文件查找命令

1.which-whereis-locate-grep-find 命令使用

which　　查看命令的可执行文件位置

whereis　　查看命令的可执行文件位置及相关文件

locate　　配合数据库缓存，快速查看文件位置

grep　　过滤匹配，它是一个文件搜索工具

find　　查找相关文件

```
[root@exercise1 ~]# which cd
/usr/bin/cd
[root@exercise1 ~]# whereis cd
cd: /usr/bin/cd /usr/share/man/man1/cd.1.gz
/usr/share/man/mann/cd.n.gz
[root@exercise1 ~]# whereis ls
ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz
[root@exercise1 ~]#
```

2.locate

语法： locate 文件/目录名

locate 命令和find -name 功能差不多，是它的另外一种写法，但是这个要比 find 搜索快的多，因为 find 命令查找的是具体目录文件，而 locate 它搜索的是一个数据库/var/lib/mlocate/mlocate.db ，这个数据库中存有本地所有的文件信息；这个数据库是 Linux 自动创建并每天自动更新维护。相关的配置信息在/etc/updatedb.conf， 查看定时任务信息在/etc/cron.daily/mlocate

优点：基于数据库查询，效率非常高

缺点：查询时要确保数据库是最新的，否则查找可能不正确

```
[root@exercise1 ~]# locate
-bash: locate: 未找到命令
[root@exercise1 ~]# yum search locate
[root@exercise1 ~]# yum -y install mlocate.x86_64
[root@exercise1 ~]# locate /opt/   #没有更新数据库文件
locate: 无法执行 stat () `/var/lib/mlocate/mlocate.db': 没有那个文件或目录
[root@exercise1 ~]# updatedb   #如果对当天文件查找，需要手动更新数据库 updatedb
[root@exercise1 ~]# touch /opt/test.txt
[root@exercise1 ~]# locate /opt/test.txt   #发现找不到
[root@exercise1 ~]# updatedb
[root@exercise1 ~]# locate /opt/test.txt   #更新过后可以查询到
/opt/test.txt
[root@exercise1 ~]#
```

解决缺点方法：

```
[root@exercise1 ~]# alias locate='updatedb && locate'   #临时别名
[root@exercise1 ~]# vim ~/.bashrc   #当前用户生效
#插入这条别名命令
alias locate='updatedb && locate'
[root@exercise1 ~]# source ~/.bashrc   #刷新生效
[root@exercise1 ~]# vim /etc/bashrc   #全局使用
#插入这条别名命令
alias locate='updatedb && locate'
```

3.grep 查找使用(脚本三剑客之一)

作用：过滤,它能够使用正则表达式来搜索文本，并把结果打印出来

```
[root@home opt]# which grep
alias grep='grep --color=auto'
    /usr/bin/grep
如何临时修改关键字颜色
#a,b的意义分别表示加颜色的方式和颜色值
export GREP_COLOR='a;b'

a可以选择:【0,1,4,5,7,8】 0关闭所有属性; 1设置高亮度; 4下划线; 5闪烁; 7反显; 8消隐; 也不知道为什么没有2,3,6,9
b可以选择:【30-37或40-47】
30 black
31 red
32 green
33 yellow
34 blue
35 purple
36 cyan
37 white
30-37 设置前景色
40-47 设置背景色
```

参数:

```
    -v    取反

    -A    后接行数n：可以显示关键字后几行的内容

    -i    忽略大小写

    ^#     以#开头

    #$     以#结尾

    ^$     空行

    -n 对过滤的内容加上行号

    -o 只显示被匹配到的内容
```

```
    |    或者的意思

    -w    精确匹配

    -r       递归,一般跟-l参数连用

    -l       查看某一目录下含有文件内容

    -E        支持正则表达式
```

例子

```
例1:
[root@exercise1 opt]# vim a.txt
#插入以下内容
aaa
bbb
ccc
ddd
abcd
[root@exercise1 opt]# cat a.txt
aaa
bbb
ccc
ddd
abcd
[root@exercise1 opt]# grep a a.txt
aaa
abcd
[root@exercise1 opt]#
```

