第 6 章　Linux 文件和目录管理

文件和目录管理是使用操作系统过程中经常涉及的基本工作。本节将系统地介绍 Linux 文件系统的组织结构、文件系统的权限管理，并对 Red Hat Enterprise Linux 5 下如何完成对文件和目录的操做进行详细的介绍。

6.1　文件系统的概念

文件系统(File System)是操作系统用来存储和管理文件的方法。从系统角度来看，文件系统对文件存储空间进行组织、分配，并对文件的存储进行保护和检查。从用户角度来看，文件系统可以帮助用户建立文件，并对文件的读、写和删除操作提供保护和控制。

6.2　Linux 文件系统的组织方式

不同的操作系统对文件的组织方式不同，其所支持的文件系统数量和种类也不一定相同。Linux 文件系统的组织方式称做 Filesystem Hierarchy Standard（文件系统分层标准，简称 FHS），即采用层次式的树状目录结构。在此结构的最上层是根目录“/”(斜杠)，然后在此根目录下创建其他的目录和子目录，如图 6.1 所示。

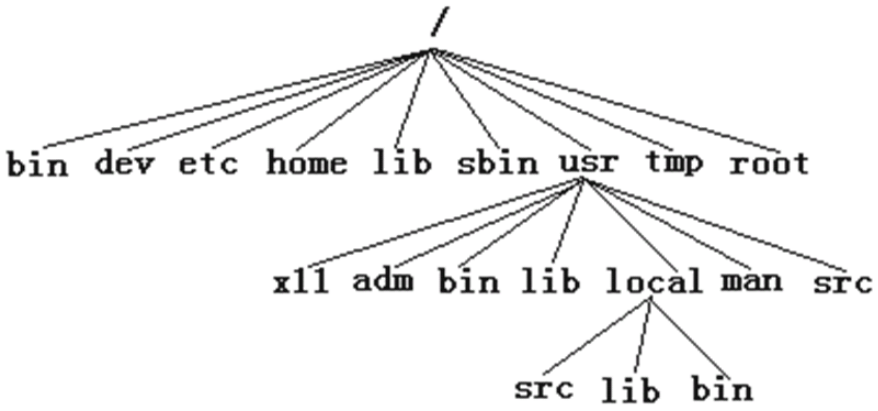

图 6.1 Linux 文件系统目录层次结构

Linux 与 DOS 及 Windows 一样，采用“路径”来表示文件或目录在文件系统中所处的层次。路径是由以“/”为分隔符的多个目录名字符串组成，分为绝对路径和相对路径。所谓绝对路径是指由根目录“/”为起点来表示系统中某个文件或目录的位置的方法。例如如果用绝对路径表示图 6.1 中第 4 层目录中的 bin 目录，应为“/usr/local/bin”。相对路径则是以当前目录为起点，表示系统中某个文件或目录在文件系统中的位置的方法。若当前工作目录是“/usr”，则用相对路径表示图 6.1 中第 4 层目录中的 bin 目录，应为“local/bin”或“./local/bin”，其中“./”表示当前目录，通常可以省略。

Linux 文件系统的组织与 Windows 操作系统不同。对于在 Linux 下使用的设备，不需要像 Windows 那样创建驱动器盘符，Linux 会将包括本地磁盘、网络文件系统、CD-ROM、U 盘等所有设备识别为设备文件，并嵌入到 Linux 文件系统中来进行管理。一个设备文件不占用文件系统的任何空间，仅仅是访问某个设备驱动程序的入口。Linux 系统中有两类特殊文件：面向字符的特殊文件和面向块(block)的特殊文件。前者允许 I/O 操作以字符的形式进行，而后者通过内存缓冲区来使数据的读写操作以数据块的方式实现。当对设备文件进行 I/O 操作时，该操作会被转给相应的设备驱动程序。一个设备文件是用主设备号（指出设备类型）和从设备号（指出是该类型中的第几个设备）来表示，可以通过 mknod 命令进行创建。软盘、光盘、硬盘在 Linux 系统中的表示方法见表 6-1。

表 6-1 典型设备文件在Linux系统中的表示方法

设备名	Linux系统中的表示方法
第一个IDE接口的Master硬盘	/dev/hda
第一个IDE接口的Slave硬盘	/dev/hdb
第二个IDE接口的Master硬盘	/dev/hdc
第二个IDE接口的Slave硬盘	/dev/hdd
第一个SCSI接口的Master硬盘	/dev/sda
第一个SCSI接口的Slave硬盘	/dev/sdb
第二个SCSI接口的Master硬盘	/dev/sdc
第二个SCSI接口的Slave硬盘	/dev/sdd
光驱	/dev/cdrom
第一个软盘	/dev/fd0

Linux 文件名称最长可允许 256 个字符，可以包括数字、字符以及“.”、“-”、“_”等符号。Linux 文件名不像 DOS 或 Windows 由主文件名和扩展文件名两部分组成，Linux 文件名没有扩展名的概念。Linux 环境下，文件名大小写敏感（Case Sensitive），例如 test.txt 与 Test.txt 是会被识别成两个不同的文件，而 DOS 或 Windows 平台是不进行大小写区分的。

6.3　Linux 系统的默认安装目录

按着 FHS 的要求（关于 FHS 的详细信息可以登录 http://www.pahtname.com/fhs/查询），Linux 系统在安装过程中会创建一些默认的目录。这些默认的目录都有其特殊的功能，不可随便将其更名，以免造成系统的错误。表 6-2 列出了这些目录及功能说明。

表 6-2 Linux默认目录及功能说明

目录名称	说　　明
/	Linux文件系统的最上层根目录，其他所有目录均是该目录的子目录。
/bin	Binary的缩写，存放用户的可执行程序，例如cp和mv等，也存放Shell，如bash和csh。不应把该目录放到一个单独的分区中，否则Linux Rescue模式无法使用这些命令。
/boot	操作系统启动时所需的文件，包括vmlinuz和initrd.img，这些文件若是损坏常会导致系统无法正常启动，因此最好不要做任意改动。
/dev	设备文件目录，例如/dev/sda表示第一块SCSI设备，/dev/hda 表示第一块IDE设备。
/etc	有关系统设置与管理的文件，包括密码、守护程序以及X-Window相关的配置。可以通过编辑器（如VI、gedit等）打开并编辑相关的配置文件。
/etc/X11	X-Window System的配置目录。
/home	普通用户的主目录或FTP站点目录，一般存放在/home目录下。
/lib	存放共享函数库(1ibrary)。
/mnt	文件系统挂载点(Mount)，例如光盘的挂载点可以是/mnt/cdrom，软盘的挂载点可以是/mnt/floppy，Zip

	驱动器为/mnt/zip。
/opt	该目录通常提供给第三方较大型的应用程序使用，例如Sun Staroffice、Corel WordPerfect，这可避免将文件分散至整个文件系统。
/proc	保存目前系统内核与程序执行的相关信息，和利用ps命令看到的内容相同。例如/proc/interrupts文件保存了当前分配的中断请求端口号，/proc/cpuinfo保存了当前处理器信息。
/root	根用户的主目录。
/sbin	System Binary的缩写。此目录存放的是系统启动时所需执行的系统程序。
/tmp	temporary的缩写，用来存放临时文件的目录。
/usr	存放用户使用的系统命令和应用程序。
/usr/bin	存放用户可执行的程序，例如OpenOffice的可执行程序。
/usr/doc	存放各种文档的目录。
/usr/include	存放C言语用到的头文件。
/usr/include/X11	存放X-Window程序使用的头文件。
/usr/info	存放GNU文档的目录。
/usr/lib	函数库
/usr/lib/X11	X-Window的函数库。
/usr/local	提供自动安装的应用程序位置。
/usr/man	存放在线手册的目录。
/usr/sbin	存放用户经常使用的程序。
/usr/src	保存程序的源文件的目录，一般系统内核源码存放在/usr/src/linux目录下。
/usr/X11R6/bin	存放X-Window的可执行程序。
/var	Variable的缩写，存放包括日志、邮件等经常变化的文件。由于/var目录的大小经常变动，为了防止失去控制而侵占其他目录所需要的空间，建议将/var安装到一个独立的分区上。

6.4　Linux 文件系统的类型

Linux 是一种兼容性很高的操作系统，除了能够挂载各种类型的设备，还可以把其他各种文件系统挂载到 Linux 系统上。在文件/proc/filesystem 文件中列出了系统当前可用的文件系统类型，其中不仅包括 UNIX 支持的各种文件系统类型，也包括 Windows 9x/NT/2000/XP 文件系统。对于普通用户而言，这些功能最普通的意义是允许用户使用软盘、U 盘、CD-ROM 内的文件。

为了查看系统当前可用的文件系统类型，可以使用“cat”命令，如下所示：

```
#cat   /proc/filesystems
```

Linux 所支持的文件系统类型包括：

- adfs：acron 磁盘文件系统，是在 RiscOS 操作系统中使用的标准文件系统。
- befs：BeOS 操作系统使用的文件系统。
- cifs：通用 Internet 文件系统(Commnn Intemet File System，简写 CIFS)，用于访问符合 SNIA CIFS 标准的服务器。CIFS 对 SMB 协议进行改进和标准化（SMB 协议可用于在 Linux 和 Windows 之间共享文件），是一种是虚拟文件系统。
- ext：ext 文件系统的第一个版本，现在已经很少使用。
- ext2：ext2 是专门为 Linux 系统设计的，在 Red Hat Linux 7. 2 版本之前是 Linux 缺省文件系统类型，具有速度快和 CPU 占用率低等特点，既可以用于标准的块设备，也可以应用到移动存储介质上。ext2 不包括日志功能。
- ext3：ext3 文件系统是 Linux 中最常用的文件系统，是 Red Hat Linux 7.2 的新特性，也被称为第 3 次扩展(Third Extented)的文件系统。现在 ext3 已经成为许多 Linux 系统的默认文件系统类型(包括 Fedora 和 RHEL)。在 ext2 和 ext3 之间可以方便地进行转换，转换前不需重新格式化文件系

统。与 ext2 文件系统相比，ext3 包含了日志功能。日志功能维护了最近更改的源数据（源数据是指和文件有关的信息，包括权限、所有者、创建时间、访问时间等）的记录，如果源数据由于非法关机等原因遭到破坏，文件系统将不能正常工作。通过 ext3 的日志系统，可以进行适当的恢复。此外，ext3 的日志功能可使硬盘读写头的移动达到最佳化。

- is09660：从 High Sierra (CD-ROM 使用的最初标准)发展而来的文件系统，是 CD-ROM 的标准文件系统。
- kafs： AFS 客户端文件系统，用于分布式计算环境，可与 Linux、Windows 和 Macintosh 客户端共享文件。
- minix：Minix 文件系统类型，最初用于 UNIX 的 Minix 版本，只支持长度为 30 个字符以下的文件名。
- msdos：MS-DOS 文件系统。DOS、Windows 和 OS/2 使用该文件系统，不支持长文件名，主要用于挂载 Microsoft 操作系统生成的软盘。
- vfat：Microsoft 扩展 FAT(VFAT)文件系统，支持长文件名，被 Windows 9x/2000/xp 使用。
- umsdos：扩展的 MS-DOS 文件系统，不仅支持长文件名，还保持了对 UID/GID、POSIX 权限和特殊文件（如管道、设备）的兼容。
- proc：proc 是一个基于内存的伪文件系统，不占用外存空间，只是以文件的方式为访问 Linux 内核数据提供接口。由于 proc 文件系统是虚拟的，因此无需挂载。用户和应用程序可以通过/proc 得到系统的运行信息，并可以改变内核的某些参数。许多应用程序和工具依靠 proc 来访问 Linux 内核信息。
- reiser：Reiserfs 日志文件系统。
- swap：用于交换(swap)分区。交换分区是系统虚拟内存的一部分，用于在当前内存不足时暂时保存数据。数据被交换到交换分区，当再次需要时调回内存。
- nfs：网络文件系统(Network File System，简写 NFS)类型，详情见第 15 章。
- hpfs：该文件系统用于只读挂载 OS/2 HPFS 文件系统。
- ncpfs：Novell Netware 文件系统，可以通过网络挂载。
- affs：Amiga 计算机使用的文件系统。
- ufs：Sun Microsystems 操作系统(即 Solaris 和 SunOS)。
- xfs：一种在高性能环境中很有用的日志文件系统，支持完整的 64 位寻址。
- jfs： JFS 主要适合于企业系统，是为大文件系统和高性能环境而设计的。
- xiafs：与 minux 文件系统相比，这种文件系统支持长文件名和更大的 i 结点。
- coherent：System V 使用的文件系统类型。
- smb：支持 SMB 协议的网络文件系统，可用于实现 Linux 与 Windows 系统的文件共享，详情见第 16 章。

6.5　使用 fstab 文件挂载文件系统

在 Linux 系统中与文件系统密切相关的配置文件是/etc/fstab。该文件列出了系统开机启动时自动加载的文件系统类型、安装点及可选参数。/etc/fstab 文件在系统安装完毕后会自动建立，也可用编辑器进行手动修改。以下是/etc/fstab 文件的实例：

```
# more /etc/fstab
```

```
/dev/VolGroup00/LogVol00   /                       ext3     defaults                1 1
LABEL=/boot                /boot                   ext3     defaults                1 2
devpts                     /dev/pts                devpts   gid=5,mode=620          0 0
tmpfs                      /dev/shm                tmpfs    defaults                0 0
proc                       /proc                   proc     defaults                0 0
sysfs                      /sys                    sysfs    defaults                0 0
fd0                        /media/floppy           auto     noauto,owner            0 0
hda1                       /mnt/win                vfat     noauto                  0 0
hdc                        /mnt/cdrom              auto     noauto,user,exec,ro     0 0
/dev/VolGroup00/LogVol01 swap
```

其中每一行是一个文件系统的配置信息，配置项用空格分隔，各列含义如下。

- 第 1 列代表被加载的文件系统、块设备或网络上的文件系统的设备名。
- 第 2 列是文件系统的挂载点，交换分区不存在挂载点。
- 第 3 列是文件系统类型。合法的文件系统类型在上一节中已经介绍。
- 第 4 列是该文件系统的加载参数，各选项之间用逗号分隔。其中 “noauto”选项指定不在启动时挂载该文件系统，“ro”是 readonly 的缩写，表示文件系统为只读(对于 CD-ROM 驱动器)，不需要写权限。“default”表示该文件系统为可读写、允许 set-UID 和 set-GID、允许程序执行（exec）、允许异步 IO 模式。关于其他支持选项的信息见表 6-3。通常，只允许 root 用户使用 mount 命令挂载文件系统。如果希望允许所有用户挂载文件系统，可以添加 user 和 owner 选项。
- 第 5 列指定文件系统是否需要备份。“1”表示文件系统需要备份，“0”表示该文件系统不需要备份。
- 第 6 列指定文件系统启动时是否需要使用 fsck 进行检查。“1”表示该文件系统需要检查，“0”表示该文件系统不需要检查。

表 6-3 fstab中常用选项说明

选项	说明
async	异步方式执行该文件系统的输入输出操作。
user	普通用户允许执行加载操作。
atime	每次存取操作都自动更新inode的时间，取消该选项使用noatime。
sync	同步方式执行该文件系统的输入输出操作。
suid	启动set-UID和set-GID选项。
auto	自动加载。
noauto	不自动加载，启动系统后可使用mount命令手动加载。
defaults	使用默认的选项。
rw	可读写模式加载。
ro	只读模式加载。
remount	进行重新加载。
nosuid	关闭set-UID和set-GID选项。
nouser	普通用户无法加载该文件系统。
exec	可执行二进制程序文件。
dev	可解读文件系统上的字符或区块设备。

文件系统/proc 、/sys 、/dev/shm 、/dev/pts 并不与特定的设备相关联。除了第 4 个字段设置为“noauto”的记录行以外，该文件中列出的所有文件系统都会在系统启动时自动挂载。在本例中，软盘(/dev/fd0)和 CD-ROM 驱动器(/dev/hdc)由于在参数行设定了“noauto”，故未在启动时被挂载。此外，/dev/hdal 用于挂载计算机上的 Windows 分区，这样就可以在 Linux 系统中访问 Windows 分区的内容。

本例中第 1 行和最后一行显示了将根(/)分区和交换分区设置为 LVM (Logical Volume Management，逻辑卷管理)。

6.6　LVM——逻辑卷管理

LVM（Logic Volume Management）为磁盘空间的调整带来了极大的方便，以往在安装 Red Hat Enterprise Linux 之前，一项重要的工作就是对驱动器空间的划分进行决策，因为一旦划分好就很难重新进行调整。LVM 允许在多个不同的文件系统之间重新分配大块的磁盘空间。LVM 可以包含多个硬盘分区，如果一个卷耗尽了空间，可以简单地附加上一个新物理卷(例如，一个硬盘分区)来扩展它的容量，而不必重新调整分区大小或创建一个新分区。与 LVM 相关的定义如表 6-4 所示：

表 6-4 LVM相关定义

定义	说明
物理卷（PV）	物理卷（Physical volume，PV）相当于硬盘驱动器的一个标准主分区或逻辑分区
物理盘区(PE)	物理盘区（Physical extent，PE）就是一块磁盘空间。物理卷被划分成多个同等大小的PE。
逻辑卷(LV)	逻辑卷（Logical volume，LV）是LE的集合。
逻辑盘区(LE)	在LVM系统中，一个逻辑盘区(Logical extent，PE)和一个PE相对应，大小相同。
卷组(VG)	卷组（Volume group，VG）是LV的集合

