

1 文件查找和打包压缩

1.1 文件查找

1.1.1 locate命令

1.1.2 find命令

1.1.3 权限查找

1.1.4 类型查找

1.1.5 组合查找

1.1.6 组合进阶

1.1.7 文件属性查找

1.1.8 动作进阶

1.1.9 xargs组合

1.2 打包压缩

1.2.1 gz包

1.2.2 tar包

1.2.3 zip包

1.2.4 z包

1.2.5 bz包

1.2.6 xz包

1.2.7 其他信息

1.2.9 split命令

1 文件查找和打包压缩

1.1 文件查找

1.1.1 locate命令

命令解读

locate简介

locate命令是Linux系统中用于快速查找文件和目录的一个非常实用的命令行工具。它主要通过搜索事先构建的文件数据库来定位文件，而不是像**find**命令那样实时遍历整个文件系统，因此其搜索速度非常快。

工作特点

- 非实时查找
索引的构建是在系统较为空闲时自动进行(周期任务)，执行**updatedb**可以更新数据库
- 查找速度快
locate 查询系统上预建的文件索引数据库 `/var/lib/mlocate/mlocate.db`
- 模糊查找
- 搜索的是文件全路径，不仅仅是文件名
索引构建过程需要遍历整个根文件系统，很消耗资源
- 可能只搜索用户具备读取和执行权限的目录

locate命令

命令格式:

```
locate [OPTIONS] PATTERN
```

一般选项

-i: 忽略大小写地匹配模式。例如，`locate -i myfile`会同时匹配"myfile"、"MyFile"等。

-c: 只显示匹配项的数量（count），而不是文件名。

-l NUM: 限制输出的行数，每行显示一个匹配项。

-n NUM: 限制输出的匹配项数目，只显示前NUM个结果。

-r: 使用正则表达式模式匹配。例如，`locate --regex '.*\.txt$'`会匹配所有以".txt"结尾的文件。

-b: 只匹配基准名，忽略路径。

-w: 仅匹配完整单词。

| | |
|-----------------------------|-------------------------------|
| -A --all | #输出所有能匹配到的文件名，不管文件是否存在 |
| -b --basename | #仅匹配文件名部份，而不匹配路径中的内容 |
| -c --count | #只输出找到的数量 |
| -d --database DBPATH | #指定数据库 |
| -e --existing | #仅打印当前现有文件的条目 |
| -L --follow | #遇到软链接时则跟随软链接去其对应的目标文件中查找（默认） |
| -i --ignore-case | #忽略大小写 |
| -l --limit -n N | #只显示前N条匹配数据 |
| -P --nofollow, -H | #不跟随软链 |
| -r --regex REGEXP | #使用基本正则表达式 |
| --regex | #使用扩展正则表达式 |
| -s --stdio | #忽略向后兼容 |
| -w --wholename | #全路径匹配，就是只要在路径里面出现关键字（默认） |

注意事项

数据库更新:

`locate`命令的搜索结果依赖于其背后的数据库。因此，在首次使用`locate`命令之前，或者当文件系统发生较大变化时（如新增或删除了大量文件），建议运行`sudo updatedb`命令来更新数据库，以确保搜索结果的准确性。

性能优势:

由于`locate`命令基于数据库搜索，因此其搜索速度远快于`find`命令等实时遍历文件系统的工具。这使得`locate`命令在处理大规模文件系统时尤为高效。

局限性:

由于`locate`命令依赖于数据库，因此它可能无法立即反映出文件系统的最新变化（如最近添加或删除的文件）。此外，`locate`命令的搜索结果还可能受到数据库更新频率和系统配置的影响。

简单实践

安装环境

Rocky系统

```
yum install -y mlocate
```

ubuntu系统

```
apt install -y Plocate
```

locatedb创建数据库

Rocky系统

检查配置

```
[root@rocky9 ~]# locate conf
```

```
locate: can not stat () `/var/lib/mlocate/mlocate.db': No such file or directory
```

更新数据库

```
[root@rocky9 ~]# updatedb
```

检查效果

```
[root@rocky9 ~]# ll /var/lib/mlocate/mlocate.db
-rw-r----- 1 root slocate 3143608 Jun 18 08:58 /var/lib/mlocate/mlocate.db
[root@rocky9 ~]# locate -n 3 conf
/boot/config-4.18.0-348.el8.0.2.x86_64
/boot/grub2/i386-pc/configfile.mod
/boot/loader/entries/c0298860b41f4cd296da0d2853451604-0-rescue.conf
```

Ubuntu系统:

检查配置文件

```
root@ubuntu24:~# locate conf
/var/lib/plocate/plocate.db: No such file or directory
```

更新数据库

```
root@ubuntu24:~# updatedb
```

确认效果

```
root@ubuntu24:~# ll /var/lib/plocate/plocate.db
-rw-r----- 1 root plocate 3142215 May 10 12:17 /var/lib/plocate/plocate.db
root@ubuntu24:~# locate -n 3 conf
/boot/config-5.15.0-25-generic
/boot/config-5.15.0-71-generic
/boot/grub/i386-pc/configfile.mod
```

文件新创建和删除，无法马上更新locate数据库

新文件创建后无法查找

```
root@ubuntu24:~# touch test.log
root@ubuntu24:~# locate test.log
```

更新数据库之后再查找

```
root@ubuntu24:~# updatedb
root@ubuntu24:~# locate test.log
/root/test.log
```

文件被删除，还能查到

```
root@ubuntu24:~# rm -f test.log
root@ubuntu24:~# locate test.log
/root/test.log
```

但实际上文件已经不存在了

```
root@ubuntu24:~# stat /root/test.log
stat: cannot statx '/root/test.log': No such file or directory
```

再次更新数据库

```
root@ubuntu24:~# updatedb
root@ubuntu24:~# locate test.log
```

参数实践

搜索名称或路径中包含“conf”的文件

```
root@ubuntu24:~# locate conf
```

搜索ect目录中以a开头的文件或目录，路径包含写法

```
root@ubuntu24:~# locate /etc/a
```

仅搜索文件名中包含share 的内容

```
root@ubuntu24:~# locate -b share
```

显示数量

```
root@ubuntu24:~# locate -c conf
```

显示前10条

```
root@ubuntu24:~# locate -n 10 conf
```

使用基本正则表达式

```
root@ubuntu24:~# locate -r '\.conf$'
```

指定数据库

```
root@ubuntu24:~# locate -d /tmp/nofile conf
```

```
locate: can not stat () `/tmp/nofile': No such file or directory
```

安静模式，不输出错误信息，新版本中没有 -q 选项

```
root@ubuntu24:~# locate -qd /tmp/nofile conf
```

1.1.2 find命令

基础知识

find简介

find命令是Unix/Linux系统中一个功能强大且灵活的命令行工具，用于在目录结构中搜索文件和目录。它不仅能够根据文件名、类型、大小、修改时间等属性进行搜索，还可以对搜索结果执行各种操作，如删除、移动、复制等。这使得**find命令**成为系统管理员和开发人员日常工作中不可或缺的工具。

工作特点

- 查找速度略慢
- 精确查找
- 实时查找
- 查找条件丰富
- 只搜索用户具备读取和执行权限的目录

命令简介

命令格式：

```
find [搜索路径] [选项] [表达式]
```

搜索路径：指定**find**命令开始搜索的目录。可以是一个或多个路径。不指定路径，表示当前路径。

选项：用于定义搜索的条件，如文件名、文件类型、大小、时间等。

表达式：用于进一步细化搜索条件，可以与选项组合使用来构建复杂的搜索逻辑。

常用选项

- name: 按文件名匹配，区分大小写。支持使用glob，如：*, ?, [], [^], 通配符要加双引号引起来
- type: 按文件类型搜索。常用类型包括f（普通文件）、d（目录）、l（符号链接）等。
- size: 按文件大小搜索。单位可以是c（字节）、k（千字节）、M（兆字节）、G（千兆字节）等。例如，-size +10M查找大于10MB的文件。
- mtime: 按天数查找文件最后修改时间。例如，-mtime -7查找最近7天内修改的文件。

-user: 按文件所有者查找。
-group: 按文件所属组查找。
-perm: 按文件权限查找。
-maxdepth: 限制搜索的最大深度。
-mindepth: 限制搜索的最小深度。
-regextype type: 正则表达式类型, `emacs` | `posix-awk` | `posix-basic` | `posix-egrep` | `posix-extended`
-regex pattern: 正则表达式

一般选项

| | |
|-------------------------|---------------------------|
| -iname name | #不区分字母大小写 |
| -inum number | #按inode号查找 |
| -samefile name | #相同inode号的文件 |
| -links n | #链接数为n的文件 |
| -regex "PATTERN" | #以PATTERN匹配整个文件路径, 而非文件名称 |