例2:
[root@exercise1 opt]# grep -v b a.txt
aaa
ccc
ddd
[root@exercise1 opt]#

例3：

[root@exercise1 opt]# grep ^# /etc/ssh/sshd_config　#以“#”开头

$OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $

This is the sshd server system-wide configuration file. See

例4：

[root@exercise1 opt]# echo Abcd >> /opt/a.txt

[root@exercise1 opt]# grep -i a a.txt

aaa

abcd

Abcd

[root@exercise1 opt]#

例5：

[root@exercise1 opt]# grep bash$ /etc/passwd　#以 bash 结尾

root❌0:0:root:/root:/bin/bash

lin05❌1112:1112::/home/lin05:/bin/bash

[root@exercise1 opt]#

例6：

[root@exercise1 opt]# grep -n a a.txt

1:aaa

5:abcd

[root@exercise1 opt]#

例7：

```
[root@exercise1 opt]# grep "nologin\|root" /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
……
```

```
#注： \ 表示转义符

[root@exercise1 opt]# egrep "nologin|root" /etc/passwd
  #查看包括 nologin 或 root 的行
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
……

#注：egrep 是 grep 加强版本

[root@exercise1 opt]# grep -E "nologin|root"
/etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
……
```

例8：

[root@exercise1 opt]# grep -w "3306|80" /etc/services

http 80/tcp www www-http # WorldWideWeb HTTP

http 80/udp www www-http # HyperText Transfer Protocol

http 80/sctp # HyperText Transfer Protocol

mysql 3306/tcp # MySQL

mysql 3306/udp # MySQL

例9：

[root@exercise1 opt]# grep -rl "root" /etc/

/etc/grub.d/00_header

/etc/grub.d/01_users

……

例10：只将/etc/ssh/sshd_config文件的配置项列出来并显示行号

[root@exercise1 opt]# grep -v "^#|^$" /etc/ssh/sshd_config

HostKey /etc/ssh/ssh_host_rsa_key

HostKey /etc/ssh/ssh_host_ecdsa_key

HostKey /etc/ssh/ssh_host_ed25519_key

SyslogFacility AUTHPRIV

AuthorizedKeysFile .ssh/authorized_keys

PasswordAuthentication yes

ChallengeResponseAuthentication no

GSSAPIAuthentication yes

GSSAPICleanupCredentials no

UsePAM yes

X11Forwarding yes

AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES

AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT

AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE

AcceptEnv XMODIFIERS

Subsystem sftp /usr/libexec/openssh/sftp-server

[root@exercise1 opt]#

或者

[root@exercise1 opt]# cat /etc/ssh/sshd_config | grep -v ^# | grep -v ^$

HostKey /etc/ssh/ssh_host_rsa_key

HostKey /etc/ssh/ssh_host_ecdsa_key

……

4.find 命令使用 （必会，参数比较多）

格式：find pathname -options [-print] [action]

命令字 路径名称 选项 输出内容 动作

参数:

pathname: find 命令所查找的目录路径，不输入代表当前目录例如用 . 来表示当前目录，用 / 来表示系统 根目录。

---

find 命令选项:

```
-name        按照文件名查找文件。        “名称”

-perm        按照文件权限来查找文件。666 777 等

-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

-atime       按照文件的访问时间来查找文件

-ctime       按照文件的属性更改时间来查找文件

-type        查找某一类型的文件

f - 普通文件

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

s - 套接字文件
```

```
-maxdepth       查找的目录深度

-size n    查找符合指定的文件大小的文件    +大于   -小于

-exec      对匹配的文件执行该参数所给出的其他 linux 命令，相应命令的形式为' 命令 {} \;，注意{ }和 \；
之间的空格，{}代表查到的内容
```