1. pvdisplay 命令

可以使用 pvdisplay 命令来查看组成 LVM 卷的物理卷（PV）。例如：

```
# pvdisplay
   --- Physical volume ---
   PV Name                /dev/sda2
   VG Name                VolGroup00
   PV Size                7.90 GB / not usable 23.41 MB        //容量 7.9G,]剩余 23.41MB
   Allocatable            yes (but full)
   PE Size (KByte)        32768
   Total PE               252
   Free PE                0
   Allocated PE           252
   PV UUID                4BrhEz-mAgc-90Qm-2LBj-i8Rg-H3pI-FE3k02
```

2. pvcreate 命令

创建物理卷可以使用 pvcreate 命令。例如在二级 SCSI 控制器的从属驱动器上创建一个新的物理卷，运行下面的命令：

```
#pvcreate /dev/sdd
```

也可以把 PV 安装到分区上。如果已经创建了分区（例如通过 fdisk 创建了 Linux 分区），只需将分区的系统 ID 改为“8e”，例如：

```
#fdisk /sdd1
Comand (m for more):t
Partition mumber:4
Hex code (type L to list codes):8e
```

然后通过 pvcreate 命令创建：

```
#pvcreate   /dev/sdd1
```

3. vgcreate 命令

可以通过 vgcreate 命令来创建一个卷组。卷组是由一个或多个硬盘驱动器上的多个物理卷（PV）所构成的一个集合。可以把现有的 PV 创建到一个 VG 上，当添加更多的 PV 时，可以直接添加到 VG 上。例如创建一个名为 myvolume 的 VG：

```
#vgcreate myvolume   /dev/sdd1   /dev/sdc2
```

4. vgextend 命令

在 VG 里添加一个新的 PV 可以使用下面的命令：

```
#vgextend myvolume   /dev/sde1
```

通过创建逻辑卷就可以最终完成磁盘空间的添加。首先需要查看逻辑卷中的一个 PE 有多大，可以使用 vgdisplay 命令实现：

```
# vgdisplay VolGroup00
  --- Volume group ---
  VG Name               VolGroup00
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  3
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                2
  Open LV               2
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               7.88 GB          //目前 VG 的的大小为 7.88G
  PE Size               32.00 MB         //一个 PE 大小为 32M
  Total PE              252
  Alloc PE / Size       252 / 7.88 GB
  Free  PE / Size       0 / 0
  VG UUID               34uiUt-Z2T6-NGMZ-EqGC-c67B-MGlY-IuoeVc
```

5. lvcreate 命令

由于一个 PE 的大小为 32M，使用 lvcreate 命令来创建一个所需的 LV。例如需要创建的名为 mylocigcl 的 LV，容量为 1.6G＝32M×50，命令行如下：

```
#lvcreate -l 50 myvolume –n mylogical
```

这样就创建了一个/dev/myvolume/mylogical 的新设备。可以像对待其他硬盘驱动器一样进行格式化、和挂载。例如，把 mylogical 格式化为 ext3 文件系统，并安装到/mnt/mynewdisk 目录上：

```
#mkfs -j   /dev/myvolume/mylogical
#mount -t ext3   /dev/myvolume/mylogical   /mnt/mynewdisk
```

如果有多余的 PE，可以使用下面的命令对/dev/myvolume/mylogical 的容量进行添加：

```
#lvextend -L 2000M /dev/myvolume/mylogical
```

6.7　Linux 文件系统的组成

在 Red Hat Enterprise Linux 5 中，系统默认安装的是 ext3 文件系统。ext3 文件系统将磁盘分为 4 个部分，如图 6.2 所示。块 0 称为引导块，包含系统启动程序的磁盘区块；块 1 称为超级块，主要是用来记录文件系统的配置方式，其中包括 i-node 数量、磁盘区块数量、未使用的磁盘区块以及 i 节点表、空闲块表在磁盘中存放的位置。由于超级块保存了极为重要的文件信息，因此系统将超级块冗余保存。系

统在使用 fsck 等命令修复处于严重瘫痪状态的文件系统时，实际上就是对超级块进行恢复操作。从块 2 开始是 i 节点表，i 节点（i-node 是 Index-Node 的缩写）表中记录的信息很多，包括文件大小、用户 UID、用户组 GID、文件存取模式（包括读、写或执行）、链接数目（文件每创建一个链接，链接计数加 1，每删除一个链接，文件计数减 1）、文件最后修改时间、磁盘区块地址、间接区块等。i 节点表之后的数据存储块用于存放文件内容。ext3 文件系统结构如下所示：

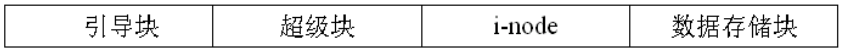

图 6.2 exte3 文件系统磁盘划分

文件有逻辑结构和物理结构两种不同的组织方式。逻辑结构是面向用户的，是用户可以看到的表示文件内容的字符流。例如使用编辑命令 vi 或显示命令 cat 所看到的文件内容。物理结构是文件在磁盘上的存储组织方式，涉及到具体的存放磁盘区块。用户所看到的文件内容是连续的，但实际上文件可能并不是以连续的方式存放在磁盘上的。事实上大于一块（block）的文件将分散地存放在磁盘上。然而当用户存取某文件时，Linux 文件系统会以正确的顺序读取此文件全部块，并将文件的逻辑结构提供给用户。设计 i 节点表实际上就是为了帮助实现这种物理结构到逻辑结构的转换。

i 节点是一个 64 字节长的表，表中包含了文件的相关信息和磁盘地址表。在磁盘地址表中有 13 个块号。前 10 个块号是文件前 10 块的存放地址，文件将根据块号在磁盘地址表中出现的顺序依次读取相应的块。当文件长度大于 10 个块，将会用到后面的 3 块所给出的间接地址块。

磁盘地址表中的第 11 块给出了扩展块的位置，这个块号指出的块中含有 256 个块号，加上前 10 块，文件的长度为 266 块(272,384 字节)。如果文件大于 266 块，再启用磁盘地址表的第 12 块所定义的扩展块。当文件的大小大于 266＋256*256 块时，再启用第 13 个扩展块。这时文件的大小所占的块数为 266+256*256+256*256*256。

文件系统采用了一对一映射的方法来实现文件名到 i 节点的转换。目录实际上是一个含有目录表的文件，对于目录中的每个文件，在目录表中都会有一个入口项, 入口项中含有文件名和与文件相应的 i 节点号。当用户键入“more ABC.txt”时，文件系统就在当前目录表中查找名为“ABC.txt”的表项，由此得到与文件 ABC.txt 相应的 i 节点号，然后开始读取含有该文件内容的数据块。

可以通过 df 命令来查看系统中的 i-node 的大小与数量。例如：

```
# df  -i
文件系统                    Inode (I)    已用    (I)可用    (I)已用% 挂载点
/dev/mapper/VolGroup00-LogVol00
                          1918208   157550 1760658     9%          /
/dev/sda1                   26104       35    26069     1%          /boot
tmpfs                       31737        1    31736     1%          /dev/shm
```

其中/sda1 文件系统，挂载点是/boot，总共有 26104 个 i-node，已经使用 35 个，剩余 26069 个，已用 i-node 占总数的 1%。

若要查看文件的 i-node 编号，可以使用命令 ls。例如：

```
# ls  -il
总计 162
 650241 drwxr-xr-x   2 root root  4096 07-20 03:03 bin
      2 drwxr-xr-x   4 root root  1024 07-15 19:59 boot
    657 drwxr-xr-x  14 root root  3940 07-25 05:45 dev
 292609 drwxr-xr-x 103 root root 12288 07-25 05:44 etc
… …
```

可以看到，目录也是一种特殊的文件，其中/bin 目录的 i-node 号为 650241。

6.8　创建 Linux 文件系统

在创建文件系统之前需要对磁盘空间进行分区处理。Linux 提供了多种分区创建方案，见表 6-5。

表 6-5 Linux可能的分区方案

分区方案	说明
根分区、swap分区	适用于磁盘空间有限的计算机。
根分区、/boot分区、swap分区	较大磁盘空间的典型配置，也是Red Hat Enterprise Linux 默认配置。
根分区、/boot分区、/var分区、swap分区	可以避免日志文件大小失控。
根分区、/boot分区、/home分区、swap分区	对于一台为许多用户提供服务的计算机，可以帮助控制用户占用的空间量。

在安装 Red Hat Enterprise Linux 5 过程中，系统默认把根（/）和/boot 目录安装到独立的分区上。

fdisk 是一种功能强大的磁盘分区工具，其详细使用方法见第 5 章磁盘管理。

在利用 fdisk 完成磁盘分区后，需要使用 mkfs 命令将分区进行格式化。mkfs 命令的格式如下：

```
mkfs [ -t 文件系统类型] [选项] 文件系统
```

- -t 文件系统类型：文件系统的类型。如果没有指定默认为 ext2。
- -c：在建立文件系统之前检查设备是否有物理坏块。
- -l filename：从文件中读出坏块列表。

例如对第一个 IDE 接口的主硬盘的第 4 个分区进行格式化，并创建 ext3 文件系统，命令行如下：

```
#mkfs  -t  ext3 /dev/hda4
```

例如使用 mkfs 命令在软盘上创建文件系统：

```
#mkfs  -t  ext2 /dev/fd0
```

必须对所创建的磁盘分区进行挂载才能正常使用：

```
#mkdir  /mnt/disk4                  //创建挂载点
#mount  /dev/hda4  /mnt/disk4
#mkdir  /mnt/floppy                 //创建挂载点
#mount  /dev/fd0  /mnt/floppy
```

如果希望系统启动时该分区能够自动加载，需要编辑/etc/fstab 文件，加入如下的记录项：

```
hda4  /mnt/disk4  ext3  defaults 1 2
```

6.9　Linux 文件的类型

在 Linux 中文件的类型是通过文件权限的首位定义的，可以分为以下几类：

1. 普通文件

普通文件包括源程序文件、脚本文件、可执行程序文件以及各种数据文件。普通文件的文件类型标识位为“-”，使用 ls 命令可以查看文件的类型，例如：

```
#ls  -l  suple
-rw-r--r--    1 root root    540 07-23 08:31 suple
```

2. 目录文件

目录实际上是一种特殊的文件。目录下可以包含文件和子目录。目录文件的类型标识位为“d”，如下所示：

```
#ls   -l   tmp
drwxrwxrwt   21 root root   4096 07-25 22:49 tmp
```

3. 套接字文件

套接字（socket）是用来进行网络之间通信的常用方法之一。Linux 文件系统可以通过套接字文件实现网络通信。套接字的文件类型标识位为“s”，例如：

```
#ls   -l   x1
srwx------   2 root root   0 07-25 17:49 x1
```

4. 命名管道

文件系统通过命名管道文件可以实现进程间的通信。命名管道的文件类型标识位为“p”，例如：

```
#ls   -l   p1
prwx------   2 root root   0 07-25 17:59 p1
```

5. 设备文件

Linux 系统将设备识别为特殊的文件进行处理。设备文件可以分为两类：字符设备和块设备。字符设备的文件类型标识位为“c”，打印机，键盘等都属于字符设备。磁盘、磁带等都属于块设备，块设备的文件类型标识位为“b”。在系统的/dev 目录下存放了大量的设备文件，例如字符终端 tty1 的设备文件为/dev/tty1。使用 ls 命令可以看到字符设备的首字符为“c”，块设备的首字符为“b”，如下所示：

```
#ls   -l   /dev/tty1
crw-------   1 root root       4,      1 07-25 05:44 tty1
#ls   -l   /dev/sda1
brw-r-----   1 root disk       8,      1 07-25 05:41 sda1
```

6. 链接文件

为了使用、管理的方便和节省磁盘空间，Linux 允许一个物理文件有一个以上的逻辑名，即可以为一个文件创建一个链接文件，用来表示该文件的另一个名字。链接中的不同的文件可为之指定不同的访问权限，从而实现既可共享，又可安全控制的的目的。

Linux 文件系统中有两类链接文件：一类叫做硬链接，一类叫做符号链接。硬链接的文件类型标识位与被链接的文件相同。不带参数使用 ln 命令可以建立硬链接文件，例如对 sysv 文件建立硬链接命令如下：

```
# ls   -il   sysv
390162 -rw-r--r-- 1 root root 0 07-26 00:51 sysv
# ln   sysv syslink
# ls   -il   sysv slink
390162 -rw-r--r-- 2 root root 0 07-26 00:51 slink
390162 -rw-r--r-- 2 root root 0 07-26 00:51 sysv
```

从本例中可以看硬链接文件 slink 与被链接的文件 sysv 指向同一个 i 节点（节点编号 390162），硬链接与被链接的文件具有相同的文件类型标识位“-”，建立硬链接后，文件的链接数由 1 变为 2。

实际上硬链接只是源文件的一个硬复制，它们在目录文件中的入口项指向的是同一个 i 节点。只有当硬链接的全部链接被删除时才能够释放此 i 节点。任何对这个文件所做的修改，所有的硬链接都可以同步看到。硬链接的文件必须在同一个文件系统中，目录不能建立硬链接。

建立符号连接可以使用带参数“-s”的 ln 命令，符号链接只是指定到真实文件的访问路径上，与源文件的 i 节点号不同。如果源文件被删除，符号链接就被损坏。符号链接的文件类型标识位为“l”。例如为文件 ftpuser 建立符号链接 fuser，如下所示：

```
# ls   -il   ftpuser
390161 -rw-r--r-- 1 root root 0 07-26 01:17 ftpuser
```

```
# ln   -s   ftpuser   fuser
# ls   -il   ftpuser fuser
390161 -rw-r--r-- 1 root root 0 07-26 01:17 ftpuser
390162 lrwxrwxrwx 1 root root 7 07-26 01:18 fuser -> ftpuser
# rm   ftpuser
rm：是否删除 一般空文件 “ftpuser”？y
# ls   -il   ftpuser   fuser
ls: ftpuser: 没有那个文件或目录
390162 lrwxrwxrwx 1 root root 7 07-26 01:18 fuser -> ftpuser
```

可以看到 ftpuser 与 fuser 的 i 节点号不同（ftpuser 为 390161，fuser 为 390162），fuser 的文件类型标识位为“l”，源文件 ftpuser 被删除后，符号链接文件报错。

与硬链接相比较，符号链接可以跨文件系统建立，并且可以指定到目录。硬链接与符号链接的关系如图 6.3 所示。

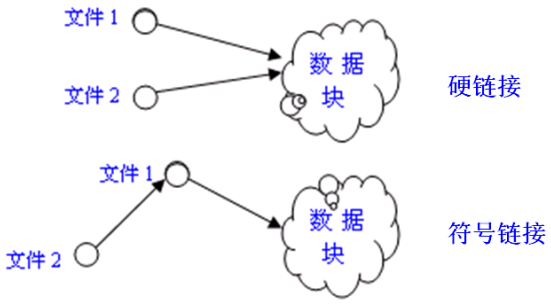

图 6.3 硬链接与符号链接

6.10　文件和目录的权限

Linux 是一个多用户操作系统，权限管理是实现 Linux 系统安全的主要途径。通过权限设置可以有效保护系统和用户的数据安全。

6.10.1　权限的常规表示

在 Linux 中每一个文件和目录都具有相应的权限，如表 6-6 所示。

表 6-6 文件和目录的权限

文件	目录
无权限(-)	无权限(-)
读(r): 允许读文件的内容	读(r): 允许查看目录中有哪些文件和目录
写(w): 允许向文件中写入数据	写(w): 允许在目录下创建（或删除）文件、目录
执行(x): 允许将文件作为程序执行	执行(x): 允许访问目录(用cd 命令进入该目录，并查看目录中可读文件的内容)

权限分为五类，表示方法与含义见表 6-7。

表 6-7 权限的表示法

符号表示	八进制表示	含义
-	0	没有权限
r	4	read的缩写，拥有读权限

w	2	write的缩写，拥有写权限
x	1	execute的缩写，拥有执行权限
s、S、t、T		特殊权限

权限的作用范围可以分为四类，如表 6-7 所示。

表 6-7 权限的作用范围

符号表示	含义
u	user的缩写，文件所有者(文件的创建者)
g	group的缩写，同组用户(与文件所有者同组的用户)
o	other的缩写，其他用户(系统中除所有者、同组用户以外的用户)
a	all的缩写，全部的用户，包括所有者、同组用户以及其他的用户

在 Linux 中，每个文件和目录都与三个实体相关：

- 属主：文件或目录的所有者。通常情况下就是该文件或目录的创建者，对应表 6-7 中的“u”。
- 用户组：该文件或目录所在的用户组，对应表 6-7 中的“g”。
- 其他用户：其他所有可能对该文件或目录进行操作的用户，对应表 6-7 中的“o”。

相应地，文件或目录的权限由以上三部分及文件类型，共 10 个字符位，4 部分组成，如表 6-8 所示。

表 6-8 文件和目录的权限字段

位	1	2	3	4	5	6	7	8	9	10
值	-	r或-	w或-	x或-	r或-	w或-	x或-	r或-	w或-	r或-
说明	文件类型	属主的权限			组权限			其他用户的权限		

