

## 【Linux】linux 常用基本命令

### 1、显示日期的指令： date

```
[vbird@www ~]$ date  
Mon Aug 17 17:02:52 CST 2009
```

```
[vbird@www ~]$ date +%Y/%m/%d  
2009/08/17
```

```
[vbird@www ~]$ date +%H:%M  
17:04
```

## 2、显示日历的指令：cal

```
[vbird@www ~]$ cal
  August 2009
Su Mo Tu We Th Fr Sa
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
```

[vbird@www ~]\$ cal 2009

2009

January

February

March

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 1 2 3 4 5 6 7 1 2 3 4 5 6 7

4 5 6 7 8 9 10 8 9 10 11 12 13 14 8 9 10 11 12 13 14

11 12 13 14 15 16 17 15 16 17 18 19 20 21 15 16 17 18 19 20 21

18 19 20 21 22 23 24 22 23 24 25 26 27 28 22 23 24 25 26 27 28

25 26 27 28 29 30 31 29 30 31

April

May

June

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 1 2 1 2 3 4 5 6

5 6 7 8 9 10 11 3 4 5 6 7 8 9 7 8 9 10 11 12 13

12 13 14 15 16 17 18 10 11 12 13 14 15 16 14 15 16 17 18 19 20

19 20 21 22 23 24 25 17 18 19 20 21 22 23 21 22 23 24 25 26 27

26 27 28 29 30 24 25 26 27 28 29 30 28 29 30

31

```
[vbird@www ~]$ cal 10 2009
```

```
October 2009
```

```
Su Mo Tu We Th Fr Sa
```

```
1 2 3
```

```
4 5 6 7 8 9 10
```

```
11 12 13 14 15 16 17
```

```
18 19 20 21 22 23 24
```

```
25 26 27 28 29 30 31
```

3、简单好用的计算器：bc

```
[vbird@www ~]$ bc
```

```
bc 1.06
```

```
Copyright 1991-1994, 1997, 1998, 2000 Free Software Foundation, Inc.
```

```
This is free software with ABSOLUTELY NO WARRANTY.
```

```
For details type `warranty'.
```

```
_<==这个时候，光标会停留在这里等待你的输入
```

```
[vbird@www ~]$ bc
```

```
bc 1.06
```

```
Copyright 1991-1994, 1997, 1998, 2000 Free Software Foundation, Inc.
```

```
This is free software with ABSOLUTELY NO WARRANTY.
```

```
For details type `warranty'.
```

```
1+2+3+4 <==只有加法时
```

```
10
```

```
7-8+3
```

```
2
```

```
10*52
```

```
520
```

```
10%3 <==计算『余数』
```

```
1
```

```
10^2
```

```
100
```

```
10/100 <==这个最奇怪！不是应该是 0.1 吗？
```

```
0
```

```
quit <==离开 bc 这个计算器
```

怎么 10/100 会变成 0 呢？这是因为 bc 预设仅输出整数，如果要输出小数点下位数，那么就必须要执行 `scale=number` ，那个 number 就是小数点位数，例如：

```
This is free software with ABSOLUTELY NO WARRANTY.
```

```
For details type `warranty'.
```

```
scale=3    <==没错！就是这里！！
```

```
1/3
```

```
.333
```

```
340/2349
```

```
.144
```

```
quit
```

## 4、重要的几个热键[Tab], [ctrl]-c, [ctrl]-d

[Tab]按键——具有『命令补全』不『档案补齐』的功能

[Ctrl]-c 按键——让当前的程序『停掉』

[Ctrl]-d 按键——通常代表着：『键盘输入结束(End Of File, EOF 戒 End OfInput)』的意思；另外，他也可以用来取代 exit

## 5、man

退出用 q,



```
man -f man
```

```
[vbird@www ~]$ man -f man
```

```
man          (1) - format and display the on-line manual pages
```

```
man          (7) - macros to format man pages
```

```
man.config [man] (5) - configuration data for man
```

```
[vbird@www ~]$ man 1 man <==这里是用 man(1) 的文件数据
```

```
[vbird@www ~]$ man 7 man <==这里是用 man(7) 的文件数据
```

