0000020 c d e n l a b c d e f n l a b c d e
0000040 f g n l a b c d e f g h n l a b c d

```

## 5.40 od 指令：输出文件内容

```
0000060 e f g h i nl a b c d e f g h i j
0000100 nl a b c d e f g h i j k nl a b c
0000120 d e f g h i j k l nl
0000132
```

[root@localhost ~]#

本例使用指令“od -t a abc”以名称字符的形式显示文件abc的内容。

◎ 范例 468 ◎ 以 ASCII 码的形式显示文件内容。

```
[root@localhost ~]# od -t c abc //以ASCII的形式显示文件内容
0000000 a \n a b \n a b c \n a b c d \n a b
0000020 c d e \n a b c d e f \n a b c d e
0000040 f g \n a b c d e f g h \n a b c d
0000060 e f g h i \n a b c d e f g h i j
0000100 \n a b c d e f g h i j k \n a b c
0000120 d e f g h i j k l \n
0000132
```

[root@localhost ~]#

本例使用指令“od -t c abc”以 ASCII 的形式显示文件abc的内容。

◎ 范例 469 ◎ 以十六进制的形式显示文件内容。

```
[root@localhost ~]# od -t x abc //以十六进制的形式显示文件内容
0000000 62610a61 6362610a 6362610a 62610a64
0000020 0a656463 64636261 610a6665 65646362
0000040 610a6766 65646362 0a686766 64636261
0000060 68676665 62610a69 66656463 6a696867
0000100 6362610a 67666564 6b6a6968 6362610a
0000120 67666564 6b6a6968 00000a6c
0000132
```

[root@localhost ~]#

本例使用指令“od -t x abc”以十六进制的形式显示文件abc的内容。

◎ 范例 470 ◎ 以十进制的形式显示文件内容。

```
[root@localhost ~]# od -t d abc //以十进制的形式显示文件内容
0000000 1650526817 1667391754 1667391754 1650526820
0000020 174416995 1684234849 1628071525 1701077858
0000040 1628071782 1701077858 174614374 1684234849
0000060 1751606885 1650526825 1717920867 1785292903
0000100 1667391754 1734763876 1802135912 1667391754
0000120 1734763876 1802135912 2668
```



0000132

[root@localhost ~]#

本例使用指令“od -t d abc”以十进制的形式显示文件abc的内容。

◎ 范例 471 ◎ 以浮点的形式显示文件内容。

[root@localhost ~]# od -t f abc //以浮点的形式显示文件内容

0000000 5.548951967018402e+170 7.850410473011620e+165

0000020 3.835460019933841e+175 2.643788621179123e+180

0000040 2.643788621179241e+180 3.835460020072590e+175

0000060 7.850445758610149e+165 3.983039467726032e+204

0000100 1.247323092606656e+190 5.548952129638052e+170

0000120 2.713464665207136e+209 1.318167143104446e-320

0000132

[root@localhost ~]#

本例使用指令“od -t f abc”以浮点的形式显示文件abc的内容。



cat

## 5.41 paste 指令：合并文件的列



paste[必要参数][选择性参数][文件]



paste 指令合并指定文件的列，然后将内容输出到标准输出。如果没有指定文件，将从标准输入输入内容。



必要参数的具体说明如下：

| 参数 | 功能       |
|----|----------|
| -s | 一次处理一个文件 |

选择性参数的具体说明如下：

| 参数       | 功能      |
|----------|---------|
| -d<字符>   | 指定间隔的字符 |
| -help    | 帮助信息    |
| -version | 版本信息    |

## 5.41 paste 指令：合并文件的行



## ○范例 472 ○ 合并文件的行。

[root@localhost ~]# cat abc //显示文件 abc 的内容

```
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefgij
```

[root@localhost ~]# cat 123 //显示文件 123 的内容

```
0000000000000000
1111111111111111
2222222222222222
3333333333333333
4444444444444444
5555555555555555
6666666666666666
7777777777777777
8888888888888888
9999999999999999
```

[root@localhost ~]# paste 123 abc //合并文件的行

```
0000000000000000 a
1111111111111111 ab
2222222222222222 abc
3333333333333333 abcd
4444444444444444 abcde
5555555555555555 abcdef
6666666666666666 abcdefg
7777777777777777 abcdefgh
8888888888888888 abcdefghi
9999999999999999 abcdefgij
```

[root@localhost ~]# paste abc 123 //合并文件的行

```
a 0000000000000000
ab 1111111111111111
abc 2222222222222222
```



```
abcd 33333333333333
abcde 44444444444444
abcdef 55555555555555
abcdefg 66666666666666
abcdefgh 77777777777777
abcdefghi 88888888888888
abcdefhij 99999999999999
[root@localhost ~]#
```

本例使用指令“paste 123 abc”合并文件 123 和文件 abc 的行。

◎ 范例 473 ◎ 合并文件的行，并指定之间的间隔符。

```
[root@localhost ~]# paste -d @ 123 abc //合并文件的行，并指定间隔符
00000000000000@a
11111111111111@ab
22222222222222@abc
33333333333333@abcd
44444444444444@abcde
55555555555555@abcdef
66666666666666@abcdefg
77777777777777@abcdefgh
88888888888888@abcdefghi
99999999999999@abcdefhij
[root@localhost ~]#
```

本例使用指令“paste -d @ 123 abc”合并文件 123 和文件 abc 的行，并指定间隔为字符@，显示的结果如上所示。



cat

## 5.42 patch 指令：修补文件

**语法** patch[必要参数][选择性参数][文件]

**功能说明** patch 指令用来为文件打补丁，可以一次可多个补丁，这样的话可以为系统升级。



必要参数的具体说明如下：

## 5.42 patch 指令：修补文件

| 参数       | 功能            |
|----------|---------------|
| -b       | 备份            |
| -c       | 解释文件          |
| -e       | 解释结果          |
| -f       | 强制执行          |
| -l       | 忽略空格和 [tab] 键 |
| -s       | 不显示处理信息       |
| -binary  | 二进制           |
| -verbose | 运行时显示详细的处理信息  |

选择性参数的具体说明如下：

| 参数     | 功能        |
|--------|-----------|
| -d<目录> | 指定目录      |
| -i<路径> | 修补文件的路径   |
| -v     | 版本信息      |
| -help  | 帮助信息      |
| -o<文件> | 指定输出文件的名称 |

### ◎ 范例 474 ◎ 比较两个文件的不同，并生成补丁。

```
[root@localhost ~]# cat 123 //显示文件 123 的内容
i am from bj
i like sports
espacially basketball and football
my favorite football club is acmilan
i will love it for ever
[root@localhost ~]# cat abc //显示文件 abc 的内容
i am a student
i am from bj
i like sports
espacially basketball and football
my favorite football club is intermilan
```



```
i will love it for ever
[root@localhost ~]# diff -ruN abc 123 >kk.diff //比较文件 abc 和 123, 生成差异文件
[root@lccalhost ~]# cat kk.diff
--- abc 2007-05-15 10:05:28.000000000 +0800
+++ 123 2007-05-15 10:06:08.000000000 +0800
@@ -1,6 +1,5 @@
-i am a student
i am from bj
i like sports
espacially basketball and football
-my favorite football club is intermilan
+my favorite football club is acmilan
i will love it for ever
[root@localhost ~]#
```

本例使用指令“`diff -ruN abc 123 >kk.diff`”比较文件 abc 和文件 123 的不同，并生成差异文件，此差异文件可以作为补丁。

#### ○ 范例 475 ○ 打补丁。

```
[root@localhost ~]# patch -p1 <kk.diff //打补丁
missing header for unified diff at line 3 of patch
can't find file to patch at input line 3
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:

|--- abc 2007-05-15 10:05:28.000000000 +0800
+++ 123 2007-05-15 10:06:08.000000000 +0800
File to patch: 123 //123 为输入的文件
patching file 123
Reversed (or previously applied) patch detected! Assume -R? [n] y
[root@localhost ~]# cat 123
i am a student
i am from bj
i like sports
espacially basketball and football
my favorite football club is intermilan
i will love it for ever
[root@localhost ~]#
```

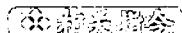
## 5.43 rcp 指令：复制远端主机的文件或目录

本例使用指令“patch -p1 <kk.diff”进行打补丁，运行中要选择打补丁的文件，此处选择的打补丁的文件是 123，如上所示，打完补丁之后，文件 123 的内容和文件 abc 的内容相同，这就是打补丁的过程。

### ○ 范例 476 ○ 为 Linux 系统打补丁。

```
[root@localhost ~]#patch -p0 /home/kk/patch-2.6.8.1 //为系统打补丁
```

本例使用指令“patch -p0 /home/kk/patch-2.6.8.1”为系统打补丁，补丁文件为/home/kk/patch-2.6.8.1。

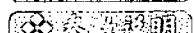


diff

## 5.43 rcp 指令：复制远端主机的文件或目录

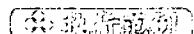
rcp[必要参数][源目录或者文件][目标文件或者目录]

rcp 指令在主机之间复制文件或目录。



必要参数的具体说明如下：

| 参    数 | 功    能                |
|--------|-----------------------|
| -p     | 保留源的属性                |
| -r     | 同时处理文件夹下的文件和子目录下的所有文件 |

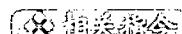


### ○ 范例 477 ○ 复制远端主机的文件。

```
[root@localhost ~]# rcp kk@192.168.88.59:/home/apple /home/kk/
//复制文件
```

```
[root@localhost ~]#
```

本例使用指令“rcp kk@192.168.88.59:/home/apple /home/kk/”复制远端主机 192.168.88.59 上的文件/home/apple 到本地主机/home/kk/目录下。



rsh



大家网  
TopSage.com

## 5.44 rhmask 指令：产生加密文件

rhmask[必要参数][加密文件][源文件][目标文件]

rhmask 指令可以对文件进行加密，增强安全性。

必要参数的具体说明如下：

| 参数 | 功能     |
|----|--------|
| -d | 生成加密文件 |

○范例 478 ○ 加密文件。

```
[root@localhost ~]#rhmask -d jiamifile srcfile desfile
//加密文件
```

本例使用指令“rhmask -d jiamifile srcfile desfile”用文件 jiamifile 加密文件 srcfile，生成目标文件 desfile。

passwd

## 5.45 rm 指令：删除文件或目录

rm[必要参数][选择性参数][目录或者文件]

rm 指令用来删除文件或者目录，如果使用 rm 来删除文件，通常仍可以将文件恢复。如果想保证该文件的内容无法复原，可以考虑使用 shred 指令。

必要参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -d | 删除可能仍有数据的目录    |
| -f | 强制删除           |
| -i | 交互模式           |
| -r | 同时删除该目录下的所有子目录 |
| -v | 运行时显示详细的信息     |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◎执行范例

#### ○ 范例 479 ○ 删 除 文件。

```
[root@localhost kk]# ll //显示文件的详细信息
总用量 7400
-rw-r--r-- 1 root root 8997 5月 22 10:21 111
-rw-r--r-- 1 root root 53 5月 14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月 13 15:53 a.c
-rw-r--r-- 1 root root 97 5月 13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月 13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月 28 14:53 bu
-rw-r--r-- 1 root root 100 5月 14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月 14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月 14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月 13 15:57 Desktop
-rw-r--r-- 1 root root 675 5月 14 09:11 fcntl.c
-rw-r--r-- 1 root root 225 5月 14 09:11 file.c
-rw-r--r-- 1 root root 497 5月 14 09:11 findconf.c
-rw-r--r-- 1 root root 505 5月 14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月 14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月 14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月 14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月 14 09:11 long.c
```



## 第5章 文件管理

```
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
-rw-r--r-- 1 root root 474 5月14 09:11 zuobiao.c
[root@localhost kk]# rm zuobiao.c //删除文件
[root@localhost kk]# ll //显示文件
总用量 7392
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
//此处省略了部分结果
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]#
```

本例使用指令“rm zuobiao.c”删除文件 zuobiao.c。

### ○范例 480 ○ 删除文件夹。

```
[root@localhost kk]# ll //显示文件信息
总占用量 7400
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月28 14:53 bu
-rw-r--r-- 1 root root 100 5月14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月13 15:57 Desktop
-rw-r--r-- 1 root root 675 5月14 09:11 fcntl.c
-rw-r--r-- 1 root root 225 5月14 09:11 file.c
-rw-r--r-- 1 root root 497 5月14 09:11 findconf.c
-rw-r--r-- 1 root root 505 5月14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月14 09:11 getsec.c
```

## 5.45 rm 指令：删除文件或目录

```

-rw-r--r-- 1 root root 115 5月14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月14 09:11 long.c
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
drwxr-xr-x 2 root root 4096 5月28 16:03 temp
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpvl2e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]# rm -rf temp/ //删除目录temp/
[root@localhost kk]# ll //显示文件信息
总占用量 7392
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月28 14:53 bu
-rw-r--r-- 1 root root 100 5月14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月13 15:57 Desktop
-rw-r--r-- 1 root root 675 5月14 09:11 fcntl.c
-rw-r--r-- 1 root root 225 5月14 09:11 file.c
-rw-r--r-- 1 root root 497 5月14 09:11 findconf.c
-rw-r--r-- 1 root root 505 5月14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月14 09:11 long.c
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpvl2e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]#

```



本例使用指令“rm -rf temp/”删除当前文件夹下的目录temp/。

◎ 范例 481 ◎ 利用通配符删除文件。

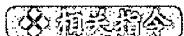
```
[root@localhost kk]# ll //显示文件信息
总占用量 7392
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月28 14:53 bu
-rw-r--r-- 1 root root 100 5月14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月13 15:57 Desktop
-rw-r--r-- 1 root root 675 5月14 09:11 fcntl.c
-rw-r--r-- 1 root root 225 5月14 09:11 file.c
-rw-r--r-- 1 root root 497 5月14 09:11 findconf.c
-rw-r--r-- 1 root root 505 5月14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月14 09:11 long.c
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]# rm -f f*
//删除文件
[root@localhost kk]# ll //显示文件信息
总占用量 7368
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月28 14:53 bu
```

## 5.46 rmdir 指令：删除目录

第  
5  
章

```
-rw-r--r-- 1 root root 100 5月14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月13 15:57 Desktop
-rw-r--r-- 1 root root 505 5月14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月14 09:11 long.c
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]#
```

本例使用指令“rm -f f\*”删除所有以字母“f”打头的文件。

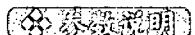


chattr、shred

## 5.46 rmdir 指令：删除目录

**rmdir[必要参数][选择性参数][目录]**

rm 指令用来删除空目录，如果目录非空，将出现错误。



必要参数的具体说明如下：

| 参    数                           | 功    能                  |
|----------------------------------|-------------------------|
| <b>-ignore-fail-on-non-empty</b> | 忽略任何因目录仍有数据而造成的错误       |
| <b>-p</b>                        | 删除目录，然后尝试删除指定路径中的所有上层指令 |
| <b>-v</b>                        | 运行时显示详细的处理信息            |



选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### 执行范围

#### ◎ 范例 482 ◎ 删除目录。

```
[root@localhost kk]# ll //显示文件
总占用量 7376
-rw-r--r-- 1 root root 8997 5月 22 10:21 111
-rw-r--r-- 1 root root 53 5月 14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月 13 15:53 a.c
-rw-r--r-- 1 root root 97 5月 13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月 13 15:46 a.c@
-rw-r--r-- 1 root root 5867520 5月 28 14:53 bu
-rw-r--r-- 1 root root 100 5月 14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月 14 09:11 cmh.c .
-rw-r--r-- 1 root root 7376 5月 14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月 13 15:57 Desktop
-rw-r--r-- 1 root root 505 5月 14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月 14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月 14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月 14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月 14 09:11 long.c
-rw-r--r-- 1 root root 126 5月 14 09:11 max.c
-rw-r--r-- 1 root root 151 5月 14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月 14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月 14 09:11 strrchr.c
drwxr-xr-x 2 root root 4096 5月 28 16:29 temp
-rw-r--r-- 1 root root 1563 5月 22 10:09 test
-rw-r--r-- 1 root root 1471488 5月 3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月 14 09:11 wenjian.c
[root@localhost kk]# rmdir -v temp/ //删除目录
rmdir: 正在删除目录 temp/
[root@localhost kk]# ll //显示文件信息
总占用量 7368
```

## 5.47 scp 指令：远程复制文件

第5章

```
-rw-r--r-- 1 root root 8997 5月22 10:21 111
-rw-r--r-- 1 root root 53 5月14 09:40 abc.c
-rw-r--r-- 1 root root 97 5月13 15:53 a.c
-rw-r--r-- 1 root root 97 5月13 15:42 a.c~
-rw-r--r-- 1 root root 97 5月13 15:46 a.cé
-rw-r--r-- 1 root root 5867520 5月28 14:53 bu
-rw-r--r-- 1 root root 100 5月14 09:40 chch.c
-rw-r--r-- 1 root root 205 5月14 09:11 cmh.c
-rw-r--r-- 1 root root 7376 5月14 09:11 D.c
drwxr-xr-x 2 root root 4096 5月13 15:57 Desktop
-rw-r--r-- 1 root root 505 5月14 09:11 getcurtime.c
-rw-r--r-- 1 root root 296 5月14 09:11 getsec.c
-rw-r--r-- 1 root root 115 5月14 09:11 gettimeofday.c
-rw-r--r-- 1 root root 513 5月14 09:11 getusec.c
-rw-r--r-- 1 root root 92 5月14 09:11 long.c
-rw-r--r-- 1 root root 126 5月14 09:11 max.c
-rw-r--r-- 1 root root 151 5月14 09:11 rand.c
-rw-r--r-- 1 root root 101 5月14 09:11 sprintf.c
-rw-r--r-- 1 root root 131 5月14 09:11 strrchr.c
-rw-r--r-- 1 root root 1563 5月22 10:09 test
-rw-r--r-- 1 root root 1471488 5月3 16:10 unpv12e.tar
-rw-r--r-- 1 root root 142 5月14 09:11 wenjian.c
[root@localhost kk]#
```

本例使用指令“rmkdir -v temp/”删除空目录 temp/，注意，如果删除的对象不是空目录，将会出现错误信息。

 **相关命令**

rm

## 5.47 scp 指令：远程复制文件

 **语法** scp[必要参数][选择性参数][主机 1][文件][主机 2][文件]

 **功能说明** scp 指令用于在网络中的不同主机之间复制文件。

 **参数说明**

必要参数的具体说明如下：



| 参数 | 功能              |
|----|-----------------|
| -l | 强制 scp 指令使用协议 1 |
| -2 | 强制 scp 指令使用协议 2 |
| -4 | 强制使用 IPv4 协议    |
| -6 | 强制使用 IPv6 协议    |
| -f | 后台执行            |
| -N | 测试              |
| -n | 重定位             |
| -q | 不显示处理信息         |
| -v | 运行时显示详细的处理信息    |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -b<地址>   | 指定地址   |
| -F<文件>   | 指定配置文件 |
| -i<认证文件> | 指定认证文件 |
| -l<用户账号> | 指定用户账号 |
| -p<端口号>  | 指定端口号  |
| -V       | 版本信息   |

### ④ 支持范围

○ 范例 483 ○ 复制文件到远程主机。

```
[root@localhost ~]# scp -v root@192.168.88.76:/root/123 kk@192.168.88.59:/home/kk/
//复制文件到远程主机
Executing: /usr/bin/ssh -v -x -o'ClearAllForwardings yes' -t -l root
192.168.88.76 scp -v /root/123 'kk@192.168.88.59:/home/kk/'
OpenSSH_4.0p1, OpenSSL 0.9.7f 22 Mar 2005
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to 192.168.88.76 [192.168.88.76] port 22.
```

## 5.48 slocate 指令：查找文件或目录

```

debug1: Connection established.
debug1: permanently_set_uid: 0/0
debug1: identity file /root/.ssh/identity type -1
debug1: identity file /root/.ssh/id_rsa type -1
debug1: identity file /root/.ssh/id_dsa type -1
debug1: Remote protocol version 2.0, remote software version OpenSSH_4.0
debug1: match: OpenSSH_4.0 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_4.0
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: server->client aes128-cbc hmac-md5 none
debug1: kex: client->server aes128-cbc hmac-md5 none
debug1: SSH2_MSG_KEX_DH_GEX_REQUEST(1024<1024<8192) sent
//此处省略了部分结果

```

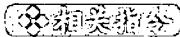
Connection to 192.168.88.76 closed.

```

debug1: Transferred: stdin 0, stdout 0, stderr 37 bytes in 9.0 seconds
debug1: Bytes per second: stdin 0.0, stdout 0.0, stderr 4.1
debug1: Exit status 0
[root@localhost ~]#

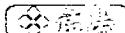
```

本例使用指令“`scp -v root@192.168.88.76:/root/123 kk@192.168.88.59:/home/kk/`”从本地主机 192.168.88.76 复制文件/root/123 到远端主机 192.168.88.59 的目录/home/kk/。

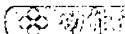


`scp`、`sftp`、`ssh`、`ssh-add`、`ssh-agent`、`ssh-keygen`、`ssh_config`、`sshd`

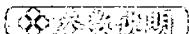
## 5.48 slocate 指令：查找文件或目录



**slocate[必要参数][选择性参数][文件]**



**slocate** 指令用来查找文件，它可以自己建一个数据库，直接在自己的数据库里进行查找，从而提高查找速度。



必要参数的具体说明如下：



| 参数 | 功能           |
|----|--------------|
| -I | 对大小写不敏感      |
| -u | 更新数据库        |
| -q | 不显示处理信息      |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功能      |
|----------|---------|
| -n<行数>   | 显示指定行数  |
| -d<目录>   | 指定数据库目录 |
| -help    | 帮助信息    |
| -version | 版本信息    |

### 范例 484 ◎ 查找文件。

```
-sh-2.05b# slocate today //查找文件
/usr/share/doc/HTML/en/korganizer/i_go_to_today.png
/usr/share/doc/HTML/en/kdevelop/reference/C/CONTRIB/SNIP/todaybak.c
/usr/share/apps/korganizer/icons/crystalsvg/16x16/actions/today.png
/usr/share/apps/korganizer/icons/crystalsvg/22x22/actions/today.png
/usr/share/apps/korganizer/icons/crystalsvg/32x32/actions/today.png
/usr/share/icons/Bluecurve/16x16/actions/today.png
/usr/share/icons/Bluecurve/22x22/actions/today.png
/usr/share/icons/ikons/16x16/actions/today.png
/usr/share/icons/ikons/22x22/actions/today.png
/usr/share/icons/kdeclassic/16x16/actions/today.png
/usr/share/icons/kdeclassic/22x22/actions/today.png
/usr/share/icons/kdeclassic/32x32/actions/today.png
/usr/share/images/evolution/evolution-today-mini.png
/usr/share/images/evolution/evolution-today.png
-sh-2.05b#
```

本例使用指令“slocate today”查找含有字符串 today 的文件。

### 相关指令

locate

## 5.49 split 指令：切割文件

**语法** split[必要参数][选择性参数][要切割的文件][输出文件名]

**功能说明** split 指令将指定的文件切割成小的文件。

### 参数说明

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -d | 数字表示切割后的小文件  |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数       | 功能        |
|----------|-----------|
| -b<字节>   | 每个分割文件的大小 |
| -c<字节>   | 输出的每行的大小  |
| -l<行数>   | 指定行数      |
| -help    | 帮助信息      |
| -version | 版本信息      |

### 执行范例

○ 范例 485 ○ 以指定行数分割文件。

```
[root@localhost temp]# ll //文件列表
总用量 1584
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]# split -l 300 xen-syms //分割文件
[root@localhost temp]# ll //文件列表
总用量 3376
```



## 第5章 文件管理

```

-rw-r--r-- 1 root root 225361 5月 29 18:05 xaa
-rw-r--r-- 1 root root 80344 5月 29 18:05 xab
-rw-r--r-- 1 root root 14084 5月 29 18:05 xac
-rw-r--r-- 1 root root 110711 5月 29 18:05 xad
-rw-r--r-- 1 root root 58369 5月 29 18:05 xae
-rw-r--r-- 1 root root 31677 5月 29 18:05 xaf
-rw-r--r-- 1 root root 39869 5月 29 18:05 xag
-rw-r--r-- 1 root root 36620 5月 29 18:05 xah
-rw-r--r-- 1 root root 36082 5月 29 18:05 xai
-rw-r--r-- 1 root root 35826 5月 29 18:05 xaj
 //此处省略部分结果
-rw-r--r-- 1 root root 72253 5月 29 18:05 xbb
-rw-r--r-- 1 root root 57360 5月 29 18:05 xbc
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]#

```

本例使用指令“split -l 300 xen-syms”分割文件 xen-syms，每个文件最多有 300 行，输出结果如上所示。

### ○范例 486 ○ 按字节数分割文件。

```

[root@localhost temp]# ll //显示文件列表
总用量 1584
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]# split -b 30k xen-syms //分割文件
[root@localhost temp]# ll //显示文件列表
总用量 3476
-rw-r--r-- 1 root root 30720 5月 29 20:17 xaa
-rw-r--r-- 1 root root 30720 5月 29 20:17 xab
-rw-r--r-- 1 root root 30720 5月 29 20:17 xac
-rw-r--r-- 1 root root 30720 5月 29 20:17 xad
-rw-r--r-- 1 root root 30720 5月 29 20:17 xae
-rw-r--r-- 1 root root 30720 5月 29 20:17 xaf
-rw-r--r-- 1 root root 30720 5月 29 20:17 xag
-rw-r--r-- 1 root root 30720 5月 29 20:17 xah
 //此处省略了部分结果
-rw-r--r-- 1 root root 30720 5月 29 20:17 xbv
-rw-r--r-- 1 root root 30720 5月 29 20:17 xbw
-rw-r--r-- 1 root root 30720 5月 29 20:17 xbx
-rw-r--r-- 1 root root 30720 5月 29 20:17 xby
-rw-r--r-- 1 root root 30720 5月 29 20:17 xbz
-rw-r--r-- 1 root root 12612 5月 29 20:17 xca

```

## 5.50 stat 指令：显示 inode 内容

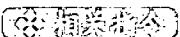
```
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]#
```

本例使用指令“split -b 30k xen-syms”按照字节数分割文件 xen-syms，每个文件的大小为 30KB。

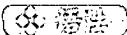
## ◎ 范例 487 ◎ 以数字显示分割的小文件的序号。

```
[root@localhost temp]# ll //显示文件列表
总用量 1584
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]# split -b 30k -d xen-syms //分割文件
[root@localhost temp]# ll //显示文件列表
总用量 3476
-rw-r--r-- 1 root root 30720 5月 29 20:20 x00
-rw-r--r-- 1 root root 30720 5月 29 20:20 x01
-rw-r--r-- 1 root root 30720 5月 29 20:20 x02
-rw-r--r-- 1 root root 30720 5月 29 20:20 x03
-rw-r--r-- 1 root root 30720 5月 29 20:20 x04
-rw-r--r-- 1 root root 30720 5月 29 20:20 x05
-rw-r--r-- 1 root root 30720 5月 29 20:20 x06
-rw-r--r-- 1 root root 30720 5月 29 20:20 x07
//此处省略了部分结果
-rw-r--r-- 1 root root 30720 5月 29 20:20 x47
-rw-r--r-- 1 root root 30720 5月 29 20:20 x48
-rw-r--r-- 1 root root 30720 5月 29 20:20 x49
-rw-r--r-- 1 root root 30720 5月 29 20:20 x50
-rw-r--r-- 1 root root 30720 5月 29 20:20 x51
-rw-r--r-- 1 root root 12612 5月 29 20:20 x52
-rw-r--r-- 1 root root 1610052 5月 29 18:03 xen-syms
[root@localhost temp]#
```

本例使用指令“split -b 30k -d xen-syms”分割文件 xen-syms，分割出的小文件的序号以数字排序，而不是以字母排序。



cut

**5.50****stat 指令：显示 inode 内容**

stat[必要参数][选择性参数][文件]



## 功能说明

stat 指令用来显示文件的详细信息。

## 参数说明

必要参数的具体说明如下：

| 参数 | 功能     |
|----|--------|
| -f | 显示文件系统 |
| -L | 显示符号链接 |
| -t | 简单模式   |

选择性参数的具体说明如下：

| 参数        | 功能   |
|-----------|------|
| --help    | 帮助信息 |
| --version | 版本信息 |

## 执行范例

## 范例 488 显示文件信息。

```
-sh-2.05b# ll //显示文件列表
总用量 6816
-rwxrwxrwx 1 root kk 2398 5月 8 20:15 1.log
-rw-r--r-- 1 root root 28 5月 7 16:05 abc
-rw-r--r-- 1 root root 28 5月 14 20:43 abc.bak
-rw-r--r-- 1 root root 176 5月 3 22:50 a.c
-rw-r--r-- 1 root root 1516 4月 20 19:31 anaconda-ks.cfg
-rw-r--r-- 1 root root 80 5月 28 23:11 examplesed
-rw-r--r-- 1 root root 27023 5月 3 22:49 install.log
-rw-r--r-- 1 root root 27035 5月 3 22:46 install.log.bak
-rw-r--r-- 1 root root 4138 4月 20 19:28 install.log.syslog
-rw-r--r-- 1 root root 6038 5月 9 22:59 ls.man.txt
-rw-r--r-- 1 root root 7598 5月 9 23:07 ls.man.txt2
-rw-r--r-- 1 root root 183833 4月 23 15:59 magick.miff
-rw----- 1 root root 102 5月 25 15:46 nohup.out
-rw-r--r-- 1 root root 6649992 5月 28 20:37 RealPlayer10GOLD.rpm
-rw-r--r-- 1 root root 150 5月 29 15:23 temp
-rw-r--r-- 1 root root 75 5月 29 15:29 templ
-rw-r--r-- 1 root root 45 5月 29 15:29 temp2
```

## 5.51 sum 指令：计算文件的校验码

```
-rw-r--r-- 1 root root 30 5月 29 15:29 temp3
-sh-2.05# stat RealPlayer10GOLD.rpm //显示文件信息
 File: 'RealPlayer10GOLD.rpm'
 Size: 6649992 Blocks: 13016 IO Block: 4096 Regular File
Device: 803h/2051d Inode: 294206 Links: 1
Access: (0644/-rw-r--r--)Uid: (0/ root) Gid: (0/ root)
Access: 2007-05-28 20:38:03.000000000 +0800
Modify: 2007-05-28 20:37:19.000000000 +0800
Change: 2007-05-28 20:37:19.000000000 +0800
-sh-2.05#
```

本例使用指令“stat RealPlayer10GOLD.rpm”显示文件 RealPlayer10GOLD.rpm 的详细信息。

◎ 范例 489 ◎ 显示文件系统信息。

```
-sh-2.05# stat /dev/hda //显示文件系统的信息
 File: '/dev/hda'
 Size: 0 Blocks: 0 IO Block: 4096 Block Device
Device: 803h/2051d Inode: 65355 Links: 1 Device type: 3,0
Access: (0660;brw-rw----)Uid: (0/ root) Gid: (6/ disk)
Access: 2003-01-30 18:24:36.000000000 +0800
Modify: 2003-01-30 18:24:36.000000000 +0800
Change: 2007-04-20 18:48:23.000000000 +0800
-sh-2.05#
```

本例使用指令“stat /dev/had”显示文件系统/dev/hda 的详细信息。



ls

## 5.51 sum 指令：计算文件的校验码

**sum[必要参数][选择性参数][文件]**

sum 指令可以用来计算文件的校验码，同时可以显示块数。

必要参数的具体说明如下：



| 参数 | 功能         |
|----|------------|
| -r | BCD 兼容的演算码 |
| -s | 系统 V 演算码   |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

#### ◆ 功能说明

○ 范例 490 ○ 计算文件校验码。

```
sh-2.05b# sum install.log //计算文件的校验码
08354 27
sh-2.05b#
```

本例使用指令“sum install.log”计算文件 install.log 的校验码，得到的“08354”即为校验码，“27”为块数。

#### ◆ 相关指令

cksum、md5sum

## 5.52

## tee 指令：读取标准输入到标准输出 并可保存为文件

#### ◆ 基础

tee[必要参数][选择性参数][文件]

◆ 功能说明 tee 指令用来将标准输入的内容输出到标准输出并可以保存为文件。

#### ◆ 基础说明

必要参数的具体说明如下：

| 参数 | 功能    |
|----|-------|
| -a | 附加到文件 |
| -i | 无视中断  |

## 5.52 tee 指令：读取标准输入到标准输出并可保存为文件

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### ◎ 实例范例

#### ○ 范例 491 ○ 将输入存成文件。

```
[root@localhost ~]# tee inter //写文件
i love intermilan
i love intermilan
i love recoba
i love recoba
i love figo
i love figo
the city of milan have two great teams
the city of milan have two great teams
intermilan and intermilan young
intermilan and intermilan young
[root@localhost ~]# cat inter //显示文件内容
i love intermilan
i love recoba
i love figo
the city of milan have two great teams
intermilan and intermilan young
[root@localhost ~]#
```

本例使用指令“tee inter”将标准输入的内容写到文件 inter。

#### ○ 范例 492 ○ 文件备份。

```
[root@localhost ~]# cat inter|tee inter_backup1 inter_backup2
 //备份文件
i love intermilan
i love recoba
i love figo
the city of milan have two great teams
intermilan and intermilan young
[root@localhost ~]# cat inter_backup1 //显示文件内容
i love intermilan
```



```
i love recoba
i love figo .
the city of milan have two great teams
intermilan and intermilan young
[root@localhost ~]# cat inter_backup2
//显示文件内容
i love intermilan
i love racoba
i love figo
the city of milan have two great teams
intermilan and intermilan young
[root@localhost ~]#
```

本例使用指令“cat inter>tee inter\_backup1 inter\_backup2”显示文件 inter 的内容，并将标准输出保存到文件 inter\_backup1 和 inter\_backup2。

#### ◆◆相关指令◆◆

cat

## 5.53

## tmpwatch 指令：删除临时文件

#### ◆◆基本用法◆◆

tmpwatch[必要参数][超时时间][目录]

#### ◆◆功能说明◆◆

tmpwatch 指令可以用来删除指定时间内没有使用过的临时文件。

#### ◆◆参数说明◆◆

必要参数的具体说明如下：

| 参 数   | 功 能       |
|-------|-----------|
| -3    | 符号链接文件也删除 |
| -f    | 强制删除      |
| -q    | 不显示处理信息   |
| -v    | 显示详细的处理信息 |
| -test | 测试        |

#### ◆◆执行范例◆◆

- 范例 493 ○ 删 除临时文件。

## 5.54 touch 指令：更新文件或目录时间

```
[root@localhost ~]# tmpwatch 200 /tmp/
[root@localhost ~]#
```

本例使用指令“tmpwatch 200 /tmp/”删除/tmp 目录下 200 小时内没有使用的文件。



cron、ls、rm、fuser

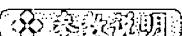
## 5.54 touch 指令：更新文件或目录时间



**touch[必要参数][选择性参数][目录或者文件]**



使用 touch 指令更新指定目录或者文件的存取时间为当前时间。

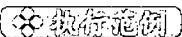


必要参数的具体说明如下：

| 参    数 | 功    能   |
|--------|----------|
| -a     | 更改存取时间   |
| -c     | 不新建文件    |
| -f     | BSD 版本兼容 |
| -m     | 只更新变动时间  |

选择性参数的具体说明如下：

| 参    数    | 功    能         |
|-----------|----------------|
| -d<日期>    | 使用指定日期，而不是当前时间 |
| -t<日期>    | 等同于“-d<日期>”    |
| --help    | 帮助信息           |
| --version | 版本信息           |



○ 范例 494 ○ 更新文件时间。



## 第5章 文件管理

```
[root@localhost other]# ll //显示文件信息
总用量 40
-rw-r--r-- 1 root root 10659 5月 9 09:37 111
-rw-r--r-- 1 root root 97 5月 15 21:11 abc.dvi
drwxr-xr-x 2 root root 4096 3月 30 14:30 DJ
drwxr-xr-x 9 root root 4096 5月 3 17:51 helloworld
[root@localhost other]# touch 111 //更新文件存取时间
[root@localhost other]# ll //显示文件信息
总用量 40
-rw-r--r-- 1 root root 10659 5月 30 17:22 111
-rw-r--r-- 1 root root 97 5月 15 21:11 abc.dvi
drwxr-xr-x 2 root root 4096 3月 30 14:30 DJ
drwxr-xr-x 9 root root 4096 5月 3 17:51 helloworld
[root@localhost other]#
```

本例使用指令“touch 111”更新文件的存取时间。

◎ 范例 495 ◎ 更新文件的存取时间为指定时间。

```
[root@localhost other]# touch 111 -t 198208031500.00//更新文件存取时间
[root@localhost other]# ll //显示文件信息
总用量 40
-rw-r--r-- 1 root root 10659 1982-08-03 111
-rw-r--r-- 1 root root 97 5月 15 21:11 abc.dvi
drwxr-xr-x 2 root root 4096 3月 30 14:30 DJ
drwxr-xr-x 9 root root 4096 5月 3 17:51 helloworld
[root@localhost other]#
```

本例使用指令“touch 111 -t 198208031500.00”更新文件的存取时间为1982年8月3日下午15:00点整。



ls

### 5.55

### tree 指令：以树状图显示目录内容



**tree[必要参数][选择性参数][目录]**



tree 指令将列出指定目录下的所有文件，包括子目

## 5.55 tree 指令：以树状图显示目录内容

录里的所有文件。

### ◎ 参数说明

必要参数的具体说明如下：

| 参数 | 功能               |
|----|------------------|
| -a | 显示所有文件和目录        |
| -A | 以 ASCII 绘图符显示树状图 |

选择性参数的具体说明如下：

| 参数     | 功能              |
|--------|-----------------|
| -l<格式> | 不显示与格式匹配的文件或者目录 |

### ◎ 操作范例

#### ◎ 范例 496 ◎ 显示目录树。

```
[root@localhost ~]# tree /root //显示目录树
/root
|-- 1.log
|-- 1.log.new
|-- 123
|-- 123.pcx
|-- 2.log
|-- 456
|-- 7.log
|-- DEADJOE
|-- Desktop
|-- System.map-2.6.11-1.1369_FC4
|-- abc
|-- anaconda-ks.cfg
|-- config-2.6.11-1.1369_FC4
|-- ddd
|-- examplesed
|-- file1
|-- file2
|-- gmon.out
|-- grub
| |-- device.map
```



```

| |-- e2fs_stage1_5
| |-- fat_stage1_5
| |-- ffs_stage1_5
| |-- grub.conf
| |-- iso9660_stage1_5
| |-- jfs_stage1_5
| |-- menu.lst -> ./grub.conf
| |-- minix_stage1_5
| |-- reiserfs_stage1_5
| |-- splash.xpm.gz
| |-- stage1
| |-- stage2
| |-- ufs2_stage1_5
| |-- vstafs_stage1_5
| '-- xfs_stage1_5
|-- initrd-2.6.11-1.1369_FC4.img
|-- install.log
|-- install.log.new
|-- install.log.syslog
|-- inter
|-- inter_backup1
|-- inter_backup2
|-- lib
|-- lib.a
|-- lib.new
|-- lost+found
|-- memtest86+-1.55.1
|-- nc
|-- nohup.out
|-- restoresymtable
|-- sushe
|-- temp
| '-- DEADJOE
|-- test
|-- uhvbo8ac.pfa
|-- uhvbo8ac.pfb
|-- vmlinuz-2.6.11-1.1369_FC4
|-- workspace
|-- xen-syms

```

## 5.55 tree 指令：以树状图显示目录内容

```
!-- xen.gz
!-- xuehao
5 directories, 56 files
[root@localhost ~]#
```

本例使用指令“tree /root”显示/root 目录的目录树。

○ 范例 497 ○ 以 ASCII 字符显示目录树。

```
[root@localhost ~]# tree -A /root //显示目录树
/root
├── 1.log
├── 1.log.new
├── 123
├── 123.pcx
├── 2.log
├── 456
├── 7.log
├── DEADJOE
├── Desktop
├── System.map-2.6.11-1.1369_FC4
├── abc
├── anaconda-ks.cfg
├── config-2.6.11-1.1369_FC4
├── ddd
├── examplesed
├── file1
├── file2
├── gmon.out
└── grub
 ├── device.map
 ├── e2fs_stage1_5
 ├── fat_stage1_5
 ├── ffs_stage1_5
 ├── grub.conf
 ├── iso9660_stage1_5
 ├── jfs_stage1_5
 ├── menu.lst -> ./grub.conf
 ├── minix_stage1_5
 ├── reiserfs_stage1_5
 ├── splash.xpm.gz
 └── stage1
```



## 第5章 文件管理

```

 └── stage2
 └── ufs2_stagel_5
 └── vstafs_stagel_5
 └── xfs_stagel_5
 └── initrd-2.6.11-1.1369_FC4.img
 └── install.log
 └── install.log.new
 └── install.log.syslog
 └── inter
 └── inter_backup1
 └── inter_backup2
 └── lib
 └── lib.a
 └── lib.new
 └── lost+found
 └── memtest86+-1.55.1
 └── nc
 └── nohup.out
 └── restoresymtable
 └── sushe
 └── temp
 └── DEADJOE
 └── test
 └── uhvbo8ac.pfa
 └── uhvbo8ac.pfb
 └── vmlinuz-2.6.11-1.1369_FC4
 └── workspace
 └── xen-syms
 └── xen.gz
 └── xuehao

5 directories, 56 files
[root@localhost ~]#

```

本例使用指令“tree -A /root”以 ASCII 绘图字符显示/root 目录的指令树。

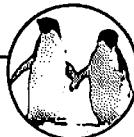
- 范例 498 ○ 显示完全目录树。

```
[root@localhost ~]# tree -a /root //显示完全目录树
```

```
!-- grub
```

## 5.55 tree 指令：以树状图显示目录内容

```
|--- device.map
|--- e2fs_stage1_5
|--- fat_stage1_5
|--- ffs_stage1_5
|--- grub.conf
|--- iso9660_stage1_5
|--- jfs_stage1_5
|--- menu.lst -> ./grub.conf
|--- minix_stage1_5
|--- reiserfs_stage1_5
|--- splash.xpm.gz
|--- stage1
|--- stage2
|--- ufs2_stage1_5
|--- vstafs_stage1_5
| '-- xfs_stage1_5
|-- initrd-2.6.11-1.1369_FC4.img
|-- install.log
|-- install.log.new
|-- install.log.syslog
|-- inter
|--- inter_backup1
|--- inter_backup2
|-- lib
|-- lib.a
|-- lib.new
|-- lost+found
|-- memtest86+-1.55.1
|-- nc
|-- nohup.out
|-- restoresymtable
|-- sushe
|-- temp
| '-- DEADJOE
|-- test
|-- uhvbo8ac.pfa
|-- uhvbo8ac.pfb
|-- vmlinuz-2.6.11-1.1369_FC4
```



```

|-- workspace
| |-- .metadata
| | |-- .lock
| | |-- .plugins
| | | |-- org.eclipse.core.resources
| | | | |-- .root
| | | | '-- .safetable
| | | '-- org.eclipse.debug.core
| | '-- org.eclipse.debug.ui
| | '-- org.eclipse.ui
| | '-- org.eclipse.ui.ide
| '-- org.eclipse.ui.workbench
|-- version.ini
|-- xen-syms
|-- xen.gz
'-- xuehao
246 directories, 1180 files
[root@localhost ~]#

```

本例使用指令“tree -a /root”显示目录/root 下的详细的目录树，包括/root 目录下的文件和/root 目录下的子目录的所有文件。由于内容较多，前边部分省略。

### 相关指令

dircolors、ls、find

## 5.56

### umask 指令：指定在建立文件时预设的权限掩码

#### 命令

**umask[必要参数][权限掩码]**

#### 功能说明

umask 指令可以用来显示和设置权限掩码。

#### 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能         |
|-----|-------------|
| -S  | 文字的形式显示权限掩码 |

## 5.56 umask 指令：指定在建立文件时预设的权限掩码

### ◎ 执行范例

- ◎ 范例 499 ◎ 显示当前用户的权限掩码。

```
[root@localhost ~]# umask //显示权限掩码
0022
[root@localhost ~]# umask -S //显示权限掩码
u=rwx,g=rx,o=rx
[root@localhost ~]#
```

本例使用指令 umask 显示权限掩码，并使用指令“umask -S”显示权限掩码。

- ◎ 范例 500 ◎ 设置权限掩码。

```
[root@localhost ~]# umask 111 //设置权限掩码
[root@localhost ~]# umask //显示权限掩码
0111
[root@localhost ~]# umask -S //显示权限掩码
u=rw,g=rw,o=rw
[root@localhost ~]#
```

本例使用指令“umask 111”设置权限掩码为 111。

### ◎ 相关指令

chmod

<http://www.TopSage.com>

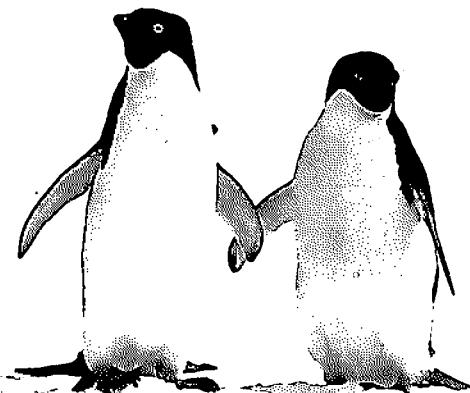




## 第6章

### 备份压缩

备份压缩指令的主要功能是对 Linux 系统下的文件进行压缩和解压缩。在 Linux 系统中，压缩文件主要包括.zip、.tar、.tar.bz2、.tar.gz、.tgz.bz 等格式。对于各种格式，Linux 系统都提供了对应的指令，本章将进行详细介绍。





## 6.1

# ar 指令：建立、修改或从档案文件中提取文件

**命令格式**

ar[-][必要参数][选择性参数][成员文件][count][目标文件]

**功能说明**

ar 指令可以将一组文件合并到一个档案文件中，也可以从建立的档案文件中提取文件。档案文件中的文件的原有属性不变。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功 能           |
|----|---------------|
| -d | 删除档案文件中的文件    |
| -m | 改变档案文件中的文件的次序 |
| -p | 显示档案文件中的文件的内容 |
| -q | 向档案文件追加文件     |
| -r | 替换档案文件中的文件    |
| -t | 显示档案文件中的文件夹   |
| -x | 从档案文件中提取文件    |

选择性参数的具体说明如下：

| 参数    | 功 能                     |
|-------|-------------------------|
| a<文件> | 将文件追加在档案文件中指定的文件之后      |
| b<文件> | 将文件追加在档案文件中指定的文件之前      |
| c     | 创建档案文件                  |
| f     | 截掉过长的档案文件名              |
| i<文件> | 跟“b”参数作用相同              |
| o     | 保留档案文件中文件的信息            |
| s     | 创建档案文件的符号表              |
| S     | 不创建档案文件的符号表             |
| u     | 与“r”一起使用，替换档案文件中已经修改的文件 |

## 6.2 bunzip2 指令：解压缩.bz2 类型的文件

续表

| 参数 | 功能           |
|----|--------------|
| v  | 运行指令时输出详细的信息 |
| V  | 版本信息         |

## 技巧范例

- 范例 501 ○ 编译文件产生目标文件。

```
root@localhost:~# gcc -o a.o a.c //编译文件 a.c
```

用 gcc 编译 C 源文件 a.c 产生目标文件 a.o。

- 范例 502 ○ 编译文件产生目标文件

```
root@localhost:~# gcc -o b.o b.c //编译文件 b.c
```

用 gcc 编译 C 源文件 b.c 产生目标文件 b.o。

- 范例 503 ○ 创建新库并把建立的模块加入。

```
root@localhost:~# ar rv lib.a a.o b.o //创建档案文件
```

a.o

b.o

“ar rv lib.a a.o b.o” 创建了库 lib，它包含两个成员文件 a.o 和 b.o，它同时具有 a.o 和 b.o 模块所具有的功能，进行链接时可以用 “-llib”。

- 范例 504 ○ 显示档案文件的信息。

```
root@localhost:~# ar t lib.a //显示档案文件的信息
```

a.o

b.o

显示 lib 库中的模块，为 a.o 和 b.o。

- 范例 505 ○ 删除档案文件中的文件。

```
root@localhost:~# ar d lib.a a.o //删除档案文件的文件
```

```
root@localhost:~# ar t lib.a
```

b.o

删除库 lib 中的模块 a.o，重新显示库 lib 中的模块，发现只剩下 b.o。

## 相关命令

nm、ranlib

## 6.2

## bunzip2 指令：解压缩.bz2 类型的文件

## 命令

bunzip2[必要参数][选择性参数][文件]



**功能说明** bunzip2 指令相当于“bzip2 -d”，将解压所有指定的文件。不是被 bzip2 创建的文件将被检测到并被忽略，并将产生警告信息。文件被解压后的形式如下。

- filename.bz2 变成 filename。
- filename.bz 变成 filename。
- filename.tbz2 变成 filename.tar。
- filename.tbz 变成 filename.tar。
- anyothername 变成 anyothername.out。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功 能                  |
|----|----------------------|
| -f | 强制解压文件，覆盖当前存在的同名的文件  |
| -k | 解压后保留源文件，默认情况下不保留源文件 |
| -s | 减少程序执行时占用的内存         |
| -v | 运行时输出详细的处理信息         |
| -c | 将解压后的数据直接输出到标准输出     |

选择性参数的具体说明如下：

| 参数 | 功 能    |
|----|--------|
| -l | 显示版本信息 |
| -h | 显示帮助信息 |

**执行范例**

● 范例 506 ● 解压.bz2 文件。

```
[root@localhost root]# bunzip2 abc.bz2 //解压文件 abc.bz2
```

解压之后，将生成文件 abc，原文件 abc.bz2 将消失。

● 范例 507 ● 解压文件，解压后的数据输出到标准输出。

