第5学时　操作与搜索命令

在本学时教程中，将学习文件和子目录进行建立、拷贝、删除及移动等操作命令。还将学习怎样检索文件以及对文件进行压缩 /解压缩的方法。这些内容建立在前一学时教程的学习基础上，而在本学时教程中学习的命令在今后的学习中也会经常用到。

5.1　操作文件和子目录

使用Linux操作系统与使用其他的计算机操作系统没有什么不同。你会在硬盘驱动器上对文件和子目录进行拷贝、删除或者移动，以便更好地组织信息。本小节将介绍怎样才能既快速又简单地完成这些任务。

虽然Linux操作系统的图形化界面 X窗口系统提供了拖放和多重选择等功能，方便了对文件进行拷贝或者删除等操作，但是在这里学习到的各种命令是构成上述这些操作的基础。即使不在控制台状态下使用 Linux操作系统，了解掌握这些程序的执行原理也是很有必要的。

5.1.1　使用touch命令建立文件

touch命令使用起来很简单，一般会在两种情况下用到它。第一种情况是建立文件，第二种情况是更新文件的修改日期。 touch命令是GNU文件工具包的一个组成部分，另外还有几个命令参数。

如果想使用touch命令建立一个文件，请使用下面的方法：

```
# touch newfile
# ls -l newfile
- r w - r - - r - -          1 bball users       0 Jan 5 12 : 40     newfile
```

正如所看到的， touch命令建立了一个长度或者大小为0的文件，也可以使用下面的命令：

```
# > newfile2
# ls -l new*
- r w - r - - r - -          1 bball users       0 Jan 5 12 : 40     newfile
- r w - r - - r - -          1 bball users       0 Jan 5 12 : 41     newfile2
```

类似与touch命令，上面的操作也建立了一个长度为 0的文件。那么为什么通过命令也可以建立文件的情况下还要使用 touch命令呢？这是因为touch命令可以更新一个文件的建立日期或者时间。甚至可以使用 touch命令把一个文件的建立日期或者时间改为过去或者将来时间，如下所示：

```
# touch newfile2
# ls -l newfile2
- r w - r - - r - -          1 bball users       0 Jan 5 12 : 44     newfile2
```

正如你所看到的，在表示 newfile2文件建立的时间记录上增加了三分钟(从 12：41变成了12：44)。还可以使用touch命令的-t参数加上一个代表年月日时的数字把某个文件的建立日期或者时间设置为一个具体的时间，如下所示：

```
# touch -t 1225130000 newfile2
```

```
# ls -l --full-time new*
- r w - r - - r - -        1 bball users      0 Tue Jan 05 12 : 40 : 14 1999      newfile
- r w - r - - r - -        1 bball users      0 Mon Dec 25 13 : 00 : 00 2000      newfile2
```

使用ls命令的--full-time参数和完整信息格式清单列表可以看到： newfile2现在有一个2000年圣诞节那一天下午一点的建立时间记录(你会发现那是一个星期一)。

touch命令的一个用处是使用在进行备份操作的过程中。在备份一些文件和子目录之前或者之后，使用touch命令就可以更新这些文件的时间记录，这样备份程序在进行下一次备份工作的时候就有了一个可供参考的时间。 touch命令还有另外一个用法，就是在由 cron命令(请阅读第24学时教程“ 使用任务计划实现系统管理自动化 ”中的“ 使用 cron 程序包 ”一节)管理的任务计划程序对文件自动进行整理的时候控制对登录记录文件的删除或者保留。如果把某一个登录文件设定得足够“ 陈旧 ”，它就将被删除掉。而如果更新了它，这个文件就将被保留。

5.1.2 使用rm命令删除文件

rm命令是用来删除文件的。这个命令只有几个简单的参数，但是在使用的时候千万要小心谨慎。为什么呢？因为如果 rm命令删除了某个文件的话，这个文件就不再存在了(虽然还是有可能再恢复某些文本文件，请阅读本学时教程后面介绍 mc命令的时候关于指针的说明)。

经常以根操作员的身份登录进入系统并使用 rm命令已经造成了许多不为人知的不幸与灾难。为什么呢？因为只需要一个简单的命令，就不仅可能会毁掉Linux系统，还可能会毁掉包括DOS分区，RAM卡，活动硬盘等等在内的任何已挂装的其他类型的文件系统，这个命令就是：

```
# rm -fr /*
```

这个带着 -r参数的命令会从根目录(/)开始递归地删除所有的文件和子目录。如果你必须以根操作员的身份运行 Linux操作系统的话，一定要先对系统进行备份，并且要学习第 23学时教程“ 备份和恢复系统 ”。

rm命令可以从命令行上一次删除一个或者几个文件。可以使用下面的几种方法中的任何一种：

```
# rm file
# rm file1 file2 file3
# rm file*
```

上面的第一个命令行删除了一个名称为 file的文件，第二个命令行删除了三个文件，而第三个命令行则删除了当前子目录中文件名以字母 file开头的所有文件。使用rm命令比较安全的办法之一是使用它的 -i交互操作参数，这样在操作过程中会被问到是否真的想删除某个文件，如下所示：

```
# rm -i new*
rm : remove ' newfile ' ? y
rm : remove ' newfile2 ' ? y
```

还可以使用 -f参数强行删除某个文件，如下所示：

```
# rm -f new*
```

如果使用了 -f参数但是没有文件能够匹配 new*格式的时候， rm命令失效，但是不会显示任何出错信息。而且当 rm命令遇到子目录的时候，即使子目录是空的，也不进行删除操作，

并且还会显示出错信息，甚至在使用了 -f参数时也是如此。如下所示：

```
# rm -f temp*
rm : temp : is a directory
rm : temp2 : is a directory
```