6、数据同步写入磁盘： sync

输入 sync，那举在内存中尚未被更新的数据，就会被写入硬盘中；所以，这个指令在系统关机或重新启动前，很重要喔！最好多执行几次！

```
[root@www ~]# sync
```

## 7、惯用的关机指令： shutdown

```
[root@www ~]# /sbin/shutdown [-t 秒] [-arkhncff] 时间 [警告讯息]
```

选项与参数：

-t sec : -t 后面加秒数，亦即『过几秒后关机』的意思

-k : 不要真的关机，只是发送警告讯息出去！

-r : 在将系统的服务停掉之后就重新启动(常用)

-h : 将系统的服务停掉后，立即关机。(常用)

-n : 不经过 init 程序，直接以 shutdown 的功能来关机

-f : 关机并开机之后，强制略过 fsck 的磁盘检查

-F : 系统重新启动之后，强制进行 fsck 的磁盘检查

-c : 取消已经在进行的 shutdown 指令内容。

时间 : 这是一定要加入的参数！指定系统关机的时间！时间的范例底下会说明。

范例：

```
[root@www ~]# /sbin/shutdown -h 10 'I will shutdown after 10 mins'
```

# 告诉大家，这部机器会在十分钟后关机！并且会显示在目前登入者的屏幕前方！

# 至于参数有哪些呢？以下介绍几个吧！

此外，需要注意的是，时间参数请务必加入指令中，否则 shutdown 会自动跳到 run-level 1（就是单人维护的登入情况），这样就伤脑筋了！底下提供几个时间参数的例子吧：

```
[root@www ~]# shutdown -h now
```

立刻关机，其中 now 相当于时间为 0 的状态

```
[root@www ~]# shutdown -h 20:25
```

系统在今天的 20:25 分会关机，若在 21:25 才下达此指令，则隔天才关机

```
[root@www ~]# shutdown -h +10
```

系统再过十分钟后自动关机

```
[root@www ~]# shutdown -r now
```

系统立刻重新启动

```
[root@www ~]# shutdown -r +30 'The system will reboot'
```

再过三十分钟系统会重新启动，并显示后面的讯息给所有在线的使用者

```
[root@www ~]# shutdown -k now 'This system will reboot'
```

仅发出警告信件参数！系统并不会关机啦！吓唬人！

重启，关机： reboot, halt, poweroff

```
[root@www ~]# sync; sync; sync; reboot
```

```
[root@www ~]# shutdown -h now
```

```
[root@www ~]# poweroff -f
```

## 8、切换执行等级： init

Linux 共有七种执行等级：

--run level 0 :关机

--run level 3 :纯文本模式

--run level 5 :含有图形接口模式

--run level 6 :重新启动

使用 init 这个指令来切换各模式:

如果你想要关机的话, 除了上述的 shutdown -h now 以及 poweroff 之外, 你也可以使用如下的指令来关机:

```
[root@www ~]# init 0
```

## 9、改变文件的所属群组: chgrp



```
[root@www ~]# chgrp [-R] dirname/filename ...
```

选项与参数：

-R：进行递归(recursive)的持续变更，亦即连同次目录下的所有档案、目录都更新成为这个群组之意。常常用在变更某一目录内所有的档案之情况。

范例：

```
[root@www ~]# chgrp users install.log
```

```
[root@www ~]# ls -l
```

```
-rw-r--r-- 1 root users 68495 Jun 25 08:53 install.log
```

```
[root@www ~]# chgrp testing install.log
```

```
chgrp: invalid group name `testing' <== 发生错误讯息啰~找不到这个群组名~
```

## 10、改变文件拥有者：chown

他还可以顺便直接修改群组的名称

```
[root@www ~]# chown [-R] 账号名称 档案或目录
```

```
[root@www ~]# chown [-R] 账号名称:组名 档案或目录
```

选项与参数：

-R：进行递归(recursive)的持续变更，亦即连同次目录下的所有档案都变更

范例：将 install.log 的拥有者改为 bin 这个账号：

```
[root@www ~]# chown bin install.log
```

```
[root@www ~]# ls -l
```

```
-rw-r--r-- 1 bin users 68495 Jun 25 08:53 install.log
```

范例：将 install.log 的拥有者与群组改回为 root：

```
[root@www ~]# chown root:root install.log
```

```
[root@www ~]# ls -l
```

```
-rw-r--r-- 1 root root 68495 Jun 25 08:53 install.log
```