```
[root@localhost root]# bunzip2 -c abc.bz2 //解压文件 abc.bz2 并将解压后的数据输出到标准输出
i am a student
```

## 6.3 bzip2 指令：解压缩.bz2 文件

```
i am from bj
[root@localhost root]#
```

解压文件 abc.bz2，生成文件 abc，文件 abc 的内容为“i am a student”（第一行）、“i am from bj”（第二行），使用“-c”参数后，系统将解压后的数据直接输出到标准输出，如上面所示。



bzip2

## 6.3

## bzip2 指令：解压缩.bz2 文件

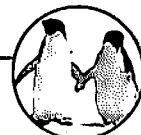
**语法** bzip2[必要参数][选择性参数][文件]

**功能说明** bzip2 指令使用 Burrows-Wheeler 块分类文本压缩算法压缩文件，并用 Huffman 进行编码。比基本的压缩算法效果更好，与 PPM 压缩效果相近。



必要参数的具体说明如下：

| 参数       | 功 能                          |
|----------|------------------------------|
| -c       | 压缩或者解压缩的结果输出到标准输出            |
| -d       | 强制解压                         |
| -z       | “-d”参数的补充，强制压缩               |
| -t       | 检查指定文件的完整性，但是不解压它们           |
| -f       | 输出文件的强制覆盖                    |
| -k       | 当进行解压缩或者压缩时，不删除输入的文件         |
| -s       | 压缩、解压缩或者测试时，减少内存的使用          |
| -q       | 不输出警告信息                      |
| -v       | 运行时输出详细的处理信息                 |
| -L       | 输出软件版本信息                     |
| -l -fast | 当压缩时将块的大小设置为 100KB，当解压缩时没有效果 |
| -9 -best | 当压缩时将块的大小设置为 900KB，当解压缩时没有效果 |



选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

### 【深入浅出】

- 范例 508 ○ 压缩文件。

```
[root@localhost root]# bzip2 abc //压缩文件 abc
```

压缩后生成文件 abc.bz2，原文件 abc 不再存在。

- 范例 509 ○ 压缩文件，将产生的压缩数据输出到标准输出。

```
[root@localhost root]# bzip2 -c abc //压缩文件 abc, 将产生的压缩数据输出到标准输出
```

- 范例 510 ○ 检查文件的完整性。

```
[root@localhost root]# bzip2 -t abc //检查文件 abc
bzip2: abc: bad magic number (file not created by bzip2)
You can use the 'bzip2recover' program to attempt to recover
data from undamaged sections of corrupted files.
```

### 【深入浅出】

bunzip2、tar、zcat

## 6.4

### bzip2recover 指令：损坏.bz2 文件的修复

#### 【深入浅出】 bzip2recover[.bz2 文件]

【深入浅出】 使用 bzip2recover 指令可以修复损坏的.bz2 文件的数据。

### 【深入浅出】

- 范例 511 ○ 修复.bz2 文件。

```
[root@localhost root]# bzip2recover abc.bz2 //修复.bz2 文件
bzip2recover 1.0.2: extracts blocks from damaged .bz2 files.
bzip2recover: searching for block boundaries ...
block 1 runs from 80 to 421.
```

## 6.5 compress 指令：压缩解压文件

```
bzip2recover: splitting into blocks
writing block 1 to 'rec00001abc.bz2' ...
bzip2recover: finished
[root@localhost root]#
```



bzip2、bunzip2

## 6.5 compress 指令：压缩解压文件



compress[必要参数][选择性参数][目录或者文件]



compress 指令通过使用 Lempel-Ziv 算法压缩指定文件的大小。每一个文件将被扩展名为.Z 的文件代替，并保存与源文件相同的拥有权、读写权限和更新时间。如果没有文件指定，那么标准输入将被压缩，并且输出到标准输出。



必要参数的具体说明如下：

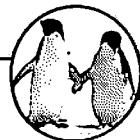
| 参数 | 功 能                  |
|----|----------------------|
| -c | 保留原始文件，压缩结果输出到标准输出   |
| -d | 文件的解压缩               |
| -f | 强制压缩                 |
| -r | 处理指定目录以及目录的子目录下的所有文件 |
| -v | 运行时输出详细的处理过程         |

选择性参数的具体说明如下：

| 参数     | 功 能            |
|--------|----------------|
| -b<数值> | 按指定数值的压缩效率进行压缩 |
| -V     | 版本信息           |



压缩效率：压缩的比例，值越大，压缩比例越高，数值在 9~16 之间。


◆ 执行范例 ◆

## ◎ 范例 512 ◎ 压缩文件。

```
[root@localhost ~]# compress a.c //压缩文件 a.c
[root@localhost ~]# ls
a.c.Z
```

本例使用指令“compress a.c”压缩文件 a.c，产生压缩文件 a.c.Z，源文件 a.c 消失。

## ◎ 范例 513 ◎ 解压缩文件。

```
[root@localhost ~]# compress -d a.c.Z //解压缩文件 a.c.Z
[root@localhost ~]# ls
a.c
```

本例使用指令解压缩文件 a.c.Z，得到文件 a.c。

## ◎ 范例 514 ◎ 压缩文件，保留原始文件，并且结果输出到标准输出。

```
[root@localhost ~]# compress -c a.c //压缩文件 a.c，并将结果输出到标准输出
```

## ◎ 范例 515 ◎ 按指定压缩比例进行压缩。

```
[root@localhost ~]# compress -b 7 a.c //按指定压缩比例进行压缩
```

## ◎ 范例 516 ◎ 强制压缩文件夹。

```
[root@localhost home]# compress -rf chord //压缩文件夹
[root@localhost home]#
```

本例使用指令 compress -rf chord/强制压缩文件夹/chord 下的所有文件。

## ◎ 范例 517 ◎ 解压缩文件夹下的所有文件。

```
[root@localhost home]# compress -rfd chord //解压缩文件夹
[root@localhost home]#
```

本例使用指令 compress -rfd chord/强制解压缩文件夹/chord 下的所有文件。

◆ 相关指令 ◆

pack、compact

## 6.6 cpio 指令：备份文件

◆ 简介 ◆ cpio[必要参数][选择性参数]

◆ 功能说明 ◆ cpio 指令用来创建或者提取文档，或者将一个文件

从一个地方复制到另外一个地方，它同时可以处理 tar 文件。

### ◎ 参数说明

必要参数的具体说明如下：

| 参数                    | 功能                   |
|-----------------------|----------------------|
| -0 –null              | 开启新增列控制字符            |
| -a                    | 更新文件的存取时间            |
| -A                    | 追加备份                 |
| -b                    | 等同于 “-sS”            |
| -B                    | 指定区块大小为 5210 字节      |
| -c                    | ASCII 备份             |
| <u>-d</u>             | 可新建目录                |
| <u>-f</u>             | 对不符合模板的文件进行操作        |
| <u>-i</u>             | 备份档还原                |
| <u>-l</u>             | 复制硬链接而不复制文件          |
| <u>-L</u>             | 复制文件而不是符号链接          |
| <u>-m</u>             | 忽略文件存取时间的更改          |
| <u>-n</u>             | 与 “-tv” 一起使用，设置用户和群组 |
| <u>-o</u>             | 建立备份档                |
| -p                    | 直接复制文件，忽略备份          |
| -r                    | 交互模式，改动时先询问用户        |
| -s                    | 交换字节                 |
| -S                    | 每半个字节交换              |
| -t                    | 显示输入内容               |
| -u                    | 强制更新所有文件             |
| <u>-v</u>             | 运行时显示详细的处理信息         |
| <u>-V</u>             | 在执行程序前加 “.”          |
| -force-local          | 强制备份                 |
| -no-absolute-filename | 使用相对路径               |



续表

| 参数                 | 功能        |
|--------------------|-----------|
| -no-preserve-owner | 文件的原用户不保留 |
| -only-verify-crc   | CRC 校验    |
| -quiet             | 不显示调试信息   |
| -sparse            | 稀疏文件处理    |

选择性参数的具体说明如下：

| 参数                | 功能                            |
|-------------------|-------------------------------|
| -C<字节数>           | 指定区块的字节数                      |
| -E<文件>            | 将指定文件作为范本文件                   |
| -F<文件>            | 不从标准输入/输出读取或者写入，而是从文件读取或者写入文件 |
| -H<格式>            | 指定备份格式                        |
| -l<备份档>           | 指定备份档名称                       |
| -M<回传信息>          | 指定回传信息                        |
| -O<指定备份档>         | 不使用标准输出，而是指定备份档               |
| -block-size=<字节数> | 设置区块大小                        |
| -version          | 版本信息                          |

### ◎ 范例与实践

#### ◎ 范例 518 ◎ 制作备份文件。

```
[root@localhost test]# ll //显示当前目录下的文件
总占用量 140
-rw-r--r-- 1 root root 100 5月 14 09:02 chch.c
-rw-r--r-- 1 root root 205 5月 14 09:02 cmh.c
-rw-r--r-- 1 root root 7376 5月 14 09:02 D.c
-rw-r--r-- 1 root root 675 5月 14 09:02 fcntl.c
-rw-r--r-- 1 root root 225 5月 14 09:02 file.c
-rw-r--r-- 1 root root 497 5月 14 09:02 findconf.c
-rw-r--r-- 1 root root 505 5月 14 09:02 getcurtime.c
-rw-r--r-- 1 root root 296 5月 14 09:02 getsec.c
-rw-r--r-- 1 root root 115 5月 14 09:02 gettime.c
```

## 6.6 cpio 指令：备份文件

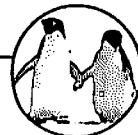
```
-rw-r--r-- 1 root root 513 5月 14 09:02 getusec.c
-rw-r--r-- 1 root root 92 5月 14 09:02 long.c
-rw-r--r-- 1 root root 126 5月 14 09:02 max.c
-rw-r--r-- 1 root root 151 5月 14 09:02 rand.c
-rw-r--r-- 1 root root 101 5月 14 09:02 sprintf.c
-rw-r--r-- 1 root root 131 5月 14 09:02 strrchr.c
-rw-r--r-- 1 root root 142 5月 14 09:02 wenjian.c
-rw-r--r-- 1 root root 474 5月 14 09:02 zuobiao.c
[root@localhost test]# ls|cpio -o >123.cpio //制作备份文件
25 blocks
[root@localhost test]# ll //显示当前目录下的文件
总占用量 160
-rw-r--r-- 1 root root 12800 5月 14 09:03 123.cpio
-rw-r--r-- 1 root root 100 5月 14 09:02 chch.c
-rw-r--r-- 1 root root 205 5月 14 09:02 cmh.c
-rw-r--r-- 1 root root 7376 5月 14 09:02 D.c
-rw-r--r-- 1 root root 675 5月 14 09:02 fcntl.c
-rw-r--r-- 1 root root 225 5月 14 09:02 file.c
-rw-r--r-- 1 root root 497 5月 14 09:02 findconf.c
-rw-r--r-- 1 root root 505 5月 14 09:02 getcurtime.c
-rw-r--r-- 1 root root 296 5月 14 09:02 getsec.c
-rw-r--r-- 1 root root 115 5月 14 09:02 gettimeofday.c
-rw-r--r-- 1 root root 513 5月 14 09:02 getusec.c
-rw-r--r-- 1 root root 92 5月 14 09:02 long.c
-rw-r--r-- 1 root root 126 5月 14 09:02 max.c
-rw-r--r-- 1 root root 151 5月 14 09:02 rand.c
-rw-r--r-- 1 root root 101 5月 14 09:02 sprintf.c
-rw-r--r-- 1 root root 131 5月 14 09:02 strrchr.c
-rw-r--r-- 1 root root 142 5月 14 09:02 wenjian.c
-rw-r--r-- 1 root root 474 5月 14 09:02 zuobiao.c
[root@localhost test]#
```

本例使用指令“\$|cpio -o >123.cpio”将当前文件夹下的所有文件打包备份到文件 123.cpio，备份文件的格式为.cpio。

## ◎ 范例 519 ◎ 解压缩备份文件。

```
[root@localhost kk]# cpio -i -l 123.cpio //解压缩文件
25 blocks
[root@localhost kk]#
```

本例使用指令 cpio -i -l 123.cpio 解压缩文件 123.cpio 到当前文件夹下。



## ◎ 范例 520 ◎ 解压缩备份文件，并列出详细信息。

```
[root@localhost test]# cpio -t -I 123.cpio //解压缩文件
123.cpio
chch.c
cmh.c
D.c
fcntl.c
file.c
findconf.c
getcurtime.c
getsec.c
gettime.c
getusec.c
long.c
max.c
rand.c
sprintf.c
strrchr.c
wenjian.c
zuobiao.c
25 blocks
[root@localhost test]#
```

本例使用指令“`cpio -t -I 123.cpio`”解压缩文件 `123.cpio`，并列出详细的信息。

## ◎ 范例 521 ◎ 强行解压缩，如果原文件存在将直接覆盖。

```
[root@localhost test]# cpio -i -u -I 123.cpio //强行解压缩
```

## ◎ 范例 522 ◎ 解压缩时进行反向匹配，指定文件不解压。

```
[root@localhost test]# cpio -i -I 123.cpio -f *.c //解压缩时反向匹配
```

本例使用指令“`cpio -i -I 123.cpio -f *.c`”解压缩文件 `123.cpio`，并在解压缩时进行反向匹配，指定的.c 文件将不被解压缩，其他类型的文件将全被解压缩出来。

## ◎ 范例 523 ◎ 向指定的.cpio 文件添加文件。

```
[root@localhost test]# ls
123.cpio abc.c
[root@localhost test]# cpio -o -O 123.cpio -A //向.cpio文件添加文件
abc.c //用户输入
```

```
1 block
[root@localhost test]#
```

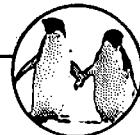
本例使用指令“cpio -o -O 123.cpio -A”向已经存在的 123.cpio 文件添加文件。输入文件 abc.c，然后按 Ctrl+D 组合键完成输出，这样文件 abc.c 就添加到 123.cpio 文件中了，完成了向指定.cpio 文件中添加文件的工作。

◎ 范例 524 ◎ 从标准输入备份文件。

```
[root@localhost test]# ls
abc.c D.c findconf.c gettimeofday.c max.c strrchr.c
chch.c fcntl.c getcurtime.c getusec.c rand.c wenjian.c
cmh.c file.c getsec.c long.c sprintf.c zuobiao.c
[root@localhost test]# cpio -o >123.cpio //备份文件
abc.c
chch.c
cmh.c
fcntl.c
file.c
D.c
findconf.c
getsec.c
getcurtime.c
gettime.c
getusec.c
long.c
max.c
rand.c
sprintf.c
strrchr.c
wenjian.c
zuobiao.c
25 blocks
[root@localhost test]#
```

本例使用指令“cpio -o >123.cpio”备份文件到指定的.cpio 文件，被备份的文件通过标准输入进行输入，所有以上出现的.c 文件都是用户输入。输入结束时，使用“Ctrl+D”组合键结束输入。这样输入的文件将被备份到指定文件 123.cpio。

◎ 范例 525 ◎ 复制文件。



```
[root@localhost test]# cpio -p /home/kk/ //复制文件
abc.c //用户输入
chch.c //用户输入
1 block
[root@localhost test]# cd /home/kk/
[root@localhost kk]# ls
abc.c chch.c
[root@localhost kk]#
```

本例使用指令“`cpio -p /home/kk/`”复制文件到指定目录`/home/kk`下，要复制的文件通过标准输入进行输入。用户输入的文件也就是要复制的文件`abc.c`和`chch.c`，用户输入结束时，按“`Ctrl+D`”结束。这样两个文件`abc.c`和`chch.c`就复制到了指定的目录`/home/kk/`，指令“`cpio -p`”相当于指令`cp`。

#### 命令大全

`cp`

## 6.7 dump 指令：文件系统备份

**功能说明** `dump`[必要参数][选择性参数][目录或者文件系统]

**功能说明** `dump` 指令检测 ext2/3 文件系统的文件，决定哪些文件需要备份。这些文件被复制到给定的硬盘、磁带或者别的存储介质。如果输出的部分比存储的媒体的容量大，那么输出部分将被分成多个块。在大多数存储介质上，大小是由写入决定的，直到存储介质返回一个终止符之后，就不能再写入了。

#### 命令说明

必要参数的具体说明如下：

| 参 数                      | 功 能                                                   |
|--------------------------|-------------------------------------------------------|
| <code>-0123456789</code> | 备份的等级                                                 |
| <code>-c</code>          | 变更磁带密度和容量                                             |
| <code>-n</code>          | 管理员设置                                                 |
| <code>-u</code>          | 记录完毕后，记录文件系统的层级、日期、时间等到文件 <code>/etc/dumpdates</code> |

续表

| 参数 | 功能                   |
|----|----------------------|
| -w | 显示要备份的文件             |
| -W | 显示备份文件和最新的备份层级、时间和日期 |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -b<区块大小> | 指定区块大小 |
| -B<区块数目> | 指定区块数目 |
| -d<磁带密度> | 设置磁带密度 |
| -f<设备名称> | 指定设备   |
| -h<层级>   | 设定备份层级 |
| -s<磁带长度> | 指定磁带长度 |
| -T<日期>   | 指定备份日期 |

### ◎ 执行范例

◎ 范例 526 ◎ 备份文件到磁带。

```
[root@localhost root]#dump -0 -u /dev/tape /home //备份文件系统
```

本例使用指令“dump -0 -u /dev/tape /home”备份/home 下的内容到指定的磁带。其中“-0”参数指定的是备份的等级，“-u”参数要求备份完毕之后将相应的信息存储到文件/etc/dumpdates，留作记录。

### ◎ 相关命令

fstab、restore、rmt

## 6.8 fdisk 指令：Linux 分区表控制

◎ 基础 **fdisk[选择性参数][必要参数][设备名称]**

◎ 功能说明 **fdisk** 指令是一个创建和维护分区表的驱动程序。它兼容 DOS 类型的分区表、BSD 或者 SUN 类型的磁盘列表。



### 基本说明

必要参数的具体说明如下：

| 参    数 | 功    能           |
|--------|------------------|
| -l     | 列出分区表情况          |
| -u     | 与“-l”搭配使用，显示分区数目 |

选择性参数的具体说明如下：

| 参    数   | 功    能 |
|----------|--------|
| -k<分区大小> | 指定分区大小 |
| -s<分区编号> | 指定分区   |
| -v       | 版本信息   |

### 实际应用

- 范例 527 ○ 显示当前磁盘的情况。

```
[root@localhost ~]# fdisk -l //显示当前磁盘的情况
Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
 Device Boot Start End Blocks Id System
/dev/sdal * 1 13 104391 83 Linux
/dev/sda2 14 78 522112+ 82 Linux swap / Solaris
/dev/sda3 79 1305 9855877+ 83 Linux
[root@localhost ~]#
```

本例使用指令“fdisk -l”显示当前磁盘的情况。

- 范例 528 ○ 显示 SCSI 硬盘的每个分区的情况。

```
[root@localhost ~]# fdisk -lu /dev/sda3 //显示SCSI硬盘的分区情况
Disk /dev/sda3: 10.0 GB, 10092418560 bytes
255 heads, 63 sectors/track, 1227 cylinders, total 19711755 sectors
Units = sectors of 1 * 512 = 512 bytes
Disk /dev/sda3 doesn't contain a valid partition table
[root@localhost ~]#
```

本例使用指令“fdisk -lu /dev/sda3”显示 SCSI 硬盘每个分区/dev/sda3

## 6.8 fdisk 指令：Linux 分区表控制

的磁盘信息。

○ 范例 529 ○ 进入 fdisk 命令进行相应操作。

```
[root@localhost ~]# fdisk /dev/sda3 //进入 fdisk 命令进行相应操作
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.
```

```
The number of cylinders for this disk is set to 1227.
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)

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): m
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)

Command (m for help): p
Disk /dev/sda3: 10.0 GB, 10092418560 bytes
255 heads, 63 sectors/track, 1227 cylinders
```



## 第6章 备份压缩

```
Units = cylinders of 16065 * 512 = 8225280 bytes
Device Boot Start End Blocks Id System
Command (m for help):
```

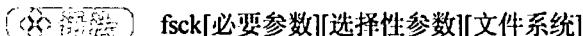
本例使用指令“fdisk /dev/sda3”进入 fdisk 指令进行相应的指令操作，相应的指令如下。

- a: 标记分区。
- d: 删 除 分 区。
- l: 列出分区类型。
- m: 帮助信息。
- n: 新建分区。
- p: 列出分区信息。
- q: 退出但不保存。
- t: 设置分区编号。
- v: 进行分区的检测。
- w: 写回硬盘。



mkfs、parted、sfdisk

## 6.9 fsck 指令：检查文件系统并尝试修复错误



fsck[必要参数][选择性参数][文件系统]

fsck 指令用于检测并修复出错的文件系统，操作对象可以是设备名，例如：/dev/hdc1；可以是挂载点，例如：/、/usr、/home；可以是 ext2 label，例如：LABEL=/root；还可以是 UUID，例如：UUID=8868abf6-88c5-4a83-98b8-bfc24057f7bd。



必要参数的具体说明如下：

| 参数 | 功 能         |
|----|-------------|
| -a | 非交互式模式，自动修复 |
| -r | 交互模式        |

## 6.9 fsck 指令：检查文件系统并尝试修复错误

续表

| 参数 | 功能             |
|----|----------------|
| -R | 忽略目录           |
| -s | 顺序检查           |
| -v | 运行指令时显示详细的处理信息 |
| -f | 强制进行检查         |

选择性参数的具体说明如下：

| 参数       | 功能     |
|----------|--------|
| -t<文件类型> | 指定文件类型 |
| -V       | 版本信息   |

**执行范例****◎ 范例 530 ◎ 检查文件系统。**

```
[root@localhost ~]# fsck //检查文件系统
fsck 1.37 (21-Mar-2005)
e2fsck 1.37 (21-Mar-2005)
/dev/sda3 is mounted.
WARNING!!! Running e2fsck on a mounted filesystem may cause
SEVERE filesystem damage.
Do you really want to continue (y/n)? no //用户输入
check aborted.
e2fsck 1.37 (21-Mar-2005)
ext2fs_check_if_mount: Permission denied while determining whether
/dev/sdal is mounted.
/boot: recovering journal
/boot: clean, 35/26104 files, 14614/104388 blocks
[root@localhost ~]#
```

本例使用指令 fsck 检查文件系统，如果有错误，将尝试修复。

**相关指令**

fstab、mkfs、fsck.ext2、e2fsck、cramfsck、fsck.minix、fsck.jfs、fsck.xfs、fsck.xiafs、reiserfsck



## 6.10 fsck.ext2 指令：检查 ext2 文件系统

**fsck.ext2[必要参数][选择性参数][设备代号]**

针对 ext2 型文件系统进行检测。

必要参数的具体说明如下：

| 参数 | 功 能            |
|----|----------------|
| -a | 非交互式模式，自动修复    |
| -i | 交互模式           |
| -R | 忽略目录           |
| -s | 顺序检查           |
| -v | 运行指令时显示详细的处理信息 |
| -f | 强制进行检查         |
| -t | 显示时序信息         |

选择性参数的具体说明如下：

| 参数             | 功 能            |
|----------------|----------------|
| -b<分区第一磁盘分区地址> | 指定第一个磁盘分区的起始地址 |
| -B<区块大小>       | 指定区块的大小        |
| -l<mode 缓冲区块数> | 指定缓冲区的块数       |

◎ 范例 531 ◎ 检测磁盘。

```
[root@localhost ~]# fsck.ext2 /dev/fd0 //检测/dev/fd0
```

本例使用指令“fsck.ext2 /dev/fd0”检测磁盘/dev/fd0。

◎ 范例 532 ◎ 检测磁盘并显示时序信息。

```
[root@localhost ~]# fsck.ext2 -ft /dev/fd0//检测/dev/fd0 并显示时序信息
```

## 6.11 fsck.ext3 指令：检查 ext3 文件系统

第  
6  
章

本例使用指令“fsck.ext2 -ft /dev/fd0”检测磁盘/dev/fd0，并显示时序信息。



## 相关指令

fstab、mkfs、fsck、e2fsck、cramfsck、fsck.minix、fsck.jfs、fsck.xfs、fsck.xiafs、reiserfsck、fsck.ext3

## 6.11 fsck.ext3 指令：检查 ext3 文件系统



fsck.ext3[必要参数][选择性参数][设备代号]



针对 ext3 型文件系统进行检测。



必要参数的具体说明如下：

| 参    数 | 功    能         |
|--------|----------------|
| -a     | 非交互式模式，自动修复    |
| -t     | 交互模式           |
| -R     | 忽略目录           |
| -s     | 顺序检查           |
| -v     | 运行指令时显示详细的处理信息 |
| -f     | 强制进行检查         |
| -t     | 显示时序信息         |

选择性参数的具体说明如下：

| 参    数          | 功    能         |
|-----------------|----------------|
| -b<分区第一磁盘分区地址>  | 指定第一个磁盘分区的起始地址 |
| -B<区块大小>        | 指定区块的大小        |
| -l<inode 缓冲区块数> | 指定缓冲区的块数       |



◎ 范例 533 ◎ 检测磁盘。



[root@localhost ~]# **fsck.ext3 /dev/fd0** //检测/dev/fd0

本例使用指令“**fsck.ext3 /dev/fd0**”检测磁盘/dev/fd0。

◎ 范例 534 ◎ 检测磁盘，并显示时序信息。

[root@localhost ~]# **fsck.ext3 -ft /dev/fd0** //检测/dev/fd0，并显示时序信息

本例使用指令“**fsck.ext3 -ft /dev/fd0**”检测磁盘/dev/fd0，并显示时序信息。

相关指令

fstab、mkfs、fsck、e2fsck、cramfsck、fsck.minix、fsck.jfs、fsck.xfs、fsck.xiafs、reiserfsck、fsck.ext2

## 6.12

### fsck.minix 指令：检查文件系统并尝试修复错误

命令

**fsck.minix[必要参数][选择性参数][设备号]**

功能说明

针对 minix 文件系统进行磁盘检测

参数说明

必要参数的具体说明如下：

| 参 数 | 功 能          |
|-----|--------------|
| -a  | 非交互模式，直接检测   |
| -f  | 强制检测         |
| -m  | 警告信息         |
| -r  | 交互模式         |
| -s  | 显示第一个分区信息    |
| -v  | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

**◎ 执行范例****◎ 范例 535 ◎ 检测磁盘。**

```
[root@localhost ~]# fsck.minix /dev/fd0 //检测/dev/fd0
```

本例使用指令“fsck.minix /dev/fd0”检测磁盘/dev/fd0。

**◎ 相关指令**

fstab、mkfs、fsck、e2fsck、cramfsck、fsck.ext3、fsck.jfs、fsck.xfs、fsck.xiafs、reiserfsck、fsck.ext2

## 6.13

## gunzip 指令：解压缩文件

**◎ 基本用法**

gunzip[必要参数][选择性参数][文件……]

**◎ 功能说明**

gunzip 指令用来解压缩文件，是通过 gzip 指令来完成。

**◎ 参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能               |
|-----|-------------------|
| -a  | ASCII 模式          |
| -c  | 结果输出到标准输出         |
| -f  | 强制解压缩             |
| -l  | 列出压缩文件信息          |
| -q  | 不显示警告信息           |
| -v  | 运行时显示详细的处理信息      |
| -r  | 解压缩时同时处理子目录下的所有文件 |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

**◎ 执行范例****◎ 范例 536 ◎ 解压缩文件。**



[root@localhost root]#gunzip chord.gz //解压缩文件

本例使用指令“gunzip chord.gz”解压缩文件 chord.gz。

相关指令

gzip

## 6.14 gzexe 指令：压缩可执行文件

**gzexe**[必要参数][选择性参数][可执行文件……]

**gzexe** 指令可以用来压缩可执行文件，以节省空间。

功能说明

必要参数的具体说明如下：

| 参数 | 功能       |
|----|----------|
| -d | 解压缩可执行文件 |

选择性参数的具体说明如下：

| 参数       | 功能   |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |

相关命令

◎ 范例 537 ◎ 压缩可执行文件。

[root@localhost root]#gzexe lsd //压缩可执行文件

本例使用指令“gzexe lsd”压缩可执行文件。

相关指令

gzip

## 6.15 gzip 指令：压缩文件

gzip

**gzip**[必要参数][选择性参数][文件或者目录]

**功能说明** gzip 是个使用广泛的压缩程序，这类压缩文件的扩展名为.gz。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能                   |
|-----|-----------------------|
| -a  | ASCII 模式              |
| -c  | 保留原始文件，把压缩后的文件输出到标准输出 |
| -d  | 解压缩文件                 |
| -l  | 压缩文件的信息列表             |
| -N  | 不保存原文件名和时间戳进行压缩       |
| -r  | 同时处理指定目录下的所有文件和子目录    |
| -q  | 忽略警告信息                |
| -t  | 测试压缩文件的正确性            |
| -v  | 显示执行过程                |

选择性参数的具体说明如下：

| 参 数   | 功 能                    |
|-------|------------------------|
| -h    | 在线帮助                   |
| -L    | 版本信息和版权                |
| -S    | 压缩字尾字符串                |
| -V    | 显示版本信息                 |
| -fast | 等同于“-1”，指定的值越大，压缩效率就越高 |
| -best | 等同于“-9”                |

**执行范例**

○ 范例 538 ○ 假设一个目录/home 下有两个文件 abc 和 123。

```
$ cd /home
$ gzip *
//压缩文件
$ ls
abc.gz 123.gz
```



## 第6章 备份压缩

通过通配符“\*”将/home目录下的所有文件压缩成了.gz文件。

- 范例 539 ○ 把[范例 538]中每个压缩的文件解压缩，并列出详细的信息。

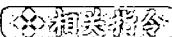
```
$ gzip -dv *
abc.gz 41.5%-----replaced with abc
123..gz 43.1%-----replaced with 123
$ ls
abc 123
```

- 范例 540 ○ 显示[范例 538]中所有压缩文件的详细信息。

```
$ gzip -l * //显示压缩文件信息
compressed uncompr. ratio uncompressed_name
277 445 41.5% abc
278 445 43.1% 123
$ ls
abc.gz 123.gz
```

- 范例 541 ○ 压缩 tar 文件，如 abc.tar，其扩展名为.tar.gz

```
$ gzip abc.tar //压缩文件
$ ls
abc.tar.gz
```



cat、gunzip、zcat、compress、tar、bzip2。

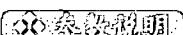
## 6.16

### hdparm 指令：显示和设定硬盘参数



**hdparm[必要参数][选择性参数][设备]**

**功能说明** hdparm 指令提供了一个命令行接口来显示设置硬盘参数。一些参数可能只能在最新的内核下工作。最好在最新的内核下执行 haparm 指令。



必要参数的具体说明如下：

| 参数 | 功能               |
|----|------------------|
| -C | 电源管理模式           |
| -f | 缓冲区数据写入硬盘，并清空缓冲区 |

## 6.16 hdparm 指令：显示和设定硬盘参数

续表

| 参数 | 功能         |
|----|------------|
| -g | 显示硬盘参数     |
| -I | 硬盘规格       |
| -q | 结果不输出到标准输出 |
| -t | 读取效率       |
| -T | 缓存的读取效率    |
| -v | 显示硬盘信息     |
| -y | 省电模式       |
| -Y | 休眠模式       |
| -Z | 关闭省电模式     |

选择性参数的具体说明如下：

| 参数       | 功能           |
|----------|--------------|
| -a<快取扇区> | 读文件时预先存入快取扇区 |
| -A 0     | 关闭快取         |
| -A 1     | 开启快取         |
| -d 0     | 关闭 DMA       |
| -d 1     | 开启 DMA       |
| -m<扇区数>  | 指定扇区数        |
| -n 0     | 忽略磁盘写入错误     |
| -n 1     | 不忽略磁盘写入错误    |
| -P PIO   | PIO 模式       |
| -p<扇区数>  | 指定缓存扇区数      |
| -S<时间>   | 指定时间后进入省电模式  |
| -X<传输模式> | 指定传输模式       |
| -h       | 帮助信息         |

## 执行范例

(◎) 范例 542 (◎) 显示电源管理模式。

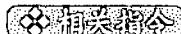
```
[root@localhost ~]# hdparm -c /dev/hdc //显示电源管理模式
/dev/hdc:
IO_support = 0 (default 16-bit)
[root@localhost ~]#
```

本例使用指令“hdparm -c /dev/hdc”显示/dev/hdc 的电源管理模式。

◎范例 543 ◎ 开启关闭 DMA 模式。

```
[root@localhost ~]# hdparm -d 1 /dev/hdc //开启 DMA 模式
/dev/hdc:
setting using_dma to 1 (on)
using_dma = 1 (on)
[root@localhost ~]# hdparm -d 0 /dev/hdc //关闭 DMA 模式
/dev/hdc:
setting using_dma to 0 (off)
using_dma = 0 (off)
[root@localhost ~]#
```

本例首先使用指令“hdparm -d 1 /dev/hdc”开启 DMA 模式；然后使用指令“hdparm -d 0 /dev/hdc”关闭 DMA 模式。



fdisk

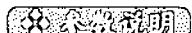
## 6.17 lha 指令：压缩或解压缩文件



**lha** [必要参数] [选择性参数] [压缩文件] [文件]



从 lharc 指令演变而来的压缩程序，文件压缩过后会产生具有.lhz 扩展名的文件。



必要参数的具体说明如下：

| 参数 | 功能             |
|----|----------------|
| -a | 压缩文件，并加入至压缩文件内 |
| -c | 重新构建压缩文件，并将其加入 |
| -e | 解开压缩文件         |

## 6.18 tar 指令：压缩/解压缩文件

续表

| 参数 | 功能                          |
|----|-----------------------------|
| -f | 执行 lha 时不询问，强行覆盖已有的文件而不加以询问 |
| -l | 显示压缩文件的信息                   |
| -p | 从压缩文件内输出到标准设备               |
| -v | 显示压缩文件的详细信息                 |
| -x | 解压缩文件                       |

选择性参数的具体说明如下：

| 参数         | 功能                                        |
|------------|-------------------------------------------|
| -a <0/1/2> | 设置压缩文件不同的头，“0”不记录每个文件的路径，“1”为预设值，“2”较节省空间 |
| -<a/c/u>d  | 将文件压缩，更新原有的压缩文件，加入新的文件，并将原文件删除            |
| -<e/h>i    | 解压缩时，忽略压缩文件指定的路径，将文件解压后放在现行的目录下           |
| -<a/u>z    | 新文件加入压缩文件时，不将文件压缩                         |

### ◎ 范例范例

- 范例 544 ◎ 将文件名为 hisword 的文件压缩后保存为名为 abc 的压缩文件。

```
[root@wjc linuxzhiling]# lha -a abc hisword
```

- 范例 545 ◎ 将/home/wjc/linuxzhiling 目录下的所有文件一起压缩。

```
[root@wjc linuxzhiling]# lha -a filosip /home/wjc/linuxzhiling
```

- 范例 546 ◎ 将压缩文件 abc 解压缩到现行目录的 agis 目录里，并忽略文件内记录的地址。

```
[root@wjc linuxzhiling]# lha -xiw=agis abc
```

### ◎ 词典指南

tar

## 6.18

## tar 指令：压缩/解压缩文件

### ◎ 热点

tar[必要参数][选择性参数][文件]



**功能说明** tar 指令用来压缩和解压缩文件。tar 本身不具有压缩/解压缩功能，它调用其他指令来实现压缩/解压缩功能。

**参数说明**

必要参数的具体说明如下：

| 参 数 | 功 能                    |
|-----|------------------------|
| -A  | 新增压缩文件到已存在的压缩文件        |
| -B  | 设置区块大小                 |
| -c  | 建立新的压缩文件               |
| -d  | 记录文件的差别                |
| -r  | 添加文件到已经压缩的文件           |
| -u  | 添加改变了的和现有的文件到已经存在的压缩文件 |
| -x  | 从压缩文件中提取文件             |
| -t  | 显示压缩文件的内容              |
| -z  | 支持 gzip 解压缩文件          |
| -j  | 支持 bzip2 解压缩文件         |
| -Z  | 支持 compress 解压缩文件      |
| -v  | 显示操作过程                 |
| -l  | 文件系统边界设置               |
| -k  | 保留原有文件不被覆盖             |
| -m  | 保留文件的原更改时间             |
| -w  | 确认压缩文件是否正确无误           |

选择性参数的具体说明如下：

| 参 数      | 功 能     |
|----------|---------|
| -b       | 设置区块数目  |
| -C       | 切换到指定目录 |
| -f       | 指定压缩文件  |
| -version | 版本信息    |
| -help    | 帮助信息    |

## 6.19 umount 指令：卸除文件系统

## 范例执行范例

## ◎ 范例 547 ◎ 压缩文件。

```
[root@localhost home]# tar -czvf kk.tar.gz kk //压缩文件 kk 为 kk.tar.gz
kk/
kk/.kde/
kk/.kde/Autostart/
[root@localhost home]# ls
kk kk.tar.gz
```

利用“-czvf”参数，压缩 kk 文件为 kk.tar.gz。现在/home 目录下多了个文件 kk.tar.gz。

## ◎ 范例 548 ◎ 利用“-tzvf”参数，列出压缩文件的内容。

```
[root@localhost home]# tar -tzvf kk.tar.gz //列出压缩文件 kk.tar.gz 的内容信息
drwx----- kk/kk 0 2007-04-23 15:30:26 kk/
drwxr-xr-x kk/kk 0 2002-08-12 17:26:50 kk/.kde/
drwxr-xr-x kk/kk 0 2007-04-20 19:22:53 kk/.kde/Autostart/
```

## ◎ 范例 549 ◎ 删除/home 文件夹下的 kk 文件夹，解压缩文件 kk.tar.gz，重新得到文件夹 kk。

```
[root@localhost home]# tar -xzvf kk.tar.gz //解压缩文件 kk.tar.gz
kk/
kk/.kde/
kk/.kde/Autostart/
[root@localhost home]# ls
kk kk.tar.gz linuxcommand other
```

## 相关指令

gzip、bzip2、compress

## 6.19 umount 指令：卸除文件系统

## 基本

umount[必要参数][选择性参数][文件系统]

## 功能说明

umount 指令用来卸载挂载的文件系统。



## 第6章 备份压缩

### ◆ ◆ ◆ 功能说明

必要参数的具体说明如下：

| 参 数 | 功 能          |
|-----|--------------|
| -a  | 卸载所有文件系统     |
| -n  | 卸载时不记录信息     |
| -r  | 只读方式         |
| -v  | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参 数     | 功 能    |
|---------|--------|
| -h      | 显示帮助信息 |
| -<文件类型> | 指定文件类型 |

### ◆ ◆ ◆ 执行范例

◎ 范例 550 ◎ 卸载文件系统。

```
[root@localhost ~]# umount /mnt/cdrom/ //卸载文件系统
```

本例使用指令“umount /mnt/cdrom/” 卸载文件系统/mnt/cdrom。

### ◆ ◆ ◆ 相关指令

mount

## 6.20 unarj 指令：解压缩文件

### ◆ ◆ ◆ 命令

unarj[必要参数][文件]

### ◆ ◆ ◆ 功能说明

unarj 指令用来解压缩.arj 文件。

### ◆ ◆ ◆ 功能说明

必要参数的具体说明如下：

| 参 数 | 功 能        |
|-----|------------|
| e   | 解压缩.arj 文件 |
| l   | 显示文件信息     |

## 6.21 uncompress 指令：解压缩.Z 文件

续表

| 参数 | 功能   |
|----|------|
| -t | 校验   |
| -x | 保留路径 |

**执行范例**

- 范例 551 ○ 解压缩.arj 文件。

```
[root@localhost ~]# unarj e abc.arj //解压缩文件
```

本例使用指令“unarj e abc.arj”解压缩.arj 文件。

**相关指令**

arj

## 6.21 uncompress 指令：解压缩.Z 文件

**语法** uncompress[必要参数][选择性参数].[Z 压缩文件]

**功能说明** uncompress 指令用来解压缩.Z 文件。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -c | 输出结果到标准输出    |
| -f | 强制执行         |
| -v | 运行时显示详细的处理信息 |

选择性参数的具体说明如下：

| 参数 | 功能   |
|----|------|
| -V | 版本信息 |

**执行范例**

- 范例 552 ○ 解压缩文件。



[root@localhost ~]# **uncompress abc.Z** //解压缩.Z文件

本例使用指令“uncompress abc.Z”解压缩.Z文件 abc.Z。

相关指令

**compress**

## 6.22 unzip 指令：解压缩 zip 文件

指令

**unzip[必要参数][选择性参数][文件]**

功能说明 **unzip** 指令用来解压缩 zip 文件。zip 压缩格式是现在常见的压缩格式。

功能说明

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -c | 将压缩结果输出到标准输出 |
| -f | 更新文件         |
| -l | 显示压缩包里的文件    |
| -t | 校验           |
| -v | 运行时显示详细的处理过程 |
| -z | 显示备注信息       |
| -q | 运行时不显示处理信息   |

选择性参数的具体说明如下：

| 参数     | 功能        |
|--------|-----------|
| -x<文件> | 指定的文件不处理  |
| -d<目录> | 解压缩到指定的目录 |

技术范例

- ◎ 范例 553 ◎ 显示压缩文件信息。

## 6.22 unzip 指令：解压缩 zip 文件

```
[root@localhost sfsnet]# unzip -l chord.zip//显示压缩文件的内容
```

Archive: chord.zip

| Length | Date     | Time  | Name                    |
|--------|----------|-------|-------------------------|
| 18665  | 03-27-06 | 16:07 | chord/accordion.C       |
| 4241   | 03-27-06 | 16:07 | chord/accordion.h       |
| 11269  | 03-27-06 | 16:07 | chord/accordion_table.C |
| 892    | 03-27-06 | 16:07 | chord/accordion_table.h |
| 1073   | 03-27-06 | 16:07 | chord/aclnt_chord.h     |
| 14588  | 03-27-06 | 16:07 | chord/chord.C           |
| 10709  | 03-27-06 | 16:07 | chord/chord_client.C    |
| 7572   | 03-27-06 | 16:07 | chord/chord.h           |
| 8114   | 03-27-06 | 16:07 | chord/chord_impl.h      |
| 10274  | 03-27-06 | 16:07 | chord/comm.C            |

//此处省略部分结果

-----  
246010 42 files

```
[root@localhost sfsnet]#
```

本例使用指令“unzip -l chord.zip”显示压缩文件 chord.zip 内的压缩文件的信息。

○ 范例 554 ○ 解压缩文件。

```
[root@localhost temp]# unzip -v chord.zip //解压缩文件
```

Archive: chord.zip

| Length | Method | Size | Ratio | Date     | Time  | CRC-32   | Name                     |
|--------|--------|------|-------|----------|-------|----------|--------------------------|
| 18665  | Defl:N | 4843 | 74%   | 03-27-06 | 16:07 | 60031156 | chord/ accordion.C       |
| 4241   | Defl:N | 1326 | 69%   | 03-27-06 | 16:07 | 3dd3bc9f | chord/ accordion.h       |
| 11269  | Defl:N | 3141 | 72%   | 03-27-06 | 16:07 | 2065fcf5 | chord/ accordion_table.C |
| 892    | Defl:N | 401  | 55%   | 03-27-06 | 16:07 | 882b024b | chord/ accordion_table.h |
| 1073   | Defl:N | 490  | 54%   | 03-27-06 | 16:07 | 7eb128e6 | chord/ aclnt_chord.h     |



## 第6章 备份压缩

```

14588 Defl:N 4237 71% 03-27-06 16:07 007cb521 chord/ chord.C
10709 Defl:N 3877 64% 03-27-06 16:07 d594d4e6 chord/ chord_client.C
7572 Defl:N 2595 66% 03-27-06 16:07 98ec87e4 chord/ chord.h
8114 Defl:N 2508 69% 03-27-06 16:07 1fb550ac chord/ chord_impl.h
 //此处省略部分结果
8314 Defl:N 2387 71% 03-27-06 16:07 d8c8cd32 chord/succ_list.C
1579 Defl:N 592 63% 03-27-06 16:07 1c2ca9df chord/succ_list.h

246010 78184 68%
 42 files
[root@localhost temp]#

```

本例使用指令“unzip -v chord.zip”解压缩文件 chord.zip，并显示详细的处理信息。



zip

## 6.23 zip 指令：压缩文件

**zip[必要参数][选择性参数][文件]**

**功能说明** zip 指令是一个应用广泛的跨平台的压缩工具，压缩后产生后缀为.zip 的文件。



必要参数的具体说明如下：

| 参数 | 功能       |
|----|----------|
| -A | 自动解压缩文件  |
| -c | 给压缩文件加注释 |
| -d | 删除文件     |
| -F | 修复损坏文件   |
| -k | 兼容 DOS   |

续表

| 参数 | 功能                 |
|----|--------------------|
| -m | 压缩完毕之后，删除原文件       |
| -q | 运行时不显示详细的处理过程      |
| -r | 处理指定目录和指定目录下的使用子目录 |
| -v | 运行时显示详细的处理信息       |
| -y | 保留符号链接             |

选择性参数的具体说明如下：

| 参数     | 功能        |
|--------|-----------|
| -b<目录> | 指定压缩到的目录  |
| -i<格式> | 匹配格式进行压缩  |
| -L     | 显示版权信息    |
| -t<日期> | 指定压缩文件的日期 |
| <压缩率>  | 指定压缩率     |

### ☆ 实行范例

#### ◎ 范例 555 ◎ 压缩文件。

```
[root@localhost kk]# zip -v inter.zip getsec.c long.c sprintf.c
```

//压缩文件

```
zip info: local extra (21 bytes) != central extra (13 bytes): getsec.c
zip info: local extra (21 bytes) != central extra (13 bytes): long.c
zip info: local extra (21 bytes) != central extra (13 bytes): sprintf.c
updating: getsec.c (in=296) (out=159) (deflated 46%)
updating: long.c (in=92) (out=69) (deflated 25%)
updating: sprintf.c (in=101) (out=87) (deflated 14%)
total bytes=489, compressed=315 -> 36% savings
[root@localhost kk]
```

本例使用指令“zip -v inter.zip getsec.c long.c sprintf.c”压缩文件 getsec.c long.c sprintf.c 到文件 inter.zip。



## 第6章 备份压缩

○范例 556 ○ 使用通配符压缩文件。

```
[root@localhost kk]# zip -v inter.zip *
//压缩文件
 zip info: local extra (21 bytes) != central extra (13 bytes): getsec.c
 zip info: local extra (21 bytes) != central extra (13 bytes): long.c
 zip info: local extra (21 bytes) != central extra (13 bytes): sprintf.c
file matches zip file -- skipping
updating: getsec.c (in=296) (out=159) (deflated 46%)
updating: long.c (in=92) (out=69) (deflated 25%)
updating: sprintf.c (in=101) (out=87) (deflated 14%)
 adding: 111 (in=8997) (out=1114) (deflated 88%)
 adding: abc.c (in=53) (out=39) (deflated 26%)
 adding: a.c (in=97) (out=57) (deflated 41%)
 adding: a.c~ (in=97) (out=57) (deflated 41%)
 adding: a.c@ (in=97) (out=57) (deflated 41%)
 adding: bu.....
.....(in=5867520) (out=4047368) (deflated 31%)
 adding: chch.c (in=100) (out=85) (deflated 15%)
 adding: cmh.c (in=205) (out=159) (deflated 22%)
 adding: D.c (in=7376) (out=1653) (deflated 78%)
 adding: ddd (in=348) (out=94) (deflated 73%)
 adding: Desktop/ (in=0) (out=0) (stored 0%)
 adding: getcurtime.c (in=505) (out=228) (deflated 55%)
 adding: gettimeofday.c (in=115) (out=70) (deflated 39%)
 adding: getusec.c (in=513) (out=232) (deflated 55%)
 adding: max.c (in=126) (out=110) (deflated 13%)
 adding: rand.c (in=151) (out=107) (deflated 29%)
 adding: strrchr.c (in=131) (out=99) (deflated 24%)
 adding: test (in=1563) (out=882) (deflated 44%)
 adding: unpv12e.tar..... (in=1471488)
(out=230578) (deflated 84%)
 adding: wenjian.c (in=142) (out=125) (deflated 12%)
total bytes=7360113, compressed=4283429 -> 42% savings
[root@localhost kk]#
```

本例使用指令“zip -v inter.zip \*”把当前文件夹下的所有文件压缩到文件 inter.zip。

## ◎ 范例 557 ◎ 递归压缩目录。

```
[root@localhost kk]# zip -r mac.zip /home/kk/ /home/temp//递归压缩目录
 adding: home/kk/ (stored 0%)
 adding: home/kk/.gnome2/ (stored 0%)
 adding: home/kk/.gnome2/panel2.d/ (stored 0%)
 adding: home/kk/.gnome2/panel2.d/default/ (stored 0%)
 adding: home/kk/.gnome2/panel2.d/default/launchers/ (stored 0%)
 adding: home/kk/.gnome2/panel2.d/default/launchers/ frarbonate- 00615bc2da.
desktop (deflated 44%)
 adding: home/kk/.gnome2/panel2.d/default/ launchers/larry- 00662b
43ce.desktop (deflated 49%)
 adding: home/kk/.gnome2/panel2.d/default/ launchers/frobate- 006120
b888.desktop (deflated 46%)
 adding: home/kk/.gnome2/panel2.d/default/ launchers/foo- 0078975455.
desktop (deflated 50%)
 adding: home/kk/.gnome2/panel2.d/default/ launchers/hammer- 005cca7974.
desktop (deflated 46%)
 adding: home/kk/.gnome2/nautilus-scripts/ (stored 0%)
 adding: home/kk/.gnome2/keyrings/ (stored 0%)
 adding: home/kk/.gnome2/accecs/ (stored 0%)
 adding: home/kk/.gnome2/share/ (stored 0%)
 //此处省略了部分结果
 adding: home/kk/.nautilus/metafiles/ x-nautilus- desktop: %2F%2F%2F.
xml (deflated 46%)
 adding: home/kk/.htpasswd (stored 0%)
 adding: home/kk/111 (deflated 88%)
 adding: home/temp/ (stored 0%)
[root@localhost kk]#
```

本例使用指令“zip -r mac.zip /home/kk/ /home/temp/”递归压缩目录到 mac.zip 文件。

## ◎ 范例 558 ◎ 从压缩文件中删除文件。