例子

```
例1：查看当前目录下所有的 TXT 格式的文件
[root@exercise1 opt]# find . -name "*.txt"
./a.txt
[root@exercise1 opt]#
```

```
例2：按权限查找
[root@exercise1 opt]# find /bin/ -perm 755   #等于 0755 权限的文件或目录
[root@exercise1 opt]# find /bin/ -perm -644   #-perm -644    至少有 644 权限的文件或目录
```

#查看系统中权限至少为 777 的文件或目录
#创建一些测试文件：
[root@exercise1 opt]# mkdir ccc
[root@exercise1 opt]# chmod 777 ccc
[root@exercise1 opt]# mkdir test
[root@exercise1 opt]# chmod 1777 test
[root@exercise1 opt]# touch b.sh
[root@exercise1 opt]# chmod 4777 b.sh
#查找
[root@exercise1 opt]# find /opt/ -perm 777
/opt/ccc
[root@exercise1 opt]# find /opt/ -perm 1777
/opt/test

[root@exercise1 opt]# find /opt/ -perm 4777
/opt/b.sh
[root@exercise1 opt]#

```
例3：按文件类型查找
把系统中权限不低于 777 的危险目录查找出来
[root@exercise1 opt]# find / -type d -perm -777

把系统种权限不低于777的危险文件查找出来
[root@exercise1 opt]# find / -type f -perm -777
find: ‘/proc/1277/task/1277/fdinfo/6’: 没有那个文件或目录
find: ‘/proc/1277/fdinfo/6’: 没有那个文件或目录
/opt/b.sh
```

```
例4：查找系统中所有者是root的所有文件
[root@exercise1 opt]# find / -user root

例5：查找系统中所有者不是root的所有文件
[root@exercise1 opt]# find / -not -user root
[root@exercise1 opt]# find / ! -user root
```

```
例6：按照更改时间或访问时间等查找文件
如果希望按照更改时间来查找文件，可以使用 mtime,atime 或 ctime 选项
```

mtime: 文件最后一次修改的时间

atime: 最后一次访问时间

ctime: 文件属性的最后一次变化时间

```
希望在 root 目录下查找更改时间在 1 天以内，被黑客修改的文件
[root@exercise1 opt]# find /root/ -mtime -1
/root/
/root/.bash_history
/root/.viminfo
```

```
例7：查找的目录深度：

-maxdepth 1   #只查找目录第一层的文件和目录
如：查找/bin 目录下 第一层的文件
[root@exercise1 opt]# find /bin/ -maxdepth 1   #/bin 后面要有/
[root@exercise1 opt]# find /bin -maxdepth 1   #这个命令无法满足我们的需求
/bin
[root@exercise1 opt]# find /sbin/ -maxdepth 2   #/bin 与/sbin 是特别的
[root@exercise1 opt]# find /opt -maxdepth 1
/opt
/opt/a.txt
/opt/ccc
/opt/test
/opt/b.sh
/opt/bak
/opt/pkcs11.txt
[root@exercise1 opt]#
```

例8：查找多个类型文件 比较符的使用：

-a and 并且

-o or 或者

\+ 超过

\- 低于

[root@exercise1 opt]# touch a.pdf back.sh

[root@exercise1 opt]# find . -name "*.sh" -o -name* ".pdf"

./b.sh

./a.pdf

./back.sh

[root@exercise1 opt]# find /etc/ -size +20k -a -size -50k #找到大于20k小于50k的文件

/etc/etc/sysconfig/network-scripts/network-functions-ipv6

/etc/etc/selinux/targeted/active/modules/100/apache/hll

……

[root@exercise1 opt]# find /etc/ -size +20k

```
例9：多条件合并查找，使用(  ) 分组，  \(   前后空格，带转义符号 \)
例：找到当前目录下，大于10k 且小于20k，并且文件名为txt结尾或 tar 结尾的文件

[root@exercise1 opt]# find / -size +10k -a -size -20k -a \( -name "*.txt" -o -name "*.tar" \)
```

```
例10：过滤用法
[root@exercise1 opt]# cd /opt && touch a{1..10}.txt
[root@home opt]# find -type f  |grep -v a1.txt
#找出文件，但不要a1.txt文件
[root@exercise1 opt]# find ./ -type -f |xargs grep "123"
#找出文件中内容包含123的文件
```