其中 2、5、8 位表示读权限，若要赋于读取的权限，可以将这 3 位对应值设为“r”，若不允许读取则输入“-”；3、6、9 位表示写入权限，若要赋于写权限，可以将这 3 位对应值设为“w”，若不允许写入则设为“-”；4、7、10 位表示可执行权限，若要赋于可执行权限，可以将这 3 位对应值设为“x”，若不允许执行则设为“-”。例如设定普通文件 ABC.exe 的属主权限为读、写，执行，组权限为读写，其他用户的权限为读，则文件 ABC.exe 的权限字段为：-rwxrx-r--。

权限除了字符表示法，还有数字表示法。数字表示法是指将读（r）、写入（w）和执行（x）分别以二进制对应位置“1”或置“0”来表示是否有读、写或执行的权限。权限的字符表示法与数字表示法的对应关系见表 6-9。

表 6-9 权限的字符、二进制、八进制表示法

权限	二进制	八进制	权限	二进制	八进制
---	000	0	- - x	001	1
- w -	010	2	- w x	011	3
r - -	100	4	r - x	101	5
r w -	110	6	r w x	111	7

表 6-10 是几个存取权限的范例。

表 6-10 存取权限的范例

字符表示	二进制表示	八进制表示
rwxrwxrwx	111,111,111	777
rw-r-----	110,100,000	640
rw-r--rw-	110,100,110	646
rw-r--r--	110,100,100	644

可以使用带参数“-l”的 ls 命令来查看文件或目录的权限。“ls -l”会以长格式显示文件的信息，包括文件的权限、链接数、创建日期、时间等。例如以长格式显示文件 p.c 的详细信息如图 6.4 所示。

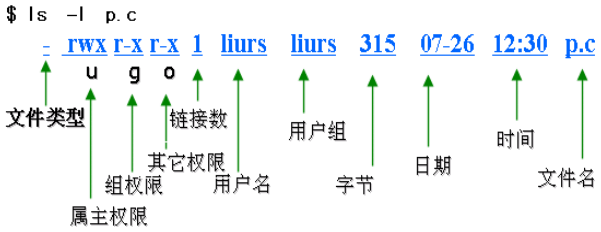

图 6.4 文件 p.c 的详细信息

6.10.2　使用 chmod 命令进行权限设置

可以使用命令 chmod 来为文件或目录赋予权限。chmod 命令格式如下：

```
chmod [选项] [模式] [参考文件=文件名]
```

- -c, --changes ：与 verbose 相反，只在有更改时才显示结果。
- -f, --silent, --quiet：去除大部份的错误信息，静默模式。
- -v, --verbose：显示全部信息，冗余模式。
- --reference=file：不再使用自行指定权限的模式进行权限赋值，而使用[参考文件]的模式。
- -R, --recursive：以递归方式更改所有的文件及子目录。
- -help：显示帮助信息并退出。
- -version：显示版本信息并退出。

1. 八进制模式

chmod 命令中的[模式]可以是八进制模式，即八进制数字表示的权限，同表 6-10。例如预设置普通文件 ABC.exe 的属主权限为读、写，执行，组权限为读写，其他用户的权限为读，即文件 ABC.exe 的权限为：rwxrx-r--，转化为八进制可表示为：764，则使用命令如下：

```
#chmod   764   ABC.exe
#ls   -l   ABC.exe
-rwxrx-r--   1 root root 224 07-21 23:10 ABC.exe
```

例如把/home/teacher 目录的权限设为：属主可以读、写、执行，用户组也可以读、写、执行，其他用户只能读，则权限应设置为：rwxrwxr--，转化为数字表示应为：774，命令如下所示：

```
#chmod   774   /home/teacher
#ls   -l
drwxrwxr--   2 root root 224 07-21 23:30 teacher
…   …
```

上例中的命令只能修改/home/teacher 目录的权限，如果需要将/home/teacher 目录及其目录中的文件和子目录的权限一并进行修改，则需使用递归参数“-R”。例如将/home/teacher 中的所有文件及子目录一并修改为 rwxrwxrwx，命令如下所示：

```
# chmod   -R   777   teacher1
# ls   -l   teacher1
drwxrwxrwx 2 root root 4096 07-26 15:58 homework
-rwxrwxrwx 1 root root       0 07-26 15:58 pp.c
#cd ..
# ls   -l
drwxrwxrwx   4 teacher1 teacher1 4096 07-26 15:58 teacher1
…   …
```

可以看到目录 teacher、目录 teacher 下的子目录 homework 以及目录 teacher 下的文件 pp.c 具有相同

的权限。

2. 字符模式

chmod 的权限模式既可以用八进制数字的方式表示，也可以使用字符方式，配合运算符“＋”，“—”，“＝”，可以实现增加、减少权限或指定权限。修改权限可选用的字符选项见表 6-11 所示。

表 6-11 权限的字符选项

用户对象表	修改操作	权限表示
u 文件主 g 同组用户 o 其他用户 a 所有用户	+ 赋予权限 - 拒绝权限 = 设置权限	r 读权限 w 写权限 x 执行权限 - 无权限 s，S 设置set-UID、set-GID t，T 粘滞位

例如给文件 ABC.exe 的属主增加执行权限，命令行如下：

```
# ls  -l
-rw-r--r-- 1 root root 0 07-26 08:16 ABC.exe
# chmod u+x ABC.exe
# ls  -l
-rwxr--r-- 1 root root 0 07-26 08:16 ABC.exe
```

例如去掉文件的同组和其他用户的读权限，可以使用命令：

```
# ls  -l
-rwxr--r-- 1 root root 0 07-26 08:16 ABC.exe
# chmod g-r,o-r ABC.exe
# ls  -l
-rwx------ 1 root root 0 07-26 08:16 ABC.exe
```

注意：设置的各权限之间用“,”分隔，且“,”前后不可有空格，否则无法执行命令。

例如重新设定文件 ABC.exe 的其他用户的权限为读取，可以使用命令：

```
# ls  -l
-rwx------ 1 root root 0 07-26 08:16 ABC.exe
# chmod o=r ABC.exe
# ls  -l
-rwx---r-- 1 root root 0 07-26 08:16 ABC.exe
```

例如重新设定文件 ABC.exe 的其他用户的权限为读取、写入和执行，可以使用命令：

```
# ls  -l
-rwx---r-- 1 root root 0 07-26 08:16 ABC.exe
# chmod o=rwx  ABC.exe
# ls  -l
-rwx---rwx 1 root root 0 07-26 08:16 ABC.exe
```

例如为所有用户（包括属主，组，其他）赋予写权限，可以使用命令：

```
# ls  -l
-r-x---r-- 1 root root 0 07-26 08:16 ABC.exe
# chmod a+w ABC.exe
# ls  -l
-rwx-w-rwx 1 root root 0 07-26 08:16 ABC.exe
```

利用 chmod 的[reference=fie]选项可以对文件的权限进行复制，将“reference（参考）”文件的权限直接进行拷贝。例如：

```
# ls  -l
```

```
-rwx---r-- 1 root root 0 07-26 08:16 ABC.exe
# chmod   -reference=ABC.exe   copy.x
# ls   -l
-rwx---rwx 1 root root 0 07-26 08:19 ABC.exe
-rwx---rwx 1 root root 0 07-26 08:19 copy. x
…   …
```

可以看到 copy. x 文件与 ABC.exe 文件拥有了相同的权限。

6.10.3　设置特殊权限

除了读、写、执行权限之外，在 Red Hat Enterprise Linux 文件系统中还有类特殊权限，包括 set-UID、set-GID 以及粘滞位（sticky）。

从表 6-11 可以看到，使用 chmod 命令可以对特殊权限进行设置。如果要加上特殊权限，chmod 命令必须使用 4 位八进制数字格式。其中第 1 位表示特殊权限，set-UDI 对应值为“4”，set-GDI 对应值为“2”，粘滞位对应值为“1”。其余三位表示普通权限（包括属主、用户组及其他用户的权限）。

1. 设置 set-UID

如果文件的权限被设置为 set-UID，则表示如果该文件是可执行的，则运行该文件的用户将在程序运行期间拥有与该文件的属主相同的权限。例如，假设有下面的 Shell 脚本程序被存储到 test.exe 文件中：

```
#cat   test.exe
#!/bin/bash
echo $PATH
ps –aux >>/home/process
#
```

使用 chown 命令设置 test 脚本的属主，使用 chmod 命令设定脚本的权限：

```
chown   root:root   test.exe
chmod   4777   test.exe
# ls   -l   test.exe
-rwsrwxrwx 1 root root 0 07-26 08:19 test.exe
```

此时 test.exe 的属主是 root，由于设置了 set-UID 位，任何运行该文件的用户都会临时具有 root 身份，无疑这是非常危险的，应慎重使用 set-UID 权限。

2.设置 set-GID

与 set-UID 类似，如果文件的权限被设置为 set-GID，则表示如果该文件是可执行的，则运行该文件的用户会在运行期间拥有与该文件的用户组相同的权限。例如可以对文件 test1.exe 设定 set-GID：

```
#chown   root:nobody   test1.exe
#chmod   2775   test1.exe
# ls   -l   test1.exe
-rwxrwsr-x 1 root root 0 07-26 09:19 test1.exe
```

3.设置粘滞位

如果文件被设置了粘滞位(sticky-bit)，则表示该文件如果是可执行的，一旦被装载进内存就一直驻留内存。例如对文件 test2.exe 设定粘滞位，命令如下：

```
#chmod   1775   test2.exe
# ls   -l   test2.exe
-rwxrwxr-t 1 root root 0 07-26 10:19 test1.exe
```

注意：只需将特殊权限所对应的值相加，就可一次完成文件的特殊权限设置。例如对文件 text3.exe 设置 set-UDI、set-GID 及粘滞位，各相值累加为：4 + 2 + 1 = 7，则命令行为：

```
#chmod   7775   test3.exe
# ls   -l   test2.exe
-rwsrwsr-t 1 root root 0 07-23 13:19 test3.exe
…   …
```

6.10.4　设置文件或目录的默认权限

每一个新创建的文件或目录系统都会自动赋于一个默认的权限。可以使用 umask 命令设置文件或目录的默认权限。umask 命令的格式如下所示：

```
umask   [mask]
```

其中[mask]可以是由 4 个 8 进制数字组成的权限掩码，直接使用 umask 命令可以显示系统默认的权限掩码：

```
#umask
0022
```

通常新建文件的默认权限值为 0666，新建目录的默认权限值为 0777，与当前的权限掩码为 0022 相减，所以每个新增加的文件的最终权限值为：0666－0022＝0644，而新建目录的最终权限值为：0777－0022＝0755。例如新建文件 test，新建目录 T，通过 ls 命令可以看到生成的最终权限：

```
#umask
0022
# touch   test
# ls   -l   test
-rw-r--r-- 1 root root 0 07-26 09:06 test        //test 的权限为 rw-r--r--，即 644
# mkdir   T
# ls
 T   test
# ls   -l
drwxr-xr-x 2 root root 4096 07-26 09:07 T        //T 的权限为 rwxr-xr-x，即 755
-rw-r--r-- 1 root root       0 07-26 09:06 test
```

可以使用 umask 命令重新设置权限掩码。例如将系统默认的权限掩码设为 0002，则新建文件的最终权限为：0666－0002＝0664（即：rw-rw-r--），新建目录的最终权限为：0777－0002＝0775（ 即：rw-rw-r--），如下所示：

```
# umask   0002
# touch   test3
# ls   -l   test3
-rw-rw-r-- 1 root root 0 07-26 09:40 test3
# mkdir   new
# ls   -l
drwxrwxr-x 2 root root 4096 07-26 09:42 new
-rw-rw-r-- 1 root root       0 07-26 09:40 test3
```

umask 命令也可以直接通过表 6-11 的权限参数直接设置新建文件或目录的默认权限。例如将默认权限改为属主读、写、执行，同组用户读，其他用户为读、执行，可以使用如下命令：

```
# umask   u=rwx,g=r,o=rw
# umask
0031
```

```
# touch   p
# ls   -l   p
-rw-r--rw- 1 root root 0 07-26 09:14 p
# mkdir   M
# ls   -l
drwxr--rw- 2 root root 4096 07-26 09:15 M
-rw-r--rw- 1 root root       0 07-26 09:14 p
```

从本例可以看出，“umask u=rwx,g=r,o=rw”与“umask 0031”作用相同，但文件的执行权限不可由 umask 命令的“x”选项进行指定。

6.10.5　访问控制列表 ACL

基于用户和用户组的权限机制奠定了 Linux 系统安全的基础，但在十几年的使用中也暴露出一些不足，例如权限只能基于用户或用户组进行设定，无法为用户组中的个别几个用户设定不同的权限。为了增加文件或目录权限管理的灵活性，从 Red Hat Enterprise Linux 3 开始，访问控制列表（ACL）被引入到系统中。ACL 可以根据需要对用户的权限进行定制，支持标准的 ext3 文件系统、NTFS 文件系统以及 Samba 文件系统。

启动 ACL 之前首先应对需要进行访问控制的分区或目录进行挂载，具体可以参照 mount 命令，格式如下：

```
mount   -o   acl   <device>   <mount point>
```

例如在/myfile 目录上挂载带有 ACL 支持的/dev/sda1 分区，命令行如下：

```
#mount   -o   acl   /dev/sda1   /myfile
```

ACL 在分区中成功启动后，可以使用 setfacl 命令来添加、修改或删除访问权限。setfacl 常用选项及格式如表 6-12 所示：

表 6-12 setfacl常用选项及格式

setfacl选项格式	说明
[d[efault]:] [u[ser]:]uid [:perms]	设置用户的权限，4个字段分别表示：默认设置：用户名：用户ID：权限
[d[efault]:] g[roup]:gid [:perms]	设置用户组的权限，4个字段分别表示：默认设置：用户组名：用户组ID：权限
[d[efault]:] m[ask][:] [:perms]	设置权限掩码，3个字段分别表示：默认设置：掩码：权限
[d[efault]:] o[ther][:] [:perms]	设置其他用户的权限，4个字段分别表示：默认设置：其他用户名：其他用户名ID：权限

可以使用“-m”参数来添加用户或用户组的权限。例如对用户 teacher1、teacher2 设置对/homework 目录的读、写和执行权限，可以运行下面的命令：

```
#setfacl   -m u:teacher1:rwx   /home/homework
#setfacl   -m u:teacher2:rwx   /home/homework
```

对用户 student1、student2 设置对/homework 目录的读、写权限，可以运行下面的命令：

```
#setfacl   -m u:student1:rw    /home/homework
#setfacl   -m u: student2:rw   /home/homework
```

例如对用户组 director 设置对名为/home/director 目录的读权限，命令行如下：

```
#setfacl   -m u: director:r   /home/director
```

要对权限进行修改，与添加 ACL 相同，可以使用带“-m”选项的 setfacl 命令，例如修改上例中的 director 用户组的权限为读、写、执行，命令如下：

```
 #setfacl   -m u: director:rwx   /home/director
```

可以使用“-x”参数来删除一个用户或用户组的权限，例如删除上例中 director 用户组的权限可以

使用命令如下：

```
#setfac l -x u: director    /home/director
```

可以使用“-d”选项对一个文件或目录设置默认的 ACL。例如对/home/ftp 目录的用户组设定默认的权限为读、写入，命令如下：

```
#setfacl    -m d:g:rw /home/ftp
```

注意：若组中用户重新指定权限，则默认的组权限被覆盖，即具体指定的 ACL 权限优先于默认 ACL 权限。

6.10.6　权限的图形化管理

在 Red Hat Enterprise Linux 5 环境下，如果需要使用 X-Window 修改一个文件或目录的权限，首先打开 Nautilus 文件管理器。Nautilus 文件管理器与 Windows 资源管理器类似，可以浏览系统的目录结构，可以查找、打开、移动、复制和删除文件或目录，还可以运行脚本程序。

（1）在 Nautilus 中找到需要修改的文件或目录，单击鼠标右键，在快捷菜单中选择【属性】选项，如图 6.5 所示。

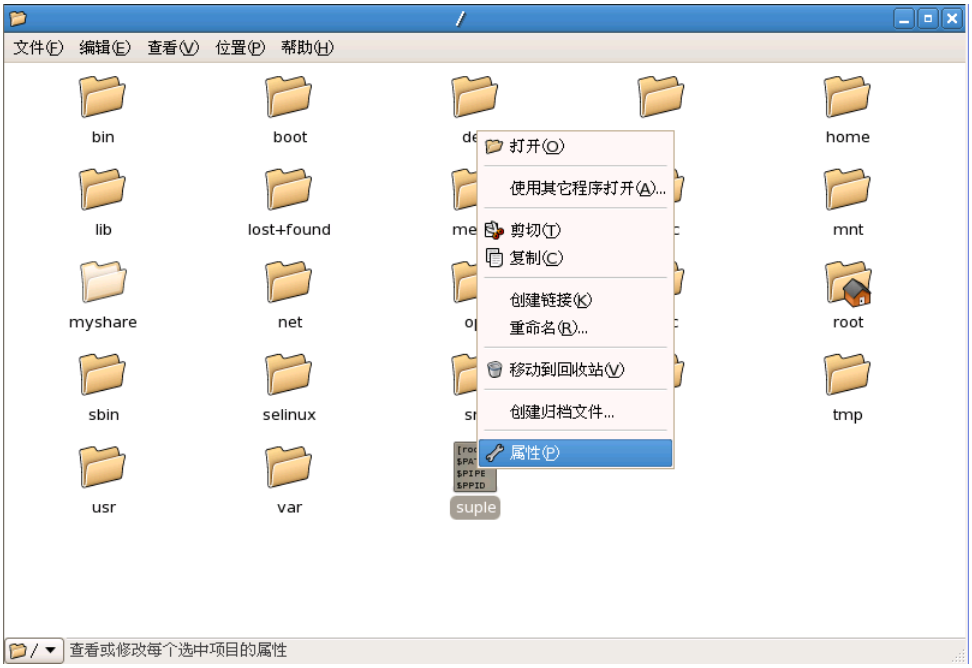

图 6.5　在 Nautilus 文件管理器右键菜单中选择【属性】

（2）在打开的【属性】对话框中，可以设置文件或目录的图标、名称、打开方式等。选择【权限】标签，打开【权限】选项卡，如图 6.6 所示。可以在【权限】选项卡中对文件或目录的所有者、用户组进行修改。

