第2章　使用find和xargs

有时可能需要在系统中查找具有某一特征的文件 (例如文件权限、文件属主、文件长度、文件类型等等)。这样做可能有很多原因。可能出于安全性的考虑，或是一般性的系统管理任务，或许只是为了找出一个不知保存在什么地方的文件。 Find是一个非常有效的工具，它可以遍历当前目录甚至于整个文件系统来查找某些文件或目录。

在本章中，我们介绍以下内容：

• find命令选项。

• 使用find命令不同选项的例子。

• 配合find使用xargs命令的例子。

由于find具有如此强大的功能，所以它的选项也很多，其中大部分选项都值得我们花时间来了解一下。即使系统中含有网络文件系统 (NFS)，find命令在该文件系统中同样有效，只要你具有相应的权限。

在运行一个非常消耗资源的 find命令时，很多人都倾向于把它放在后台执行，因为遍历一个大的文件系统可能会花费很长的时间 (这里是指30G字节以上的文件系统)。

Find命令的一般形式为：

```
find pathname -options [-print -exec -ok]
```

让我们来看看该命令的参数：

pathname　find命令所查找的目录路径。例如用 .来表示当前目录，用 /来表示系统根目录。

-print　find命令将匹配的文件输出到标准输出。

-exec　find命令对匹配的文件执行该参数所给出的 shell命令。相应命令的形式为 'command' {} \;，注意{}和\；之间的空格。

-ok　和-exec的作用相同，只不过以一种更为安全的模式来执行该参数所给出的 shell命令，在执行每一个命令之前，都会给出提示，让用户来确定是否执行。

2.1　find命令选项

find命令有很多选项或表达式，每一个选项前面跟随一个横杠 -。让我们先来看一下该命令的主要选项，然后再给出一些例子。

-name　按照文件名查找文件。

-perm　按照文件权限来查找文件。

-prune　使用这一选项可以使find命令不在当前指定的目录中查找，如果同时使用了 -depth选项，那么-prune选项将被find命令忽略。

-user　按照文件属主来查找文件。

-group　按照文件所属的组来查找文件。

-mtime -n +n　按照文件的更改时间来查找文件， -n表示文件更改时间距现在n天以内， +n表示文件更改时间距现在n天以前。Find命令还有-atime和-ctime选项，但它们都和-mtime选项

相似，所以我们在这里只介绍-mtime选项。

-nogroup 查找无有效所属组的文件，即该文件所属的组在/etc/groups中不存在。

-nouser 查找无有效属主的文件，即该文件的属主在/etc/passwd中不存在。

-newer file1！file2 查找更改时间比文件file1新但比文件file2旧的文件。

-type 查找某一类型的文件，诸如：

b - 块设备文件。

d - 目录。

c - 字符设备文件。

p - 管道文件。

l - 符号链接文件。

f - 普通文件。

-size n[c] 查找文件长度为n块的文件，带有c时表示文件长度以字节计。

-depth 在查找文件时，首先查找当前目录中的文件，然后再在其子目录中查找。

-fstype 查找位于某一类型文件系统中的文件，这些文件系统类型通常可以在配置文件/etc/fstab中找到，该配置文件中包含了本系统中有关文件系统的信息。

-mount 在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，就跟踪至链接所指向的文件。

-cpio 对匹配的文件使用cpio命令，将这些文件备份到磁带设备中。

2.1.1 使用name选项

文件名选项是find命令最常用的选项，要么单独使用该选项，要么和其他选项一起使用。可以使用某种文件名模式来匹配文件，记住要用引号将文件名模式引起来。

不管当前路径是什么，如果想要在自己的根目录 $HOME中查找文件名符合*.txt的文件，使用~作为'pathname参数，波浪号~代表了你的$HOME目录。

```
$ find ~ -name "*.txt" -print
```