## 11、改变文件的权限：chmod

权限的设定方法有两种， 分别可以使用数字或者是符号来进行权限的变更。

—数字类型改变档案权限：

```
[root@www ~]# chmod [-R] xyz 档案或目录
```

选项与参数：

xyz：就是刚刚提到的数字类型的权限属性，为 rwx 属性数值的相加。

-R：进行递归(recursive)的持续变更，亦即连同次目录下的所有档案都会变更

```
[root@www ~]# ls -al .bashrc
```

```
-rw-r--r-- 1 root root 395 Jul  4 11:45 .bashrc
```

```
[root@www ~]# chmod 777 .bashrc
```

```
[root@www ~]# ls -al .bashrc
```

```
-rwxrwxrwx 1 root root 395 Jul  4 11:45 .bashrc
```

—符号类型改变档案权限：

chmod	u	+(加入)	r	档案或目录
	g	-(除去)	w	
	o	=(设定)	x	
	a			

```
[root@www ~]# chmod u=rwx,go=rx .bashrc
# 注意喔！那个 u=rwx,go=rx 是连在一起的，中间并没有任何空格符！
[root@www ~]# ls -al .bashrc
-rwxr-xr-x 1 root root 395 Jul 4 11:45 .bashrc
```

```
[root@www ~]# ls -al .bashrc
-rwxr-xr-x 1 root root 395 Jul 4 11:45 .bashrc
[root@www ~]# chmod a+w .bashrc
[root@www ~]# ls -al .bashrc
-rwxrwxrwx 1 root root 395 Jul 4 11:45 .bashrc
```

```
[root@www ~]# chmod a-x .bashrc
[root@www ~]# ls -al .bashrc
-rw-rw-rw- 1 root root 395 Jul 4 11:45 .bashrc
```

## 12、查看版本信息等

```
[root@www ~]# uname -r
2.6.18-128.el5 <==可以察看实际的核心版本

[root@www ~]# lsb_release -a
LSB Version: :core-3.1-amd64:core-3.1-ia32:core-3.1-noarch:graphics-
3.1-amd64:
graphics-3.1-ia32:graphics-3.1-noarch <==LSB 的版本
Distributor ID: CentOS
Description: CentOS release 5.3 (Final) <==distribution 的版本
Release: 5.3
Codename: Final
```

13、变换目录：cd



```
[root@www ~]# cd [相对路径或绝对路径]
```

# 最重要的就是目录的绝对路径与相对路径，还有一些特殊目录的符号啰！

```
[root@www ~]# cd ~vbird
```

# 代表去到 vbird 这个用户的家目录，亦即 /home/vbird

```
[root@www vbird]# cd ~
```

# 表示回到自己的家目录，亦即是 /root 这个目录

```
[root@www ~]# cd
```

# 没有加上任何路径，也还是代表回到自己家目录的意思喔！

```
[root@www ~]# cd ..
```

# 表示去到目前的上层目录，亦即是 /root 的上层目录的意思；

```
[root@www /]# cd -
```

# 表示回到刚刚的那个目录，也就是 /root 啰~

```
[root@www ~]# cd /var/spool/mail
```

# 这个就是绝对路径的写法！直接指定要去的完整路径名称！

```
[root@www mail]# cd ../mqueue
```

# 这个是相对路径的写法，我们由/var/spool/mail 去到/var/spool/mqueue 就这样写！

14、显示当前所在目录: `pwd`

范例：单纯显示出目前的工作目录：

```
[root@www ~]# pwd
```

/root <== 显示出目录啦~

范例：显示出实际的工作目录，而非链接文件本身的目录名而已

```
[root@www ~]# cd /var/mail <==注意，/var/mail 是一个连结档
```

```
[root@www mail]# pwd
```

/var/mail <==列出目前的工作目录

```
[root@www mail]# pwd -P
```

/var/spool/mail <==怎么回事？有没有加 -P 差很多~

```
[root@www mail]# ls -ld /var/mail
```

```
lrwxrwxrwx 1 root root 10 Sep  4 17:54 /var/mail -> spool/mail
```