图 6.6 在【属性】窗口中选择【权限】选项卡

（3）单击【所有者】下拉菜单，会显示目前系统中的所有用户，重新选择一个用户即可更改该文件或目录的属主。同样单击【群组】菜单，也可以更改文件或目录的所属的用户组，如图 6.7 所示。

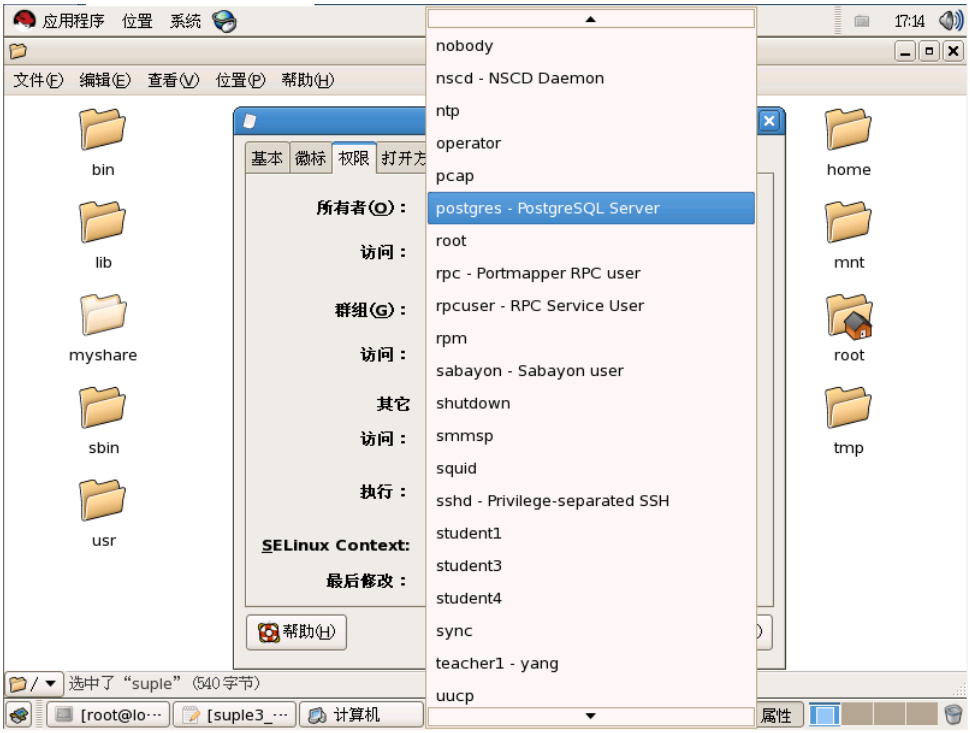

图 6.7 修改文件或目录的所有者

（4）在【权限】选项卡中单击【访问】下拉菜单，可以对所有者、用户组以及其他用户的访问权限进行重新设定，所图 6.8 所示

图 6.8 修改文件或目录的权限

（5）单击【关闭】按钮，完成权限的设置。

在 Red Hat Enterprise Linux 4 的【权限】选项卡中还可以对特殊权限进行设定，由于特殊权限的设定很可能带来安全隐患，在 Red Hat Enterprise Linux 5 中已将该选项从该选项卡中去掉。同时在图 6.8 中可以看到 Red Hat Enterprise Linux 5 中加入了 SELinux 来加强安全控制。

6.11　文件和目录管理常用命令

文件和目录管理涉及的命令比较多，在现存的各个版本的 Linux 系统中，各命令功能大体相同。

6.11.1　文件和目录操作中常用通配符

在 Linux 文件系统中，可以使用通配符来匹配多个选择。常用的通配符如表 6-13 所示。

表 6-13 常用通配符说明

通配符	说明
*	用来代表文件中的任意长度的任意字符。
?	用来代表文件中的任一个字符。
[…]	匹配任意一个在中括号中的字符，中括号里可以是一个用破折号分割的字母或数字范围。
前导字符串{…}后继字符串	大括号中的字串逐一匹配前导字符串和后继字符串。

例如当前目录下存在 cars 、cat、can、cannon、truck、bus、bike 几个文件，列出所有以字母“c”开头的文件，可使用命令：

```
#touch  car  cats  can  cannon  truck  bus  bike
#ls  c*
cars can cannon
```

列出所有以字母“b”开头的文件名，命令行为：

```
#ls  b*
bike bus
```

列出所有第一个字母为“c”，最后一个字母为“n”的文件，命令行为：

```
#ls  c*n
can cannon
```

列出包含字母“a”的所有文件，命令行为：

```
#ls  *a*
cars cat can cannon
```

列出当前目录下的所有文件，命令行为：

```
#ls  *
bike bus cars cat can cannon truck
```

通匹符“？”只能匹配任意一个字符。例如上例中列出所有第 3 个字母是“n”的文件，命令行为：

```
#ls  ??n*
can cannon
```

列出第一个字母是“b”，第三个字母是“s”的所有文件，命令行为：

```
#ls  b?s*
bus
```

中括号表示一个匹配的字符集，例如[123456]与[1-6]都表示数字 1、2、3、4、5 和 6。大写字母 A 到 D 之间的任意一个字符可用[A-D]表示。多个集合之间可以用逗号分隔，例如[1-10,a-z,A-Z]表示数字 1 到 10，小写字母 a 到小写字母 z，大写字母 A 到大写字母 Z。一个集合前若有前缀“！”，则表示除了集合包含的字符以外的所有字符组成的集合。例如表示所有的辅音组成的字符集可写成[!aeiou]。例如上例中显示所有以字母 b 或 c 开头的文件名，命令行为：

```
#ls  [b,c]*
bike bus car cat can cannon
```

例如列出所有以字母 b 或 c 开头，以字母 s 或 k 结尾的文件，命令行为：

```
#ls  [b,c]*[s,k]
bus cars
```

大括号是用来查找文件的一个常用方法，例如以长格式列出 cars、cans 和 cats 文件信息，可以使用命令：

```
#ls  -l  c{ar,an,at}s
```

6.11.2　ls 或 dir 命令：列出当前目录的内容

ls（list 的缩写）命令可以列出当前目录的内容，dir 命令是 ls 命令的一个别名，取 directory 的缩写。通常列出的文件会以不同的颜色进行显示，不同的颜色代表不同的文件类型，表 6-14 列出了文件类型与颜色的对应关系。

6-14 文件类型与颜色的对应关系

文件类型	颜色
目录	深蓝色
一般文件	浅灰色
执行文件	绿色
图形文件	紫色
链接文件	浅监色
匪缩文件	红色
FIFO文件（命名管道）	棕色
设备文件	黄色

ls 还会对一特定类型的文件用符号进行标识，表 6-15 列出了常用的标识符号。

表 6-15 特定文件类型标识符

符号	说明
.	表示隐藏文件
/	表示一个目录名
*	表示一个可执行文件
@	表示一个符号链接文件
|	表示管道文件
=	表示socket文件

ls 命令的格式如下：

```
ls [选项]　目录或文件名
```

- -a：列出指定目录下所有文件和子目录的信息(包括隐含文件）。
- -A：同-a,但不列出. 和..。
- -b：当文件名中有不可显示的字符时，将显示该字符的八进制数字。
- -c：按文件的最后修改时间排序。
- -C：分成多列显示。
- -d：显示目录名而不是显示目录下的内容，一般与-l 连用。
- -f：在列出的文件名后加上符号来区别不同类型。
- -R：递归地显示指定目录的各级子目录中的文件。
- -s：给出每个目录项所用的块数，包括间接块。
- -t：按最后修改时间排序（新的在前，旧的在后）。
- -l：以长格式显示文件的详细信息，包括：文件的类型与权限、链接数、文件所有者、文件所有者所属的组、文件大小、最近修改时间、文件名。详情见图 6-4。

下面以不同的格式显式目录的内容：

```
# ls                  //以缩略格式显示目录内容
cal_txt         finger_txt     id_txt       newgrp_txt     suple_txt     who.txt
finger2_txt    groups_txt     last_txt    suple3_txt     suple_txt~    w.txt
finger3_txt    groups_txt2    ln             suple3_txt~    w
# ls   -l              //以长格式显示目录内容，包括权限、用户名、修改时间等
总计 224
-rw-r--r-- 1 root root   2163 07-24 13:00 cal_txt
-rw-r--r-- 1 root root    212 07-23 06:37 finger2_txt
-rw-r--r-- 1 root root      0 07-23 06:39 finger3_txt
-rw-r--r-- 1 root root    248 07-23 06:31 finger_txt
-rw-r--r-- 1 root root     35 07-23 04:22 groups_txt
-rw-r--r-- 1 root bin      140 07-23 04:32 groups_txt2
-rw-r--r-- 1 root root    144 07-23 04:11 id_txt
-rw-r--r-- 1 root root   1655 07-22 19:19 last_txt
drwxr-xr-x 5 root root   4096 07-26 09:42 ln
-rw-r--r-- 1 root bin      143 07-23 04:24 newgrp_txt
-rw-r--r-- 1 root root 25098 07-26 19:15 suple3_txt
-rw-r--r-- 1 root root 24031 07-26 16:00 suple3_txt~
-rw-r--r-- 1 root root 25198 07-24 20:16 suple_txt
-rw-r--r-- 1 root root 24827 07-24 20:06 suple_txt~
-rw-r--r-- 2 root root    196 07-23 06:48 w
-rw-r--r-- 1 root root     46 07-23 06:48 who.txt
-rw-r--r-- 2 root root    196 07-23 06:48 w.txt
```

```
# ls –s          //显示所用的块数
总计 224
 8 cal_txt        8 groups_txt      8 ln              32 suple_txt      8 w.txt
 8 finger2_txt    8 groups_txt2     8 newgrp_txt      32 suple_txt~
 4 finger3_txt    8 id_txt          32 suple3_txt     8 w
 8 finger_txt     8 last_txt        28 suple3_txt~    8 who.txt
```

选项可以组合使用。例如，如果需要列出当前目录的所有内容(包括那些以“.”开头的隐含文件)，并以“冗余格式”从屏幕输出文件的详细信息，可以使用选项“-al”。以冗余格式显示/root 目录下的所有文件，可以使用命令：

```
# ls   -al   /root
drwxr-x--- 18 root root   4096 07-26 16:00 .                    //当前目录
drwxr-xr-x 24 root root   4096 07-25 05:43 ..                   //父目录
-rw-------   1 root root    997 07-15 20:58 anaconda-ks.cfg      //普通文件
-rw-------   1 root root   2827 07-24 21:32 .bash_history       //隐藏文件
-rw-r--r--   1 root root     24 2006-07-13 .bash_logout         //隐藏文件
…   …
```

其中文件名为“.”表示当前目录，对应行列出了当前目录的详细信息。文件名为“..”表示当前目录的上一级目录，即父目录，对应行列出了父目录的详细信息。文件名前有“.”符号的文件表示隐藏文件，只有使用了“-a”参数才会显示出来。

6.11.3　cd 命令：更改当前目录

cd（Chage Directory）命令用于更改当前目录，表 6-16 列出了常用的 cd 命令：

表 6-16 cd命令与运行结果

命令	说明
cd	切换到当前用户的主目录
cd ..	切换到当前目录的上一层目录，例如当前目录为/home/student，使用该命令可以将当前目录移到/home
cd ../..	切换到当前目录的上二层目录，例如当前目录为/home/student/student1，使用该命令可以将当前目录移动到/home
cd ~	切换当前目录为当前用户的主目录，适用于任何用户
cd /	切换当前目录到根目录，即返回到/

注意：在 Linux 中，引用目录名、计算名或者域名时使用正斜杠“/”，而在 Windows 中使用反斜杠“\”。

例如当前系统中存在目录结构如图 6.9 所示，其中用户当前目录为/home/student。

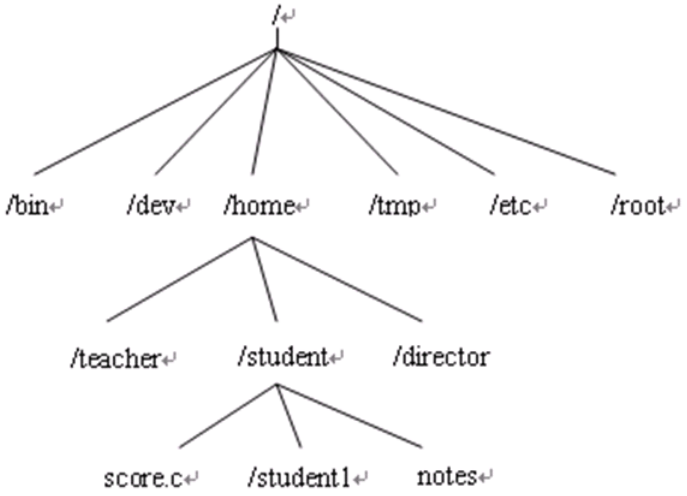

图 6.9 目录结构图

若改变当前目录为/home/director，可以使用相对路径，命令如下：

```
#pwd
/home/student
#cd ../director
#pwd
/home/director
```

也可以使用绝对路径切换到/home/dirctor，命令如下：

```
#pwd
/home/student
#cd /home/director
#pwd
/home/dirctor
```

将当前目录由/home/stuent 切换为/home/student/student1。由于 student1 目录位于 student 目录下，可以采用相对路径：

```
#pwd
/home/student
#cd student1
#pwd
/home/student/student1
```

也可以采用绝对路径，命令如下：

```
#pwd
/home/student
#cd /home/student/student1
#pwd
/home/student/student1
```

从 student1 目录返回根目录，可以使用命令：

```
#pwd
/home/student/student1
#cd /
 #pwd
/
```

也可以使用相对路径，命令行为：

```
#pwd
/home/student/student1
#cd ../../..
#pwd
/
```

6.11.4　cp 命令：复制目录或者文件

cp 命令用于实现文件或目录的复制，与 DOS 下的 copy 命令相似。cp 命令格式如下：

```
cp　[选项]　源文件或目录　目标文件或目录
```

- -a：该选项常在复制目录时使用，该选项保留链接、文件属性，并递归地复制目录。
- -f：如果目标文件或目录已存在，就覆盖它，并且不做提示。
- -i：与 f 选项正好相反，在覆盖已有文件时，会让用户回答“Y”来确认。
- -r：若给出的源目标是一个目录，那么 cp 将递归复制该目录下所有的子目录和文件，不过这要求目标也是一个目录名。

常用的命令行组合与执行结果见表 6-17。

表 6-176 cp命令格式与运行结果

命令格式	运行结果
cp　文件1　　文件2	复制源文件1的内容到目标文件2上，目标文件有一个新的创建日期和索引节点号。
cp　多个文件（文件名之间用空格分隔）　　目录1	复制多个文件到目录1中。
cp　-f　文件1　文件2	如果已有一个文件2存在，该命令不显示任何提示就用文件1覆盖文件2。
cp　-i　文件1　文件2	如果已有一个文件2存在，该命令在覆盖文件2之前会给出提示。
cp　-p　目录1　目录2	复制目录1的内容到目录2,如果目录1下还有子目录，则一并进行复制。
cp　-u　文件1　文件2	如果文件2已存在，但文件1比文件2新，则不显示任何提示就覆盖文件2。

例如将把 a.txt 和 b.txt 文件复制到/home/teacher 目录中，命令行如下：

```
#cp　a.txt　b.txt　/home/user1
```

cp 命令可以在同一目录下，换名复制一个文件，源文件保持不变。例如将当前目录下 a.txt 复制一份且取名为 b.txt，放在当前目录下，命令行为：

```
#cp　a.txt　b.txt
```

若源文件是普通文件，则直接拷贝到目标文件；若是目录，需要使用“-r”选项才能将整个目录复制到目标位置。例如复制/home/teacher 目录下所有内容到/home/student 目录下，命令行为：

```
#cp　-r　/home/teacher /home/student
```

6.11.5　rm 命令：删除目录或文件

rm 命令用于删除文件或目录，可删除一个目录中的一个或多个文件或目录；也可删除某个目录及其下面的所有文件和子目录。对于链接文件只是删除该链接，原文件保持不变。rm 格式如下：

```
rm　[选项]　文件…
```