想要在当前目录及子目录中查找所有的‘ *.txt ’文件，可以用：

```
$ find . -name "*.txt" -print
```

想要的当前目录及子目录中查找文件名以一个大写字母开头的文件，可以用：

```
$ find . -name "[A-Z]*" -print
```

想要在/etc目录中查找文件名以host开头的文件，可以用：

```
$ find /etc -name "host*" -print
```

想要查找$HOME目录中的文件，可以用：

```
$ find ~ -name "*" -print或find . -print
```

要想让系统高负荷运行，就从根目录开始查找所有的文件。如果希望在系统管理员那里保留一个好印象的话，最好在这么做之前考虑清楚！

```
$ find / -name "*" -print
```

如果想在当前目录查找文件名以两个小写字母开头，跟着是两个数字，最后是 *.txt的文件，下面的命令就能够返回名为ax37.txt的文件：

```
$ find . -name "[a-z][a-z][0--9][0--9].txt" -print
```

2.1.2 使用perm选项

如果希望按照文件权限模式来查找文件的话，可以采用 -perm选项。你可能需要找到所有用户都具有执行权限的文件，或是希望查看某个用户目录下的文件权限类型。在使用这一选项的时候，最好使用八进制的权限表示法。

为了在当前目录下查找文件权限位为 755的文件，即文件属主可以读、写、执行，其他用户可以读、执行的文件，可以用：

```
$ find . -perm 755 -print
```

如果希望在当前目录下查找所有用户都可读、写、执行的文件（要小心这种情况），我们可以使用find命令的-perm选项。在八进制数字前面要加一个横杠 -。在下面的命令中 -perm代表按照文件权限查找，而‘ 007 ’和你在chmod命令的绝对模式中所采用的表示法完全相同。

```
$ find . -perm -007 -print
```

2.1.3 忽略某个目录

如果在查找文件时希望忽略某个目录，因为你知道那个目录中没有你所要查找的文件，那么可以使用-prune选项来指出需要忽略的目录。在使用 -prune选项时要当心，因为如果你同时使用了-depth选项，那么-prune选项就会被find命令忽略。

如果希望在/apps目录下查找文件，但不希望在 /apps/bin目录下查找，可以用：

```
$ find /apps -name "/apps/bin" -prune -o -print
```

2.1.4 使用user和nouser选项

如果希望按照文件属主查找文件，可以给出相应的用户名。例如，在 $HOME目录中查找文件属主为dave的文件，可以用：

```
$ find ~ -user dave -print
```

在/etc目录下查找文件属主为uucp的文件：

```
$ find /etc -user uucp -print
```

为了查找属主帐户已经被删除的文件，可以使用 -nouser选项。这样就能够找到那些属主在/etc/passwd文件中没有有效帐户的文件。在使用 -nouser选项时，不必给出用户名；find命令能够为你完成相应的工作。例如，希望在 /home目录下查找所有的这类文件，可以用：

```
$ find /home -nouser -print
```

2.1.5 使用group和nogroup选项

就像user和nouser选项一样，针对文件所属于的用户组， find命令也具有同样的选项，为了在/apps目录下查找属于accts用户组的文件，可以用：

```
$ find /apps -group accts -print
```

要查找没有有效所属用户组的所有文件，可以使用 nogroup选项。下面的find命令从文件系统的根目录处查找这样的文件

```
$ fine/-nogroup-print
```

2.1.6 按照更改时间查找文件

如果希望按照更改时间来查找文件，可以使用mtime选项。如果系统突然没有可用空间了，很有可能某一个文件的长度在此期间增长迅速，这时就可以用 mtime选项来查找这样的文件。用减号 - 来限定更改时间在距今 n日以内的文件，而用加号 +来限定更改时间在距今 n日以前的文件。

希望在系统根目录下查找更改时间在5日以内的文件，可以用：

```
$ find / -mtime -5 -print
```

为了在/var/adm目录下查找更改时间在3日以前的文件，可以用：

```
$ find /var/adm -mtime +3 -print
```

2.1.7 查找比某个文件新或旧的文件

如果希望查找更改时间比某个文件新但比另一个文件旧的所有文件，可以使用 -newer选项。它的一般形式为：

```
newest_file_name ! oldest_file_name
```

