中国矿业大学计算机学院

2019-2020(2)本科生 Linux 操作系统课程报告

内容范围 Linux 命令

指 标 点 1.2 占 比 20%

学生姓名 袁孝健 学 号 06172151

专业班级 信息安全 2017-01 班

任课教师 杨东平

课程基础理论掌握程度	熟练 □	较熟练 □	一般 □	不熟练 □
综合知识应用能力	强 □	较强 □	一般 □	差 □
报告内容	完整 □	较完整 □	一般 □	不完整 □
报告格式	规范 □	较规范 □	一般 □	不规范 □
实验完成状况	好 □	较好 □	一般 □	差 □
工作量	饱满 □	适中 □	一般 □	欠缺 □
学习、工作态度	好 □	较好 □	一般 □	差 □
抄袭现象	无 □	有 □ 姓名:		
存在问题				
总体评价				

综合成绩: 任课教师签字:

年 月

摘　要

Linux 是一个开源、稳定、高效的多用户网络操作系统，熟练的使用它对于计算机专业学生来说也越来越普遍。而要想熟练的使用 Linux 操作系统，对其命令的掌握程度至关重要，不仅要知道常用命令的作用，也要对一些命令参数的组合与使用非常了解。因此，本文介绍了 grep、ls、find、tar、sort 五种常用的 Linux 命令，并对它们的参数进行了整理与总结。最后，对各常用参数的组合以及其实现的功能做进一步的实践、展示与阐述。

关键词：Linux 操作系统; 命令操作; 信息技术

Abstract

Linux is an open source, stable, and efficient multi-user network operating system. Skillful use of it is becoming more and more common for computer students. To be proficient in using the Linux operating system, it is important to master the command. Not only do you need to know the role of common commands, but you also need to know the combination and use of some command parameters. Therefore, this article introduces five commonly used Linux commands, including grep, ls, find, tar, and sort, and summarizes and summarizes their parameters. Finally, the combination of various common parameters and the functions realized by it are further practiced, demonstrated and explained.

Keywords: Linux system; command operation; information technology

目　录

1 Linux 命令

要求：列举 5 条以上 Linux 命令，对每条命令必须阐明该命令的作用、参数、使用方法并将命令运行结果截图附上，最后阐明体会与收获。

根据所列命令的难度与阐述的详细和完整程度给出分值，若不够 5 条，扣相应分值，每少列一条扣 4 分。若无命令运行截图，扣 2 分。

说明：如果撰写规范不符合《计算机学院考查类课程报告撰写规范》要求的，整体上酌情扣除 1-10 分。

1.1 grep 命令

1.1.1 作用及基本语法

（1）grep 命令是 Linux 中一个全局文本搜索工具，可以使用正则表达式搜索文本，并把匹配的行打印出来。

（2）grep 命令基本语法：grep [option] pattern file

1.1.2 全部参数的总结

参数	作用
-a	不要忽略二进制数据
-A<显示列数>	除了显示符合范本样式的那一行之外，并显示该行之后的内容
-b	在显示符合范本样式的那一行之外，并显示该行之前的内容。
-c =<显示行数>	计算符合范本样式的列数。
-C<显示列数>	除了显示符合范本样式的那一列之外，并显示该列之前后的内容。
-d<进行动作>	当指定要查找的是目录而非文件时，必须使用这项参数，否则 grep 命令将回报信息并停止动作。
-e<范本样式>	指定字符串作为查找文件内容的范本样式。
-F	将范本样式视为固定字符串的列表。
-f<范本文件>	指定范本文件，让 grep 查找符合范本条件的文件内容。
-G	将范本样式视为普通的表示法来使用。
-h	在显示符合范本样式的那一列之前，不标示该列所属的文件名称。

-H	在显示符合范本样式的那一列之前，标示该列的文件名称。
-i	忽略字符大小写的差别。
-l	列出文件内容符合指定的范本样式的文件名称。
-L	列出文件内容不符合指定的范本样式的文件名称。
-n	在显示符合范本样式的那一列之前，标示出该列的编号。
-q	不显示任何信息。
-r	此参数的效果和指定“-d recurse”参数相同。
-s	不显示错误信息。
-v	反转查找。
-w	只显示全字符合的列。
-x	只显示全列符合的列。
-o	只输出文件中匹配到的部分。

1.1.3 常用参数的使用

（1）在文件或文本中搜索一个单词:

```
ubuntu@VM-0-14-ubuntu:~$ grep CUMT test.txt
I am YuanXiaojian and i am from CUMT.
```

（2）搜索多个文件并查找匹配文本在哪些文件中（-l）:

```
ubuntu@VM-0-14-ubuntu:~$ grep CUMT -l test.txt test2.txt test3.txt
test.txt
test2.txt
```

（3）grep 静默输出（-q），不会输出任何信息，如果命令运行成功返回 0，失败则返回非 0 值，一般用于条件测试:

```
ubuntu@VM-0-14-ubuntu:~$ grep CUMT -q test.txt
ubuntu@VM-0-14-ubuntu:~$
```

（4）输出除匹配对象之外的所有行（-v）:

```
ubuntu@VM-0-14-ubuntu:~$ cat test2.txt
aaa
CUMT
bbb
ccc
ubuntu@VM-0-14-ubuntu:~$ grep -v CUMT test2.txt
aaa
bbb
ccc
```