表达式组合

`find`命令支持使用逻辑操作符(如**-and**、**-or**、**-not**)来组合多个搜索条件, 以构建复杂的搜索逻辑。此外, 还可以使用括号来分组条件, 但括号需要被转义或使用引号括起来。

搜索结果的处理

除了搜索功能外, `find`命令还可以对搜索结果执行各种操作。常用的操作选项包括:

-print: 打印搜索结果(默认行为)。
-exec: 对每个匹配的文件执行指定的命令。命令以`{}`代表当前文件, 命令结尾以`;`或`+`结束。
-ok: 与**-exec**类似, 但在执行每个命令前都会提示用户确认。
-delete: 直接删除匹配的文件或目录(使用时要小心)。
-ls: 以`ls -dils`的格式显示匹配的文件。

简单实践

默认列出当前目录下的所有文件

准备工作

```
mkdir find; cd find/  
mkdir dir1/dir2/dir3 -p  
touch dir1/dir2/dir3/f{x,y}  
touch dir1/dir2/f{a,b}  
touch dir1/f{1,2}  
cp /etc/fstab ./  
cp /etc/issue .issue
```

命令演示

```
root@ubuntu24:~# find  
.  
./dir1  
./dir1/dir2  
./dir1/dir2/dir3  
...
```

指定搜索目录层级

最大搜索深度

```
root@ubuntu24:~# find /etc/ -maxdepth 2
```

最小搜索深度

```
root@ubuntu24:~# find /etc/ -mindepth 2
```

仅搜索第二层目录

```
root@ubuntu24:~# find /etc/ -maxdepth 2 -mindepth 2
```

先处理文件再处理目录

显示目录结构

```
root@ubuntu24:~# tree -a
```

```
.
├── dir1
│   ├── dir2
│   │   ├── dir3
│   │   │   ├── fx
│   │   │   └── fy
│   │   ├── fa
│   │   └── fb
│   ├── f1
│   └── f2
├── fstab
└── .issue
```

3 directories, 8 files

默认先显示目录

```
root@ubuntu24:~# find
```

```
.
./dir1
./dir1/dir2
./dir1/dir2/dir3
./dir1/dir2/dir3/fx
./dir1/dir2/dir3/fy
./dir1/dir2/fa
./dir1/dir2/fb
./dir1/f1
./dir1/f2
./fstab
./issue
```

先显示文件

```
root@ubuntu24:~# find -depth
```

```
./dir1/dir2/dir3/fx
./dir1/dir2/dir3/fy
./dir1/dir2/dir3
./dir1/dir2/fa
./dir1/dir2/fb
./dir1/dir2
./dir1/f1
./dir1/f2
./dir1
```

```
./fstab
./issue
.
```

根据文件名和inode查找

查看当前目录下的文件

```
root@ubuntu24:~# touch test-{a,b,A,B}.{log,txt}
root@ubuntu24:~# ls
dir1  fstab  test-a.log  test-A.log  test-a.txt  test-A.txt  test-b.log  test-
B.log  test-b.txt  test-B.txt
```

指定文件名查找

```
root@ubuntu24:~# find -name test-a.log
./test-a.log
```

指定文件名，忽略大小写

```
root@ubuntu24:~# find -iname test-a.log
./test-a.log
./test-A.log
```

通配符

```
root@ubuntu24:~# find -name "*txt"
./test-a.txt
...
./test-B.txt
```

#通配符

```
root@ubuntu24:~# find -name "test-a*"
./test-a.log
./test-a.txt
```

#正则表达式

```
root@ubuntu24:~# find -regex ".*\.log$"
./test-a.log
./test-b.log
./test-A.log
./test-B.log
```

#正则表达式

```
root@ubuntu24:~# find -regex ".*test-[a-z].*"
./test-a.log
./test-a.txt
./test-b.log
./test-b.txt
```

#正则表达式，路径匹配

```
root@ubuntu24:~# find -regex ".*dir3.*"
./dir1/dir2/dir3
./dir1/dir2/dir3/fx
./dir1/dir2/dir3/fy
```

#正则表达式，路径匹配

```
root@ubuntu24:~# find -regex ".*dir3$"
./dir1/dir2/dir3
```

1.1.3 权限查找

基础知识

用户属性解读

| | |
|-----------------------------|-----------------------------|
| <code>-user USERNAME</code> | #查找属主为指定用户(UID)的文件 |
| <code>-group GRPNAME</code> | #查找属组为指定组(GID)的文件 |
| <code>-uid UserID</code> | #查找属主为指定的 UID 号的文件 |
| <code>-gid GroupID</code> | #查找属组为指定的 GID 号的文件 |
| <code>-nouser</code> | #查找没有属主的文件 |
| <code>-nogroup</code> | #查找没有属组的文件 |

权限属性解读

| | |
|------------------------------|--|
| <code>-perm [/ -]MODE</code> | |
| MODE | #精确权限匹配 |
| /MODE | #任何一类(u,g,o)对象的权限中只要有一位匹配即可，表示或者(or)关系 |
| +MODE | #从CentOS 7开始已淘汰 |
| -MODE | #每一类对象都必须同时拥有指定权限，表示与(and)关系 |

注意：

0 如果要找时权限位上的值为0，则表示不关注该角色权限

简单实践

准备工作

执行命令

```
touch dir1/f{a..d}.txt
chown sswang:root dir1/f{a,b}.txt
chown root:sswang dir1/f{c,d}.txt
chown 123:456 dir1/f1
chown 789:root dir1/f2
```

查看效果

```
root@ubuntu24:~# ll dir1/
```

总计 12

```
drwxr-xr-x 3 root  root  4096 10月 29 21:49 ./
drwxr-xr-x 3 root  root  4096 10月 29 21:48 ../
drwxr-xr-x 3 root  root  4096 10月 29 21:38 dir2/
-rw-r--r-- 1 123   456    0 10月 29 21:38 f1
-rw-r--r-- 1 789  root    0 10月 29 21:38 f2
-rw-r--r-- 1 sswang root    0 10月 29 21:49 fa.txt
-rw-r--r-- 1 sswang root    0 10月 29 21:49 fb.txt
-rw-r--r-- 1 root  sswang   0 10月 29 21:49 fc.txt
-rw-r--r-- 1 root  sswang   0 10月 29 21:49 fd.txt
```

基于所有者查找

指定属主

```
root@ubuntu24:~# find -user sswang
./dir1/fa.txt
./dir1/fb.txt
```

指定属主可以用 UID

```
root@ubuntu24:~# find -user 1010
./fa.txt
./fb.txt
```

指定属主属组

```
root@ubuntu24:~# find -user sswang -group root
./dir1/fa.txt
./dir1/fb.txt
```

指定属主ID

```
root@ubuntu24:~# find -uid 1010
./dir1/fa.txt
./dir1/fb.txt
```

基于归属组查找

指定属组

```
root@ubuntu24:~# find -group sswang
./dir1/fc.txt
./dir1/fd.txt
```

指定属组可以用GID

```
root@ubuntu24:~# find -group 1000
./fc.txt
./fd.txt
```

指定属组ID

```
root@ubuntu24:~# find -gid 456
./dir1/f1
```

指定属主属组ID

```
root@ubuntu24:~# find -uid 1010 -gid 0
./dir1/fa.txt
./dir1/fb.txt
```

非属性查找-拓展

属主用户不存在

```
root@ubuntu24:~# find -nouser
./dir1/f1
./dir1/f2
```

属组不存在

```
root@ubuntu24:~# find -nogroup
./dir1/f1
```

属主属组不存在

```
root@ubuntu24:~# find -nouser -nogroup
./dir1/f1
```

文件权限查找

命令

```
touch file/f-{1..8}.txt
chmod 000 file/f-1.txt
chmod 444 file/f-2.txt
chmod 222 file/f-3.txt
chmod 111 file/f-4.txt
chmod 644 file/f-5.txt
chmod 777 file/f-6.txt
chmod 421 file/f-8.txt
```

查看效果

```
root@ubuntu24:~# ll file/
```

总计 8

```
----- 1 root root    0 10月 29 21:53 f-1.txt
-r--r--r-- 1 root root    0 10月 29 21:53 f-2.txt
--w--w--w- 1 root root    0 10月 29 21:53 f-3.txt
---x--x--x 1 root root    0 10月 29 21:53 f-4.txt*
-rw-r--r-- 1 root root    0 10月 29 21:53 f-5.txt
-rwxrwxrwx 1 root root    0 10月 29 21:53 f-6.txt*
-rw-r--r-- 1 root root    0 10月 29 21:53 f-7.txt
-r---w---x 1 root root    0 10月 29 21:53 f-8.txt*
```

