基本命令-2

压缩和归档

打包：即归档，类似于旅游之前收拾行李

压缩：为了减少占用的磁盘空间，可以做备份，在网络上传输时节省网络带宽。

打包压缩软件

windows：winrar 360压缩 好压 7zip winzip

linux：压缩格式：gz ,bz2,xz,zip，Z

压缩算法不同，导致压缩比不同

压缩软件 gzip bzip2 xz zip

既能打包又能压缩的软件：tar ****

一、zip：

压缩后的文件一般以.zip结尾，可以压缩目录

压缩的语法： zip filename.zip file1 file2 ...

zip 压缩后的文件名 待压缩文件

压缩后不删除原文件

例如 [root@server150 acltest]# zip com.zip com.txt

adding: com.txt (deflated 99%)

解压缩：unzip

-d：指定解压路径

二、gzip

1）gzip /path/to/somefile

默认会删除原文件

-d 解压缩

-#: 指定压缩比，压缩比越小，速度越大

2）gunzip /path/to/some_compress_file

3）zcat some_compress_file 不解压的情况下查看文本的压缩文件

例子：#cp /var/log/messages ./

gzip messages

默认后缀名：.gz

gzip压缩时，原文件消失，生成压缩文件

解压：gunzip

gzip的压缩包，在解压后，压缩包消失，解压后的文件出现。

压缩其实是有级别的：1~9 1级别最低，速度最快，效率最低；9级别最高，速度最慢，效率最高。

默认级别是6。

三、bzip2

默认情况下，压缩完成，

原文件也是消失的，压缩包必须以.bz2结尾的

通常能够生成比使用gzip压缩更小的文件(压缩比较高)

1）bzip2 /path/to/somefile

-d:解压

-#：指定压缩比

-k:保留原文件

2）bunzip2 /path/to/somefile

3)bzcat /path/to/some_compress_file 不解压查看

解压缩：bunzip2

四. xz 压缩比更大

1)压缩 xz somefile

2）解压

unxz

或 xzdec

-d:解压

-k:保留原文件

-c:输入到屏幕

3)xzcat 不解压查看

以后常见的压缩包的格式

.zip .tar.gz .tar.bz2

五、tar *** 既可以打包，又可以压缩

tar 选项 包名 打包的文件或目录 //切记：一定要注意语法格式，先是打包后的名字，然后才是要打包的东西

tar: 归档工具, .tar

例如：tar -cf

-c: 创建归档文件

-f FILE.tar: 操作的归档文件

-x: 展开归档

--xattrs: 归档时，保留文件的扩展属性信息

-t: 不展开归档，直接查看归档了哪些文件

-C:解压时指定路径

-r：向包中追加文件

-p:保留权限信息

-v:显示详细过程

-zcf: 归档并调用gzip压缩
-zxf: 调用gzip解压缩并展开归档，-z选项可省略

-jcf: bzip2
-jxf:

-Jcf: xz
-Jxf:

1）打包压缩同时进行
-z：表示使用gzip压缩方式压缩或者解压缩
-j：表示使用bzip2压缩方式压缩或者解压缩
-c：表示创建 --create
-v：显示详细过程
-f：指定文件，一般后面跟包名
-zcvf zcvf .tar.gz
-jcvf jcvf .tar.bz2
tar zcvf com.tar.gz com.txt
com.txt
ll
total 67968
-rw-r--r-- 1 root root 367957 Jul 30 09:24 com.tar.gz
tar zcvf /tmp/acltest.tar.gz /acltest/

2）解包 .tar.gz .tar.bz2
-zxvf zxvf
-jxvf jxvf
-C：指定解压路径
tar zxvf com.tar.gz -C /usr/local/src/
ls /usr/local/src/
com.txt vmware-tools-distrib

3）其他选项
-t：不解包查看包中的内容
tar -tf /tmp/acltest.tar.gz

acltest/
acltest/f1
acltest/com.txt
acltest/f3
acltest/f2
acltest/com.zip
acltest/com.tar.gz