```
[root@localhost kk]# zip -dv inter.zip abc.c //删除文件
zip info: local extra (21 bytes) != central extra (13 bytes): getsec.c
```



```

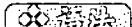
zip info: local extra (21 bytes) != central extra (13 bytes): long.c
zip info: local extra (21 bytes) != central extra (13 bytes): sprintf.c
zip info: local extra (21 bytes) != central extra (13 bytes): l11.c
zip info: local extra (21 bytes) != central extra (13 bytes): abc.c
zip info: local extra (21 bytes) != central extra (13 bytes): a.c
zip info: local extra (21 bytes) != central extra (13 bytes): a.c-
zip info: local extra (21 bytes) != central extra (13 bytes): a.c@
zip info: local extra (21 bytes) != central extra (13 bytes): bu.c
zip info: local extra (21 bytes) != central extra (13 bytes): chch.c
zip info: local extra (21 bytes) != central extra (13 bytes): cmh.c
zip info: local extra (21 bytes) != central extra (13 bytes): D.c
zip info: local extra (21 bytes) != central extra (13 bytes): ddd.c
zip info: local extra (21 bytes) != central extra (13 bytes): Desktop/
zip info: local extra (21 bytes) != central extra (13 bytes): getcurtime.c
zip info: local extra (21 bytes) != central extra (13 bytes): gettime.c
zip info: local extra (21 bytes) != central extra (13 bytes): getusec.c
zip info: local extra (21 bytes) != central extra (13 bytes): max.c
zip info: local extra (21 bytes) != central extra (13 bytes): rand.c
zip info: local extra (21 bytes) != central extra (13 bytes): test.c
zip info: local extra (21 bytes) != central extra (13 bytes): unpv12e.tar
zip info: local extra (21 bytes) != central extra (13 bytes): wenjian.c

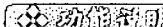
deleting: abc.c
total bytes=7359929, compressed=4283291 -> 42% savings
[root@localhost kk]#

```

本例使用指令“zip -dv inter.zip abc.c”从压缩文件 inter.zip 中删除文件 abc.c。

## 6.24 zipinfo 指令：显示压缩文件的信息

 **基础说明** zipinfo[必要参数][选择性参数][压缩文件][文件]

 **功能说明** zipinfo 指令用来显示压缩文件的信息。

 **参数说明**

必要参数的具体说明如下：

## 6.24 zipinfo 指令：显示压缩文件的信息

| 参数 | 功能            |
|----|---------------|
| -1 | 只显示文件名        |
| -2 | 只显示文件名        |
| -h | 显示文件名、大小、文件数目 |
| -T | 显示时间设置        |
| -v | 显示每个文件的内容     |
| -z | 显示注释          |

选择性参数的具体说明如下：

| 参数     | 功能         |
|--------|------------|
| -x<格式> | 匹配格式的文件不显示 |

### 运行实例

#### ●范例 559 ● 显示压缩文件的信息。

```
[root@localhost kk]# zipinfo inter.zip //显示压缩文件的信息
Archive: inter.zip 4285881 bytes 21 files
-rw-r--r-- 2.3 unx 296 tx defN 14-May-07 09:11 getsec.c
-rw-r--r-- 2.3 unx 92 tx defN 14-May-07 09:11 long.c
-rw-r--r-- 2.3 unx 101 tx defN 14-May-07 09:11 sprintf.c
-rw-r--r-- 2.3 unx 8997 tx defN 22-May-07 10:21 111
-rw-r--r-- 2.3 unx 97 tx defN 13-May-07 15:53 a.c
-rw-r--r-- 2.3 unx 97 tx defN 13-May-07 15:42 a.c~
-rw-r--r-- 2.3 unx 97 tx defN 13-May-07 15:46 a.c@
-rw-r--r-- 2.3 unx 5867520 tx defN 28-May-07 14:53 bu
-rw-r--r-- 2.3 unx 100 tx defN 14-May-07 09:40 chch.c
-rw-r--r-- 2.3 unx 205 tx defN 14-May-07 09:11 cmh.c
-rw-r--r-- 2.3 unx 7376 tx defN 14-May-07 09:11 D.c
-rw-r--r-- 2.3 unx 348 tx defN 28-May-07 22:06 ddd
drwxr-xr-x 2.3 unx 0 bx stor 13-May-07 15:57 Desktop/
-rw-r--r-- 2.3 unx 505 tx defN 14-May-07 09:11 getcurtime.c
-rw-r--r-- 2.3 unx 115 tx defN 14-May-07 09:11 gettimeofday.c
-rw-r--r-- 2.3 unx 513 tx defN 14-May-07 09:11 getusec.c
```



## 第6章 备份压缩

```
-rw-r--r-- 2.3 unx 126 tx defN 14-May-07 09:11 max.c
-rw-r--r-- 2.3 unx 151 tx defN 14-May-07 09:11 rand.c
-rw-r--r-- 2.3 unx 1563 tx defN 22-May-07 10:09 test
-rw-r--r-- 2.3 unx 1471488 tx defN 3-May-07 16:10 unpv12e.tar
-rw-r--r-- 2.3 unx 142 tx defN 14-May-07 09:11 wenjian.c
21 files, 7359929 bytes uncompressed, 4283291 bytes compressed: 41.8%
[root@localhost kk]#
```

本例使用指令“zipinfo inter.zip”显示压缩文件 inter.zip 的信息。

○ 范例 560 ○ 显示压缩文件中每个文件的信息。

```
[root@localhost kk]# zipinfo -v inter.zip //显示每个文件的信息
```

```
Archive: inter.zip 4285881 bytes 21 files
```

```
End-of-central-directory record:
```

---

```
Actual offset of end-of-central-dir record: 4285859 (004165A3h)
Expected offset of end-of-central-dir record: 4285859 (004165A3h)
(based on the length of the central directory and its expected offset)
This zipfile constitutes the sole disk of a single-part archive; its
central directory contains 21 entries. The central directory is 1368
(00000558h) bytes long, and its (expected) offset in bytes from the
beginning of the zipfile is 4284491 (0041604Bh).
```

```
There is no zipfile comment.
```

```
Central directory entry #1:
```

---

//此处省略了部分结果

|                                                                |              |
|----------------------------------------------------------------|--------------|
| length of filename:                                            | 9 characters |
| length of extra field:                                         | 13 bytes     |
| length of file comment:                                        | 0 characters |
| disk number on which file begins:                              | disk 1       |
| apparent file type:                                            | text         |
| Unix file attributes (100644 octal):                           | -rw-r--r--   |
| MS-DOS file attributes (00 hex):                               | none         |
| The central-directory extra field contains:                    |              |
| - A subfield with ID 0x5455 (universal time) and 5 data bytes. |              |
| The local extra field has UTC/GMT modification/access times.   |              |

## 6.24 zipinfo 指令：显示压缩文件的信息

- A subfield with ID 0x7855 (Unix UID/GID) and 0 data bytes.  
There is no file comment.

本例使用指令“zipinfo -v inter.zip”显示压缩 inter.zip 文件中每个文件的详细信息，内容比较多，中间部分已省略，但不影响阅读。



zip

<http://www.TopSage.com>



# LINUX

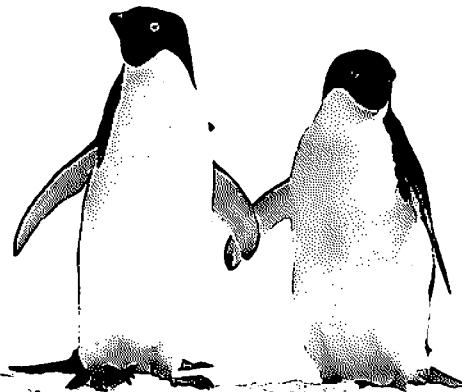
网  
上  
学  
习

TopSage.com

## 第7章

### 网络通信

网络通信是指 Linux 系统的信息的发送和接收。根据 Linux 系统的功能要求，网络通信指令执行的功能主要包括主机检测、普通网卡的设置、无线网卡的设置、主机间消息的发送、域名查询、远程登录和网络数据包的抓取等。本章将详细讲解网络通信的相关指令。





## 7.1

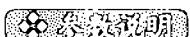
# apachectl 指令：apache HTTP 服务器控制接口



apachectl[参数]

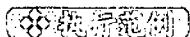


apachectl 指令是 apache HTTP 服务器的一个控制接口，通过 apachectl 指令可以控制 apache HTTP 服务器的开启、关闭等功能。



参数具体的说明如下：

| 参数         | 功能                   |
|------------|----------------------|
| configtest | 检查配置文件               |
| graceful   | 保留原有连接，重启 apache 服务器 |
| fullstatus | 列出服务器的状态信息           |
| restart    | 重启 apache 服务器        |
| start      | 启动服务器                |
| stop       | 关闭服务器                |
| help       | 帮助信息                 |
| status     | 显示服务器的主要状态报告         |
| startssl   | 开启服务器并使其支持 SSL       |



- 范例 561 ○ 启动 apache HTTP 服务器。

```
[root@localhost ~]# apachectl start //启动apache HTTP服务器
httpd (pid 2781) already running
```

```
[root@localhost ~]# apachectl restart//重新启动apache HTTP服务器
```

- 范例 562 ○ 检查配置文件的语法是否正确。

```
[root@localhost ~]# apachectl configtest //检查配置文档的语法是否正确
Syntax OK
```

- 范例 563 ○ 关闭 apache HTTP 服务器。

```
[root@localhost ~]# apachectl stop //关闭apache HTTP服务器
```

**相关指令**

httpd

**7.2****arp 指令：系统 ARP 缓存****【语法】** arp[必要参数][选择性参数]**【功能说明】** arp 指令处理内核的 ARP 缓存信息。如果为了调试的目的，arp 程序允许 ARP 缓存的完全清理。**【参数说明】**

必要参数的具体说明如下：

| 参 数 | 功 能        |
|-----|------------|
| -v  | 当前信息的详细输出  |
| -n  | 以编号的形式进行输出 |
| -D  | 使用硬件地址     |

选择性参数的具体说明如下：

| 参 数           | 功 能              |
|---------------|------------------|
| -H<类型>        | 指定 ARP 检测的类型     |
| -i<接口>        | 指定检测的接口          |
| -f<文件名>       | 地址信息从指定的文件中读取    |
| netmask<子网掩码> | 设置子网掩码           |
| -s<主机名 硬件地址>  | 为指定主机创建 ARP 地址映射 |
| -a<主机名>       | 显示指定主机的入口        |
| -d<主机名>       | 删除指定主机的入口        |

**【执行范例】**

- 范例 564 ○ 输出当前系统的 ARP 缓存。

```
root@localhost:~# arp //输出 ARP 缓存信息
Address HWtype HWaddress Flags Mask Iface
```

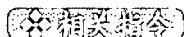


|                 |       |                   |   |      |
|-----------------|-------|-------------------|---|------|
| 202.112.146.85  | ether | 00:04:61:64:5D:68 | C | eth1 |
| 202.112.146.241 | ether | 00:0E:A6:19:3F:F7 | C | eth1 |
| 192.168.88.105  | ether | 00:A0:D1:D5:38:85 | C | eth0 |
| 192.168.88.105  | ether | 00:A0:D1:D5:38:85 | C | eth1 |
| 192.168.88.33   | ether | 00:18:FE:6D:64:EC | C | eth0 |
| 202.112.146.150 | ether | 00:0C:76:8A:97:D3 | C | eth1 |
| 192.168.88.7    | ether | 00:18:FE:6D:6F:06 | C | eth0 |
| 192.168.88.76   | ether | 00:0C:29:32:45:1C | C | eth0 |
| 202.112.146.2   | ether | 00:0F:E2:1B:31:C0 | C | eth1 |

○范例 565 ○ 输出指定网卡的 ARP 缓存。

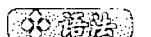
```
root@localhost:~# arp -i eth0 //输出网卡 eth0 的 ARP 缓存信息
```

| Address       | HWtype | HWaddress         | Flags | Mask | Iface |
|---------------|--------|-------------------|-------|------|-------|
| 192.168.88.33 | ether  | 00:18:FE:6D:64:EC | C     |      | eth0  |
| 192.168.88.7  | ether  | 00:18:FE:6D:6F:06 | C     |      | eth0  |
| 192.168.88.76 | ether  | 00:0C:29:32:45:1C | C     |      | eth0  |



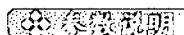
arp、route、ifconfig、netstat

## 7.3 arpwatch 指令：监听 ARP 记录



arpwatch[必要参数][选择性参数]

④ 功能说明 arpwatch 指令跟踪以太网地址和 IP 地址的配对情况，它将通过 E-mail 的形式报告当前的变化。arpwatch 使用 pcap 来侦听本地网卡的 ARP 数据包。



必要参数的具体说明如下：

| 参    数 | 功    能    |
|--------|-----------|
| -d     | 输出调试信息    |
| -N     | 使报告不能正常进行 |

选择性参数的具体说明如下：

## 7.4 arping 指令：向邻居主机发送 ARP 请求

| 参数       | 功能                                   |
|----------|--------------------------------------|
| -f<文件>   | 监听的 ARP 信息将保存在指定的文件中                 |
| -i<网卡接口> | 指定监听的网卡                              |
| -n       | 指定附加的本地网络                            |
| -r<文件>   | 不从网络上监听 ARP 信息，而是从文件中读取 ARP 的记录信息    |
| -u       | 指定用户和用户组                             |
| -e       | arpwatch 将发送邮件到指定用户，而不是默认的用户 root    |
| -s       | arpwatch 将指定的用户名作为返回地址，而不是默认的用户 root |

### 【系统报告信息】

- ethernet broadcast：主机的 MAC 地址是广播地址。
- ip broadcast：主机的 IP 地址是广播地址。
- bogon：源 IP 地址不是本地子网地址。
- ethernet mismatch：源 MAC 地址与 ARP 数据包里的地址不匹配。
- reused old ethernet address：MAC 地址发生变化。
- suppressed DECnet flip flop：禁止“flip flop”报告。

### ◎ 范例

- 范例 566 ○ 监听网卡 eth0 的 ARP 信息。

```
root@localhost:~# arpwatch -i eth0 //监听网卡 eth0 上的 ARP 信息
```

- 范例 567 ○ 监听 ARP 信息，将相关信息记录到相应的文件。

```
root@localhost:~# arpwatch -i eth0 -f 1.log
//监听网卡 eth0，将监听到的信息记录到 1.log 文件中
```

### ◎ 相关指令

arp、arpsnmp、bpf、tcpdump、pcapture、pcap

## 7.4 arping 指令：向邻居主机发送 ARP 请求

 arping[必要参数][选择性参数][目的地址]

 在指定网卡上发送 ARP 请求到指定地址，源地址由“-s”参数指定。



### 参数说明

必要参数的具体说明如下：

| 参数 | 功能              |
|----|-----------------|
| -A | ARP 回复模式，更新邻居   |
| -b | 保持广播            |
| -D | 复制地址检测模式        |
| -f | 得到第一个回复时就退出     |
| -h | 帮助信息            |
| -q | 运行指令时不显示调试信息    |
| -U | 主动的 ARP 模式，更新邻居 |
| -V | 版本信息            |

选择性参数的具体说明如下：

| 参数         | 功能                      |
|------------|-------------------------|
| -c<数据包的数目> | 发送的数据包的数目               |
| -w<超时时间>   | 设置超时时间                  |
| -I<网卡>     | 使用指定的以太网设备，默认情况下使用 eth0 |
| -s<IP 地址>  | 指定源 IP 地址               |

### 跟踪实例

○范例 568 ○ 向指定主机发送 ARP 请求。

```
[root@localhost ~]# arping 192.168.88.2
//向 IP 地址为 192.168.88.2 主机发送 ARP 请求
ARPING 192.168.88.2 from 192.168.88.76 eth0
Unicast reply from 192.168.88.2 [00:13:46:72:98:E6] 3.310ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 37.110ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 126.480ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 134.795ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 96.589ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 176.594ms
Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 112.922ms
Sent 6 probes (1 broadcast(s))
```

## 7.4 arping 指令：向邻居主机发送 ARP 请求

Received 7 response(s)

[root@localhost ~]#

当运行以上指令，向指定主机发送 ARP 数据包，当不是人为终止时，本地主机将一直不停地向指定主机发送 ARP 数据包。

◎ 范例 569 ◎ 向指定主机发送 ARP 请求，当收到第一个 ARP 回复包时，退出。

[root@localhost ~]# arping -f 192.168.88.2

//向指定主机发送 ARP 请求包，收到第一个包后退出

ARPING 192.168.88.2 from 192.168.88.76 eth0

Unicast reply from 192.168.88.2 [00:13:46:72:98:E6] 0.954ms

Sent 1 probes (1 broadcast(s))

Received 1 response(s)

[root@localhost ~]#

◎ 范例 570 ◎ 向指定主机发送 ARP 请求，并设定发送的 ARP 包的数目为 5。

[root@localhost ~]# arping -c 5 192.168.88.2

//向指定主机发送 ARP 数据包，发送数据包的个数为 5

ARPING 192.168.88.2 from 192.168.88.76 eth0

Unicast reply from 192.168.88.2 [00:13:46:72:98:E6] 15.842ms

Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 144.305ms

Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 96.410ms

Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 127.986ms

Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 157.652ms

Unicast reply from 192.168.88.2 [00:E0:4C:E0:48:38] 181.159ms

Sent 5 probes (1 broadcast(s))

Received 6 response(s)

[root@localhost ~]#

◎ 范例 571 ◎ 向指定主机发送数据包，但不输出相关的调试信息。

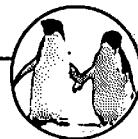
[root@localhost ~]# arping -q 192.168.88.2

//向主机 192.168.88.2 发送 ARP 包但不输出调试信息

使用“-q”参数将不输出调试信息，如果不是人为地终止，本地主机将不停地向指定主机发送 ARP 请求数据包。



ping、clockdiff、tracepath



## 7.5 cu 指令：主机间通信



cu[必要参数][选择性参数]



cu 指令用来进行主机间的通信。



必要参数的具体说明如下：

| 参数      | 功能              |
|---------|-----------------|
| -d      | 排错              |
| -e      | 双同位检查           |
| -h      | 半双工模式           |
| -n      | 交互模式，等待用户输入电话号码 |
| -o      | 单同位检查           |
| -t      | 用“LF+CR”代替CR    |
| -nostop | 关闭流量控制          |

选择性参数的具体说明如下：

| 参数           | 功能            |
|--------------|---------------|
| -a<端口号>      | 指定端口号         |
| -c<电话号码>     | 指定电话号码进行拨打    |
| -E<脱离字符>     | 指定脱离字符        |
| -f<配置文件>     | 从指定的配置文件中读取文件 |
| -l<设备号>      | 指定连接设备        |
| -s<速率>       | 设置连线速率        |
| -v           | 版本信息          |
| -x<排错模式>     | 指定排错模式        |
| -z<主机>       | 指定连接的主机       |
| -help        | 帮助信息          |
| -parity=none | 关闭同位检测        |

**执行范例**

◎ 范例 572 ◎ 与远端主机连线。

[root@localhost temp]# cu -c 51467519 //与远端主机连线

本例使用指令“cu -c 51467519”与远端主机连线。

**相关指令**

mesg

## 7.6 dip 指令：IP 拨号连接

**功能说明** dip[必要参数][选择性参数][脚本文件]

**功能说明** dip 指令用来控制调制解调器，建立拨号连接。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功    能          |
|----|-----------------|
| -a | 用户名和密码          |
| -i | 开启拨号服务器         |
| -k | 杀死 dip 进程       |
| -l | 与“-k”一起使用杀死指定连线 |
| -m | 最大传输单元， 默认为 296 |
| -t | dip 指令模式        |
| -v | 运行时显示详细的处理信息    |

选择性参数的具体说明如下：

| 参数       | 功    能 |
|----------|--------|
| -p<通信协议> | 通信协议   |

**执行范例**

◎ 范例 573 ◎ 建立拨号连接。

[root@localhost ~]# dip -t //建立拨号连接



本例使用指令“dip -t”建立拨号连接。



ppp

## 7.7 gaim 指令：即时信息传输

 gaim[必要参数][选择性参数]

 gaim 指令用于即时信息的传输。



必要参数的具体说明如下：

| 参数     | 功 能            |
|--------|----------------|
| -a     | 开启账号编辑器而不是登录窗口 |
| -debug | 调试信息           |

选择性参数的具体说明如下：

| 参数       | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



◎ 范例 574 ◎ 启动 gaim。

[root@localhost root]# gaim //启动 gaim

本例使用指令 gaim 启动 gaim。



dmesg、wall、write

## 7.8 getty 指令：设置终端配置

 getty[必要参数][选择性参数][终端号]

## 7.9 host 指令：dns 查询



**功能说明** getty 指令用来开启终端，进行终端初始化，设置终端。



必要参数的具体说明如下：

| 参数 | 功 能    |
|----|--------|
| -h | 自动停止设置 |

选择性参数的具体说明如下：

| 参数     | 功 能    |
|--------|--------|
| -c<文件> | 指定配置文件 |
| -r<时间> | 设置延迟   |
| -t<时间> | 超时设置   |



◎ 范例 575 ◎ 开启终端。

```
[root@localhost littleProg]# getty tty7 //开启终端
```

本例使用指令“getty tty7”开启终端 7。



tty

## 7.9 host 指令：dns 查询



host[必要参数][选择性参数]名字[服务器]

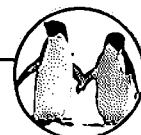


**功能说明** host 指令是一个执行 DNS 查询的简单工具。它通常转换指定的主机名称为 IP 地址。当不指定参数时，它将显示 host 指令的用法。



必要参数的具体说明如下：

| 参数 | 功 能                   |
|----|-----------------------|
| -a | 等用于“-v -t”            |
| -C | 在需要认证的域名服务器上查找 SOA 记录 |



续表

| 参数 | 功能             |
|----|----------------|
| -d | 等同于“-v”        |
| -l | 列出一个域内的所有主机    |
| -i | 反向查找           |
| -N | 改变点数           |
| -r | 不使用递归处理        |
| -R | 指定 UDP 包数      |
| -T | 支持 TCP/IP 模式   |
| -v | 运行时显示详细的处理信息   |
| -w | 永远等待回复         |
| -W | 指定等待回复的时间      |
| -4 | 仅用于 IPv4 的传输查询 |
| -6 | 仅用于 IPv6 的传输查询 |

选择性参数的具体说明如下：

| 参数     | 功能                  |
|--------|---------------------|
| -c<类>  | 查询类                 |
| -t<类型> | 指定类型，包括 a、all、mx、ns |

### 执行范例

○ 范例 576 ○ 显示 host 的帮助信息。

```
[root@localhost shal]# host //host 帮助信息
Usage: host [-aCdIriTwv] [-c class] [-N ndots] [-t type] [-W time]
 [-R number] hostname [server]

-a is equivalent to -v -t *
-c specifies query class for non-IN data
-C compares SOA records on authoritative nameservers
-d is equivalent to -v
-l lists all hosts in a domain, using AXFR
-i IP6.INT reverse lookups
-N changes the number of dots allowed before root lookup is done
```

## 7.9 host 指令: dns 查询

```
-r disables recursive processing
-R specifies number of retries for UDP packets
-t specifies the query type
-T enables TCP/IP mode
-v enables verbose output
-w specifies to wait forever for a reply
-W specifies how long to wait for a reply
-4 use IPv4 query transport only
-6 use IPv6 query transport only
[root@localhost shal]#
```

直接输入 host, 按“Enter”键, 将输出 host 的帮助信息。

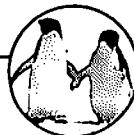
◎ 范例 577 ◎ DNS 查询。

```
[root@localhost shal]# host www.bjtu.edu.cn //查询主机名称
www.bjtu.edu.cn has address 202.112.144.31
[root@localhost shal]# host 202.112.144.31 //查询 IP 地址
Host 31.144.112.202.in-addr.arpa not found: 3(NXDOMAIN)
[root@localhost shal]#
```

本例首先使用指令“host www.bjtu.edu.cn”查询主机名称, 然后使用指令“host 202.112.144.31”查询 IP 地址。

◎ 范例 578 ◎ 查询详细的信息。

```
[root@localhost shal]# host -a www.bjtu.edu.cn //查询详细的 DNS 信息
Trying "www.bjtu.edu.cn"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50411
;; flags:qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2
;; QUESTION SECTION:
;www.bjtu.edu.cn. IN ANY
;; ANSWER SECTION:
www.bjtu.edu.cn. 10800 IN A 202.112.144.31
;; AUTHORITY SECTION:
bjtu.edu.cn. 10800 IN NS jupiter.njtu.edu.cn.
bjtu.edu.cn. 10800 IN NS mars.njtu.edu.cn.
;; ADDITIONAL SECTION:
mars.njtu.edu.cn. 10800 IN A 202.112.144.30
jupiter.njtu.edu.cn. 10800 IN A 202.112.144.65
Received 127 bytes from 202.112.144.30#53 in 10 ms
[root@localhost shal]#
```



本例使用指令“host -a www.bjtu.edu.cn”查询详细的 DNS 信息，可以看到有两个 DNS 服务器。

◎ 范例 579 ◎ 指定 DNS 服务器进行查询。

```
[root@localhost shal]# host -a www.bjtu.edu.cn jupiter.njtu.edu.cn
//指定服务器进行 DNS 查询
```

```
Trying "www.bjtu.edu.cn"
Using domain server:
Name: jupiter.njtu.edu.cn
Address: 202.112.144.65#53
Aliases:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15236
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 5, ADDITIONAL:7
;; QUESTION SECTION:
;www.bjtu.edu.cn. IN ANY
;; AUTHORITY SECTION:
edu.cn. 85190 IN NS NS2.NET.edu.cn.
edu.cn. 85190 IN NS NS2.CUHK.HK.
edu.cn. 85190 IN NS dns2.edu.cn.
edu.cn. 85190 IN NS DENEBCDFN.DE.
edu.cn. 85190 IN NS DNS.edu.cn.
;; ADDITIONAL SECTION:
DNS.edu.cn. 171217 IN A 202.112.0.35
DNS.edu.cn. 171217 IN AAAA 2001:da8:1:100::20
NS2.NET.edu.cn. 84817 IN A 202.112.0.33
NS2.NET.edu.cn. 84817 IN AAAA 2001:250:c006::2
NS2.CUHK.HK. 55619 IN A 137.189.6.21
dns2.edu.cn. 85190 IN A 202.112.0.13
DENEBCDFN.DE. 84817 IN A 192.76.176.9
Received 279 bytes from 202.112.144.65#53 in 10 ms
```

```
[root@localhost shal]#
```

本例使用指令“host -a www.bjtu.edu.cn jupiter.njtu.edu.cn”指定 DNS 服务器为 jupiter.njtu.edu.cn 查询主机 www.bjtu.edu.cn。

◎ 范例 580 ◎ 按照类型进行查询。

```
[root@localhost shal]# host -t MX bjtu.edu.cn //查询邮件交换机的记录
bjtu.edu.cn mail is handled by 10 smg.njtu.edu.cn.
[root@localhost shal]#
```

## 7.10 httpd 指令：apache http 服务器程序

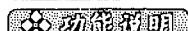
本例使用指令“host -t MX bjtu.edu.cn”查询邮件交换机的记录。

## 相关指令

dig、named、nslookup

## 7.10 httpd 指令：apache http 服务器程序

 **语法** httpd[选择性参数][必要参数]

 **功能说明** httpd 是一个 apache 超文本传输协议服务器程序。它作为一个单独的守护进程来运行。当使用时，它会创建一个子进程池或者多线程来处理请求。

## 相关参数说明

必要参数的具体说明如下：

| 参数 | 功    能    |
|----|-----------|
| -V | 显示编译设置    |
| -l | 显示编译模块    |
| -L | 显示可用的配置文件 |
| -t | 检查配置文件的语法 |

选择性参数的具体说明如下：

| 参数             | 功    能      |
|----------------|-------------|
| -D<名字>         | 定义使用名称      |
| -d<文件夹>        | 指定可选的服务启动   |
| -f<文件>         | 指定可选的服务配置文件 |
| -C “directive” | 读取文件前处理指示   |
| -c “directive” | 读取文件后处理指示   |
| -e<等级>         | 显示启动错误等级    |
| -E<文件>         | 将错误信息输出到文件  |


◆ 执行范例

## ○ 范例 581 ○ 检查语法配置文件的错误。

```
[root@localhost shal]# httpd -t //检查配置文件的语法错误
httpd: Could not determine the server's fully qualified domain name, using
127.0.0.1 for ServerName
Syntax OK
```

本例使用指令“httpd -t”检查配置文件的语法错误。

## ○ 范例 582 ○ 启动 httpd。

```
[root@localhost shal]# ps -Algrep httpd //查找 httpd 进程
[root@localhost shal]# httpd //启动 httpd
httpd: Could not determine the server's fully qualified domain name, using
127.0.0.1 for ServerName
[root@localhost shal]# ps -Algrep httpd //查找 httpd 进程
 6464 ? 00:00:00 httpd
 6467 ? 00:00:00 httpd
 6468 ? 00:00:00 httpd
 6469 ? 00:00:00 httpd
 6470 ? 00:00:00 httpd
 6471 ? 00:00:00 httpd
 6472 ? 00:00:00 httpd
 6473 ? 00:00:00 httpd
 6474 ? 00:00:00 httpd
```

本例使用指令 httpd 启动 httpd 进程。

## ○ 范例 583 ○ 显示编译模块。

```
[root@localhost shal]# httpd -l //显示编译模块
Compiled in modules:
 core.c
 prefork.c
 http_core.c
 mod_so.c
[root@localhost shal]#
```

## 7.11 ifconfig 指令：显示或者配置网络设备

本例使用指令“httpd -l”显示编译模块。

### ◎ 范例 584 ◎ 显示配置文件。

```
[root@localhost shal]# httpd -L >1.log|tail -n 20 1.log
//显示配置文件信息，输出最后 20 行
Maximum number of children alive at the same time
Allowed in *.conf only outside <Directory>, <Files> or <Location>
ServerLimit (prefork.c)
Maximum value of MaxClients for this run of Apache
Allowed in *.conf only outside <Directory>, <Files> or <Location>
KeepAliveTimeout (http_core.c)
Keep-Alive timeout duration (sec)
Allowed in *.conf only outside <Directory>, <Files> or <Location>
MaxKeepAliveRequests (http_core.c)
Maximum number of Keep-Alive requests per connection, or 0 for infinite
Allowed in *.conf only outside <Directory>, <Files> or <Location>
KeepAlive (http_core.c)
Whether persistent connections should be On or Off
Allowed in *.conf only outside <Directory>, <Files> or <Location>
LoadModule (mod_so.c)
a module name and the name of a shared object file to load it from
Allowed in *.conf only outside <Directory>, <Files> or <Location>
LoadFile (mod_so.c)
shared object file or library to load into the server at runtime
Allowed in *.conf only outside <Directory>, <Files> or <Location>
```

[root@localhost shal]#

本例使用指令“httpd -L >1.log|tail -n 20 1.log”显示配置文件的信息，将结果重定向到文件 1.log，并显示文件的最后 20 行信息。



httpd.worker

## 7.11 ifconfig 指令：显示或者配置网络设备



ifconfig[网络设备][必要参数][选择性参数]



 **功能说明** ifconfig 指令用来配置网络设备。当系统启动时，将调用此指令配置网络设备；此外，当进行调试或者系统的网络环境发生变化时，将调用此指令进行相应的网络配置。



必要参数的具体说明如下：

| 参数        | 功能                                                 |
|-----------|----------------------------------------------------|
| down      | 关闭指定网卡                                             |
| up        | 启动指定网卡                                             |
| -arp      | 设置指定的网卡是否支持 ARP 协议                                 |
| -promisc  | 设置是否支持网卡的 promiscous 模式，如果选择此参数，网卡将接收网络中发给它的所有的数据包 |
| -allmulti | 设置是否支持多播模式，如果选择此参数，网卡将接收网络中所有的多播数据包                |

选择性参数的具体说明如下：

| 参数                | 功能               |
|-------------------|------------------|
| add<IPv6 地址>      | 给指定网卡配置 IPv6 地址  |
| del<IPv6 地址>      | 删除指定网卡的 IPv6 地址  |
| <硬件地址>            | 配置网卡的硬件地址        |
| mtu<字节数>          | 设置网卡的最大传输单元      |
| netmask<子网掩码>     | 设置网卡的子网掩码        |
| tunnel<IP 地址>     | 建立隧道             |
| dstaddr<IP 地址>    | 设定一个远端地址，建立点对点通信 |
| -broadcast<地址>    | 为指定网卡设置广播协议      |
| -pointtopoint<地址> | 为指定网卡设置点对点通信协议   |
| multicast         | 为网卡设置组播标志        |
| <IP 地址>           | 为网卡设置 IPv4 地址    |
| txqueuelen<长度>    | 为网卡设置传输队列的长度     |

## 7.11 ifconfig 指令：显示或者配置网络设备

### 执行范例

- 范例 585 ○ 显示当前网络设备的信息。

```
[root@localhost ~]# ifconfig //显示当前网络设备的信息
eth0 Link encap:Ethernet HWaddr 00:0C:29:32:45:1C
 inet addr:192.168.88.76 Bcast:192.168.88.255 Mask:255.255.255.0
 inet6 addr: 2001:da8:205:130:20c:29ff:fe32:451c/64 Scope:Global
 inet6 addr: fe80::20c:29ff:fe32:451c/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:12508 errors:0 dropped:0 overruns:0 frame:0
 TX packets:100 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:1349900 (1.2 MiB) TX bytes:6807 (6.6 KiB)
 Interrupt:10 Base address:0x1080

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:1618 errors:0 dropped:0 overruns:0 frame:0
 TX packets:1618 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:1753452 (1.6 MiB) TX bytes:1753452 (1.6 MiB)
```

- 范例 586 ○ 为网卡配置地址信息。

```
[root@localhost ~]# ifconfig eth0 192.168.88.7 netmask 255.255.255.0
//为网卡eth0配置IPv4地址和子网掩码
```

- 范例 587 ○ 启动/关闭指定网卡。

```
[root@localhost ~]# ifconfig eth0 down //关闭网卡eth0
[root@localhost ~]# ifconfig eth0 up //开启网卡eth0
```

- 范例 588 ○ 为网卡配置和删除 IPv6 地址。

```
[root@localhost ~]# ifconfig eth0 add 3ffe:3240:8007:1005::2/64 //为网卡eth0配置IPv6地址
[root@localhost ~]# ifconfig eth0 del 3ffe:3240:8007:1005::2/64 //删除网卡eth0的IPv6地址
```


相关指令

route、netstat、arp、iwconfig

## 7.12

### iptables 指令：IPv4 的包过滤和 nat 的管理

语法

**iptables[-t 表][必要参数][选择性参数]**

功能说明

**iptables** 指令用来设置、维护或者检测 Linux 内核的 IP 数据包表，并且定义了不止一种数据表。每一个表包含一定数量的路由链，也可能维护用户定义的路由链。每一个路由链定义了相应的接收数据包的规则，表的类型如下所示。

- filter：信息包过滤表。
- mangle：高级路由表。
- nat：转发包过滤表。

参数说明

必要参数的具体说明如下：

| 参 数     | 功 能     |
|---------|---------|
| -A<路由链> | 增加路由链   |
| -D<路由链> | 删除路由链   |
| -F<路由链> | 清除路由链规则 |
| -I<路由链> | 新建路由链规则 |
| -L      | 列出表信息   |
| -N<路由链> | 新建路由链   |
| -X<路由链> | 删除路由链   |

选择性参数的具体说明如下：

| 参 数    | 功 能  |
|--------|------|
| -d<地址> | 目的地址 |
| -i<网卡> | 选择网卡 |

## 7.12 iptables 指令：IPv4 的包过滤和 nat 的管理

续表

| 参数       | 功能     |
|----------|--------|
| -o<网卡>   | 退出时的网卡 |
| -p<通信协议> | 选择通信协议 |
| -s<地址>   | 源地址    |

## ☆ 实际范例

## ◎ 范例 589 ◎ 显示信息包过滤表。

```
root@localhost:~# iptables -t filter -L //显示信息包过滤表
```

## Chain INPUT (policy ACCEPT)

| target | prot | opt | source   | destination                        |
|--------|------|-----|----------|------------------------------------|
| ACCEPT | 0    | --  | anywhere | anywhere                           |
| ACCEPT | 0    | --  | anywhere | anywhere                           |
| ACCEPT | icmp | --  | anywhere | anywhere                           |
| ACCEPT | udp  | --  | anywhere | anywhere dpt:2304                  |
| ACCEPT | tcp  | --  | anywhere | anywhere dpts:ftp:telnet           |
| ACCEPT | tcp  | --  | anywhere | anywhere dpt:www                   |
| ACCEPT | tcp  | --  | anywhere | anywhere dpt:8888                  |
| ACCEPT | 0    | --  | anywhere | anywhere state RELATED,ESTABLISHED |

## Chain FORWARD (policy ACCEPT)

| target | prot | opt | source          | destination                        |
|--------|------|-----|-----------------|------------------------------------|
| ACCEPT | 0    | --  | anywhere        | 192.168.88.88                      |
| ACCEPT | 0    | --  | 192.168.88.0/24 | anywhere                           |
| ACCEPT | 0    | --  | anywhere        | anywhere state RELATED,ESTABLISHED |

## Chain OUTPUT (policy ACCEPT)

| target | prot | opt | source | destination |
|--------|------|-----|--------|-------------|
|--------|------|-----|--------|-------------|

```
root@localhost:~#
```

本例使用指令“iptables -t filter -L”显示信息包过滤表。

## ◎ 范例 590 ◎ 显示高级路由表。

```
root@localhost:~# iptables -t mangle -L //显示高级路由表
```

## Chain PREROUTING (policy ACCEPT)

| target | prot | opt | source | destination |
|--------|------|-----|--------|-------------|
|--------|------|-----|--------|-------------|

## Chain INPUT (policy ACCEPT)



```

target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
Chain POSTROUTING (policy ACCEPT)
target prot opt source destination
root@localhost:~#

```

本例使用指令“`iptables -t mangle -L`”显示主机的高级路由表。

○ 范例 591 ○ 显示转发包过滤表。

```

root@localhost:~# iptables -t nat -L //显示转发包过滤表
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
DNAT tcp -- anywhere iplab604 tcp dpt:3389 to:192 .168.88.88:3389
Chain POSTROUTING (policy ACCEPT)
target prot opt source destination
MASQUERADE 0 -- 192.168.88.0/24 anywhere
SNAT 0 --192.168.88.88 anywhere to:202.112.146.70
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
root@localhost:~#

```

本例使用指令“`iptables -t nat -L`”显示转发包过滤表信息。

○ 范例 592 ○ INPUT 链设置。

```
[root@localhost ~]# iptables -P INPUT DROP //设置 INPUT 链的规则
```

本例使用指令“`iptables -P INPUT DROP`”设置 INPUT 链的默认的规则为 DROP，与 INPUT 链的规则不匹配的数据包将被丢弃。

○ 范例 593 ○ OUTPUT 链设置。

```
[root@localhost ~]# iptables -P OUTPUT DROP //设置 OUTPUT 链的规则
```

本例使用指令“`iptables -P OUTPUT DROP`”设置 OUTPUT 链的默认规则为 DROP 与 OUTPUT 链的规则不匹配的数据包将被丢弃。

○ 范例 594 ○ FORWARD 链设置。

```
[root@localhost ~]# iptables -P FORWARD DROP //设置 FORWARD 链的规则
```

本例使用指令“`iptables -P FORWARD DROP`”设置 FORWARD 链的默认的规则为 DROP，与 FORWARD 链的规则不匹配的数据包将被

## 7.12 iptables 指令：IPv4 的包过滤和 nat 的管理

丢弃。

◎ 范例 595 ◎ 添加规则。

```
[root@localhost ~]# iptables -A INPUT -s 192.168.88.7 -j ACCEPT
//添加规则
[root@localhost ~]#
```

本例使用指令“iptables -A INPUT -s 192.168.88.7 -j ACCEPT”添加规则，接收源地址是 192.168.88.7 的数据包。

```
[root@localhost ~]# iptables -A INPUT -s 192.168.88.7 -j DROP
//添加规则
[root@localhost ~]#
```

然后使用指令“iptables -A INPUT -s 192.168.88.7 -j DROP”添加规则，丢弃源地址是 192.168.88.7 的数据包。

◎ 范例 596 ◎ 删除规则。

```
[root@localhost ~]# iptables -D INPUT --dport 80 -j DROP
//删除规则
[root@localhost ~]#
```

本例使用指令“iptables -D INPUT --dport 80 -j DROP”删除 INPUT 链中阻塞 80 端口的数据包的规则。

```
[root@localhost ~]# iptables -D INPUT 3 //删除规则
[root@localhost ~]#
```

然后使用指令“iptables -D INPUT 3”，删除 INPUT 链中的第 3 条规则。

◎ 范例 597 ◎ 插入规则。

```
[root@localhost ~]# iptables -I INPUT 1 -s 192.168.88.7 -j ACCEPT
//插入规则
```

本例使用指令“iptables -I INPUT 1 -s 192.168.88.7 -j ACCEPT”在 INPUT 链中的 1 号规则的位置上插入一条新的规则，新规则的内容是源于 192.168.88.7 的数据包。

◎ 范例 598 ◎ 更换规则。

```
[root@localhost ~]# iptables -R INPUT 1 -s 192.168.88.7 -j ACCEPT
//更换规则
```

本例使用指令“iptables -R INPUT 1 -s 192.168.88.7 -j ACCEPT”更换 INPUT 链中的 1 号规则为新的规则，新规则的内容是源于 192.168.88.7 的数据包。


相关指令

`iptables-save`、`iptables-restore`、`ip6tables`、`ip6tables-save`、`ip6tables-restore`

## 7.13

### iptables-save 指令：IP 列表存储

执行范例

`iptables-save[-c][-t 表]`

功能说明

`iptables-save` 指令用来输出 `iptable` 到标准输出。

参数说明

必要参数的具体说明如下：

| 参数              | 功 能         |
|-----------------|-------------|
| <code>-c</code> | 包含当前所有数据包的值 |

选择性参数的具体说明如下：

| 参 数                      | 功 能          |
|--------------------------|--------------|
| <code>-t&lt;表&gt;</code> | 指定 table 表类型 |

执行范例

○ 范例 599 ○ 显示 `iptable` 设置。

```
root@localhost:~# iptables-save //显示 iptable 设置
Generated by iptables-save v1.3.6 on Thu May 24 04:40:16 2007
*mangle
:PREROUTING ACCEPT [834359:490508468]
:INPUT ACCEPT [66789:18462492]
:FORWARD ACCEPT [613800:454909818]
:OUTPUT ACCEPT [24584:15999458]
:POSTROUTING ACCEPT [638382:470909124]
COMMIT
Completed on Thu May 24 04:40:16 2007
Generated by iptables-save v1.3.6 on Thu May 24 04:40:16 2007
*nat
:PREROUTING ACCEPT [71744143:7018491963]
:POSTROUTING ACCEPT [32953:1853286]
```

## 7.13 iptables-save 指令：IP 列表存储

```

:OUTPUT ACCEPT [32163:1791912]
-A PREROUTING -d 202.112.146.70 -p tcp -m tcp --dport 3389 -j DNAT
--to-destination 192.168.88.88:3389
-A POSTROUTING -s 192.168.88.0/255.255.255.0 -o eth1 -j MASQUERADE
-A POSTROUTING -s 192.168.88.88 -j SNAT --to-source 202.112.146.70
COMMIT
Completed on Thu May 24 04:40:16 2007
Generated by iptables-save v1.3.6 on Thu May 24 04:40:16 2007
*filter
:INPUT ACCEPT [26710398:9570338513]
:FORWARD ACCEPT [15991:19168654]
:OUTPUT ACCEPT [4969768:2145883292]
-A INPUT -i lo -j ACCEPT
-A INPUT -i eth0 -j ACCEPT
-A INPUT -i eth1 -p icmp -j ACCEPT
-A INPUT -i eth1 -p udp -m udp --dport 2304 -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 21:23 -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 8888 -j ACCEPT
-A INPUT -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -d 192.168.88.88 -j ACCEPT
-A FORWARD -s 192.168.88.0/255.255.255.0 -j ACCEPT
-A FORWARD -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT
Completed on Thu May 24 04:40:16 2007
root@localhost:~#

```

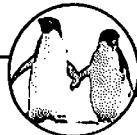
本例使用指令“iptables-save”显示当前 iptables 的设置。

○ 范例 600 ○ 显示 filter 表的设置。

```

root@localhost:~# iptables-save -t filter //filter 表的设置
Generated by iptables-save v1.3.6 on Thu May 24 04:44:31 2007
*filter
:INPUT ACCEPT [26711145:9570386824]
:FORWARD ACCEPT [15991:19168654]
:OUTPUT ACCEPT [4970272:2145916938]
-A INPUT -i lo -j ACCEPT
-A INPUT -i eth0 -j ACCEPT
-A INPUT -i eth1 -p icmp -j ACCEPT
-A INPUT -i eth1 -p udp -m udp --dport 2304 -j ACCEPT

```



```

-A INPUT -i eth1 -p tcp -m tcp --dport 21:23 -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 8888 -j ACCEPT
-A INPUT -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -d 192.168.88.88 -j ACCEPT
-A FORWARD -s 192.168.88.0/255.255.255.0 -j ACCEPT
-A FORWARD -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
COMMIT
Completed on Thu May 24 04:44:31 2007
root@localhost:~#

```

本例使用指令“iptables-save -t filter”显示filter表的设置。

○范例 601 ○ 显示 mangle 表的设置。

```

root@localhost:~# iptables-save -t mangle //显示mangle表的设置
Generated by iptables-save v1.3.6 on Thu May 24 04:46:08 2007
*mangle
:PREROUTING ACCEPT [851347:495665608]
:INPUT ACCEPT [68335:18578218]
:FORWARD ACCEPT [624216:459405852]
:OUTPUT ACCEPT [25298:16048556]
:POSTROUTING ACCEPT [649512:475454256]
COMMIT
Completed on Thu May 24 04:46:08 2007
root@localhost:~#

```

本例使用指令“iptables-save -t mangle”显示当前 mangle 表的设置。

○范例 602 ○ 显示 nat 表的设置。

```

root@localhost:~# iptables-save -t nat //显示nat表的设置
Generated by iptables-save v1.3.6 on Thu May 24 04:47:19 2007
*nat
:PREROUTING ACCEPT [71751530:7019235259]
:POSTROUTING ACCEPT [32954:1853338]
:OUTPUT ACCEPT [32164:1791964]
-A PREROUTING -d 202.112.146.70 -p tcp -m tcp --dport 3389 -j DNAT
--to-destination 192.168.88.88:3389
-A POSTROUTING -s 192.168.88.0/255.255.255.0 -o eth1 -j MASQUERADE
-A POSTROUTING -s 192.168.88.88 -j SNAT --to-source 202.112.146.70
COMMIT
Completed on Thu May 24 04:47:19 2007
root@localhost:~#

```

## 7.14 iwconfig 指令：配置无线网络设备

本例使用指令“iptables-save -t nat”显示当前 nat 表的设置。



**iptables**

## 7.14 iwconfig 指令：配置无线网络设备

**iwconfig[网络设备][必要参数][选择性参数]**

**功能说明** iwconfig 为系统配置无线网络设备或显示无线网络设备信息。iwconfig 指令类似与 ifconfig 指令，但是它配置的对象是无线网卡。它将对网络设备进行无线操作，如设置无线通信频段。



必要参数的具体说明如下：

| 参 数      | 功 能                     |
|----------|-------------------------|
| essid    | 设置 ESSID (网络的名称)        |
| nwid     | 设置网络 ID                 |
| freq     | 设置无线网络通信频段              |
| channel  | 设置无线网络通信频段              |
| sens     | 设置无线网络设备的感知阈值           |
| mode     | 设置无线网络设备的通信模式           |
| ap       | 强迫无线网卡向给定地址的接入点注册       |
| nick<名字> | 为网卡设定别名                 |
| rate<速率> | 设定无线网卡的传输速率             |
| rts<阀值>  | 在传输数据包之前增加一次握手，确保信道是正常的 |
| power    | 无线网卡的功率设置               |

选择性参数的具体说明如下：

| 参 数      | 功 能  |
|----------|------|
| -help    | 帮助信息 |
| -version | 版本信息 |



### 执行范例

- ◎ 范例 603 ◎ 显示当前的无线网络设备信息。

```
[root@localhost ~]#iwconfig //显示无线网络信息
```

- ◎ 范例 604 ◎ 设置 ESSID。

```
[root@localhost ~]iwconfig eth0 essid any //关闭无线网卡的ESSID检测
```

[root@localhost ~]iwconfig eth0 essid "My Network" //设置无线网卡的ESSID为"My Network"

ESSID是用来表示不同网络通信小区的，当 ESSID 检测开启时，只有 ESSID 相同的网卡才能直接通信。

- ◎ 范例 605 ◎ 设置网络 ID。

```
[root@localhost ~]iwconfig eth0 nwid AB34 //设置网络 ID
```

```
[root@localhost ~]iwconfig eth0 nwid off //关闭网络 ID 设置
```

网络 ID 类似于 ESSID，可以将网络 ID 的设置关闭。

- ◎ 范例 606 ◎ 设置通信频段和通道。