但是当你把-f和-r参数一起使用的时候，就可以删除这个子目录和这个子目录下面的所有的文件和子目录，条件是只要你拥有它们或者拥有这个操作的权限；请阅读第 22学时教程“管理文件和文件系统”。-f和-r参数可以象下面这样联合使用：

```
# rm -fr temp*
```

-fr参数使得rm命令的执行情况就象 rmdir命令(一会儿就介绍)一样。使用这个参数时千万要小心！一般来说，如果在 OpenLinux操作系统中删除了一个文件，它就永远消失了！

有些X窗口环境比如KDE或者套装软件比如 StarOffice等等采用提供“回收站”的办法来删除文件 —— 文件并不是真的被删除了，而只是移入到一个临时的子目录中。这是一个安全的、但并非万无一失的对文件进行删除或者恢复的办法。如果确实在恢复文件方面需要帮助，那么最好是使用本学时教程后面将要介绍的 mc命令，也就是Midnight Commander程序。

5.1.3 使用mkdir命令建立子目录

mkdir命令一次可以建立一个或者几个子目录。 mkdir命令还可以只使用一个命令行一次就建立起包括全部的父目录和子目录在内的一个完整的子目录继承结构，对这一点是不是有点意想不到呢。

这个命令，再加上cp命令和mv命令，将会是用来组织资料信息的基本工具之一。现在先来看一些示例。如下所示，一条简单的命令建立起了一个子目录：

```
# mkdir temp
```

还可以使用下面的方法一次建立好几个子目录：

```
# mkdir temp2 temp3 temp4
```

还可以输入下面的内容在子目录temp下再建立一个名为child的子目录：

```
# mkdir temp/child
```

因为子目录temp已经是存在的了(刚才建立了它)，所以上面这条命令是有效的。但是如果输入下面的内容：

```
# mkdir temp5/child
mkdir : cannot make directory ' temp5/child ' : No such file or directory
```

正如所看到的， mkdir命令会提示子目录temp5不存在。如果想使用mkdir命令建立一系列完整的子目录结构，就必须使用它的 -p参数，即父操作参数，如下面的例子中那样：

```
# mkdir -p temp5/parent/child
# tree temp5
temp5
`-- parent
    `-- child
```

```
2 directories, 0 files
```

正如所看到的，mkdir命令不仅建立了子目录temp5，还在它的下一层建立了parent子目录，而子目录parent中又有一个child子目录。

现在应该已经掌握建立子目录的方法了，我们下面来学习如何删除它们。

5.1.4 使用rmdir命令删除子目录

rmdir命令是用来删除子目录的。如果希望删除某个子目录，只需要输入下面的内容：

```
# rmdir tempdirectory
```

但是要注意：这个子目录必须是空的。如果试图删除一个其中还有文件的子目录，就会得到一个下面这样的出错信息：

```
# rmdir temp5
rmdir : temp5 : Directory not empty
```

在上面的例子中，子目录temp5中还有其他的子目录。如果某个子目录的名称代表的其实是一个文件而不是子目录的话， rmdir命令也会显示出错信息。可以先用 rm命令删除这些文件(如果使用的是-fr 参数，千万要小心)，或者先把这些文件转移到其他的地方，或者还可以使用mv命令给子目录改个名称。 mv命令我们将在下面的小节讨论。

类似于mkdir命令，rmdir命令也有一个-p参数。可以使用这个参数来删除某个子目录的全部继承结构，如下所示：

```
# rmdir -p temp5
rmdir : temp5 : Directory not empty
```

怎么回事！好象没起作用，那么输入下面的内容：

```
# rmdir -p temp5/parent
rmdir : temp5/parent : Directory not empty
```

啊，好象还是不行，再试试下面的内容：

```
# rmdir -p temp5/*
rmdir : temp5/parent : Directory not empty
```

这可是让人有些恼火了！再试一次：

```
# rmdir -p temp5/parent/child
```

成功了！正如你所看到的，必须指明某个子目录完整的结构才能够删除它。如果使用了上面同样的命令但是没有加上 -p参数，那么就只有子目录 child被删除。那么对于有两个或者更多个子目录的情况又会怎么样呢？如下所示：

```
# mkdir -p temp5/parent/child
# mkdir temp5/parent/child2
# tree temp5
temp5
`-- parent
    |-- child
    `-- child2

3 directories, 0 files
```

为了删除子目录temp5整个的子目录结构，必须使用下面的命令：

```
# rmdir temp5/parent/*
```

现在应该已经掌握建立和删除子目录的操作了。下面，我们来学习 mv命令，可以使用这个命令对文件和子目录进行移动或者改名操作。

5.1.5 使用mv命令给文件改名

mv命令，也就是文件名更改命令，可以对文件或者子目录的名称进行更改 —— 但是它也可以被称为文件移动命令并可以用来在文件系统内移动文件或者子目录。

事实上，从技术观点出发，那些文件或者子目录并没有被真正地移动。如果一定要了解其中的细节，可以从Linux操作系统系统管理员指南(简称SAG)中阅读到有关的细节， SAG可以在下列站点上找到：

http：//metalab.unc.edu/LDP/LDP/sag/index.html

想不想在线阅读SAG？应该可以找到一份安装在OpenLinux操作系统上的拷贝。试试使用一下lynx网页文本浏览器程序，如下所示：

lynx /usr/doc/LDP/system-admin-guide/sga-0.6-html/sag.html

如果想了解更多关于如何使用lynx浏览器程序的资料，请阅读第13学时教程“ 因特网下载与浏览 ”。

mv命令最普通的用法是更改文件名，如下所示：

```
# touch file1
# mv file1 file2
```

上面的命令把文件file1改名为文件file2。除了更改文件名之外， mv命令还可以用来更改子目录名而不管这个子目录是空的还是存有文件。比如说，即使使用 mkdir命令建立了一个完整的子目录结构，仍然可以使用mv命令更改那个新的顶层子目录的名字，如下所示：

```
# mkdir -p temp/temp2/temp3
# mv temp newtemp
```

mv命令有9个不同的参数，但是本小节只讨论其中最常用的两个，-b(备份)和-i(交互操作)。这两个参数可以让你相当安全地使用 mv命令 —— 因为mv命令不仅可以用来更改名称，还可以悄无声息地迅速进行文件覆盖(根本不提示你)！第一个参数 -b在把某个文件或者子目录的名字改为其他文件或者子目录已经使用过的名字的时候，将会对所有原有的文件或者子目录进行备份，如下所示：

```
# touch uno deux tres
# ls uno deux tres
deux  tres  uno
# mv uno deux
# ls uno deux tres
ls: uno: No such file or directory
deux  tres
```