动作处理

查找到一个文件后，需要对文件进行如何处理，find的默认动作是-print

动作	含义
-print	打印查找到的内容（默认）
-ls	以长格式显示的方式打印查找到的内容
-delete	删除查找到的文件与目录
-ok	后面跟自定义shell命令（会提示是否操作）
-exec	后面跟自定义shell命令

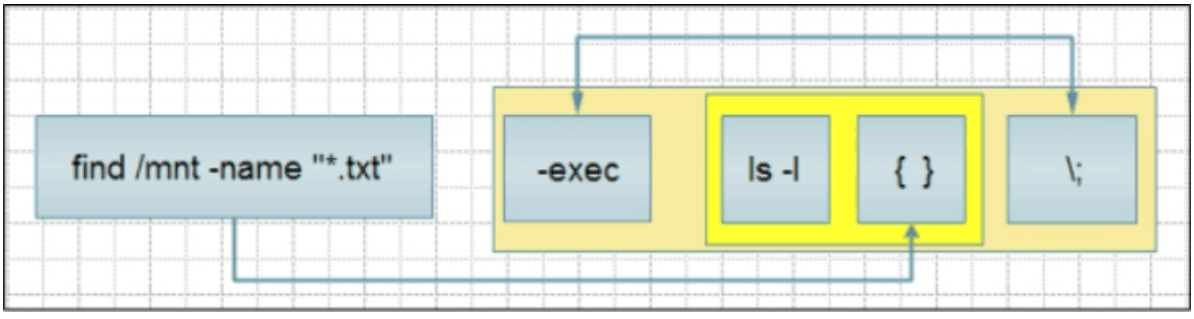
```
例11：使用-print打印查找一的文件
[root@exrcise1 ~]# find /etc -name "ifcfg"
[root@exrcise1 ~]# find /etc -name "ifcfg" -print


例12：使用-ls打印查找到的文件，以长格式显示
[root@exrcise1 ~]# find /etc -name "ifcfg" -ls


例13：使用-delete删除文件与目录
[root@exrcise1 ~]# find ./ -name "a*" -delete

例14：批量删除空目录
[root@exrcise1 ~]# find . -type d -empty -delete
```

```
例15：对查找内容执行相应命令
-exec       这个选项参数后面可以跟自定义的 SHELL 命令，格式如下：
```



```
1）、把查找到的文件列出来
[root@exercise1 opt]# touch {1..3}.bak
[root@exercise1 opt]# find . -name "*.bak" -exec ls -l {} \;   #exec参数不能使用别名
-rw-r--r--. 1 root root 0 1月  20 20:20 ./1.bak
-rw-r--r--. 1 root root 0 1月  20 20:20 ./2.bak
-rw-r--r--. 1 root root 0 1月  20 20:20 ./3.bak
```

```
2）、把查找到的文件移动到一个指定的目录
[root@exercise1 opt]# find . -name "*.bak" -exec mv {} /opt/bak/ \;
mv: "./bak/1.bak" 与"/opt/bak/1.bak" 为同一文件
mv: "./bak/2.bak" 与"/opt/bak/2.bak" 为同一文件
mv: "./bak/3.bak" 与"/opt/bak/3.bak" 为同一文件
[root@exercise1 opt]# ls /opt/bak/
1.bak  2.bak  3.bak
```

```
3）、把查找到的文件复制到一个指定的目录
[root@exercise1 opt]# find /root/ -name "*.txt" -exec cp {} /opt/ \;
```

```
4）、
[root@home opt]# ll
总用量 0
-rw-r--r-- 1 root root 0 7月  20 17:03 a10.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a4.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a5.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a6.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a7.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a8.txt
-rw-r--r-- 1 root root 0 7月  20 17:03 a9.txt
```

```
[root@home opt]# find . -type f -exec tar -cvf test.tar {} \;
./a4.txt
./a5.txt
./a6.txt
./a7.txt
./a8.txt
./a9.txt
./a10.txt
[root@home opt]# tar -tvf test.tar
#因为-exec是一次次执行命令
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a10.txt
[root@home opt]# find . -type f -exec tar -rvf test.tar {} \;       #使用递归追加打包才行
./a4.txt
./a5.txt
./a6.txt
./a7.txt
./a8.txt
./a9.txt
./a10.txt
[root@home opt]# tar -tvf test.tar
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a10.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a4.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a5.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a6.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a7.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a8.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a9.txt
-rw-r--r-- root/root         0 2022-07-20 17:03 ./a10.txt
```

```
例16：使用-ok实现文件拷贝，但会提示删除
[root@exrcise1 ~]# find /etc -name "ifcfg" -ok cp  {} /tmp \;
```