-r：向包中追加文件，不能追加压缩的文件

tar -rf 包名 追加的文件

三.文件搜索

which：用来查找命令的绝对路径

-- 显示shell命令的绝对路径

-- 仅仅会在PATH变量中搜索要查找的命令

-- 搜索时先查找别名，然后从PATH中查找

1、查看用户的PATH变量：命令的搜索路径

echo $PATH

/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin:/root/bin

---

command not found可能原因：

1）敲错了

2）命令没有安装

3）命令所在路径没在PATH变量的定义中

---

cp `which vim` /tmp/vim2

vim2 /etc/passwd

bash: vim2: command not found

/tmp/vim2 /etc/passwd //绝对路径执行

2、添加路径到PATH

1）临时修改PATH值

PATH=$PATH:/tmp //$PATH：保留变量原有值

echo $PATH

/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin:/root/bin:/tmp

2）永久修改PATH值　工作中一定会用的

/etc/profile　　//全局配置文件，对所有用户生效

~username/.bash_profile　　//局部配置文件，只对特定用户生效

vim /root/.bash_profile

PATH=$PATH:$HOME/bin:/tmp

上述文件不是即时生效的，正常情况下，它是用户登录时执行的。

source /root/.bash_profile　　//重新读取配置文件，使修改生效

echo $PATH

/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin:/root/bin:/root/bin:/tmp

弊端：每次新开启一个终端或标签，都需要执行# source /root/.bash_profile

如果想一劳永逸，那么需要退出系统，重新登录，即注销。

System ——> Log out root ——> Log out

echo $PATH

/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin:/root/bin:/tmp

---

which ls

alias ls='ls --color=auto'

/bin/ls

which vim

/usr/bin/vim

命令的别名：　alias

1、查看当前系统中有哪些别名(root用户和普通用户的别名可能不一样)

alias

2、设置命令的别名

1）临时

alias vi='vim'

vi /etc/passwd　　//执行vi时候，实际上执行的是vim

2）永久，改文件

（1）/root/.bashrc　　cp　rm　mv

（2）/etc/profile.d

colorls.sh

which2.sh

3、取消别名

[profile.d]# unalias vi
[profile.d]# vi /etc/passwd //没颜色了

locate
-- 通过文件名检索文件，检索速度最快
-- 所有能够检索的东西，都是存放在数据库中的
-- locate局限性，有的文件系统、有的文件及有的目录默认是不会搜索的

1、假设我知道网卡配置文件的名字，但是不知道具体路径：
locate ifcfg-eth0
/etc/sysconfig/network-scripts/ifcfg-eth0
locate ifcfg
/etc/dbus-1/system.d/nm-ifcfg-rh.conf
/etc/sysconfig/network-scripts/ifcfg-eth0
/etc/sysconfig/network-scripts/ifcfg-lo
/sbin/ifcfg
/usr/lib64/NetworkManager/libnm-settings-plugin-ifcfg-rh.so
/usr/share/man/man8/ifcfg.8.gz
/var/log/anaconda.ifcfg.log

2、手动更新数据库
cp `which vim` /root/vim3
locate vim3 //未查询到结果
原因：因为locate的数据库是一天一更新，不是实时更新的。
updatedb
locate vim3
/root/vim3

数据库文件：/var/lib/mlocate/mlocate.db

报错：
1）数据库文件不存在
2）手动生成它
updatedb

3、locate数据库配置文件
vim /etc/updatedb.conf
ls /tmp/vim2
/tmp/vim2

locate vim2 //搜索不到，因为/tmp在排除列表中

find *****

-- 全局性搜索文件

-- 工作方式：沿着文件的层次结构依次向下搜索，找到符合条件的，打印或者是执行相应的操作

一、语法格式

find 要搜索路径 条件(选项) [动作]

1、基本例子

```
# find /etc/ -name network
  /etc/vmware-tools/scripts/vmware/network
  /etc/sysconfig/network
  /etc/rc.d/init.d/network
```

一般情况下，查找范围越大，目录层级越深，查找速度就越慢。

```
# mkdir -p /q/w/e/r/t/y/u/i/o/p/a/s/d/f/g/h/j/k/l/z/x/c/v/b/n/m
# touch /q/w/e/r/t/y/u/i/o/p/a/s/d/f/g/h/j/k/l/z/x/c/v/b/n/m/test.txt
# time find / -name test.txt
  /q/w/e/r/t/y/u/i/o/p/a/s/d/f/g/h/j/k/l/z/x/c/v/b/n/m/test.txt
  /acl/test.txt
  /tmp/test.txt
  real    0m3.188s
  user    0m0.061s
  sys     0m2.953s
```