```
[root@localhost ~]iwconfig eth0 freq 2.422G //设置eth0的通信频段为2.422G
```

```
[root@localhost ~]iwconfig eth0 channel 3 //设置eth0的通信通道为3
```

只有在同一频段或者通道的无线网络设备才能直接通信。

### 相关指令

ifconfig

## 7.15 mesg 指令：控制终端的写入

### 基本用法

mesg[必要参数]

功能说明：控制其他用户能否使用 write 指令将信息发送到当前用户使用的终端上。

### 参数说明

必要参数的具体说明如下：

| 参 数 | 功 能             |
|-----|-----------------|
| y   | 允许其他用户发送信息到当前终端 |
| n   | 禁止其他用户发送信息到当前终端 |

## 7.16 mingetty 指令：精简版的 getty

**◎ 实例与练习**

- ◎ 范例 607 ◎ 允许其他用户发送信息到当前终端。

```
[root@localhost root]# mesg y
[yingzheng@yingzheng ~]$ write root pts/4 //在另一个终端上写入
hi
hi //用Ctrl+D结束
[root@yingzheng ~]$ //在被写入终端上输出
Message from yingzheng@yingzheng on pts/3 at 08:57 ...
hi
hiEOF
```

- ◎ 范例 608 ◎ 禁止其他用户发送信息到当前终端。

```
[root@localhost root]# mesg n //首先在一个终端上设置为禁止
[yingzheng@yingzheng ~]$ write root pts/4//另一个终端上用write 写入被禁止
write: root has messages disabled on pts/4
```

**◎ 相关指令**

write、talk、who

## 7.16 mingetty 指令：精简版的 getty

**◎ 概述** mingetty[必要参数][选择性参数][tty]

**◎ 功能说明** 适用于本机上的登录程序，不适合串口。

**◎ 参数说明**

必要参数的具体说明如下：

| 参数             | 功能                                   |
|----------------|--------------------------------------|
| -noclear       | 在没有得到登录名的提示下不清除屏幕                    |
| -newline       | 在显示版本号之前不新开一行                        |
| -noissuc       | 不显示/etc/issue                        |
| -nohangup      | 不调用vhangup()函数禁止其他应用程序向这个串行终端(tty)写入 |
| -nohostname    | 在登录提示符前不显示主机名称                       |
| -long-hostname | 完整显示通过gethostname函数得到的全部文本           |



选择性参数的具体说明如下：

| 参数                  | 功能                       |
|---------------------|--------------------------|
| -loginprog<程序可执行文件> | 更换登录的应用程序                |
| -nice=<数字>          | 通过调用 nice 更改数字指定的优先级     |
| -delay=<数字>         | 在启动登录程序前休眠指定的秒数          |
| -chdir=<路径>         | 在登录前进入指定路径               |
| -chroot=<路径>        | 通过调用 chroot 更改登录的根目录的目录名 |



mgetty、agetty

## 7.17 minicom 指令：调制解调器通信程序



minicom[必要参数][选择性参数]



一个开源的友好的串行通信程序。



必要参数的具体说明如下：

| 参数         | 功能                                   |
|------------|--------------------------------------|
| -s         | 设置程序运行时直接使用/etc/minirc.dfl 中的默认设置初始化 |
| -o         | 程序运行时跳过初始化代码                         |
| -m         | 设置“Alt”或“Meta”键作为特殊指令键               |
| -M         | 与“-m”类似                              |
| -z         | 使用终端状态列                              |
| -l         | 有些字符将被直接显示而不是转化为 ASCII 码             |
| -w         | 在默认情况下开启行限制功能                        |
| -a<on off> | 设置终端机属性开关                            |
| -t<类型>     | 设置终端机类型                              |
| -c<on off> | 设置终端颜色模式的开关                          |

续表

| 参数     | 功能               |
|--------|------------------|
| -S<文件> | 使用指定脚本开始运行       |
| -d<实体> | 指定程序启动就向往指定的实体拔号 |
| -p<终端> | 使用伪终端            |
| -C<文件> | 启动时打开指定的文件       |
| -8     | 无修改地按 8 位传输      |

选择性参数的具体说明如下：

| 参数   | 功能     |
|------|--------|
| 配置文件 | 指定配置文件 |

### 相关指令

mingetty

## 7.18 mkfifo 指令：创建管道

 **语法** mkfifo[选择性参数][必要参数]

 **功能说明** 根据指定的名字创建 FIFO 管道。

 **参数说明**

必要参数的具体说明如下：

| 参数     | 功能     |
|--------|--------|
| <管道名称> | 指定管道名称 |

选择性参数的具体说明如下：

| 参数          | 功能           |
|-------------|--------------|
| --mode=<模式> | 设置创建的管道的权限模式 |
| --help      | 打印帮助信息       |
| --version   | 打印版本信息       |



### ◆ 执行范例

◎ 范例 609 ◎ 创建一个只读权限的管道。

```
[yingzheng@yingzheng ~]$ ls -l
```

总用量 4

```
pr---r--r-- 1 yingzheng yingzheng 0 5月 20 03:06 newfifo
```

### ◆ 相关指令

info coreutils mkfifo

## 7.19 mtr 指令：网络诊断工具

◆ 基础 ◆ mtr[必要参数][选择性参数]

◆ 功能说明 ◆ 集中了 traceroute 指令和 ping 指令功能的网络诊断程序。

### ◆ 参数说明

必要参数的具体说明如下：

| 参 数  | 功 能      |
|------|----------|
| <主机> | 指定要诊断的主机 |

选择性参数的具体说明如下：

| 参 数    | 功 能                   |
|--------|-----------------------|
| -h     | 帮助信息                  |
| -v     | 版本信息                  |
| -r     | 报告模式，以“-c”指定的循环数记录统计  |
| -c<数量> | 指定“-r”模式的循环数          |
| -p<字节> | 指定探测包的大小              |
| -n     | 强迫显示数字格式的 IP 地址       |
| -g     | 指定使用基于 X11 窗口的 GTK+接口 |
| -4     | 只使用 IPv4              |
| -6     | 只使用 IPv6              |

**执行范例**

◎ 范例 610 ◎ 诊断主机到 192.168.11.44 之间的网络状况。

```
[root@yingzheng ~]# mtr 192.168.11.44
```

```
My traceroute [v0.69]
```

```
yingzheng (0.0.0.0) (tos=0x0 psize=64 bitpattern=0x00) Mon May 21
```

07:15:28 2007

```
Keys: Help Display mode Restart statistics Order of fields quit
```

```
Last 50 pings
```

```
Scale:.:979 ms 1:949 ms 2:919 ms 3:849 ms a:669 ms b:499 ms c:199 ms
```

**相关指令**

traceroute、ping

## 7.20 nc 指令：设置路由器

**语法** nc[必要参数][选择性参数][主机][端口号]

**功能说明** nc 指令用来设置路由器参数。它可以打开 TCP 链接，发送 UDP 数据包，监听 TCP 和 UDP 端口，扫描端口，同时处理 IPv4 和 IPv6，它实现的功能如下。

- 简单 TCP 代理。
- 基于 HTTP 客户端和服务端的 shell 脚本。
- socks 或者 HTTP 代理指令的 ssh。

**参数说明**

必要参数的具体说明如下：

| 参数 | 功能           |
|----|--------------|
| -l | 监听模式         |
| -n | 使用 IP 地址     |
| -u | UDP          |
| -v | 运行时显示详细的处理信息 |
| -z | 关闭输入/输出      |



选择性参数的具体说明如下：

| 参数      | 功能     |
|---------|--------|
| -g<网关>  | 指定网关   |
| -h      | 帮助信息   |
| -i<时间>  | 延迟时间   |
| -o<文件>  | 指定输出文件 |
| -p<端口号> | 指定端口号  |
| -s<源地址> | 指定源地址  |
| -w<时间>  | 设置超时时间 |

### ◎ 实例范例

#### ◎ 范例 611 ◎ TCP 端口扫描。

```
[root@localhost ~]# nc -v -z -w2 192.168.88.76 1-140//TCP 端口扫描
nc: connect to 192.168.88.76 port 1 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 2 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 3 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 4 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 5 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 6 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 7 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 8 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 9 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 10 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 11 (tcp) failed: Connection refused
.....
nc: connect to 192.168.88.76 port 130 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 131 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 132 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 133 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 134 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 135 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 136 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 137 (tcp) failed: Connection refused
```

## 7.20 nc 指令：设置路由器

```
nc: connect to 192.168.88.76 port 138 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 139 (tcp) failed: Connection refused
nc: connect to 192.168.88.76 port 140 (tcp) failed: Connection refused
[root@localhost ~]#
```

本例使用指令“`nc -v -z -w2 192.168.88.76 1-140`”进行TCP端口的扫描，得到的结果如上所示，有些端口是可以链接的，有些不可以。

## ◎范例612 ◎ UDP端口扫描。

```
[root@localhost ~]# nc -u -v -z -w2 192.168.88.76 1-10000 //UDP端口扫描
Connection to 192.168.88.76 111 port [udp/sunrpc] succeeded!
Connection to 192.168.88.76 631 port [udp/ipp] succeeded!
Connection to 192.168.88.76 641 port [udp/*] succeeded!
Connection to 192.168.88.76 5353 port [udp/*] succeeded!
[root@localhost ~]#
```

本例使用指令“`nc -u -v -z -w2 192.168.88.76 1-10000`”得到的结果如上所示，UDP端口号111、631、641、5353可以连接。

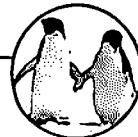
## ◎范例613 ◎ 连接到远端主机的指定端口号。

```
[root@localhost ~]# nc -nvv 192.168.88.2 80 //连接远端主机
Connection to 192.168.88.2 80 port [tcp/*] succeeded!
abcdefg //用户输入
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved here.</p>
<hr>
<address>Apache/2.2.3 (Debian) mod_python/3.2.10 Python/2.4.4
PHP/4.4.4-8 mod_perl/2.0.2 Perl/v5.8.8 Server at 202.112.146.70 Port
80</address>
</body></html>
[root@localhost ~]#
```

本例使用指令“`nc -nvv 192.168.88.2 80`”连接到远端主机192.168.88.2的80端口上。用户输入后回车得到以上结果。



route



## 7.21

# netconfig 指令：设置各项网络功能



netconfig



**功能说明** netconfig 指令用来设置网络环境，它提供图形化界面，可以设置本地主机名称、域名、IP 地址、网络掩码、网关以及域名服务器。

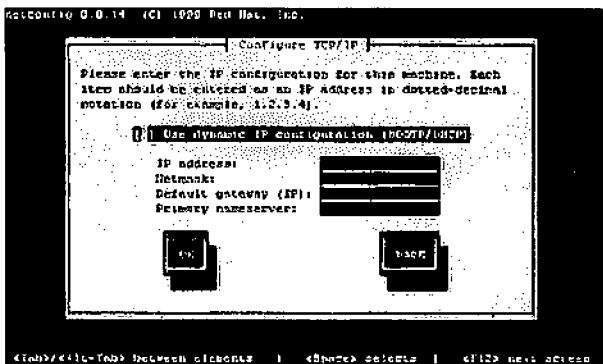


○ 范例 614 ○ 网络设置。

```
[root@localhost ~]# netconfig //网络配置
```



输入指令 netconfig，之后会出现以上的图形界面，选择 Yes，进行本地主机的网络配置，将进入下面的图形界面。



## 7.22 netstat 指令：显示网络状态

可以根据以上的图形界面和其中的提示信息进行本地网络的配置。



setup

## 7.22

### netstat 指令：显示网络状态



**语法** netstat[必要参数][选择性参数]



**功能说明** netstat 指令打印出 Linux 网络子系统的信息。



必要参数的具体说明如下：

参数	功能
-r	显示路由表
-i	显示网卡列表
-g	显示组插组的关系
-s	显示网络统计信息
-M	显示隐藏链接
-v	运行时显示详细的处理信息
-n	不解析名字
--numeric-hosts	不解析主机名
--numeric-port	不解析端口名
--numeric-users	不解析用户名
-N	解析硬件名
-e	显示更多信息
-c	连续列表
-l	显示监听的服务套接口
-a	显示所有的套接口
-o	显示定时器
-F	显示基本发送信息
-C	显示路由缓存



选择性参数的具体说明如下：

参数	功能
-l<网卡>	显示指定网卡列表
-h	帮助信息
-v	版本信息

### 【其他说明】

套接口类型如下。

- t: TCP。
- u: UDP。
- raw: RAW 类型。
- unix: UNIX 域类型。
- ax25: AX25 类型。
- ipx: ipx 类型。
- netrom: netrom 类型。

支持的地址族如下。

- inet: DARPA 网络。
- inet6: IPv6。
- ax25: AMPR AX.25。
- netrom: AMPR NET/ROM。
- ipx: Novell IPX。
- ddp: Appletalk DDP。
- x25: CCITT X.25。

### ● 演示范例

#### ● 范例 615 ● 显示路由表。

```
[root@localhost ~]# netstat -r //显示路由表
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
192.168.88.0 * 255.255.255.0 U 0 0 0 eth0
169.254.0.0 * 255.255.0.0 U 0 0 0 eth0
```

## 7.22 netstat 指令：显示网络状态

```
default 192.168.88.2 0.0.0.0 UG 0 0 0 eth0
[root@localhost ~]#
```

本例使用指令“netstat -r”显示本地主机的路由表。

## ◎ 范例 616 ◎ 显示详细的网络状况。

```
[root@localhost ~]# netstat -a //显示详细的网络状况
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 *:32769 *:* LISTEN
tcp 0 0 *:sunrpc *:* LISTEN
tcp 0 0 localhost.localdomain:ipp *:* LISTEN
tcp 0 0 localhost.localdomain:5335 *:* LISTEN
tcp 0 0 localhost.localdomain:smtp *:* LISTEN
tcp 0 0 *:ssh *:* LISTEN
tcp 0 0 ::ffff:192.168.88.76:ssh ::ffff:192.168.88.7:1526 ESTABLISHED
udp 0 0 *:32768 *:* LISTEN
udp 0 0 *:641 *:* LISTEN
udp 0 0 *:5353 *:* LISTEN
udp 0 0 *:sunrpc *:* LISTEN
udp 0 0 *:ipp *:* LISTEN

Active UNIX domain sockets (servers and established)
Proto RefCnt Flags Type State I-Node Path
unix 2 [ACC] STREAM LISTENING 6512 /tmp/.font-unix/fs7100
unix 2 [ACC] STREAM LISTENING 5604 /var/run/sdp
unix 2 [ACC] STREAM LISTENING 5890 /var/run/acpid.socket
unix 2 [ACC] STREAM LISTENING 7692 /tmp/.gdm_socket
unix 2 [ACC] STREAM LISTENING 6346
 //此处省略了部分结果
unix 3 [] STREAM CONNECTED 5561
unix 3 [] STREAM CONNECTED 5560
unix 2 [] DGRAM 5465
unix 2 [] DGRAM 5421
unix 2 [] DGRAM 5374
[root@localhost ~]#
```

本例使用指令“netstat -a”显示详细的网络状况。

## ◎ 范例 617 ◎ 显示当前主机 UDP 的联机状况。

```
root@localhost:~# netstat -nu //显示 UDP 联机状况
Active Internet connections (w/o servers)
```



## 第7章 网络通信

```
Proto Recv-Q Send-Q Local Address Foreign Address State
udp 0 0 127.0.0.1:32768 127.0.0.1:32768 ESTABLISHED
```

本例使用指令“netstat -nu”显示本地主机 UDP 的连接状况。

○范例 618 ○ 显示 UDP 端口号的使用状况。

```
root@localhost:~# netstat -apu //显示 UDP 端口号使用情况
Active Internet connections (servers and established)
```

```
Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name
udp 0 0 *:2304 *:* 26541/poweron
udp 0 0 localhost:32768 localhost:32768 ESTABLISHED 3723/postmaster
udp 0 0 *:32769 *:* 3891/rpc.statd
udp 0 0 *:32770 *:* 26541/poweron
udp 0 0 *:675 *:* 3891/rpc.statd
udp 0 0 *:sunrpc *:* 3201/portmap
```

本例使用指令“netstat -apu”显示本地主机 UDP 端口号的使用情况。

○范例 619 ○ 显示网卡列表。

```
[root@localhost ~]# netstat -i //显示网卡列表
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 114420 0 0 0 1538 0 0 0 BMRU
lo 16436 0 1752 0 0 0 1752 0 0 0 LRU
```

本例使用指令“netstat -i”显示网卡列表，网卡信息如上所示。

○范例 620 ○ 显示组播组的关系。

```
[root@localhost ~]# netstat -g //显示组播关系
IPv6/IPv4 Group Memberships
Interface RefCnt Group

lo 1 ALL-SYSTEMS.MCAST.NET
eth0 1 224.0.0.251
eth0 1 ALL-SYSTEMS.MCAST.NET
lo 1 ff02::1
eth0 2 ff02::1:ff32:451c
eth0 1 ff02::1
[root@localhost ~]#
```

本例使用指令“netstat -g”显示组播关系。

## 7.22 netstat 指令：显示网络状态

## ◎ 范例 621 ◎ 显示网络统计信息。

```
[root@localhost ~]# netstat -s //显示网络统计信息
```

Ip:

```
61905 total packets received
 0 forwarded
 86 with unknown protocol
 0 incoming packets discarded
 7954 incoming packets delivered
 3276 requests sent out
```

Icmp:

```
2 ICMP messages received
 0 input ICMP message failed.
 ICMP input histogram:
 destination unreachable: 2
 1 ICMP messages sent
 0 ICMP messages failed
 ICMP output histogram:
 destination unreachable: 1
```

Tcp:

```
18 active connections openings
 3 passive connection openings
 0 failed connection attempts
 0 connection resets received
 1 connections established
 4013 segments received
 3186 segments send out
 2 segments retransmitted
 0 bad segments received.
 1 resets sent
```

Udp:

```
185 packets received
 1 packets to unknown port received.
 0 packet receive errors
 88 packets sent
```

TcpExt:

```
8 TCP sockets finished time wait in fast timer
 420 delayed acks sent
 1 delayed acks further delayed because of locked socket
 3 packets directly queued to recvmsg prequeue.
```



```
2049 packets directly received from prequeue
1599 packets header predicted
1 packets header predicted and directly queued to user
631 acknowledgments not containing data received
702 predicted acknowledgments
0 TCP data loss events
2 other TCP timeouts
[root@localhost ~]#
```

本例使用指令“netstat -s”显示网络的统计信息，包括IP、TCP、UDP、ICMP、TCPEXT等数据包的接收信息。

◎ 范例 622 ◎ 显示监听的套接口。

```
[root@localhost ~]# netstat -l //显示监听的套接口
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 *:32769 *:* LISTEN
tcp 0 0 *:sunrpc *:* LISTEN
tcp 0 0 localhost.localdomain:ipp *:* LISTEN
tcp 0 0 localhost.localdomain:5335 *:* LISTEN
tcp 0 0 localhost.localdomain:smtp *:* LISTEN
tcp 0 0 *:ssh *:* LISTEN
udp 0 0 *:32768 *:*
udp 0 0 *:641 *:*
udp 0 0 *:5353 *:*
udp 0 0 *:sunrpc *:*
udp 0 0 *:ipp *:*
Active UNIX domain sockets (only servers)
Proto RefCnt Flags Type State I-Node Path
unix 2 [ACC] STREAM LISTENING 6512 /tmp/.font-unix/fs7100
unix 2 [ACC] STREAM LISTENING 5604 /var/run/sdp
unix 2 [ACC] STREAM LISTENING 5890 /var/run/acpid.socket
unix 2 [ACC] STREAM LISTENING 7692 /tmp/.gdm_socket
unix 2 [ACC] STREAM LISTENING 6346 /var/run/iiim/.iiimp-unix/9010
unix 2 [ACC] STREAM LISTENING 6352 /var/run/.ircha_unix/IROHA
unix 2 [ACC] STREAM LISTENING 6578 /var/run/dbus/system_bus_socket
unix 2 [ACC] STREAM LISTENING 7748 /tmp/.X11-unix/X0
unix 2 [ACC] STREAM LISTENING 6709 @/tmp/hald-local/dbus- EpLlhcleHR
unix 2 [ACC] STREAM LISTENING 6305 /dev/gpmctl
[root@localhost ~]#
```

本例使用指令“netstat -l”显示监听的套接口信息。



route、nc

**7.23****nslookup 指令: dns 查找****nslookup[必要参数][选择性参数][主机名称][服务器名称]**

nslookup 指令是一个查找 Internet 域名服务器的程序。nslookup 有两种模式：交互模式和非交互模式。



必要参数的具体说明如下：

参 数	功 能
exit	退出
host	查找主机信息

选择性参数的具体说明如下：

参 数	功 能
server<服务器>	指定 DNS 服务器
set<关键词>	改变查询信息的状态



## ◎ 范例 623 ◎ DNS 查询。

```
[root@localhost ~]# nslookup www.bjtu.edu.cn //DNS 查询
Server: 202.112.144.30
Address: 202.112.144.30#53
Name: www.bjtu.edu.cn
Address: 202.112.144.31
[root@localhost ~]#
```

本例使用指令“nslookup www.bjtu.edu.cn”查询域名 www.bjtu.edu.cn 的 IP 地址，返回结果如上所示。

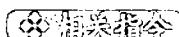
## ◎ 范例 624 ◎ 交互方式查询域名。



## 第7章 网络通信

```
[root@localhost ~]# nslookup //启动nslookup程序
> www.bjtu.edu.cn //查询www.bjtu.edu.cn域名
Server: 202.112.144.30
Address: 202.112.144.30#53
Name: www.bjtu.edu.cn
Address: 202.112.144.31
> set all //设置关键字
Default server: 202.112.144.30
Address: 202.112.144.30#53
Set options:
 novc nodebug nod2
 search recurse
 timeout = 0 retry = 3 port = 53
 querytype = A class = IN
 srchlist = localdomain
> server 202.112.144.30 //指定DNS服务器为202.112.144.30
Default server: 202.112.144.30
Address: 202.112.144.30#53
> www.bjtu.edu.cn //查询www.bjtu.edu.cn
Server: 202.112.144.30
Address: 202.112.144.30#53
Name: www.bjtu.edu.cn
Address: 202.112.144.31
> exit //退出程序
[root@localhost ~]#
```

本例首先使用指令 nslookup 启动 nslookup 指令，进入交互模式，输入“www.bjtu.edu.cn”，然后按“Enter”键，查询域名 www.bjtu.edu.cn；然后使用指令“set all”设置关键字，发现当前的配置下主 DNS 服务器为 202.112.144.65，第二服务器为 202.112.144.30；接着使用指令“server 202.112.144.30”设置 DNS 查询的主服务器为 202.112.144.30，这时再查询域名为 www.bjtu.edu.cn 的 IP 地址时，是通过服务器 202.112.144.30 查询的，而不是通过 202.112.144.65 查询的；最后使用指令 exit 退出 nslookup 程序。



**相关命令**

dig、host、named

## 7.24

# ping 指令：检测主机（IPv4）

 **语法** ping[必要参数][选择性参数][主机名或者 IP 地址]

 **功能说明** ping 指令向指定的主机发送 ICMP 包并等待回复，以确定指定的主机是否存在。

 **参数说明**

必要参数的具体说明如下：

参数	功能
-b	广播 ping 包
-n	只输出数值
-q	不显示处理过程
-R	记录路由
-v	运行时显示详细的处理信息
-t	不通过网关传送包

选择性参数的具体说明如下：

参数	功能
-c<次数>	指定接收包的次数
-i<时间>	周期性发送的间隔
-l<网卡>	指定网卡
-s<字节>	包的大小
-t<时间>	设置 TTL 值

 **执行范例**

◎ 范例 625 ◎ 检测主机是否存在。

```
[root@localhost ~]# ping www.sina.com.cn //检测主机是否存在
PING jupiter.sina.com.cn (202.205.3.143) 56(84) bytes of data.
64 bytes from 202.205.3.143: icmp_seq=0 ttl=52 time=7.16 ms
```



```

64 bytes from 202.205.3.143: icmp_seq=1 ttl=52 time=2.97 ms
64 bytes from 202.205.3.143: icmp_seq=2 ttl=52 time=3.41 ms
64 bytes from 202.205.3.143: icmp_seq=3 ttl=52 time=3.59 ms
64 bytes from 202.205.3.143: icmp_seq=4 ttl=52 time=2.85 ms
64 bytes from 202.205.3.143: icmp_seq=5 ttl=52 time=5.14 ms
64 bytes from 202.205.3.143: icmp_seq=6 ttl=52 time=2.50 ms
64 bytes from 202.205.3.143: icmp_seq=7 ttl=52 time=3.01 ms
64 bytes from 202.205.3.143: icmp_seq=8 ttl=52 time=6.46 ms
64 bytes from 202.205.3.143: icmp_seq=9 ttl=52 time=2.91 ms
64 bytes from 202.205.3.143: icmp_seq=10 ttl=52 time=6.57 ms
--- jupiter.sina.com.cn ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 14065ms
rtt min/avg/max/mdev = 2.502/4.238/7.169/1.670 ms, pipe 2
[root@localhost ~]#

```

本例使用指令“ping www.sina.com.cn”检测主机www.sina.com.cn是否存在，如果不手动终止，将一直ping下去。

○ 范例 626 ○ 指定接收包的次数。

```

[root@localhost ~]# ping -c 6 www.sina.com.cn //指定接收包的次数
PING jupiter.sina.com.cn (202.205.3.130) 56(84) bytes of data.
64 bytes from 202.205.3.130: icmp_seq=0 ttl=52 time=4.16 ms
64 bytes from 202.205.3.130: icmp_seq=1 ttl=52 time=2.84 ms
64 bytes from 202.205.3.130: icmp_seq=2 ttl=52 time=2.61 ms
64 bytes from 202.205.3.130: icmp_seq=3 ttl=52 time=4.25 ms
64 bytes from 202.205.3.130: icmp_seq=4 ttl=52 time=2.55 ms
64 bytes from 202.205.3.130: icmp_seq=5 ttl=52 time=3.38 ms
--- jupiter.sina.com.cn ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 9044ms
rtt min/avg/max/mdev = 2.558/3.305/4.258/0.695 ms, pipe 2
[root@localhost ~]#

```

本例使用指令“ping -c 6 www.sina.com.cn”检测主机www.sina.com.cn是否存在，并指定接收包的次数为6，接收到6个包将自动终止。

○ 范例 627 ○ 其他参数设置。

```

[root@localhost ~]# ping -i 3 -s 1024 -t 255 www.sina.com.cn //
检测主机是否存在
PING jupiter.sina.com.cn (202.205.3.130) 1024(1052) bytes of data.
1032 bytes from 202.205.3.130: icmp_seq=0 ttl=52 time=6.86 ms
1032 bytes from 202.205.3.130: icmp_seq=1 ttl=52 time=5.43 ms

```

## 7.25 ping6 指令：检测主机（IPv6）

```
1032 bytes from 202.205.3.130: icmp_seq=2 ttl=52 time=5.79 ms
1032 bytes from 202.205.3.130: icmp_seq=3 ttl=52 time=9.41 ms
1032 bytes from 202.205.3.130: icmp_seq=4 ttl=52 time=7.30 ms
1032 bytes from 202.205.3.130: icmp_seq=5 ttl=52 time=11.5 ms
1032 bytes from 202.205.3.130: icmp_seq=6 ttl=52 time=5.48 ms
1032 bytes from 202.205.3.130: icmp_seq=7 ttl=52 time=5.86 ms
--- jupiter.sina.com.cn ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 23074ms
rtt min/avg/max/mdev = 5.437/7.220/11.598/2.068 ms, pipe 2
[root@localhost ~]#
```

本例使用指令“ping -i 3 -s 1024 -t 255 www.sina.com.cn”检测主机 www.sina.com.cn 是否存在，并用“-i”参数指定周期性发送包的间隔为 3s，用“-s”参数指定发送包的大小为 1024 字节，用“-t”参数指定 TTL 值为 255。



traceroute

## 7.25

## ping6 指令：检测主机（IPv6）



ping6[必要参数][选择性参数][主机名称或者 IPv6 地址]



ping6 指令是针对于 IPv6 的 ping 程序。



必要参数的具体说明如下：

参数	功能
-b	广播 ping 包
-n	只输出数值
-q	不显示处理过程
-R	记录路由
-v	运行时显示详细的处理信息
-r	不通过网关传送包

选择性参数的具体说明如下：



参数	功能
-c<次数>	指定接收包的次数
-i<时间>	周期性发送的间隔
-l<网卡>	指定网卡
-s<字节>	包的大小
-t<时间>	设置 TTL 值

### ◎ 检测范例

#### ◎ 范例 628 ◎ 检测主机是否可达。

```
[root@localhost ~]# ping6 2001:da8:205:130:20c:29ff:fe32:451c
//检测主机是否可达
PING 2001:da8:205:130:20c:29ff: fe32:451c(2001:da8:205: 130:20c:29ff:
fe32:451c) 56 data bytes
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=0 ttl=64 time=6.64 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=1 ttl=64 time=0.218 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=3 ttl=64 time=0.052 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=5 ttl=64 time=0.053 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=6 ttl=64 time=0.045 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=7 ttl=64 time=0.051 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=8 ttl=64 time=0.052 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=9 ttl=64 time=0.052 ms
64 bytes from 2001:da8:205:130:20c:29ff:fe32:451c: icmp_seq=10 ttl=64 time=0.051 ms
--- 2001:da8:205:130:20c:29ff:fe32:451c ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10003ms
rtt min/avg/max/mdev = 0.045/0.665/6.645/1.891 ms, pipe 2
[root@localhost ~]#
```

本例使用指令“ping6 2001:da8:205:130:20c:29ff:fe32:451c”检测主机是否可达。

### ◎ 相关指令

ping

## 7.26 pppd 指令：ppp 连线的守护进程

pppd[必要参数][选择性参数][IP 地址]

**功能说明** pppd 指令用来建立主机间的 ppp 链接。

卷数说明

必要参数的具体说明如下：

参 数	功 能
auth	认证要求
noauth	不要求认证
nolog	不记录 log 信息

选择性参数的具体说明如下：

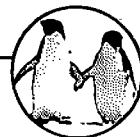
参 数	功 能
connect<文件>	指定 script 文件
file<文件>	指定配置文件
idle<时间>	超时设置
logfile<文件>	log 信息记录

执行范例

### ● 范例 629 ● 建立 ppp 链接。

```
[root@localhost ~]# pppd //建立ppp链接
~y)@A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)
61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)
)))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)'")
("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)
61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6)
)))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y)
@A!))))))4)"$)6))))))8)61 +.)'")("3)$~~y) @A!))))))4)"$)6))))))8)61 +.)
'")("3)$~~root@localhost ~#
```

本例使用 pppd 指令尝试建立 ppp 连线，这时会出现一堆乱码，这是正



常的，尝试失败后将返回命令行。

### 相关指令

pppsetup、ppp-off

## 7.27

### ppp-off 指令：关闭 ppp 连线

#### 语法

ppp-off

#### 功能说明

ppp-off 指令用来关闭 ppp 连线。

#### 执行范例

- 范例 630 ○ 关闭 ppp 连线。

```
[root@localhost ~]#ppp-off //关闭 ppp 连线
```

本例使用指令 ppp-off 关闭 ppp 连线。

### 相关指令

pppdump、pppsetup、pppstats

## 7.28

### pppsetup 指令：设置 ppp 连线

#### 语法

pppsetup

#### 功能说明

pppsetup 指令用来设置 ppp 连线。

#### 执行范例

- 范例 631 ○ 设置 ppp 连线。

```
[root@localhost ~]#pppsetup //设置 ppp 连线
```

本例使用指令 pppsetup 设置 ppp 连线。

### 相关指令

pppd、ppp-off、pppstats

## 7.29

### pppstats 指令：显示 ppp 连线状态

#### 语法

pppstats[必要参数][选择性参数]

## 7.30 rdate 指令：显示其他主机的日期与时间



**pppstats 指令用来显示 ppp 连线的状态。**



必要参数的具体说明如下：

参    数	功    能
-a	显示绝对数值
-r	显示附加的统计信息
-v	显示附加的统计信息
-z	不输出到标准输出

选择性参数的具体说明如下：

参    数	功    能
-c<次数>	重复显示的次数
-w<时间>	每次显示之间暂停指定时间，即设置显示的周期



○ 范例 632 ○ 显示 ppp 的连线状态。

[root@localhost ~]#pppstats //显示 ppp 的连线状态

本例使用指令 pppstats 显示 ppp 的连线状态。



pppd

7.30

## rdate 指令：显示其他主机的日期与时间



**rdate[必要参数][选择性参数][主机]**



**执行 rdate 指令将查询指定主机的系统时间。**



必要参数的具体说明如下：



参数	功能
-p	打印从远端主机返回的时间
-s	根据远端主机的时间设置本地机器的系统时间
-u	传输协议使用 UDP 协议
-l	使用 syslog 显示错误信息

选择性参数的具体说明如下：

参数	功能
-t<时间>	设置超时时间

### 运行范例

◎ 范例 633 ◎ 从远端主机获取时间。

```
[root@localhost ~]# rdate 192.168.88.7 //从192.168.88.7获取时间
```

本例使用指令“rdate 192.168.88.7”从远端主机 192.168.88.7 获取系统时间。

### 相关命令

date、telnet

## 7.31 route 指令：显示并设置路由

### 路由

route[必要参数][选择性参数]

功能说明 route 指令维护了内核的 IP 路由表。它的原始用途是为特定主机或者网络设置静态路由。可以通过“add”或者“del”参数改变路由表。

### 功能说明

必要参数的具体说明如下：

## 7.31 route 指令：显示并设置路由

参数	功能
-v	运行时显示详细的处理信息
-n	不解析名字
-e	显示更多的信息
-F	显示发送信息
-C	显示路由缓存

选择性参数的具体说明如下：

参数	功能
-help	帮助信息
-version	版本信息

### 【支持的地址族】

- inet: DARPA 互联网。
- inet6: IPv6。
- ax25: APRP AX.25。
- netrom: APRP NET/ROM。
- ipx: Novell IPX。
- ddp: Appletalk DDP。
- X25: CCITT X.25。

### ◎ 范例

#### ◎ 范例 634 ◎ 显示当前的路由表。

```
-sh-2.05b# route //显示路由表
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.88.0 * 255.255.255.0 U 0 0 0 eth0
169.254.0.0 * 255.255.0.0 U 0 0 0 eth0
127.0.0.0 * 255.0.0.0 U 0 0 0 lo
default 192.168.88.2 0.0.0.0 UG 0 0 0 eth0
-sh-2.05b#
```

本例使用指令 route 显示当前系统的路由表。



## ○范例 635 ○ 删除默认网关。

```
-sh-2.05b# route del default gw 192.168.88.2 //删除默认网关
-sh-2.05b# route //显示路由表
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.88.0 * 255.255.255.0 U 0 0 0 eth0
169.254.0.0 * 255.255.0.0 U 0 0 0 eth0
127.0.0.0 * 255.0.0.0 U 0 0 0 lo
-sh-2.05b#
```

本例使用指令“route del default gw 192.168.88.2”删除默认网关192.168.88.2，然后使用指令route显示路由表，发现默认网关被删除。

## ○范例 636 ○ 增加默认网关。

```
-sh-2.05b# route add default gw 192.168.88.2 //增加默认网关
-sh-2.05b# route //显示路由表
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.88.0 * 255.255.255.0 U 0 0 0 eth0
169.254.0.0 * 255.255.0.0 U 0 0 0 eth0
127.0.0.0 * 255.0.0.0 U 0 0 0 lo
default 192.168.88.2 0.0.0.0 UG 0 0 0 eth0
-sh-2.05b#
```

本例使用指令“route add default gw 192.168.88.2”增加默认网关192.168.88.2；使用指令route显示路由表，可以看到默认网关192.168.88.2以被加到路由表中。

## ○范例 637 ○ 增加一条路由。

```
-sh-2.05b# route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0
 //增加路由
-sh-2.05b# route //显示路由表
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.88.0 * 255.255.255.0 U 0 0 0 eth0
169.254.0.0 * 255.255.0.0 U 0 0 0 eth0
127.0.0.0 * 255.0.0.0 U 0 0 0 lo
224.0.0.0 * 240.0.0.0 U 0 0 0 eth0
default 192.168.88.2 0.0.0.0 UG 0 0 0 eth0
-sh-2.05b#
```

## 7.32 samba 指令：控制 Samba 服务端

本例使用指令“route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0”增加一条到达 224.0.0.0 的路由。

◎ 范例 638 ◎ 增加一条屏蔽路由。

```
-sh-2.05b# route add -net 10.0.0.0 netmask 255.0.0.0 reject //
```

增加一条屏蔽路由

```
-sh-2.05b# route
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.88.0	*	255.255.255.0	U	0	0	0	eth0
169.254.0.0	*	255.255.0.0	U	0	0	0	eth0
10.0.0.0	-	255.0.0.0	!	0	-	0	-
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
224.0.0.0	*	240.0.0.0	U	0	0	0	eth0
default	192.168.88.2	0.0.0.0	UG	0	0	0	eth0

```
-sh-2.05b#
```

本例使用指令“route add -net 10.0.0.0 netmask 255.0.0.0 reject”增加一条屏蔽路由，目的地为 10.x.x.x 的路由将被拒绝。



ifconfig、netstat、arp、rarp

## 7.32 samba 指令：控制 Samba 服务端



samba[必要参数]

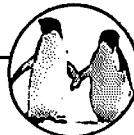


samba 指令用来控制 samba 服务端。



必要参数的具体说明如下：

参数	功能
start	启动 samba
stop	关闭 samba
status	samba 的状态
restart	重新启动 samba


**执行范例**

- ◎ 范例 639 ◎ 启动 Samba。

```
[root@localhost root]#samba start //启动 samba
```

本例使用指令“samba start”启动 Samba 服务端。

- ◎ 范例 640 ◎ 显示 Samba 的状态。

```
[root@localhost root]#samba status //启动 samba 状态
```

本例使用指令“samba status”显示 Samba 的状态。

**相关指令**

smbd

## 7.33 smbd 指令：Samba 服务端

**语法** smbd[必要参数][选择性参数]

**功能说明** smbd 指令是 Samba 程序的服务端。

**参数说明**

必要参数的具体说明如下：

参数	功能
-a	记录所有的连线信息
-D	后台执行
-P	测试

选择性参数的具体说明如下：

参数	功能
-h	帮助信息
-d<等级>	指定排错等级
-s<文件>	指定配置文件

**执行范例**

- ◎ 范例 641 ◎ 启动 Samba 服务器。

```
[root@localhost root]#smbd -D //启动 Samba 服务器
```

本例使用指令“smbd -D”在后台启动 Samba 服务器。



Samba

## 7.34 ssh 指令：远程登录

 ssh[必要参数][选择性参数][主机]

 ssh 指令用来远程登录主机。

 本页说明

必要参数的具体说明如下：

参数	功能
-1	强制 ssh 指令使用协议 1
-2	强制 ssh 指令使用协议 2
-4	强制使用 IPv4 协议
-6	强制使用 IPv6 协议
-f	后台执行
-N	测试
-n	重定位
-q	不显示处理信息
-v	运行时显示详细的处理信息

选择性参数的具体说明如下：

参数	功能
-b<地址>	指定地址
-F<文件>	指定配置文件
-i<认证文件>	指定认证文件
-l<用户账号>	指定用户账号
-p<端口号>	指定端口号
-V	版本信息



大家网

**◎ 技术说明**

- 范例 642 ○ 远程登录主机。

```
[root@localhost temp]# ssh 192.168.88.76 //远程登录主机
root@192.168.88.76's password:
Last login: Tue May 29 09:22:49 2007 from 192.168.88.7
[root@localhost ~]#
```

本例使用指令“ssh 192.168.88.76”远程登录主机 192.168.88.76。

- 范例 643 ○ 指定用户名登录远程主机。

```
[root@localhost temp]# ssh 192.168.88.76 -l kk //远程登录主机
kk@192.168.88.76's password:
Last login: Fri Dec 22 16:04:41 2006
[kk@localhost ~]$
```

本例使用指令“ssh 192.168.88.76 -l kk”以账号 kk 远程登录主机 192.168.88.76。

**◎ 相关命令**

telnet、scp

## 7.35 statserial 指令：samba 服务器程序

**◎ 基本用法** statserial[必要参数][设备]

**◎ 功能说明** statserial 指令可以显示串口的信息，通过组合键“Ctrl+C”可以终止状态的显示。

**◎ 参数说明**

必要参数的具体说明如下：

参 数	功 能
-n	不能循环显示
-d	以十进制数显示状态
-x	以十六进制数显示状态

**◎ 技术说明**

- 范例 644 ○ 显示串口状态。

## 7.35 statserial 指令：samba 服务器程序

```
-sh-2.05b# statserial /dev/ttyS1 //显示串口状态
Device: /dev/ttyS1
Signal Pin Pin Direction Status Full
Name (25) (9) (computer) Name

FG 1 - - - Frame Ground
TxD 2 3 out - Transmit Data
RxD 3 2 in - Receive Data
RTS 4 7 out 1 Request To Send
CTS 5 8 in 1 Clear To Send
DSR 6 6 in 1 Data Set Ready
GND 7 5 - - Signal Ground
DCD 8 1 in 1 Data Carrier Detect
DTR 20 4 out 1 Data Terminal Ready
RI 22 9 in 0 Ring Indicator
```

本例使用指令“statserial /dev/ttyS1”显示串口/dev/ttyS1 的详细信息。  
如果不用户组合键“Ctrl+C”终止，系统将循环显示串口的状态。

## ◎ 范例 645 ◎ 只显示一次串口的状态。

```
-sh-2.05b# statserial -n /dev/ttyS1 //显示一次串口的状态
```

```
Device: /dev/ttyS1
Signal Pin Pin Direction Status Full
Name (25) (9) (computer) Name

FG 1 - - - Frame Ground
TxD 2 3 out - Transmit Data
RxD 3 2 in - Receive Data
RTS 4 7 out 1 Request To Send
CTS 5 8 in 1 Clear To Send
DSR 6 6 in 1 Data Set Ready
GND 7 5 - - Signal Ground
DCD 8 1 in 1 Data Carrier Detect
DTR 20 4 out 1 Data Terminal Ready
RI 22 9 in 0 Ring Indicator
```

```
-sh-2.05b#
```

本例使用指令“statserial -n /dev/ttyS1”显示串口/dev/ttyS1 的状态，只显示一次，不循环进行显示。



stat



## 7.36 talk 指令：与其他用户交谈

**语法** talk[用户名][终端机号]

**功能说明** talk 指令用来进行系统中在线用户间的交谈。

**参数说明**

必要参数的具体说明如下：

参数	功能
[用户名]	系统用户名，如：root
[终端号]	如：tty1

**执行范例**

◎ 范例 646 ◎ 与其他用户交谈。

```
[root@localhost ~]# who //显示当前用户
kk tty1 May 30 10:00
root pts/1 May 30 09:16 (192.168.88.7)
root pts/2 May 30 09:28 (192.168.88.7)
root :0 May 30 09:58
root pts/3 May 30 09:59 (:0.0)
[root@localhost ~]# talk kk tty4 //与用户 kk 交谈
[root@localhost ~]#
```

本例使用指令“talk kk tty1”与 tty1 下的用户 kk 交谈。

**相关命令**

wall

## 7.37 tcpdump 指令：倾倒网络传输数据

**语法** tcpdump[必要参数][选择性参数]

**功能说明** tcpdump 指令用于显示网络中传输的 tcp 数据包的信息，只有系统管理员才能执行此指令。

## 7.37 tcpdump 指令：倾倒网络传输数据



必要参数的具体说明如下：

参数	功能
-a	显示名称
-d	转换数据包编码
-dd	数据包编码的 C 语言格式
-ddd	数据包编码的十进制格式
-e	显示文件头
-l	输出缓冲区信息
-n	数字形式显示
-N	不显示域名
-q	不显示详细的处理信息
-t	不显示时间
-v	运行时显示详细的处理信息
-X	以十六进制显示

选择性参数的具体说明如下：

参数	功能
-c<数据包数>	收到指定数据包后停止输出
-F<文件>	指定表达方式文件
-i<网卡>	指定网卡
-r<文件>	从指定文件读入信息
-w<文件>	把数据包写入指定文件



◎ 范例 647 ◎ 显示 tcp 包信息。

```
[root@localhost ~]# tcpdump //显示tcp包信息
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
```



```

10:22:26.173767 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 271912:
272060(148) ack 1145 win 8576
10:22:26.174259 IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 271764 win 16252
//此处省略了部分结果
10:22:26.179954 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 276144:
276292(148) ack 1145 win 8576
10:22:26.180097 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 276292:
276440(148) ack 1145 win 8576
10:22:26.180211 IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 276292 win 16680
2062 packets captured
3163 packets received by filter
1011 packets dropped by kernel
[root@localhost ~]#

```

本例使用指令 `tcpdump` 显示系统的 tcp 包的相关信息。

◎ 范例 648 ◎ 显示指定数量的数据包。

```

[root@localhost ~]# tcpdump -c 50 //显示 50 个数据包
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
10:26:46.703677 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3977410840:
3977410956(116) ack 450005671 win 8576
10:26:46.710797 IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 116 win 17196
10:26:46.711884 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 116:232(116)
ack 1 win 8576 //此处省略了部分结果
10:26:46.737923 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 5060:5192(132)
ack 1 win 8576
10:26:46.738023 IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 5192 win 17176
10:26:46.738872 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 5192:5324(132)
ack 1 win 8576
10:26:46.739780 IP 192.168.88.76.ssh > 192.168.88.7.1632: P 5324:5456(132)
ack 1 win 8576
10:26:46.739841 IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 5456 win 16912
50 packets captured
50 packets received by filter

```

### 7.37 tcpdump 指令：倾倒网络传输数据

0 packets dropped by kernel

[root@localhost ~]#

本例使用指令“tcpdump -c 50”显示50个TCP数据包的信息。

○ 范例 649 ○ 不显示时间。

```
[root@localhost ~]# tcpdump -c 50 -t //显示50个数据包，但不显示时间
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3977426564:3977426680(116) ack
450007259 win 9648
```

```
IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 116 win 16340
```

```
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 116:232(116) ack 1 win 9648
```

```
IP 192.168.88.76.34978 > 202.112.144.30.domain: 49931+ PTR? 7.88. 168.
```

192.in-addr.arpa. (43)

```
IP 202.112.144.30.domain > 192.168.88.76.34978: 49931 NXDomain 0/1/0 (120)
```

```
IP 192.168.88.76.34978 > 202.112.144.30.domain: 51802+ PTR? 76.88.
```

168.192.in-addr.arpa. (44) //此处省略了部分结果

```
IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 3012 win 16592
```

```
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3012:3128(116) ack 1 win 9648
```

```
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3128:3244(116) ack 1 win 9648
```

```
IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 3244 win 16360
```

```
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3244:3360(116) ack 1 win 9648
```

```
IP 192.168.88.76.ssh > 192.168.88.7.1632: P 3360:3476(116) ack 1 win 9648
```

```
IP 192.168.88.7.1632 > 192.168.88.76.ssh: . ack 3476 win 16128
```

50 packets captured

87 packets received by filter

0 packets dropped by kernel

[root@localhost ~]#

本例使用指令“tcpdump -c 50 -t”显示50个数据包，但不显示时间。

○ 范例 650 ○ 精简模式显示。

```
[root@localhost ~]# tcpdump -c 50 -q //精简模式显示50个数据包
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
10:37:08.134692 IP 192.168.88.76.ssh > 192.168.88.7.1632: tcp 116
```



```

10:37:08.139285 IP 192.168.88.7.1632 > 192.168.88.76.ssh: tcp 0
10:37:08.139521 IP 192.168.88.76.ssh > 192.168.88.7.1632: tcp 116
10:37:08.136111 IP 192.168.88.76.34978 > 202.112.144.30.domain: UDP, length 43
10:37:08.137369 IP 202.112.144.30.domain > 192.168.88.76.34978: UDP, length 120
10:37:08.137619 IP 192.168.88.76.34978 > 202.112.144.30.domain: UDP, length 44
10:37:08.138841 IP 202.112.144.30.domain > 192.168.88.76.34978: UDP, length 121
10:37:08.139078 IP 192.168.88.76.ssh > 192.168.88.7.1632: tcp 116
 //此处省略了部分结果
10:37:08.146959 IP 192.168.88.76.ssh > 192.168.88.7.1632: tcp 116
10:37:08.147176 IP 192.168.88.76.ssh > 192.168.88.7.1632: tcp 116
10:37:08.147296 IP 192.168.88.7.1632 > 192.168.88.76.ssh: tcp 0
50 packets captured
68 packets received by filter
0 packets dropped by kernel
[root@localhost ~]#

```

本例使用指令“tcpdump -c 50 -q”以精简模式显示 50 个 tcp 数据包。

- 范例 651 ○ 将数据包编码转换成可阅读格式。

```

[root@localhost ~]# tcpdump -d //可阅读格式
(000) ret #96
[root@localhost ~]#

```

本例使用指令“tcpdump -d”将数据包转化为可阅读的格式。

- 范例 652 ○ 将数据包编码转换成 C 语言格式。

```

[root@localhost ~]# tcpdump -dd //C 语言格式
{ 0x6, 0, 0, 0x00000060 },
[root@localhost ~]#

```

本例使用指令“tcpdump -dd”将数据包编码转换成 C 语言格式。

- 范例 653 ○ 将数据包编码转换成十进制。

```

[root@localhost ~]# tcpdump -ddd //十进制格式
1
6 0 0 96
[root@localhost ~]#

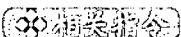
```