正如你所看到的，在没有使用 -b参数的情况下， mv命令不仅仅是把文件 uno改名为文件deux，还在操作过程中删除了文件deux。很危险吧？现在试一试-b参数：

```
# touch uno
# ls uno deux tres
deux  tres  uno
# mv -b uno deux
# ls deux* tres
deux   deux~  tres
```

上面的例子显示虽然文件uno已被改名并取代了文件deux，但是已经生成了文件deux的一个备份，这个备份文件有一个缺省的波浪号(~)后缀。

mv命令在执行的时候是没有命令回显的，因此应该像对待 rm命令一样，给它加上一个 -i

参数，如下所示：

```
# touch file2 file3
# mv file2 file3
# touch file2
# mv -i file2 file3
mv: replace `file3'? y
```

在上面的例子中，建立了两个文件，然后把文件 file2改名为文件file3，并且这样做的结果就是删除了文件file3。接着，使用了-i参数，mv命令就会询问是否真的想覆盖文件 file3。如果没有发生覆盖，即使使用了 -i参数mv命令也不会要求核实。还可以把 -i和-b参数一起使用，如下所示：

```
# mv -bi file2 file3
```

现在已经学会了对文件的删除、改名和移动操作，那么怎样才能对文件进行拷贝操作呢？

5.1.6 使用cp命令进行拷贝操作

cp命令，即拷贝命令，是用来对文件或者子目录进行拷贝操作的。这个命令有将近 40个命令行参数。本小节中我们不准备全部介绍这些参数，但是将学习其中几个最常用的参数的使用方法，它们可以节省时间，减少麻烦。

第一次使用cp命令的时候，用到的可能是它最简单的形式，如下所示：

```
# cp file1 file2
```

上面的命令把文件file1拷贝到文件file2，同时file1还依然存在。但是在使用cp命令的时候必须要小心，因为在把一个文件拷贝到另外一个文件上的时候，会有可能完全覆盖掉原来的文件。在这一点上， cp命令和mv命令是一样的。为了说明这种情况发生的过程，请先使用 cat 命令建立三个文件，并在每个文件中加入一行文本：

```
# cat > file1
this is file1
# cat > file 2
this is file2
# cat > file3
this is the third file
# ls -l file*
- r w - r - - r - -        1 bball users      14 Jan 5 13 : 29    file1
- r w - r - - r - -        1 bball users      15 Jan 5 13 : 29    file2
- r w - r - - r - -        1 bball users      23 Jan 5 13 : 29    file3
```

现在，拷贝一个文件到另外一个文件，再检查文件的长度和新文件的内容，如下所示：

```
# cp file1 file2
# ls -l file*
- r w - r - - r - -        1 bball users      14 Jan 5 13 : 29 file1
- r w - r - - r - -        1 bball users      14 Jan 5 13 : 31    file2
- r w - r - - r - -        1 bball users      14 Jan 5 13 : 29    file3
# cat file2
this is file1
```

很明显，文件file1取代了文件file2。为了避免这类问题的发生(除非真的想覆盖那个文件)，可以像在mv命令中那样使用-i和-b参数，示例如下：

```
# cp -i file1 file2
cp : overwrite ' file2 ' ? n
```

```
# cp -bi file1 file2
cp : overwrite ' file2 ' ? y
# ls file*
file1file2      file2~      file3
```

请注意已经被覆盖的文件file2已经有了备份。

cp命令还可以用来一次拷贝多个文件。下面的例子说明了如何把子目录 tempdir1中所有的文件拷贝到子目录tempdir2中去：

```
# cp tempdir1/* tempdir2
# tree tempdir2
tempdir2
¦-- temp1file1
¦-- temp1file2
`-- temp1file3

0 directories, 3 files
```

类似于rm命令，cp命令也有一个-r参数。可以使用这个参数把一个子目录拷贝到另外一个子目录中去。举例来说，如果想把子目录 tempdir1及其中的文件拷贝到子目录 tempdir2中去，请使用下面的方法：

```
# cp -r tempdir1 tempdir2
# tree tempdir2
tempdir2
¦-- temp1file1
¦-- temp1file2
¦-- temp1file3
`-- tempdir1
    ¦-- temp1file1
    ¦-- temp1file2
    `-- temp1file3

1 directory, 6 files
```

最后，cp命令还有一个-p参数，它的作用类似于 mkdir命令中的-p参数。通常，当把存放在几个子目录中的文件拷贝到另外一个子目录中去的时候，只有那些文件被拷贝过去了。下面的例子就只把文件temp1file1拷贝到了子目录tempdir3中去：

```
# tree tempdir2
tempdir2
¦-- temp1file1
¦-- temp1file2
¦-- temp1file3
`-- tempdir1
    ¦-- temp1file1
    ¦-- temp1file2
    `-- temp1file3

1 directory , 6 files
# cp tempdir2/tempdir1/temp1file1 tempdir3
```

但是如果想把文件连同它的子目录结构一起拷贝过去，那么该怎样做呢？这个时候就需要使用-p参数，如下所示：

```
# cp -P tempdir2/tempdir1/temp1file1 tempdir3
# tree tempdir3
tempdir3
`-- tempdir2
    `-- tempdir1
        `-- temp1file1

2 directories, 1 file
```