练习：

查找/etc目录下名字为config的文件

```
# find /etc -name config
    /etc/vmware-tools/config
    /etc/selinux/config
```

2、按条件查找

1）按照文件名搜索

-name：按名字查找 *****

-iname：忽略大小写

```
# find /etc -name networkmanager
# find /etc -iname networkmanager
  /etc/rc.d/init.d/NetworkManager
  /etc/NetworkManager
```

通配符：* 代表任意字符

？代表单个字符

查找/etc目录下，所有以.conf结尾的文件

find /etc/ -name *.conf

查找/etc/目录下，以.conf结尾，名称是5个字符的

find /etc/ -name "?????.conf"

练习：

查找/etc目录下，名字为ntp.conf的文件

find /etc -name ntp.conf

/etc/ntp.conf

查找 / 目录下，名字为passwd的文件

find / -name passwd

2）按照文件类型查找 -type

f：普通文件

b c d l s p

find /var -type l //查找/var目录下类型是软链接的文件

ll `find /var -type l` //验证是否查找到的都是软链接

find /tmp/ -type f

-type 后面只能跟一个字母

练习： 查找系统中类型是套接字的文件

find / -type s

查找的时候，可能会遇到No such file or directory的错误提示，是正常的，若不想看到可以将错误重定向到“黑洞”(/dev/null)

find / -type s 2> /dev/null

3）按照时间查找 (笔试题)

-atime n 以天为单位

-ctime n

-mtime n

-amin n 以分钟为单位

-cmin n

-mmin n

n为数字，前面可以带+或者-号 -mtime n

+n：n+1天之前

n：n到n+1天之间

-n：n天以内

以n等于7为例：

搜索最近七天内被访问过的所有文件

find . -type f -atime -7

搜索恰好在七天前被访问过的所有文件

find . -type f -atime 7

搜索超过七天内被访问过的所有文件

find . -type f -atime +7

搜索访问时间超过10分钟的所有文件

find . -type f -amin +10

找出比file.log修改时间更长的所有文件

find . -type f -newer file.log

插曲：

查看文件的属性信息？ stat 文件名

```
# stat passwd
  File: `passwd'
  Size: 1030            Blocks: 8          IO Block: 4096   regular file
Device: 802h/2050d      Inode: 917613      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2015-08-08 20:38:22.080984537 +0800
Modify: 2015-08-08 20:38:22.080984537 +0800
Change: 2015-08-08 20:38:22.080984537 +0800
```

一个文件有三个时间：

atime：访问时间，cat more less ... ...

mtime：文件的内容发生变化的时间 vim ... ... (也是 ll 所显示的时间)

ctime：文件的属性发生变化的时间 比如：权限改变、大小改变、所有者、所属组等改变

例子：

```
# echo hello > file
# stat file
  File: `file'
  Size: 6               Blocks: 8          IO Block: 4096   regular file
Device: 802h/2050d      Inode: 917619      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
```

```
Access: 2015-08-13 12:16:19.494018488 +0800
Modify: 2015-08-13 12:16:19.495018470 +0800
Change: 2015-08-13 12:16:19.495018470 +0800
# cat file
hello
# stat file
  File: `file'
  Size: 6               Blocks: 8          IO Block: 4096   regular file
Device: 802h/2050d      Inode: 917619      Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2015-08-13 12:17:32.381018468 +0800
Modify: 2015-08-13 12:16:19.495018470 +0800
Change: 2015-08-13 12:16:19.495018470 +0800
# chmod +x file
# stat file
  File: `file'
  Size: 6               Blocks: 8          IO Block: 4096   regular file
Device: 802h/2050d      Inode: 1197781     Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2015-08-13 12:17:32.381018468 +0800
Modify: 2015-08-13 12:16:19.495018470 +0800
Change: 2015-08-13 12:21:55.563018148 +0800    //ctime发生变化
# echo hahaha >> file
# stat file
  File: `file'
  Size: 13          Blocks: 8          IO Block: 4096   regular file
Device: 802h/2050d      Inode: 1197781     Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2015-08-13 12:17:32.381018468 +0800
Modify: 2015-08-13 12:23:54.286017831 +0800
Change: 2015-08-13 12:23:54.286017831 +0800
```