- -f：在删除过程中不给任何指示，直接删除。
- -r：将参数中列出的全部目录和子目录都递归地删除。
- -i：与 f 选项相反，交互式删除，每个文件在删除时都给出提示。

删除文件可以直接使用 rm 命令，若删除目录则必须配合选项“-r”，例如：

```
# rm   pp.c
rm：是否删除 一般文件 “pp.c”？y
# rm   homework
rm: 无法删除目录“homework”：是一个目录
# rm   -r   homework
rm：是否删除 目录 “homework”？y
例如删除当前目录下所有文件和目录：
```

例如，删除当前目录下的所有文件及目录，命令行为：

```
# rm   -r   *
```

文件一旦通过 rm 命令被删除，则无法恢复，所以一定格外小心使用。

6.11.6　mkdir 命令：创建目录

mkdir 命令用于创建目录。mkdir 命令格式为：

```
mkdir [选项] 目录名
```

- -m 数字：设置新建的目录的权限，权限用数字表示。
- -p ：如果目录名中的路径中包含着不存在的子目录，那么就逐一地建立，直到最后的子目录为止。

创建的目录时，如果目录名前没有指定目录的路径，那么就表示在当前目录下创建；如果有路径名，则指定的路径下建立。新建的子目录必须不能和已经存在的文件名或目录名重名。例如：

```
# pwd                                    //在当前目录下创建子目录 zhang
/home/teacher1
# ls
# mkdir zhang
# ls
zhang
# mkdir   /home/teacher1/yang            //使用绝对路径创建子目录 yang
# ls
yang   zhang
# mkdir zhang                            //不能重得创建
mkdir: 无法创建目录 “zhang”：文件已存在
```

在创建子目录的时候，如果子目录的父目录不存在，则无法创建。使用选项“-p”，则可以逐级创建目录。例如在当前目录下创建 li/document 目录：

```
# pwd
/home/teacher1
#mkdir   li/document          //由于不存在 li 子目录，li/document 子目录无法创建
mkdir: 无法创建目录 “li/document”：没有那个文件或目录
# mkdir   -p   li/document       //参数“-p”允许逐级创建目录
# ls
li   yang   zhang
# cd li
# ls
document
```

例如创建新目录/usr/yang/example，且指定权限为 700，命令行为：

```
$ mkdir   -m   700   /usr/Bob/example
```

6.11.7　rmdir 命令：删除空目录

rmdir 命令用于删除一个空目录。被删除的目录必须是一个空目录否则无法删除。rmdir 只用于删除目录，无法删除文件。rmdir 命令格式为：

```
rmdir　[选项]　目录名
```

选项：　-p：删除目录下所有的空目录，如果有非空的子目录，则保留下来，如果所有的子目录都删除了，则删除该目录。

例如删除 document 子目录，命令行为：

```
# ls
document
# cd　document
# ls
# rmdir　document　　　//在 document 子目录内无法删除 document 子目录
rmdir: document: 没有那个文件或目录
# cd ..
# rmdir　document
# ls
```

若没有 document 的写权限或目录不空则无法删除，例如：

```
# ls
li　yang　zhang
# rmdir　li
rmdir: li: 目录非空
```

6.11.8　mv 命令：移动文件或者目录

mv 命令用于实现文件或目录的移动。mv 命令格式如下：

```
mv　[选项]　源文件或目录　目标文件或目录
```

- -f：如果操作要覆盖某个已有的目标文件时不给任何指示。
- -i：交互式的操作，如果操作要覆盖某个已有的目标文件时会询问用户是否覆盖。

mv 与 cp 命令明显不同之处在于：mv 命令是移动文件，文件个数没增加；cp 命令是复制文件，文件个数增加。mv 命令还可以进行文件或目录的改名，其参数设置与运行结果对应关系如表 6-17 所示。

表 6-17 mv参数设置说明

命令格式	结果
mv　文件名　文件名	将源文件名改为目标文件名
mv　文件名　目录名	将文件移动到目标目录
mv　目录名　目录名	目标目录已存在：源目录移动到目标目录；目标目录不存在：改名
mv　目录名　文件名	出错

例如将 m1.c 文件改名为 m2.c，命令行为：

```
$ mv m1.c m.c
```

例如将/usr/student 下的所有文件和目录移到当前目录下，命令行为：

```
$ mv /usr/student/*　.
```

6.11.9　find 命令：查找文件

find 命令用于查找文件，其命令格式为：

```
find　[起始目录]　[搜索条件]　[操作]
```

其中[起始目录]是指：命令将从该目录起，遍历其下所有的子目录，查找满足条件的文件，缺省是当前目录。[搜索条件]是一个逻辑表达式，当表达式为“真”时，搜索条件成立，为“假”时不成立。搜索条件一般形式见表 6-18：

表 6-18 find命令搜索条件的一般表达式

搜索条件	说明
-name ‘字符串’	查找文件名中包含所给字符串的所有文件。
-user ‘用户名’	查找属于指定用户的文件。
-group ‘用户组名	查找属于指定用户组的文件。
-type x	查找类型为x的文件，类型包括：b 块设备文件；c 字符设备文件；d 目录文件；p 命名管道文件；f 普通文件；l 符号链接文件；s socket文件。
-atime n	查找n天以前被访问过的文件。
-size n	指定文件大小为n。
-perm	查找符合指定权限值的文件或目录。
-mount	要查找文件时不跨越文件系统mount点。
-follow	如果find命令遇到符号链接文件，就跟踪到链接所指向的文件。
-cpio	对匹配的文件使用cpio命令，将文件备份到磁带设备中。
-newer file1！file2	查找更改时间比文件file1新但比文件file2旧的文件。
-prume	不在指定的目录中查找，如果同时指定-depth选项，那么-prune将被find命令忽略。
-ok	和exec作用相同，但在执行每一个命令之前，都会给出提示，由用户来确定是否执行。
-depth	在查找文件时，首先查找当前目录，然后再在其他子目录中查找。

可执行的操作见表 6-19。

表 6-19 find命令常用可执行操作

可执行操作	说明
-exec 命令名 {} \;	不需确认执行命令。注意：“{}”代表找到的文件名，“}”与“\”之间有空格。
-print	送往标准输出。

例如从当前目录查找所有以.txt 结尾的文件并在屏幕上显示出来，命令行为：

```
$ find　.　-name　‘*.txt’　–print
```

例如从根目录查找类型为符号连接的文件，并将其删除，命令行为：

```
$ find　/　-type　l　–exec　rm { } \;
```

例如从当前目录查找用户 tom 的所有文件并在屏幕上显示，命令行为：

```
$ find　.　–user　‘tom’　-print
```

例如显示当前目录中大于 20 字节长的.c 文件名，命令行为：

```
$ find　. -name　“*.c”　-size　+20c　-print
```

显示当前目录中恰好 10 天前访问的文件名，命令行为：

```
$ find　.　-atime　10　-print
```

例如显示当前目录中不到 10 天前访问的文件名，命令行为：

```
$ find　.　-atime　-10　-print
```

例如查找/home 目录下权限为 640 的文件或目录，命令行为：

```
#find　/home　-perm 640
```

例如搜索根目录下大于 100KB 的文件，命令行为：

```
#find　/　-size　+100K　-print
```

例如搜索根目录下小于 500KB 的文件，命令行为：

```
#find  /  -size  -500K  -print
```

例如在当前目录中查找所有文件名以.doc 结尾，且更改时间在 3 天以上的文件，找到后进行删除，且删除前给出提示，命令行为：

```
#find  .  -name  '*.doc'  -mtime +5  -ok  rm { } \;
```

例如在当前目录下查找所有链接文件，并以长格式显示文件的基本信息，命令行为：

```
# find  .  -type l  -exec  ls  -l {} \;
lrw-rw-r-- 1 root root 36 07-27 14:34 ./example2
lrw-rw-r-- 1 root root 72 07-27 14:36 ./example3
lrw-rw-r-- 1 root root 36 07-27 14:36 ./example1
```

例如在当前目录中查找文件名由一个小写字母、一个大写字母、两个数字组成，且扩展名为.doc 的文件，命令行为：

```
#find  .  -name  '[a-z][A-Z][0-9][0-9].doc'  -print
```

6.11.10　grep 命令：在文件中搜索制定的字符串

grep 命令是“global regular expression print”的缩写，用于在文件中搜索指定的字符串模式，列出含有匹配模式字符串的文件名，并输出含有该字符串的文本行，命令格式为：

```
grep [选项] [查找模式][文件名……]
```

- -F：将查找模式看成是单纯的字符串。
- -i：要查找的字符串不区分字母的大小写。
- -r：以递归方式查询目录下的所有子目录的文件。
- -n：标出符合指定字符串的行编号。

例如在文件 example 中查找包含“aa”字串的行，命令如下：

```
# cat  example
aa bb cc dd
aa bb ff
ee
# grep  aa  example
aa bb cc dd
aa bb ff
```

如果待查找的字符串模式的字数大于 1，则必须在字符串模式两边使用单引号，否则系统会只把第一个字做为搜索目录，例如：

```
# cat  example
aa bb cc dd
aa bb ff
ee
# grep  bb  cc  example
grep: cc: 没有那个文件或目录
example:aa bb cc dd
example:aa bb ff
# grep  'bb cc'  example
aa bb cc dd
```

例如在/passwd 文件中查找包含“teacher”字符串的行，命令为：

```
$ grep  -F  teacher  /etc/passwd
teacher:*:500:500: teacher:/home/ teacher:/bin/bash
```

例如在 file1 中查找包含“print” 字符串的所有行，不管字符的大小写，命令行如下：

```
$ grep   -i   ‘print’   file1
```

例如查找包含字符串“bb cc”的行，输出该行，并输出该行所在的行号，命令为：

```
# cat   example
aa bb cc dd
aa bb ff
ee
# grep   -n 'bb cc'   example
1:aa bb cc dd
```

通常 grep 命令配合管道符（|）还用来作为其他命令的输入，例如统计指定文件中包含某字符串的行数、字数和字节数：

```
# cat   example
aa bb cc dd
aa bb ff
ee
# grep   'bb' example |wc
       2          7         21
```

grep 命令除了做为其他命令的输入，也可以做为一些命令（例如 ls，ps）的输出。例如在当前运行的进程中查找 vi 程序的进程信息，命令行如下：

```
# ps   aux   |   grep   vi
root        5716   0.0   0.2    4956    736 pts/1      T      Jul25     0:00 vi
root       20681   0.3   0.4    4960   1012 pts/1      T      14:21     0:00 vi
root       20689   5.0   0.2    4132    668 pts/1      R+     14:22     0:00 grep vi
```

注意：与 grep 非常相似地还有两个命令：一个是“egrep”，表示 Extend grep，执行效率比 grep 高，但需占用较大内存空间；另一个是“fgrep”，占用空间比 egrep 小，且速度也比 grep 快。由于三个命令的结构、功能类似，大部分参数可以共享使用。

6.11.11　chown 命令：改变文件或目录的拥有者

chown 命令用于改变文件或目录的拥有者，chown 命令格式为

```
chown [选项]  用户名  文件或目录
```

选项：-R：可以一次修改某个目录下所有文件的所有者。

其中用户名为新拥有者的用户标识符。

例如将文件 data 的属主改为 teacher：

```
$chown   teacher   data
```

例如将/root/file1.doc 文件复制到用户 teacher1 的主目录/home/teacher1 中，复制之后可以发现此文件的拥有者仍然是 root，命令行为：

```
# ls   -l   file1.doc
-rw-rw-r-- 1 root root 18 07-27 16:21 file1.doc
# cp   file1.doc   /home/teacher1
# ls   -l   /home/teacher1/file1.doc
-rw-rw-r-- 1 root root 18 07-27 16:22 /home/teacher1/file1.doc
```

为此，使用 chown 将 file.doc 文件的所有权赋予 teacher1，命令行为：

```
# chown   teacher1   /home/teacher1/file1.doc
# ls   -l   /home/teacher1/file1.doc
-rw-rw-r-- 1 teacher1 root 18 07-27 16:22 /home/teacher1/file1.doc
```

可以看到 file1.doc 文件的所有者已改为 teacher1，但用户组仍为 root，也可以使用 chown 命令修改文件所属的用户组，命令格式为：

```
chown　用户名:用户组 文件或目录
```

在上例中一并更改 file1.doc 文件的属主和用户组为 teacher1，命令行为：

```
# chown　teacher1:teacher1　　/home/teacher1/file1.doc
# ls　-l　/home/teacher1/file1.doc
-rw-rw-r-- 1 teacher1 teacher1 18 07-27 16:22 /home/teacher1/file1.doc
```

6.11.12　chrgp 命令：修改文件或目录的所属的用户组

chgrp 允许用户修改文件或目录的所属的用户组，但该用户必须或者是根用户或者同时属于被设置的新的用户组，否则无法修改成功。chrgp 命令格式为：

```
chgrp [选项] 用户组名　文件或目录
```

选项：-R：可以一次修改某个目录下所有文件的用户组。

其中用户组名可以是用户组名称或用户组的 GID。

例如将文件 data 的组别改为 staff

```
$ chgrp　staff　data
```

6.11.13　cat 命令：把一个文件发送到标准输出设备

cat 是 Concatenate 的缩写，用于把一个文件发送到标准输出设备，与 DOS 或 Windows 下的 type 命令相似。cat 命令可以对任意一个文件使用，屏幕会一次显示文件的所有内容，中间不停顿，不分屏。除了显示文件内容外，cat 还具有由键盘读取数据、将多个文件合并的功能。命令格式为：

```
cat [选项] [文件]…
```

选项说明如表 6-20 所示。

表 6-20 cat选项说明

选项	说明
-A,--show-all	等价于-vET
-b,--number-nonblank	对非空行输出行编号
-e	等价于-vE
-E,--show-ends	在每行结束处显示$
-n,--number	对输出的所有行编号
-S,--squeeze-blank	不输出多行空行
-t	与-vT等价
-T,--show-tabs	将跳格字符TAB显示为^I
-v,--show-nonprinting	使用^和M-符号显示非打印字符，但除了LFD和TAB之外
--help	显示帮助信息并离开

例如显示 hello.c 文件内容如下所示：

```
$ cat　hello.c
hello world!
```

该命令可配合重定向“ >”建立小型文本。例如将键盘键入内容输出重定向到文件 example1 中，按 Ctrl+d（或 Ctrl+c）键存盘退出，命令行为：

```
# cat　> example1
aa bb cc dd
```

```
bb cc dd ee
cc dd ee ff
# cat   example1
aa bb cc dd
bb cc dd ee
cc dd ee ff
```

cat 命令可以联合输出多个文件的内容，例如：

```
# cat   example1
aa bb cc dd
bb cc dd ee
cc dd ee ff
# cat   example2
dd ee ff gg
ee ff gg hh
ff gg hh ii
# cat   example1   example2
aa bb cc dd
bb cc dd ee
cc dd ee ff
dd ee ff gg
ee ff gg hh
ff gg hh ii
```

上例中如果将 example1 和 example2 合并后放入新文件 example3 中，命令行为：

```
# cat   example1   example2   >   example3     //合并后输出到 example3 文件
# cat   example3
aa bb cc dd
bb cc dd ee
cc dd ee ff
dd ee ff gg
ee ff gg hh
ff gg hh ii
```

如果在显示输出的每一行前自动添加行号（空白行除外），可使用选项“-b”，命令行为：

```
# cat   -b   example3
      1   aa bb cc dd
      2   bb cc dd ee
      3   cc dd ee ff
      4   dd ee ff gg
      5   ee ff gg hh
      6   ff gg hh ii
```

注意：使用选项“-n”会对所有行加上行号，即使空行也不例外。

6.11.14　more 命令：一次显示一屏信息

more 命令一次显示一屏，若信息未显示完屏幕底部出现“-More-(xx%) ”， 按 Space，显示下一屏内容；按 Enter，显示下一行内容；按 B,显示上一屏；按 Q 退出 more 命令。more 命令格式：

```
more [选项]  文件名
```

常用的 more 选项如表 6-21 所示。

表 6-21 常用的more选项

参数	说明
+n	从第n行开始显示。
-n	定义屏幕大小为n行。
+/pattern	从pattern前两行开始显示。
-c	从顶部清屏，然后显示。
-d	提示“Press space to continue，’q’ to quit（按空格键继续，按q键退出）”，禁用响铃功能。
-l	忽略Ctrl+l(换页)字符。
-p	通过清除窗口而不是滚屏来对文件进行换页，与-c选项相似。
-s	把连续的多个空行显示为一行。
u	把文件内容中的下划线去掉。

在查看一个内容较多，无法在一屏内显示的文件时，经常要用到的 more 操作命令，常用的操作命令及说明见表 6-22。

表 6-22 常用的more操作命令及说明

操作命令	说明
Enter	向下n行，需要定义，默认为1行
Ctrl+F	向下滚动一屏
空格键	向下滚动一屏
Ctrl+B	返回上一屏
=	输出当前行的行号
：f	输出文件名和当前行的行号
V	调用vi编辑器
!命令	调用Shell，并执行命令
q	退出more