在上面的例子中，cp命令不仅拷贝了指定的文件，还建立了同样的子目录结构。

5.1.7 使用ln命令建立硬链接和符号链接

Linux操作系统同时支持硬链接和符号链接。 Linux操作系统中的链接到底是怎样工作的这一点是否了解并不重要，但是必须了解这两种形式的链接之间的区别以及在使用 Linux操作系统的过程中如何去使用链接。建立硬链接和符号链接都需要使用 ln命令。

新术语 ln命令可以建立两种形式的链接并把文件与其他的文件(包括子目录，因为在OpenLinux操作系统中子目录也是简单的文件)链接在一起。两者之间重要的区别是“**硬链接**”直接链接两个同时保存在相同的文件系统上的文件。而另一方面，可以使用“**符号链接**”用来在扩展到不同的文件系统上的子目录或者文件之间建立链接。硬链接还有其他方面的优势。如果使用ln命令建立了一个硬链接，就必须在命令行中明确指定另外一个文件，然后就可以使用这个文件来对应那个原始文件，如下所示：

```
# cat > file1
This is file1
# ln file1 file2
# ls -l file*
-rw-r--r--   2 bball    users          14 Jan  5 13:32 file1
-rw-r--r--   2 bball    users          14 Jan  5 13:32 file2

# cat file2
This is file1
```

会看到文件file2和文件file1是完全相同的。如果你删除了文件 file1，文件file2依然存在。如果修改了文件file1，比如添加了文本，这些修改都会反映到文件 file2中；而如果修改了文件file2，文件file1也会随之更新。虽然可以看到两个文件，每个文件的长度都是十四个字符，但是在硬盘驱动器上只有原始文件占用了十四个字符的空间(当然，从技术角度讲会不止这几个字符的长度，但是这取决于该硬盘分区的块长度或者是硬盘的类型)。

另一方面，虽然符号链接很有用，但是它们也存在一个不足。下面的例子说明了为什么会这样。首先，使用ln命令的-s参数建立一个符号链接：

```
# cat >file1
This is file1
# ln -s file1 file2
# ls -l file*
-rw-r--r--   1 bball    users          14 Jan  5 13:48 file1
lrwxrwxrwx   1 bball    users           5 Jan  5 13:48 file2 -> file1
```

请注意那个从文件file2指向文件file1的箭头。它告诉你文件file2是文件file1的一个符号链接。还要注意的是文件 file2的长度比文件file1的要短。符号链接和硬链接的区别是符号链接只不过是指向原始文件的一个alias(假名)而已。如果删除了符号链接，原始文件不会发生任何变化。而一旦你删除了原始文件，符号链接一点也帮不上什么忙：

```
# rm -f file1
# cat file2
cat : file2 : No such file or directory
```

因为原始文件 —— 也就是文件file1，不存在了，所以也就无法通过符号链接 —— 即文件file2，存取其中的内容。但是符号链接比起硬链接来确实还有一个优势：可以使用一个符号链接指向OpenLinux文件系统(甚至可以是其他类型的文件系统)中的某个子目录。在下面的

例子中，如果打算建立一个对子目录 /usr/local/games的硬链接，ln命令会显示出错信息并退出：

```
# ln /usr/local/games play
ln : /usr/local/games : hard link not allowed for directory
```

但是可以使用一个符号链接，如下所示：

```
# ln -s /usr/local/games play
# ls -l play
lrwxrwxrwx    1 bball    users      16 Jan 5 14 : 12 play -> /usr/local/games
```

现在，不必键入一个向下面那么长的命令：

```
# cd /usr/local/games
```

而只需要使用：

```
# cd play
```

命令就可以进入子目录/usr/local/games了。

使用其他类型的文件系统(比如DOS或者Windows分区)上的其他子目录的符号链接可能还是很方便的。举例来说，如果从用户子目录中有一个叫做dos的子目录是到/mnt/dos/windows/desktop的符号链接，那么就可以使得在Linux操作系统和Windows两者之间互相拷贝或者移动文件的操作变得非常容易。比如说，假定想在用户子目录中建立一个名为 dos的链接子目录，并且DOS文件系统已经挂装在子目录 /mnt/dos上的话。那么首先检查DOS文件系统确实已经挂装上了，然后再使用 ln命令按照下面的方法建立一个符号链接，如下所示：

```
# ln -s /mnt/dos/windows/desktop  dos
```

请阅读第22学时教程中关于安装其他文件系统的详细资料。

看到这里，已经学习了使用命令行的方法。如果对通过更为图形化的界面对文件进行操作的方法比较熟悉的话，应该会喜欢上后面将要介绍的 mc程序。

5.1.8 使用Midnight Commander程序处理文件

mc程序的全称是Midnight Commander，它是一个对文件进行管理的图形化界面(如图 5-1所示)。它是一个可视化的 shell(将在下一学时教程“使用 shell”中学习到更多关于 shell的知识)。如果想运行mc命令，请在命令行上键入下面的内容：

```
# mc
```

本小节不准备讨论mc程序所有的细节。下面是它的一些主要特点：

- 提供了同时对两个子目录的可视化接口
- 允许使用鼠标操作对子目录中的文件进行浏览
- 允许带有对话框、鼠标、键盘和功能键的菜单驱动的文件操作
- 有一个开放的命令行允许使用 shell中的命令
- 使用鼠标操作执行命令
- 具备扩展的、内建的超文本屏幕帮助画面
- 仿真并支持ls、cp、ln、mv、mkdir、rmdir、rm、cd、pwd、find、chown、chgrp以及

tree命令

- 可以比较子目录中的内容
- 使用定制菜单，便于建立自己的命令
- 可以使用网络链接进行telnet或者FTP操作(参见第13学时教程)
- 提供鼠标操作的文件解压缩操作(参见本学时教程后面的“使用 gzip命令压缩文件”一节)
- 当Linux文件系统配置有删除恢复支持功能时，可以对被删除的文件进行删除恢复操作