精确匹配，ugo都只能是 r权限

```
root@ubuntu24:~# find -perm 444 -ls
```

```
1835416      0 -r--r--r--   1 root    root          0 Jul 25 09:12 ./f-
2.txt
```

或关系，ugo只要有一个角色有r权限即可

```
root@ubuntu24:~# find -perm /444 -ls
```

```
1835414      0 drwxr-xr-x   2 root    root          11 Jul 25 09:22 .
1835416      0 -r--r--r--   1 root    root          0 Jul 25 09:12 ./f-
2.txt
1835419      0 -rw-r--r--   1 root    root          0 Jul 25 09:17 ./f-
5.txt
1835420      0 -rwxrwxrwx   1 root    root          0 Jul 25 09:17 ./f-
6.txt
1835421      0 -r---w---x   1 root    root          0 Jul 25 09:22 ./f-
8.txt
```

报错

```
root@ubuntu24:~# find -perm +444 -ls
find: invalid mode '+444'
```

ugo三个角色至少都要有r权限

```
root@ubuntu24:~# find -perm -444 -ls
```

```
1835414      0 drwxr-xr-x   2 root    root          11 Jul 25 09:22 .
1835416      0 -r--r--r--   1 root    root          0 Jul 25 09:12 ./f-
2.txt
1835419      0 -rw-r--r--   1 root    root          0 Jul 25 09:17 ./f-
5.txt
1835420      0 -rwxrwxrwx   1 root    root          0 Jul 25 09:17 ./f-
6.txt
```

只关注属主权限，6拆分成4+2，所以只要属主有r或w权限即可

```

root@ubuntu24:~# find -perm /600 -ls
1835414      0 drwxr-xr-x   2 root    root      111 Jul 25 09:22 .
1835416      0 -r--r--r--   1 root    root        0 Jul 25 09:12 ./f-
2.txt
1835417      0 --w--w--w-   1 root    root        0 Jul 25 09:12 ./f-
3.txt
1835419      0 -rw-r--r--   1 root    root        0 Jul 25 09:17 ./f-
5.txt
1835420      0 -rwxrwxrwx   1 root    root        0 Jul 25 09:17 ./f-
6.txt
1835421      0 -r---w---x   1 root    root        0 Jul 25 09:22 ./f-
8.txt

```

只关注属主权限，6拆分成4+2，所以只要属主至少要有有rw权限

```

root@ubuntu24:~# find -perm -600 -ls
1835414      0 drwxr-xr-x   2 root    root      111 Jul 25 09:22 .
1835419      0 -rw-r--r--   1 root    root        0 Jul 25 09:17 ./f-
5.txt
1835420      0 -rwxrwxrwx   1 root    root        0 Jul 25 09:17 ./f-
6.txt

```

1.1.4 类型查找

基础知识

选项解析

```

-empty      #空文件或空目录
-prune      #跳过，排除指定目录，必须配合 -path使用
-type TYPE  #指定文件类型 *****

```

type 值

```

f      #普通文件
d      #目录文件
l      #符号链接文件
s      #套接字文件
b      #块设备文件
c      #字符设备文件
p      #管道文件

```

简单实践

根据类型查找

查看当前目录下的所有目录文件

```
root@ubuntu24:~# find -type d
```

```
.  
./dir1  
./dir1/dir2  
./dir1/dir2/dir3
```

查找run 目录下所有管道文件

```
root@ubuntu24:~# find /run/ -type lnp
```

空文件查找

空文件或空目录

```
root@ubuntu24:~# find dir1/dir2/dir3/ -empty
```

```
dir1/dir2/dir3/fx  
dir1/dir2/dir3/fy  
dir1/dir2/dir3/dir4
```

查找空目录

```
root@ubuntu24:~# find dir1/dir2/dir3/ -empty -type d
```

```
dir1/dir2/dir3/dir4
```

查找空文件

```
root@ubuntu24:~# find dir1/dir2/dir3/ -empty -type f
```

```
dir1/dir2/dir3/fx  
dir1/dir2/dir3/fy
```

排查查找

#查找所有 txt文件

```
root@ubuntu24:~# find -name "*.txt"
```

```
./dir1/fa.txt  
./dir1/fb.txt  
./test-a.txt  
./test-b.txt  
./dir4/f-x.txt  
./dir4/f-y.txt
```

#排除 dir1 目录中的 txt 文件，但还是会输出 dir1

```
root@ubuntu24:~# find -path './dir1' -prune -o -name "*.txt"
```

```
./dir1  
./test-a.txt  
./test-b.txt  
./dir4/f-x.txt  
./dir4/f-y.txt
```

#action 作用在后一个条件上

```
root@ubuntu24:~# find -path './dir1' -prune -o -name "*.txt" -print
```

```
./test-a.txt  
./test-b.txt  
./dir4/f-x.txt  
./dir4/f-y.txt
```

```
#排除多个目录
root@ubuntu24:~# find \( -path './dir1' -o -path './dir4' \) -prune -o -name
"*.txt" -print
./test-a.txt
./test-b.txt
```

1.1.5 组合查找

--- 扩展资料 ---

基础知识

功能简介

`find`命令支持使用逻辑操作符（如`-and`、`-or`、`-not`）来组合多个搜索条件，以构建复杂的搜索逻辑。此外，还可以使用括号来分组条件，但括号需要被转义或使用引号括起来。

属性解析

| | |
|---------------------|---------------------------------|
| <code>-a</code> | <code>#与</code> ，多条件默认就是与关系，可省略 |
| <code>-o</code> | <code>#或</code> |
| <code>-not !</code> | <code>#非</code> |

搜索结果的处理

除了搜索功能外，`find`命令还可以对搜索结果执行各种操作。常用的操作选项包括：

- `-print`：打印搜索结果（默认行为）。
- `-exec`：对每个匹配的文件执行指定的命令。命令以`{}`代表当前文件，命令结尾以`;`或`+`结束。
- `-ok`：与`-exec`类似，但在执行每个命令前都会提示用户确认。
- `-delete`：直接删除匹配的文件或目录（使用时要小心）。
- `-ls`：以`ls -dils`的格式显示匹配的文件。

简单实践

与或非实践

```
默认 -a, 可省略
root@ubuntu24:~# find -name "*.log" -a -type f
./test-a.log
./test-b.log
./test-A.log
./test-B.log

或
root@ubuntu24:~# find -name "test*.log" -o -name "test*.txt"
./test-a.log
./test-a.txt
./test-b.log
./test-b.txt
```

非

```
root@ubuntu24:~# find dir1/dir2/dir3/ -empty -not -type d
dir1/dir2/dir3/fx
dir1/dir2/dir3/fy
```

非

```
root@ubuntu24:~# find dir1/dir2/dir3/ -empty ! -type d
dir1/dir2/dir3/fx
dir1/dir2/dir3/fy
```

动作实践

```
root@ubuntu24:~# find -user sswang -o -name "*log"
./dir1/fa.txt
./dir1/fb.txt
./test-a.log
./test-b.log
./test-A.log
./test-B.log
```

此处 `ls` 只列出了后一个条件的匹配

```
root@ubuntu24:~# find -user sswang -o -name "*log" -ls
138733453      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
a.log
138733455      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
b.log
138733459      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
A.log
138733461      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
B.log
```

把条件括起来才表示全部

```
root@ubuntu24:~# find \( -user sswang -o -name "*log" \) -ls
202397817      0 -rw-r--r--   1 sswang   root          0 Jul 23 10:48
./dir1/fa.txt
202397821      0 -rw-r--r--   1 sswang   root          0 Jul 23 10:48
./dir1/fb.txt
138733453      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
a.log
138733455      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
b.log
138733459      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
A.log
138733461      0 -rw-r--r--   1 root    root          0 Jul 23 10:30 ./test-
B.log
```

1.1.6 组合进阶

--- 扩展资料 ---

基础知识

进阶属性

德·摩根定律:

- (非A) 且 (非B) = 非(A或B)
- (非A) 或 (非B) = 非(A且B)

$\neg A \wedge \neg B = \neg(A \vee B)$

$\neg A \vee \neg B = \neg(A \wedge B)$

简单实践

组合实践

```
# !A -a !B
root@ubuntu24:~# find ! -name "f*" -a ! -name "dir*"
.
./issue
./test-a.log
./test-a.txt
./test-b.log
./test-b.txt

# !(A -o b)
root@ubuntu24:~# find ./ ! \( -name "f*" -o -name "dir*" \)
./
./issue
./test-a.log
./test-a.txt
./test-b.log
./test-b.txt

# !A -o !B
root@ubuntu24:~# find ! -type f -o ! -empty
.
./dir1
./dir1/dir2
./dir1/dir2/dir3
./dir1/dir2/dir3/dir4
./fstab
./issue

# !(A -a B)
root@ubuntu24:~# find ! \( -type f -a -empty \)
.
./dir1
./dir1/dir2
./dir1/dir2/dir3
./dir1/dir2/dir3/dir4
./fstab
./issue
```