例如若显示文件 test 中从第 3 行起的内容，命令行为：

```
#more  +3  test
```

例如使用“+/pattern”选项，从文件 test 中查找第一个出现“teacher”字符串的行，并从该处前两行开如显示输出，命令行为：

```
#more  +/teacher  test
```

若每屏显示 8 行，命令行为：

```
#more  -8  test
```

例如从终端顶部开始显示文件内容，并给出提示信息，命令行为：

```
#more  -dc  test
```

6.11.15　less 命令：显示文件时允许用户既可以向前又可以向后翻阅文件

less 命令和 more 功能相似，显示文件时允许用户既可以向前又可以向后翻阅文件。向前翻按 pageup 键，向后翻按 pagedown 键，退出按 q 键。less 命令格式：

```
less [选项] 文件名
```

less 命令的常用选项见表 6-23。

表 6-23 less命令选项

选项	说明
-c	从顶部(从上到下)刷新屏幕，并显示文件内容，而不是通过底部滚动完成刷新。
-f	强制打开文件，如果是二进制文件也不提示警告。
-i	搜索时忽略大小写，但搜索串中包含大写字母除外。

-I	搜索时忽略大小写，但搜索串中包含小写字母除外。
-m	显示读取文件的百分比。
-M	显示读取文件的百分比、行号及总行数。
-N	在每行前输出行号。
-p pattern	例如在/etc/ftpuser中搜索单词student，可以使用“less　-p　student　/etc/ftpuser”。
-S	把连续多个空白行作为一个空白行显示。
-Q	在终端下不响铃。

在用 less 命令查看文件时，使用一些常用的操作指令可以加快查找、定位的速度。常用的操作指令见表 6-24。

表 6-24 less常用的操作指令

操作命令	说明
回车键	向下移动一行。
y	向上移动一行。
空格键	向下滚动一屏。
b	向上滚动一屏。
d	向下滚动半屏。
h	less的帮助。
u	向上滚动半屏。
w	从指定行数的下一行显示，例如指定的值是9,则从第10行开始显示。
g	跳到第一行。
G	跳到最后一行。
p n%	跳到n%，例如50%，表示从整个文档的50%处开始显示。
/pattern	搜索pattern，例如/ftpuser，表示从文件中搜索单词ftpuser。
v	调用vi编辑器。
q	退less。
!command	调用Shell命令，例如使用“！ls”，表示列出当前目录下的所有文件。

例如，查看当前目录下 test 文件的内容，命令行为：

```
#less　test
```

如果在显示文件 example3 内容的同时，加上行号，命令行为：

```
#less　-N　example3
1　aa bb cc dd
2　bb cc dd ee
3　cc dd ee ff
4　dd ee ff gg
5　ee ff gg hh
6　ff gg hh ii
```

6.11.16　head 命令：查看文件前面的部分内容

cat 命令一次会输出文件的全部内容，而 head 命令则用于查看文件前面的部分内容。head 命令格式为：

```
head　[-n]　文件名
```

其中-n 用于指定显示文件的前 n 行，如果未指定行数 n，则使用默认值 10。

例如，显示 example 文件的前 5 行，命令行为：

```
$ head   -5   example
```

16.tail 命令

tail 命令与 head 命令类似，用于显示文件后面的部分内容，缺省为显示末尾 10 行。命令格式为：

```
tail   [+/-n]   文件名
```

其中 “+ n”表示从文件的第 n 行开始显示。“- n”表示从距文件尾 n 行处开始显示。例如显示文件 test 最后 10 行的内容，可以使用命令：

```
#tail   -10   test
```

例如从文件 test 的第 10 行开始显示文件的内容，可以使用命令：

```
#tail   +10   test
```

6.11.17 touch 命令：改变文件的时间戳

touch 命令用来改变文件的时间戳，如果 file 文件不存在，则创建该文件。例如，使用 touch 命令创建 sample 文件：

```
 # ls -a
.   ..
# touch   example
# ls   -a
.   ..   example
```

使用 touch 文件更新时间戳，命令如下：

```
# ls   -al
drwxrwxr-x 2 root root 4096 07-27 00:47 .
drwxr-xr-x 6 root root 4096 07-27 00:47 ..
-rw-rw-r-- 1 root root       0 07-27 00:47 example     //创建时间为 07-27 00:47
# touch   example
# ls   -l   example
-rw-rw-r-- 1 root root 0 07-27 00:48 example          //修改时间戳为 07-27 00:48
```

6.11.18 sort 命令：对文件中的所有行进行排序

sort 命令用于对文件中的所有行进行排序，并将结果显示在屏幕上。sort 命令的格式如下：

```
sort   [选项]   文件列表
```

- -m：把已经排过序的文件列表合并成一个文件，并送往标准输出。
- -c：检查给定的文件是否排过序。
- -d：按字典顺序排序，可比较的字符仅包含字母、数字、空格和制表符。
- -f：忽略大小写。
- -r：按降序排序，缺省时是升序。

例如对文件 student 安字典进行排序，命令行为：

```
$ cat   student
Tom    A   B   C   D
Mike    B   C   D   E
Mary    C   D   E   F
Jean    D   E   F   G
$ sort   student
Jean    D   E   F   G
Mary    C   D   E   F
```

```
Mike    B   C   D   E
Tom     A   B   C   D
```

sort 命令常和管道（|）联合使用，例如对当前目录的文件按字典顺序进行排列并显示：

```
# ls
anaconda-ks.cfg   Desktop      install.log             man_chage
chage_li          file1.doc    install.log.syslog
# ls  |  -d  sort          //按字典顺序排列
anaconda-ks.cfg
chage_li
Desktop
file1.doc
install.log
install.log.syslog
man_chage
# ls  |  sort  -r              //按逆序排列文件
man_chage
install.log.syslog
install.log
file1.doc
Desktop
chage_li
anaconda-ks.cfg
```

6.11.19　comm 命令：对两个已排序文件逐行进行比较

comm 命令对两个已排序文件逐行进行比较，输出结果由 3 列组成，其中第 1 列表示仅在第 1 个文件的行，第 2 列表示仅在第 2 个文件出现的行，第 3 列表示在 2 个文件中都存在的行。comm 命令格式为：

```
comm  [-[1][2][3]]  file1  file2
```

选项：-[1][2][3]：分别表示不显示输出的列。

例如对文件 student1 和 student2 进行比较，显示其中的异同，命令行如下：

```
# cat  student1
Tom is a tall man
Mike is Tom's brother
Mary is a beautiful girl
Jean is Mary's sister
# cat  student2
Tom is Mary's brother
Mike is Tom's brother
Mary is a beautiful girl
Jean is Tom's sister
# sort student1 > student1_1          //对 student1 排序
# sort student2 > student2_2          //对 student2 排序
# cat student1_1  student2_2
Jean is Mary's sister
Mary is a beautiful girl
Mike is Tom's brother
Tom is a tall man
```

```
Jean is Tom's sister
Mary is a beautiful girl
Mike is Tom's brother
Tom is Mary's brother
# comm    student1_1 student2_2
Jean is Mary's sister
        Jean is Tom's sister
                Mary is a beautiful girl
                Mike is Tom's brother
Tom is a tall man
        Tom is Mary's brother
```

6.11.20　diff 命令：比较两个文本文件，并显示它们的不同

diff 命令比较两个文本文件，并显示它们的不同，命令格式为：

```
diff   文件 1   文件 2
```

diff 文件的输出结果说明见表 6-25。

表 6-25 diff文件的输出结果说明

输出结果形式	说明
n1 a n2	表示第一个文件的 n1行添加了…成为第二个文件的n2行。
n1,n2 c n3,n4	表示将第一个文件的n1到n2行(在<号之后的内容)改变成第二个文件的n3到n4行(在 >号之后的内容)。
n1,n2 d n3	表示第一个文件中n1到n2行删除成为文件2中的n3行。
a表示添加，c 表示改变，d 表示删除	

例如对文件 student1、student2 进行比较，并输出文件的不同之处，命令行为：

```
# cat   student1                  //student1 文件内容
Tom is a tall man
Mike is Tom's brother
Mary is a beautiful girl
Jean is Mary's sister
# cat   student2                  //student2 文件内容
Tom is Mary's brother
Mike is Tom's brother
Mary is a beautiful girl
Jean is Tom's sister
# diff   student1   student2       //比较结果
1c1
< Tom is a tall man
---
> Tom is Mary's brother
4c4
< Jean is Mary's sister
---
> Jean is Tom's sister
```

6.11.21　cut 命令：移出文件中的部分内容

cut 命令用来移除文件中的部分内容。命令格式为：

```
cut　[选项]　file
```

选项：-c：显示每行中指定字符。

例如显示 student1 文件中每行第 1 到第 10 个字符，命令行为：

```
# cut　-c1-10　student1
Tom is a t
Mike is To
Mary is a
Jean is Ma
```

cut 命令并没有改变源文件的内容，只是改变了显示方式，从下面的命令可以看到：

```
# cat　student1
Tom is a tall man
Mike is Tom's brother
Mary is a beautiful girl
Jean is Mary's sister
```

6.11.22　locate 命令：查找所有名字中包含指定字符串的文件

locate 命令用于查找所有名字中包含指定字符串的文件。locate 命令是通过已建立的数据库 /var/lib/slocate 来进行搜所，而不是直接在硬盘当中逐一寻找。因此使用 locate 比使用 find 命令更快、更简便。但因为 locate 是经由数据库来搜寻，而数据库的更新一般是每天一次（多数在夜间进行），所以当数据库更新之前搜寻用户新建立的文件，locate 命令是无法查到的。locate 命令格式为：

```
locate 字符串
```

例如查找所有包含“shadow”字符串的文件名，命令行为：

```
# locate　shadow
/etc/shadow
/usr/bin/pgmdeshadow
/usr/share/icons/gnome/16x16/stock/image/stock_shadow.png
/usr/share/icons/gnome/16x16/stock/text/stock_fontwork-2dshadow.png
...　...
```

6.11.23　split 命令：将多个文件的内容合并到一个文件中

利用 cat 命令可以将多个文件的内容合并到一个文件中。与之相反，split 命令用于将一个较大的文件拆分成相同大小的几个小文件。split 命令的格式如下：

```
split　[选项]　输入文件　输出文件前缀
```

- -l n：将输入文件每 n 行拆分为一个文件，默认值为 1000。
- -b n[bkm]：以字节为单位进行拆分，设定每个拆分后的文件的大小：b 代表 512K，k 代表 1KB，m 代表 1MB
- -：从标准输入读取数据。

例如对文件 file 进行拆分，每两行存为一个新文件，新文件名以“files”为前缀：

```
# ls
file
```

```
# cat file
aaaaaaaaaaaaaa
bbbbbbbbbbbbbb
ccccccccccccc
ddddddddddddd
eeeeeeeeeeeee
fffffffffffff
# split   -l 2   file   files
```

可以看到文件 file 被拆分成了 filesaa、filesab、filesac 三个文件，如下所示：

```
# ls
file   filesaa   filesab   filesac
```

查看文件 filesaa、filesab 和 filesac，可以看到每个文件包含 2 行，如下所示：

```
# more filesa?
::::::::::::::
filesaa
::::::::::::::
aaaaaaaaaaaaaa
bbbbbbbbbbbbbb
::::::::::::::
filesab
::::::::::::::
ccccccccccccc
ddddddddddddd
::::::::::::::
filesac
::::::::::::::
eeeeeeeeeeeee
fffffffffffff
```

使用选项“-b”可以按大小对文件进行拆分。例如文件 file 为 90 字节，按每 30 字节一个文件进行拆分，新拆分后的文件以“filebysize”为前缀，如下所示：

```
# ls   -l   file
-rw-r--r-- 1 root root 90 08-09 00:22 file
# split   -b 30   file   filebysize
```

可以看到文件 file 被拆分成了 filebysizeaa、filebysizeab、filebysizeac 三个文件，如下所示：

```
# ls
file   filebysizeaa   filebysizeab   filebysizeac
```

每个文件各为 30 字节，如下所示：

```
# ls   -l
-rw-r--r-- 1 root root 90 08-09 00:22 file
-rw-r--r-- 1 root root 30 08-09 00:26 filebysizeaa
-rw-r--r-- 1 root root 30 08-09 00:26 filebysizeab
-rw-r--r-- 1 root root 30 08-09 00:26 filebysizeac
```

使用选项“-”可以从标准输入设备读入数据，并按设定值进行拆分。例如从标准键盘读入数据，每 2 行拆分成一个新文件，且新文件的前缀为“standinput”，如下所示：

```
# split   -l   2   -   standinput
aaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaa
bbbbbbbbbbbbbbbbbbbbbbbbbbb
```

```
bbbbbbbbbbbbbbbbbbbbbbbbbb
ccccccccccccccccccccccccc
ccccccccccccccccccccccccc
```

按 Ctrl+d 或 Ctrl+c 结束文件的输入。使用命令 ls 查看读入的数据是否被拆分，如下所示：

```
# ls
standinputaa        standinputab      standinputac
```

查看文件 standinputaa、standinputab 和 standinputac，可以看到每个文件包含 2 行，如下所示：

```
# more standinputa?
::::::::::::::
standinputaa
::::::::::::::
aaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaa
::::::::::::::
standinputab
::::::::::::::
bbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbb
::::::::::::::
standinputac
::::::::::::::
ccccccccccccccccccccccccc
ccccccccccccccccccccccccc
```

可以使用命名管道对标准输入进行重新指定，例如：

```
# ls   -al   /etc   |   split   -l 50   -   etcfile
# ls
etcfileaa   etcfilead   etcfileab   etcfileae   etcfileac   etcfileaf
```

该命令以 ls 长格式列出/etc 目录下的所有文件，并把命令 ls 的输出结果通过命名管道输入 split 命令。由 split 命令按着每 50 行另存为一个新文件，对输入的文件列表进行拆分。新拆分的文件以“etcfile”为前缀，一共生成了 etcfileaa、etcfilead、etcfileab、etcfileae、etcfileac 和 etcfileaf，总共 6 个文件。

6.11.24　IO 重定向及管道操作

在 Linux 系统中，数据流可分为三类：数据输入、数据输出和错误输出。相应地，每一个进程也都有三个特殊的文件描述指针：标准输入（standard input，文件描述指针为 0）、标准输出（standard output，文件描述指针为 1）和标准错误输出（standard error，文件描述指针为 2）。这三个特殊的文件描述指针使进程在通常情况下接收标准输入设备的输入，由标准输出设备显示输出。

标准输入通常指传给一个命令的键盘输入。例如运行命令“ls -al”，其中“-al”是传给 ls 命令的标准输人。标准输出是一个命令的执行结果，例如运行“cat /file1.doc”命令后所看到的文件内容就是标准输出。通常标准输出是被定向到显示器。如果在执行命令过程中发生错误，可能会存在一条错误消息，这是标准错误数据流，通常也被定向到显示器。例如在上例中，如果 file1.doc 文件不存在，则系统会提示“cat：/file1.doc：没有此文件或目录”，并从显示器输出。

有两种基本的方法可以用来重定向标准输入（stdin）、标准输出（stdout）和标准错误（stderr）。可以利用管道把这些数据流之一输送给另外一条命令；也可以利用 I/0 重定向把这些数据流之一重定向到一个文件。管道以及 I/0 重新定向是 LinuX 系统中特有的概念。所谓管道是指将某个命令的输出结果

传送到另一个命令，当成另外一个命令的输入，其代表符号是“|”。 所谓 I/0 重定向是指将命令执行的结果重新导出到其他的设备或文件(以“>”或“>>”来表示)，或是重新导入到其他的设备或文件(以“<”或“<<”来表示)。常用输入输出重定向命令格式与执行结果如表 6-26 所示

表 6-26 常用输入输出重定向命令格式说明

命令行	说明
命令1　>　文件1	将命令1的输出结果重定向到文件1。
命令1　&>　文件1	将命令1的输出结果和标准错误输出一起重定向到文件1。
命令1　>>　文件1	将命令1的输出结果追加到文件1中。
命令1　2>　文件1	将命令1的标准错误输出的结果重定向到文件1中。
命令1　<　文件1	将文件1做为命令1的标准输入。
命令1　<<　字符串1	允许连续输入数据，直到接收到字符串1。

1. 输出重定向(>)

一般在 Linux 下执行任何命令或程序过程中，默认都是将结果输出到屏幕，但是有时希望将标准输出保存到一个文件中，此时可以使用输出重定向的功能。例如对/etc 目录下的所有文件建立一个清单，并保存到/etc_list 文件中，命令行为：

```
# ls  /etc  >  /etc_list
# cat  /etc_list          //查看建立的例表文件
a2ps.cfg
a2ps-site.cfg
acpi
adjtime
...  ...
```

在执行上例中的“ls”命令时，由于所有原来会出现的输出内容都已重新定向到/etc_list 文件中，屏幕不会再显示任何输出结果。重定向的文件，如本例中的/etc_list，不需预先创建，系统在执行此命令中若发现重新定向的文件不存在，则会自动产生命令中指定的文件。若文件已经存在，则会进行写覆盖。可以使用附加输出重定向（>>）把新添加的数据追加到已存在文件的尾部，来避免写覆盖的发生。