本例使用指令“tcpdump -ddd”将数据包编码转换成十进制格式。

## ○范例 654 ○ 显示指定网卡的tcp数据包。

```
[root@localhost ~]# tcpdump -c 50 -i eth0 //显示指定网卡的tcp数据包
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
10:46:05.543602 IP 192.168.88.76.ssh > 192.168.88.7.1868: P 4212038691:
4212038807(116) ack 1607253131 win 8576
10:46:05.555958 IP 192.168.88.7.1868 > 192.168.88.76.ssh: . ack 116 win 16108
10:46:05.556184 IP 192.168.88.76.ssh > 192.168.88.7.1868: P 116:232(116)
ack 1 win 3576
10:46:05.545069 IP 192.168.88.76.34978 > 202.112.144.30.domain: 47200+
PTR? 7.88.168.192.in-addr.arpa. (43)
//此处省略了部分结果
10:46:05.564214 IP 192.168.88.76.ssh > 192.168.88.7.1868: P 3216:3348(132)
ack 1 win 8576
10:46:05.564276 IP 192.168.88.7.1868 > 192.168.88.76.ssh: . ack 3348 win 16152
10:46:05.564486 IP 192.168.88.76.ssh > 192.168.88.7.1868: P 3348:3480(132)
ack 1 win 8576
10:46:05.564720 IP 192.168.88.76.ssh > 192.168.88.7.1868: P 3480:3612(132)
ack 1 win 8576
50 packets captured
91 packets received by filter
0 packets dropped by kernel
[root@localhost ~]#
```

本例使用指令“tcpdump -c 50 -i eth0”显示指定网卡 eth0 上的 tcp 数据包信息。



stty、pcap、bpf、nit、pfconfig

## 7.38

## telnet 指令：远程登录



telnet[必要参数][选择性参数][主机]



telnet 指令用于远程登录，可用 telnet help 显示帮助



信息。

#### ◎ 参数说明

必要参数的具体说明如下：

参 数	功 能
-8	使用 8 位字符
-a	自动登录
-d	调试模式
-K	关闭自动登录

选择性参数的具体说明如下：

参 数	功 能
-e<字符>	设置脱离字符
-k<域名>	指定域名
-l<用户名>	指定用户名
-n<文件>	指定记录文件
-b<别名>	指定主机的别名

#### ◎ 运行范例

##### ○ 范例 655 ○ 远程登录主机。

```
[root@localhost ~]# telnet 192.168.88.2 //远程登录
Trying 192.168.88.2...
Connected to 192.168.88.2 (192.168.88.2).
Escape character is '^]'.
Debian GNU/Linux 4.0
iplab604 login: root //输入用户名
Password: //输入密码
Last login: Wed May 30 17:07:12 2007 from 192.168.88.7 on pts/4
root@localhost:~#
```

本例使用指令“telnet 192.168.88.2”远程登录主机 192.168.88.2。

#### ◎ 相关命令

ssh

## 7.39 testparm 指令：测试 Samba 配置

**7.39 testparm 指令：测试 Samba 配置****【语法】** testparm[必要参数][配置文件][主机]**【功能说明】** testparm 指令可以用来测试 Samba 服务的配置是否正确无误。**【参数说明】**

必要参数的具体说明如下：

参数	功能
-s	不显示提示符

**【执行范例】**

◎ 范例 656 ◎ 查看 Samba 配置。

```
[root@localhost ~]# testparm //测试 Samba 配置
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Loaded services file OK.

Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
//按“回车”键继续

Global parameters
[global]
 workgroup = MYGROUP
 server string = Samba Server
 log file = /var/log/samba/%m.log
 max log size = 50
 socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
 printcap name = /etc/printcap
 dns proxy = No
 idmap uid = 16777216-33554431
 idmap gid = 16777216-33554431
 cups options = raw

[homes]
```



```

comment = Home Directories
read only = No
browseable = No
[printers]
comment = All Printers
path = /var/spool/samba
printable = Yes
browseable = No
[root@localhost ~]#

```

本例使用指令 testparm 检测 samba 配置。中间过程中显示提示符“Press enter to see a dump of your service definitions”，确认后才能继续。

○ 范例 657 ○ 不显示提示符，检测 samba 配置。

```

[root@localhost ~]# testparm -s //检测 samba 配置，不显示提示符
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Loaded services file OK.

Global parameters
[global]

 workgroup = MYGROUP
 server string = Samba Server
 log file = /var/log/samba/%m.log
 max log size = 50
 socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
 printcap name = /etc/printcap
 dns proxy = No
 idmap uid = 16777216-33554431
 idmap gid = 16777216-33554431
 cups options = raw

[homes]
 comment = Home Directories
 read only = No
 browseable = No

[printers]
 comment = All Printers
 path = /var/spool/samba
 printable = Yes

```

## 7.40 tracepath 指令：追踪路径

```
browsable = No
[root@localhost ~]#
```

本例使用指令“testparm -s”检测 samba 的配置，但不显示提示符。

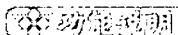


smbd

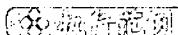
## 7.40 tracepath 指令：追踪路径



tracepath[目的地址][端口号]



tracepath 指令用来实现主机连接到目的地址时所经过的路由。



○ 范例 658 ○ 显示连接到目的地址时的路由。

```
[root@localhost ~]# tracepath www.sohu.com //显示连接路由
1: 192.168.88.76 (192.168.88.76) 0.219ms pmtu 1500
1: 192.168.88.2 (192.168.88.2) 2.295ms
2: 202.112.146.2 (202.112.146.2) 3.384ms
3: 10.10.146.1 (10.10.146.1) 3.444ms
4: 10.10.147.1 (10.10.147.1) 19.639ms
5: 10.20.70.1 (10.20.70.1) 10.353ms
6: 10.10.42.2 (10.10.42.2) 8.194ms
7: 202.112.42.65 (202.112.42.65) 3.761ms
8: 202.112.62.241 (202.112.62.241) 4.146ms
9: 202.112.53.74 (202.112.53.74) 4.066ms
10: 202.112.38.82 (202.112.38.82) 3.822ms
11: 202.205.13.218 (202.205.13.218) 6.186ms
12: no reply
13: no reply
[root@localhost ~]#
```

本例使用指令“tracepath www.sohu.com”显示连接到目的地 www.sohu.com 的路由，由于主机没有登录账号，不能连接到 www.sohu.com。



traceroute



## 7.41

### traceroute 指令：显示数据包到主机间的路径

**要点**

traceroute[必要参数][选择性参数][主机]

**功能说明**

traceroute 指令追踪网络数据包的路由。

**使用说明**

必要参数的具体说明如下：

参 数	功 能
-d	开启调试功能
-I	使用 ICMP
-v	运行时显示详细的处理信息
-x	关闭数据包的检测
-r	不经过路由表路由

选择性参数的具体说明如下：

参 数	功 能
-f<TTL 值>	设置 TTL 值
-g<网关>	设置网关
-m<次数>	设置跳数
-p<端口号>	指定端口号
-w<时间>	设置检测的间隔时间

**执行范例**

○ 范例 659 ○ 显示到达目的地的数据包路由。

```
root@localhost:~# traceroute www.baidu.com //追踪路由
traceroute to www.a.shifen.com (211.94.144.100), 30 hops max, 40 byte packets
 1 202.112.146.2 (202.112.146.2) 1.427 ms 2.398 ms 13.981 ms
 2 10.10.146.1 (10.10.146.1) 1.553 ms 12.339 ms 16.526 ms
```

## 7.42 tty 指令：显示标准输入设备名称

```

3 10.10.147.1 (10.10.147.1) 14.018 ms 19.101 ms 19.803 ms
4 10.20.70.1 (10.20.70.1) 13.360 ms 19.112 ms 20.123 ms
5 10.10.42.2 (10.10.42.2) 3.863 ms 3.688 ms 3.741 ms
6 202.112.42.65 (202.112.42.65) 1.574 ms 1.588 ms 1.289 ms
7 202.112.62.241 (202.112.62.241) 2.315 ms 2.099 ms 1.640 ms
8 202.112.53.178 (202.112.53.178) 1.739 ms 1.553 ms 1.252 ms
9 202.38.123.22 (202.38.123.22) 155.748 ms 155.701 ms 155.364 ms
10 211.94.44.2 (211.94.44.2) 153.914 ms 153.561 ms 153.683 ms
11 211.94.68.110 (211.94.68.110) 155.247 ms 154.164 ms 154.652 ms
12 211.94.68.126 (211.94.68.126) 157.155 ms 157.394 ms 159.635 ms
13 * *
14 * *
15 * *
16 * *
17 * *
18 *

```

本例使用指令“traceroute www.baidu.com”显示到达目的地 www.baidu.com 所经过的路由。



tracepath

## 7.42 tty 指令：显示标准输入设备名称

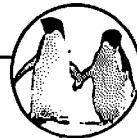
**tty[必要参数][选择性参数]**

**tty** 指令用来查询当前使用的终端的名称。

必要参数的具体说明如下：

参数	功 能
-s	不显示处理信息

选择性参数的具体说明如下：



参数	功能
-help	帮助信息
-version	版本信息

### ◎执行范例

◎ 范例 660 ◎ 显示当前终端的名称。

```
[root@localhost ~]# tty //显示终端的名称
/dev/pts/1
[root@localhost ~]#
```

本例使用指令 **tty** 显示当前终端的名称，显示为 /dev/pts/1。

### ◎相关指令

**fdisk**

## 7.43 uulog 指令：显示 uucp 记录信息

◎功能 **uulog[必要参数][选择性参数]**

◎功能说明 **uulog** 指令显示 UUCP 记录文件中的记录信息。

### ◎参数说明

必要参数的具体说明如下：

参数	功能
-D	显示调试信息
-F	循环显示文件内容
-S	统计信息模式

选择性参数的具体说明如下：

参数	功能
-f<主机>	指定主机
-l<配置文件>	指定配置文件
-n<行数>	指定显示的行数

## 7.44 uuname 指令：显示 uucp 远端主机

续表

参数	功能
-u<用户>	指定用户
-s<主机>	指定主机
-debug<等级>	调试等级
-help	帮助信息

## 执行范例

## ◎ 范例 661 ◎ 显示 uucp log 信息。

```
[root@localhost ~]# uulog //显示uucp 记录信息
uucico -- (2007-05-30 23:52:24.96 5640) Incoming call (login root port pts/1)
uucico -- (2007-05-30 23:54:24.18 5640) ERROR: Timeout
uucico -- (2007-05-30 23:56:12.00 5700) No work
uucico -- (2007-05-30 23:56:32.97 5706) Incoming call (login root port pts/1)
uucico -- (2007-05-31 09:13:45.03 2653) No work
[root@localhost ~]#
```

本例使用指令 uulog 显示 uucp 记录信息。

## 相关指令

uucp

**7.44****uuname 指令：显示 uucp 远端主机**

## 命令

uucp[必要参数][选择性参数]

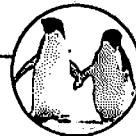
## 功能说明

uuname 指令显示全部的 uucp 远端主机。

## 无关说明

必要参数的具体说明如下：

参数	功能
-a	别名显示
-l	输出本机名称



选择性参数的具体说明如下：

参 数	功 能
-<配置文件>	指定配置文件
-help	帮助信息
-version	版本信息

### ◎ 次执行范例

◎ 范例 662 ◎ 显示 uucp 远端主机名称。

```
[root@localhost ~]# uname //显示 uucp 远端主机名称
```

本例使用指令 uname，显示 uucp 远端主机名称。

### ◎ 相关命令

uucp

## 7.45 uustat 指令：显示 uucp 状态

### ◎ 语法

uustat[必要参数][选择性参数]

◎ 功能说明 uustat 指令显示当前的 uucp 状态，它还可以显示、删除或者启动 uucp 任务。

### ◎ 交互说明

必要参数的具体说明如下：

参 数	功 能
-a	显示全部任务
-e	显示等待执行的任务
-i	交互式模式
-kill-all	删除所有任务
-status	显示状态
-mail	启动 mail

## 7.46 uux 指令：在远端的 uucp 主机上运行指令

选择性参数的具体说明如下：

参 数	功 能
-k<任务>	删除指定任务
-help	帮助信息
-version	版本信息

### ○ 实例范例

○ 范例 663 ○ 显示所有任务。

```
[root@localhost ~]# uustat -a //显示所有任务
```

本例使用指令“uustat -a”显示所有 uucp 任务。

○ 范例 664 ○ 显示等待执行的任务。

```
[root@localhost ~]# uustat -o //显示等待执行的任务
```

本例使用指令“uustat -o”显示等待执行的 uucp 任务。

### ○ 相关命令

uucp

## 7.46

## uux 指令：在远端的 uucp 主机上运行指令

○ 命令 uux[必要参数][选择性参数][指令]

○ 功能说明 uux 指令可在远端 uucp 主机上执行指令。

○ 参数说明

必要参数的具体说明如下：

参 数	功 能
-p	从标准输入读取指令
-b	显示状态
-l	指定配置文件
-j	任务编号

选择性参数的具体说明如下：



参数	功能
<code>--debug&lt;等级&gt;</code>	指定调试等级
<code>-help</code>	帮助信息
<code>--version</code>	版本信息

#### ❖ 执行范例

- 范例 665 在远端 uucp 主机执行指令。

```
[root@localhost ~]# uux kk!ls -l /home/kk/ //在远端主机执行指令
```

本例使用指令“uux kk!ls -l /home/kk/”在远端主机执行指令“ls -l /home/kk/”。

#### ❖ 相关指令

uucp

## 7.47 wall 指令：发送信息

#### ❖ 基础 wall[信息]

功能说明 执行 wall 指令可以向所有的终端用户发送广播信息。

#### ❖ 执行范例

- 范例 666 广播消息。

```
[root@localhost ~]# wall iloveintermilan //广播消息
Broadcast message from root (pts/1) (Thu May 31 11:30:33 2007):
iloveintermilan
[root@localhost ~]#
```

本例使用指令“wall iloveintermilan”广播消息 iloveintermilan。

#### ❖ 相关指令

mesg

## 7.48 wget 指令：从互联网下载资源

#### ❖ 基础

wget[必要参数][选择性参数]

## 7.48 wget 指令：从互联网下载资源

**功能说明** wget 指令用于从互联网下载资源。

**参数说明**

必要参数的具体说明如下：

参    数	功    能
-c	支持断点续传功能
-nc	不覆盖当前文件
-v	运行时显示详细的处理信息
-x	强制建立目录
-spider	不下载资料
-b	后台执行
-r	递归处理

选择性参数的具体说明如下：

参    数	功    能
-Q<容量>	容量限制
-h	帮助信息

**执行范例**

◎ 范例 667 ◎ 下载文件。

```
root@localhost:~# wget -Q 10M www.sina.com.cn //下载文件
--18:15:00-- http://www.sina.com.cn/
 => `index.html'
Resolving www.sina.com.cn... 202.205.3.142, 202.205.3.143, 202.205.3.130
Connecting to www.sina.com.cn|202.205.3.142|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]

[=>] 277,867 8.59K/s

18:15:27 (11.10 KB/s) - `index.html' saved [277867]

root@localhost:~#
```



本例使用指令“wget -Q 10M www.sina.com.cn”从网站www.sina.com.cn下载10MB的资料。

**相关指令**

get

## 7.49

### write 指令：传送信息

**语法** write[用户名][终端名]

**功能说明** 一个用户可以使用 write 指令传送信息到另一个用户。

**执行范例**

○ 范例 668 ○ 写消息。

```
[root@localhost ~]# who //显示系统登录用户
root tty1 May 31 15:15
root tty2 May 31 15:15
root tty3 May 31 15:37
kk tty4 May 31 15:38
root pts/1 May 31 15:31 (192.168.88.7)
root :0 May 31 15:15
[root@localhost ~]# whoami //显示当前用户
root
[root@localhost ~]# write kk //发送信息
i love intermilan
[root@localhost ~]#
```

本例使用指令“write kk”向kk发送信息。

```
[kk@localhost ~]$
Message from root@localhost.localdomain on pts/1 at 15:40 ...
i love intermilan
EOF
```

消息接收端的显示如上所示。

**相关指令**

wall

## 7.50 ytalk 指令：与其他用户交谈

 **ytalk**[必要参数][选择性参数][用户名]

 **功能说明** 使用 ytalk 指令可以与在线的其他用户交谈，也可以通过指定主机名和用户名与不同主机上的用户交谈。

 **参数说明**

必要参数的具体说明如下：

参 数	功 能
-x	关闭图形界面
-Y	用 Y 或者 N 回答问题
-s	开启对话框
-i	开启提示

选择性参数的具体说明如下：

参 数	功 能
-h<主机>	指定主机

 **运行实例**

◎范例 669 ◎ 与指定用户交谈。

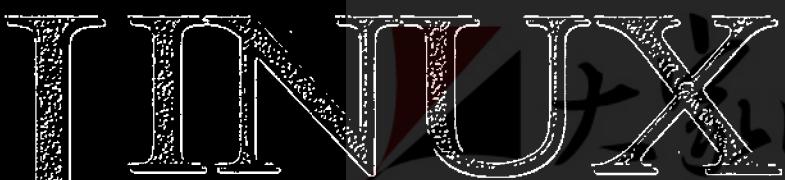
```
[root@localhost temp]# who //显示用户信息
root tty1 May 31 15:15
root tty2 May 31 15:15
root tty3 May 31 15:37
kk tty4 May 31 15:38
root pts/1 May 31 15:31 (192.168.88.7)
root :0 May 31 15:15
[root@localhost temp]# ytalk kk //与用户 kk 交谈
```

本例使用指令“ytalk kk”与用户 kk 进行交谈。

 **相关指令**

write、wall、talk

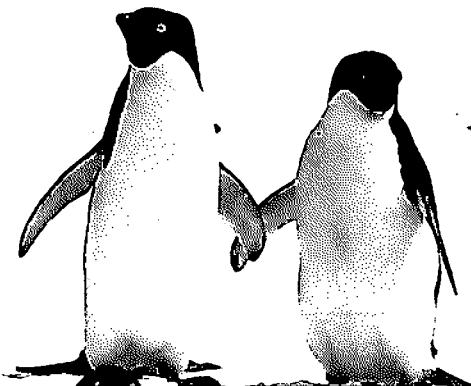




## 第8章

### 磁盘管理

磁盘管理指令的功能是管理 Linux 系统的磁盘，主要包括移动存储设备的挂载、磁盘检测、磁盘空间查询、磁盘分区等。磁盘管理是系统管理的重要部分，因此了解磁盘管理可以方便用户更好地维护系统。





## 8.1

# automount 指令：为 auto 文件系统配置挂载点

**【启动】** automount[必要参数][选择性参数]挂载点映射类型[格式]  
映射[映射参数]

**【功能说明】** automount 指令为 auto 文件系统配置挂载点。  
automount 指令根据挂载点和相应的文件来自动挂载文件系统。

**【参数说明】**

必要参数的具体说明如下：

参 数	功 能
-p	将守护进程的进程号写入指定的文件
-t	设置超时时间

选择性参数的具体说明如下：

参 数	功 能
-v	版本信息

### 【其他参数】

automount 指令至少需要 3 个参数，挂载点、映射类型和映射。

挂载点：auto 文件系统要链接的基本的位置信息，必须是已经存在的目录名。

映射参数：所有不使用“-”参数的指令将被看作“-o”参数来挂载。

- 文件：映射是一个通常的文字文件。
- 程序：映射是一个可执行的程序。
- yp：映射是一个 NIS (YP) 数据库。
- nisplus：映射是一个 NIS+数据库。
- hesiod：映射是一个 hesiod 数据库。
- ldap：映射是一个 ldap 参数。

格式：映射数据的格式。

**映射：**使用的映射文件的位置信息。

**映射参数：**所有不使用“-”选项的指令将被看作“-o”参数来挂载。

### 【概念说明】

**守护进程：**可以简单地看作一般的程序。

#### ○○ 实例范例

- 范例 670 ○ 设置超时时间。

```
[root@localhost root]#automount -t 10 //设置超时时间
```

本例使用指令“automount -t 10”设置超时时间。

#### ○○ 相关命令

mount、autofs

## 8.2 df 指令：磁盘信息

#### ○○ 简介

df[必要参数][选择性参数][文件或者设备]

○○ 功能说明

df 指令显示指定磁盘文件的可用空间。如果没有文件名被指定，则所有当前挂载的文件系统的可用空间将被显示。默认情况下，磁盘空间将以 1KB 为单位进行显示，除非环境变量 .POSIXLY\_CORRECT 被指定，那样的话将以 512 字节为单位进行显示。

#### ○○ 参数说明

必要参数的具体说明如下：

参 数	功 能
-a	全部文件系统的列表
-h	以便于用户读取的方式显示信息
-H	等同于“-h”，但是计算时，1K=1000，而不是1K=1024
-i	显示 inode 信息
-k	区块为 1024 字节
-l	只显示本地文件系统
-m	区块为 1048576 字节
-no-sync	忽略 sync 指令



续表

参数	功能
-P	输出格式为 POSIX
-sync	在取得磁盘信息前，先执行 sync 指令
-T	文件系统类型

选择性参数的具体说明如下：

参数	功能
-block-size=<区块大小>	指定区块大小
-t<文件系统类型>	只显示选定文件系统的磁盘信息
-x<文件系统类型>	不显示选定文件系统的磁盘信息
-help	帮助信息
-version	版本信息

### ◎ 实例范例

#### ◎ 范例 671 ◎ 显示磁盘使用情况。

```
[root@localhost ~]# df //显示当前磁盘的使用情况
Filesystem 1K-块 已用 可用 已用% 挂载点
/dev/sda3 9546940 7797944 1256204 87% /
/dev/sda1 101086 11312 84555 12% /boot
/dev/shm 257644 0 257644 0% /dev/shm
[root@localhost ~]#
```

本例使用指令 df 显示当前磁盘的使用情况。

#### ◎ 范例 672 ◎ 以 inode 模式来显示磁盘的使用情况。

```
[root@localhost ~]# df -i //显示磁盘信息
Filesystem Inode (I) 已用 (I) 可用 (I) 已用% 挂载点
/dev/sda3 2466048 342721 2123327 14% /
/dev/sda1 26104 35 26069 1% /boot
/dev/shm 64411 1 64410 1% /dev/shm
[root@localhost ~]#
```

本例使用指令 “df -i” 显示当前磁盘的使用情况。显示的是文件系统的全部、已用、可用的 inode 数量，而非磁盘数。

#### ◎ 范例 673 ◎ 以用户容易识别的方式显示磁盘信息。

### 8.3 dirs 指令：显示目录信息

```
[root@localhost ~]# df -h //显示磁盘容量
Filesystem 容量 已用 可用 已用% 挂载点
/dev/sda3 9.2G 7.5G 1.2G 87% /
/dev/sdal 99M 12M 83M 12% /boot
/dev/shm 252M 0 252M 0% /dev/shm
[root@localhost ~]#
```

本例使用指令“df -h”以用户更容易识别的方式显示当前磁盘的使用情况。

◎ 范例 674 ◎ 只显示指定磁盘类型的信息。

```
[root@localhost ~]# df -t ext3 //显示磁盘信息
Filesystem 1K-块 已用 可用 已用% 挂载点
/dev/sda3 9546940 7797964 1256184 87% /
/dev/sdal 101086 11312 84555 12% /boot
[root@localhost ~]#
```

本例使用指令“df -t ext3”显示当前磁盘的使用信息，并且通过“-t”参数只显示 ext3 类型的磁盘信息。

◎ 范例 675 ◎ 不显示指定磁盘类型的信息。

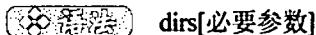
```
[root@localhost ~]# df -x ext3 //显示磁盘信息
Filesystem 1K-块 已用 可用 已用% 挂载点
/dev/shm 257644 0 257644 0% /dev/shm
[root@localhost ~]#
```

本例使用指令“df -x ext3”显示磁盘的信息，并且通过“-x”参数，不显示指定磁盘类型的磁盘使用信息。



mount

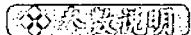
## 8.3 dirs 指令：显示目录信息



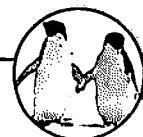
dirs[必要参数]



dirs 指令用来显示目录信息。



必要参数的具体说明如下：



参数	功能
-l	显示目录完整信息

功能说明

◎ 范例 676 ◎ 显示堆栈中的记录。

```
[root@localhost ~]# dirs -l //显示堆栈中的记录
/root
```

[root@localhost ~]#

本例使用指令“dirs -l”显示堆栈中的记录。

相关命令

dir

## 8.4

## du 指令：显示目录或者文件所占的磁盘空间

功能

du[必要参数][选择性参数][目录或者文件]

功能说明

显示每个文件的磁盘使用空间，对目录同样可以适用。

公式说明

必要参数的具体说明如下：

参数	功能
-a	显示所有文件的大小
-b	以字节为单位显示文件的大小
-c	显示目录或者文件的总和
-D	显示符号链接的源文件大小
-h	以更便于用户阅读的方式显示结果
-H	与“-h”参数作用相同，但是进制为 1000，即 1K=1000
-k	以 1024 字节为单位
-L	显示符号链接所指向的文件的大小
-m	单位为 1MB
-s	只显示总和大小

## 8.4 du 指令：显示目录或者文件所占的磁盘空间

续表

参数	功能
-s	只包含当前目录，不包含子目录大小
-x	文件系统的识别，只处理遇到的第一个文件系统类型，忽略其他的文件系统类型

选择性参数的具体说明如下：

参数	功能
-X<文件>	从指定文件中读取
-exclude=<文件>	忽略指定的文件
-max-depth=<层数>	忽略超过指定层数的目录
-help	帮助信息
-version	版本信息

## ◎ 实行范例

## ◎ 范例 677 ◎ 显示目录或者文件所占的磁盘空间。

```
[root@localhost other]# du //显示目录或者文件所占的磁盘空间
56 ./DJ
28 ./helloworld/debug/doc/en
60 ./helloworld/debug/doc
80 ./helloworld/debug/src/.deps
8 ./helloworld/debug/src/.libs
860 ./helloworld/debug/src
24 ./helloworld/debug/po
1372 ./helloworld/debug
92 ./helloworld/doc/en
156 ./helloworld/doc
80 ./helloworld/src/.deps
8 ./helloworld/src/.libs
320 ./helloworld/src
1184 ./helloworld/autom4te.cache
48 ./helloworld/po
24 ./helloworld/templates
1028 ./helloworld/admin
6468 ./helloworld
```



```
6548
[root@localhost other]#
```

本例使用指令 du 显示目录或者文件所占的磁盘空间。

◎ 范例 678 ◎ 显示某个文件所占的磁盘空间。

```
[root@localhost other]# du 111 //显示文件 111 所占的磁盘空间
16 111
[root@localhost other]#
```

本例使用指令“du 111”显示文件 111 所占的磁盘空间。

◎ 范例 679 ◎ 显示多个文件所占的磁盘空间。

```
[root@localhost helloworld]# du config.h config.h.in config.log //显示多个文件所占的磁盘空间
```

```
12 config.h
12 config.h.in
112 config.log
[root@localhost helloworld]#
```

本例使用指令“du config.h config.h.in config.log”分别显示文件 config.h、config.h.in、config.log 所占的磁盘空间。在输出结果中，文件前面的数字就是文件所占的磁盘空间，如文件 config.h 前的数字为 12，表示文件 config.h 所占的磁盘空间为 12 字节。

◎ 范例 680 ◎ 显示目录所占的磁盘空间。

```
[root@localhost helloworld]# du debug //显示目录所占的磁盘空间
28 debug/doc/en
60 debug/doc
80 debug/src/.deps
8 debug/src/.libs
860 debug/src
24 debug/po
1372 debug
[root@localhost helloworld]#
```

本例使用指令“du debug”显示目录 debug 所占的磁盘空间。

◎ 范例 681 ◎ 只显示总和的大小。

```
[root@localhost helloworld]# du -s debug/ //显示目录所占空间
1372 debug/
[root@localhost helloworld]#
```

本例使用指令“du -s debug”显示目录 debug 所占的总的空间大小。

◎ 范例 682 ◎ 以更便于用户阅读的方式显示目录文件所占的磁盘空间。

```
[root@localhost other]# du -h //显示目录文件所占的磁盘空间
56K ./DJ
28K ./helloworld/debug/doc/en
60K ./helloworld/debug/doc
80K ./helloworld/debug/src/.deps
8.0K ./helloworld/debug/src/.libs
860K ./helloworld/debug/src
24K ./helloworld/debug/po
1.4M ./helloworld/debug
92K ./helloworld/doc/en
156K ./helloworld/doc
80K ./helloworld/src/.deps
8.0K ./helloworld/src/.libs
320K ./helloworld/src
1.2M ./helloworld/autcm4te.cache
48K ./helloworld/po
24K ./helloworld/templates
1.1M ./helloworld/admin
6.4M ./helloworld
6.4M .

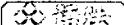
[root@localhost other]#
```

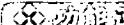
本例使用指令“du -h”以更便于用户识别的方式，显示当前文件夹下的目录和文件所占的磁盘空间的大小。

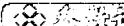
### 相关指令

df、wc、stat

## 8.5 mt 指令：磁带驱动操作

 mt[选择性参数][必要参数][选择性参数]

 mt 指令用来进行磁带驱动的操作。

 具体说明

必要参数的具体说明如下：



## 第8章 磁盘管理

参数	功能
fsf<数字>	使磁带定位在指定文件的下一个文件所在的一个块上
Fsfm<数字>	使磁带定位在指定文件的前一个文件所在的最后一个块上
bsf<数字>	使磁带定位在指定文件的前一个文件所在的最后一个块上
bsfm<数字>	使磁带定位在指定文件的下一个文件所在的一个块上
asf<数字>	磁带定位在指定文件的开始
fsr<数字>	前进指定数目的记录
bsr<数字>	后退指定数目的记录
fss<数字>	SCSI 磁带前进指定数目的设置标记 (setmarks)
bss<数字>	SCSI 磁带后退指定数目的设置标记
eof,seod	到可用数据的末端
rewind	回卷磁带
offline,rewoffl,eject	回卷磁带, 如果可能的话, 卸载磁带
retention	回卷磁带, 然后回卷磁带
weof,eof<数字>	在当前位置写入指定的 EOF 字符
wset<数字>	在当前位置写入指定的设置标记(setmark)
erase	擦除磁带
status	打印磁带的当前状态信息
seek<数字>	在磁带上搜寻指定的块
tell	判断当前的块
setpartition<数字>	切换到指定的分区

选择性参数的具体说明如下:

参数	功能
-h	列出帮助信息
-v	列出版本信息
-<设备>	指定要操作的设备



st

## 8.7 quota 指令：显示磁盘已使用的空间与限制

## 8.6

## mzip 指令：zip/jaz 磁盘驱动器控制指令



mzip[选择性参数]



在 zip/jaz 驱动器上改变保护模式或者退出盘



选择性参数的具体说明如下：

参 数	功 能
-e	退出磁盘
-f	与“e”共同使用强制退出磁盘
-r	写保护磁盘
-w	去除写保护
-p	密码写保护
-x	密码保护
-u	退出磁盘前暂时解除磁盘的保护状态
-q	查询状态



zip

## 8.7

## quota 指令：显示磁盘已使用的空间与限制



quota[必要参数][选择性参数][用户名称或者群组名称]



quota 指令用来显示磁盘的信息。



必要参数的具体说明如下：

参 数	功 能
-g	群组的磁盘信息
-q	列表的简化



续表

参数	功能
-u	列出磁盘空间限制
-v	显示磁盘限制的文件系统

选择性参数的具体说明如下：

参数	功能
-version	版本信息

### ☆ 次执行范例

◎ 范例 683 ◎ 显示磁盘使用情况。

```
-sh-2.05b# quota iloveblues ilove //显示磁盘的使用情况
Disk quotas for user iloveblues (uid 502): none
Disk quotas for user ilove (uid 501): none
-sh-2.05b#
```

本例使用指令“quota iloveblues ilove”显示用户 ilove 和 iloveblues 的磁盘使用情况。

### ☆ 相关指令

quotacheck、quotaoff、quotaon

## 8.8

## quotacheck 指令：检查磁盘的使用空间与限制

### ☆ 基本用法

quotacheck[必要参数][选择性参数][文件系统]

☆ 功能说明 quotacheck 命令用于扫描挂载的文件系统并设置磁盘的空间限制。

### ☆ 选择说明

必要参数的具体说明如下：

参数	功能
-a	扫描所有的分区
-b	备份

## 8.9 quotaoff 指令：关闭磁盘空间限制

续表

参数	功能
-d	调试模式
-c	重新创建文件
-f	强迫检查
-i	交互模式
-v	运行时显示详细的处理信息
-g	显示群组所占的目录文件数
-u	显示每个用户所占的目录文件数

选择性参数的具体说明如下：

参数	功能
-h	帮助信息
-version	版本信息

**范例 684 检查磁盘配额**

● 范例 684 ● 检查磁盘配额。  
 [root@localhost ~]#quotacheck -augv //检测磁盘配额  
 本例使用指令“quotacheck -augv”检测磁盘配额。

**相关命令**

quota、quotaoff、quotaon

## 8.9 quotaoff 指令：关闭磁盘空间限制

**主要参数** quotaoff[必要参数][选择性参数][文件系统]

**功能说明** quotaoff 指令用来关闭用户的磁盘空间的限制。

**参数说明**

必要参数的具体说明如下：

参数	功能
-a	关闭所有的磁盘配额限制
-g	组磁盘配额限制

续表

参数	功能
-p	列出状态
-u	用户配额限制
-v	运行时显示详细的处理信息

选择性参数的具体说明如下：

参数	功能
-help	帮助信息
-version	版本信息

#### ❖ 执行范例

① 范例 685 ① 关闭配额限制。

```
[root@localhost ~]# quotaoff -a //关闭配额限制
```

本例使用指令“quotaoff -a”关闭所有文件系统的配额限制。

#### ❖ 相关指令

quota、quotacheck、quotaon

## 8.10 quotaon 指令：开启磁盘空间限制

❖ 基本 quotaon[必要参数][选择性参数][文件系统]

❖ 功能说明 quotaon 指令用来开启磁盘空间限制。

#### ❖ 参数说明

必要参数的具体说明如下：

参数	功能
-a	关闭所有的磁盘配额限制
-g	组磁盘配额限制
-p	列出状态
-u	用户配额限制
-v	运行时显示详细的处理信息

## 8.11 quotastats 指令：显示磁盘空间的限制

选择性参数的具体说明如下：

参    数	功    能
<code>-help</code>	帮助信息
<code>-version</code>	版本信息

### ○○○执行范例

- 范例 686 ○ 开启磁盘空间限制。

```
[root@localhost ~]# quotaon -a //开启磁盘限制
```

本例使用指令“`quotaon -a`”开启所有用户磁盘空间的限制。

### ○○○相关指令

`quota`、`quotacheck`、`quotaoff`

## 8.11 quotastats 指令：显示磁盘空间的限制

### ○○○命令

`quotastats`

○○○功能说明

`quotastats` 指令用来显示系统的磁盘空间限制的当前状态。

### ○○○执行范例

- 范例 687 ○ 显示磁盘配额限制状态。

```
[root@localhost ~]# quotastats //显示磁盘配额限制的状态
Kernel quota version: 6.5.1
Number of dquot lookups: 0
Number of dquot drops: 0
Number of dquot reads: 0
Number of dquot writes: 0
Number of quotaf file syncs: 19
Number of dquot cache hits: 0
Number of allocated dquots: 0
Number of free dquots: 0
Number of in use dquot entries (user/group): 0
[root@localhost ~]#
```

本例使用指令 `quotastats` 显示当前系统下磁盘配额限制的状态。


◆ 相关指令

quota、quotacheck、quotaoff、quotaon

## 8.12

# raidstop 指令：关闭软件控制的磁盘阵列

◆ 语法

raidstop[必要参数][选择性参数][磁盘]

◆ 功能说明

raidstop 指令用来关闭磁盘阵列。

◆ 大家说明

必要参数的具体说明如下：

参 数	功 能
-a	关闭所有的磁盘阵列

选择性参数的具体说明如下：

参 数	功 能
-<文件>	指定配置文件
-help	帮助信息
-version	版本信息

◆ 教你范例

◎ 范例 688 ◎ 关闭所有的磁盘阵列。

```
[root@localhost ~]# raidstop -a //关闭所有的磁盘阵列
```

本例使用指令“raidstop -a”关闭所有的软件控制的磁盘阵列。

◆ 相关指令

raidstart

## 8.13

# repquota 指令：检查磁盘空间限制的状态

◆ 语法

repquota[必要参数][文件系统]

**功能说明** 执行 repquota 指令可以显示磁盘空间的限制以及磁盘的使用情况。

**参数说明**

必要参数的具体说明如下：

参数	功能
-a	列出所有文件系统的情况
-g	显示群组的使用情况
-n	以数字的形式显示
-s	显示详细的情况
-u	显示用户的磁盘使用情况
-v	显示所有文件系统的磁盘使用情况

**运行范例**

◎ 范例 689 ◎ 显示所有文件系统的磁盘使用情况。

```
[root@localhost ~]# repquota -a //显示文件系统的磁盘使用情况
```

本例使用指令“repquota -a”显示/etc/mtab 中定义的 quota 文件系统的磁盘使用情况。

**相关指令**

quota

## 8.14 rmt 指令：远端磁带传输协议模块

**语法** rmt

**功能说明** rmt 指令通过进程间通信远程控制磁带机。

**相关指令**

rcmd、rexec、rdump、rrestore

## 8.15 sfdisk 指令：硬盘分区工具程序

**语法**

sfdisk[必要参数][选择性参数]



**功能说明** sfdisk 有 4 个主要的用途：显示分区大小、显示设备上的分区、检测设备上的分区和对设备进行重新分区。

**参数说明**

必要参数的具体说明如下：

参数	功能
-s	显示分区大小
-c	打印分区 ID 的变化
-l	列出每个设备的分区
-i	从 1 开始排序
-T	列出已知的分区类型
-D	兼容 DOS
-R	让内核重新读取分区表
-n	测试

选择性参数的具体说明如下：

参数	功能
-O<文件>	备份到文件
-I<文件>	从文件恢复
-version	版本信息
-help	帮助信息

**范例 690 显示分区信息**

```
-sh-2.05b# sfdisk -l //显示分区信息
Disk /dev/sda: 1044 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting
from 0
 Device Boot Start End #cyls #blocks Id System
 /dev/sda1 * 0+ 12 13- 104391 83 Linux
 /dev/sda2 13 77 65 522112+ 82 Linux swap
 /dev/sda3 78 1043 966 7759395 83 Linux
```

## 8.16 sync 指令：将内存缓冲区内的数据写入磁盘

```
/dev/sda4 0 - 0 0 0 Empty
-sh-2.05#
```

本例使用指令“`sfdisk -l`”显示分区信息。

 **相关指令**

`fdisk`

## 8.16

### sync 指令：将内存缓冲区内的数据写入磁盘

 **语法** `sync[选择性参数]`

 **功能说明** `sync` 指令将缓冲区数据写入磁盘。

 **大致说明**

选择性参数的具体说明如下：

参数	功 能
<code>-help</code>	帮助信息
<code>-version</code>	版本信息

 **执行范例**

◎ 范例 691 ◎ 将内存数据写入磁盘。

```
[root@localhost ~]# sync //将内存数据写入磁盘
```

本例使用指令 `sync` 将内存数据写入磁盘。

 **相关指令**

`init`、`shutdown`

<http://www.TopSage.com>

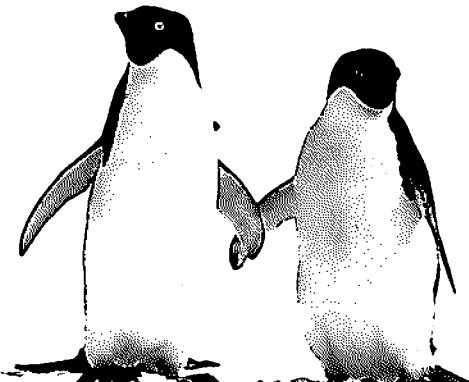




## 第9章

### 磁盘维护

磁盘维护是系统维护的重要部分。在 Linux 系统中磁盘维护主要包括磁盘的检测、磁盘的修复等。本章将详细介绍相关的指令。





## 9.1

**badblocks 指令：检查磁盘坏道**

 **技巧** badblocks[必要参数][选择性参数][磁盘][磁盘块数][启动区块]

 **功能说明** badblocks 指令用来检查磁盘，指定磁盘装置和磁盘块数来进行磁盘的坏道检查，以及进行相应的修复。badblocks 是 e2fsprogs 包的一部分。

 **参数说明**

必要参数的具体说明如下：

参数	功能
-s	显示检查进度
-v	运行时显示详细的处理信息
-w	做写入检测

选择性参数的具体说明如下：

参数	功能
-b<字节数>	指定区块的字节数
-o<文件>	将检测结果写入指定文件
[磁盘]	指定检测的磁盘
[磁盘块数]	指定磁盘的块数
[启动区块]	指定开始检测的起始区块

 **执行实例**

◎ 范例 692 ◎ 检测磁盘的坏道。

[root@localhost root]# badblocks /dev/fd0H720 720//检测磁盘的坏道

检测指定磁盘/dev/fd0H720 的坏道信息，指定区块的大小为 720 字节。

 **相关指令**

e2fsck、mke2fs

## 9.2 cfdisk 指令：磁盘分区

 **语法** cfdisk[必要参数][选择性参数]

 **功能说明** cfdisk 指令是一个基于鼠标的、用于硬盘分区的程序。硬盘装置的典型参数值如下所示。

- /dev/hda[default]。
- /dev/hdb。
- /dev/sda。
- /dev/sdb。
- /dev/sdc。
- /dev/sdd。

 **参数说明**

必要参数的具体说明如下：

参数	功能
-a	用箭头表示选取，而不是以反白表示
-P	显示磁盘分区的内容
-z	当作新磁盘直接使用

选择性参数的具体说明如下：

参数	功能
-c<数目>	指定柱面数目，而不考虑 BIOS 的设置
-h<数目>	指定磁头数目，而不考虑 BIOS 的设置
-s<数目>	指定磁盘分区数目，而不考虑 BIOS 的设置
-v	版本信息

 **操作范例**

◎ 范例 693 ◎ 进行磁盘分区。

root@localhost:~# cfdisk

//进行磁盘分区



输入指令 `cfdisk` 之后，将出现上面的图形界面，它是基于鼠标进行操作的，利用它可以进行磁盘的分区操作。

- ◎ 范例 694 ◎ 使用箭头进行操作，而不使用白色表示。

`root@localhost:~# cfdisk -a//`进行磁盘分区，使用箭头进行操作，而不使用反白表示

- ◎ 范例 695 ◎ 指定磁盘分区数目进行磁盘分区。

`root@localhost:~# cfdisk -s 3//`进行磁盘分区，使用箭头进行操作，而不使用反白表示



`fdisk`

## 9.3 dd 指令：转换复制文件

 `dd[选择性参数]`

 复制文件，或根据参数进行转换和格式化。



选择性参数的具体说明如下：

参数	功能
<code>bs=&lt;字节数&gt;</code>	设置输入/输出的字节数
<code>cbs=&lt;字节数&gt;</code>	设置转换的字节数
<code>conv=&lt;关键字&gt;</code>	转换方式
<code>cout=&lt;区块数&gt;</code>	指定读取的区块数

续表

参    数	功    能
ibs=<字节数>	设置每次读取的字节数
if=<文件>	指定文件作为读取的对象
obs=<字节数>	设置每次输出的字节数
of=<文件>	结果输出到指定的文件
seek=<区块数>	忽略指定的区块数，不进行输出
skip=<区块数>	忽略指定的区块数，不进行读取
-help	帮助信息
-version	版本信息

**执行范例**

## ◎ 范例 696 ◎ 压缩文件。

```
[root@localhost ~]# dd if=abc of=abc.bak //压缩文件
读入了 0+1 个块
输出了 0+1 个块
[root@localhost ~]#
```

本例使用指令“dd if=abc of=abc.bak”压缩文件，输入的文件为 abc，输出的文件为 abc.bak，压缩之后源文件仍然存在。

## ◎ 范例 697 ◎ 指定输入输出字节数。

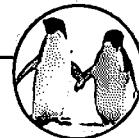
```
[root@localhost ~]# dd if=abc of=abc.bak bs=256 //压缩文件
读入了 0+1 个块
输出了 0+1 个块
[root@localhost ~]#
```

本例使用指令“dd if=abc of=abc.bak bs=256”压缩文件，并且指定文件的输入和输出的字节数为 256。

## ◎ 范例 698 ◎ 设定读取的区块数。

```
[root@localhost ~]# dd if=abc of=abc.bak bs=256 cbs=1 //压缩文件
读入了 0+1 个块
输出了 0+1 个块
[root@localhost ~]#
```

本例使用指令“dd if=abc of=abc.bak bs=256 cbs=1”压缩文件，并且指定读取的区块数为 1。



**相关指令**

cpio、tar

## 9.4

# e2fsck 指令：检查 ext2 文件系统

**操作**

e2fsck[必要参数][选择性参数][磁盘]

**功能说明**

e2fsck 指令用来检查 ext2 文件系统的正确性。

**参数说明**

必要参数的具体说明如下：

参 数	功 能
-a	自动修复模式
-C	记录检查信息
-f	强制检查
-n	只读模式
-p	自动修复
-t	显示时间
-v	执行时显示详细的处理信息

选择性参数的具体说明如下：

参 数	功 能
-V	显示版本信息
-B<字节数>	设置区块大小

**执行范例**

◎ 范例 699 ◎ 检查磁盘。

```
[root@localhost chord]# e2fsck /dev/hdc //检查磁盘
e2fsck 1.37 (21-Mar-2005)
e2fsck: Read-only file system while trying to open /dev/hdc
Disk write-protected; use the -n option to do a read-only
check of the device.
[root@localhost chord]#
```

## 9.5 mkbootdisk 指令：建立目前系统的启动盘

本例使用指令 e2fsck /dev/hdc 检查 ext2 文件系统。



fsck

## 9.5

## mkbootdisk 指令：建立目前系统的启动盘



mkbootdisk[选择性参数][必要参数]



制作的启动盘完全自包含，而且包含了一个初始化 RAM 盘的镜像。这个镜像将载入所有系统必须的 SCSI 模块。创建的启动盘根据/etc/fstab 的建议寻找设备上的根文件系统。惟一需要的参数是要写入启动盘的内核版本。



必要参数的具体说明如下：

参数	功能
<内核版本>	制作的启动盘的内核版本

选择性参数的具体说明如下：

参数	功能
--device<设备文件>	指定启动盘放置的设备和设备文件，默认为/dev/fd0
--noprompt	不输出提示符引导用户插入软盘
--verbose	打印程序执行信息
--kernelargs<参数>	附加指定的参数到内核命令行
--size<大小>	指定镜像文件的大小，默认为标准的 1.44MB



◎ 范例 700 ◎ 制作一张启动盘。

```
[root@yingzheng modules]# mkbootdisk 2.6.11-1.1369_FC4
Insert a disk in /dev/fd0. Any information on the disk will be lost.
```



grubby、mkinitrd



## 9.6 mke2fs 指令：建立 ext2 文件系统

**语法** mke2fs[必要参数][选择性参数]

**功能说明** 建立 Linux 系统使用的 ext2 或者 ext3 文件系统。

**参数说明**

必要参数的具体说明如下：

参数	功 能
<设备名称>	与设备对应的文件，例如 /dev/hd1
-b<块大小>	指定块大小
-c	在创建文件系统之前检查指定的设备
-g<块组数量>	指定一个块组中块的数量

选择性参数的具体说明如下：

参数	功 能
-E<扩展参数>	为要创建的文件系统指定一些参数
-f<不连续区段大小>	指定不连续区段的大小
-F	强制执行，即使指定设备没有被挂载或者不是块设备
-l<文件>	从指定的文件中读取坏的块列表
-i<字节>	指定每个 inode 的字节数
-j	创建使用 ext3 卷号的文件系统
-J<扩展参数>	为使用 ext3 卷号的文件系统指定一些参数
-L	设置创建的文件系统的标签
-N	指定 inode 的数量
-q	执行时不显示任何信息
-V	显示版本信息
-E<扩展选项>	为要创建的文件系统指定一些可选项
-f<不连续区段大小>	指定不连续区段的大小

## 9.7 mkfs 指令：建立各种文件系统

续表

参 数	功 能
-F	强制执行，即使指定设备没有被挂载或者不是块设备
-l<文件>	从指定的文件中读取坏的块列表
-i<字节>	指定每个 inode 的字节数
-j	创建使用 ext3 卷号的文件系统
-J<扩展选项>	为使用 ext3 卷号的文件系统指定一些选项
-L	设置创建的文件系统的标签
-N	指定 inode 的数量
-q	执行时不显示任何信息
-V	显示版本信息

**执行范例**

- 范例 701 ○ 创建指定的 ext2 文件系统。

```
[root@localhost root]# mke2fs -q /dev/hda1
```

**相关指令**

badblocks、dump2fs、e2fsck、tune2fs

**9.7****mkfs 指令：建立各种文件系统**

**功能说明** mkfs[选择性参数][必要参数][选择性参数]

**功能说明** 必要参数为指定的设备或者挂载点，后一个选择性参数是指定文件系统的分区数，常用于硬盘分区。事实上，mkfs 只是不同文件系统构建工具的一个前端，不同的系统实际使用的工具不同。

**参数说明**

必要参数的具体说明如下：

参 数	功 能
<文件系统>	指定设备或挂载点

选择性参数的具体说明如下：



参数	功能
-V	提供详细输出信息
-t<文件系统类型>	指定创建的文件系统类型，默认为 ext2
<文件系统特有参数>	针对不同的文件系统的不同参数，这些参数将被传到实际的文件系统构建工具

### ◎ 常用命令

- 范例 702 ○ 将一个软盘创建为 ext2 文件系统。

```
[root@yingzheng 100dpi]#mkfs /dev/fd0 2
```

**相关指令** fs、badblocks、fsck、mkdosfs、mke2fs、mkfs.bfs、mkfs.ext2、mkfs.ext3、mkfs.minix、mkfs.msdos、mkfs.vfat、mkfs.xfs、mkfs.xiafs

## 9.8 mkfs.minix 指令：建立 minix 文件系统

### ◎ 命令

mkfs.minix[选择性参数][必要参数][选择性参数]