Midnight Commander是一个用来对文件进行管理和对子目录进行切换的简单易用的工具程序。必须花费一些时间去学习这个程序的使用方法，但是如果曾经使用过类似的文件管理界面，那么你的感觉将会是宾至如归。

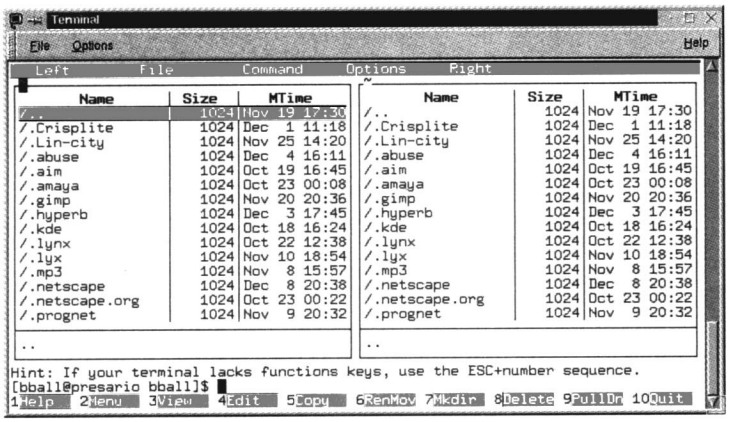

图5-1 Midnight Commander的可视化shell显示出来的Linux文件操作命令图形界面

5.2 搜索文件

本小节将介绍复杂的通配符、即规则表达式的使用方法以及一些使用 grep类的程序进行文件搜索的小例子。如果能够掌握并使用这些表达式，也就掌握了可以反复运用的精细的查询技巧，在进行工作的时候就会节省很多的时间和精力。而在学习过程中付出的代价会在使用Linux操作系统的日子里给你以满意的回报。

5.2.1 什么是规则表达式

新术语 “规则表达式”是使用特殊的语法对字符串(除非是在对文件名进行搜索，字符串一般都在文件的文本之中)进行匹配的字符串格式。另外还有一种扩展规则表达式，它可以在搜索模式字符串中使用额外的操作符。但是两者之间的区别——对语法来说这很重要——应该不会使你对学习如何构造那些能够与你希望查询的内容精确地匹配的字符串格式的方法打退堂鼓。在文件中对某些文本进行查询的时候，使用规则表达式很重要；而在执行可能会造成意外损失的任务(比如在系统中对多个文件进行删除操作)的时候，使用规则表达式就更为关键了。

可以仅仅使用一个很小的格式字符集就构造出无数的规则表达式。表 5-1就是这些格式字符的一个简短的清单。从以前学习过的例子中至少应该已经熟悉了其中的一个(星号“ * ”)。

表5-1 常用规则表达式

表达式	匹　配
*	任意字符
?或者.	一个字符
{ x }	前导字符x个
{ x , y }	前导字符至少x个，但是不超过y个
{ x , }	前导字符至少x个
(xxx) ¦ (XXX)	字符串xxx或者XXX
[xxx]或者[x-x]	在方括号中的字符范围内中的某个字符
[XYZ]+	X、Y\Z至少一次
\x	? 或\等符号字符
^ pattern	一行开头的字符排列形式
$ pattern	一行结尾的字符排列形式

上面只是一个格式字符的简明清单。如果你想了解更多的详细资料，你可以阅读 ed命令的使用手册页(ed命令将在第14学时教程“ 文字处理程序 ”中讨论)。现在举几个使用了不同格式字符的简单例子。

星号在查找对任意字符的匹配时非常有用。例如，如果想在子目录中查找全部扩展名为.txt的文本文件，可以使用下面的方法。

```
# ls *.txt
14days.txt     96hours.txt        datalog.txt        datebook.txt       day67.txt
```

但是想找出子目录中所有的文件名中带有数字的文件的时候该怎么办呢？可以试试在 ls命令行中使用字符串的多个字符匹配的办法，如下所示：

```
# ls *0* *1* *2* *3* *4* *5* *6* *7* *8* *9*
08100097.db       14days.txt        backup001.file    phonelog.111597
08100097.db       32days.msg        day67.txt         phonelog.111597
08100097.db       32days.msg        day67.txt         phonelog.111597
08100097.db       96hours.txt       message.76
08100097.db       96hours.txt       message.76
14days.txt        backup001.file    phonelog.111597
```

很明显，上面显示的并不是你所希望的结果，因为这个多重搜索输出了许多重复的文件名。如果想准确地查找到想要的文件，就需要使用规则表达式来告诉 ls命令列出文件名中带有数字的任何一个文件。如下所示：

```
# ls *[ 0123456789 ]*
0001file.0009          32days.msg          day67.txt
08100097.db            96hours.txt         message.76
14days.txt             backup001.file      phonelog.111597
```

这样就显示出了全部文件名中带有数字的文件，因为已经指定一个字符范围，对上面的例子来说就是搜索字符格式串中的数字。还可以简化这个规则表达式，这样采用短一点儿的表达式就可以完成同样的操作，如下所示：

```
# ls *[ 0 - 9 ]*
0001file.0009          32days.msg          day67.txt
08100097.db            96hours.txt         message.76
14days.txt             backup001.file      phonelog.111597
```

在表达式中如何指定搜索格式字符串的方式是很重要的。如果只想列出那些文件名以数字结尾的文件，可以使用下面的方法：

```
# ls *[ 0 - 9 ]
```

```
0001file.0009          message.76          phonelog.111597
```

如果只想列出那些文件名以数字开头的文件，可以使用：

```
# ls [ 0 - 9 ]*
0001file.0009          08100097.db          14days.txt      32days.msg      96hours.txt
```