利用前面介绍过的“cat”命令，配合输出重定导向操作，可以将数据直接从键盘录入文件中。例如创建 file1.doc 文件并录入“Hello　world !!!”， 按 Ctrl+C 或 Ctrl+d 可以结束文件的编辑。命令行为：

```
# cat  >  file1.doc
Hello World!!!          //按 Enter 键
<按 Ctrl+C>或<Ctrl+d>
#
```

在输入完毕后先按 Enter 键(否则最后一行不会存储)，再按 Ctrl+C 或 Ctrl+d 就可结束文件的编辑，系统会将此内容以 file1.doc 文件名存储。

2. 附加输出重定向(>>)

如果重新定向的文件原来已经存在，那么在使用输出重定向（>）给文件输入新的数据后，新数据将覆盖所有旧的数据。在硬盘空间足够的条件下，为了避免覆盖旧数据，可以采用 Linux 提供的“附加输出重定向”（>>）。附加输出重定向操作可以将新输入的数据附加(Append)在原有旧数据之后，表示符号为“>>”。从下例中可以看出输出重定向与附加输出重定向的区别：

```
# cat file1.doc
Hello World!!!          //一源文件内容
# cat > file1.doc
This is a new world!
# cat file1.doc
This is a new world!    //一源文件内容被覆盖
```

```
# cat >> file1.doc
I like this world.
# cat file1.doc              //一附加在源文件后
This is a new world!
I like this world.
```

3. 输入重定向(<)

输出重定向与输入重定向相反，前者是将命令或程序的执行结果通过屏幕或文件来输出，但是后者却是将输入设备（如键盘或文件）提供给命令来执行。例如下面的程序实现对/etc 目录进行备份，其中使用输入重定向来读入 file 文件的内容：

```
#!/bin/sh
i=0;
find / -name etc > file;
while read LINE
do
   DIRS=$LINE
   BACKUP=”/tmp/backup${i}.tar.gz”
   tar –zcvf   $BACKUP   $DIRS
   i=$(($i+1))
done < file
```

4. 附加输入重定向(<<字符串)

附加输入重定向经常使用在电子邮件系统中，可以让用户自行定义一个字符串，例如 exit，系统在收到此字符串前，会持续地将数据输入文件。下面是一个编写电子邮件的范例，定义的结束字符串是“exit”，用户可以连续输入邮件内容，直到输入“exit”：

```
# mail yang_mh@bit.edu.cn << exit
> This mail is a test.
> exit
```

5. 错误输出重定向(2>)

输出重定向（>）操作在命令执行发生错误时，将错误信息直接显示到屏幕，并不记录到文件中。而错误输出重定向(2>)会在命令执行发生错误时，将错误信息直接输入到文件中，不会再将信息结果显示在屏幕上。例如：

```
# ls
cal_txt      finger2_txt   groups_txt    last_txt   newgrp_txt    suple_txt    who.txt
file1.doc    finger3_txt   groups_txt2   ll         suple3_txt    suple_txt~   w.txt
file2.doc    finger_txt    id_txt        ln         suple3_txt~   w
# ls   exec
ls: exec: 没有那个文件或目录
# ls exec   2> example
# cat example
ls: exec: 没有那个文件或目录
```

6. 标准输出与错误输出重定向（&>）

标准输出与错误输出重定向（&>）可以将标准输出和错误输出信息一并重新定向到文件，屏幕上不会显示任何信息，例如：

```
# ls
cal_txt      file2.doc      finger_txt    id_txt      ln              suple3_txt~   w
example      finger2_txt    groups_txt    last_txt    newgrp_txt      suple_txt     who.txt
```

```
file1.doc   finger3_txt   groups_txt2   ll          suple3_txt   suple_txt~    w.txt
# ls id_txt file
ls: file: 没有那个文件或目录
id_txt
# ls   id_txt file   2>   doc              //将错误输出重定向到文件 doc
id_txt
# cat   doc
ls: file: 没有那个文件或目录
# ls   id_txt file &> doc             //将标准输出与错误输出一并重定向到文件 doc
# cat   doc
ls: file: 没有那个文件或目录
id_txt
```

7. 管道(Pipe)

在 Linux 系统中，管道的主要功能是将其他程序的输出结果直接导出到另一个程序来当成输入数据，即将前一个程序的输出做为后一个程序的输入，符号表示为“|”。管道语法格式为：

```
命令 1  |  命令 2  [ |  命令 3 ... ]
```

可以将标准错误输出一起送入管道，命令格式为：

```
命令 1  |&   命令 2  [ |&   命令 3 ... ]
```

例如将 ps 命令的输出结果作为 more 命令的输入，以实现分页查看进程信息，命令行为：

```
#ps   -aux   |   more
```

例如以长格式查看/etc 目录下的所有文件。由于/etc 目录下的文件有很多，直接使用“1s　-al”命令显示的内容会很快卷过屏幕，无法仔细查看。可以利用管道将“1s　-al”命令的执行结果输入 more 或 less 命令，实现分页显示，命令行为：

```
#ps   -aux   |   less
```

6.12　文件和目录的图形化管理

在 Red Hat Enterprise Linux 5 的 GNOME 桌面环境中，采用了 Nautilus 文件管理器来实现对文件和目录的管理。Nautilus 文件管理器是一个强大的集成化图形工具，除了具有传统的文件、目录管理的功能外，还允许用户配置 Linux 系统、访问网络资源、浏览图片、运行脚本等。

6.12.1　启动 Nautilus 文件管理器

Nautilus 文件管理器可以通过使用下面的任一方法启动：

- 从【位置】菜单中选择【主文件夹】或【root】。
- 双击桌面上的用户主目录。
- 双击桌面上的软盘或光盘图标。
- 双击桌面上的【计算机】图标。
- 在终端窗口中输入“Nautilus”命令。

打开 Nautilus 文件管理器如图 6.10 所示。

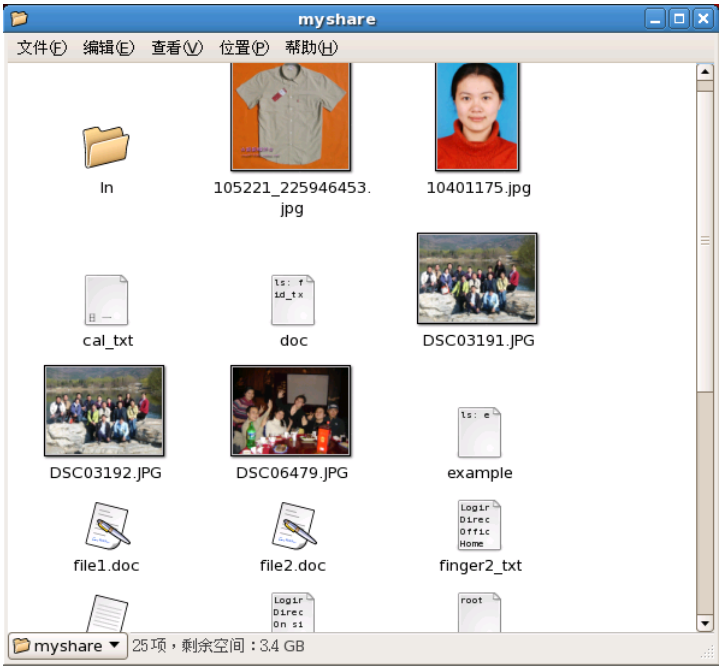

图 6.10 Nautilus 文件管理器窗口

Nautilus 文件管理器有图标和列表两种显示方式，在首次使用时，默认以图标方式进行显示。通过单击【查看】菜单，选择【以列表视图查看】选项，可以打开列表视图方式，如图 6.11 所示

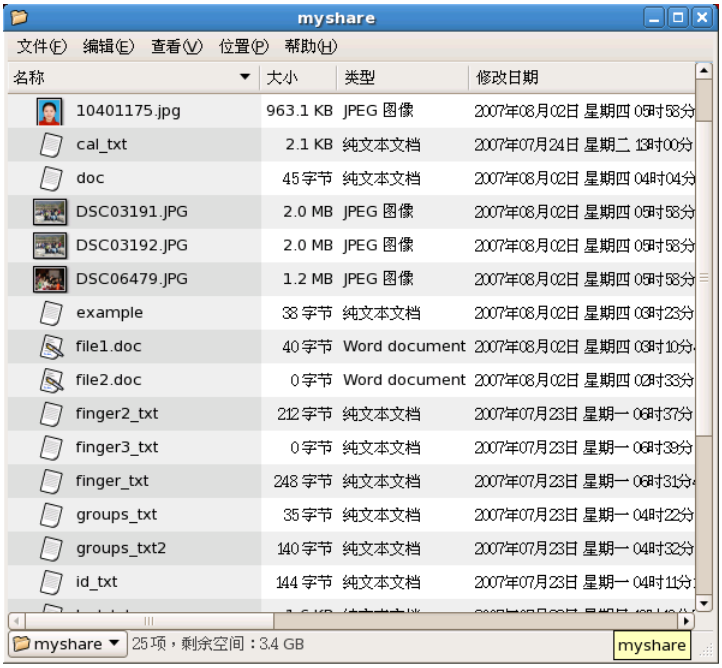

图 6.11　以列表视图显示目录内容

通过选择【查看】菜单中的【排列项目】选项，可以对所显示的内容按名称、文件大小、类型、修改时间等类型进行排列。如图 6.12 所示，按名称显示/root 根目录的内容。

图 6.12　选择排列方式

在【位置】菜单中可以选择新窗口打开的位置，包括主文件夹、计算机、网络、CD/DVD 等，如图 6.13 所示。

图 6.13 Nautilus 位置菜单

在 Nautilus 文件管理器主窗口中，右键单击某一文件或文件夹后可以弹出快捷菜单，如图 6.14（a）所示。在主窗口的空白处右击，可以弹出如图 6.14（b）所示的快捷菜单。

（a）　　（b）

图 6.14 Nautilus 文件管理器的快捷菜单

6.12.2　打开文件和目录

在 Nautilus 主窗口中双击一个文件夹，可以在一个新窗口中打开该文件夹，显示该文件夹中的所有文件和子目录。对于文件，Nautilus 可以直接打开可执行文件、文本文件，BMP、JPEG 等图形文件。可执行文件通过双击可以直接运行；对于文本文件，双击或单击右键并在快捷菜单中选择打开，可以自动调用 gedit 程序打开该文件；对于图形文件，Nautilus 会自动调用 eyeofgnome 程序进行显示。eyeofgnome 是 Linux 下的图片查看工具，又被称做“Gnome 之眼”，可以打开 BMP、JPEG、PNG、PCX、ICO、GIF、XBM、ANI、WBMP 等多种图片格式，可以对图像按比例缩放，旋转。打开一个文件步骤如下：

（1）选择一个文件，然后双击该或单击右键并在快捷菜单中选择打开，系统会自动调用该文件已经关联的应用程序对该文件进行处理。

（2）如果用户不想使用默认关联的方式运行该文件，可以在该文件的右键快捷菜单中选择【其他应用程序】，系统将弹出如图 6.15 所示的【打开方式】选择对话框，其中列出了系统已经安装的所有应该程序。用户可以选择一个应用程序并按【打开】按钮，即可以用所选定的应用程序打开该文件。

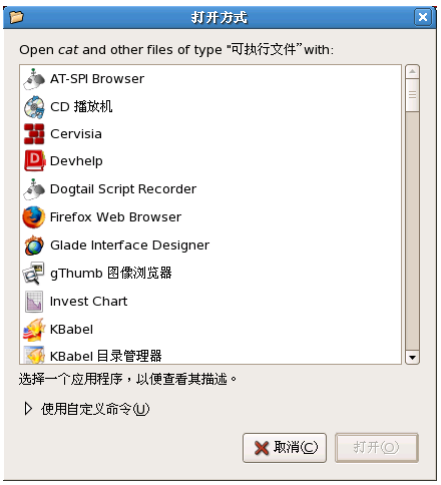

图 6.15　打开方式对话框

（3）若用户未能在图 6.15 所示的列表中找到需要的应用程序，也可以单击图 6.15 中的【使用自定义命令】，然后单击【浏览】按钮，系统会弹出【选择应用程序】对话框，如图 6.16 所示。可以从/usr 或/bin 或其他目录中选择需要的应用程序或命令，也可以在【位置】文本框直接输入应用程序或命令的路径和名称。

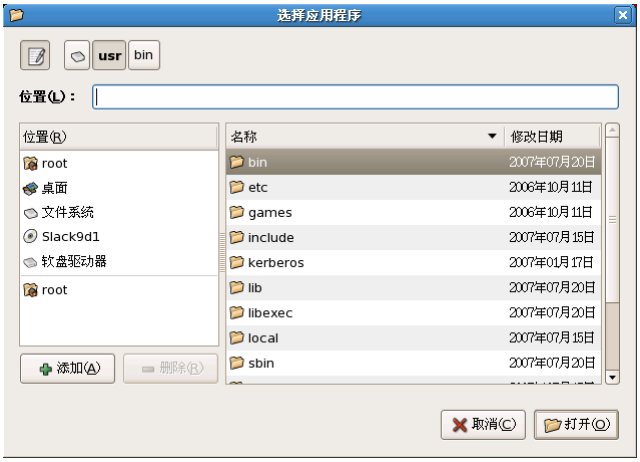

图 6.16 选择应用程序对话框

（4）单击【打开】按钮，即可用指定的应用程序打开该文件。

如果需要更改文件默认关联的应用程序，可以在该文件快捷菜单中（如图 6.14（a）所示）选择【属性】，在弹出的【属性】对话框中选择【打开方式】，然后单击【添加】，系统会弹出【打开方式】对话框，从中选择新的应用程序，完成文件与应用程序的重新关联。

6.12.3　书签

对用户经常访问的文件夹，可以使用 Nautilus 提供的书签功能进行标记，以方便以后的使用。

设置书签只需打开要进行标记的文件夹，在【位置】菜单中选择【添加标签】，如图 6.17 所示。可以看到当前的文件夹作为标签被添加到【位置】菜单中的第三栏中。以后用户可以在任一打开的 Nautilus 窗口中，通过【位置】菜单直接打开标记的文件夹。

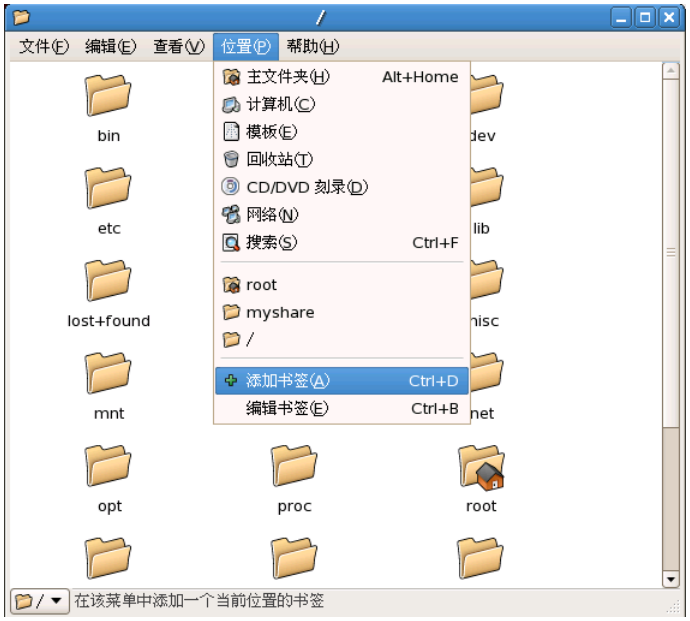

图 6.17 添加书签

对于已经添加的书签，可以使用【编辑书签】进行修改，单击【位置】菜单中【编辑标签】，打开

如图 6.18 所示编辑书签对话框。

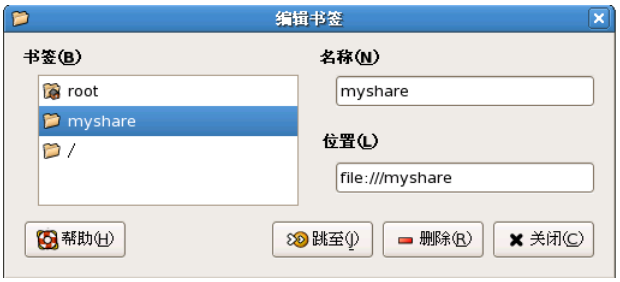

图 6.18 编辑书签对话框

在【编辑书签】对话框中可以修改书签的名称、位置；选择一个书签并按【删除】按钮可以删除该书签；按【跳至】按钮可以打开该书签标记的文件夹。

6.12.4　文件和目录的创建、重命名和删除

在 Red Hat Enterprise Linux 5 中有多种方法可以创建文件或目录，对文件或目录重命名，删除无用的文件或目录，详细操作步骤见表 6-27。

表 6-27 文件和目录的创建/重命名/删除

操作	方法	备注
新建目录或文件	1.在Nautilus【文件】菜单中选择【创建文件夹】或【创建文档】选择【空文件】。 2在Nautilus主视图窗口空白处单击右键，从【快捷菜单】中选择【创建文档】或【创建文件夹】。	新创建的目录系统默认名为“未命令的文件夹”，新创建的空文件系统默认名为“新文件”。
重命令	1.选择需要重命令的文件或目录，在Nautilus【编辑】菜单中选择【重命名】。 2.选择需要重命令的文件或目录，单击右键并在快捷菜单中选择【重命名】 3. 选择需要重命令的文件或目录，按F2键。	在Linux中，无法直接通过单击文件或目录的名称，使其变成可编辑状态。
删除目录或文件	1.选择需要删除的文件或目录，在Nautilus【编辑】菜单中选择【移动到回收站】。 2.选择需要删除的文件或目录，单击右键并在快捷菜单中选择【移动到回收站】。 3.选择要删除的文件或目录，按Del键或Ctrl+T。选择的对象将被送到回收站。 4. 选择要删除的文件或目录，按Shift+Del键可以永久删除文件或目录。永久删除的文件或目录无法从【回收站】中找回。 5. 选择需要删除的文件或目录，然后通过鼠标直接拖放到【回收站】中。 6. 选择需要删除的文件或目录，在Nautilus【编辑】菜单或快捷菜单中选择【删除】。该方法为永久删除。	1.与Windows回收站类似，Linux回收站可以将删除的文件或目录进行恢复，但Linux回收站没有“还原”功能，用户必须手工将预恢复的文件或目录复制到其他位置。 2.在【回收站】的快捷菜单中选择【清空回收站】，将永久删除【回收站】中的文件和目录。