1.1.7 文件属性查找

基础知识

文件大小属性解读

`-size [+|-]N UNIT`

N为数字, UNIT为常用单位 k, M, G, c(byte) 等

#解释

10k

#表示(9k,10k],大于9k 且小于或等于10k

-10k

#表示[0k,9k],大于等于0k 且小于或等于9k

+10k

#表示(10k,∞), 大于10k

文件时间属性解读

以天为单位

`-atime [+|-]N``-mtime [+|-]N``-ctime [+|-]N`

以分钟为单位

`-amin [+|-]N``-mmin [+|-]N``-cmin [+|-]N`

#解释

N

#表示[N,N+1),大于或等于N, 小于N+1, 表示第N天(分钟)

+N

#表示[N+1,∞),大于或等于N+1, 表示N+1天之前(包括)

-N

#表示[0,N),大于或等于0, 小于N, 表示N天(分钟)内



简单实践

文件大小查找

大于2K, 小于或等于3K的文件

root@ubuntu24:~# find /var/log/ -size 3k -ls

658666 4 -rw-r----- 1 root adm 2232 May 4 17:01

/var/log/installer/autoinstall-user-data

658664 4 -rw-r----- 1 root adm 2948 May 4 17:00

/var/log/installer/subiquity-curtin-install.conf

658719 4 -rw----- 1 root root 2218 May 10 11:07 /var/log/vmware-

vmtoolsd-root.log

657067 4 -rw----- 1 root root 3057 May 10 08:10 /var/log/vmware-

vmsvc-root.3.log


```
655614 4 -rw----- 1 root root 2394 May 10 11:06 /var/log/vmware-
vmsvc-root.1.log
656087 4 -rw----- 1 root root 2394 May 10 11:04 /var/log/vmware-
vmsvc-root.2.log
```

小于或等于2k

```
root@ubuntu24:~# find /var/log/ -size -3k -name "*.log" -ls
658964 0 -rw-r--r-- 1 root root 0 May 4 21:03
/var/log/landscape/sysinfo.log
657068 4 -rw-r--r-- 1 root root 193 May 9 22:28
/var/log/vmware-network.log
658782 4 -rw-r--r-- 1 root root 193 May 8 08:22
/var/log/vmware-network.3.log
```

大于2k

```
root@ubuntu24:~# find /var/log/ -size +2k -name "*.log" -ls
655369 4 drwxrwxr-x 10 root syslog 4096 May 10 11:07
/var/log/
658886 1832 -rw-r----- 1 syslog adm 1875675 May 10 13:20
/var/log/syslog
```

大于2k, 小于或等于9k

```
root@ubuntu24:~# find /var/log/ -size +2k -size -10k -name "*.log" -ls
655369 4 drwxrwxr-x 10 root syslog 4096 May 10 11:07
/var/log/
658719 4 -rw----- 1 root root 2218 May 10 11:07
/var/log/vmware-vmtoolsd-root.log
657067 4 -rw----- 1 root root 3057 May 10 08:10
/var/log/vmware-vmsvc-root.3.log
```

时间范围查找

查找五分钟前被访问过的文件

```
root@ubuntu24:~# find -amin +5
```

查找十分钟前被访问过的文件

```
root@ubuntu24:~# find -amin +10
```

1.1.8 动作进阶

基础知识

处理动作

| | |
|----------------------------------|---|
| <code>-print</code> | #默认的处理动作, 显示至屏幕 |
| <code>-print0</code> | #不换行输出, 常用于配合xargs |
| <code>-ls</code> | #类似于对查找到的文件执行" <code>ls -ils</code> "命令格式输出 |
| <code>-fls file</code> | #查找到的所有文件的长格式信息保存至指定文件中, 相当于 <code>-ls > file</code> |
| <code>-delete</code> | #删除查找到的文件, 慎用! |
| <code>-ok COMMAND {} \;</code> | #对查找到的每个文件执行由COMMAND指定的命令, 对于每个文件执行命令之前, 都会交互式要求用户确认 |
| <code>-exec COMMAND {} \;</code> | #对查找到的每个文件执行由COMMAND指定的命令 |
| <code>{}</code> | #用于引用查找到的文件名称自身 |

简单实践

动作演示实践

默认 -print

```
root@ubuntu24:~# find
```

```
.  
./f-1.txt  
./f-2.txt  
./f-3.txt
```

长格式显示

```
root@ubuntu24:~# find -ls
```

```
 1835414      0 drwxr-xr-x   2 root    root          213 Jul 25 09:45 .  
 1835415      0 -----   1 root    root           0 Jul 25 09:12 ./f-  
1.txt  
 1835416      0 -r--r--r--   1 root    root           0 Jul 25 09:12 ./f-  
2.txt  
 1835417      0 --w--w--w-   1 root    root           0 Jul 25 09:12 ./f-  
3.txt
```

查找结果保存至文件

```
root@ubuntu24:~# find -fls ls.log
```

删除

```
root@ubuntu24:~# find -name "*.sh" -delete
```

备份以log结尾的文件

```
root@ubuntu24:~# find -name "*.log" -exec cp {} {}.bak \;
```

```
root@ubuntu24:~# ls *log*
```

```
f-1.log  f-1.log.bak  f-2.log  f-2.log.bak  f-3.log  f-3.log.bak
```

删除15分钟内没被访问过的文件

```
root@ubuntu24:~# find -amin +15 -ok rm {} \;
```

```
< rm ... ./f-1.txt > ?
```

将other权限有w的文件的权限去掉w权限

```
root@ubuntu24:~# find -perm -002 -exec chmod o-w {} \;
```

查找权限为644，后缀为sh的普通文件，增加执行权限

```
root@ubuntu24:~# find -type f -perm 644 -name "*.sh" -exec chmod 755 {} \;
```

批量改名

```
root@ubuntu24:~# find -name "*.bak" -exec rename 's/.bak$//' {} \;
```

1.1.9 xargs组合

基础知识

属性解析

由于很多命令不支持管道|来传递参数，为了使用更灵活的参数，我们就要用 `xargs` 产生命令参数，`xargs` 可以读入 `stdin` 的数据，并且以空格符或回车符将 `stdin` 的数据分隔，使其成为另一个命令的参数，
另外，许多命令不能接受过多参数，命令执行可能会失败，`xargs` 也可以解决此问题；

格式解读

```
xargs [OPTION]... COMMAND [INITIAL-ARGS]...
```

#常用选项

| | |
|---------------------------------------|---|
| <code>-0 --null</code> | #用 <code>ascii</code> 中的0或 <code>null</code> 作分隔符 |
| <code>-a --arg-file=FILE</code> | #从文件中读入作为输入 |
| <code>-d --delimiter=CHARACTER</code> | #指定分隔符 |
| <code>-E END</code> | #指定结束符，执行到此处时停止，不管后面的数据 |
| <code>-L --max-lines=N</code> | #从标准输入一次读取N行送给 <code>command</code> 命令 |
| <code>-l</code> | #同上 |
| <code>-n --max-args=MAX-ARGS</code> | #一次执行用几个参数 |
| <code>-p --interactive</code> | #每次执行前确认 |
| <code>-r --no-run-if-empty</code> | #当xargs的输入为空的时候则停止xargs，不用再去执行了 |
| <code>-s --max-chars=MAX-CHARS</code> | #命令行最大字符数 |
| <code>-t --verbose</code> | #显示过程，先打印要执行的命令 |
| <code>-x --exit</code> | #退出，主要配合-s使用 |

简单实践

ls实践

```
root@ubuntu24:~# ls f-1.log
f-1.log
```

无法用管道传参

```
root@ubuntu24:~# echo "f-1.log" | ls
...
```

命令展开

```
root@ubuntu24:~# ls `echo f-1.txt`
f-1.txt
```

`xargs`

```
root@ubuntu24:~# echo "f-1.log" | xargs ls
f-1.log
```

还可以让不支持标准输入的命令支持标准输入(ctr+d 结束)

```
root@ubuntu24:~# xargs ls -l
f-1.log
-rw-r--r-- 1 root root 0 Jul 25 09:43 f-1.log
```

让命令支持标准输入重定向

```
root@ubuntu24:~# cat ls.log
f-1.log
f-2.log
```

```
root@ubuntu24:~# xargs -a ls.log ls -l
-rw-r--r-- 1 root root 0 Jul 25 09:43 f-1.log
-rw-r--r-- 1 root root 0 Jul 25 09:43 f-2.log
```