其中，！是逻辑非符号。

这里有两个文件，它们的更改时间大约相差两天。

```
-rwxr-xr-x      1 root      root        92 Apr 18 11:18  age.awk
-rwxrwxr-x      1 root      root      1054 Apr 20 19:37  belts.awk
```

下面给出的find命令能够查找更改时间比文件age.awk新但比文件belts.awk旧的文件：

```
$ find . -newer age.awk ! -newer belts.awk -exec ls -l {} \;
-rwxrwxr-x   1 root  root    62 Apr 18 11:32 ./who.awk
-rwxr-xr-x   1 root  root    49 Apr 18 12:05 ./group.awk
-rw-r--r--   1 root  root   201 Apr 20 19:30 ./grade2.txt
-rwxrwxr-x   1 root  root  1054 Apr 20 19:37 ./belts.awk
```

如果想使用find命令的这一选项来查找更改时间在两个小时以内的文件，除非有一个现成的文件其更改时间恰好在两个小时以前，否则就没有可用来比较更改时间的文件。为了解决这一问题，可以首先创建一个文件并将其日期和时间戳设置为所需要的时间。这可以用 touch命令来实现。

假设现在的时间是23:40，希望查找更改时间在两个小时以内的文件，可以首先创建这样一个文件：

```
$ touch -t 05042140 dstamp
$ ls -l dstamp
-rw-r--r--  1 dave     admin         0 May  4 21:40 dstamp
```

一个符合要求的文件已经被创建；这里我们假设今天是五月四日，而该文件的更改时间是21:40，比现在刚好早两个小时。

现在我们就可以使用find命令的-newer选项在当前目录下查找所有更改时间在两个小时以内的文件：

```
$ find . -newer dstamp -print
```

2.1.8 使用type选项

UNIX或LINUX系统中有若干种不同的文件类型，这部分内容我们在前面的章节已经做了

介绍，这里就不再赘述。如果要在/etc目录下查找所有的目录，可以用：

```
$ find /etc -type d -print
```

为了在当前目录下查找除目录以外的所有类型的文件，可以用：

```
$ find . ! -type d -print
```

为了在/etc目录下查找所有的符号链接文件，可以用：

```
$ find /etc -type l -print
```

2.1.9 使用size选项

可以按照文件长度来查找文件，这里所指的文件长度既可以用块（ block）来计量，也可以用字节来计量。以字节计量文件长度的表达形式为 Nc；以块计量文件长度只用数字表示即可。

就我个人而言，我总是使用以字节计的方式，在按照文件长度查找文件时，大多数人都喜欢使用这种以字节表示的文件长度，而不用块的数目来表示，除非是在查看文件系统的大小，因为这时使用块来计量更容易转换。

为了在当前目录下查找文件长度大于1M字节的文件，可以用：

```
$ find . -size +1000000c -print
```

为了在/home/apache目录下查找文件长度恰好为100字节的文件，可以用：

```
$ find /home/apache -size 100c -print
```

为了在当前目录下查找长度超过10块的文件（一块等于512字节），可以用：

```
$ find . -size +10 -print
```

2.1.10 使用depth选项

在使用find命令时，可能希望先匹配所有的文件，再在子目录中查找。使用 depth选项就可以使find命令这样做。这样做的一个原因就是，当在使用 find命令向磁带上备份文件系统时，希望首先备份所有的文件，其次再备份子目录中的文件。

在下面的例子中， find命令从文件系统的根目录开始，查找一个名为 CON.FILE的文件。它将首先匹配所有的文件然后再进入子目录中查找。

```
$ find / -name "CON.FILE" -depth -print
```

2.1.11 使用mount选项

在当前的文件系统中查找文件（不进入其他文件系统），可以使用find命令的mount选项。在下面的例子中，我们从当前目录开始查找位于本文件系统中文件名以 XC结尾的文件：

```
$ find . -name "*.XC" -mount -print
```

2.1.12 使用cpio选项

cpio命令可以用来向磁带设备备份文件或从中恢复文件。可以使用 find命令在整个文件系统中（更多的情况下是在部分文件系统中）查找文件，然后用 cpio命令将其备份到磁带上。