5、文件删除的几种方式

1.rm -rf *

2.使用 find 命令，再进行删除操作。
find /test -mtime +7 -exec rm {} \;
除了在 find 中借助 -exec 参数调用 rm 命令外，还有一个更好的选择，那就是使用 -delete 选项。
比如：
find /test -size +7M -delete
达到的效果与上一条命令一样。

1、删除巨量文件时用什么命令最快?

话不多说，我们直接上测试。
首先借助一个简单的 bash for 循环创建 50 万个文件。
[root@master test2]# mkdir /opt/test2 && cd /opt/test2
[root@master test2]# for i in $(seq 1500000); do echo testing >> i.txt; done
在创建了 50 万个文件后，我们将尝试使用多方式来删除它们，看看哪种方式删除巨量文件速度最快。

Round 1：rm 命令

首先让我们使用简单的 rm 命令，同时我们使用 time 命令来计时。

```
[root@master test2]# time rm -rf *
-bash: /usr/bin/rm:Argument list too long
real 0m0.929s
user    0m0.902s
sys  0m0.022s
```

rm 命令的执行结果是 Argument list too long ，这意味着该命令没有完成删除，因为给 rm 命令的文件数量太大而无法完成，所以它直接就躺平罢工了。

不要注意 time 命令显示的时间，因为 rm 命令没有完成它的操作，time 命令只管显示你命令执行了多长时间，而不关心命令的最终结果。

Round 2：使用 -exec 参数的 find 命令

```
[root@master test2]# time find ./ -type f -exec rm {} \;
real 14m53.473s
user    0m35.688s
sys  4m17.976s
```

从我们使用

time 命令得到的输出可以看出，从单个目录中删除 50 万个文件需要 14 分 53 秒。

这是相当长的时间，因为对于每个文件，都会执行一个单独的 rm 命令，直到删除所有文件。

Round 3：使用 -delete 参数的 find 命令

```
[root@master test2]# time find ./ -type f -delete;
real 0m14.182s
user    0m0.161s
sys  0m12.136s
```

删除速度大大提高，只用了 14 秒！当你在 Linux 中删除数百万个文件时，这是速度的惊人改进。

Round 4： rsync 命令

我们可以通过将具有大量文件的目标目录与空目录进行同步来实现删除的操作。
在我们的例子中， /opt/test2 目录（目标目录）有 50 万个文件，我们再创建一个名为 /opt/test1 的空目录（源目录）。现在，我们将在
rsync 命令中使用
-delete 选项，这将删除目标目录中的所有源目录中不存在文件。

[root@master test2]# time rsync --delete-before -aH -progress /opt/test/ /opt/test2/
real 0m0.075s
user 0m0.001s
sys 0m0.004s
可以看到，仅用 1 秒不到就完成删除。
因此与 find 命令相比，如果您想清空包含数百万个文件的目录，使用
rsync 命令会更好。

2、小结

下表总结了 Linux 中采用不同方式删除 50 万个文件的速度，方便大家参考。

命令	**花费时间**
rm	命令无法删除大量文件
使用 -exec 参数的 find 命令	14 分 53 秒
使用 -delete 参数的 find 命令	14 秒
rsync 命令	不到1秒