**功能说明** mkfs 的一种实际构建工具。中间的必要参数是设备或者挂载点，后面的选择性参数是块大小。

### ◎ 参数说明

必要参数的具体说明如下：

参数	功能
<设备名>	设备或者挂载点

选择性参数的具体说明如下：

参数	功能
-c	检查设备上的坏块的数量
-n<文件名长度>	指定文件名的最大长度
-i<inode 数量>	指定文件系统的 inode 的数量
-l<文件>	从指定文件中读取坏块的数量
-v	制作 Minix 版本 2 的文件系统

## 9.9 mkinitrd 指令：建立要载入 ramdisk 的映像文件



◎ 范例 703 ◎ 把软盘制作作为 Minix 文件系统

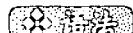
```
[root@yingzheng 100dpi]#mkfs /dev/fd0
```



**mkfs**

### 9.9

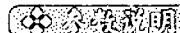
## mkinitrd 指令：建立要载入 ramdisk 的映像文件



**mkinitrd[选择性参数][必要参数]**



根据内核版本和指定参数生成映像文件。



必要参数的具体说明如下：

参    数	功    能
<映像文件>	指定映像文件名称和路径
<内核版本>	指定所依据的内核版本

选择性参数的具体说明如下：

参    数	功    能
<b>-builtin=&lt;模块&gt;</b>	认为指定模块已经装入内核，忽略错误
<b>-f</b>	允许覆盖已存在的映像文件
<b>-image-version</b>	内核版本号将附加到建立的映像文件的目录前
<b>-fstab=&lt;文件系统列表&gt;</b>	使用列表自动探测根设备所建立的文件系统类型
<b>-nocompress</b>	不压缩生成的映像文件
<b>-nopivot</b>	不使用 pivot_root 系统调用作为映像的一部分
<b>-omit-lvm-modules</b>	不载入任何 lvm 模块
<b>-omit-scsi-modules</b>	不载入任何 scsi 模块
<b>-preload=&lt;模块&gt;</b>	将指定的模块载入映像中
<b>-v</b>	在创建映像过程中打印信息
<b>-version</b>	打印程序版本信息


◎ 执行范例

- ◎ 范例 704 ◎ 在/boot 目录下建立一个映像文件

```
[root@yingzheng 100dpi]# mkinitrd /boot/initrd-2.2.5-15.img 2.2.5-15
```

◎ 相关命令

fstab、insmod、kerneld、lilo

## 9.10

## mkisofs 指令：建立 iso 9660 映像文件

◎ 语法

**mkisofs[选择性参数][必要参数]**

◎ 功能说明

根据参数生成 iso9660 镜像文件。

◎ 处理说明

必要参数的具体说明如下：

参 数	功 能
<目录>	将被制作成 ISO 镜像的目录

选择性参数的具体说明如下：

参 数	功 能
-o<文件>	指定将生成的文件名称
-x<目录>	指定将不被制作的目录
-v	制作时输出过程信息
-J	生成 Joliet 命名记录

◎ 执行范例

- ◎ 范例 705 ◎ 将文件 fonts.dir 制作为 iso9660 映像。

```
[root@yingzheng yingzheng]# mkisofs -o testiso fonts.dir
```

```
INFO: UTF-8 character encoding detected by locale settings.
```

```
Assuming UTF-8 encoded filenames on source filesystem,
use -input-charset to override.
```

```
Total translation table size: 0
```

```
Total rockridge attributes bytes: 0
```

```
Total directory bytes: 0
```

## 9.11 mkswap 指令：设置交换区(swap area)

Path table size(bytes): 10  
Max brk space used 0  
174 extents written (0 MB)



cdrecord、mkzfstree、magic、apple\_driver

## 9.11

## mkswap 指令：设置交换区(swap area)



**mkswap**[选择性参数][必要参数][选择性参数]



在一个文件或者设备上建立交换区。在建立完之后要使用 swapon 指令开始使用这个交换区。最后一个选择性参数指定了交换区的大小，但是这个参数是为了后向兼容设置的，没有使用的必要。一般都将整个文件或者设备作为交换区。

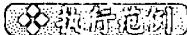


必要参数的具体说明如下：

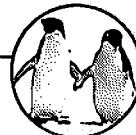
参    数	功    能
<设备>	指定将作为交换区的设备或者文件

选择性参数的具体说明如下：

参    数	功    能
-c	检查坏区
-f	强迫执行指令，不管指令是否合理
-p<页大小>	指定使用的页大小
-L<标签>	指定一个标签，此后 swapon 可以使用这个标签
-v0	建立一个旧式的交换区
-v1	建立一个新式的交换区



- 范例 706 ○ 指定一个设备作为交换区。



[root@yingzheng yingzheng]# mkswap -c -v1 -L swapnew /dev/hda3



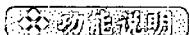
fdisk、swapon

## 9.12

### restore 指令：还原 dump 操作备份的文件



restore[必要参数][选择性参数][文件]



restore 指令执行 dump 指令的反向功能。一个文件系统的完全备份可能被回复。单个文件的子树也许会从全部的或者部分的备份中恢复。restore 指令可以通过网络工作，具体的细节可参见“-f”参数。其他的参数是文件或者目录，用来指定那些将要被恢复的文件。除非“-h”参数指定，否则所有的目录名指的是当前目录以及其下面的子目录。



必要参数的具体说明如下：

参数	功能
-c	备份文件的格式为旧的格式
-C	对比文件和备份文件
-d	启动调试模式
-i	交互模式
-m	打开目录
-r	进行恢复操作
-R	恢复时进行编号
-t	设置文件名称
-v	运行时显示详细的处理过程
-V	打开设备，如 CD-ROM
-x	指定文件的名称
-y	对于交互模式下出现的问题都回答“y”即同意

选择性参数的具体说明如下：

## 9.12 restore 指令：还原 dump 操作备份的文件

参数	功能
-b<字节数>	指定区块的大小
-D<文件系统>	指定文件系统
-f<文件>	读取备份文件
-s<编号>	指定编号

### ○○○运行范例

#### ○ 范例 707 ○ 数据恢复。

```
[root@localhost ~]# dump -0 -f /home/kk/bu /boot/ //备份数据
DUMP: Date of this level 0 dump: Mon May 28 14:53:32 2007
DUMP: Dumping /dev/sdal (/boot) to /home/kk/bu
DUMP: Label: /boot
DUMP: Writing 10 Kilobyte records
DUMP: mapping (Pass I) [regular files]
DUMP: mapping (Pass II) [directories]
DUMP: estimated 5679 blocks.
DUMP: Volume 1 started with block 1 at: Mon May 28 14:53:32 2007
DUMP: dumping (Pass III) [directories]
DUMP: dumping (Pass IV) [regular files]
DUMP: Closing /home/kk/bu
DUMP: Volume 1 completed at: Mon May 28 14:53:33 2007
DUMP: Volume 1 5730 blocks (5.60MB)
DUMP: Volume 1 took 0:00:01
DUMP: Volume 1 transfer rate: 5730 kB/s
DUMP: 5730 blocks (5.60MB) on 1 volume(s)
DUMP: finished in 1 seconds, throughput 5730 kBytes/sec
DUMP: Date of this level 0 dump: Mon May 28 14:53:32 2007
DUMP: Date this dump completed: Mon May 28 14:53:33 2007
DUMP: Average transfer rate: 5730 kB/s
DUMP: DUMP IS DONE
[root@localhost ~]# restore -r -f /home/kk/bu //恢复数据
[root@localhost ~]#
```

本例使用指令“dump -0 -f /home/kk/bu /boot/”将/boot/下的数据备份到文件/home/kk/bu；使用指令“restore -r -f /home/kk/bu”恢复备份的数据。

### ○○○相关指令

dump、mount、mke2fs、rmt



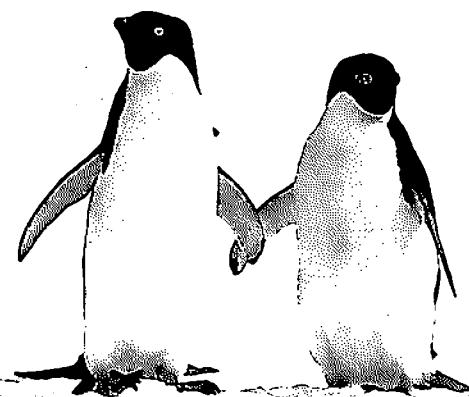


TopSage.com

## 第10章

### 程序编译

程序编译指令的功能是对 Linux 系统的应用程序进行编译安装。这些指令的主要功能包括生成头文件、生成配置文件、编译、安装生成可执行文件等。





## 10.1 as 指令：标准 GNU 汇编程序

as[必要参数] [选择性参数]文件

as 指令可以利用指定的汇编文件生成目标文件，也就是由.s 文件生成.o 文件。

必要参数的具体说明如下：

参 数	功 能
-a	开启列表
-alternate	开启可选择的宏语法
-D	产生汇编器调试信息
-defsym SYM=VAL	定义符号 SYM 的值为指定的值
-execstack	要求可执行的堆栈
-noexecstack	不要求堆栈
-f	跳过空白和内容的预处理
-g -gen-debug	产生调试信息
-gstabs	产生 STABS 调试信息
-gdwarf-2	产生 DWARF2 信息
-target-help	显示目标指定的参数
-I <目录>	增加指定的目录到搜索列表中
-J	溢出时不显示警告信息
-L	保持本地符号信息
-M	兼容性模式
-MD <文件>	向文件中写入依赖信息
-o <目标文件>	命名输出的目标文件（默认为 a.out）
-R	生成目标文件后删除源文件

## 10.1 as 指令：标准 GNU 汇编程序

续表

参数	功能
--statistics	从执行中打印静态变量信息
-W	禁止警告信息
--warn	不禁止警告信息
-q	忽略部分警告信息

选择性参数的具体说明如下：

参数	功能
-help	帮助信息
--version	版本信息
-V	打印汇编器版本序列

**执行范例**

## ◎ 范例 708 ◎ 显示 as 的帮助信息。

root@localhost:-# as --help //显示帮助信息

Usage: as [option...] [asmfile...]

## Options:

-a[sub-option...]	turn on listings
	Sub-options (default hls):
c	omit false conditionals
d	omit debugging directives
h	include high-level source
l	include assembly
m	include macro expansions
n	omit forms processing
s	include symbols
=FILE	list to FILE (must be last sub-option)
--alternate	initially turn on alternate macro syntax
-D	produce assembler debugging messages
--defsym SYM=VAL	define symbol SYM to given value
--execstack	require executable stack for this object



--noexecstack	don't require executable stack for this object
-f	skip whitespace and comment preprocessing
-g --gen-debug	generate debugging information
--gstabs	generate STABS debugging information
--gstabs+	generate STABS debug info with GNU extensions
--gdwarf-2	generate DWARF2 debugging information
--hash-size=<value>	set the hash table size close to <value>
--help	show this message and exit
--target-help	show target specific options
-I DIR	add DIR to search list for .include directives
-J	don't warn about signed overflow
-K	warn when differences altered for long displacements
-L,--keep-locals	keep local symbols (e.g. starting with 'L')
-M,--mri	assemble in MRI compatibility mode
--MD FILE	write dependency information in FILE (default none)
-nocpp	ignored
-o OBJFILE	name the object-file output OBJFILE (default a.out)
-R	fold data section into text section
--reduce-memory-overheads	prefer smaller memory use at the cost of longer assembly times
--statistics	print various measured statistics from execution
--strip-local-absolute	strip local absolute symbols
--traditional-format	Use same format as native assembler when possible
--version	print assembler version number and exit
-W --no-warn	suppress warnings
--warn	don't suppress warnings
--fatal-warnings	treat warnings as errors
--itbl INSTTBL	extend instruction set to include instructions matching the specifications defined in file INSTTBL
-w	ignored
-X	ignored
-Z	generate object file even after errors

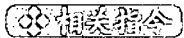
## 10.2 autoconf 指令：产生配置脚本

--listing-lhs-width set the width in words of the output data column of the listing  
 --listing-lhs-width2 set the width in words of the continuation lines of the output data column; ignored if smaller than the width of the first line  
 --listing-rhs-width set the max width in characters of the lines from the source file  
 --listing-cont-lines set the maximum number of continuation lines used for the output data column of the listing  
 @FILE read options from FILE  
 -Q ignored  
 -v print assembler version number  
 -k ignored  
 -n Do not optimize code alignment  
 -q quieten some warnings  
 -s ignored  
 --divide ignored

Report bugs to <URL:<http://www.sourceforge.org/bugzilla/>>

root@localhost:~#

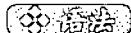
本例使用指令“as -help”显示 as 指令的帮助信息。



gcc、ld

## 10.2

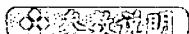
## autoconf 指令：产生配置脚本



autoconf[必要参数][选择性参数][模板文件]



autoconf 指令将根据给定的模板文件产生一个配置脚本。这个给定的模板的形式为 configure.ac 或者 configure.in。如果模板文件给定，那么结果将输出到标准输出或者输出到 configure 文件。



必要参数的具体说明如下：



参数	功能
-v	详细报告处理过程
-d	输出调试信息
-f	认为所有的文件都是过期的
-W	报告警告信息

选择性参数的具体说明如下：

参数	功能
-h	帮助信息
-V	版本信息
-o<文件>	将输出写入指定的文件

### 【警告信息】

输出的警告信息可能包括信息。

- cross: 交叉编译。
- obsolet: 旧的架构。
- syntax: 不确定的语法结构。
- all: 所有的警告信息。
- no-CATEGORY: 关闭警告信息。
- none: 关闭所有的警告。
- error: 出现错误。

### ◎执行范例

◎ 范例 709 ◎ 产生执行脚本。

```
[root@localhost root]# autoconf //产生执行脚本
```

运行 autoconfig 指令产生 configure 文件,为下一步的配置工作做准备。

### ◎相关命令

automake、autoreconf、autoupdat、autoheader、autoscan、config.sub、config.guess、ifnames、libtool、automount

### 10.3 autoheader 指令：为 configure 产生模板头文件

## 10.3

### autoheader 指令：为 configure 产生模板头文件



**autoheader[必要参数][选择性参数][模板文件]**



**功能说明** autoheader 指令用来产生模板头文件，以便于后面生成的 configure 文件的使用。运行此指令时，其将扫描模板文件、configure.ac（如果存在的话）或者 configure.in（如果存在的话）。



必要参数的具体说明如下：

参    数	功    能
-v	详细报告处理的全过程
-d	输出调试信息
-f	认为所有的文件都是过期的
-W	报告警告信息

选择性参数的具体说明如下：

参    数	功    能
-h	帮助信息
-V	版本信息

#### 【警告信息】

输出的警告信息可能包括如下信息。

- obsolet:** 旧的架构。
- all:** 所有的警告信息。
- no-CATEGORY:** 关闭警告信息。
- none:** 关闭所有的警告。
- error:** 出现错误。



**范例 710** 产生模板头文件。

```
[root@localhost root]# autoheader //产生模板头文件
```

autoheader 指令用于开发软件包或者安装相应的软件时，如果当前文



件夹下存在 `configure.in` 或者 `configure.ac` 文件，可以直接运行 `autoheader` 指令来产生相应的头文件，为下一步的配置工作做准备。

### 【相关指令】

`automake`、`autoreconf`、`autoupdat`、`autoconf`、`autoscan`、`config.sub`、`config.guess`、`ifnames`、`libtool`、`automount`

## 10.4

### autoreconf 指令：更新已经生成的配置文件

 **语法** `autoreconf[必要参数][选择性参数][文件]`

 **功能说明** 执行指令 `autoconf` 可以重复地编译指定目录下的系统文件或者 `CONFIGURE-AC` 指定的目录树。默认情况下，它只重新编译比以前编译的文件更新的文件。

 **参数说明**

必要参数的具体说明如下：

参 数	功 能
<code>-d</code>	不删除临时文件
<code>-f</code>	认为所有的文件为老版本的文件
<code>-i</code>	复制辅助文件
<code>-s</code>	创建符号链接，而不是复制
<code>-m</code>	当可用时，重新运行指令 <code>/configure</code> 和 <code>make</code>
<code>-W</code>	报告语法错误信息

选择性参数的具体说明如下：

参 数	功 能
<code>-h</code>	帮助信息
<code>-V</code>	版本信息

### 【警告信息】

输出的警告信息和 `autoheader` 指令相同。

## 10.5 autoscan 指令：产生初步的 configure.in 文件

### ❖ 执行范例

- ① 范例 711 ② 更新执行脚本。

```
[root@localhost root]# autoreconf //更新执行脚本
```

指令 autoconf 用来产生 configure 文件，为下一步的配置工作做准备。当需要更新 configure 文件时可以直接运行 autoreconf 指令。

### ❖ 执行范例

automake、autoconf、autoupdat、autoheader、autoscan、config.sub、config.guess、ifnames、libtool、automount

## 10.5

## autoscan 指令：产生初步的 configure.in 文件

**❖ 语法**    autoscan[必要参数][选择性参数][源目录]

**❖ 功能说明**    autoscan 指令用来产生一个初步的 configure.in 文件。

### ❖ 关键说明

必要参数的具体说明如下：

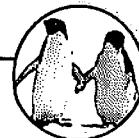
参数	功能
-v	详细报告处理信息
-d	不要删除临时文件

选择性参数的具体说明如下：

参数	功能
-h	帮助信息
-V	版本信息
--prepend/include=<目录>	设定指定目录为搜索路径
--include=<目录>	附加指定目录到搜索路径

### ❖ 执行范例

- ① 范例 712 ② 产生初步的 configure.in 文件。



[root@localhost root]# autoscan //产生初步的 configure.in 文件

此指令 autoscan 用于产生 configure.in 文件。

 **执行范例**

automake、autoreconf、autoupdate、autoheader、autoconf、config.sub、config.guess、ifnames、libtool、automount

## 10.6

## autoupdate 指令：更新 configure.in 文件

 **语法**

autoupdate[必要参数][选择性参数][模板文件]

 **功能说明**

如果给定模板文件，如 configure.in 或者 config.ac，那么 autoupdate 指令将更新 configure.in 文件，语法标准参考当前版本的 autoconf 的语法标准。autoupdate 支持对原始的、没经过编译的文件进行编译。

 **参数说明**

必要参数的具体说明如下：

参 数	功 能
-v	详细报告处理信息
-d	不要删除临时文件
-f	认为所有的文件是过期文件

选择性参数的具体说明如下：

参 数	功 能
-h	帮助信息
-V	版本信息
--prepend/include=<目录>	设定指定目录为搜索路径
--include=<目录>	附加指定目录到搜索路径

 **执行范例**

◎ 范例 713 ◎ 更新 configure.in 文件。

[root@localhost root]# autoupdate //更新configure.in 文件

## 10.7 gcc 指令：GNU 的 C 和 C++ 编译器

autoscan 指令用来产生 configure.in 文件，autoupdate 指令用来更新 configure.in 文件。



automake、autoreconf、autoupdate、autoheader、autoconf、config.sub、config.guess、ifnames、libtool、automount

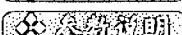
## 10.7 gcc 指令：GNU 的 C 和 C++ 编译器



gcc[必要参数][选择性参数][文件名]



gcc 指令用来编译 C 和 C++ 文件。

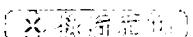


必要参数的具体说明如下：

参数	功能
-c	编译、汇编而不链接
-S	编译而不进行汇编和链接
-E	预处理
-v	同时显示编译器的代码

选择性参数的具体说明如下：

参数	功能
-o<文件>	将输出输入到指定文件
-x<语言>	指定语言
--help	帮助信息
--version	版本信息



○ 范例 714 ○ 编译文件。

```
[root@localhost littleProg]# cat file.c //显示文件内容
#include <stdio.h>
```



```

int main ()
{
 FILE *fp;
 int a, b;
 fp = fopen ("abc.mp3", "rb");
 a = ftell (fp);
 printf ("%d\n", a);
 fseek (fp, 10, 0);
 b = ftell (fp);
 printf ("%d\n", b);
 fclose (fp));
}

[root@localhost littleProg]# gcc -o file file.c //编译文件
[root@localhost littleProg]# ./file //运行
0
10
[root@localhost littleProg]#

```

本例使用指令“gcc -o file file.c”编译文件 file.c。



g++、gdb

## 10.8 gdb 指令：GNU 调试器



**【技巧】** gdb [可执行程序]



**【功能说明】** gdb 指令是 GNU 的调试工具。

### 【内部指令】

- b/break:** 设置断点。
- delete:** 消除断点。
- continue/c:** 继续终止的程序。
- run/r:** 运行程序。
- step/s:** 单步运行，进入函数。
- next/n:** 单步运行，不进入函数。
- list/l:** 显示源代码，每次显示 10 行。
- bt/backtrace:** 显示栈中的内容。

- print/p: 显示 queue 中的内容。
- kill: 终止正在调试的程序。
- file: 装入将要调试的文件。
- q: 退出 gdb。

**执行范例**

## ◎ 范例 715 ◎ 指定 gdb 进行调试。

```
[root@localhost littleProg]# cat file.c //显示文件内容
#include <stdio.h>
int main ()
{
 FILE *fp;
 int a, b;
 fp = fopen ("abc.mp3", "rb");
 a = ftell (fp);
 printf ("%08d\n", a);
 fseek (fp, 10, 0);
 b = ftell (fp);
 printf ("%08d\n", b);
 fclose (fp);}

[root@localhost littleProg]# gcc -g -o file file.c //编译文件
[root@localhost littleProg]# gdb file //进入 gdb 调试
GNU gdb Red Hat Linux (6.3.0.0-1.21rh)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.

There is absolutely no warranty for GDB. Type "show warranty" for details.

This GDB was configured as "i386-redhat-linux-gnu"...Using host
libthread_db library "/lib/libthread_db.so.1".

(gdb) l
1 #include <stdio.h>
2 int main ()
3 {
4 FILE *fp;
5 int a, b;
```



```

6 fp = fopen ("abc.mp3", "rb");
7 a = ftell (fp);
8 printf ("%d\n", a);
9 fseek (fp, 10, 0);
10 b = ftell (fp);
(gdb) b 10
Breakpoint 1 at 0x80484d2: file file.c, line 10.
(gdb) r
Starting program: /home/littleProg/file
Reading symbols from shared object read from target memory...done.
Loaded system supplied DSO at 0x5a5000
#####
Breakpoint 1, main () at file.c:10
10 b = ftell (fp);
(gdb) c
Continuing.
#####
Program exited normally.
(gdb) q
[root@localhost littleProg]#

```

本例使用指令“gdb file”进入 gdb 调试。

#### 【相关指令】

gcc

## 10.9

### gdbserver 指令：远端 GNU 服务器

**gdbserver[ tty ][ 必要参数 ][ 进程号 ]**

**gdbserver** 指令用来远端控制 **gdb** 调试。

必要参数的具体说明如下：

参 数	功 能
<b>-attend</b>	指定进程号

**◎执行范例**

◎范例 716 ◎ 远端调试程序。

[root@localhost littleProg]# gdbserver ttym --attend 4549//远端调试程序

本例使用指令“gdbserver ttym --attend 4549”远端调试程序。

**◎相关指令**

gdb

## 10.10 make 指令：编译内核或模块

**◎语法**

make[必要参数][选择性参数][文件]

**◎功能说明**

make 指令用来编译内核或者模块。

**◎参数说明**

必要参数的具体说明如下：

参 数	功 能
-d	排错模式
-e	环境变量设置
-i	忽略错误
-n	调试模式
-q	不显示调试信息
-s	执行时不显示任何信息

选择性参数的具体说明如下：

参 数	功 能
-v	版本信息
-j<工作编号>	指定工作编号
-o<文件>	指定的文件不编译
-W<文件>	指定目标文件



◆ 执行范例

- ◎ 范例 717 ◎ 编译。

```
[root@localhost root]#make
```

- ◎ 范例 718 ◎ 安装。

```
[root@localhost root]#make install
```

- ◎ 范例 719 ◎ 安装模块。

```
[root@localhost root]#make modules_install
```

以上 3 步是内核安装的一般过程。

◆ 相关指令

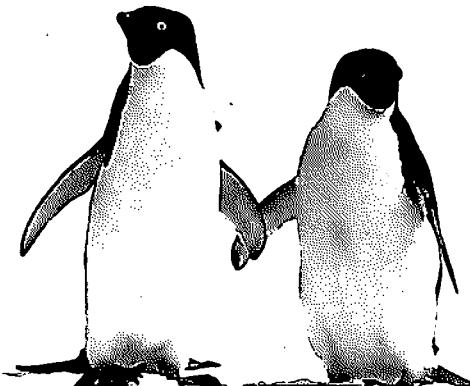
configure



## 第11章

### 打印作业

打印作业是指 Linux 系统对文本输出的控制，主要包括文本输出的设置、文本输出的编排、打印的顺序、文件的打印等。本章将详细介绍相关指令。





## 11.1 cat 指令：输出文件内容

**帮助** cat[必要参数][选择性参数][文件]

**功能说明** cat 指令用于连接文件到标准输出。

**参数说明**

必要参数的具体说明如下：

参 数	功 能
-n	对输出的文本行进行编号
-b	对输出的文本行进行编号，但是不对空白行编号
-s	将两行以上的空白行替换为一行空白行
-A	等同于 “-vET”
-e	等同于 “-vE”
-E	在每一行的结尾显示 “\$”
-t	等同于 “-vT”
-T	用 “^” 代替 “TAB” 作为显示

选择性参数的具体说明如下：

参 数	功 能
-v	显示版本信息
-h	显示帮助信息

**范例**

- 范例 720 ○ 显示文件内容到标准输出。

```
[root@localhost root]# cat abc //显示文件 abc 的内容到标准输出
```

```
i am a student
```

```
i am from bj
```

```
[root@localhost root]#
```

- 范例 721 ○ 对输出的内容进行编号。

```
[root@localhost root]# cat -n abc //显示文件abc的内容到标准输出，并对输出进行编号
 1 i am a student
 2 i am from bj
[root@localhost root]#
```

◎ 范例 722 ◎ 显示文件内容到标准输出，并在每一行的末尾以“\$”结束。

```
[root@localhost root]# cat -E abc //显示文件abc的内容到标准输出，并在每一行的
末尾以“$”结束
```

```
i am a student$
i am from bj$
[root@localhost root]#
```



vi

## 11.2 cut 指令：剪切文件

**语法** cut[必要参数][选择性参数][文件]

**功能说明** 从每个文件中选择指定的行输出到标准输出。

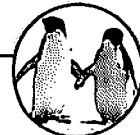
**参数说明**

必要参数的具体说明如下：

参数	功能
-n	与“-b”参数同时使用时不分割多字节字符
-only-delimited	不输出不包含定界符的行

选择性参数的具体说明如下：

参数	功能
-bytes=<LIST>	只输出指定的字节数
-characters=<LIST>	只输出指定的字符
-delimiter=<DELIM>	使用定界符代替“TAB”键
-fields=<LIST>	只输出指定的域。同时输出任何不包含定界符的行，除非指定了“-s”参数



## 第 11 章 打印作业

### ☆ 执行范例

◎ 范例 723 ◎ 显示文件的指定格式的内容。

```
[root@localhost ~]# cat a.c //显示文件的内容
kdjflkajksdklfjjjjjjjjj
sajkldfdjdlskjfklkjflksd
jdsklfjlksdjflkjsdlkfjsdlkjf
love apple

ls
ksdjflksdjlkf

ja
[root@localhost ~]# cut -c 2 a.c //显示文件每行的第二个字符
d
a
d
o
s
s
a
```

[root@localhost ~]#

本例使用指令“cut -c 2 a.c”显示文件 a.c 的每一行的第二个字符，显示结果如上所示。

◎ 范例 724 ◎ 以字节的模式显示文件的内容。

```
[root@localhost ~]# cat a.c //显示文件的内容
kdjflkajksdklfjjjjjjjjj
sajkldfdjdlskjfklkjflksd
jdsklfjlksdjflkjsdlkfjsdlkjf
love apple

ls
ksdjflksdjlkf

ja
[root@localhost ~]# cut -b 1 a.c //显示每行的第一个字节
k
s
j
l
l
k
j
```

```
[root@localhost ~]# cat -b2 a.c
d
a
d
o
s
s
a
```

//显示每一行的第二个字节

[root@localhost ~]#

本例使用指令“cut -b1 a.c”显示文件 a.c 的每一行第一个字节的内容，并使用指令“cut -b2 a.c”显示文件 a.c 的每一行第二个字节的内容。

### 相关命令

paste

## 11.3 pr 指令：编排文件格式

**功能说明** pr[必要参数][选择性参数][文件]

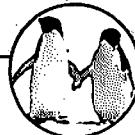
**功能说明** pr 指令对指定的文件进行编排，然后将结果输出到标准输出。

### 参数说明

必要参数的具体说明如下：

参数	功能
-a	直栏变成竖栏
-b	平衡栏位
-c	使用“^”来表示控制字符
-d	两行之间加入空白行
-m	一次打印多个文件
-t	不显示页首和页尾

选择性参数的具体说明如下：



参数	功能
-l<行数>	每页显示的行数
-o<偏移量>	设置偏移量
-help	帮助信息
-version	版本信息
-<栏数>	每页显示的栏数

### ◎ 执行范例

#### ◎ 范例 725 ◎ 编排文件。

```
[root@localhost ~]# cat 123 //显示文件 123 的内容
000000 00000000
111111 11111111
222222 22222222
333333 33333333
444444 44444444
555555 55555555
666666 66666666
777777 77777777
888888 88888888
999999 99999999
[root@localhost ~]# pr 123 > 456 //编排文件 123
[root@localhost ~]# cat 456 //显示文件 456 的内容
```

2007-05-27 17:18

123

Page 1

```
000000 00000000
111111 11111111
222222 22222222
333333 33333333
444444 44444444
555555 55555555
666666 66666666
777777 77777777
888888 88888888
999999 99999999
[root@localhost ~]#
```

本例使用指令“pr 123 > 456”编排文件 123，将结果重定向到文件 456；并使用指令 cat 456 显示文件 456 的内容，显示的结果如上所示。

◎ 范例 726 ◎ 设置文件为 3 栏。

```
[root@localhost ~]# cat abc //显示文件内容
a
ab
abc
abcd
abcde
abcdef
abcdefg
abcdefgh
abcdefghi
abcdefhij
[root@localhost ~]# pr -3 abc>456 //编排文件
[root@localhost ~]# cat 456 //显示文件内容
```

2007-05-27 09:01	abc	Page 1
------------------	-----	--------

a	abcde	abcdefg
ab	abcdef	abcdefghi
abc	abcdefg	abcdefghi
abcd		

[root@localhost ~]#

本例使用指令“pr -3 abc>456”编排文件 abc，分 3 栏显示文件内容，并将结果重定向到文件 456；本例还使用指令 cat 456，显示文件的内容，如上所示。



cat

## 11.4 tac 指令：反序输出文件

tac[必要参数][选择性参数][文件]

tac 指令以反序的方式显示文件的内容。



## 第 11 章 打印作业

### ◆ 参数说明

必要参数的具体说明如下：

参 数	功 能
-b	把间隔字符放在前面
-t	视间隔字符为普通字符

选择性参数的具体说明如下：

参 数	功 能
-s<字符>	指定间隔字符
-help	帮助信息
-version	版本信息

### ◆ 执行范例

#### ● 范例 727 ● 反序显示文件。

```
[root@localhost ~]# cat file1 //显示文件内容
this is line 1...
this is line 2...
this is line 3...
this is line 4...
this is line 5...
this is line 6...
this is line 7...
[root@localhost ~]# tac file1 //反序显示文件内容
this is line 7...
this is line 6...
this is line 5...
this is line 4...
this is line 3...
this is line 2...
this is line 1...
[root@localhost ~]#
```

本例使用指令“tac file1”反序显示文件内容。

### ◆ 相关命令

cat

## 11.5 tail 指令：显示文件的末尾内容

 **语法** tail[必要参数][选择性参数][文件]

 **功能说明** tail 指令用于读取文件的内容，将指定的文件的末尾部分输出到标准输出。不指定文件时，将把标准输入作为输入信息进行处理。

 **参数说明**

必要参数的具体说明如下：

参 数	功 能
-f	循环读取
-q	不显示处理信息
-v	运行时显示详细的处理信息

选择性参数的具体说明如下：

参 数	功 能
-c<数目>	要显示的字节数
-n<行数>	显示的行数
--help	帮助信息
--version	版本信息

 **执行范例**

- 范例 728 ○ 显示文件的末尾内容。

```
[root@localhost ~]# tail -n 30 install.log //显示文件的末尾内容
GFS-kernel-2.6.11.8-20050601.152643.FC4.2.i586.rpm
GFS-kernel-smp-2.6.11.8-20050601.152643.FC4.2.i686.rpm
GFS-kernel-xen0-2.6.11.8-20050601.152643.FC4.2.i686.rpm
GFS-kernel-xenU-2.6.11.8-20050601.152643.FC4.2.i686.rpm
GFS-kernheaders-2.6.11.8-20050601.152643.FC4.2.i586.rpm
```



```
cman-kernel-2.6.11.5-20050601.152643.FC4.2.i586.rpm
cman-kernel-smp-2.6.11.5-20050601.152643.FC4.2.i686.rpm
cman-kernel-xen0-2.6.11.5-20050601.152643.FC4.2.i686.rpm
cman-kernel-xenU-2.6.11.5-20050601.152643.FC4.2.i686.rpm
cman-kernheaders-2.6.11.5-20050601.152643.FC4.2.i586.rpm
dlm-kernel-2.6.11.5-20050601.152643.FC4.2.i586.rpm
dlm-kernel-smp-2.6.11.5-20050601.152643.FC4.2.i686.rpm
dlm-kernel-xen0-2.6.11.5-20050601.152643.FC4.2.i686.rpm
dlm-kernel-xenU-2.6.11.5-20050601.152643.FC4.2.i686.rpm
dlm-kernheaders-2.6.11.5-20050601.152643.FC4.2.i586.rpm
glibc-2.3.5-10.i386.rpm
gnbd-kernel-2.6.11.2-20050420.133124.FC4.35.i586.rpm
gnbd-kernel-smp-2.6.11.2-20050420.133124.FC4.35.i686.rpm
gnbd-kernel-xen0-2.6.11.2-20050420.133124.FC4.35.i686.rpm
gnbd-kernel-xenU-2.6.11.2-20050420.133124.FC4.35.i686.rpm
gnbd-kernheaders-2.6.11.2-20050420.133124.FC4.35.i586.rpm
kernel-2.6.11-1.1369_FC4.i586.rpm
kernel-devel-2.6.11-1.1369_FC4.i586.rpm
kernel-smp-2.6.11-1.1369_FC4.i686.rpm
kernel-smp-devel-2.6.11-1.1369_FC4.i686.rpm
kernel-xen0-2.6.11-1.1369_FC4.i686.rpm
kernel-xen0-devel-2.6.11-1.1369_FC4.i686.rpm
kernel-xenU-2.6.11-1.1369_FC4.i686.rpm
kernel-xenU-devel-2.6.11-1.1369_FC4.i686.rpm
openssl-0.9.7f-7.i386.rpm
[root@localhost ~]#
```

本例使用指令“tail -n 30 install.log”显示文件 install.log 的末尾 30 行内容。

- 范例 729 ○ 显示更新的 log 文件的末尾内容。

```
t@localhost:~# tcpdump >&2.log //监测tcp流量
```

本例使用指令“tcpdump >&2.log”检测网络 tcp 流量，并将结果重定向到文件 2.log。这个指令运行在一个单独的终端上。

同时，可以在另一个终端下显示文件内容。

```
root@localhost:~# tail -f 2.log //显示文件内容
17:10:42.886082 IP 192.168.88.46.1413 > 202.112.150.121.www: . ack 97401 win 17520
```

## 11.5 tail 指令：显示文件的末尾内容

```

17:10:42.887368 IP 192.168.88.46.1413 > 202.112.150.121.www: . ack 98861 win 17520
17:10:42.923042 IP 192.168.88.46.1408 > 202.112.150.121.www: P
3442022626:3442023397(771) ack 1727122249 win 17520
17:10:42.941150 IP 202.112.150.121.www > 192.168.88.46.1408: P1:165(164)
ack 771 win 65535
17:10:42.959328 IP 192.168.88.46.1410 > 202.112.150.121.www: P1:788(787)
ack 1376 win 16144
//此处省略了部分结果
17:10:43.514885 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
26348:27808(1460) ack 1 win 5840
17:10:43.516126 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
27808:29268(1460) ack 1 win 5840
17:10:43.516383 IP 192.168.88.7.1683 > 192.168.88.2.telnet: . ack 24888 win 17520
17:10:43.517426 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
29268:30728(1460) ack 1 win 5840
17:10:43.518664 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
30728:32188(1460) ack 1 win 5840
17:10:43.518919 IP 192.168.88.7.1683 > 192.168.88.2.telnet: . ack 27808 win 17520
17:10:43.519968 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
32188:33648(1460) ack 1 win 5840
17:10:43.521206 IP 192.168.88.2.telnet > 192.168.88.7.1683: .
33648:35108(1460) ack 1 win 5840
17:10:43.521461 IP 192.168.88.7.

```

本例使用指令“tail -f 2.log”显示文件 2.log 的末尾的内容。因为文件 2.log 是始终更新的，所以通过参数“-f”的指定，tail 指令会不断读取文件 2.log 的内容，不停地将更新的文件的末尾的内容发送到标准输出。这在程序调试中将非常有用。

○ 范例 730 ○ 显示多个文件的末尾内容。

```

root@localhost:~# tail -n.5 *.log //显示文件末尾内容
==> 1.log <==
17:01:27.241870 IP 202.112.146.68.1539 > iplab604.telnet: P
54717045:54717046(1) ack 2552300598 win 65335
1730 packets captured
2797 packets received by filter

```



```

1067 packets dropped by kernel
=> 2.log <=
17:12:22.600449 IP 192.168.88.7.1656 > 192.168.88.2.ssh: P 33792
69317:3379269369(52) ack 2677970800 win 16752
6064 packets captured
6226 packets received by filter
162 packets dropped by kernel
=> 7.log <=
17:19:08.600064 IP 192.168.88.2.ssh > 192.168.88.7.1656: P 3993: 4045(52)
ack 2860 win 8576
17:19:08.620153 arp who-has 211.71.75.1 tell 211.71.75.2
17:19:08.702110 IP 202.112.147.192.netbios-ns > 202.112.147.
255.netbios-ns: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST
17:19:08.709861 IPX 00123456.00:0a:e6:db:d2:fb.4000 > 00000000.
ff:ff:ff:ff:ff:ff.0452: ipx-sap-nearest-req FileServer
17:19:08.710769 IP 192.168.88.2.telnet > 192.168.88. root@loca lhost:-#
```

本例使用指令“tail -n 5 \*.log”显示多个文件 1.log、2.log 和 7.log 的末尾的内容。

### 相关指令

head

## 11.6 zcat 指令：显示压缩文件的内容

### 基本用法

zcat[必要参数][文件]

功能说明 zcat 指令用于查看压缩文件的信息，并将压缩文件的信息输出到标准输出设备。

### 参数说明

必要参数的具体说明如下：

参 数	功 能
-f	强制执行

选择性参数的具体说明如下：

## 11.6 zcat 指令：显示压缩文件的内容

参数	功能
-help	帮助信息
-license	版权信息
-version	版本信息

 执行范例

## ◎ 范例 731 ◎ 显示压缩文件的内容。

```
[root@localhost temp]# zcat /usr/share/info/uucp.info-8.gz //显示压缩文件内容
This is uucp.info, produced by makeinfo version 4.1 from uucp.texi.
```

## START-INFO-DIR-ENTRY

\* UUCP: (uucp). Transfer mail and news across phone lines.

## END-INFO-DIR-ENTRY

This file documents Taylor UUCP, version 1.07.

Copyright (C) 1992, 1993, 1994, 1995, 2002 Ian Lance Taylor

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided also that the section entitled "Copying" are included exactly as in the

//此处省略了部分结果

* seven-bit in port file:	port File.
* speed in port file:	port File.
* speed in sys file:	Placing the Call.
* speed-range:	port File.
* spool:	Miscellaneous (config).
* statfile:	Log File Names.
* stream:	port File.
* strip-login:	Miscellaneous (config).
* strip-proto:	Miscellaneous (config).
* sysfile:	Configuration File Names.
* system:	Naming the System.
* time:	When to Call.
* timegrade:	When to Call.
* timetable:	Miscellaneous (config).
* transfer:	File Transfer Control.
* type:	port File.



## 第 11 章 打印作业

- \* unknown:
- \* username:
- \* v2-files:
- \* version:

[root@localhost temp]#

本例使用指令“zcat /usr/share/info/uucp.info-8.gz”显示/usr/share/info/uucp.info-8.gz 的内容，由于显示内容较多，中间部分已省略，不影响本例阅读。



zip

Miscellaneous (config).

Miscellaneous (config).

Miscellaneous (config).

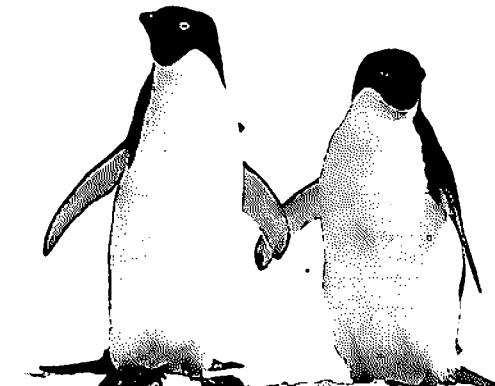
port File.



## 第12章

### 电子邮件新闻组

电子邮件新闻组主要包括电子邮件服务、新闻服务器的设置与新闻的获取服务。具体包括电子邮件的收取和发送，电子邮件的查询，新闻服务器的设置，新闻服务的获取等。本章将详细介绍相关的指令。





## 12.1

### fetchmail 指令：获得邮件

#### ● 语法

`fetchmail[必要参数][选择性参数][邮件服务器]`

#### ● 功能说明

fetchmail 指令用于邮件收取和发送，它从远端邮件服务器获取邮件并将其发送到本地客户端。用户可以用通常的邮件用户代理如 mutt、elm 或者 mail 来处理收到的邮件，也可以通过设置使得 fetchmail 指令周期性地收取邮件。

#### ● 参数说明

必要参数的具体说明如下：

参    数	功    能
<code>-a</code>	接收所有的邮件
<code>-envelope</code>	复制邮件的源地址
<code>-flush</code>	删除旧的邮件之后再收取新的邮件
<code>-k</code>	邮件服务器上保留邮件备份
<code>-K</code>	邮件服务器上不保留邮件备份
<code>-m</code>	启动 mda 模式
<code>-N</code>	在前台运行 fetchmail 守护进程
<code>-n</code>	不更改文件头
<code>-q</code>	关闭 fetchmail 程序
<code>-s</code>	程序运行时不报告错误信息
<code>-U</code>	启动 UIDL
<code>-v</code>	程序运行时显示详细的处理过程
<code>-w</code>	当邮件大小过大时显示警告信息
<code>-bsmtp</code>	增加 SMTP 附件
<code>-configdump</code>	读取.fetchmailrc 配置文件
<code>-invisibl</code>	隐藏 fetchmail 指令
<code>-lmtp</code>	支持 lmtp 协议

## 12.1 fetchmail 指令：获得邮件

续表

参数	功能
-nobounce	返回邮件错误信息至邮件管理者
-nosyslog	不记录 log 信息
-postmaster	邮件如果找不到收件者，将会寄给邮件管理者
-syslog	记录 log 信息
-c	检查邮件状态而不收取

选择性参数的具体说明如下：

参数	功能
-auth<认证类型>	指定认证类型
-b<邮件数目>	指定邮件数目
-d<时间>	指定 fetchmail 定期收取邮件的周期
-D<域名>	指定域名
-f<配置文件>	指定配置文件
-i<文件>	指定用于保存用户识别码的文件
-l<邮件大小>	指定邮件的大小限制
-L<文件名>	指定保存 log 信息的文件名
-p<传输协议>	指定传输协议
-P<端口号>	指定与邮件服务器连接的端口号
-Q<字符串>	删除指定的字符串
-r<邮件目录>	指定接收邮件的目录
-s<时间>	设置超时时间
-u<账号>	登录邮件服务器的账号
-plugin<外挂>	指定外挂
-help	帮助信息
-version	版本信息



- ◎ 范例 732 ◎ 从邮件服务器下载邮件。



## 第 12 章 电子邮件新闻组

```
[root@localhost ~]# fetchmail -p POP3 -u 04515674 mail.njtu.edu.cn
//下载邮件
Enter password for 04515674@mail.njtu.edu.cn:
5 messages for 04515674 at mail.njtu.edu.cn (1063170 octets).
reading message 04515674@mail.njtu.edu.cn:1 of 5 (2001 octets) .. flushed
reading message 04515674@mail.njtu.edu.cn:2 of 5 (1666 octets) .. flushed
reading message 04515674@mail.njtu.edu.cn:3 of 5 (1899 octets) .. flushed
reading message 04515674@mail.njtu.edu.cn:4 of 5 (1899 octets) .. flushed
reading message 04515674@mail.njtu.edu.cn:5 of 5 (1055705
octets)
.....
```

本例使用指令“`fetchmail -p POP3 -u 04515674 mail.njtu.edu.cn`”下载邮件。通过参数“`-p POP3`”指定传输协议为 POP3，通过参数“`-u 04515674`”指定用户名为 04515674，最后一个参数为指定的邮件服务器。

- 范例 733 ○ 收取所有的邮件，并在邮件服务器上保留原邮件。

```
[root@localhost ~]# fetchmail -ak -p POP3 -u 06145891 mail.bjtu.edu.cn
//下载邮件
Enter password for 06145891@mail.bjtu.edu.cn:
183 messages for 06145891 at mail.bjtu.edu.cn (30069496 octets).
reading message 06145891@mail.bjtu.edu.cn:1 of 183 (2268 octets) .. not flushed
reading message 06145891@mail.bjtu.edu.cn:2 of 183 (2949 octets) .. not flushed
reading message 06145891@mail.bjtu.edu.cn:3 of 183 (12573
octets) not flushed
reading message 06145891@mail.bjtu.edu.cn:4 of 183 (7550 octets)
not flushed
reading message 06145891@mail.bjtu.edu.cn:5 of 183 (25220
```

## 12.2 getlist 指令：下载新闻

```

octets) not flushed
reading message 06145891@mail.bjtu.edu.cn:6 of 183 (9214 octets)
not flushed

reading message 06145891@mail.bjtu.edu.cn:7 of 183 (3118 octets) ... not flushed
reading message 06145891@mail.bjtu.edu.cn:8 of 183 (2889 octets) .. not flushed
reading message 06145891@mail.bjtu.edu.cn:9 of 183 (76784
octets) not flushed
reading message 06145891@mail.bjtu.edu.cn:10 of 183 (1350 octets) . not flushed
reading message 06145891@mail.bjtu.edu.cn:11 of 183 (1827 octets) . not flushed
reading message 06145891@mail.bjtu.edu.cn:12 of 183 (215465
octets) not flushed
..... //此处省略了部分结果
reading message 06145891@mail.bjtu.edu.cn:97 of 183 (1786 octets) . not flushed
reading message 06145891@mail.bjtu.edu.cn:98 of 183 (6568
octets) . fetchmail: terminated with signal 2

```

You have new mail in /var/spool/mail/root

[root@localhost ~]#

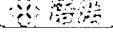
本例使用指令“fetchmail -ak -p POP3 -u 06145891 mail.bjtu.edu.cn”以用户 06145891 登录到邮件服务器 mail.bjtu.edu.cn 下载邮件，通过参数“-ak”的设定收取所有的邮件，并保存邮件在邮件服务器上的备份。

### ◆ 相关指令

mutt、elm、mail、sendmail、popd、imapd、netrc

## 12.2

### getlist 指令：下载新闻

 **getlist[选择性参数]**

 **getlist** 指令从新闻服务器上下载新闻列表，并将结果输出到标准输出。



选择性参数的具体说明如下：



参数	功能
-h<主机>	指定服务器
-p<端口号>	指定端口号

### ❖ 执行范例

#### ● 范例 734 ● 下载新闻列表。

[root@localhost littleProg]# getlist -h news.jinghua.net //下载新闻列表

本例使用指令“getlist -h news.jinghua.net”从主机 news.jinghua.net 上  
下载新闻列表。

### ❖ 相关指令

getopt

## 12.3

## mail 指令：收发邮件

### ❖ 语法

mail[必要参数][选择性参数][邮箱地址]

❖ 功能说明 mail 指令是一个邮件管理工具，它在命令行下工作。

### ❖ 重要说明

必要参数的具体说明如下：

参数	功能
-i	不输出处理信息
-l	交互模式
-n	不使用配置文件中的配置
-N	不显示邮件标题
-v	运行时显示详细的处理信息

选择性参数的具体说明如下：

参数	功能
-b<邮箱地址>	指定邮箱地址
-c<邮箱地址>	指定邮箱副本地址

续表

参    数	功    能
-s<邮件主题>	指定邮件主题
-u<用户>	指定用户
-f<邮件>	读取指定邮件

**【进入 mail 后的指令】**

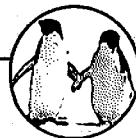
- 数字键：选择邮件。
- +：上一封。
- ：下一封。
- h：显示标题。
- q：退出 mail 程序，退出前删除标记的邮件。
- x：退出 mail 程序。
- ?: 帮助信息。
- p<邮件号>：读取指定邮件号的邮件。
- d<邮件号>：删除邮件。
- u<邮件号>：恢复邮件。
- R<邮件号>：回复邮件。
- r<邮件号>：回复邮件，既寄给发件人，也寄给其他收件人。
- m<邮箱地址>：发邮件到指定邮箱。