辅助处理

```
root@ubuntu24:~# seq 3
1
2
3

root@ubuntu24:~# seq 3 | xargs
1 2 3

指定分割符
root@ubuntu24:~# echo -e "1-2-3\c" | xargs -d '-'
1 2 3

root@ubuntu24:~# echo {1..5}
1 2 3 4 5

root@ubuntu24:~# echo {1..5} | xargs
1 2 3 4 5

指定每次使用2个参数
root@ubuntu24:~# echo {1..5} | xargs -n 2
1 2
3 4
5

批量创建用户并显示命令
root@ubuntu24:~# echo user{1..5} |xargs -t -n1 useradd
useradd user1
useradd user2
useradd user3
useradd user4
useradd user5

#批量删除用户
root@ubuntu24:~# echo user{1..5} |xargs -n1 userdel -r
```

批量创建文件

```
root@ubuntu24:~# df -i
```

| Filesystem | Inodes | IUsed | IFree | IUse% | Mounted on |
|-----------------------------------|---------|--------|---------|-------|-------------|
| tmpfs | 248529 | 821 | 247708 | 1% | /run |
| /dev/mapper/ubuntu--vg-ubuntu--lv | 6488064 | 129118 | 6358946 | 2% | / |
| tmpfs | 248529 | 1 | 248528 | 1% | /dev/shm |
| tmpfs | 248529 | 4 | 248525 | 1% | /run/lock |
| /dev/sda2 | 131072 | 320 | 130752 | 1% | /boot |
| tmpfs | 49705 | 26 | 49679 | 1% | /run/user/0 |

```
root@ubuntu24:~# cd /boot/
[root@ubuntu24 boot]# mkdir test
```

```
[root@ubuntu24 boot]# cd test/

#参数太长, 执行失败
[root@ubuntu24 test]# touch f-{1..130752}.txt
-bash: /usr/bin/touch: Argument list too long

#一次创建1w个
[root@ubuntu24 test]# echo f-{1..130752}.txt | xargs -n 10000 touch

#inode 资源耗尽
[root@ubuntu24 test]# df -i
```

| Filesystem | Inodes | IUsed | IFree | IUse% | Mounted on |
|-----------------------------------|---------|--------|---------|-------|-------------|
| tmpfs | 248529 | 821 | 247708 | 1% | /run |
| /dev/mapper/ubuntu--vg-ubuntu--lv | 6488064 | 129118 | 6358946 | 2% | / |
| tmpfs | 248529 | 1 | 248528 | 1% | /dev/shm |
| tmpfs | 248529 | 4 | 248525 | 1% | /run/lock |
| /dev/sda2 | 131072 | 131072 | 0 | 100% | /boot |
| tmpfs | 49705 | 26 | 49679 | 1% | /run/user/0 |

批量下载B站视频

```
root@ubuntu24:~# yum install python3-pip -y
root@ubuntu24:~# pip3 install you-get

如果前一条install报错, 可以用这个
root@ubuntu24:~# pip3 --default-timeout=100 install -U you-get
```

批量下载

```
root@ubuntu24:~# seq 60 | xargs -i -P3 you-get
https://www.bilibili.com/video/BV1kX4y1S7dT?p={}
```

以ascii中的空白符分隔参数

```
root@ubuntu24:~# ls
'a b'  f-1.txt  f-2.txt  f-3.txt

root@ubuntu24:~# find -type f | xargs echo
./f-1.txt ./f-2.txt ./f-3.txt ./a b

#在此处有空格的文件名被拆分成两个文件了, xargs 默认以空格拆分
root@ubuntu24:~# find -type f | xargs ls
ls: cannot access './a': No such file or directory
ls: cannot access 'b': No such file or directory
./f-1.txt ./f-2.txt ./f-3.txt

#正常显示
root@ubuntu24:~# find -type f -print0 |xargs -0 ls
'./a b'  ./f-1.txt  ./f-2.txt  ./f-3.txt
```

批量改名

```
find ./ -name "*.txt" | xargs -I {} rename 's/.txt$//' {}
find ./ -name "*-bak" | xargs rename 's/-bak$//'
find ./ -name "*.txt" | xargs rename 's/.txt$//' {}
```

1.2 打包压缩

1.2.1 gz包

基础知识

属性简介

来自于 `gzip` 包
对应的文件是 `.gz` 后缀

命令解读

命令格式

```
gzip [OPTION]... FILE ...
gunzip [OPTION]... FILE ...
```

常用选项

| | |
|------------------------------|-------------------------------|
| <code>-c --stdout</code> | #将压缩数据输出到标准输出中，并保留原文件 |
| <code>-d --decompress</code> | #解压缩, 相当于 <code>gunzip</code> |
| <code>-f --force</code> | #覆盖已存在目标文件 |
| <code>-k --keep</code> | #保留原文件 |
| <code>-l --list</code> | #显示原文件大小，压缩文件大小，压缩比，压缩前文件名 |
| <code>-q --quiet</code> | #安静模式，忽略警告 |
| <code>-r --recursive</code> | #递归压缩目录内所有文件 |
| <code>-S --suffix=SUF</code> | #指定压缩文件后缀 |
| <code>-t --test</code> | #测试，检测压缩文件是否完整 |
| <code>-v --verbose</code> | #显示过程 |
| <code>-1 --fast</code> | #最快压缩，压缩比最底，但压缩速度快 |
| <code>-9 --best</code> | #最好压缩，压缩比最高，但压缩速度慢 |
| <code>-N</code> | #指定压缩等级，取值为1-9之间，默认6 |

简单实践

普通实践

```
#保留原文件，并显示压缩过程
[root@ubuntu24 0510]# gzip -vk fstab passwd
fstab:  50.3% -- created fstab.gz
passwd: 61.2% -- created passwd.gz

[root@ubuntu24 0510]# ls
dir1  fstab  fstab.gz  passwd  passwd.gz  shadow

#重定向到文件
[root@ubuntu24 0510]# gzip fstab -c > fstab.gz
```

```
#管道
[root@ubuntu24 0510]# cat passwd | gzip > pwd.gz

#自定义后缀名
[root@ubuntu24 0510]# gzip -kv fstab -S .gzzz
fstab: 50.3% -- created fstab.gzzz

#递归压缩目录
[root@ubuntu24 0510]# gzip -vrk dir1/
dir1/dir2/messages: 86.1% -- created dir1/dir2/messages.gz
dir1/fstab: 50.3% -- created dir1/fstab.gz
dir1/dnf.log: 91.2% -- created dir1/dnf.log.gz
dir1/passwd: 61.2% -- created dir1/passwd.gz

#显示压缩文件信息
[root@ubuntu24 0510]# gzip -l fstab.gz
      compressed      uncompressed   ratio uncompressed_name
          382             720  50.3%  fstab

[root@ubuntu24 0510]# gzip -l fstab.gz passwd.gz
      compressed      uncompressed   ratio uncompressed_name
          382             720  50.3%  fstab
         1134            2858  61.2%  passwd
         1516            3578  58.3% (totals)
```

解压实践

```
#解压缩
[root@ubuntu24 0510]# gunzip -vkf fstab.gz passwd.gz
fstab.gz: 50.3% -- replaced with fstab
passwd.gz: 61.2% -- replaced with passwd
```

1.2.2 tar包

基础知识

简介

tar 即 Tape ARchive 磁带归档，可以对目录和多个文件打包成一个文件进行归档；其本身不具备压缩功能，但可以使用参数调用相应的压缩命令进行压缩；前提是当前系统中有安装该压缩工具。此命令可以保留文件属性，推荐使用；对应的文件是 **.tar** 后缀。

命令解读

```
tar [OPTION...] [FILE]...
#tar {A|c|d|r|t|u|x}[GnSkUWompsMBiajJzZhPlRvwo] [ARG...]

#必选项 {A|c|d|r|t|u|x}
-A|--catenate|--concatenate      #追加 tar 文件至归档
-c|--create                        #创建一个新归档
-d|--diff|--compare               #找出归档和文件系统的差异
--delete                          #从归档(非磁带!)中删除
-r|--append                       #追加文件至归档结尾
-t|--list                         #列出归档内容
--test-label                      #测试归档卷标并退出
-u|--update                       #仅追加比归档中副本更新的文件
-x|--extract|--get                #从归档中解出文件
```