笔试题

1、打印文件的数字权限

```
# stat -c %a file
  755
```

2、打印文件的字母权限

```
# stat -c %A file
```

-rwxr-xr-x

按时间查找小例子

mkdir /find

date

Thu Aug 13 14:22:43 CST 2015

cd /find/

[find]# touch -t 08131425.30 f0813 //-t：指定文件的创建时间 MMDDHHmm.SS

[find]# touch -t 08121425.50 f0812

[find]# touch -t 08111426.30 f0811

[find]# touch -t 08101426.30 f0810

[find]# ll f*

-rw-r--r-- 1 root root 0 Aug 10 14:26 f0810

-rw-r--r-- 1 root root 0 Aug 11 14:26 f0811

-rw-r--r-- 1 root root 0 Aug 12 14:25 f0812

-rw-r--r-- 1 root root 0 Aug 13 14:25 f0813

-- 查找/find下修改时间在24小时(1天)之内的文件

[find]# find . -mtime -1

.

./f0813

-- 查找/find下修改时间在2天前的普通文件

[find]# find . -type f -mtime +1

./f0810

./f0811

4）按照用户和组查找

-user 用户名

-group 组名

-uid uid

-gid gid

-nouser：孤儿文件 没有所有者的文件

-nogroup：没有所属组的文件

--查找系统中所有者是quota2的文件

[home]# find / -user quota2 -type f

[home]# find / -user quota2 -type f 2>/dev/null

-- 查找系统中的孤儿文件

[home]# userdel quota2

[home]# find . -type f -nouser
./quota2/.bash_history
./quota2/.bashrc
./quota2/.bash_profile
./quota2/.bash_logout
[home]# ll `find . -type f -nouser`
-rw------- 1 502 quota2 29 Aug 11 10:16 ./quota2/.bash_history
-rw-r--r-- 1 502 quota2 18 Aug 29 2012 ./quota2/.bash_logout
-rw-r--r-- 1 502 quota2 176 Aug 29 2012 ./quota2/.bash_profile
-rw-r--r-- 1 502 quota2 124 Aug 29 2012 ./quota2/.bashrc

-- 查找系统中所有者不是root的普通文件 ！或者 -not

[home]# find / ! -user root -type f

或者 -or 或者 -o

-- 查找系统中所有者不是root或者类型是套接字的文件

[home]# find / ! -user root -o -type s

5）按照权限查找 -perm

+222 或者 （用户可写or组可写or其他人可写） 二进制中有1的位置，只要满足其中一个位就可以

-222 并且 （用户可写and组可写and其他人可写） 二进制中有1的位置必须都要有1

rm -f /find/*
cd /find/
[find]# touch p{r,w,x}_{1,2,3}
[find]# chmod 400 pr_1
[find]# chmod 440 pr_2
[find]# chmod 444 pr_3
[find]# chmod 200 pw_1
[find]# chmod 220 pw_2
[find]# chmod 222 pw_3
[find]# chmod 100 px_1
[find]# chmod 110 px_2
[find]# chmod 111 px_3
[find]# ll `find ./ -perm +020 -type f`
--w--w---- 1 root root 0 Aug 13 15:13 ./pw_2
--w--w--w- 1 root root 0 Aug 13 15:13 ./pw_3
[find]# ll `find ./ -perm -020 -type f`
--w--w---- 1 root root 0 Aug 13 15:13 ./pw_2

--w--w--w- 1 root root 0 Aug 13 15:13 ./pw_3

当权限位只有一位的时候，+和-是一样的。

[find]# ll `find ./ -perm -222 -type f`

--w--w--w- 1 root root 0 Aug 13 15:13 ./pw_3

[find]# ll `find ./ -perm +222 -type f`

--w------- 1 root root 0 Aug 13 15:13 ./pw_1

--w--w---- 1 root root 0 Aug 13 15:13 ./pw_2

--w--w--w- 1 root root 0 Aug 13 15:13 ./pw_3

思考：查找系统中拥有suid权限的文件

6）按照文件大小查找 -size

\+ 大于

\- 小于

直接数字 等于

'b' for 512-byte blocks (this is the default if no suffix is used) //0.5KB

'c' for bytes

'w' for two-byte words

'k' for Kilobytes (units of 1024 bytes)

'M' for Megabytes (units of 1048576 bytes)

'G' for Gigabytes (units of 1073741824 bytes)