（5）标记匹配颜色（–color=auto）:

```
ubuntu@VM-0-14-ubuntu:~$ grep --color=auto CUMT test2.txt
CUMT
```

（6）使用正则表达式（-E），只输匹配到的部分（-o）：

```
ubuntu@VM-0-14-ubuntu:~$ echo abc123qwe | grep -E -o "[0-9]+"
123
```

（7）统计文本中包含匹配字符串的行数（-c）：

```
ubuntu@VM-0-14-ubuntu: $ cat test.txt
I am YuanXiaojian and i am from CUMT.
abc
CUMT
123
CUMT
ubuntu@VM-0-14-ubuntu:~$ grep -c CUMT test.txt
3
```

（8）输出包含匹配字符串的行数（-n）：

```
ubuntu@VM-0-14-ubuntu:~$ grep -n CUMT test.txt
1:I am YuanXiaojian and i am from CUMT.
3:CUMT
5:CUMT
```

（9）忽略大小写进行匹配（-i）：

```
ubuntu@VM-0-14-ubuntu:~$ cat test.txt
I am YuanXiaojian and i am from CUMT.
abc
cumt
123
ubuntu@VM-0-14-ubuntu:~$ grep -i cumt test.txt
I am YuanXiaojian and i am from CUMT.
cumt
```

（10）匹配多个样式（-e）：

```
ubuntu@VM-0-14-ubuntu: $ cat test.txt
I am YuanXiaojian and i am from CUMT.
abc
cumt
123
ubuntu@VM-0-14-ubuntu:~$ grep -e CUMT -e 123 test.txt
I am YuanXiaojian and i am from CUMT.
123
```

（11）打印出匹配文本之前或者之后的行（-A/-B/-C）：

```
ubuntu@VM-0-14-ubuntu:~$ cat test.txt
I am YuanXiaojian and i am from CUMT.
abc
cumt
123
ubuntu@VM-0-14-ubuntu:~$ grep -A1 cumt test.txt
cumt
123
ubuntu@VM-0-14-ubuntu:~$ grep -B1 cumt test.txt
abc
cumt
ubuntu@VM-0-14-ubuntu:~$ grep -C1 cumt test.txt
abc
cumt
123
```

（12）打印样式匹配所位于的字符或字节偏移（-b -o）：

```
ubuntu@VM-0-14-ubuntu:~$ echo 123abcCUMTdef | grep -b -o CUMT
6:CUMT
```

1.1.4 体会与收获

grep 命令在 Linux 中还是比较常用的，可以使我们很快的搜索我来想要的文本，并且可以使用正则表达式进行匹配，而正则表达式是十分强大的。如果可以很好的掌握正则表达式，那么将大大提高 grep 使用的效率与灵活性。这次实验中，我了解了大部分 grep 命令的参数，并且对一些常用参数或组合进行了实践，深深的感受到了此命令的方便。

1.2 ls 命令

1.2.1 作用及基本语法

（1）ls 命令用于显示指定工作目录下之内容（列出目前工作目录所含之文件及子目录)
（2）ls 命令的基本语法：ls [-alrtAFR] [name…]

1.2.2 全部参数的总结

参数	作用
-a	显示所有档案及目录
-A	显示除影藏文件“.”和“…”以外的所有文件列表
-C	多列显示输出结果（默认选项）
-l	所有输出信息用单列格式输出，不输出为多列
-F	在每个输出项后追加文件的类型标识符
-b	将文件中的不可输出的字符以反斜线加字符编码的方式输出
-c	与“-lt”选项连用时，按照文件状态时间排序输出目录内容，排序的依据是文件的索引节点中的 ctime 字段
-d	仅显示目录名，而不显示目录下的内容列表。
-f	此参数的效果和同时指定“aU”参数相同，并关闭“lst”参数的效果
-i	显示文件索引节点号（inode）。一个索引节点代表一个文件
–file-type	与“-F”选项的功能相同，但是不显示“*”
-k	以 KB（千字节）为单位显示文件大小
-l	以长格式显示目录下的内容列表
-m	用“,”号区隔每个文件和目录的名称
-n	以用户识别码和群组识别码替代其名称
-r	以文件名反序排列并输出目录内容列表
-s	显示文件和目录的大小，以区块为单位
-t	用文件和目录的更改时间排序

-L	如果遇到性质为符号链接的文件或目录，直接列出该链接所指向的原始文件或目录
–full-time	列出完整的日期与时间
-R	递归处理，将指定目录下的所有文件及子目录一并处理
–color[=WHEN]	用不同的颜色高亮显示不同类型的

1.2.3 常用参数的使用

（1）显示当前目录下非影藏文件与目录：

```
ubuntu@VM-0-14-ubuntu:/$ ls
bin   home                            lib64       proc  srv  vmlinuz
boot  imgcreate_linux_install_0.1.23  lost+found  root  sys  vmlinuz.old
data  initrd.img                      media       run   tmp
dev   initrd.img.old                  mnt         sbin  usr
etc   lib                             opt         snap  var
```