下面是个有趣的练习：如果只想列出那些文件名中间有数字或者那些文件名两头有数字的文件该怎么办呢？试试下面的方法：

```
# ls *[ -a-z ]* [ 0-9 ]*
backup001.file          day67.txt
# ls [ 0-9 ]*[ a-z ]*[ 0-9 ]
0001file.0009
```

最后，当需要在格式字符串中含有格式匹配字符时又该怎么办呢？很简单！使用反斜线引导这个字符就可以了，如下所示：

```
# ls *\?*
cathy?.message
```

正如你所看到的，使用规则表达式需要多加练习，但是你的付出会有好的回报的。可以使用其他的表达式试试看能否得到和本小节中给出的例子相似的结果。

5.2.2 使用grep命令在文件内进行搜索

本小节介绍grep命令家族。将会学到grep、egrep、和fgrep等命令。如果想使用这几个命令，就必须掌握前面讨论过的格式匹配技巧。需要使用这几个命令在文件中进行查找并抽出文本。这几个程序命令中的每一个都会查找文件的每一行。可以搜索一个文件或者一组文件。

grep命令家族中的各个命令基本上都差不多，都有二十来个不同的命令行参数。它们真正的区别在于egrep命令使用了稍微不同的语法进行格式匹配，而 fgrep命令使用的是固定字符串。将看到每个命令使用它的一些常见的参数进行操作的例子。如果希望了解这几个程序在文本格式匹配方面的细节，请阅读grep命令的使用手册页。

为了说明每个程序的搜索格式语法的不同，将使用由 Matt Welsh编写的《Linux基本安装指南》(从站点：http://metalab.unc.edu/LDP，或者OpenLinux系统子目录/usr/dos/LDP中可以找到这本书)这本书，对这本书的内容按照不同的匹配格式进行查找。例如，如果想在这本书中找出所有以数字开头的行，可以使用下面的语法：

```
# grep ^[0-9] guide.txt
1 Introduction to Linux                                    1
2 Obtaining and Installing Linux                          40
3 Linux Tutorial                                          85
4 System Administration                                  137
...
# egrep ^[0-9] guide.txt
1 Introduction to Linux                                    1
2 Obtaining and Installing Linux                          40
3 Linux Tutorial                                          85
4 System Administration                                  137
...
# fgrep ^[0-9] guide.txt
```

可以看到grep命令和egrep命令返回了搜索结果(只留了四行表示一下，其他的都省略了)。注意， fgrep命令不能处理规则表达式。在 fgrep命令中必须使用固定字符串格式，如下所示：

```
# fgrep friend guide.txt
Large extent by the window manager . This friendly program is in
copy Linux from a friend who may already have the software , or share
( Unfortunately , the system was being unfriendly .)
```

现在使用egrep命令在文件中查找所有包含字符串 (b)的行：

```
# egrep " \([b]\) " guide.txt
( see section 1.8 for a list of compatible boards ), or (b) there is an
connect to the network , or (b) you have a " dynamic " IP address,
```

可以看到文件中有两行文本中有字符串 (b)。现在来看看使用 grep命令进行搜索会有什么结果：

```
# grep  " \([b]\) " guide.txt
This is version 2.2.2 of the book , " Linux Installation and Getting
to PostScript printers . This document was generated by a set of tools
from LaTeX source , so there may be a number of formatting problems .
This is not the " official " version of the book ! Please see
. . .
```

哎呀，不是想要的，对吧？正如你所看到的， grep命令和egrep命令使用的语法是不一样的。但是可以使用一个比较简单的方法：

```
# grep " (b) " guide.txt
( see Section 1.8 for a list of compatible boards ), or (b) there is an
connect to the network , or (b) you have a " dynamic " IP address,
```

上面的搜索格式串可以用在 grep命令和fgrep命令中。如果在egrep命令中使用了这个格式串，就会得到与在grep命令中使用扩展规则表达式(每行都有一个字符串(b))同样的结果。

每个grep命令程序都能够接受几乎相同的命令行参数。其中常用的一个是 -n，即显示行号参数。因为能够看到文件中到底有哪些行里包含有匹配的字符串，所以它是非常方便的。这个例子适合每个grep命令程序：

```
# egrep -n " friend " guide.txt
1242 : large extent by the window manager . This friendly program is
1942 : copy Linux from a friend who may already have the software , or
5161 : ( Unfortunately , the system was being unfriendly .)
```

可以看到在第1242、1942和5161行找到了格式字符的匹配。这些程序的另外一个特点是在开始每次搜索时不必重新输入搜索字符格式串。举个简单的例子，如果需要在文件中反复搜索不同的单词，可以先把这些单词放在一个文件中供 grep命令使用。首先，要建立一个文本文件，然后使用-f参数来指定这个文件，如下所示：

```
# cat > mywords
wonderful
Typewriter
War
# grep -nf mywords guide.txt
574:Typewriter Used to represent screen interaction, as in
617:software since the original Space War,  or, more recently, Emacs
1998:Now you must be convinced  of how wonderful Linux  is, and all
2549:inanimate object is a wonderful  way to relieve the occasional
3790:    Warning:  Linux  cannot currently  use 33090  sectors of
7780:to wear the magic hat when it is not needed, despite wonderful
10091:wonderful programs and configurations are available with a bit
```

因为同时还使用了 -n显示行号参数，所以必须注意把它放在 -f参数之前，否则grep命令就会显示出错信息，说明它无法找到文件 n，然后退出运行。

可以使用grep命令的-F参数使它能够模仿 fgrep命令的执行情况；或者使用 -E参数模仿egrep命令的执行情况。还可以在系统上找到一个独特的 grep命令家族成员， zgrep命令，可以使用它来搜索压缩文件，而这正是我们下一小节的话题。