[find]# rm -f /find/*

[find]# dd if=/dev/zero of=f1M bs=1M count=1

[find]# dd if=/dev/zero of=f2M bs=1M count=2

[find]# dd if=/dev/zero of=f3M bs=1M count=3

[find]# dd if=/dev/zero of=f4M bs=1M count=4

[find]# find . -type f -size -3M

./f2M

./f1M

[find]# find . -type f -size 3M

./f3M

[find]# find . -type f -size +3M

./f4M

3、动作

-exec 动作 -- 找到结果之后直接执行动作

-ok 动作 -- 执行动作之前先提示，即需要交互

[find]# find . -type f -size +3M -exec ls -l {} \;

-rw-r--r-- 1 root root 4194304 Aug 13 15:51 ./f4M

{} —— 用来代替找到的结果

\; —— 表示结束标志

[find]# find . -type f -size +3M -ok ls -l {} \;

< ls ... ./f4M > ? y

-rw-r--r-- 1 root root 4194304 Aug 13 15:51 ./f4M

[find]# find . -type f -size +3M -ok ls -l {} \;

< ls ... ./f4M > ? n

练习:

1、查找/find目录下，类型是 普通文件的文件将其移动到/test目录下

[find]# find . -type f -exec mv {} /test \;

[find]# ls /test/

f1M f2M f3M f4M

或者

[find]# mv `find . -type f` /test

2、查找/test目录下类型为普通文件的文件，对其进行备份，备份文件的后缀名为.bak

[find]# find /test -type f -exec cp {} {}.bak \;

[find]# ls /test/

f1M f1M.bak f2M f2M.bak f3M f3M.bak f4M f4M.bak

3、删除/test目录下修改时间在一天以内的普通文件

[find]# find /test/ -type f -mtime -1 -exec rm {} \;

grep文本过滤

一.grep：目的是过滤出用户感兴趣的内容 ***

语法：grep [选项] 模式或关键字 文件列表

简单例子:

[loring ~]# grep root /etc/passwd

root:x:0:0:root:/root:/bin/bash

operator:x:11:0:operator:/root:/sbin/nologin

1、--color 带颜色显示匹配到的关键字

[loring ~]# grep --color root /etc/passwd

root:x:0:0:root:/root:/bin/bash

operator:x:11:0:operator:/root:/sbin/nologin

2、-i 忽略大小写

[loring tmp]# grep --color -i root /find/passwd

Root:x:0:0:root:/root:/bin/bash

operator:x:11:0:operator:/root:/sbin/nologin

3、-v 取反

过滤出不包含nologin的行

[loring tmp]# grep -v nologin /etc/passwd

4、^ 以某关键字开头

显示/root/.bashrc文件中的非注释行

[loring tmp]# grep -v ^# /root/.bashrc

5、$ 以某关键字结尾

显示passwd文件中以sh结尾的行

[loring tmp]# grep sh$ /etc/passwd

6、^$ 空行

显示/root/.bashrc文件中的非注释行和非空行

[loring tmp]# grep -v ^# /root/.bashrc | grep -v ^$

7、-c count，统计匹配到的行数

[loring tmp]# grep -c root /etc/passwd

2

8、 -l 一般和-r联用，只显示包含关键字的文件的名字，而不是显示文件内容

9、-r 递归检索

显示test目录下文件内容中含有root的文件名

[loring tmp]# grep -rl root /test

10、-q quiet 静默输出 一般在写脚本时候用

[loring tmp]# grep -q root /etc/passwd

[loring tmp]# echo $? //$?表示上一条命令的执行结果

0

返回结果为0：表示上一条命令的执行时成功的

返回结果非0：表示上一条命令执行失败

[loring tmp]# grep -q jsjdjjdfhfh /etc/passwd
[loring tmp]# echo $?
1
[loring tmp]# grep -q root /asdaf
grep: /asdaf: No such file or directory
[loring tmp]# echo $? //文件不存在返回2
2

11、-n 显示匹配行的行号
[loring tmp]# grep -n root /etc/passwd
1:root:x:0:0:root:/root:/bin/bash
11:operator:x:11:0:operator:/root:/sbin/nologin

grep练习：
1、显示/etc/group文件中含有root的行
[loring test]# grep root /etc/group
2、显示/etc/passwd文件中以rp开头的行
[loring test]# grep ^rp /etc/passwd
3、显示/etc/group文件中不以:为结尾的行
[loring test]# grep -v :$ /etc/group
4、显示/etc/rc.local文件中的空行，要求带行号
[loring test]# grep -n ^$ /etc/rc.local
6:
5、显示仅/mnt/cdrom目录下的文件类型为目录的文件(前提是光盘已挂载，且不使用find)
[loring test]# ll /mnt/cdrom/ | grep ^d
二、cut：就是截取的意思，它的处理对象是“一行”文本，可以从中选取出用户所需要的部分，不影响原文件
可以指定分隔符，然后打印出以分隔符隔开的具体某一列或某几列
语法：cut -f 指定的列 -d ‘分隔符’
-d：指定字段分隔符
-f：指定要输出的区域，多个之间用逗号分隔
-c：指定列的字符