如果希望使用cpio命令备份/etc、/home和/apps目录中的文件，可以使用下面所给出的命令，不过要记住你是在文件系统的根目录下：

```
$ cd /
$ find etc home apps -depth -print | cpio -ivcdC65536 -o  \
 /dev/rmt0
```

(在上面的例子中，第一行末尾的\告诉shell命令还未结束，忽略\后面的回车。)

在上面的例子中，应当注意到路径中缺少 /。这叫作相对路径。之所以使用相对路径，是因为在从磁带中恢复这些文件的时候，可以选择恢复文件的路径。例如，可以将这些文件先恢复到另外一个目录中，对它们进行某些操作后，再恢复到原始目录中。如果在备份时使用了绝对路径，例如/etc，那么在恢复时，就只能恢复到/etc目录中去，别无其他选择。在上面的例子中，我告诉find命令首先进入/etc目录，然后是/home和/apps目录，先匹配这些目录下的文件，然后再匹配其子目录中的文件，所有这些结果将通过管道传递给 cpio命令进行备份。

顺便说一下，在上面的例子中cpio命令使用了C65536选项，我本可以使用B选项，不过这样每块的大小只有 512字节，而使用了 C65536选项后，块的大小变成了 64K字节(65536/1024)。

2.1.13 使用exec或ok来执行shell命令

当匹配到一些文件以后，可能希望对其进行某些操作，这时就可以使用 -exec选项。一旦find命令匹配到了相应的文件，就可以用 -exec选项中的命令对其进行操作（在有些操作系统中只允许-exec选项执行诸如ls或ls -l这样的命令)。大多数用户使用这一选项是为了查找旧文件并删除它们。这里我强烈地建议你在真正执行 rm命令删除文件之前，最好先用 ls命令看一下，确认它们是所要删除的文件。

exec选项后面跟随着所要执行的命令，然后是一对儿 {}，一个空格和一个\，最后是一个分号。

为了使用exec选项，必须要同时使用print选项。如果验证一下find命令，会发现该命令只输出从当前路径起的相对路径及文件名。

为了用ls -l命令列出所匹配到的文件，可以把ls -l命令放在find命令的-exec选项中，例如：

```
$ find . -type f -exec ls -l {} \;
-rwxr-xr-x  10 root  wheel        1222 Jan  4  1993./sbin/C80
-rwxr-xr-x  10 root  wheel        1222 Jan  4  1993./sbin/Normal
-rwxr-xr-x  10 root  wheel        1222 Jan  4  1993./sbin/Revvid
```

上面的例子中，find命令匹配到了当前目录下的所有普通文件，并在 -exec选项中使用ls -l命令将它们列出。

为了在/logs目录中查找更改时间在5日以前的文件并删除它们，可以用：

```
$ find logs -type f -mtime +5 -exec rm {} \;
```

记住，在shell中用任何方式删除文件之前，应当先查看相应的文件，一定要小心！

当使用诸如mv或rm命令时，可以使用-exec选项的安全模式。它将在对每个匹配到的文件进行操作之前提示你。在下面的例子中， find命令在当前目录中查找所有文件名以 .LOG结尾、更改时间在5日以上的文件，并删除它们，只不过在删除之前先给出提示。

```
$ find . -name "*.LOG" -mtime +5 -ok rm {} \;
< rm ... ./nets.LOG > ? y
```

按y键删除文件，按n键不删除。

任何形式的命令都可以在 -exec选项中使用。在下面的例子中我们使用 grep命令。 find命令

首先匹配所有文件名为“ passwd* ”的文件，例如passwd、passwd.old、passwd.bak，然后执行grep命令看看在这些文件中是否存在一个 rounder用户。

```
$ find /etc -name "passwd*" -exec grep "rounder" {} \;
rounder:JL9TtUqk8EHwc:500:500::/home/apps/nets/rounder:/bin/sh
```