**(☆) 实行范例****○ 范例 735 ○ 发送邮件。**

```
[root@localhost ~]# mail 02211025@bjtu.edu.cn //发送邮件
Subject: life //输入邮件主题
life is like a box of chocolate,before you eat it,you never know what you will get
//输入邮件内容
Cc: 06111021@bjtu.edu.cn //输入副本的发送
```

本例使用指令“mail 02211025@bjtu.edu.cn”将邮件发送到 02211025@bjtu.edu.cn，发送邮件的内容为“life is like a box of chocolate,before you eat it,you never know what you will get”，副本发送到 06111021@bjtu.edu.cn。

**○ 范例 736 ○ 查询当前用户的邮件信息。**



## 第 12 章 电子邮件新闻组

```
[root@localhost ~]# mail
Mail version 8.1 6/6/93. Type ? for help.
"/var/spool/mail/root": 472 messages 472 unread
>U 1 root@localhost.local Fri Dec 22 17:28 38/1405 "LogWatch for
localhost.localdomain"
 U 2 root@localhost.local Sat Dec 30 13:07 38/1405 "LogWatch for
localhost.localdomain"
 U 3 root@localhost.local Sun Dec 31 14:44 124/3411 "LogWatch for
localhost.localdomain"
```

//当前用户的邮件信息

```
.....
```

//此处省略了部分结果

```
U 42 root@localhost.local Mon Apr 30 20:31 25/1040 "flip flop"
```

```
&
```

```
Message 1:
```

```
From root@localhost.localdomain Fri Dec 22 17:28:45 2006
Date: Fri, 22 Dec 2006 17:28:45 +0800
From: root <root@localhost.localdomain>
To: root@localhost.localdomain
Subject: LogWatch for localhost.localdomain
```

```
LogWatch 6.0.1 (02/24/05)

```

```
Processing Initiated: Fri Dec 22 17:28:45 2006
```

```
Date Range Processed: yesterday
```

```
Detail Level of Output: 10
```

```
Type of Output: unformatted
```

```
Logfiles for Host: localhost.localdomain
```

```
#####
----- Disk Space Begin -----
```

/dev/sda3	9.2G	6.9G	1.8G	80%	/
/dev/sda1	99M	12M	83M	12%	/boot
/dev/shm	252M	0	252M	0%	/dev/shm

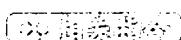
```
----- Disk Space End -----
```

```
#####
LogWatch End #####

```

```
&
```

本例使用指令 mail 查询当前用户的邮件信息。

 不用装此命令

fetchmail

## 12.4 mailq 指令：显示发件箱的邮件



**mailq[必要参数]**



mailq 指令用来显示发件箱的邮件。



必要参数的具体说明如下：

参数	功能
-v	运行时显示详细的处理过程



◎ 范例 737 ◎ 显示待发送的邮件。

```
root@localhost:~# mailq -v //显示待发送的邮件
```

```
150d 867 1HAvys-0005CP-Ti <root@localhost>
 root@localhost
```

```
143d 867 1HDTJi-0001Uc-S1 <root@localhost>
 root@localhost
```

```
120d 867 1HLLoHc-0001DN-F6 <root@localhost>
 root@localhost
```

```
118d 867 1HMXEQ-0001Yp-Pn <root@localhost>
 root@localhost
```

```
80d 867 1HaJLK-0000KZ-Jp <root@localhost>
 root@localhost
```

```
59d 867 1HhvLo-0007TR-FQ <root@localhost>
 root@localhost
```

```
45d 867 1Hn01V-0003iz-Nc <root@localhost>
```



```
root@localhost
24d 867 1Huc20-0002Uo-40 <root@localhost>
root@localhost
17d 867 1Hx9Mr-0004mo-94 <root@localhost>
root@localhost
```

root@localhost:~#

本例使用指令“mailq -v”显示待发送的邮件。

#### 相关指令

mail

## 12.5 mutt 指令：e-mail 管理

### 语法

mutt

 功能说明 mutt 指令是一个使用文字模式的邮件管理器。

### 使用范例

◎ 范例 738 ◎ 运行 mutt。

```
root@localhost:~# mutt //执行mutt
q:Quit d:Del u:Undel s:Save m:Mail r:Reply g:Group ?:Help
```

---Mutt: (no mailbox) [Msgs:0]---(threads/ date)-----

--(all)--

运行指令 mutt，进入以上界面。

◎ 范例 739 ◎ 显示帮助信息。在【范例】的界面中输入“？”，显示帮助信息。

```
i:Exit -:PrevPg <Space>:NextPg ?:Help
^B M <pipe-message> ur... call urlview to extract URLs out of a message
^D delete-thread delete all messages in thread
^E edit-type edit attachment content type
^F forget-passphrase wipe passphrase(s) from memory
```

## 12.6 nntpget 指令：从新闻服务器下载文章

<Tab>	next-new-then-unread	jump to the next new or unread message
<Return>	display-message	display a message
^K	extract-keys	extract supported public keys
^N	next-thread	jump to the next thread
^P	previous-thread	jump to previous thread
^R	read-thread	mark the current thread as read
^T	untag-pattern	untag messages matching a pattern
^U	undelete-thread	undelete all messages in thread
<Esc><Tab>	previous-new-then-unre	jump to the previous new or unread message
<Esc>C	decode-copy	make decoded (text/plain) copy
<Esc>P	check-traditional-pgp	check for classic pgp
<Esc>V	collapse-all	collapse/uncollapse all threads
<Esc>b	M <search>~b	search in message bodies
<Esc>c	change-folder-readonly	open a different folder in read only mode
<Esc>d	delete-subthread	delete all messages in subthread
<Esc>e	resend-message	use the current message as a template for a new one
+		

Help for index

-- (158)



mail

## 12.6

## nntpget 指令：从新闻服务器下载文章



nntpget[必要参数][选择性参数][nntp 服务器]



nntpget 指令从指定新闻服务器下载文章。



必要参数的具体说明如下：

参数	功能
-o	只下载本机没有的文章
-v	运行时显示详细的处理信息



## 第 12 章 电子邮件新闻组

选择性参数的具体说明如下：

参 数	功 能
-d<类型>	指定新闻组类型
-f<文件>	下载比指定文件时间新的文章
-t<时间>	比指定时间新的文章将被下载



ncftpget

## 12.7 pine 指令：收发邮件



pine[必要参数][选择性参数][邮件地址]



pine 指令用来收发邮件，它具有互动式界面，具有导向性，操作比较方便，同时可以给多人发送邮件。



必要参数的具体说明如下：

参 数	功 能
-a	匿名方式
-conf	产生配置文件，并输出到标准输出
-i	进入收件箱
-o	只读模式
-r	展示模式
-z	可进入后台作业

选择性参数的具体说明如下：

参 数	功 能
-attach<文件>	粘贴附件
-h	帮助信息

续表

参数	功能
-P<文件>	指定配置文件
-sort<排序法>	指定收件箱中文件的排序方式
-v	版本信息

**◎执行范例**

- ◎ 范例 740 ◎ 发送邮件。

```
root@localhost:~#pine 02211025@bjtu.edu.cn //发送邮件
```

本例使用指令“pine 02211025@bjtu.edu.cn”实现向 02211025@bjtu.edu.cn”发送邮件。

**◎相关指令**

mail

## 12.8 slrn 指令：新闻阅读程序

**◎概述** slrn[必要参数][选择性参数]

**◎功能说明** slrn 指令用来阅读和管理新闻组。

**◎参数说明**

必要参数的具体说明如下：

参数	功能
-c	彩色模式
-d	附带说明
-n	不检查新闻的新旧
-p	设置端口号

选择性参数的具体说明如下：

参数	功能
-h<服务器>	指定服务器
-f<文件>	存储新闻的文件



续表

参数	功能
-help	帮助信息
-version	版本信息

### ❖ 执行范例

- ① 范例 741 ② 连接新闻组服务器。

```
[root@localhost root]# slrn -h news.usc.net -f /root/.jnewsrc //
连接新闻组服务器
```

本例使用指令“slrn -h news.usc.net -f /root/.jnewsrc”连接新闻组服务器。

### ❖ 相关指令

nntpget

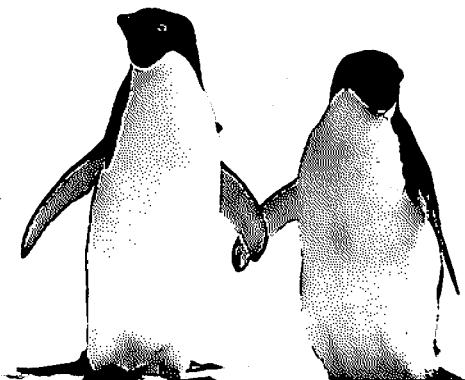
# LINUX

网  
上  
学  
习

## 第13章

### X Windows System

X Windows System 是 Linux 系统的图形界面系统，提供类似于 Windows 的图形界面。其相关的指令提供了各种常见操作功能，包括图形界面的启动、图形界面的设置等。





## 13.1 startx 指令：启动图形界面

**语法** startx

**功能说明** startx 指令用来启动图形界面

**执行范例**

◎ 范例 742 ◎ 启动图形界面。

```
[root@localhost ~]# startx //启动X Windows
```

```
Fatal server error:
```

```
Server is already active for display 0
```

```
If this server is no longer running, remove /tmp/.X0-lock
and start again.
```

```
Please consult the The X.Org Foundation support
```

```
at http://wiki.X.Org
```

```
for help.
```

```
Xlib: connection to ":0.0" refused by server
```

```
Xlib: Invalid MIT-MAGIC-COOKIE-1 key
```

```
giving up.
```

```
xinit: unable to connect to X server
```

```
xinit: No such process (errno 3): unexpected signal 2.
```

```
[root@localhost ~]#
```

本例使用指令 startx 启动 X Windows，因为是远程登录，所以会出现以上的错误，正常时将启动 X Windows。

**语法大全**

xinit

## 13.2 xset 指令：设置 X windows

**语法** xset[必要参数][选择性参数]

**功能说明** xset 指令用来设置 X windows 的各项参数。

## 13.2 xset 指令：设置 X windows

**(☆) 参数说明**

必要参数的具体说明如下：

参 数	功 能
c	关闭键盘声音
q	显示现在的设置
s	屏幕保护设置
r	按键信号设置

选择性参数的具体说明如下：

参 数	功 能
dpm<状态>	设置电源模式
-display<设备>	指定设备
led on/off	显示灯设置

**(☆) 软件范例**

○ 范例 743 ○ 显示当前设置。

```
[root@localhost root]# xset //显示当前设置
usage: xset [-display host:dpw] option ...
To turn bell off:
 -b b off b 0
To set bell volume, pitch and duration:
 b [vol [pitch [dur]]] b on
To disable bug compatibility mode:
 -bc
To enable bug compatibility mode:
 bc
To turn keyclick off:
 -c c off c 0
To set keyclick volume:
 c [0-100] c on
To control Energy Star (DPMS) features:
 -dpms Energy Star features off
 +dpms Energy Star features on
```



## 第 13 章 X Windows System

```
dpms [standby [suspend [off]]]
 force standby
 force suspend
 force off
 force on
 (also implicitly enables DPMS features)
 a timeout value of zero disables the mode
 //此处省略了部分结果

 s expose s noexpose
 s activate s reset

 For status information: q
[root@localhost root]#
```

本例使用指令 xset 显示当前系统下 X Windows 的设置情况。



X、xserver

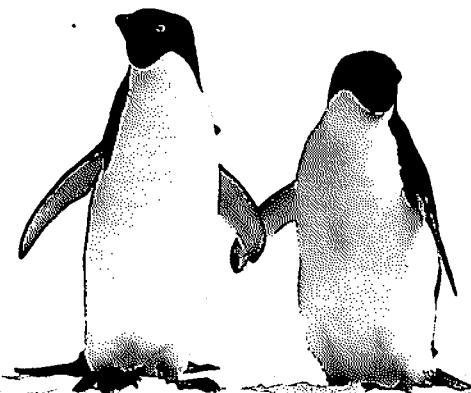
# Linux

网

## 第14章

### 格式转换

格式转换是指 Linux 系统不同文件类型的转换，主要包括 DVI 文件和 Postscript 文件之间的转换，fiasco 镜像文件和 pgm 或 ppm 格式文件之间的转换，find 格式文件和 perl 格式文件间的转换，gif 格式文件和 png 格式文件间的转换，字符集的转换等。本章将详细介绍格式转换的指令，熟练掌握这些指令可以很方便地实现各种转换。





## 14.1 dvips 指令：将 DVI 文件转换为 Postscript 文件

 **语法** dvips[必要参数][选择性参数][源文件][目标文件]

 **功能说明** dvips 指令用于将 DVI 文件转换为 Postscript 文件。

 **参数说明**

必要参数的具体说明如下：

参 数	功 能
-a	内存设置
-A	只打印奇数页
-B	只打印事件页
-f	作为一个过滤器运行
-F	引发一个“ctrl-D”操作
-i	使得每一部分作为一个单独的文件
-j	从类型 I 字体下线需要的字体
-k	打印 crop 标志
-m	指定打印机的手册指南
-M	自动产生字体工具
-q	不输出调试信息
-r	页数按相反的顺序排列
-R	以安全模式运行

选择性参数的具体说明如下：

参 数	功 能
-b<数目>	只产生每一页的指定数目的备份
-c<数目>	只产生指定数目的每一页的备份，默认是 1
-C<数目>	产生指定数目的备份

## 14.2 fiascotopnm 指令：将压缩的 fiasco 镜像文件转换为 pgm 或 ppm 格式

续表

参数	功能
-d<数目>	设置调试标签
-h<名称>	指定文件名作为附加的头文件名称
-m<模式>	指定模式
-n<数目>	指定打印的最大页数
-o<文件名>	输出文件名
-version	版本信息
-help	帮助信息

## 执行范例

◎ 范例 744 ◎ 转换 DVI 文件为 Postscript 文件。

```
[root@localhost other]# dvips -o abc.ps abc.dvi //转换文件为 Postscript 文件
本例使用指令“dvips -o abc.ps abc.dvi”转换 abc.dvi 文件为 abc.ps
文件。
```

## 相关指令

mf、afm2tfm、tex、latex、lpr

## 14.2

fiascotopnm 指令：将压缩的 fiasco  
镜像文件转换为 pgm 或 ppm 格式

语句说明 fiascotopnm[必要参数][选择性参数][文件]

功能说明 fiascotopnm 指令用来转换 fiasco 文件为 pgm 或者  
ppm 格式文件。

## 语句说明

必要参数的具体说明如下：

参数	功能
-d	快速转换

选择性参数的具体说明如下：



参数	功能
-help	帮助信息
-version	版本信息

### ❖ 执行范例

- 范例 745 ● 转换文件格式。

```
[root@localhost ~]#fiascotopnm foo.wfa >foo.ppm //转换文件
```

本例使用指令“fiascotopnm foo.wfa >foo.ppm”转换文件类型。

### ❖ 相关指令

pnmm

## 14.3

### find2perl 指令：将 find 指令行转换为 perl 代码

#### ❖ 命令

find2perl[路径][predicates]]|perl

#### ❖ 功能说明

find2perl 指令用来将 find 指令行转换为 perl 代码。

#### ❖ 相关指令

find

## 14.4

### gemtopbm 指令：转换图形文件

#### ❖ 命令

gemtopbm[必要参数][文件]

#### ❖ 功能说明

gemtopbm 指令用来将 gem 文件转换为 pbm 文件。

#### ❖ 举一反三

必要参数的具体说明如下：

参数	功能
-d	调试模式

### ❖ 执行范例

- 范例 746 ● 转换图形文件。

[root@localhost littleProg]#gemtopbm abc.img >abc.pbm //转换文件

本例使用指令“gemtopbm abc.img >abc.pbm”将.img 文件转换为.pbm 文件。

### ◆ 相关指令

gemtopnm

## 14.5

## giftopnm 指令：文件转换

### ◆ 基础

giftopnm[必要参数][选择性参数][文件]

### ◆ 功能说明

使用 giftopnm 指令将 GIF 文件转换为 PNM 文件。

### ◆ 参数说明

必要参数的具体说明如下：

参 数	功 能
-v	运行时显示详细的处理信息
-comments	显示注释

选择性参数的具体说明如下：

参 数	功 能
-image<页数>	指定页数

### ◆ 执行范例

- 范例 747 ○ 转换 gif 文件。

```
[root@localhost littleProg]# giftopnm /var/www/manual/mod/mod_python/icons/up.gif >111.pnm //转换文件
```

```
[root@localhost littleProg]#
```

本例使用指令“giftopnm /var/www/manual/mod/mod\_python/icons/up.gif >111.pnm”将文件 /var/www/manual/mod/mod\_python/icons/up.gif 转换为 111.pnm 文件。

### ◆ 相关指令

gemtopbm



大家网

## 14.6 iconv 指令：字符集转换

**语法** iconv[必要参数][选择性参数][文件]

**功能说明** iconv 指令可以用于相关字符集的转换。

**参数说明**

必要参数的具体说明如下：

参 数	功 能
-l	列出已知的所有字符集
-c	从输出中忽略无效的字符
-s	不显示警告信息

选择性参数的具体说明如下：

参 数	功 能
-f<名字>	原始文件编码
-t<名字>	输出编码
-o<文件>	输出文件
--help	帮助信息
--version	版本信息

**执行范例**

◎ 范例 748 ◎ 列出已知的编码字符集合。

```
[root@localhost ~]# iconv -l //显示支持的格式
```

```
The following list contain all the coded character sets known. This does
not necessarily mean that all combinations of these names can be used for
the FROM and TO command line parameters. One coded character set can be
listed with several different names (aliases).
```

```
437, 500, 500V1, 850, 851, 852, 855, 856, 857, 860, 861, 862, 863, 864, 865,
```

866, 866NAV, 869, 874, 904, 1026, 1046, 1047, 8859\_1, 8859\_2, 8859\_3, 8859\_4,  
 8859\_5, 8859\_6, 8859\_7, 8859\_8, 8859\_9, 10646-1:1993, 10646-1: 1993/UCS4,  
 ANSI\_X3.4-1968, ANSI\_X3.4-1986, ANSI\_X3.4, ANSI\_X3.110-1983, ANSI\_X3.110,  
 ARABIC, ARABIC7, ARMSCII-8, ASCII, ASMO-708, ASMO\_449, BALTIC, BIG-5,  
 BIG-FIVE, BIG5-HKSCS, BIG5, BIG5HKSCS, BIGFTVE, BS\_4730, CA, CN-BIG5, CN-GB,  
 CN, CP-AR, CP-GR, CP-HU, CP037, CP038, CP273, CP274, CP275, CP278, CP280,  
 CP281, CP282, CP284, CP285, CP290, CP297, CP367, CP420, CP423, CP424, CP437,  
 CP500, CP737, CP775, CP813, CP819, CP850, CP851, CP852, CP855, CP856, CP857,  
 CP860, CP861, CP862, CP863, CP864, CP865, CP866, CP86QNAV, CP868, CP869,  
 CP870, CP871, CP874, CP875, CP880, CP891, CP903, CP904, CP905, CP912, CP915,  
 CP916, CP918, CP920, CP922, CP930, CP932, CP933, CP935, CP936, CP937, CP939,

//此处省略了部分结果

OSF10020129, OSF10020352, OSF10020354, OSF10020357, OSF100 20359,  
 OSF10020360,

OSF10020364, OSF10020365, OSF10020366, OSF10020367, OSF100 20370,  
 OSF10020387,

OSF10020388, OSF10020396, OSF10020402, OSF10020417, PT, PT2, PT154, R8,  
 RK1048, ROMAN8, RUSCII, SE, SE2, SEN\_850200\_B, SEN\_850200\_C, SHIFT-JIS,  
 SHIFT\_JIS, SHIFT\_JISX0213, SJIS-OPEN, SJIS-WIN, SJIS, SS636127,  
 STRK1048-2002, ST\_SEV\_358-88, T.61-8BIT, T.61, T.618BIT, TCVN-5712, TCVN,  
 TCVN5712-1, TCVN5712-1:1993, TIS-620, TIS620-0, TIS620.2529-1,  
 TIS620.2533-0,

TIS620, TS-5881, TSCII, UCS-2, UCS-2BE, UCS-2LE, UCS-4, UCS-4BE, UCS-4LE,  
 UCS2, UCS4, UHC, UJIS, UK, UNICODE, UNICODEBIG, UNICODELITTLE, US-ASCII, US,  
 UTF-7, UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF-32, UTF-32BE, UTF-32LE, UTF7,  
 UTF8, UTF16, UTF16EE, UTF16LE, UTF32, UTF32EE, UTF32LE, VISCII, WCHAR\_T,  
 WIN-SAMI-2, WINBALTRIM, WINDOWS-31J, WINDOWS-874, WINDOWS-936,  
 WINDOWS-1250,

WINDOWS-1251, WINDOWS-1252, WINDOWS-1253, WINDOWS-1254, WINDOWS-1255,  
 WINDOWS-1256, WINDOWS-1257, WINDOWS-1258, WINSAMI2, WS2, YU

[root@localhost ~]#

本例使用指令 “iconv -l” 显示所有支持的编码字符格式。



fmt

**14.7****pcxto ppm 指令：转换图形文件**

pcxto ppm[必要文件][选择性参数][文件]



pcxto ppm 指令用来转换 pcx 图形文件，转换后的文件的后缀为.ppm，并将结果输出到标准输出。

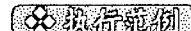


必要参数的具体说明如下：

参    数	功    能
-stapalette	色彩少于 16 时使用默认的色彩配置

选择性参数的具体说明如下：

参    数	功    能
-help	帮助信息
--version	版本信息



◎ 范例 749 ◎ 转换 pcx 文件。

```
[root@localhost ~]# pcxto ppm 123.pcx //转换 pcx 文件
```

本例使用指令“pcxto ppm 123.pcx”转换 pcx 文件 123。



ppmto pcx、ppm

**14.8****pfbtops 指令：转换字体文件**

pfbtops[选择性参数][文件]

**功能说明** pfbtops 指令用来转换字体文件。

**参数说明**

选择性参数的具体说明如下：

参    数	功    能
-v	版本信息

**执行说明**

◎ 范例 750 ◎ 转换字体文件。

```
[root@localhost ~]# pfbtops /usr/share/texmf/fonts/type1/urw/ helvetica/
uhvbo8ac.pfb >uhvbo8ac.pfa //转换字体文件
[root@localhost ~]# tail -n 30 uhvbo8ac.pfa //显示文件的最后 30 行
e9379748ecb285c670091b884f0bab2e5e9a80af18d59c666bdd0334ac34ef85
f7257d1597d834f14948201b5e469525eda86873fd96b72ef0d04e7e5655946
5c36fd96b028286ddb813f02341b0104d050f04d4ca15366c33eb1e6a0dadf46
ee0787d23221c01412c9755466257617bbda509ffe67aa38bf03874b68eeec64
f81af569049e4a1196320274669801b0af5fed1fb7a2c082ef896b15a13f3a8
 //此处省略了部分结果
00
00
00
00
00
00
cleartomark
```

```
[root@localhost ~]#
```

本例使用指令“pfbtops /usr/share/texmf/fonts/type1/urw/ helvetica/uhvbo8ac.pfb >uhvbo8ac.pfa”转换字体文件 uhvbo8ac.pfb。

**相关命令**

grops

## 14.9

## pictoppm 指令：文件转换

**功能说明**

pictoppm[选择性参数][必要参数][文件]



## 第 14 章 格式转换

**功能说明** pictoppm 指令将 pict 类型文件转换为 ppm 类型文件。

**参数说明**

必要参数的具体说明如下：

参 数	功 能
-fullres	最大分辨率
-noheader	不忽略头部数据
-quickdraw	快速绘制
-v	运行时显示详细的处理信息

选择性参数的具体说明如下：

参 数	功 能
-fontdir<文件>	指定配置文件

**执行范例**

◎ 范例 751 ◎ 转换文件。

```
root@localhost:~#pictoppm abc.pict //转换 pict 文件
本例使用指令“pictoppm abc.pict”转换文件 pictoppm abc.pict.
```

**相关指令**

pfbtops、pcxtoppm

## 14.10

## pi1oppm 指令：转换图形文件

**语 法** pi1oppm[选择性参数][图形文件]

**功能说明** pi1oppm 指令用来将 pi1 文件转换为 ppm 文件，并将结果输出到标准输出。

**参数说明**

选择性参数的具体说明如下：

参 数	功 能
-help	帮助信息
-version	版本信息

**◆ 执行范例**

- ◎ 范例 752 ◎ 转换文件。

`[root@localhost ~]#piltoppm abc.pil //转换 pil 文件`

本例使用指令“piltoppm abc.pil”转换 pil 文件 abc。

**◆ 相关指令**

pcxppm、pictoppm

**14.11****pjtoppm 指令：转换打印文件****◆ 命令** pjtoppm[文件]

**◆ 功能说明** pjtoppm 指令将打印文件转换成 Portable Pixmap 格式的文件，并将结果输出到标准输出。

**◆ 执行范例**

- ◎ 范例 753 ◎ 转换文件。

`[root@localhost ~]#pjtoppm fromprinter //转换文件`

本例使用指令“pjtoppm fromprinter”，将打印文件转换成 Portable Pixmap 格式的文件，并将结果输出到标准输出。

**◆ 相关指令**

pictoppm、pcxtoppm

**14.12****qrttoppm 指令：转换 qrt 文件****◆ 命令** qrttoppm[qrt 文件]

**◆ 功能说明** qrttoppm 指令可以把 qrt 文件转换为 ppm 文件，并将结果输出到标准输出设备。

**◆ 执行范例**

- ◎ 范例 754 ◎ 转换扫描文件 fromraytracer。

`[root@localhost ~]#qrttoppm fromraytracer //转换 qrt 文件`

本例使用指令“qrttoppm fromraytracer”转换指定的 qrt 文件。

fromraytracer 为 ppm 文件，并将结果输出到标准输出。



pcxtoppm、pictoppm

## 14.13 sox 指令：音频文件转换



**sox[必要参数][选择性参数]**



sox 指令可转换各种音频文件。



必要参数的具体说明如下：

参 数	功 能
-e	结果输出到标准输出
-p	快速转换
-v	运行时显示详细的处理信息
[stat]	分析文件

选择性参数的具体说明如下：

参 数	功 能
-<频道>	指定频道
-t<文件类型>	指定文件类型
-f<频率>	指定频率
-v<音量>	设置音量



◎ 范例 755 ◎ 分析音频文件。

```
-sh-2.05b# locate *.au //查找音频文件
/usr/lib/python2.2/test/audiotest.au
/usr/share/apps/kbounce/sounds/wallstart.au
/usr/share/apps/kbounce/sounds/death.au
/usr/share/apps/kbounce/sounds/reflect.au
```

```
/usr/share/apps/kbounce/sounds/seconds.au
/usr/share/apps/kbounce/sounds/timeout.au
/usr/share/apps/kbounce/sounds/wallend.au
-sh-2.05b# sox /usr/share/apps/kbounce/sounds/wallstart.au -e stat
//显示音频文件信息
Samples read: 1231
Length (seconds): 0.153875
Scaled by: 2147483647.0
Maximum amplitude: 0.253784
Minimum amplitude: -0.253784
Midline amplitude: 0.000000
Mean norm: 0.026462
Mean amplitude: 0.009702
RMS amplitude: 0.055587
Maximum delta: 0.507568
Minimum delta: 0.000000
Mean delta: 0.030106
RMS delta: 0.062714
Rough frequency: 1435
Volume adjustment: 3.940
-sh-2.05b#
```

本例使用指令“sox /usr/share/apps/kbounce/sounds/wallstart.au -e stat”分析音频文件 wallstart.au 的信息。



play、rec、soxexam

<http://www.TopSage.com>

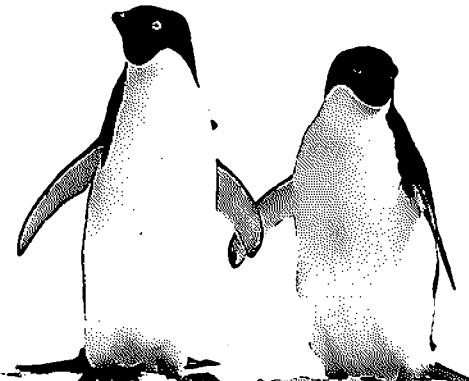




## 第15章

### 系统软件工具

系统软件工具是指 Linux 系统中的软件工具，包括各种计算器等。使用这些指令，用户可以很方便地使用这些软件工具。下面详细介绍这些指令的使用。





## 15.1 dc 指令：一个任意精度的计算器

**语法** dc[选择性参数][文件]

**功能说明** dc 指令是一个 reverse-polish 桌面计算器，它支持无限制精度算法。通常情况下，dc 指令从标准输入读取数据，如果指定了文件名的命令行参数，那么指令将从指定的文件中读取数据，而不是从标准输入读取数据。结果将输出到标准输出，所有的错误信息将输出到标准输出。

**参数说明**

选择性参数的具体说明如下：

参 数	功 能
-version	版本信息
-help	帮助信息
<<脚本>	增加脚本中的指令到程序的指令设置
-f<脚本文件>	增加脚本文件中的指令到程序的指令设置

### 【运算类型】

- +: 加法。
- : 加法。
- \*: 乘法。
- /: 除法。
- %: 余数。
- ^: 指数。
- v: 开方。

### 【堆栈操作】

- c: 清空堆栈。
- f: 显示堆栈的所有信息。
- nk: 精度设置，默认精确到个位。

## 15.1 dc 指令：一个任意精度的计算器

- q: 结束程序。
- P: 输出堆栈最顶端的值。
- p: 弹出堆栈最顶端的值。

 实际范例

## ◎ 范例 756 ◎ 利用 dc 指令进行乘法运算。

```
[root@localhost ~]# dc //运行 dc 指令
4 //输入被乘数
5 //输入被乘数
*
p //弹出堆栈最顶端的值
20 //输出结果
q //退出
```

[root@localhost ~]#

乘法的运算过程如上所示，其他的加法、减法、除法、开方等的运算过程与乘法运算类似。

## ◎ 范例 757 ◎ 一个综合运算的例子。

```
[root@localhost ~]# dc //利用 dc 指令进行计算
7 //用户输入
8 //输出结果
*
p //用户输入
56 //输出结果
3 2 ^ p //用户输入
9 //输出结果
2 / p //用户输入
4 //输出结果
f //用户输入
4 //输出结果
56 //用户输入
+p //输出结果
60 //用户输入
q //用户输入
```

[root@localhost ~]#

 相关指令

expr、xcalc



## 15.2 expr 指令：简单计算器

**语法**    `expr[选择性参数][表达式]`

**功能说明**    `expr` 指令打印表达式的值到标准输出。

**参数说明**

选择性参数的具体说明如下：

参 数	功 能
<code>-help</code>	帮助信息
<code>-version</code>	版本信息

### 【表达式】

- `|`: 逻辑或。
- `&`: 逻辑与。
- `<`: 小于。
- `<=`: 小于等于。
- `=`: 等于。
- `!=`: 不等于。
- `>=`: 大于等于。
- `>`: 大于。
- `+`: 加法。
- `-`: 减法。
- `*`: 乘法。
- `/`: 除法。
- `%`: 取余。
- `stringregexp`: 检查 `string` 的模式是否匹配 `regexp`。
- `match string regexp`: 等同于 `string regexp`。
- `index string chars`: `string` 中 `chars` 的索引。
- `length string`: `string` 的长度。
- `+ token`: 将 `token` 看作为一个字符串。

**执行范例****◎ 范例 758 ◎ 计算字符串的长度。**

```
[root@localhost ~]# expr length jdlkfjlkadjflkjdsjlfjdlsk //计算字符串长度
25
```

```
[root@localhost ~]#
```

本例使用指令“expr length jdlkfjlkadjflkjdsjlfjdlsk”计算字符串 jdlkfjlkadjflkjdsjlfjdlsk 的长度，得到的结果是字符串的长度为 25，如上所示。

**◎ 范例 759 ◎ 判断表达式的真假。**

```
[root@localhost ~]# expr 15 '>' 25 //判断 15 大于 25 的真假
```

```
0
```

```
[root@localhost ~]# expr 15 '<' 25 //判断 15 小于 25 的真假
```

```
1
```

```
[root@localhost ~]#
```

通过以上指令判断 15 和 25 的大小关系。如果表达式为真，则返回 1；如果表达式为假，则返回 0。

**◎ 范例 760 ◎ 数学计算。**

```
[root@localhost ~]# expr 6 '*' 8 //计算 6*8 的值
```

```
48
```

```
[root@localhost ~]#
```

本例使用指令“expr 6 '\*' 8”计算 6\*8 的结果。

**相关命令**

dc

<http://www.TopSage.com>



## 附录 指令索引(按首字母排序)

### A

- adduser 指令: 增加一个系统用户 2
- alias 指令: 设置指令的别名 158
- apachectl 指令: apache HTTP 服务器控制接口 482
- apm 指令: 高级电源管理 3
- apmd 指令: 高级电源管理 158
- apmsleep 指令: APM 进入备用或者休眠状态 4
- apropos 指令: 查找使用手册的名字和相关描述 5
- arp 指令: 系统 ARP 缓存 483
- arpwatch 指令: 监听 ARP 记录 484
- ar 指令: 建立、修改或从档案文件中提取文件 438
- arch 指令: 输出主机的体系结构 8
- arping 指令: 向邻居主机发送 ARP 请求 485
- as 指令: 标准 GNU 汇编程序 598
- aspell 指令: 检查文件的错误 338
- at 指令: 系统将执行的任务排队 159
- atd 指令: 执行已经排队的任务 162
- atq 指令: 检查排队的任务 163
- atrm 指令: 删除已经排队的任务 164
- atrun 指令: 执行已排队的任务 164
- attr 指令: XFS 文件系统对象的扩展属性 340
- aumix 指令: 设置音效设备 165
- authconfig 指令: 配置系统的认证信息 167



## 附录 指令索引（按首字母排序）

autoconf 指令：产生配置脚本 601

autoheader 指令：为 configure 产生模板头文件 603

automount 指令：为 auto 文件系统配置挂载点 562

autoreconf 指令：更新已经生成的配置文件 604

autoscan 指令：产生初步的 configure.in 文件 605

autoupdate 指令：更新 configure.in 文件 606

awk 指令：模式匹配语言 248

### B

badblocks 指令：检查磁盘坏道 582

basename 指令：显示文件或者目录的基本名称 341

batch 指令：执行批处理指令 8

bg 指令：将程序放在后台执行 9

bind 指令：显示或者设置按键组合 168

bunzip2 指令：解压缩.bz2 类型的文件 439

bye 指令：中断 FTP 连接 312

bzip2 指令：解压缩.bz2 文件 441

bzip2recover 指令：损坏.bz2 文件的修复 442

### C

cat 指令：输出文件内容 614

cd 指令：切换目录 9

cfdisk 指令：磁盘分区 583

chattr 指令：改变文件的属性 341

chfn 指令：设置 finger 信息 10

chkconfig 指令：设置系统的应用程序 171

chgrp 指令：改变文件或者目录所属的群组 343

chmod 指令：设置文件或者目录的权限 345

chown 指令：改变文件的拥有者或者群组 347

chroot 指令：改变根目录 175

chsh 指令: 改变登录系统时的 shell 12  
cksum 指令: 文件的 CRC 校验 349  
clear 指令: 清除终端屏幕 13  
clock 指令: 系统 RTC 时间设置 176  
cmp 指令: 比较文件差异 350  
col 指令: 过滤控制字符 250  
colrm 指令: 删 除指定的列 251  
comm 指令: 比较排序文件 252  
compress 指令: 压缩解压文件 443  
cp 指令: 复制 352  
cpio 指令: 备份文件 444  
crontab 指令: 设置计时器 177  
csplit 指令: 分割文件 355  
cu 指令: 主机间通信 488  
cut 指令: 剪切文件 615

## D

date 指令: 显示或设置系统时间 13  
dc 指令: 一个任意精度的计算器 660  
dd 指令: 转换复制文件 584  
declare 指令: 显示或者设定 shell 变量 178  
depmod 指令: 模块关系 180  
df 指令: 磁盘信息 563  
diff 指令: 生成差异信息 356  
diffstat 指令: diff 结果的统计信息 362  
dip 指令: IP 拨号连接 489  
dircolors 指令: ls 指令对应的显示颜色设置 181  
dirname 指令: 显示文件的除名字外的路径 363  
dirs 指令: 显示目录信息 565  
dmesg 指令: 显示开机信息 182



## 附录 指令索引（按首字母排序）

**du** 指令：显示目录或者文件所占的磁盘空间 566

**dump** 指令：文件系统备份 450

**dvips** 指令：将 DVI 文件转换为 Postscript 文件 646

### E

**e2fsck** 指令：检查 ext2 文件系统 586

**echo** 指令：显示文本行 16

**ed** 指令：文本编辑器 254

**egrep** 指令：输出匹配某种模式的行 256

**enable** 指令：可用的 shell 的内置指令 184

**exec** 指令：执行完指令后，交出控制权 18

**ext2ed** 指令：ext2 文件系统编辑 256

**exit** 指令：shell 退出 18

**export** 指令：设置或者显示环境变量 187

**expr** 指令：简单计算器 662

**eval** 指令：连接过个指令 186

### F

**false** 指令：不做任何事情，表示失败 191

**fbset** 指令：设置帧缓冲区 192

**fc** 指令：修改或者执行指令 19

**fdisk** 指令：Linux 分区表控制 451

**fetchmail** 指令：获得邮件 628

**fg** 指令：将后台任务拉到前台执行 21

**fgconsole** 指令：打印虚拟终端的数目 22

**fgrep** 指令：匹配字符串 256

**fiascotopnm** 指令：将压缩的 fiasco 镜像文件转换为 pgm 或 ppm 格式 647

**file** 指令：辨识文件类型 364

**filterdiff** 指令：从 diff 文件中提取不同 365

**find** 指令：查找目录或者文件 366

附录 指令索引（按首字母排序）

find2perl 指令：将 find 指令行转换为 perl 代码 648  
findfs 指令：通过列表或者用户 ID 查找文件系统 374  
finger 指令：查找并显示用户信息 22  
fmt 指令：编排文本文件 259  
fold 指令：限制文件列宽 312  
free 指令：显示内存信息 24  
fsck 指令：检查文件系统并尝试修复错误 454  
fsck.ext2 指令：检查 ext2 文件系统 456  
fsck.ext3 指令：检查 ext3 文件系统 457  
fsck.minix 指令：检查文件系统并尝试修复错误 458  
ftp 指令：文件传输协议 314  
ftpcount 指令：显示 ftp 登录用户数 321  
ftpsshut 指令：定时关闭 ftp 服务器 321  
ftpwho 指令：显示 ftp 登录用户信息 322  
fuser 指令：用文件或者套接口表示进程 25  
fwhois 指令：显示用户的信息 27

**G**

gaim 指令：即时信息传输 490  
gcc 指令：GNU 的 C 和 C++ 编译器 607  
gcov 指令：coverage 测试工具 28  
gdb 指令：GNU 调试器 608  
gdbserver 指令：远端 GNU 服务器 610  
gdialog 指令：从 shell 显示文本信息 30  
gedit 指令：gnome 的文本编辑器 260  
gemtopbm 指令：转换图形文件 648  
getlist 指令：下载新闻 631  
getty 指令：设置终端配置 490  
giftopnm 指令：文件转换 649  
git 指令：在文字模式下管理文件 375



## 附录 指令索引（按首字母排序）

gitps 指令：显示程序情况 32  
groupadd 指令：创建一个新的群组 33  
groupdel 指令：删除一个群组 33  
groupmod 指令：改变系统群组的属性 34  
gunzip 指令：解压缩文件 459  
gzexe 指令：压缩可执行文件 460  
gzip 指令：压缩文件 460

### H

halt 指令：关闭系统 35  
hash 指令：显示和清除哈希表 193  
hdparm 指令：显示和设定硬盘参数 462  
head 指令：输出文件开头的部分信息 261  
help 指令：显示 shell 的内建指令的帮助信息 36  
history 指令：显示历史指令 38  
host 指令：dns 查询 491  
hostid 指令：打印出当前主机的标识 194  
hostname 指令：显示或者设置当前系统的主机名 195  
htpasswd 指令：创建和更新用户的认证文件 40  
httpd 指令：apache http 服务器程序 495  
hwclock 指令：显示与设定硬件时钟 (rtc) 197

### I

iconv 指令：字符集转换 650  
id 指令：显示用户的 ID 以及所属群组的 ID 42  
ifconfig 指令：显示或者配置网络设备 497  
indent 指令：调整 C 原始代码文件的格式 375  
info 指令：读取目录信息 43  
init 指令：开关机设置 46  
insmod 指令：载入模块 199

附录 指令索引 (按首字母排序)

ipcs 指令: 显示进程间通信的信息 47

ipcrm 指令: 删除消息队列、旗语设置或者共享内存的 ID 49

iptables 指令: IPv4 的包过滤和 nat 的管理 500

iptables-save 指令: Ip 列表存储 504

isosize 指令: 显示 iso9660 文件系统信息 200

ispell 指令: 拼字检查程序 263

iwconfig 指令: 配置无线网络设备 507

J

jed 指令: 编辑文本文件 266

jobs 指令: 列出所有后台程序 50

joe 指令: 编辑文本文件 267

join 指令: 将两个文件中与指定栏位内容相同的行连接起来 272

K

kbdconfig 指令: 设置键盘类型 200

kill 指令: 杀死执行中的进程 51

killall 指令: 杀死同名的所有进程 52

L

last 指令: 列出目前与过去登录系统的用户相关信息 55

lastb 指令: 列出登录系统失败的用户相关信息 57

ldconfig 指令: 配置动态链接绑定 201

ldd 指令: 打印共享库文件的相互依赖关系 202

less 指令: 一次显示一页文本 275

lha 指令: 压缩或解压缩文件 464

lilo 指令: 引导安装程序 203

liloconfig 指令: 设置程序载入 204

ln 指令: 链接文件或目录 378

ln -s 指令: 连接目录内容 380



## 附录 指令索引（按首字母排序）

**locate** 指令：查找文件 381

**login** 指令：登录系统 58

**logname** 指令：显示登录账号的信息 59

**logrotate** 指令：处理 log 文件 59

**logout** 指令：退出系统 60

**losetup** 指令：设置循环 205

**lsattr** 指令：显示文件属性 382

**lsmod** 指令：显示 Linux 内核的模块信息 61

## M

**mail** 指令：收发邮件 632

**mailq** 指令：显示发件箱的邮件 365

**make** 指令：编译内核或模块 611

**man** 指令：格式化和显示在线手册 62

**manpath** 指令：设置 man 手册的查询路径 65

**mattrib** 指令：变更或显示 MS-DOS 文件的属性 384

**mc** 指令：交互式文件管理程序 385

**mcopy** 指令：复制 MS-DOS 文件到 UNIX 系统，或者从 UNIX 系统复制 MS-DOS 文件 387

**md5sum** 指令：检验文件 387

**mdel** 指令：MS-DOS 文件删除 389

**mdir** 指令：显示 MS-DOS 目录 390

**mesg** 指令：控制终端的写入 508

**mev** 指令：监视鼠标情况 206

**minfo** 指令：显示 MS-DOS 文件系统的各项参数 207

**mingetty** 指令：精简版的 getty 509

**minicom** 指令：调制解调器通信程序 510

**mkbootdisk** 指令：建立目前系统的启动盘 587

**mkdir** 指令：建立目录 390

**mke2fs** 指令：建立 ext2 文件系统 588

**mkfifo** 指令: 创建管道 511  
**mkfontdir** 指令: 创建字体文件目录 66  
**mkfs** 指令: 建立各种文件系统 589  
**mkfs.minix** 指令: 建立 minix 文件系统 590  
**mkinitrd** 指令: 建立要载入 ramdisk 的映像文件 591  
**mkisofs** 指令: 建立 iso 9660 映像文件 592  
**mkkickstart** 指令: 建立安装的组态文件 207  
**mkraid** 指令: 初始化/升级 RAID 设备阵列 208  
**mkswap** 指令: 设置交换区 swap area) 593  
**mktemp** 指令: 建立暂存文件 391  
**mmove** 指令: 移动 MS-DOS 文件 392  
**modinfo** 指令: 显示内核信息 209  
**modprobe** 指令: 自动处理可载入模块 210  
**more** 指令: 显示文件信息 277  
**mount** 指令: 挂载文件系统 67  
**mouseconfig** 指令: 设置鼠标相关参数 211  
**mpost** 指令: 系统的绘画工具 69  
**mread** 指令: 复制 MS-DOS 文件 393  
**mren** 指令: 更改 MS-DOS 文件名称 393  
**msgcat** 指令: 合并消息目录 70  
**msgcomm** 指令: 匹配两个消息目录 71  
**msgen** 指令: 创建英语消息目录 73  
**msginit** 指令: 初始化消息目录 74  
**mshowfat** 指令: 显示 MS-DOS 文件的记录 393  
**mt** 指令: 磁带驱动操作 569  
**mtools** 指令: 显示 mtools 支持的指令 394  
**mtoolstest** 指令: 测试并显示 mtools 的相关设置 394  
**mtr** 指令: 网络诊断工具 512  
**mutt** 指令: e-mail 管理 636  
**mv** 指令: 移动或更名现有的文件或目录 397



## 附录 指令索引（按首字母排序）

**mzip** 指令：zip/jaz 磁盘驱动器控制指令 571

### N

**nano** 指令：文本编辑器 278

**nc** 指令：设置路由器 513

**ncftp** 指令：传输文件 322

**ncftpgt** 指令：下载文件 327

**ncftpls** 指令：显示文件目录 328

**ncftpput** 指令：上传文件 329

**netconfig** 指令：设置各项网络功能 516

**netstat** 指令：显示网络状态 517

**newgrp** 指令：登录另一个群组 75

**nice** 指令：设置优先权 213

**nntpget** 指令：从新闻服务器下载文章 637

**nohup** 指令：退出系统继续执行指令 76

**nslookup** 指令：dns 查找 523

**ntsysv** 指令：设置系统的各种服务 76

### O

**od** 指令：输出文件内容 398

**open** 指令：开启虚拟终端 77

### P

**passwd** 指令：设置密码 214

**paste** 指令：合并文件的列 402

**patch** 指令：修补文件 404

**pcxtoppm** 指令：转换图形文件 652

**pfbtops** 指令：转换字体文件 652

**pg** 指令：浏览文件 280

**pgrep** 指令：基于名字和其他属性的查找或信号处理 79

pico 指令：编辑文字文件 283  
pictoppm 指令：文件转换 653  
pidof 指令：找到运行程序的进程 ID 85  
piltoppm 指令：转换图形文件 654  
pine 指令：收发邮件 638  
ping 指令：检测主机（IPv4） 525  
ping6 指令：检测主机（IPv6） 527  
pjtoppm 指令：转换打印文件 655  
pkill 指令：杀死进程 86  
pmap 指令：显示程序的内存信息 89  
pppd 指令：ppp 连线的守护进程 529  
ppp-off 指令：关闭 ppp 连线 530  
pppsetup 指令：设置 ppp 连线 530  
pppstats 指令：显示 ppp 连线状态 530  
pr 指令：编排文件格式 617  
procinfo 指令：显示系统状态 94  
ps 指令：报告程序状况 95  
pstree 指令：以树状图显示程序 99  
pwconv 指令：开启用户的投影密码 215  
pwck 指令：检查密码文件 104  
pwd 指令：显示工作目录 105  
pwunconv 指令：关闭用户的投影密码 216

Q

qrtoppm 指令：转换 qrt 文件 655  
quota 指令：显示磁盘已使用的空间与限制 571  
quotacheck 指令：检查磁盘的使用空间与限制 572  
quotaoff 指令：关闭磁盘空间限制 573  
quotaon 指令：开启磁盘空间限制 574  
quotastats 指令：显示磁盘空间的限制 575



## 附录 指令索引（按首字母排序）

### R

raidstop 指令：关闭软件控制的磁盘阵列 576  
rcp 指令：复制远端主机的文件或目录 407  
rdate 指令：显示其他主机的日期与时间 531  
reboot 指令：重新启动 106  
rencie 指令：调整优先权 106  
repquota 指令：检查磁盘空间限制的状态 576  
resize 指令：设置终端机视窗的大小 216  
restore 指令：还原 dump 操作备份的文件 594  
rhmask 指令：产生加密文件 408  
rlogin 指令：远端登录 108  
rm 指令：删除文件或目录 408  
rmdir 指令：删除目录 413  
rmmod 指令：删除模块 111  
rmt 指令：远端磁带传输协议模块 577  
route 指令：显示并设置路由 532  
rpm 指令：管理 RPM 包 217  
rsh 指令：远端登录的 shell 115  
runlevel 指令：显示执行等级 219  
rwho 指令：查看系统用户 118

### S

samba 指令：控制 Samba 服务端 535  
scp 指令：远程复制文件 415  
screen 指令：多重视窗管理程序 119  
sed 指令：利用 script 指令处理文本文件 285  
set 指令：设置 shell 220  
setconsole 指令：设置系统终端 222  
setenv 指令：查询或显示环境变量 223

setserial 指令: 设置或显示串口的相关信息 224  
setup 指令: 设置公用程序 224  
sfdisk 指令: 硬盘分区工具程序 577  
shutdown 指令: 系统关机指令 122  
sleep 指令: 休眠 123  
sliplogin 指令: 将终端机之间的连接变为 sliplogin 连接 228  
slocate 指令: 查找文件或目录 417  
slrn 指令: 新闻阅读程序 639  
smbd 指令: Samba 服务端 536  
sort 指令: 将文本文件内容加以排序 291  
sox 指令: 音频文件转换 656  
spell 指令: 拼字检查程序 297  
split 指令: 切割文件 419  
ssh 指令: 远程登录 537  
startx 指令: 启动图形界面 642  
stat 指令: 显示 inode 内容 421  
statserial 指令: samba 服务器程序 538  
su 指令: 变更用户身份 124  
sudo 指令: 以其他身份来执行指令 125  
sum 指令: 计算文件的校验码 423  
suspend 指令: 暂停执行 shell 127  
swapoff 指令: 关闭系统交换分区 228  
swapon 指令: 启动系统交换分区 230  
swatch 指令: 系统监控程序 128  
symlinks 指令: 维护符号链接的工具程序 129  
sync 指令: 将内存缓冲区内的数据写入磁盘 579  
sysctl 指令: 设置系统核心参数 231