5.3 压缩和解压缩文件

本小节介绍对文件进行归档和压缩操作的基本知识。关于如何使用本小节介绍的这些程序进行系统管理或者进行系统备份的详细资料请阅读第23学时教程。我们下面介绍怎样才能建立自己的档案文件并节省磁盘空间。

5.3.1 使用磁带文件归档命令建立档案文件

tar磁带文件归档命令程序出现在还没有软盘驱动器、硬盘和光盘驱动器的计算机早期阶段。那时软件的发行和备份都需要大卷的磁带，计算机上运行的头几个程序中就得有一个是磁带的阅读程序。随着时间的推移， tar命令证明了它自己在运送计算机文件方面非常方便的优点，而许多用于 Linux操作系统的程序就是打包为 tar档案文件的形式(OpenLinux操作系统CD-ROM光盘使用rpm程序来打包文件，请阅读第22学时中的内容)。

使用tar命令可以生成一个包含有多个子目录和多个文件的档案文件。在系统上安装的 tar命令版本还支持一个-z参数，这样就可以使用gzip程序来压缩文档(gzip命令在本学时教程后面讨论)。

tar命令有非常非常多的参数，但是使用起来并不困难。可以快速而又简单地为任何需要的子目录生成档案文件。

首先，我们来建立一个有三个文件的子目录，再建立一个有另外三个文件的子目录，如下所示：

```
# mkdir mydir
# cd mydir
# touch file1 file2 file3
# mkdir mydir2
# cd mydir2
# touch file21 file22 file23
# cd ../..
# tree mydir
mydir
¦-- file1
¦-- file2
¦-- file3
`-- mydir2
    ¦-- file21
    ¦-- file22
    `-- file23

1 directory, 6 files
```

现在已经有了一个子目录和其中的文件，我们使用这个命令的 c(生成)和f(文件)参数来生成一个tar档案文件：

```
# tar cf mydir.tar mydir
# ls -l *.tar
- r w - r - - r - -          1 bball    users      10240 Jan 5 15 : 01 mydir.tar
```

请注意原来的子目录并没有发生变化。在缺省的情况下， tar命令不会删除原来的子目录

和文件。如果真的想这么做的话，可以使用它的 --remove-files参数，但是我们不推荐这样的做法。如果想看看命令执行的过程，可以使用 v参数，如下所示：

```
# tar cvf mydir.tar mydir
mydir/
mydir/file1
mydir/file2
mydir/file3
mydir/mydir2/
mydir/mydir2/file21
mydir/mydir2/file22
mydir/mydir2/file23
```

tar命令在进行操作的时候就会把正在添加到档案文件中去的子目录和文件的文件名显示出来。这是否意味着必须把子目录中所有的文件都进行归档呢？不是！可以使用 w参数，即交互参数，这样tar命令在它执行的过程当中会询问是否想加入每个文件。当想有选择地备份内容不多的子目录的时候，这样就非常地方便，如下所示：

```
# tar cwf mydir.tar mydir
add mydir?y
add mydir/file1?n
add mydir/file2?y
add mydir/file3?n
add mydir/mydir2?y
add mydir/mydir2/file21?y
add mydir/mydir2/file22?n
add mydir/mydir2/file23?y
```

在上面的例子中，没有把文件 file1、file3、和文件file22归档。但是如何保证这一点呢？办法是使用tar命令的两个参数， t参数列出档案文件中的内容， f参数定义操作所使用的 tar档案文件，如下所示：

```
# tar tf mydir.tar
mydir/
mydir/file2
mydir/mydir2/
mydir/mydir2/file21
mydir/mydir2/file23
```

需要注意的是，如果参数的顺序放错了(就象学习 grep命令时看到的例子那样)， tar命令会显示出错信息并退出。现在已经了解了如何建立档案文件和查阅档案文件中的内容，我们下面来看看如何来释放整个的档案文件或者其中的某一个文件。如果想释放其中所有的文件，可以同时使用-x释放参数和-f。为了了解命令执行的过程，我们还可以再加上 -v参数：

```
# tar xvf mydir.tar
mydir/
mydir/file2
mydir/mydir2/
mydir/mydir2/file21
mydir/mydir2/file23
```

如果只是想从档案文件中释放几个文件的话，可以再次使用 w参数：

```
# tar xvwf mydir.tar
extract mydir/?y
mydir/
extract mydir/file2?y
mydir/file2
extract mydir/mydir2/?y
mydir/mydir2/
extract mydir/mydir2/file21?y
mydir/mydir2/file21
extract mydir/mydir2/file23?y
mydir/mydir2/file23
```

在上面的例子中，查看了档案文件并交互地释放了文件。如果只想从档案文件中释放某一个文件，可以在命令行中指定这个文件。作为示例，我先删除了原始的 mydir子目录，然后使用一个空的子目录进行如下操作：

```
# tar xf mydir.tar mydir/mydir2/file23
# tree mydir
mydir
` - - mydir2
   ` - - file23
1 directory , 1 file
```

正如你所看到的，只有一个文件被释放出来。但是千万要小心！虽然 tar命令不会覆盖整个的子目录，但是它会覆盖掉那些有着相同文件名的文件。

在开始建立档案文件之前先多试用一下 tar命令。需要试用的其他功能还有从档案文件中有选择地删除某个文件或者往一个现有的档案文件中再添加一个文件，而这些操作使用下面将要介绍的cpio程序也能完成。

你可以使用其他的程序，如 BRU-2000或者taper脚本程序来备份你的系统或者你选定的文件和子目录。 OpenLinux操作系统也可以通过 cron日程安排来自动进行文件的归档整理工作。请阅读第 23学时教程中关于BRU-2000和taper脚本程序的内容；请阅读第 24学时教程中关于 cron任务计划的内容。