其他选项

```
#OPTIONS选项 [GnSkUWompsMBiajJzZhPlRvwo], 这些选项要注意位置

-G|--incremental                  #处理老式的 GNU 格式的增量备份
-n|--seek                        #归档可检索
-S|--sparse                      #高效处理离散文件
-k|--keep-old-files              #解包时不覆盖已有的文件
-U|--unlink-first                #解压之前先删除文件的链接
-w|--verify                     #在写入以后尝试校验归档
-O|--to-stdout                   #解压文件至标准输出
    --to-command=COMMAND        #将解压的文件通过管道传送至另一个程序
-m|--touch                      #不要解压文件的修改时间
-p|--preserve-permissions|--same-permissions    #保留文件权限信息
-s|--preserve-order|--same-order #成员参数按归档中的文件顺序列出
-M|--multi-volume               #创建/列出/解压多卷归档文件
-B|--read-full-records          #读取时重新分块(只对 4.2BSD 管道有效)
-i|--ignore-zeros               #忽略归档中的零字节块(即文件结尾)
-a|--auto-compress              #使用归档后缀名来决定压缩程序

-h|--dereference                 #将软链接指向的目标文件也压缩打包
    --hard-dereference          #将硬链接指向的目标文件也压缩打包
-P|--absolute-names             #不要从文件名中清除引导符 '/'
-l|--check-links                #只输出非链接文件的信息
-R|--block-number               #每个信息都显示归档内的块数
-v|--verbose                    #列出文件详细信息
-w|--interactive|--confirmation #操作前手动确认
-o                              #用老旧的 v7 tar 格式打包或解包

压缩选项
-j|--bzip2                      #使用 bzip2 压缩或解压缩
-J|--xz                        #使用 xz 压缩或解压缩
    --lzip|--lzma|--lzop        #lzip|xz --format=lzma|lzop
-z|--gzip|--gunzip|--ungzip     #通过 gzip 压缩或解压缩
-Z|--compress|--uncompress     #通过 compress 压缩或解压缩
```


其他选项

| | |
|------------------------|---------------|
| --show-defaults | #显示 tar 默认选项 |
| --exclude | #排除文件 |
| -C --directory=DIR | #指定目录 |
| -T --files-from=FILE | #从文件中读取要处理的文件 |
| -X --exclude-from=FILE | #从文件中读取要排除的文件 |
| --version | #显示版本号 |

简单实践

压缩实践

```
#默认采用相对路径
[root@ubuntu24 0510]# tar -cf etc.tar /etc
tar: Removing leading `/' from member names

#P保留路径
[root@ubuntu24 0510]# tar -cPf etc2.tar /etc
```

打包

```
#只打包，不压缩
[root@ubuntu24 0510]# tar -cvf test.tar f1.txt f2.txt
f1.txt
f2.txt

#递归打包目录
[root@ubuntu24 0510]# tar -cvf log.tar /var/log

#大小相同
[root@ubuntu24 0510]# du -sh /var/log/
13M /var/log/

[root@ubuntu24 0510]# ll -h log.tar
-rw-r--r-- 1 root root 13M Jul 26 15:57 log.tar
```

只打包目录内的文件，不所括目录本身

```
[root@ubuntu2204 0510]# cd /etc/
[root@ubuntu2204 etc]# tar -cf etc.tar *

#先指定目录
[root@ubuntu2204 0510]# tar -C /etc/ -cf etc.tar ./
```

范例：追加和删除

不支持对压缩文件追加

```
[root@ubuntu2204 0510]# tar -tvf test.tar
-rw-r--r-- root/root      4453 2022-07-26 15:19 f1.txt
-rw----- root/root    1946847 2022-07-26 14:09 f2.txt
```

```

#追加
[root@ubuntu2204 0510]# tar -rf test.tar f3.txt passwd

[root@ubuntu2204 0510]# tar -tvf test.tar
-rw-r--r-- root/root      4453 2022-07-26 15:19 f1.txt
-rw----- root/root     1946847 2022-07-26 14:09 f2.txt
-rw----- root/root     1814952 2022-07-26 14:09 f3.txt
-rw-r--r-- root/root      2858 2022-07-25 22:09 passwd

#删除
[root@ubuntu2204 0510]# tar --delete -vf test.tar f1.txt f2.txt

[root@ubuntu2204 0510]# tar -tvf test.tar
-rw----- root/root     1814952 2022-07-26 14:09 f3.txt
-rw-r--r-- root/root      2858 2022-07-25 22:09 passwd

#将test.tar 中的文件解出来再打包到 test2.tar 中, test.tar 中不删除
[root@ubuntu2204 0510]# tar -A test.tar -f test2.tar

```

范例：列出包内文件

```

[root@ubuntu2204 0510]# tar -tvf test.tar
-rw----- root/root     1814952 2022-07-26 14:09 f3.txt
-rw-r--r-- root/root      2858 2022-07-25 22:09 passwd

```

范例：解包

```

[root@ubuntu2204 0510]# tar -xf log.tar

#指定目录
[root@ubuntu2204 0510]# tar -xf log.tar -C /tmp

```

范例：打包并压缩

```

[root@ubuntu2204 0510]# tar -zcvf etc.tar.gz /etc/
[root@ubuntu2204 0510]# tar -jcvf etc.tar.bz2 /etc/
[root@ubuntu2204 0510]# tar -Jcvf etc.tar.xz /etc/

[root@ubuntu2204 0510]# ll -h etc.tar.*
-rw-r--r-- 1 root root 4.7M Jul 26 16:37 etc.tar.bz2
-rw-r--r-- 1 root root 6.4M Jul 26 16:36 etc.tar.gz
-rw-r--r-- 1 root root 4.0M Jul 26 16:37 etc.tar.xz

[root@ubuntu2204 0510]# tar -xf etc.tar.gz -C /tmp/etc-gz/
[root@ubuntu2204 0510]# tar -xf etc.tar.bz2 -C /tmp/etc-bz2/
[root@ubuntu2204 0510]# tar -xf etc.tar.xz -C /tmp/etc-xz/

```

范例：从文件中读取要打包的文件

```
[root@ubuntu2204 0510]# cat list.txt
f1.txt
f2.txt

[root@ubuntu2204 0510]# tar -zcvf x.tar.gz -T list.txt
f1.txt
f2.txt
```

排除和包含文件

```
#指定跳过的文件
[root@ubuntu2204 0510]# tar zcvf /root/a.tgz --exclude=/app/host1 --
exclude=/app/host2 /app

#从文件读取
[root@ubuntu2204 0510]# tar zcvf mybackup.tgz -T /root/includefilelist -x
/root/excludefilelist
```

查看默认选项

```
[root@ubuntu2204 0510]# tar --show-defaults
--format=gnu -f- -b20 --quoting-style=escape --rmt-command=/etc/rmt --rsh-
command=/usr/bin/ssh
```

1.2.3 zip包

基础知识

简介

zip 可以实现打包目录和多个文件成一个文件并压缩，但可能会丢失文件属性信息，如：所有者和组信息

分别来自于 zip 和 unzip 包

对应的文件是 .zip 后缀

命令解读

```
zip [OPTION]... zipfile [FILE]...
unzip [OPTION]... zipfile [FILE]...
```

#zip常用选项

| | |
|----|---|
| -f | #更换较新的文件到压缩文件内 |
| -u | #如果压缩包内有，则更新，如果没有，则追加进去 |
| -d | #从压缩包内删除指定的文件 |
| -m | #将文件压缩之后，删除原始文件 |
| -r | #递归压缩目录 |
| -j | #只保存文件名称及其内容，而不存放任何目录名称 |
| -l | #压缩文件时，把LF字符置换成LF+CR字符，unzip -l 表示显示压缩文件的内容 |
| -1 | #最快压缩，数字1 |
| -9 | #最高压缩比，数字9 |
| -q | #安静模式 |
| -v | #显示过程 |
| -c | #替每个被压缩的文件加上注释 |

```
-Z          #给压缩包加注释，unzip -z 查看注释
-x          #压缩时排除指定文件
-i          #仅压缩指定文件
-D          #压缩文件内不建立目录名称
-T          #测试，检测压缩文件是否完整
-X          #不保存额外的文件属性
-y          #直接保存符号连接，而非该链接所指向的文件
-n          #不压缩以特定字符串结尾的文件
-P          #加密码
```

#unzip常用选项

```
-p          #将压缩内容通过管道传送
-l          #显示压缩文件内所包含的文件
-t          #测试，检测压缩文件是否完整
-z          #查看注释
-v          #列出包内文件信息
-x          #指定不需要解压缩的文件
-d          #指定解压后的目标目录
-n          #解压缩时不要覆盖原有的文件
-q          #安静模式
-o          #直接覆盖
-a          #对文本文件进行必要的字符转换
-j          #不处理压缩文件中原有的目录路径
-C          #压缩文件中的文件名称区分大小写
-L          #将压缩文件中的全部文件名改为小写
-X          #解压缩时同时回存文件原来的UID/GID
-V          #保留VMS的文件版本信息
-K          #解压缩后还原权限
-M          #将输出结果送到more程序处理
```

简单实践

范例：

```
[root@ubuntu2204 0510]# zip -v msg.zip messages
  adding: messages (in=402508) (out=56530) (deflated 86%)
total bytes=402508, compressed=56530 -> 86% savings
```