（2）显示当前目录下包括影藏文件在内的所有文件列表（-a）：

```
ubuntu@VM-0-14-ubuntu:/$ ls -a
.     dev                             initrd.img.old  mnt   sbin  usr
..    etc                             lib             opt   snap  var
bin   home                            lib64           proc  srv   vmlinuz
boot  imgcreate_linux_install_0.1.23  lost+found      root  sys   vmlinuz.old
data  initrd.img                      media           run   tmp
```

（3）输出长格式列表（-1）：

```
ubuntu@VM-0-14-ubuntu:/$ ls -1
bin
boot
data
dev
etc
home
imgcreate_linux_install_0.1.23
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
```

（4）显示文件的 inode 信息（-i）：

```
ubuntu@VM-0-14-ubuntu:/$ ls -i
 131073 bin                                393230 opt
 262147 boot                                    1 proc
1703937 data                               393232 root
      2 dev                                     2 run
 262145 etc                                393258 sbin
 131195 home                               274415 snap
 277147 imgcreate_linux_install_0.1.23     393342 srv
    838 initrd.img                              1 sys
    840 initrd.img.old                     393344 tmp
 131196 lib                                393345 usr
 393228 lib64                              131787 var
     11 lost+found                            831 vmlinuz
 393217 media                                 839 vmlinuz.old
 393229 mnt
```

（5）水平输出文件列表（-m）：

```
ubuntu@VM-0-14-ubuntu:/$ ls -m
bin, boot, data, dev, etc, home, imgcreate_linux_install_0.1.23, initrd.img,
initrd.img.old, lib, lib64, lost+found, media, mnt, opt, proc, root, run,
sbin, snap, srv, sys, tmp, usr, var, vmlinuz, vmlinuz.old
```

（6）按修改时间显示文件（-t）：

```
ubuntu@VM-0-14-ubuntu:/$ ls -t
run   home  usr   sbin            imgcreate_linux_install_0.1.23  lost+found
tmp   bin   data  initrd.img.old  snap                            mnt
root  dev   sys   vmlinuz.old     lib
etc   var   proc  initrd.img      media
srv   opt   boot  vmlinuz         lib64
```

（7）显示递归文件（-R）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -R test
test:
test2

test/test2:
cumt.txt  yxj.txt
```

（8）打印文件的 UID 和 GID（-n）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -n
total 56
-rw-r--r--  1    0    0  206 Mar  7 11:08 app.py
drwxrwxr-x  3 500 500 4096 Feb 29 16:26 CheckIn
-rw-rw-r--  1 500 500 1753 Feb 29 03:26 CRT.py
drwxrwxr-x  4 500 500 4096 Mar  8 14:41 ctf
drwxr-xr-x 10    0    0 4096 Jan  7 17:21 CTFd
drwxrwxr-x  4 500 500 4096 Feb 29 03:28 CUMTCTF
-rw-rw-r--  1 500 500   10 Feb 19 22:38 flag.php
-rw-rw-r--  1 500 500 4196 Mar 11 22:09 geturl.py
-rwxrwxr-x  1 500 500   33 Mar  2 09:30 hello.sh
drwxrwxr-x  3 500 500 4096 Mar 16 16:24 test
-rw-rw-r--  1 500 500   17 Mar 16 14:41 test2.txt
-rw-rw-r--  1 500 500    7 Mar 16 14:38 test3.txt
-rw-rw-r--  1 500 500   51 Mar 16 14:58 test.txt
```

（9）按照特殊字符对文件进行分类（-F）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -F
app.py     CRT.py  CTFd/     flag.php   hello.sh*  test2.txt  test.txt
CheckIn/   ctf/    CUMTCTF/  geturl.py  test/      test3.txt
```

（10）列出文件并标记颜色分类（–color=auto）：

```
ubuntu@VM-0-14-ubuntu:~$ ls --color=auto
app.py   CRT.py  CTFd     flag.php   hello.sh  test2.txt  test.txt
CheckIn  ctf     CUMTCTF  geturl.py  test      test3.txt
```

（11）列出文件和文件夹的详细信息（-l）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -l
total 56
-rw-r--r--  1 root   root    206 Mar  7 11:08 app.py
drwxrwxr-x  3 ubuntu ubuntu 4096 Feb 29 16:26 CheckIn
-rw-rw-r--  1 ubuntu ubuntu 1753 Feb 29 03:26 CRT.py
drwxrwxr-x  4 ubuntu ubuntu 4096 Mar  8 14:41 ctf
drwxr-xr-x 10 root   root   4096 Jan  7 17:21 CTFd
drwxrwxr-x  4 ubuntu ubuntu 4096 Feb 29 03:28 CUMTCTF
-rw-rw-r--  1 ubuntu ubuntu   10 Feb 19 22:38 flag.php
-rw-rw-r--  1 ubuntu ubuntu 4196 Mar 11 22:09 geturl.py
-rwxrwxr-x  1 ubuntu ubuntu   33 Mar  2 09:30 hello.sh
drwxrwxr-x  3 ubuntu ubuntu 4096 Mar 16 16:24 test
-rw-rw-r--  1 ubuntu ubuntu   17 Mar 16 14:41 test2.txt
-rw-rw-r--  1 ubuntu ubuntu    7 Mar 16 14:38 test3.txt
-rw-rw-r--  1 ubuntu ubuntu   51 Mar 16 14:58 test.txt
```