显示/etc/passwd文件中的用户名和uid字段
[loring test]# cut -d: -f1,3 /etc/passwd

[loring test]# head -5 /etc/passwd | cut -d: -f1,3
root:0
bin:1
daemon:2
adm:3
lp:4

[loring test]# head -5 /etc/passwd > t1
[loring test]# vim t1
:%s/:/ /g
[loring test]# cat t1
root x 0 0 root /root /bin/bash
bin x 1 1 bin /bin /sbin/nologin
daemon x 2 2 daemon /sbin /sbin/nologin
adm x 3 4 adm /var/adm /sbin/nologin
lp x 4 7 lp /var/spool/lpd /sbin/nologin
以空格为分隔，取出第一个字段
[loring test]# cut -d" " -f1 t1
root
bin
daemon
adm
lp

取出/etc/group文件中组名的字段
cut -d: -f1 /etc/group
打印/etc/passwd 文件中每行的第1-5个字符，以及第7-10个字符的内容
[root@web test]# cat /etc/passwd | cut -c 1-5,7-10

三、sort 排序
-t：指定字段分隔符
-k：指定第几个字段
-n：按照数字顺序排序
-r：反向排序 reverse
-u：排序后重复行只打印一次 unique
[root@web test]# cat sort.txt
b:3

c:2
a:4
e:5
d:1
f:11
对输出内容直接排序，默认按照每行的第一个字符进行排序
[root@web test]# cat sort.txt | sort
a:4
b:3
c:2
d:1
e:5
f:11
对输出内容进行反向排序
[root@web test]# cat sort.txt | sort -r
f:11
e:5
d:1
c:2
b:3
a:4
使用“：”做分隔符，对第2个字段进行排序
[root@web test]# cat sort.txt | sort -t ":" -k 2
d:1
f:11
c:2
b:3
a:4
e:5
使用“：”做分隔符，对第2个字段进行排序，按照数字大小排序
[root@web test]# cat sort.txt | sort -t ":" -k 2 -n
d:1
c:2
b:3
a:4
e:5
f:11

对/etc/passwd文件按照uid来排序

[loring test]# sort -t ":" -k 3 -n /etc/passwd

对passwd文件按照uid由大到小的顺序排序

[loring test]# sort -t ":" -k 3 -nr /etc/passwd

按照gid由小到大的顺序打印/etc/group

[loring test]# sort -t: -k 3 -n /etc/group

四、 uniq 去重，唯一

去除相邻重复行

-c： 显示重复的行数

-i: 忽略大小写

[loring test]# uniq num.txt

111

222

333

444

222

555

使用uniq时，一般先排序，再去重

[loring test]# sort num.txt | uniq

111

222

333

444

555

[loring test]# sort num.txt | uniq -c

1 111

3 222

2 333

1 444

1 555

五、tr 主要作用在于文本转换或者删除。

将/etc/passwd文件中的小写字母转换成大写字母

[root@web test]# cat /etc/passwd | tr '[a-z]' '[A-Z]'

将/etc/passwd文件中的“：”删除掉

[root@web test]# cat /etc/passwd | tr -d ":"

六、paste 文本合并。将文件按照行进行合并，中间使用tab隔开。

[root@web test]# cat a.txt

1

2

3

4

5

[root@web test]# cat b.txt

a

b

c

d

e

按照行合并文件

[root@web test]# paste a.txt b.txt

1	a

2	b

3	c

4	d

5	e

也可以使用-d指定合并文件时行间的分隔符

[root@web test]# paste -d: a.txt b.txt

1:a

2:b

3:c

4:d

5:e

[root@web test]#

练习：

统计/etc/passwd文件中一共有几种shell，并显示每种shell有几个用户

[loring test]# cut -d: -f7 /etc/passwd | sort | uniq -c