T

tac 指令: 反序输出文件 619



## 附录 指令索引（按首字母排序）

tail 指令：显示文件的末尾内容 621  
talk 指令：与其他用户交谈 540  
tar 指令：压缩/解压缩文件 465  
tcpdump 指令：倾倒网络传输数据 540  
tee 指令：读取标准输入到标准输出并可保存为文件 424  
telinit 指令：设置系统的执行级别 233  
telnet 指令：远程登录 545  
testparm 指令：测试 samba 配置 547  
tftp 指令：传输文件 330  
timeconfig 指令：设置时区 234  
tload 指令：显示系统负载 129  
tmpwatch 指令：删除临时文件 426  
top 指令：显示进程信息 130  
touch 指令：更新文件或目录时间 427  
tr 指令：转换文件中的字符 297  
tracepath 指令：追踪路径 549  
traceroute 指令：显示数据包到主机间的路径 550  
tree 指令：以树状图显示目录内容 428  
tty 指令：显示标准输入设备名称 551

## U

ulimit 指令：控制 shell 程序的资源 235  
umask 指令：指定在建立文件时预设的权限掩码 434  
umount 指令：删除文件系统 467  
unalias 指令：删除别名 237  
uname 指令：显示系统信息 137  
unarj 指令：解压缩文件 468  
uncompress 指令：解压缩.Z 文件 469  
uniq 指令：检查文件中重复出现的行 299  
unset 指令：删除变量或函数 238

unzip 指令：解压缩 zip 文件 470  
up2date 指令：更新 Linux 系统 240  
useradd 指令：建立用户账号 139  
userconf 指令：用户账号设置 140  
usermod 指令：修改用户账号 141  
userdel 指令：删除用户账号 142  
users 指令：显示用户 143  
uucico 指令：uucp 文件传输 332  
uucp 指令：在 UNIX 系统之间传送文件 333  
uulog 指令：显示 uucp 记录信息 552  
uname 指令：显示 uucp 远端主机 553  
uupick 指令：处理文件 334  
uustat 指令：显示 uucp 状态 554  
uuto 指令：文件传送到远端主机 335  
uux 指令：在远端的 uucp 主机上运行指令 555

V

vi 指令：文字编辑器 301  
vim 指令：增强型 vi 编辑器 306  
view 指令：文字编辑器 307  
vlock 指令：锁定终端 143  
vmstat 指令：显示虚拟内存信息 242

W

w 指令：显示登录系统的用户信息 145  
wait 指令：等待程序返回状态 146  
wall 指令：发送信息 559  
watch 指令：将结果输出到标准输出设备 147  
wc 指令：计算字数 308  
wget 指令：从互联网下载资源 556



## 附录 指令索引（按首字母排序）

**whereis** 指令：查找文件 149

**which** 指令：查找文件 150

**who** 指令：显示系统用户信息 151

**whoami** 指令：显示用户名 153

**whois** 指令：显示指定用户信息 154

**write** 指令：传送信息 558

### X

**xset** 指令：设置 X windows 642

### Y

**yes** 指令：持续输出给定的字符串，每行显示一个字符串 245

**ytalk** 指令：与其他用户交谈 559

### Z

**zcat** 指令：显示压缩文件的内容 624

**zip** 指令：压缩文件 472

**zipinfo** 指令：显示压缩文件的信息 476

**&** 指令：将任务放到后台执行 154

读书笔记





## 读书笔记



---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---



## 1998-2007 年全国计算机技术与软件专业资格(水平)考试真题及答案汇总

- 2007年下半年软考网络工程师试题 Word版
- 2007年下半年信息系统项目管理师试题 Word版
- 2007年下半年信息系统监理师试题 Word版
- 2007年下半年信息处理技术员试题 Word版
- 2007年下半年系统分析师试题 Word版
- 2007年下半年网络管理员试题 Word版
- 2007年下半年数据库系统工程师试题 Word版
- 2007年下半年软件设计师试题 Word版
- 2007年下半年嵌入式系统设计师试题 Word版
- 2007年下半年多媒体应用设计师试题 Word版
- 2007年下半年电子商务设计师试题 Word版
- 2007年下半年电子商务技术员试题 Word版
- 2007年下半年程序员试题 Word版
- 2007年上半年系统分析师试题 PDF版
- 2007年上半年信息系统监理师试题 PDF版
- 2007年上半年信息系统管理工程师试题 PDF版
- 2007年上半年信息处理技术员试题 PDF版
- 2007年上半年网络管理员试题 PDF版
- 2007年上半年数据库系统工程师试题 PDF版
- 2007年上半年网络工程师试题 PDF版
- 2007年上半年网络工程师试题答案 Doc
- 2007年上半年软件设计师试题 PDF版
- 2007年上半年软件设计师试题答案 Doc
- 2007年上半年软件评测师试题 PDF版
- 2007年上半年程序员题 PDF版
- 2006年下半年信息系统项目管理师试题 PDF版
- 2006年下半年系统分析师试题 PDF版
- 2006年下半年信息系统监理师试题 PDF版
- 2006年下半年网络工程师试题 PDF版
- 2006年下半年网络工程师试题及答案 Word版
- 2006年下半年软件设计师试题 PDF版
- 2006年下半年软件设计师试卷及答案 Word版
- 2006年下半年嵌入式系统设计师试题 PDF版



- 2006年下半年电子商务设计师试题 PDF版
- 2006年下半年网络管理员试题 PDF版
- 2006年下半年程序员试题 PDF版
- 2006年上半年程序员试题 Word版
- 2006年上半年网络工程师试题(含答案) Word版
- 2006年上半年软件设计师试题及答案 Word版
- 2005年下半年软件设计师试题及答案 Word版
- 2005年下半年网络工程师试题及答案 Word版
- 2005年下半年网络管理员试题
- 2005年上半年网络工程师试题(含答案) Word版
- 2005年上半年软件设计师试题(含答案) Word版
- 2005年上半年程序员上午试题(word版)
- 2005年上半年网络管理员试题
- 2005年上半年网络管理员上午试题(word版)
- 2005年上半年网络管理员下午试题(word版)
- 2005年上半年网络工程师上午试题(word版)
- 2005年上半年网络工程师下午试题(word版)
- 2005年上半年软件设计师上午试题(word版)
- 2005年上半年软件设计师下午试题(word版)
- 2004年下半年网络工程师试题(含答案) Doc
- 2004年下半年软件设计师试题(含答案) Word版
- 2004年下半年网络工程师上午试题(word版)
- 2004年下半年网络工程师下午试题(word版)
- 2004年下半年程序员上午试题(word版)
- 2004年上半年网络工程师试题(含答案) Word版
- 2004年上半年软件设计师试题含答案 Word版
- 2004年下半年网络管理员员试题
- 2004年上半年软件设计师(高级程序员)上午试题(word版)
- 2004年上半年软件设计师(高级程序员)下午试题(word版)
- 2004年上半年程序员上午试题(word版)
- 2004年上半年程序员下午试题(word版)
- 2003年高级程序员试题(含答案) Word版
- 2003年度网络设计师试题及答案 Word版
- 2003年网络设计师上(下)午试题(word版)
- 2003年程序员考试上(下)午试题(word版)



2003年初级程序员上午试题(word版)  
2003年初级程序员下午试题(word版)  
2002年度网络设计师级试题及答案 Word版  
2002年度高级程序员级试题答案  
2002年网络程序员试题  
2002年系统设计师(高级程序员)上午试题(word版)  
2002年系统设计师(高级程序员)下午试题(word版)  
2002年系统分析员级上(下)午试题(word版)  
2002年网络程序员级上(下)午试题(word版)  
2001年度网络设计师级试题及答案 Word版  
2001年度高级程序员级试题及答案 Word版  
2001年高级程序员级下午试题(word版)  
2001年高级程序员级上午试题(word版)  
2001年网络设计师考试上午试题(word版)  
2001年网络设计师考试上午试题(word版)  
2001年程序员上午试题(word版)  
2001年初级程序员级下午试题(word版)  
2001年度网络程序员级试题  
2001年系统分析员级上午试题(word版)  
2000年高级程序员级试题及答案 Word版  
2000年初级程序员级上午试题(word版)  
2000年初级程序员级下午试题(word版)  
2000年初高程序员级上午试题(word版)  
2000年初高程序员级下午试题(word版)  
2000年程序员上午试题(word版)  
1999年初级程序员上午试题(word版)  
1999年初级程序员级下午试题(word版)  
1999年程序员上午试题(word版)  
1999年程序员下午试题(word版)  
1999年高级程序员上午试题及答案(word版)  
1999年高级程序员下午试题(word版)  
1998年高级程序员级上午试题(word版)  
1998年高级程序员级下午试题(word版)



计算机技术与软件专业资格考试推荐视频教程下载汇总

数据结构视频教程 清华大学严蔚敏主讲 全 48 讲 ASF格式

数据结构C语言版视频教程 全 52 讲

操作系统原理视频教程 浙江大学徐宗元教授主讲(全 32 讲)

数据结构与算法 浙大徐镜春教授主讲(全 40 讲)

计算机网络基础 陆魁军主讲(全 32 讲)

数据库系统概论 浙大张军主讲(全 32 讲)

软件工程基础 浙大陈天洲主讲(全 32 讲)

面向对象程序设计 浙大毛根生主讲(全 30 讲)

**程序设计语言视频教程:**

孙鑫C++视频教程 rmvb格式 全 20CD

中山大学蔡培兴 C++语言视频教程 全 51 讲

Java视频教程 孙鑫Java无难事(全 12CD)

Java视频教程 即学即会java

中山大学汇编语言视频教程 全 51 讲

汇编语言程序设计视频教程(郝玉洁主讲 全 36 讲)

**其他专项视频教程系列:**

[程序员考试视频教程] 硬件基础 二进制编码

[程序员考试视频教程] 图的遍历

[程序员考试视频教程] 数据库 关系运算

[程序员考试视频教程] 操作系统 作业调度算法

[网络工程师培训视频教程] 帧中继拥塞控制

[网络工程师培训视频教程] 以太网CSMA/CD

[网络工程师培训视频教程] 奈氏准则和香农公式

[网络工程师培训视频教程] 滑动窗口模型

[网络工程师培训视频教程] TCP数据传输

[信息系统项目管理师视频教程]前言

[信息系统项目管理师视频教程]论文写作 注意事项及例题

[信息系统项目管理师视频教程]项目管理案例分析 例题讲解

[信息系统项目管理师视频教程]项目组合管理 盈亏平衡点

[信息系统项目管理师视频教程]项目质量管理 软件质量特性

[信息系统项目管理师视频教程]项目进度管理 关键路径

[信息系统项目管理师视频教程]项目沟通管理 项目管理信息系统



- [信息系统项目管理师视频教程]项目风险分析 决策树分析
- [信息系统项目管理师视频教程]项目范围管理 实现范围变更
- [信息系统项目管理师视频教程]项目成本管理 挣值法的例子
- [信息系统项目管理师视频教程]项目成本管理 影响成本估算的因素
- [信息系统项目管理师视频教程]成本效益分析 动态投资回收期
- [软件设计师视频教程] 最小生成树
- [软件设计师视频教程] 知识产权-著作权法解读
- [软件设计师视频教程]数据流图设计(例题讲解)
- [软件设计师视频教程] 数据安全-对称加密
- [软件设计师视频教程] 面向对象继承访问控制
- [软件设计师视频教程] 操作系统习题讲解
- [软件设计师视频教程] Web Service及应用



## .NET 语言(C#/VB)电子资料、开发工具下载汇总

### Visual Studio 相关电子资料、软件汇总：

[VS代码辅助工具Visual Assist X 10.4 完美版+特别文件](#)

[CodeSmith 4.1.2 专业版 最新完美版 .NET代码模板生成工具](#)

[Altova MissionKit 2008 for Enterprise Software Architects完美版](#)

[正则表达式辅助生成工具RegexBuddy 3.0.5 破解版](#)

[Pro Visual Studio 2005 Team System](#)

[Microsoft Visual Studio 2005 Unleashed](#)

[Visual Studio Team System Better Software Development for Agile Team](#)

### .NET 开发语言电子资料汇总：

[Pro C# 2008 and the .NET 3.5 Platform](#)

[Apress出版 Accelerated C# 2008](#)

[Pro LINQ:Language Integrated Query in C# 2008](#)

[MS Press - Introducing Microsoft LINQ](#)

[LINQ for Visual C# 2005 \(07年6月出版\)](#)

[LINQ for VB 2005 \(07年6月最新PDF文字版\)](#)

[Wrox C# 入门经典](#)

[C# 设计模式](#)

[C# 网络核心编程](#)

[Windows应用高级编程 C#编程篇](#)

[C#高级编程\(第三版\)](#)

[数据结构与算法 C#语言版](#)

[C#字符串和正则表达式参考手册](#)

[O'Reilly 正则表达式参考手册 第二版 2007年最新出版](#)

[Programming Microsoft Windows with C#](#)

[C# 2005 图解教程](#)

[Visual C# 2005 Express Edition编程初学者指南](#)

[Programming .NET Framework with C#](#)

[C#语言参考](#)

[C#应用程序开发](#)

[Client Side Reporting with Visual Studio in C#](#)

[Wrox Professional VB 2005 with .NET 3.0](#)



Build A Program Now Visual Basic 2005  
.NET游戏编程入门经典—VB.NET篇  
O'Reilly Visual Basic 2005 Cookbook  
.NET Insight for Classic VB Developers  
Fast Track Visual Basic.NET  
How to Code .NET  
ADO.NET全攻略  
Apress Professional ADO.NET 2.0  
Oreilly .NET and XML  
.NET组件编程 (第二版)  
Wrox Beginning Visual C++ 2005  
Visual C++.NET专业项目  
精通.Net核心技术原理与构架  
Cross-Platform Web Services Using C# and Java  
Advanced C# Programming



## .NET、ASP.NET 控件及源码大汇总

CuteEditor 6.0 在线HTML编辑器的领航者

ComponentArt.Charting.WebChart.dll

ComponentArt.Web.UI 2007.2 源代码+实例+DLL

ComponentArt.WebUI.2007.1 破解DLL

ComponentArt.WebUI.2007.1 源代码

ComponentArt.Web.UI.2006.2

ComponentArt.Web.UI.2006.2 源代码

ComponentArt.Web.UI.2006.1

Infragistics NetAdvantage for ASP.NET 2007 Vol 2

Infragistics NetAdvantage AppStylist 2007 Vol 2

Infragistics TestAdvantage WinForms 2007 For CLR2

Infragistics TestAdvantage WinForms 2007 for CLR1.x

Infragistics NetAdvantage for Windows Forms 2007 Vol 2

Infragistics NetAdvantage 2007 for WPF

Infragistics NetAdvantage 2006 Vol2 CLR1.x

Infragistics NetAdvantage 2006 Vol2 for CLR2

Infragistics NetAdvantage 2006 Vol2 CLR1.x

ComponentOne Studio 2007 v1.5 for ASP.NET 2.0

ComponentOne Studio 2007 v1.5 for ASP.NET 1.x

ComponentOne Studio 2006 v2 for ASP.NET 2.0

ComponentOne Studio 2006 v2 for ASP.NET 1.x

ComponentOne Studio for Mobile Devices 2007 v1.5 CLR1x

ComponentOne Studio for Mobile Devices 2006 v2 CLR2

ComponentOne Studio for Mobile Devices 2006 v2 CLR1.x

ComponentOne Studio 2007 v1.5 for .NET CLR2

ComponentOne Studio 2007 v1.5 for .NET CLR1.x

ComponentOne Studio for .NET 2006 v2 CLR2

ComponentOne Studio for .NET 2006 v2 CLR1.x

ComponentOne Studio for ActiveX 2007 v1.5

ComponentOne Studio for ActiveX 2006 v2 CLR2

ComponentOne Studio for ActiveX 2006 v2 CLR1.x

Telerik RadWindow for ASP.NET 2.0 v1.8.2.0

Telerik RadUpload for ASP.NET 2.0 v2.3.2.0



Telerik RadTreeView for ASP.NET 2.0 v6.2.2.0  
Telerik RadTabStrip for ASP.NET 2.0 v3.5.2.0  
Telerik RadToolbar for ASP.NET 2.0 v1.5.2.0  
Telerik RadSplitter for ASP.NET 2.0 v1.2.2.1  
Telerik RadSpell for ASP.NET 2.0 v3.1.2.0  
Telerik RadRotator for ASP.NET 2.0 v2.6.2.0  
Telerik RadPanelbar for ASP.NET 2.0 v4.2.2.0  
Telerik RadMenu for ASP.NET 2.0 v4.2.2.0  
Telerik RadInput for ASP.NET 2.0 v2.0.2.0  
Telerik RadGrid for ASP.NET 2.0 v4.6.2.0  
Telerik RadEditor for ASP.NET 2.0 v7.1.2.0  
Telerik RadComboBox for ASP.NET 2.0 v2.7.2.0  
Telerik RadDock for ASP.NET 2.0 v1.8.2.0  
Telerik RadChart for ASP.NET 2.0 v3.2.1.0  
Telerik RadCalendar for ASP.NET 2.0 v2.1.2.0  
Telerik RadAjax for ASP.NET 2.0 v1.7.2.0  
telerik r.a.d.upload  
telerik r.a.d.window  
telerik r.a.d.ToolBar  
telerik r.a.d.Chart  
telerik r.a.d.combobox  
DotNetBar for VS2005 6.8.0.1  
DotnetCharting 4.3 破解DLL  
DotNET Charting WebForms  
dotnetCharting.WinForms  
TeeChart for .NET 3.2.2763.26084 完美DLL  
TeeChart for .NET 3.2.2699.17379 完美DLL  
DevExpress 7.3.4 完美破解DLL  
Dxperience 7.3.5 完美破解DLL  
DevExpress.LocalizationCHS.Dll  
NickLee.Web.UI  
SolpartWebControls  
AspNetPager 6.0 for ASP.NET 1.x 自定义分页控件  
AspNetPager 6.0 for ASP.NET 2.0 自定义分页控件  
数据操作类 Socut.Data.dll for .NET 2.0 v3.1

数据操作类 Socut.Data.dll for .NET 1.x v3.1  
Developer Express for .NET v7.3.5.0 全套完美无限制版





## ASP.NET、Ajax、Silverlight 学习电子资料汇总

### Silverlight 电子资料汇总：

O'Reilly Silverlight 1.1 简介

Wrox出版 Silverlight 1.0 (彩页染色代码、全面解析)

Silverlight 1.0 Development with JavaScript

Sams出版 Silverlight 1.0 Unleashed

O'Reilly Essential Silverlight

XAML简明教程 CHM+PDF版

### ASP.NET 1.x/2.0/3.5 电子资料汇总：

Pro ASP.NET 3.5 in C# 2008

Beginning ASP.NET 3.5 in VB 2008 从入门到精通

Wrox ASP.NET 2.0 MVP Hacks and Tips

Professional ASP.NET 2.0 Design

ASP.NET2.0 入门经典

Wrox ASP.NET 2.0 Visual Web Developer 2005 Express Edition Starter

Beginning ASP.NET 2.0 in C# 2005 From Novice to Professional

Wrox Professional ASP.NET 2.0

Wrox Professional ASP.NET 2.0 XML

Wrox Professional ASP.NET 2.0 Security Membership and Role Management

Wrox Beginning ASP.NET 2.0 and Databases

ASP.NET开发人员手册

ASP.NET 2.0 网络编程入门到精通

ASP.NET Web应用程序开发新思维

ASP.NET 2.0 高级应用程序设计专家教程

ASP.NET XML高级编程 C#编程篇

ASP.NET程序开发 C#篇

ASP.NET XML深入编程技术

ASP.NET 2.0 Cookbook

ASP.NET 2.0 Everyday Apps for Dummies

Pro ASP.NET for SQL Server

ASP.NET 从入门到精通

Wrox Beginning ASP.NET 1.1 with Visual C#.NET 2003



ASP.NET 2.0 揭秘

Build Your Own ASP.NET 2.0 Web Site Using C# and VB

开发Microsoft ASP.NET 2.0 网络应用程序

开发ASP.NET 2.0 核心参考

Building Websites with VB.NET and DotNetNuke 4

Wrox出版 Professional DotNetNuke 4.0

Professional DotNetNuke ASP.NET Portals

#### ASP.NET 视频教程系列汇总：

天蚕穿ASP.NET2.0 视频教程(全 106 讲，共七部分)

VS2005 环境下开发ASP.NET 2.0 Web应用程序视屏教程(swf)

[ASP.NET视频]Data 数据访问与操作

[ASP.NET视频]Masterpages 母版页

[ASP.NET视频]Caching 缓存机制

[ASP.NET视频]Contact页

[ASP.NET视频]ASP.NET详细功能介绍

[ASP.NET视频]Localization 本地化

[ASP.NET视频]Membership and Roles management

[ASP.NET视频]Profiles and Themes

[ASP.NET视频]Tips and Tricks

[ASP.NET视频]Web Parts和Personalization详解

#### Ajax, ASP.NET Ajax 电子资料汇总：

Ajax基础教程

Ajax宝典

Wrox Beginning Ajax

Ajax in Practice

Ajax模式最佳实践教程

Wrox Professional Rich Internet Applications AJAX and Beyond

O'Reilly Ajax on Java

Practical JavaScript DOM Scripting and Ajax Projects

Creating Web Pages with Asynchronous Javascript and XML

O'Reilly Securing Ajax Applications

Advanced Ajax Architecture and Best Practices

Beginning ASP.NET 2.0 AJAX

Introducing Microsoft ASP.NET AJAX  
Wrox Professional ASP.NET 2.0 AJAX  
ASP.NET AJAX Programmer's Reference





## Java 语言及其相关开发技术电子资料汇总

Java 编程初步 傻瓜书

数据结与算法 Java语言版

JSF JavaServer Faces in Action Manning

Wrox Professional Java JDK 6 Edition

Java 2 宝典

侯捷java编程思想 PDF中文版

Learning Java (第三版)

Beginning Java Programming for Dummies 第二版

Java 2 核心编程

Java How to Program (第六版)

Java All-In-One案头参考傻瓜书 (第二版)

21 天自学 Java 6 (2007 年 5 月更新出版) PDF

Java咖啡馆

Thinking in Java(第四版)

深入学习JFC SWING - Java基础类组件集

J2EE全实例教程

Java信息系统设计与开发实例(第二版)

Java优化编程

Java信息系统设计与开发实例(第二版)

Tricks of the Java Programming

Wrox Professional Java Native Interfaces with SWT JFace

Java Swing 第二版 PDF文字版 O'Reilly出版

O'Reilly - Java Database Programming with JDBC

JDBC与Java数据库程序设计

Learning JQurey (2007 年 7 月最新出版)

J2EE设计开发编程指南

Java Web Services简明教程

O'Reilly Java and XML (第二版PDF)

O'Reilly Java and XML (第三版PDF)

Java技术XML高级编程

Expert One-on-One J2EE Design and Development

Expert One-on-One J2EE Development without EJB

JBoss - A Developer's Notebook



The Java Programming Language (第四版)  
Spring in Action (第二版)  
Professional Java Development with the Spring Framework  
Core Java Server Faces 第二版  
精通Enterprise JavaBeans  
Enterprise JavaBeans EJB 第四版  
J2EE应用与BEA WebLogic Server (第二版PDF)  
O'Reilly Java Web Services  
Ant权威指南  
Ajax和Java框架高级编程  
Java Web Services简明教程  
Cross-Platform Web Services Using C# and Java  
O'Reilly Ajax on Java  
O'Reilly Java and XSLT  
O'Reilly Java and XML Binding  
O'Reilly Java and SOAP  
Design Patterns Java Companion  
J2EE Java黑客大曝光 开发安全的Java应用程序  
J2ME API 速查手册  
精通J2ME无线编程  
J2ME开发大全  
Java网页开发的艺术  
Java编程高手  
Java 5.0 Tiger程序高手秘笈  
Java2 网络协议技术内幕(附源码)  
Using Enterprise JavaBeans 2  
Java技术实用教程  
企业级Java安全性(构建安全的J2EE应用)  
Java语言集成开发环境Eclipse中文教程  
NetBeans IDE 5.5 企业版高级开发教程  
Using Enterprise JavaBeans 2  
Borland JBuilder Developer's Guide  
Eclipse精要与高级开发技术  
Java 6 3D游戏开发



视频教程系列：

[J2EE开发IDE Eclipse视频教程 全 9CD 完整版](#)

[Java视频教程 孙鑫Java无难事 \(全 12CD\)](#)

[J2EE高级开发视频教程第 01 讲](#)

[J2EE高级开发视频教程第 02 讲](#)

[J2EE高级开发视频教程第 03 讲](#)

[J2EE高级开发视频教程第 04 讲](#)

[J2EE高级开发视频教程第 05 讲](#)

[J2EE高级开发视频教程第 06 讲](#)

[J2EE高级开发视频教程第 07 讲](#)

[J2EE高级开发视频教程第 08 讲](#)

[J2EE高级开发视频教程第 09 讲](#)

[J2EE高级开发视频教程第 10 讲](#)

[J2EE高级开发视频教程第 11 讲](#)



## SQL 语言及各种数据库管理系统电子书汇总

### SQL 语言, 数据库基础电子资料:

[SQLite权威指南](#)

[SQL语法大全中文版](#)

[SQL语言案头完全参考手册](#)

[SQL - A Practical Introduction](#)

[O'Reilly SQL Tuning](#)

[O'Reilly The Art of SQL](#)

[数据库综合资料库](#)

[数据库设计指南](#)

[Wrox Beginning Database Design](#)

[SQL Puzzles and Answers](#)

[SQL Queries for Mere Mortals](#)

[SQL Puzzles and Answers](#)

[Apress出版 The Berkeley DB Book](#)

[数据库系统概论 浙江大学张军教授主讲\(全 32 讲\)](#)

### MS SQL Server 电子资料:

[Transact-SQL Cookbook](#)

[SQL Server 2005 宝典](#)

[Microsoft SQL Server 2005 完全参考](#)

[O'Reilly Learning SQL on SQL Server 2005](#)

[Beginning SQL Server 2005 Programming](#)

[Pro SQL Server 2005 High Availability](#)

[Beginning SQL Server 2005 Administration](#)

[SQL Server 2005 Unleashed](#)

[Pro SQL Server 2005](#)

[A Developer's Guide to SQL Server 2005](#)

[Pro T-SQL 2005 Programmer's Guide](#)

[Beginning Transact-SQL with SQL Server 2000 and 2005](#)

[SQL Server 2005 报表服务](#)

[Wrox Professional SQL Server 2005 Programming](#)

[Scaling Out SQL Server 2005 权威指南](#)



[Sql Server 2005 Performance Optimiztion and Tuning Handbood](#)

[Microsoft SQL Server 2005 编程傻瓜书](#)

[Pro SQL Server 2005 Assemblies](#)

[MS SQL Server 2005 Reporting Essentials](#)

[SQL Server 2005 工具箱内幕](#)

[SQL Server 2005 管理员手册](#)

[SQL Server 2005 工具箱内幕](#)

[SQL Server 2005 数据挖掘](#)

[Pro SQL Server 2005 Service Broker](#)

[Pro SQL Server 2005 Replication](#)

[Sql server 2005 的XML最佳实施策略](#)

[Microsoft SQL Server Black Book](#)

[MS SQL Server2000 宝典](#)

[SQL Server 2000 存储过程和XML编程](#)

[SQL Server 2005 高级数据分析视频教程系列](#)

[SQL Server 2005 盛宴系列视频 全 52 讲](#)

#### **MySQL 电子资料:**

[SQL for MySQL Developers](#)

[MySQL教程](#)

[MySQL 5 权威指南\(第三版\)](#)

[MySQL培训经典教程](#)

[MySQL Essential Skills](#)

[MySQL Administrators Guide](#)

[MySQL权威指南 中文版+英文版](#)

[MySQL 4.1.0 中文参考手册](#)

[MySQL in a Nutshell](#)

[Export MySQL](#)

[MySQL and PHP from Scratch](#)

#### **其他数据库电子资料:**

[Microsoft Access 2007 初学者指南 2007 年 6 月](#)

[Microsoft Access 2007 宝典](#)

[Microsoft Office Access 2007 VBA宝典](#)

[Wrox出版 Expert Access 2007 Programming](#)



Access 2007 窗体、报表和查询  
Microsoft Access 2007 数据分析  
Oracle Automatic Storage Management  
Pro Oracle Spatial for Oracle Database 11g  
Oracle 9i 数据库管理员指南  
Wrox Professional Oracle 8i Programming  
O'Reilly Oracle Security  
PL/SQL Study Guide  
Sybase实用教程  
PostgreSQL 对象关系数据库开发  
PostgreSQL 必备参考手册  
PostgreSQL 7 数据库开发指南  
PostgreSQL 8 for Windows 2007 年 3 月最新出版  
Crystal Reports 10 完全参考  
Crystal Reports 10 水晶报表 10 傻瓜书



HTML、CSS、JavaScript 等 Web 开发技术电子资料汇总

### CSS、HTML、xHTML

[CSS权威指南](#)

[The CSS Anthology \(第二版\) CSS设计大师设计思路与实践](#)

[HTML & XHTML 权威指南\(英文CHM版+中文PDF版\)](#)

[Building a Web Site 傻瓜书](#)

[HTML 4 傻瓜书 第五版](#)

[css禅意花园 \(高级CSS开发\)](#)

[CSS与DHTML精髓](#)

[CSS网页设计师](#)

[CSS Hacks and Filter](#)

[CSS Mastery高级Web开发标准](#)

[CSS 设计艺术](#)

[CSS代码效果对比学习](#)

[CSS Web Design傻瓜书](#)

[Pro CSS Techniques](#)

[CSS HTML高级设计模式 Pro CSS and HTML Design Patterns](#)

[Build your Own WebSite - The Right Way Using HTML and CSS](#)

[Mastering Integrated HTML and CSS](#)

[CSS Web Development从入门到精通](#)

[Web Publishing with HTML and CSS in One Hour a Day](#)

[Designing with Web Standards 网站重构 英文+中文版](#)

[Beginning HTML with CSS and XHTML](#)

[XHTML Moving toward XML](#)

[Building Smart Web 2.0 Applications](#)

[How to Do Everything With HTML](#)

[Spatial Data on the Web:Modeling and Management](#)

[AdvencED DOM Scripting Dynamic Web Design Techniques](#)

[精通XHTML程序设计高级编程](#)

[XHTML实例精解](#)

[XHTML技术内幕](#)

[完美HTML设计 - 使用CSS不用Table \(第二版\)](#)

[250 HTML and Web Design Secrets](#)



## Javascript

[Javascript and Ajax for the Web 第六版](#)

[Return on Design](#)

[Wrox出版 Silverlight 1.0 \(彩页染色代码、全面解析\)](#)

[Silverlight 1.0 Development with JavaScript](#)

[Learn Javascript 高清晰PDF珍藏版](#)

[JavaScript 宝典](#)

[JavaScript宝典\(第六版\)](#)

[JavaScript实例宝典 PDF文字版](#)

[Wrox Beginning JavaScript\(第三版\)](#)

[Professional JavaScript for Web Developers](#)

[JavaScript for breakfast](#)

[JavaScript in 10 Simple Steps or Less](#)

[JavaScript开发人员参考](#)

[JavaScript参考大全第二版](#)

[JavaScript权威指南\(第四版\)](#)

[JavaScript快速查询手册](#)

[Practical JavaScript DOM Scripting and Ajax Projects](#)

[JavaScript傻瓜书 第四版](#)

[高级Javascript 第二版](#)

[Build Your Own Database Driven Website](#)

[上百个超酷JS广告代码收集汇总 \(第一辑\)](#)

[上百个超酷JS广告代码收集汇总 \(第二辑\)](#)

[上百个超酷JS广告代码收集汇总 \(第三辑\)](#)



XML 及其相关技术电子书及视频汇总

Altova MissionKit 2008 for Enterprise Software Architects 完美版

XML宝典

Wrox Beginning XML 第四版 2007 出版

Wrox XML高级编程 2007 版

Wrox Professional XML Database

Wrox Professional XML 第二版

无废话XML

XAML简明教程 CHM+PDF版

XML傻瓜书 第四版 PDF文字版

XML宝典(第二版) PDG

XML编程从入门到精通

O'Reilly Learning XML

O'Reilly Learning XML (第二版)

XML终极教程 PDF版

XML轻松学习手册

XML Pocket Reference (第二版)

XML编程

XML in a Nutshell (第二版)

XML 21 天自学教程(第三版)

实战XML(第二版)

轻松搞定XML

XML实用大全

XML解决方案开发实务

XSLT Quickly

XML参考手册(第 4 版)

XML in Theory and Practice

Real World XML Web Services

Practical Transformation with XSLT and XPath

Navigating XML With XPath 1.0/2.0 Kick Start

O'Reilly XSLT Cookbook

XML之使用SOAP开发Web服务

XML编程 XQuery专家教程

XPath,XLink,XPointer,and XML学习手册



O'Reilly XSLT

Beginning XSLT 2.0 从入门到精通

XML学习系列之 XSL-FO权威指南

XML,XSLT,Java and JSP

SVG编程指南

XML Security

Beginning RSS and Atom Programming

O'Reilly XQuery 2007 年最新版

孙鑫xml视频教程 全三讲



## PHP 语言、Apache 相关电子书及视频下载汇总

[PHP Designer 2008 专业版+特别文件 完美版](#)

[PHP Designer 2007 专业版+特别文件 完美版](#)

[Practical Web 2.0 Applications with PHP\(Apress 2008 最新版\)](#)

[Practical Apache Struts2 Web 2.0 Projects](#)

[O'Reilly Learning PHP & MySQL 第二版](#)

[PHP 5 傻瓜书](#)

[Wrox Beginning PHP 5](#)

[PHP 5 Advanced](#)

[PHP 5 与MySQL编程初学者指南](#)

[PHP 5 和MySQL 5 从入门到精通 PDF文字版](#)

[PHP 4.1 从入门到精通](#)

[PHP技术内幕](#)

[PHP最新参考手册](#)

[PHP程序设计](#)

[PHP经典 100 例](#)

[Object Oriented PHP Concepts Techniques and Code](#)

[The PHP Anthology 第二版](#)

[PHP API使用完全指南](#)

[Wiley出版 Making Use of PHP](#)

[PHP实例教程](#)

[PHP in Action](#)

[PHP+MySQL网络开发技术](#)

[Dreamweaver CS3 with CSS, Ajax, and PHP](#)

[AJAX and PHP Building Responsive Web Applications](#)

[Beginning Ajax with PHP](#)

[PHP Programming with PEAR](#)

[PHP MySQL and Apache自学教程](#)

[PHP Apache和MySQL网页开发初步](#)

[24 小时学会使用PHP MySQL Apache](#)

[PHP MySQL 网络应用程序开发核心](#)

[Professional LAMP - Linux,Apache,MySQL and PHP 5 Web Development](#)

[Setting Up LAMP - Getting Linux Apache MySQL and PHP Working Together](#)

[Beginning PHP,Apache,MySQL Web Developmnet](#)



PHP Data Objects for MySQL

MySQL and PHP from Scratch

Extending and Embedding PHP

Wiley出版 Secure PHP Development

O'Reilly Building Tag Clouds in Perl and PHP

Wrox Professional Apache Tomcat 5

PHP MySQL编程初学者指南

Beginning PHP and Oracle (PDF文字版)

Advanced PHP for Web Professionals

Apache Server 2.0 实用指南

Apache管理员手册

Apache使用指南与实现原理

PHP专业项目实例开发 中文PDF版

PHP高级开发技术与实例 中文PDF影版

PHP 5 for Flash

O'Reilly - Tomcat权威指南

Wrox Professional Apache Tomcat 6

Pro Jakarta Tomcat 5

Foundations of PEAR - Rapid PHP Development

黑客基地PHP开发中级提高班 PHP视频教程 全 11 讲完整版

一周学会PHP编程 台湾中原大学PHP教程 第一讲

一周学会PHP编程 台湾中原大学PHP教程 第二讲

一周学会PHP编程 台湾中原大学PHP教程 第三讲

一周学会PHP编程 台湾中原大学PHP教程 第四讲

一周学会PHP编程 台湾中原大学PHP教程 第五讲



上百个 Linux、BSD、Unix 学习电子书+视频下载汇总

**Linux 学习必备系列之 Linux 各发行版本资料汇总**

[Linux宝典 2007 版](#)

[Linux宝典 2005 版](#)

[Beginning Ubuntu Linux](#)

[Ubuntu Linux宝典](#)

[Ubuntu部落](#)

[Ubuntu中文Wiki离线PDF版](#)

[Ubuntu Unleashed](#)

[Ubuntu Linux for Non-Geeks 第一版](#)

[Ubuntu Linux for Non Geeks 第二版](#)

[Moving to Ubuntu Linux](#)

[O'Reilly Ubuntu Hacks](#)

[Debian GNU Linux安装与基本配置](#)

[Debian GNU/Linux宝典](#)

[O'Reilly Learning Debian GNU/Linux](#)

[Redhat Linux 学习指南 第二版](#)

[Learning Red Hat Linux 第三版](#)

[Red Hat Linux 9 魔鬼式培训教程](#)

[Redhat Linux 9 从入门到精通](#)

[Fedora 6 and Red Hat Enterprise Linux宝典](#)

[Red Hat Linux网络管理工具](#)

[Red Hat Fedora Linux宝典](#)

[Red Hat Linux Fedora24 小时自学教程](#)

[Red Hat Linux Fedora for Dummies](#)

[Red Hat Fedora Linux 2 案头完全参考傻瓜书](#)

[红帽企业版Linux国际化支持语配置指南](#)

[红帽企业版 5.0 快速布置指南](#)

[红帽企业Linux安装指南](#)

[RedHat Linux AS 4 平台下Oracle 9](#)

[Red Hat Enterprise Linux 4 傻瓜书](#)

[Fedora 7 Unleashed](#)

[Redhat Fedora core 6 unleashed](#)



Fedora Core 5 初学者指南  
Redflag HA Cluster 4.1 完全参考  
Redflag Linux Server 4.0 用户手册  
Redflag Data Center 5.0 系统管理  
Redflag Linux Desktop 5 用户手册  
Freebsd简明教程  
FreeBSD 6 Unleashed  
FreeBSD使用大全(第二版)  
FreeBSD完全手册(第三版)  
FreeBSD Handbook PDF 中文版+英文版  
BSD FreeBSD Architecture Handbook  
Absolute BSD - The Ultimate Guide to FreeBSD  
Designing BSD Rootkits - An Introduction to Kernel Hacking  
FreeBSD 6.0 架设管理与应用  
Beginning SUSE Linux 第二版  
SuSe Linux  
SuSe Linux初学者从入门到精通  
SuSe Linux 10 宝典  
SuSe Linux 10 完全参考  
SuSe Linux 10 傻瓜书  
Suse Linux 10 新手指南  
SUSE Linux 企业服务器权威指南  
Suse linux 9.3 用户手册  
Suse linux 9.3 管理员手册  
Knoppix Hacks

### Linux 学习必备系列之 基础应用资料汇总

Linux新手管理指南  
Linux是如何工作的  
Linux简明教程 第四版  
Linux傻瓜书 第六版  
Linux All-In-One Desk Reference for Dummies 2006 版  
Linux All-In-One Desk Reference for Dummies 2005 版  
Learning the Vi Editor (第六版)  
GNU Emacs 参考手册



[Linux Cookbook](#)

[Linux for Non-Geeks](#)

[Linux文件查找命令find、xargs详述](#)

[Linux Desktop Hacks](#)

[LINUX 24 学时教程](#)

[Linux案头参考\(第二版\)](#)

[O'Reilly Linux简明教程 第五版](#)

[Linux系统管理白皮书](#)

[Linux系统一本通](#)

[Linux实用培训学习教程](#)

[Linux命令参考大全](#)

[送给初学Linux的穷人Linux系统指令大全](#)

[Linux命令完全参考](#)

[Linux命令字典](#)

[Linux Complete Command Reference](#)

[Linux故障排除宝典](#)

[Linux桌面系统提速法宝](#)

[从Windows转向Linux基础教程](#)

[让Linux像Windows一样方便](#)

[Learning the UNIX Operating System 第四版](#)

[UNIX和Linux权威教程 \(第三版\)](#)

[Unix完全参考\(第二版\)](#)

[Unix傻瓜书](#)

[Unix简明教程 第四版](#)

[Unix教程网络篇](#)

[UNIX for OpenVMS Users 第三版](#)

[Linux网络管理员手册](#)

**Linux 学习必备系列之 高级应用资料汇总**

[A Beginner's Guide to LVM](#)

[Novell出版 Linux防火墙 第三版 chm格式](#)

[No Starch出版 Linux Firewalls](#)

[2 小时玩转iptables企业版](#)

[Securing Optimizing Linux The Hacks Solution](#)

[SUSE Linux 企业服务器权威指南](#)



Red Hat Linux网络管理工具

Linux进程管理教程

Linux下安装Oracle完全参考

RedHat Linux AS 4 平台下Oracle 9

Redflag HA Cluster 4.1 完全参考

Hack Proofing Linux

Linux Server Hacks

Understanding Linux Network Internals

Linux Power Tools

Linux On The Mainframe

Linux Network Servers

O'Reilly Building Embedded Linux Systems

Advanced Linux Networking

Linux Security Cookbook

保护Linux系统 - Linux安全生存指南

SELinux NSAs - Open Source Security Enhanced Linux

Hardening Linux

Building Secure Servers With Linux

详细剖析Linux和Unix两系统病毒威胁

Linux黑客大曝光 - Linux安全机密与解决方案

使用Ipfilter建立FreeBSD加固防火墙

Mastering FreeBSD and OpenBSD Security

User Mode Linux

Linux Appliance Design

Linux Device Drivers

Multitool Linux Practical Uses for Open Source Software

高性能Linux集群

Linux网络构架设计与实现

Building Embedded Linux Systems

Unix Linux管理自动化

Linux Power Tools

Building Applications with the Linux Standard Base

Hacking Linux Exposed

Building Secure Servers with Linux

SSH - Unix Secure Shell tool



[UNIX 系统安全工具](#)

[Unix for Oracle DBAs Pocket Reference](#)

[Unix 网络安全实用教程](#)

[Unix备份与恢复全攻略](#)

[UNIX Power Tools \(第三版\)](#)

[Practical Unix and Internet Security \(第三版\)](#)

## **Linux 学习必备系列之 Linux 环境编程资料汇总**

[Understanding The Linux Kernel 第一版](#)

[Understanding The Linux Kernel 第二版](#)

[Understanding The Linux Kernel 第三版](#)

[Linux内核精要](#)

[Understanding the Linux Kernel - 理解Linux内核](#)

[Linux内核源代码情景分析 中文版 \(上下册\)](#)

[O'Reilly Bash Cookbook](#)

[101 个超酷Shell脚本](#)

[Bash快速参考](#)

[Bash Beginners Guide](#)

[Perl入门及高级编程](#)

[Perl语言编程](#)

[Perl指南](#)

[Perl编程思想](#)

[O'Reilly 精通Perl编程](#)

[O'Reilly Perl and XML](#)

[Perl 5 21 天自学教程](#)

[轻松学习Linux编程](#)

[Linux应用开发基础](#)

[Linux 网络编程](#)

[Python简明教程](#)

[Unix编程艺术 The Art of Unix Programming](#)

[Linux编程白皮书](#)

[A Practical Guide to Linux Commands Editors and Shell Programming](#)

[Linux案头参考\(第二版\)](#)

[Linux与Unix Shell编程指南](#)

[Linux Shell Scripting with Bash](#)



Unix Shell Programming(第三版)  
Linux Debugging And Performance Tuning  
Linux系统分析与高级编程技术  
十分钟Unix自学教程 第二版  
Korn Shell: Unix and Linux Programming Manual  
Unix shell范例教程 (第四版)  
Unix环境高级编程  
Unix环境高级编程 第二版  
Unix Systems Programming  
Linux编程从入门到精通  
Linux实例编程  
Linux环境编程 GCC完全参考  
Linux应用程序开发指南 使用Gtk+ Gnome库  
Linux C高级程序员指南  
Sams Mono Kick Start - Linux环境的.NET编程

### Linux 学习必备系列之 Linux 视频教程系列

Redhat认证 RHCE视频教程 全七部分 高清AVI格式  
楚广明 24 小时学通Linux 第一讲 基础安装与GNU历史  
楚广明 24 小时学通Linux 第二讲 基本概念与命令  
楚广明 24 小时学通Linux 第三讲 系统设置  
楚广明 24 小时学通Linux 第四讲  
楚广明 24 小时学通Linux 第五讲 DNS服务器  
楚广明 24 小时学通Linux 第六讲  
楚广明 24 小时学通Linux 第七讲 Apache与Tomcat整合实验  
SUSE Linux Enterprise 10 精妙解决方案视频教程  
FreeBSD 6.2 服务器架设视频教程 (全五部分)  
高效架设Redhat Linux服务器全过程视频教程  
楚广明主讲 FreeBSD视频教程 swf版  
红旗Linux安装全过程视频教程  
debian etch安装全程攻略视频  
Arch Linux安装全程攻略视频  
Linux基础教程视频(全四讲)  
Linux从入门到精通视频教程(swf版) 基础安装配置  
Linux从入门到精通视频教程(swf版) 服务器配置



- Linux从入门到精通视频教程(swf版) 安全配置
- Linux下访问NTFS分区配置全过程视频教程
- 黑客基地Linux视频教程系列之linux简介
- 黑客基地Linux视频教程系列之文件系统
- 黑客基地Linux视频教程系列之在字符界面下安装
- 黑客基地Linux视频教程系列之图形界面安装linux并配置
- 黑客基地Linux视频教程系列之体验linux单操作系统的安装
- 黑客基地Linux视频教程系列之linux与windows双系统安装
- 黑客基地Linux视频教程系列之GRUB的配置方法
- 黑客基地Linux视频教程系列之GRUB相关问题解决方法
- 黑客基地Linux视频教程系列之桌面简介与网络配置
- 黑客基地Linux视频教程系列之VI编辑器使用
- 黑客基地Linux视频教程系列之linux中的用户管理
- 黑客基地Linux视频教程系列之文件系统常用命令
- 黑客基地Linux视频教程系列之网络基础配置
- 黑客基地Linux视频教程系列之软件包安装使用
- 黑客基地Linux视频教程系列之目录文件的操作命令
- 黑客基地Linux视频教程系列之linux下的日志文件
- 黑客基地Linux视频教程系列之linux中的进程管理
- 黑客基地Linux视频教程系列之linux下压缩文件使用教程
- 黑客基地Linux视频教程系列之shell编程简介
- 黑客基地Linux视频教程系列之DHCP服务器
- 黑客基地Linux视频教程系列之samba服务器(二讲)
- 黑客基地Linux视频教程系列之DNS服务器设置详解(三讲)
- 黑客基地Linux视频教程系列之linux下架设apache服务器(四讲)
- 黑客基地Linux视频教程系列之linux下简单的网络安全
- 黑客基地Linux视频教程系列之iptables防火墙简单使用
- 黑客基地Linux视频教程系列之linux内核与总结
- 边学边用linux视频教程 第 01 讲(全 24 讲)
- 边学边用linux视频教程 第 02 讲(全 24 讲)
- 边学边用linux视频教程 第 03 讲(全 24 讲)
- 边学边用linux视频教程 第 04 讲(全 24 讲)
- 边学边用linux视频教程 第 05 讲(全 24 讲)
- 边学边用linux视频教程 第 06 讲(全 24 讲)
- 边学边用linux视频教程 第 07 讲(全 24 讲)



边学边用linux视频教程 第 08 讲(全 24 讲)  
边学边用linux视频教程 第 09 讲(全 24 讲)  
边学边用linux视频教程 第 10 讲(全 24 讲)  
边学边用linux视频教程 第 11 讲(全 24 讲)  
边学边用linux视频教程 第 12 讲(全 24 讲)  
边学边用linux视频教程 第 13 讲(全 24 讲)  
边学边用linux视频教程 第 14 讲(全 24 讲)  
边学边用linux视频教程 第 15 讲(全 24 讲)  
边学边用linux视频教程 第 16 讲(全 24 讲)  
边学边用linux视频教程 第 17 讲(全 24 讲)  
边学边用linux视频教程 第 18 讲(全 24 讲)  
边学边用linux视频教程 第 19 讲(全 24 讲)  
边学边用linux视频教程 第 20 讲(全 24 讲)  
边学边用linux视频教程 第 21 讲(全 24 讲)  
边学边用linux视频教程 第 22 讲(全 24 讲)  
边学边用linux视频教程 第 23 讲(全 24 讲)  
边学边用linux视频教程 第 24 讲(全 24 讲)



## 本书Linux指令的功能分类如下

- ❖ 系统管理
- ❖ 系统设置
- ❖ 文本编辑
- ❖ 文件传输
- ❖ 文件管理
- ❖ 备份压缩
- ❖ 网络通信
- ❖ 磁盘管理
- ❖ 磁盘维护
- ❖ 程序编译
- ❖ 打印作业
- ❖ 电子邮件新闻组
- ❖ X Windows System
- ❖ 格式转换
- ❖ 系统软件工具

给出了每个指令的功能、语法、参数、执行范例和相关指令。

ISBN 978-7-115-16895-5



9 787115 168955 >

ISBN 978-7-115-16895-5/TP

定价：45.00 元

【装帧设计：董志桢】

分类建议：计算机/操作系统/Linux

人民邮电出版社网址：[www.ptpress.com.cn](http://www.ptpress.com.cn)