（12）列出可读文件和文件夹详细信息（-lh）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -lh
total 56K
-rw-r--r--  1 root   root    206 Mar  7 11:08 app.py
drwxrwxr-x  3 ubuntu ubuntu 4.0K Feb 29 16:26 CheckIn
-rw-rw-r--  1 ubuntu ubuntu 1.8K Feb 29 03:26 CRT.py
drwxrwxr-x  4 ubuntu ubuntu 4.0K Mar  8 14:41 ctf
drwxr-xr-x 10 root   root   4.0K Jan  7 17:21 CTFd
drwxrwxr-x  4 ubuntu ubuntu 4.0K Feb 29 03:28 CUMTCTF
-rw-rw-r--  1 ubuntu ubuntu   10 Feb 19 22:38 flag.php
-rw-rw-r--  1 ubuntu ubuntu 4.1K Mar 11 22:09 geturl.py
-rwxrwxr-x  1 ubuntu ubuntu   33 Mar  2 09:30 hello.sh
drwxrwxr-x  3 ubuntu ubuntu 4.0K Mar 16 16:24 test
-rw-rw-r--  1 ubuntu ubuntu   17 Mar 16 14:41 test2.txt
-rw-rw-r--  1 ubuntu ubuntu    7 Mar 16 14:38 test3.txt
-rw-rw-r--  1 ubuntu ubuntu   51 Mar 16 14:58 test.txt
```

（13）显示文件夹信息（-ld）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -ld /etc
drwxr-xr-x 98 root root 4096 Mar  7 20:17 /etc
```

（14）按修改时间列出文件和文件夹详细信息（-lt）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -lt
total 56
drwxrwxr-x  3 ubuntu ubuntu 4096 Mar 16 16:24 test
-rw-rw-r--  1 ubuntu ubuntu   51 Mar 16 14:58 test.txt
-rw-rw-r--  1 ubuntu ubuntu   17 Mar 16 14:41 test2.txt
-rw-rw-r--  1 ubuntu ubuntu    7 Mar 16 14:38 test3.txt
-rw-rw-r--  1 ubuntu ubuntu 4196 Mar 11 22:09 geturl.py
drwxrwxr-x  4 ubuntu ubuntu 4096 Mar  8 14:41 ctf
-rw-r--r--  1 root   root    206 Mar  7 11:08 app.py
-rwxrwxr-x  1 ubuntu ubuntu   33 Mar  2 09:30 hello.sh
drwxrwxr-x  3 ubuntu ubuntu 4096 Feb 29 16:26 CheckIn
drwxrwxr-x  4 ubuntu ubuntu 4096 Feb 29 03:28 CUMTCTF
-rw-rw-r--  1 ubuntu ubuntu 1753 Feb 29 03:26 CRT.py
-rw-rw-r--  1 ubuntu ubuntu   10 Feb 19 22:38 flag.php
drwxr-xr-x 10 root   root   4096 Jan  7 17:21 CTFd
```

（15）以文件名反序排列并输出目录内容列表（-r）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -r
test.txt   test2.txt  hello.sh   flag.php  CTFd  CRT.py   app.py
test3.txt  test       geturl.py  CUMTCTF   ctf   CheckIn
```

1.2.4 收获与体会

ls 命令通常是我们刚接触 Linux 操作系统时学习的第一个命令，用来列出当前目录的文件，但是随着更加深入的了解，会发现 ls 命令也是有很多参数可以学习的。Linux 系统是基于文件的，所以对文件及属性的了解是非常重要的，而熟练的掌握 ls 命令就可以帮我们做到这一点。通过不同参数的组合，可以实现不同的功能，查看不同的文件信息，对我们进行文件管理是分产有用的。

1.3 find 命令

1.3.1 作用及基本语法

（1）find 命令用来在指定目录下查找文件。任何位于参数之前的字符串都将被视为欲查找的目录名。如果使用该命令时，不设置任何参数，则 find 命令将在当前目录下查找子目录与文件。并且将查找到的子目录和文件全部进行显示。

（2）find 命令基本语法：find path -option [-print]　[-exec -ok command] {} \;

1.3.2 全部参数的总结

参数	作用
-amin<分钟>	查找在指定时间曾被存取过的文件或目录，单位以分钟计算
-anewer<参考文件或目录>	查找其存取时间较指定文件或目录的存取时间更接近现在的文件或目录
-atime<24 小时数>	查找在指定时间曾被存取过的文件或目录，单位以 24 小时计算
-cmin<分钟>	查找在指定时间之时被更改过的文件或目录
-cnewer<参考文件或目录>	查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录
-ctime<24 小时数>	查找在指定时间之时被更改的文件或目录，单位以 24 小时计算
-daystart	从本日开始计算时间
-depth	从指定目录下最深层的子目录开始查找
-empty	寻找文件大小为 0 Byte 的文件，或目录下没有任何子目录或文件的空目录
-exec<执行指令>	假设 find 指令的回传值为 True，就执行该指令
-false	将 find 指令的回传值皆设为 False
-fls<列表文件>	此参数的效果和指定“-ls”参数类似，但会把结果保存为指定的列表文件