2.1.14 find命令的例子 find * -name a.txt -- 查找所有目录下包含a.txt文件

我们已经介绍了find命令的基本选项，下面给出find命令的一些其他的例子。

为了匹配$HOME目录下的所有文件，下面两种方法都可以使用：

```
$ find $HOME -print
$ find ~ -print
```

为了在当前目录中查找suid置位，文件属主具有读、写、执行权限，并且文件所属组的用户和其他用户具有读和执行的权限的文件，可以用：

```
$ find . -type f -perm 4755 -print
```

为了查找系统中所有文件长度为0的普通文件，并列出它们的完整路径，可以用：

```
$ find / -type f -size 0 -exec ls -l {} \;
```

为了查找/var/logs目录中更改时间在7日以前的普通文件，并删除它们，可以用：

```
$ find /var/logs -type f -mtime +7 -exec rm {} \;
```

为了查找系统中所有属于audit组的文件，可以用：

```
$find /-name -group audit -print
```

我们的一个审计系统每天创建一个审计日志文件。日志文件名的最后含有数字，这样我们一眼就可以看出哪个文件是最新的，哪个是最旧的。 Admin.log文件编上了序号：admin.log.001、admin.log.002等等。下面的find命令将删除/logs目录中访问时间在7日以前、含有数字后缀的admin.log文件。该命令只检查三位数字，所以相应日志文件的后缀不要超过999。

```
$ find /logs -name 'admin.log[0-9][0-9][0-9]' -atime +7 -exec rm {} \;
```

为了查找当前文件系统中的所有目录并排序，可以用：

```
$ find . -type d -print -local -mount |sort
```

为了查找系统中所有的rmt磁带设备，可以用：

```
$ find /dev/rmt -print
```

2.2 xargs - 由于exec命令有长度的限制，所以xargs出现了

在使用find命令的-exec选项处理匹配到的文件时， find命令将所有匹配到的文件一起传递给exec执行。不幸的是，有些系统对能够传递给 exec的命令长度有限制，这样在 find命令运行几分钟之后，就会出现溢出错误。错误信息通常是“ 参数列太长 ”或“ 参数列溢出 ”。这就是xargs命令的用处所在，特别是与 find命令一起使用。 Find命令把匹配到的文件传递给xargs命令，而xargs命令每次只获取一部分文件而不是全部，不像 -exec选项那样。这样它可以先处理最先获取的一部分文件，然后是下一批，并如此继续下去。在有些系统中，使用 -exec选项会为处理每一个匹配到的文件而发起一个相应的进程，并非将匹配到的文件全部作为参数一次执行；这样在有些情况下就会出现进程过多，系统性能下降的问题，因而效率不高；而使用

xargs命令则只有一个进程。另外，在使用 xargs命令时，究竟是一次获取所有的参数，还是分批取得参数，以及每一次获取参数的数目都会根据该命令的选项及系统内核中相应的可调参数来确定。

让我们来看看xargs命令是如何同find命令一起使用的，并给出一些例子。

下面的例子查找系统中的每一个普通文件，然后使用 xargs命令来测试它们分别属于哪类文件：

```
$ find / -type f  -print | xargs file
/etc/protocols: English text
/etc/securetty: ASCII text
..
```

下面的例子在整个系统中查找内存信息转储文件 (core dump)，然后把结果保存到/tmp/core.log 文件中：

```
$ find . -name "core" -print | xargs echo "" >/tmp/core.log
```

下面的例子在/apps/audit目录下查找所有用户具有读、写和执行权限的文件，并收回相应的写权限：

```
$ find /apps/audit -perm -7 -print | xargs chmod o-w
```

在下面的例子中，我们用 grep命令在所有的普通文件中搜索 device这个词：

```
$ find / -type f -print | xargs grep "device"
```

在下面的例子中，我们用 grep命令在当前目录下的所有普通文件中搜索 DBO这个词：

```
$ find . -name \* -type f -print | xargs grep "DBO"
```

注意，在上面的例子中，\用来取消find命令中的*在shell中的特殊含义。

2.3 小结

find命令是一个非常优秀的工具，它可以按照用户指定的准则来匹配文件。使用 exec和xargs可以使用户对所匹配到的文件执行几乎所有的命令。

形式上的区别：

```
find ..... | xargs grep "xxx"

find ..... - exec ls -l {} \ ;
```