CentOS 丛书目录 — 系统管理 — 网络服务 — 应用部署

重定向和管道

内容提要

1. 理解标准输入输出
2. 学会使用输入、输出重定向
3. 学会使用管道实现文本过滤

重定向

Linux命令在执行时常常期望接收输入数据，命令执行后又期望将产生的数据结果输出。Linux的大部分命令都具有标准的输入/输出设备端口。下表列出了标准设备。

名称	文件描述符	代表意思	设备	说明
STDIN	0	标准输入	键盘	命令在执行时所要的输入数据通过它来取得
STDOUT	1	标准输出	显示器	命令执行后的输出结果从该端口送出
STDERR	2	标准错误	显示器	命令执行时的错误信息通过该端口送出

所谓重定向，就是不使用系统的标准输入端口、标准输出端口或标准错误端口，而进行重新的指定，所以重定向分为输出重定向、输入重定向和错误重定向。通常情况下重定向到一个文件。在Shell中，要实现重定向主要依靠重定向符实现，即Shell是检查命令行中有无重定向符来决定是否需要实施重定向。下表列出了常用的重定向符。

重定向符	说明
<	实现输入重定向。输入重定向并不经常使用，因为大多数命令都以参数的形式在命令行上指定输入文件的文件名。尽管如此，当使用一个不接受文件名为输入参数的命令，而需要的输入又是在一个已存在的文件里时，就能用输入重定向解决问题
«!……!	实现输入重定向的特例，即here文件
>或»	实现输出重定向。输出重定向比输入重定向更常用。输出重定向使用户能把一个命令 的输出重定向到一个文件里，而不是显示在屏幕上。很多情况下都可以使用这种功能。例如，如果某个命令的输出很多，在屏幕上不能完全显示，即可把它重定向到一个文件中，稍后再用文本编辑器来打开这个文件
2>或2»	实现错误重定向
&>	同时实现输出重定向和错误重定向

下面举几个使用重定向的例子： 1. 将ls命令生成的/tmp目录的一个清单存到当前目录中的dir文件中。

```
$ ls -l /tmp >dir
```

2. 将ls命令生成的/etc目录的一个清单以追加的方式存到当前目录中的dir文件中。

```
$ ls -l /etc >>dir
```

3. 将/etc/passwd文件的内容作为wc命令的输入。

```
$ wc < /etc/passwd
```

4. 将命令随后输入的文本作为wc命令的输入。

```
$  wc <<!
> This text forms the content of the heredocument ,
> which continues until the end of text delimiter
> !
```

5. 获得apache软件包的安装文件清单并存入指定的文件。

```
$  rpm  -ql apache > apache.list
```

6. 用echo命令和输出重定向建立简单的文本文件。

```
$  echo  “Please call me : 68800000”>message
```

7. 利用cat命令、here文档和输出重定向建立简单的文本文件。

```
$  cat <<! >mytext
> This text forms the content of the heredocument ,
> which continues until the end of text delimiter
> !
```

8. 将命令myprogram的错误信息保存在当前目录下的err_file文件中。

```
$  myprogram 2> err_file
```

9. 将命令myprogram的输出信息和错误信息保存在当前目录下的output_file文件中。

```
$  myprogram &> output_file
```

10. 快速建立MP3播放列表。

```
$  find  ~  -name *.mp3 > ~/cd.play.list
```

管道

许多Linux命令具有过滤特性，即一条命令通过标准输入端口接受一个文件中的数据，命令执行后产生的结果数据又通过标准输出端口送给后一条命令，作为该命令的输入数据。后一条命令也是通过标准输入端口而接受输入数据。

Shell提供管道命令“|”将这些命令前后衔接在一起，形成一个管道线，格式为：

```
命令1 | 命令2 | ……|命令n
```

管道线中的每一条命令都作为一个单独的进程运行，每一条命令的输出作为下一条命令的输入。由于管道线中的命令总是从左到右顺序执行的，因此管道线是单向的。

管道线的实现创建了Linux系统管道文件并进行重定向，但是管道不同于I/O重定向，输入重定向导致一个程序的标准输入来自某个文件，输出重定向是将一个程序的标准输出写到一个文件中，而管道是直接将一个程序的标准输出与另一个程序的标准输入相连接，不需要经过任何中间文件。

下面举几个使用管道的例子：

1. 以长格式递归的方式分屏显示/etc目录下的文件和目录列表。

```
$ ls -Rl /etc |more
```

2. 分屏显示文本文件/etc/passwd的内容。

```
$ cat /etc/passwd | more
```

3. 统计文本文件/etc/passwd的行数、字数和字符数。

```
$ cat /etc/passwd | wc
```

4. 查看是否存在lrj用户账号。

```
$ cat /etc/passwd | grep lrj
```

5. 查看引导信息中关于第1块网卡的信息。

```
$ dmesg | grep eth0
```

6. 查看系统是否安装了apache软件包。

```
$ rpm -qa | grep apache
```

7. 解压缩tar名为xyz.tar.gz的软件包。

```
$ gzip -dc xyz.tar.gz | tar -xvf
```

8. 以排序方式查看Linux系统中目录的磁盘占据情况。

```
$ du -S | sort -n
```

9. 快速移动整个目录。

```
$ (cd /source/directory && tar cf - . ) | (cd /dest/directory && tar xvfp -)
```

10. 把 man的信息存为文本文件。

```
$  man bash | col -b > bash.txt
```

11. 只列子目录。

```
$  ls -F | grep /$
```

或者

```
$  ls -l | grep "^d"
```

12. 计算当前目录下的文件数和目录数。

```
$  ls -l * | grep "^-" | wc -l
$  ls -l * | grep "^d" | wc -l
```

13.显示文本文件中间的若干行

```
$  tail +15 myalllist | head -3
```

- 显示源文件
- 登录