-follow	排除符号连接
-fprint<列表文件>	此参数的效果和指定“-print”参数类似，但会把结果保存成指定的列表文件
-fprint0<列表文件>	此参数的效果和指定“-print0”参数类似，但会把结果保存成指定的列表文件
-fprintf<列表文件><输出格式>	此参数的效果和指定“-printf”参数类似，但会把结果保存成指定的列表文件
-fstype<文件系统类型>	只寻找该文件系统类型下的文件或目录
-gid<群组识别码>	查找符合指定之群组识别码的文件或目录
-group<群组名称>	查找符合指定之群组名称的文件或目录
-ilname<范本样式>	此参数的效果和指定“-lname”参数类似，但忽略字符大小写的差别
-iname<范本样式>	此参数的效果和指定“-name”参数类似，但忽略字符大小写的差别
-inum<inode 编号>	查找符合指定的 inode 编号的文件或目录
-ipath<范本样式>	此参数的效果和指定“-path”参数类似，但忽略字符大小写的差别
-iregex<范本样式>	此参数的效果和指定“-regexe”参数类似，但忽略字符大小写的差别
-links<连接数目>	查找符合指定的硬连接数目的文件或目录
-iname<范本样式>	指定字符串作为寻找符号连接的范本样式
-ls	假设 find 指令的回传值为 Ture，就将文件或目录名称列出到标准输出
-maxdepth<目录层级>	设置最大目录层级
-mindepth<目录层级>	设置最小目录层级
-mmin<分钟>	查找在指定时间曾被更改过的文件或目录，单位以分钟计算
-mount	参数的效果和指定“-xdev”相同
-mtime<24 小时数>	查找在指定时间曾被更改过的文件或目录，单位以 24 小时计算
-name<范本样式>	指定字符串作为寻找文件或目录的范本样式
-newer<参考文件或目录>	查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录
-nogroup	找出不属于本地主机群组识别码的文件或目录

-noleaf	不去考虑目录至少需拥有两个硬连接存在
-nouser	找出不属于本地主机用户识别码的文件或目录
-ok<执行指令>	此参数的效果和指定“-exec”类似，但在执行指令之前会先询问用户，若回答“y”或“Y”，则放弃执行命令
-path<范本样式>	指定字符串作为寻找目录的范本样式
-perm<权限数值>	查找符合指定的权限数值的文件或目录
-print	假设 find 指令的回传值为 Ture，就将文件或目录名称列出到标准输出。格式为每列一个名称，每个名称前皆有“./”字符串
-print0	假设 find 指令的回传值为 Ture，就将文件或目录名称列出到标准输出。格式为全部的名称皆在同一行
-printf<输出格式>	假设 find 指令的回传值为 Ture，就将文件或目录名称列出到标准输出。格式可以自行指定
-prune	不寻找字符串作为寻找文件或目录的范本样式
-regex<范本样式>	指定字符串作为寻找文件或目录的范本样式
-size<文件大小>	查找符合指定的文件大小的文件
-true	将 find 指令的回传值皆设为 True
-type<文件类型>	只寻找符合指定的文件类型的文件
-uid<用户识别码>	查找符合指定的用户识别码的文件或目录
-used<日数>	查找文件或目录被更改之后在指定时间曾被存取过的文件或目录，单位以日计算
-user<拥有者名称>	查找符和指定的拥有者名称的文件或目录
-xtype<文件类型>	此参数的效果和指定“-type”参数类似，差别在于它针对符号连接检查
-xdev	将范围局限在先行的文件系统中

1.3.3 常用参数的使用

（1）列出当前目录及子目录下所有文件和文件夹：

```
ubuntu@VM-0-14-ubuntu:~$ find test
test
test/test2
test/test2/cumt.txt
test/test2/yxj.txt
```

（2）在目录下查找以.php 结尾的文件名（-name），使用-iname 时将忽略大小写：

```
ubuntu@VM-0-14-ubuntu:~$ find . -name "*.php"
./ctf/Rogue-MySql-Server/roguemysql.php
./flag.php
```

（3）目录及子目录下查找所有以.txt 和.py 结尾的文件

```
ubuntu@VM-0-14-ubuntu:~$ find test -name "*.txt" -o -name "*.py"
test/3.txt
test/test2/cumt.txt
test/test2/yxj.txt
test/1.py
test/2.py
```

（4）匹配文件路径或者文件中的字符串（-path）：

```
ubuntu@VM-0-14-ubuntu:~$ find . -path "*test2*"
./test/test2
./test/test2/cumt.txt
./test/test2/yxj.txt
./test2.txt
./.test2.txt.swp
```

（5）基于正则表达式匹配文件路径（-regex）：

```
ubuntu@VM-0-14-ubuntu:~$ find test -regex ".*\(\.txt\|\.py\)$"
test/3.txt
test/test2/cumt.txt
test/test2/yxj.txt
test/1.py
test/2.py
```

（6）找出不是以.txt 结尾的文件：

```
ubuntu@VM-0-14-ubuntu:~$ find test ! -name "*.txt"
test
test/2.php
test/test2
test/1.py
test/2.py
test/1.php
```

（7）根据文件类型进行搜索，类型参数列表如下为（-type）：

① f：普通文件 ② l：符号连接 ③ d：目录 ④ c：字符设备 ⑤ b：块设备 ⑥ s：套接字 ⑦ p：FIFO

```
ubuntu@VM-0-14-ubuntu:~$ find test -type d
test
test/test2
ubuntu@VM-0-14-ubuntu:~$ find test -type f
test/3.txt
test/2.php
test/test2/cumt.txt
test/test2/yxj.txt
test/1.py
test/2.py
test/1.php
```