注意：在 Nautilus【编辑】菜单或快捷菜单中的【删除】选项，为永久删除，所删除的内容不会进入回收站，因而无法恢复。如果在 Nautilus【编辑】菜单或快捷菜单中没有【删除】选项，可单击【编辑】菜单中的【首选项】，在弹出的【文件管理首选项】中，选择【行为】选项卡，选择【包含绕过回收站的删除命令】，如图 6.19 所示。

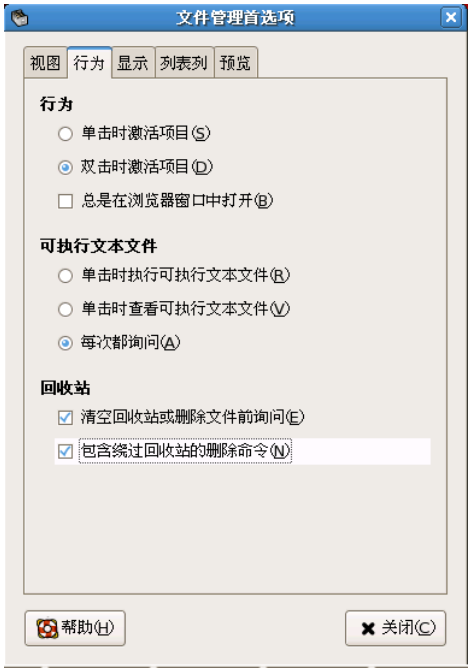

图 6.19 在【编辑】菜单中添加【删除】选项

6.12.5 文件和目录的移动/复制/链接

在 Nautilus 文件管理器中，选择文件或文件夹有多种方法。

- 选择单一文件或文件夹可以用鼠标左键单击该文件或文件夹。
- 选择连续多个文件或文件夹可以按住鼠标左键在目录窗口中拖动，拉出一个虚线框，然后释放鼠标，虚线框内的所有文件或文件夹均会被选定。
- 选择第一个文件或文件夹，然后在最后一个文件或文件夹处同时按住 shift 键并点选，则位与第一个选择与最后一个选择之间的所有文件或文件夹也均会被选中。
- 选择多个不连续的文件或文件夹可以按住 Ctrl 键不放，逐一用鼠标点选，也可以先选择目录中的全部文件和文件夹，然后按住 Ctrl 键不放，逐一用鼠标点选不需要的文件或文件夹。
- 选择文件夹内全部文件可以选择【编辑】菜单中的【全选】，也可以按【Ctrl+a】快捷方式。

可以用拖动鼠标的方法完成文件和目录的移动/复制/链接操作。操作步骤见表 6-28。

表 6-28 使用拖动鼠标方法完成文件和目录的移动/复制/链接

操作	步骤
移动	1.分别用Nautilus文件管理器打开需要移动对像所在目录和目标位置目录窗口。 2.在移动对像所在目录窗口中选择需要移动的对象。 3.按住鼠标左键拖动已选择的动象到目标目录窗口，同时释放鼠标。
复制	1.分别用Nautilus文件管理器打开需要复制对像所在目录和目标位置目录窗口。 2.在复制对像所在目录窗口中选择需要复制的对象。 3.用鼠标左键拖动已选择的动象到目标目录窗口，按住Ctrl键并同时释放鼠标。
链接	1.分别用Nautilus文件管理器打开需要链接对像所在目录和目标位置目录窗口。 2.在链接对像所在目录窗口中选择需要链接的对象。 3.用鼠标左键拖动已选择的动象到目标目录窗口，按住Ctrl+Shift键并同时释放鼠标。
拖动后询问	1.分别用Nautilus文件管理器打开需要操作对像所在目录和目标位置目录窗口。 2.在链接对像所在目录窗口中选择需要操作的对象。 3.用鼠标左键拖动已选择的动象到目标目录窗口，按住Alt键并同时释放鼠标。

	4.在弹出的快捷菜单中选择该操作是移动还是复制或是创建链接。

注意：由于 Ctrl+Shift 是中文输入法默认的切换热键，所以在用 Ctrl+Shift 创建链接时，应使系统处于西文状态下，以免发生冲突。

也可以使用 Windows 中类似的方法，利用剪贴、复制、粘贴方法实现文件或目录的移动和复制。例如移动一个对象可以首先利用 Nautilus 选择待移动对象，然后在窗口编辑菜单或快捷菜单中选择【剪切】，也可以直接使用 Ctrl+X 快捷键。然后用 Nautilus 打开目标位置目录，选择编辑菜单或快捷菜单中的【粘贴】选项，或直接使用 Ctrl+V 快捷键。

文件和目录的复制操作类似。首先利用 Nautilus 选择待复制对象，然后在窗口编辑菜单或快捷菜单中选择【复制】，也可以直接使用 Ctrl+C 快捷键，如图 6.20 所示。然后用 Nautilus 打开目标位置目录，选择编辑菜单或快捷菜单中的【粘贴】选项，或直接使用 Ctrl+V 快捷键。

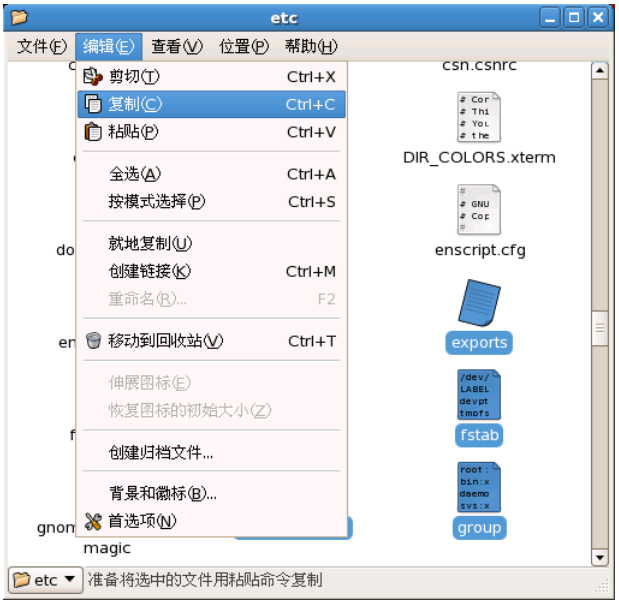

图 6.20　对选择的文件进行【复制】

对于在当前文件夹创建文件或目录的副本除了可以采用上述方法外，还可以选择复制对象后在编辑菜单中选择【就地复制】，系统会自动地为所创建的多个副本取名为“复件”、“另一个复件”、“第 4 个复件”等。

6.12.6　修改文件和目录的属性

对文件或目录的属性进行修改可以通过 Nautilus 来完成。

（1）选择需要修改的文件或目录，在 Nautilus 文件菜单或在右键快捷菜单中选择【属性】，系统将弹出文件或目录的属性设置对话框，如图 6.21 所示。

图 6.21　文件的【属性】对话框

（2）在图 6.21 所示的【属性】对话框中显示了文件的基本信息，包括名称、类型、大小、存放位置、修改时间和最近访问时间。可以在名称文本框中直接修改文件的名称。单击【徽标】、【权限】、【打开方式】、【备忘】或【图像】选项卡，可以逐项进行设置。

（3）如果需要为文件或目录设置【徽标】，单击【徽标】选项卡，打开如图 6.22 所示的徽标选择对话框。

图 6.22【徽标】选项卡

（4）在【徽标】选项卡中列出了 50 种常用的徽标，用户可以根据需要为文件或目录指定所需的徽标。

（5）单击图 6.22 中的【关闭】按钮，返回文件的【属性】对话框，再次单击【关闭】按钮，完成属性的修改。

指定徽标的文件或目录，会在图标的右上角进行显示，如图 6.23 所示。

图 6.23　设定【徽标】后的文件图标

6.12.7　使用软盘、光盘等可移动介质

在使用软盘之前，一定要对软盘进行挂载。首先将软盘插入到软盘驱动器中，双击桌面上的【计算机】图标，打开【计算机】窗口，然后右键单击【软盘驱动器】图标，在弹出的快捷菜单中选择【挂载卷】。挂载后的软盘可以通过双击【软盘驱动器】或者在快捷菜单中选择【打开】，查看到软盘中的内容。当软盘使用完毕，通过右击软盘驱动器图标，在快捷菜单中选择【弹出】，可以卸载软盘。

当用户将光盘插入光驱，系统默认将自动进行挂载，并在桌面显示光盘驱动器图标。双击该图标，系统将调用 Nautilus，打开一个新窗口显示光盘中的内容。用户也可以通过双击桌面上的【计算机】图标，打开【计算机】窗口，双击或在快捷方式中选择【打开】，可以查看光盘中的内容。

当光盘使用完毕，可以通过右击桌面上的光盘驱动器图标或【计算机】窗口中的光盘驱动器图标，在快捷菜单中选择【弹出】，卸载光盘，如图 6.24 所示。卸载后，光盘驱动器图标会自动消失。

图 6.24　卸载光盘

在 Red Hat Enterprise Linux 5 中可以使用【可移动驱动器和介质的首选项】对可移动介质（包括 CD、DVD、数码相机等）的使用进行配置。从【系统】菜单中选择【首选项】，选择【可移动驱动器和介质管理】，打开【可移动驱动器和介质的首选项】对话框，如图 6.25 所示。

图 6.25 可移动驱动器和介质管理

【可移动驱动器和介质的首选项】包括存储、多媒体、相机、PDA、打印机和扫描仪以及输入设备共六个选项卡。其主要配置说明如下：

- 存储：选择【插入时挂载可移动介质】复选框，当一个可移存储设备连接到计算机时可以允许进行自动挂载；选择【插入时浏览可移动介质】复选框，可以允许 CD 或 DVD 插入时自动调用 Nautilus 打开一个窗口并显示光盘中的内容。
- 多媒体：当插入一张音频 CD 时，系统默认会调用 Totem 播放器，自动播放第一个音轨。如果需要插入 DVD 时系统能够自动播放 DVD 视频，需选择【Play video DVD disks when inserted】复选框。
- 相机：选择【连接时导入数码照片】复选框，则当数码相机连接到计算上的 USB 设备时，相机上的存储设备将被自动加载，同时系统会调用 gThumb 图像浏览器。
- PDA：选择【连接时同步 Palm】或【连接时同步 PocketPC】选择，系统在连接 Palm 或 PocketPC 时，会自动调用 gpilotd 或 multisync。

6.13 文件和目录管理常见问题

文件和目录管理是 Linux 系统管理的重要组成部分。下面就使用中经常出现的问题进行分析。

6.13.1 无法卸载已挂载的文件系统

如果用户无法用卸载一个已经挂载的文件系统，并且系统总是提示“device is busy”，说明该文件系统正被某个程序使用。必须先退出正在使用该文件系统的程序，才能完成卸载。可以使用 lsof 命令查看是否有任何程序正在使用该文件系统，命令格式为：

```
#lsof  |  grep 文件系统的挂载点
```

从命令的执行结果可以找出当前使用该文件系统的程序。停止该程序的运行，然后再次卸载该文件系统。

6.13.2　修复损坏的文件系统

由于非正常关机或其他原因造成的对文件系统的损害，可以使用 fsck 命令进行修复。fsck 命令格式为：

fsck [-t　系统类型] [选项]　文件系统

- -a：对/etc/fstab 中列出的所有分区进行检查。
- -c：显示完整的检查进度。
- -v：显示详细信息。
- -r：如果检查有错则询问是否修复。

如果修复一个没有卸载的文件系统，系统将提示如下信息：

```
# fsck  /dev/sda1
fsck 1.39 (29-May-2006)
e2fsck 1.39 (29-May-2006)
/dev/sda1 is mounted.
WARNING!!!  Running e2fsck on a mounted filesystem may cause
SEVERE filesystem damage.
Do you really want to continue (y/n)? no
check aborted.
```

上面的提示表明在一个正被挂载的文件系统上使用 fsck 命令，可能会造成文件系统严重的损坏。系统询问如果继续使用 fsck 命令按“y”，否则按“n”。

在建立服务器时，最好建立两个以上的 Linux 主文件系统。在一个文件系统工作时，另一个文件系统做系统备份。当一个文件系统需要修复时，可以用正常的文件系统引导机器，然后卸载受损的文件系统，并用 fsck 命令进行修复。

如果服务器上只安装了一个文件系统，可以使用支持 fsck 命令的光盘或软盘启动系统，然后用 fsck 命令对该受损文件系统进行检查和修复。

6.13.3　查询设备上采用的未知文件系统

可以使用 blkid 命令对设备上所采用的文件系统类型进行查询。blkid 命令主要用来对系统中的块设备（包括交换分区）所使用的文件系统类型、LABEL、UUID 等信息进行显示。例如查看/dev/sda1 设备所采用的文件系统类型，命令行为：

```
# blkid  /dev/sda1
/dev/sda1: LABEL="/boot" UUID="bc360f28-cae0-45fd-821b-d4b3600e3858" SEC_TYPE="ext2" TYPE="ext3"
```

直接使用 blkid 命令可以列出当前系统中所有已挂载文件系统的类型，例如：

```
# blkid
/dev/mapper/VolGroup00-LogVol01: TYPE="swap"
/dev/mapper/VolGroup00-LogVol00:  UUID="034dae0c-363c-41a1-97e8-b2e6933f7c0a"  SEC_TYPE="ext2"
TYPE="ext3"
/dev/sda1: LABEL="/boot" UUID="bc360f28-cae0-45fd-821b-d4b3600e3858" SEC_TYPE="ext2" TYPE="ext3"
/dev/VolGroup00/LogVol01: TYPE="swap"
```

```
/dev/cdrom: LABEL="TomatoWinXP_SP2_v3.0" TYPE="iso9660"
```

6.13.4　恢复已删除的文件

如果在命令行下误删除了某个文件，该文件是无法从回收站中找回的，必须使用命令进行恢复。如果系统使用的是 ext3，由于 ext3 文件系统在执行删除任务时会清除指向删除文件的信息节点指针，因此一般无法进行恢复，而 ext2 文件系统在删除某一文件时只是对该块标记为未使用，只要该块没有被其他文件所占用，那么可以使用 debugfs 命令试着进行恢复。

恢复前应先将包含要恢复文件的分区进行卸载。例如恢复/boot 目录中的 example 文件，首行使用 umount 命令卸载/boot：

```
#umount   /boot
```

然后运行 debugfs 命令，并在提示符下使用 ls 命令，列出从该目录下删除的文件：

```
#debugfs    /dev/sda1
debugfs 1.39 (29-May-2006)
debugfs:   ls -ld
         2    40755 (2)        0        0      1024    5-Aug-2007 09:43 .
         2    40755 (2)        0        0      1024    5-Aug-2007 09:43 ..
        11    40700 (2)        0        0     12288 15-Jul-2007 19:43 lost+found
      8033    40755 (2)        0        0      1024 15-Jul-2007 20:58 grub
        18   100600 (1)        0        0   2318736 15-Jul-2007 19:59 initrd-2.6.18-8.el
5xen.img
        12   100644 (1)        0        0    868062 27-Jan-2007 03:51 System.map-2.6.18-8
.el5xen
        13   100644 (1)        0        0     61053 27-Jan-2007 03:51 config-2.6.18-8.el5x
en
        14   100644 (1)        0        0     84906 27-Jan-2007 03:51 symvers-2.6.18-8.el5
xen.gz
        15   100644 (1)        0        0   2076303 27-Jan-2007 03:51 vmlinuz-2.6.18-8.e
l5xen
        16   100755 (1)        0        0    608564 27-Jan-2007 04:01 xen-syms-2.6.18-8.e
l5
        17   100644 (1)        0        0    274752 27-Jan-2007 03:12 xen.gz-2.6.18-8.el5
<       0>         0 (1)        0        0         0                              example
debugfs:
```

由于当前使用的是 ext3 文件系统，在“< >”之间显示的文件信息节点号为“0”；如果“< >”之间显示的是非零值，则可以进行恢复。例如<>之间的数字为 211，则可以使用下面的命令恢复 example 文件，命令行如下：

```
#debugfs: dump <211>     /home/teacher1/example_dump.txt
```

通过使用 dump 命令，将 example 文件的内容复制到了/home/teacher1/example_dump.txt 文件里。