5.3.2 建立cpio档案文件

cpio命令可以从tar或者cpio档案文件中拷入或者拷出文件。因为 cpio命令和tar命令兼容，所以本小节我们将不再讨论它如何工作的细节，但是这个命令具备一些 tar命令没有的功能，如下所示：

- 支持cpio和tar两种档案文件格式
- 支持许多老式磁带数据格式
- 能够通过一个管道(将在6学时教程中学习到)读取文件的文件名

  只有很少(如果有的话)的Linux软件包是以cpio格式发行的。这主要是因为在因特网上查找新软件时根本碰不到任何 cpio档案文件。但是如果对 cpio命令的详细情况感兴趣的话，可以阅读它的使用手册页。在第23学时教程中有一个如何使用 cpio命令对指定文件进行备份的例子。

5.3.3 使用gzip命令压缩文件

gzip命令是用来压缩文件的。这个程序不仅可以用来压缩大的较少使用的文件以节省磁盘空间，还可以和tar命令一起构成可能是Linux操作系统中最流行的压缩文件格式。在因特网上寻找新的Linux软件时，经常会遇到.tgz或.tar.gz格式的文件。

还会发现在子目录 /usr/doc/HOWTO中的许多文件也都是使用 gzip命令压缩的。这可以节省很多的空间。根据 gzip命令的使用手册页中自由软件基金会的人们的统计， gzip命令对文本文件有60%～70%的压缩率。

gzip命令很容易使用。如果想压缩某个文件或者磁带档案文件，请输入下面的内容：

```
# gzip mydir.tar
```

在缺省的状态下，gzip命令会压缩文件、再加上一个 .gz扩展名、然后删除掉原来的文件。如果想解压缩文件，可以使用 gzip命令的对应程序命令 gunzip或者gzip命令的-d解压缩参数。必须保证这个用于解压缩的文件有 .gz(或者.Z、-gz、.z、-z、或者_z)扩展名，否则gzip命令和gunzip命令都会显示出错信息。如果想使用自己的扩展名，可以使用 -S后缀参数，如下所示：

```
# gzip -S .gzipped mydir.tar
```

gzip命令还可以处理用 zip命令、compress命令和pack命令压缩的文件包。如果想在压缩或者解压缩的过程中看到更多的信息，可以使用 -l列清单参数看到文件在被压缩或解压缩的时候的文件长度。在前一个例子中，压缩完子目录 mydir之后，可以使用gzip命令按照下面的方法得到有关的数据：

```
# gzip -l mydir.tar.gz
compressed        uncompr . ratio      uncompressed_name
       239        10240    97.9%     mydir.tar
```

最后，gzip命令还有一个很有用的参数 -t，可以用来测试压缩文件的完整性。如果文件正常，gzip命令不会给出任何显示。如果想看到 OK这两个字母，可以在测试某个文件时使用 -tv参数。

5.3.4 使用compress命令压缩文件

compress命令就象它的文件名那样，就是用来压缩文件的。这是 UNIX世界中出现得比较早的一个压缩程序。

使用compress命令生成的文件传统上都有一个 .Z扩展名。如果想压缩某个文件，请输入下面的内容：

```
# compress file
```

如果想解压缩某个文件，请输入下面的内容：

```
# uncompress file.Z
```

像使用gzip命令的时候一样，必须给出一个带有 .Z扩展名的文件名，否则compress命令显示出错信息。

对其他的 Linux的压缩方法感兴趣吗？请在系统上寻找 zip、unzip、zipcloak、zipnote、zipsplit、zless、zcat、znew、zmore、zcmp、pack、compact、shar、unshar或者zforce程序命令。可能不会发现所有这些程序命令都安装了，但是可以从喜欢的 Linux站点上(比如http://metalab.unc.edu/pub/Linux)找到这些程序。

5.4 课时小结

本学时教程介绍了一些基本的 OpenLinux操作系统中用来对文件或者子目录进行建立、删除、移动、拷贝、保存、以及压缩操作的命令。在工作中学会更多的关于 OpenLinux操作系统的知识的时候，将会在自己的基础知识上建立起更复杂和更适用的命令。

5.5 专家答疑

问：有没有其他的用于OpenLinux操作系统的文件压缩程序？

答：有。使用bzip2命令，可能会得到更好的压缩效果。这个程序使用了与 gzip命令不同的算法，存放在子目录 /bin中供OpenLinux操作系统使用。可以通过它的文件后缀 .bz2识别它的档案文件。

问：曾经见过一些文件带有后缀.uu，不知道它们是不是被压缩的文件？

答：不是。一个带有后缀 .uu的文件是从二进制格式用 uuencode命令转换成平常的文本文件的。如果想把它转换回这个文件原始的格式，需要象下面这样使用 uudecode命令：

```
uudecode somefile.uu
```

这些命令是供有些用户准备把图像、程序或者其他二进制格式的文件通过电子邮件传送或者张贴在某个Usenet新闻组(关于Usenet新闻的详细资料请阅读第12学时教程“配置因特网新闻”)中的时候用来做准备工作的。

问：怎样才能够查看某个二进制文件中的内容？如果使用 cat命令或者less命令，只看到许多的乱码。

答：请试试使用strings命令。如果想了解如何在less中使用strings命令，请阅读第6学时教程“使用shell”。

问：怎样才能从一个磁带档案文件中了解关于每个文件更多的信息？我知道 t参数可以显示出文件或者子目录的名字。

答：在tar命令行上使用 v参数和t参数能够看到一个磁带档案文件中的每个文件的权限、所有者、用户组、文件长度、建立的日期、时间以及名称。

5.6 练习题

1. tar命令可以在进行归档操作的同时进行文件压缩。使用 tar命令对一个实验子目录生成一个压缩的磁带档案文件。

2. OpenLinux中还有几个其他的压缩程序。它们的名字是什么？(提示：可以试试 apropos命令)

3. 怎样才能找到OpenLinux操作系统发行版本中全部的压缩文件？