（8）向下最大深度限制为（-maxdepth）：

```
ubuntu@VM-0-14-ubuntu:~$ ls test
1.php  1.py  2.php  2.py  3.txt  test2
ubuntu@VM-0-14-ubuntu:~$ find test -maxdepth 1 -type f
test/3.txt
test/2.php
test/1.py
test/2.py
test/1.php
```

（9）搜索最近 3 天内被访问过的所有文件（-atime）：

```
ubuntu@VM-0-14-ubuntu:~$ find test -type f -atime -3
test/3.txt
test/2.php
test/test2/cumt.txt
test/test2/yxj.txt
test/1.py
test/2.py
test/1.php
```

（10）删除当前目录下所有.txt 文件（-delete）：

```
ubuntu@VM-0-14-ubuntu:~/test/test2$ ls
cumt.txt  yxj.txt
ubuntu@VM-0-14-ubuntu:~/test/test2$ find . -type f -name "*.txt" -delete
ubuntu@VM-0-14-ubuntu:~/test/test2$ ls
ubuntu@VM-0-14-ubuntu:~/test/test2$ ls -a
.  ..
```

（11）当前目录下搜索出权限为 777 的文件（-perm）

```
ubuntu@VM-0-14-ubuntu:~/test$ ls -l
total 4
-rwxrwxrwx 1 ubuntu ubuntu    0 Mar 16 17:50 1.php
-rw-rw-r-- 1 ubuntu ubuntu    0 Mar 16 17:50 1.py
-rw-rw-r-- 1 ubuntu ubuntu    0 Mar 16 17:50 2.php
-rw-rw-r-- 1 ubuntu ubuntu    0 Mar 16 17:50 2.py
-rwxrwxrwx 1 ubuntu ubuntu    0 Mar 16 17:50 3.txt
drwxrwxr-x 3 ubuntu ubuntu 4096 Mar 16 18:03 test2
ubuntu@VM-0-14-ubuntu:~/test$ find . -type f -perm 777
./3.txt
./1.php
```

（12）找出当前目录某个用户拥有的所有文件（-user）：

```
ubuntu@VM-0-14-ubuntu:~$ ls -l
total 56
-rw-r--r--  1 root   root    206 Mar  7 11:08 app.py
drwxrwxr-x  3 ubuntu ubuntu 4096 Feb 29 16:26 CheckIn
-rw-rw-r--  1 ubuntu ubuntu 1753 Feb 29 03:26 CRT.py
drwxrwxr-x  4 ubuntu ubuntu 4096 Mar  8 14:41 ctf
drwxr-xr-x 10 root   root   4096 Jan  7 17:21 CTFd
drwxrwxr-x  4 ubuntu ubuntu 4096 Feb 29 03:28 CUMTCTF
-rw-rw-r--  1 ubuntu ubuntu   10 Feb 19 22:38 flag.php
-rw-rw-r--  1 ubuntu ubuntu 4196 Mar 11 22:09 geturl.py
-rwxrwxr-x  1 ubuntu ubuntu   33 Mar  2 09:30 hello.sh
drwxrwxr-x  3 ubuntu ubuntu 4096 Mar 16 17:50 test
-rw-rw-r--  1 ubuntu ubuntu   17 Mar 16 14:41 test2.txt
-rw-rw-r--  1 ubuntu ubuntu    7 Mar 16 14:38 test3.txt
-rw-rw-r--  1 ubuntu ubuntu   51 Mar 16 14:58 test.txt
ubuntu@VM-0-14-ubuntu:~$ find test -type f -user ubuntu
test/3.txt
test/2.php
test/test2/cumt.txt
test/test2/yxj.txt
test/test2/test3/123.txt
test/1.py
test/2.py
test/1.php
```

（13）找出当前目录某个用户组拥有的所有文件（-group）：

```
ubuntu@VM-0-14-ubuntu:~$ find test -type f -group ubuntu
test/3.txt
test/2.php
test/test2/cumt.txt
test/test2/yxj.txt
test/test2/test3/123.txt
test/1.py
test/2.py
test/1.php
```

（14）查找当前目录下某些.txt 文件并把他们拼接起来写入到 all.txt 文件中（-exec）

```
ubuntu@VM-0-14-ubuntu:~/test$ cat 1.txt 2.txt
abcde
12345
ubuntu@VM-0-14-ubuntu:~/test$ find . -type f -name "[1-2].txt" -exec cat {} \;> all.txt
ubuntu@VM-0-14-ubuntu:~/test$ cat all.txt
12345
abcde
```

（15）查找当前目录或者子目录下所有.txt 文件，但是跳过子目录某个子目录，如（test/test2）

```
ubuntu@VM-0-14-ubuntu:~$ find test -path "test/test2" -prune -o -name "*.txt" -print
test/2.txt
test/all.txt
test/1.txt
```