#查看内容

```
[root@ubuntu2204 0510]# unzip -l msg.zip
```

Archive: msg.zip

| Length | Date | Time | Name |
|--------|------------|-------|----------|
| 402508 | 07-26-2022 | 09:03 | messages |
| 402508 | | | 1 file |

#管道，往压缩包中添加新文件

```
[root@ubuntu2204 0510]# zip msg.zip passwd
```

#查看

```
[root@ubuntu2204 0510]# unzip -l msg.zip
```

```
Archive:  msg.zip
  Length      Date    Time    Name
-----
  402508  07-26-2022  09:03  messages
    2858  07-25-2022  22:09  passwd
-----
  405366
                        2 files
```

#只压缩txt

```
[root@ubuntu2204 0510]# zip -i"*txt" txt.zip *
  adding: f1.txt (deflated 86%)
  adding: f2.txt (deflated 86%)
  adding: f3.txt (deflated 86%)
```

#只压缩txt

```
[root@ubuntu2204 0510]# zip txt2.zip */*txt
  adding: f1.txt (stored 0%)
  adding: f2.txt (deflated 86%)
  adding: f3.txt (deflated 86%)
```

范例：递归压缩

```
[root@ubuntu2204 0510]# tree dir1/
dir1/
├── dir2
│   └── messages
├── dnf.log
├── fstab
└── passwd
```

1 directory, 4 files

#递归压缩

```
[root@ubuntu2204 0510]# zip -r test1.zip dir1/
  adding: dir1/ (stored 0%)
  adding: dir1/dir2/ (stored 0%)
  adding: dir1/dir2/messages (deflated 86%)
  adding: dir1/fstab (deflated 50%)
  adding: dir1/passwd (deflated 61%)
  adding: dir1/dnf.log (deflated 91%)
```

#当前目录所有文件

```
[root@ubuntu2204 0510]# cd dir1/
[root@ubuntu2204 0510]# zip ../test2.zip *
  adding: dir2/ (stored 0%)
  adding: dnf.log (deflated 91%)
  adding: fstab (deflated 50%)
  adding: passwd (deflated 61%)
```

```
[root@ubuntu2204 0510]# cd ..
[root@ubuntu2204 0510]# ll test*
-rw-r--r-- 1 root root 76768 Jul 26 15:02 test1.zip
-rw-r--r-- 1 root root 46255 Jul 26 15:02 test2.zip
```

#查看

```
[root@ubuntu2204 0510]# unzip -l test1.zip
Archive:  test1.zip
  Length      Date    Time    Name
-----
      0  07-26-2022 15:01   dir1/
      0  07-25-2022 22:30   dir1/dir2/
 216607  07-25-2022 21:41   dir1/dir2/messages
    720  07-25-2022 21:39   dir1/fstab
   2858  07-25-2022 21:43   dir1/passwd
 501645  07-25-2022 21:40   dir1/dnf.log
-----
 721830
                        6 files
```

```
[root@ubuntu2204 0510]# unzip -l test2.zip
Archive:  test2.zip
  Length      Date    Time    Name
-----
      0  07-25-2022 22:30   dir2/
 501645  07-25-2022 21:40   dnf.log
    720  07-25-2022 21:39   fstab
   2858  07-25-2022 21:43   passwd
-----
 505223
                        4 files
```

范例：设置密码

#非交互式加密解密

```
[root@ubuntu2204 0510]# zip -vP 123456 test.zip dnf.log
adding: dnf.log (in=501645) (out=44208) (deflated 91%)
total bytes=501645, compressed=44220 -> 91% savings
```

```
[root@ubuntu2204 0510]# unzip -P 123456 test.zip
Archive:  test.zip
inflating: dnf.log
```

#交互式加密解密

```
[root@ubuntu2204 0510]# zip -ve test.zip dnf.log
Enter password:
Verify password:
updating: dnf.log (in=501645) (out=44208) (deflated 91%)
total bytes=501645, compressed=44220 -> 91% savings
```

```
[root@ubuntu2204 0510]# unzip test.zip
Archive:  test.zip
[test.zip] dnf.log password:
inflating: dnf.log
```

范例：更新和删除

```
[root@ubuntu2204 0510]# unzip -l txt.zip
Archive:  txt.zip
  Length      Date    Time    Name
-----
 2226376  07-26-2022 14:12   f1.txt
 1946847  07-26-2022 14:09   f2.txt
 1814952  07-26-2022 14:09   f3.txt
-----
 5988175                      3 files

[root@ubuntu2204 0510]# echo "123">f1.txt
[root@ubuntu2204 0510]# ll f1.txt passwd
-rw----- 1 root root    4 Jul 26 14:19 f1.txt
-rw-r--r-- 1 root root 2858 Jul 25 22:09 passwd

#如果包内不存在，则追加
[root@ubuntu2204 0510]# zip -vu txt.zip f1.txt passwd
updating: f1.txt      (in=4) (out=4) (stored 0%)
  adding: passwd      (in=2858) (out=1109) (deflated 61%)
total bytes=3764661, compressed=525257 -> 86% savings

#再次查看，f1.txt 更新了
[root@ubuntu2204 0510]# unzip -l txt.zip
Archive:  txt.zip
  Length      Date    Time    Name
-----
      4  07-26-2022 14:19   f1.txt
 1946847  07-26-2022 14:09   f2.txt
 1814952  07-26-2022 14:09   f3.txt
   2858  07-25-2022 22:09   passwd
-----
 3764661                      4 files

#仅更新，不追加新文件
[root@ubuntu2204 0510]# zip -vf txt.zip f1.txt messages
freshening: f1.txt   (in=7) (out=7) (stored 0%)
total bytes=3764664, compressed=525260 -> 86% savings

#messages没有被追加到压缩文件内
[root@ubuntu2204 0510]# unzip -l txt.zip
Archive:  txt.zip
  Length      Date    Time    Name
-----
      7  07-26-2022 14:24   f1.txt
 1946847  07-26-2022 14:09   f2.txt
 1814952  07-26-2022 14:09   f3.txt
   2858  07-25-2022 22:09   passwd
-----
 3764664                      4 files

#从 txt.zip 中删除 f3.txt
```

```
[root@ubuntu2204 0510]# zip -d txt.zip f3.txt
deleting: f3.txt
```

```
[root@ubuntu2204 0510]# unzip -l txt.zip
```

```
Archive:  txt.zip
```

| Length | Date | Time | Name |
|---------|------------|-------|---------|
| 7 | 07-26-2022 | 14:24 | f1.txt |
| 1946847 | 07-26-2022 | 14:09 | f2.txt |
| 2858 | 07-25-2022 | 22:09 | passwd |
| 1949712 | | | 3 files |

范例：注释

```
#添加注释信息，ctrl+d结束
```

```
[root@ubuntu2204 0510]# zip -z txt.zip
```

```
enter new zip file comment (end with .):
```

```
this is test des
```

```
#查看注释，unzip -l 也可查看
```

```
[root@ubuntu2204 0510]# unzip -z txt.zip
```

```
Archive:  txt.zip
```

```
this is test des
```

范例：解压缩

```
#指定解压目录，不解压指定文件
```

```
[root@ubuntu2204 0510]# unzip txt.zip -x f3.txt -d ./txt
```

```
Archive:  txt.zip
```

```
  inflating: ./txt/f1.txt
```

```
  inflating: ./txt/f2.txt
```

```
[root@ubuntu2204 0510]# ll txt
```

```
total 1912
```

```
-rw-r--r-- 1 root root 4453 Jul 26 15:19 f1.txt
```

```
-rw----- 1 root root 1946847 Jul 26 14:09 f2.txt
```

```
#查看
```

```
[root@ubuntu2204 0510]# unzip -v txt.zip
```

```
Archive:  txt.zip
```

| Length | Method | Size | Cmpr | Date | Time | CRC-32 | Name |
|---------|--------|--------|------|------------|-------|----------|---------|
| 4453 | Defl:N | 1661 | 63% | 07-26-2022 | 15:19 | a30efa6e | f1.txt |
| 1946847 | Defl:N | 269213 | 86% | 07-26-2022 | 14:09 | add171e4 | f2.txt |
| 1814952 | Defl:N | 254931 | 86% | 07-26-2022 | 14:09 | eb2f160b | f3.txt |
| 3766252 | | 525805 | 86% | | | | 3 files |

1.2.4 z包

--- 扩展资料 ---

基础知识

简介

此工具来自于ncompress包，此工具目前已经很少使用

对应的文件是 .Z 后缀

命令解读

```
compress [OPTION]... [FILE]...
uncompress [OPTION]... [FILE]...
```

#常用选项

| | |
|----|--------------------|
| -d | #解压缩，相当于uncompress |
| -c | #结果输出至标准输出,不删除原文件 |
| -f | #覆盖已存在目标文件 |
| -v | #显示过程 |
| -r | #递归压缩目录里面所有文件 |