# 看到这里应该知道为啥了吧？因为 /var/mail 是连结档，连结到

/var/spool/mail

# 所以，加上 pwd -P 的选项后，会不以连结文件的数据显示，而是显示正确的完整路径啊！

15、建立新目录: `mkdir`

```
[root@www ~]# mkdir [-mp] 目录名称
```

选项与参数：

-m ：配置文件案的权限喔！直接设定，不需要看预设权限 (umask) 的脸色~

-p ：帮助你直接将所需要的目录(包含上层目录)递归建立起来！

范例：请到/tmp 底下尝试建立数个新目录看看：

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# mkdir test    <==建立一名为 test 的新目录
```

```
[root@www tmp]# mkdir test1/test2/test3/test4
```

```
mkdir: cannot create directory `test1/test2/test3/test4':
```

```
No such file or directory    <== 没办法直接建立此目录啊！
```

```
[root@www tmp]# mkdir -p test1/test2/test3/test4
```

# 加了这个 -p 的选项，可以自行帮你建立多层目录！

范例：建立权限为 rwx--x--x 的目录

```
[root@www tmp]# mkdir -m 711 test2
```

```
[root@www tmp]# ls -l
```

```
drwxr-xr-x  3 root  root 4096 Jul 18 12:50 test
```

```
drwxr-xr-x  3 root  root 4096 Jul 18 12:53 test1
```

```
drwx--x--x 2 root root 4096 Jul 18 12:54 test2
```

# 仔细看上面的权限部分，如果没有加上 -m 来强制设定属性，系统会使用默认属性。

# 那么你的默认属性为何？这要透过底下介绍的 [umask](#) 才能了解喔！ ^\_^

不建议常用 -p 这个选项，因为担心如果你打错字，那么目录名称就回变得乱七八糟的

## 16、删除『空』的目录：rmdir

```
[root@www ~]# rmdir [-p] 目录名称
```

选项与参数：

-p : 连同上层『空的』目录也一起删除

范例：将于 mkdir 范例中建立的目录(/tmp 底下)删除掉！

```
[root@www tmp]# ls -l <==看看有多少目录存在？
```

```
drwxr-xr-x 3 root root 4096 Jul 18 12:50 test
```

```
drwxr-xr-x 3 root root 4096 Jul 18 12:53 test1
```

```
drwx--x--x 2 root root 4096 Jul 18 12:54 test2
```

```
[root@www tmp]# rmdir test <==可直接删除掉，没问题
```

```
[root@www tmp]# rmdir test1 <==因为尚有内容，所以无法删除！
```

```
rmdir: `test1': Directory not empty
```

```
[root@www tmp]# rmdir -p test1/test2/test3/test4
```

```
[root@www tmp]# ls -l <==您看看，底下的输出中 test 与 test1 不见了！
```

```
drwx--x--x 2 root root 4096 Jul 18 12:54 test2
```

# 瞧！利用 -p 这个选项，立刻就可以将 test1/test2/test3/test4 一次删除～

# 不过要注意的是，这个 rmdir 仅能『删除空的目录』喔！

17、档案与目录的显示：1s



[root@www ~]# ls [-aAdfFhilnrRSt] 目录名称

[root@www ~]# ls [--color={never,auto,always}] 目录名称

[root@www ~]# ls [--full-time] 目录名称

选项与参数：

-a : 全部的档案，连同隐藏档( 开头为 . 的档案) 一起列出来(常用)

-A : 全部的档案，连同隐藏档，但不包括 . 与 .. 这两个目录

-d : 仅列出目录本身，而不是列出目录内的档案数据(常用)

-f : 直接列出结果，而不进行排序 (ls 预设会以档名排序！)

-F : 根据档案、目录等信息，给予附加数据结构，例如：

\*:代表可执行文件；/:代表目录；=:代表 socket 档案；|:代表 FIFO 档案；

-h : 将档案容量以人类较易读的方式(例如 GB, KB 等等)列出来；

-i : 列出 inode 号码，inode 的意义下一章将会介绍；

-l : 长数据串行出，包含档案的属性与权限等等数据；(常用)

-n : 列出 UID 与 GID 而非使用者与群组的名称 (UID 与 GID 会在账号管理提到！)

-r : 将排序结果反向输出，例如：原本档名由小到大，反向则为由大到小；

-R : 连同子目录内容一起列出来，等于该目录下的所有档案都会显示出来；

-S : 以档案容量大小排序，而不是用档名排序；

-t : 依时间排序，而不是用档名。

--color=never : 不要依据档案特性给予颜色显示；

--color=always : 显示颜色

--color=auto : 让系统自行依据设定来判断是否给予颜色

--full-time : 以完整时间模式 (包含年、月、日、时、分) 输出

--time={atime,ctime} : 输出 access 时间或改变权限属性时间 (ctime)  
而非内容变更时间 (modification time)

## 18、复制档案或目录：cp