（16）根据文件大小进行匹配，文件大小单元：

① b：块（512 字节） ② c：字节 ③ w：字（2 字节） ③ k：千字节 ④ M：兆字节 ⑤G：吉字节

搜索小于 10b 的文件（-size）：

```
ubuntu@VM-0-14-ubuntu:~$ find test -type f -size -10k
test/2.txt
test/2.php
test/all.txt
test/1.txt
test/test2/cumt.txt
test/test2/yxj.txt
test/test2/test3/123.txt
test/1.py
test/2.py
test/1.php
```

1.3.4 收获与体会

在进行总结的过程中，find 命令可以说是参数最多最灵活的命令之一了，同时 find 命令也是一个无处不在的命令，是 Linux 中最有用的命令之一。它不仅可以在目录（及子目录）中搜索文件，还可以指定一些匹配条件，如按文件名、文件类型、用户甚至是时间戳查找文件，所以要想使用好 find 命令，还需要对 Linux 系统下的文件类别及其对应的字母具有一定的了解。而当你熟练的掌握 find 命令之后，可以说你的 Linux 水平就会上了一个台阶，对 Linux 的管理与使用也会更加的得心应手。

1.4 tar 命令

1.4.1 作用及基本语法

（1）tar 命令可以为 Linux 的文件和目录创建档案。它可以为某一特定文件创建档案（备份文件），也可以在档案中改变文件，或者向档案中加入新的文件。通常使用 tar 命令也把多个的文件和目录全部打包成一个文件，方便在网络上的传输。

（2）tar 命令的基本语法：tar[必要参数][选择参数][文件]

1.4.2 全部参数的总结

参数	作用
-A	新增文件到以存在的备份文件
-B	设置区块大小
-c	建立新的备份文件
-C <目录>	这个选项用在解压缩，若要在特定目录解压缩，可以使用这个选项
-d	记录文件的差别
-x	从备份文件中还原文件
-t	列出备份文件的内容
-z	通过 gzip 指令处理备份文件
-Z	通过 compress 指令处理备份文件
-f<备份文件>	指定备份文件
-v	显示指令执行过程
-r	添加文件到已经压缩的文件
-u	添加改变了和现有的文件到已经存在的压缩文件
-j	支持 bzip2 解压文件
-v	显示操作过程
-k	保留原有文件不覆盖
-l	文件系统边界设置
-m	保留文件不被覆盖
-w	确认压缩文件的正确性
-p	用原来的文件权限还原文件
-P	文件名使用绝对名称，不移除文件名称前的“/”号
-N <日期格式>	只将较指定日期更新的文件保存到备份文件里

–exclude=<范本样式>	排除符合范本样式的文件

1.4.3 常用参数的使用

（1）将文件全部打包成 tar 包：

① 仅打包，不压缩

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -cvf cumt1.tar cumt1.txt
cumt1.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt1.txt  cumt2.txt  cumt3.txt
```

② 打包后，以 gzip 压缩

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -zcvf cumt2.tar.gz cumt2.txt
cumt2.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt1.txt  cumt2.tar.gz  cumt2.txt  cumt3.txt
```

③ 打包后，以 bzip2 压缩

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -jcvf cumt3.tar.bz2 cumt3.txt
cumt3.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt1.txt  cumt2.tar.gz  cumt2.txt  cumt3.tar.bz2  cumt3.txt
```

（2）查看 tar 包内有哪些文件：

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -ztvf cumt2.tar.gz
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:55 cumt2.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -tvf cumt1.tar
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:56 cumt1.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -ztvf cumt2.tar.gz
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:55 cumt2.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -jtvf cumt3.tar.bz2
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:55 cumt3.txt
```

（3）将 tar 包解压缩：

```
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt2.tar.gz  cumt3.tar.bz2
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -zxvf cumt2.tar.gz
cumt2.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt2.tar.gz  cumt2.txt  cumt3.tar.bz2
```

（4）只将 tar 内的部分文件解压出来：

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -ztvf cumtall.tar.gz
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:56 cumt1.txt
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:55 cumt2.txt
-rw-rw-r-- ubuntu/ubuntu      6 2020-03-16 20:55 cumt3.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -zxvf cumtall.tar.gz cumt2.txt cumt3.txt
cumt2.txt
cumt3.txt
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.tar  cumt2.tar.gz  cumt2.txt  cumt3.tar.bz2  cumt3.txt  cumtall.tar.gz
```

（5）文件备份下来，并且保存其权限：

```
ubuntu@VM-0-14-ubuntu:~/cumt$ ls
cumt1.txt  cumt2.txt  cumt3.txt  cumtall.tar.gz
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -zcvpf cumtall.tar.gz cumt1.txt cumt2.txt cumt3.txt
cumt1.txt
cumt2.txt
cumt3.txt
```

（6）在文件夹当中，比某个日期新的文件才备份：

```
ubuntu@VM-0-14-ubuntu:~/cumt$ tar -N "2020/01/01" -zcvf cumtall.tar.gz cumt1.txt
tar: Option --after-date: Treating date '2020/01/01' as 2020-01-01 00:00:00
cumt1.txt
```

（7）备份文件夹内容是排除部分文件：