简单实践

简单实践

```
root@ubuntu24:~# ls
fstab
```

默认选项压缩

```
root@ubuntu24:~# compress fstab
root@ubuntu24:~# ls
fstab.Z
```

解压缩

```
root@ubuntu24:~# uncompress fstab.Z
root@ubuntu24:~# ls
fstab
```

显示过程

```
root@ubuntu24:~# compress -v fstab
fstab:  -- replaced with fstab.Z Compression: 33.88%
```

解压缩

```
root@ubuntu24:~# compress -dv fstab.Z
fstab.Z:  -- replaced with fstab
```

保留源文件

```
root@ubuntu24:~# compress -c fstab > fstab.Z
```

```
root@ubuntu24:~# ls
fstab  fstab.Z
```

递归压缩目录

```
root@ubuntu24:~# tree dir1/
dir1/
├── dir2
│   └── messages
├── dnf.log
├── fstab
└── passwd

1 directory, 4 files
root@ubuntu24:~# compress -vr dir1/
dir1//dir2/messages: -- replaced with dir1//dir2/messages.Z Compression: 71.74%
dir1//fstab: -- replaced with dir1//fstab.Z Compression: 33.88%
dir1//dnf.log: -- replaced with dir1//dnf.log.Z Compression: 80.21%
dir1//passwd: -- replaced with dir1//passwd.Z Compression: 45.24%

root@ubuntu24:~# tree dir1/
dir1/
├── dir2
│   └── messages.Z
├── dnf.log.Z
├── fstab.Z
└── passwd.Z

1 directory, 4 files

#递归解压缩目录
root@ubuntu24:~# compress -drv dir1/
dir1//dir2/messages.Z: -- replaced with dir1//dir2/messages
dir1//fstab.Z: -- replaced with dir1//fstab
dir1//dnf.log.Z: -- replaced with dir1//dnf.log
dir1//passwd.Z: -- replaced with dir1//passwd
```

1.2.5 bz包

--- 扩展资料 ---

基础知识

简介

来自于 bzip2 包

对应的文件是 .bz2 后缀

命令解读

```
bzip2 [OPTION]... FILE ...
bunzip2 [OPTION]... FILE ...
```

#常用选项

-d|--decompress

#解压缩, 相当于bunzip2

| | |
|---------------|------------------------|
| -z --compress | #强制压缩 |
| -k --keep | #保留原文件 |
| -f --force | #覆盖已存在目标文件 |
| -t --test | #测试, 检测压缩文件是否完整 |
| -c --stdout | #将压缩数据输出到标准输出中, 并保留原文件 |
| -q --quiet | #安静模式, 忽略警告 |
| -v --verbose | #显示过程 |
| -N | #指定压缩等级, 取值为1-9之间, 默认9 |
| --fast | #同 -1 |
| --best | #同 -9 |

简单实践

范例:

```
#保留原文件, 并显示过程
[root@ubuntu2204 0510]# bzip2 -kv fstab passwd
fstab: 1.731:1, 4.622 bits/byte, 42.22% saved, 720 in, 416 out.
passwd: 2.516:1, 3.180 bits/byte, 60.25% saved, 2858 in, 1136 out.

#输出重定向
[root@ubuntu2204 0510]# bzip2 fstab -cv > fstab.bz2
fstab: 1.731:1, 4.622 bits/byte, 42.22% saved, 720 in, 416 out.

#管道+输出重定向
[root@ubuntu2204 0510]# cat fstab | bzip2 -cv >fstab.bz2
(stdin): 1.731:1, 4.622 bits/byte, 42.22% saved, 720 in, 416 out.

#解压缩
[root@ubuntu2204 0510]# bunzip2 -kfv fstab.bz2
fstab.bz2: done

#不解压查看文件内容
[root@ubuntu2204 0510]# bzcat fstab.bz2
```

1.2.6 xz包

--- 扩展资料 ---

基础知识

来自于 xz 包

对应的文件是 .xz 后缀

命令解读

```
xz [OPTION]... FILE ...
unxz [OPTION]... FILE ...
```

| | |
|------------------|------------------------|
| #常用选项 | |
| -z --compress | #强制压缩 |
| -d --decompress | #解压缩, 相当于unxz |
| -t --test | #测试, 检测压缩文件是否完整 |
| -l --list | #查看压缩文件相关信息 |
| -k --keep | #保留原文件 |
| -f --force | #覆盖已存在目标文件 |
| -c --stdout | #将压缩数据输出到标准输出中, 并保留原文件 |
| -T --threads=NUM | #开多线程, 默认1 |
| -q --quiet | #安静模式, 忽略警告 |
| -v --verbose | #显示过程 |
| -N | #指定压缩等级, 取值为1-9之间, 默认6 |

简单实践

范例:

```
#保留原文件
[root@ubuntu2204 0510]# xz -kv messages
messages (1/1)
 100 %          29.5 KiB / 393.1 KiB = 0.075

#查看
[root@ubuntu2204 0510]# xz -l messages.xz
Strms  Blocks  Compressed Uncompressed  Ratio  Check  Filename
   1      1      29.5 KiB   393.1 KiB  0.075  CRC64  messages.xz

#重定向
[root@ubuntu2204 0510]# xz -kcv messages > msg.xz
messages (1/1)
 100 %          29.5 KiB / 393.1 KiB = 0.075

#解压缩
[root@ubuntu2204 0510]# unxz -vfk fstab.xz msg.xz
fstab.xz (1/2)
 100 %          444 B / 720 B = 0.617

msg.xz (2/2)
 100 %          29.5 KiB / 393.1 KiB = 0.075
```

1.2.7 其他信息

--- 扩展资料 ---

zcat实践

命令简介

zcat 来源于 "zip cat" 的缩写，见字知义

其功能是在不解压的情况下查看压缩文件内容

命令解读

```
zcat [OPTION]... [FILE]...
```

#常用选项

| | |
|----|------------------|
| -c | #将内容输出到标准输出，默认 |
| -d | #解压缩 |
| -l | #显示压缩文件(包)内的文件列表 |
| -r | #在目录上递归操作 |
| -t | #测试压缩文件完整性 |

简单实践

```
[root@ubuntu24 0510]# zcat fstab.Z
[root@ubuntu24 0510]# zcat fstab.gz
[root@ubuntu24 0510]# zcat fstab.zip
```

压缩率实践

简单实践

多种压缩实践

```
[root@ubuntu24 0510]# compress hwdb.bin -vc > hwdb.bin.Z
[root@ubuntu24 0510]# gzip -kv hwdb.bin
[root@ubuntu24 0510]# bzip2 -kv hwdb.bin
[root@ubuntu24 0510]# xz -kv hwdb.bin
[root@ubuntu24 0510]# zip -v hwdb.zip hwdb.bin
```

查看压缩比例效果

```
[root@ubuntu24 0510]# ll hwdb.* -h -S
-r--r--r-- 1 root root 11M Jul 26 20:48 hwdb.bin
-rw-r--r-- 1 root root 2.8M Jul 26 21:40 hwdb.bin.Z
-rw-r--r-- 1 root root 2.0M Jul 26 21:41 hwdb.zip
-r--r--r-- 1 root root 2.0M Jul 26 20:48 hwdb.bin.gz
-r--r--r-- 1 root root 1.7M Jul 26 20:48 hwdb.bin.bz2
-r--r--r-- 1 root root 1.4M Jul 26 20:48 hwdb.bin.xz
```

1.2.9 split命令

基础知识

简介

split 命令可以分割一个文件为多个文件

命令解读

```
split [OPTION]... [FILE [PREFIX]]
```

常用选项

| | |
|-----------------------------------|---------------------------------|
| <code>-b --bytes=SIZE</code> | #按大小指定分割单位 |
| <code>-C --line-bytes=SIZE</code> | #同 -b ，但是在切割时将尽量维持每行的完整性 |
| <code>-d</code> | #切割后小文件的后缀用数字表示 |
| <code>-l --lines=NUMBER</code> | #指定行数，按多少行切一个小文件 |
| <code>--verbose</code> | #显示过程 |

简单实践

切割文件

将test.txt 以6行为单位进行切割成以 x 为前缀名称的小文件

```
[root@ubuntu24 0510]# split -l 6 test.txt
```

```
[root@ubuntu24 0510]# split -6 test.txt
```

以1M大小为单位切割，小文件以数字为后缀，etc.part 开头

```
[root@ubuntu24 0510]# split -b 1M etc.tar.gz -d etc.part
```

显示过程

```
[root@ubuntu24 0510]# split --verbose -b 1M etc.tar.gz -d etc.part
```

合并回去

```
[root@ubuntu24 0510]# cat etc.part* > etc.tar.gz
```