```
ubuntu@VM-0-14-ubuntu:~$ tar --exclude cumt/ -zcvf  cumtall.tar.gz cumt/*
cumt/cumt1.txt
cumt/cumt2.txt
cumt/cumt3.txt
cumt/cumtall.tar.gz
```

1.4.4 收获与体会

tar 命令也是在使用 Linux 时不得不提的一个命令，其不仅可以用来对系统中的文件进行备份，可以备份数据到任何存储介质上；也可以用来对多个文件进行打包、压缩、解压等操作。在 Windows 下打包文件非常的容易，但是在 Linux 的纯命令行中就得靠 tar 命令的使用了，要想在网络中传输数据，就避免不了用此命令对文件进行打包，因此这也是一个比较常用的命令，实验过程中我对一些常用的功能进行了实践，掌握了基本的使用方法。

1.5 sort 命令

1.5.1 作用及基本语法

（1）sort 命令是将文件进行排序（以行为单位），并将排序结果标准输出。sort 命令既可以从特定的文件，也可以从 stdin 中获取输入。

（2）sort 命令的基本语法：

sort[-bcdfimMnur][-o<输出文件>][-t<分隔字符>][-k<n,m>][文件]

1.5.2 全部参数的总结

参数	作用
-b	忽略每行前面开始出的空格字符
-c	检查文件是否已经按照顺序排序
-d	排序时，处理英文字母、数字及空格字符外，忽略其他的字符

-f	排序时，将小写字母视为大写字母
-i	排序时，除了 040 至 176 之间的 ASCII 字符外，忽略其他的字符
-m	将几个排序的文件进行合并
-M	将前面 3 个字母依照月份的缩写进行排序
-n	依照数值的大小排序
-u	意味着是唯一的(unique)，输出的结果是去完重了的
-o<输出文件>	将排序后的结果存入指定的文件
-r	以相反的顺序来排序
-t<分隔字符>	指定排序时所用的栏位分隔字符
-k [n,m]	按照指定的字段范围排序。从第 n 个字段开始，到第 m 个字（默认到行尾）
+<起始栏位>-<结束栏位>	以指定的栏位来排序，范围由起始栏位到结束栏位的前一栏位

1.5.3 常用参数的使用

（1）默认是从首字符向后，依次按 ASCII 码值进行比较，最后将他们按升序输出：

```
ubuntu@VM-0-14-ubuntu:~$ cat cumt.txt
aaa:10:1.1
ccc:30:3.3
ddd:40:4.4
bbb:20:2.2
eee:50:5.5
eee:50:5.5
ubuntu@VM-0-14-ubuntu:~$ sort cumt.txt
aaa:10:1.1
bbb:20:2.2
ccc:30:3.3
ddd:40:4.4
eee:50:5.5
eee:50:5.5
```

（2）忽略相同行（-u）：

```
ubuntu@VM-0-14-ubuntu:~$ cat cumt.txt
aaa:10:1.1
ccc:30:3.3
ddd:40:4.4
bbb:20:2.2
eee:50:5.5
eee:50:5.5
ubuntu@VM-0-14-ubuntu:~$ sort -u cumt.txt
aaa:10:1.1
bbb:20:2.2
ccc:30:3.3
ddd:40:4.4
eee:50:5.5
```

（3）按第 2 列数字从小到大排序（-t 指定分隔符，-k 指定列数）：

```
ubuntu@VM-0-14-ubuntu:~$ cat cumt.txt
AAA:BB:CC
aaa:30:1.6
ccc:50:3.3
ddd:20:4.2
bbb:10:2.5
eee:40:5.4
eee:60:5.1
ubuntu@VM-0-14-ubuntu:~$ sort -nk 2 -t: cumt.txt
AAA:BB:CC
bbb:10:2.5
ddd:20:4.2
aaa:30:1.6
eee:40:5.4
ccc:50:3.3
eee:60:5.1
```

（4）按第 3 列数字从大到小顺序排序（在上面的基础上使用-r 进行逆序）：

```
ubuntu@VM-0-14-ubuntu:~$ sort -nrk 3 -t: cumt.txt
eee:40:5.4
eee:60:5.1
ddd:20:4.2
ccc:50:3.3
bbb:10:2.5
aaa:30:1.6
AAA:BB:CC
```

（5）输出排序后的文件：

```
ubuntu@VM-0-14-ubuntu:~$ sort cumt.txt -o sort.txt
ubuntu@VM-0-14-ubuntu:~$ cat sort.txt
aaa:30:1.6
AAA:BB:CC
bbb:10:2.5
ccc:50:3.3
ddd:20:4.2
eee:40:5.4
eee:60:5.1
```

1.5.4 收获与体会

sort 命令也是 Linux 中一个非常常用的命令，并且要想真正的使用好 sort 命令也不是一件容易的事情。通常来说，使用 sort 命令可以对文件的内容进行排序的输出，默认是按照 ASCII 码的方式，通过一些参数的使用则可以指定排序方式。其中比较复杂的就是-k 与其他参数的配合使用，我在实验过程中进行了部分简单功能的实现，而熟练的使用 sort 命令无论实在系统使用还是 Shell 编程中，都会方便很多。

参考文献

[1] 鸟哥. 鸟哥的 Linux 私房菜: 基础学习篇[M]. 人民邮电出版社, 2010.

[2] 赵永华. 合理利用 Linux 系统命令[J]. 网管员世界, 2012(8):68-68.

[3] 菜鸟教程. Linux 教程. https://www.runoob.com/linux/linux-tutorial.html