```
[root@www ~]# cp [-adfilprsu] 来源文件(source) 目标文件(destination)
```

```
[root@www ~]# cp [options] source1 source2 source3 .... directory
```

选项与参数：

-a ：相当于 -pdr 的意思，至于 pdr 请参考下列说明；(常用)

-d ：若来源文件为链接文件的属性(link file)，则复制链接文件属性而非档案本身；

-f ：为强制(force)的意思，若目标档案已经存在且无法开启，则移除后再尝试一次；

-i ：若目标文件(destination)已经存在时，在覆盖时会先询问动作的进行(常用)

-l ：进行硬式连结(hard link)的连结档建立，而非复制档案本身；

-p ：连同档案的属性一起复制过去，而非使用默认属性(备份常用)；

-r ：递归持续复制，用于目录的复制行为；(常用)

-s ：复制成为符号链接文件 (symbolic link)，亦即『快捷方式』档案；

-u ：若 destination 比 source 旧才更新 destination ！

最后需要注意的，如果来源档有两个以上，则最后一个目的文件一定要是『目录』才行！

范例一：用 root 身份，将家目录下的 .bashrc 复制到 /tmp 下，并更名为 bashrc

```
[root@www ~]# cp ~/.bashrc /tmp/bashrc
```

```
[root@www ~]# cp -i ~/.bashrc /tmp/bashrc
```

cp: overwrite `/tmp/bashrc'? n <==n 不覆盖，y 为覆盖

# 重复作两次动作，由于 /tmp 底下已经存在 bashrc 了，加上 -i 选项后，

# 则在覆盖前会询问使用者是否确定！可以按下 n 或者 y 来二次确认呢！

范例二：变换目录到/tmp，并将/var/log/wtmp 复制到/tmp 且观察属性：

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# cp /var/log/wtmp . <==想要复制到当前目录，最后的 .  
不要忘记
```

```
[root@www tmp]# ls -l /var/log/wtmp wtmp
```

```
-rw-rw-r-- 1 root utmp 96384 Sep 24 11:54 /var/log/wtmp
-rw-r--r-- 1 root root 96384 Sep 24 14:06 wtmp
```

# 注意上面的特殊字体，在不加任何选项的情况下，档案的某些属性/权限会改变；

# 这是个很重要的特性！要注意喔！还有，连档案建立的时间也不一样了！

# 那如果你想要将档案的所有特性都一起复制过来该怎办？可以加上 -a 喔！如下所示：

```
[root@www tmp]# cp -a /var/log/wtmp wtmp_2
[root@www tmp]# ls -l /var/log/wtmp wtmp_2
-rw-rw-r-- 1 root utmp 96384 Sep 24 11:54 /var/log/wtmp
-rw-rw-r-- 1 root utmp 96384 Sep 24 11:54 wtmp_2
```

# 瞭了吧！整个资料特性完全一模一样！真是不赖~这就是 -a 的特性！

范例三：复制 /etc/ 这个目录下的所有内容到 /tmp 底下

```
[root@www tmp]# cp /etc/ /tmp
```

cp: omitting directory `/etc' <== 如果是目录则不能直接复制，要加上 -r 的选项

```
[root@www tmp]# cp -r /etc/ /tmp
```

# 还是要再次的强调喔！-r 是可以复制目录，但是，档案与目录的权限可能会被改变

# 所以，也可以利用『cp -a /etc /tmp』来下达指令喔！尤其是在备份的情况下！

范例四：将范例一复制的 bashrc 建立一个连结档 (symbolic link)

```
[root@www tmp]# ls -l bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 24 14:02 bashrc <==先观察一下档案情况
```

```
[root@www tmp]# cp -s bashrc bashrc_slink
```

```
[root@www tmp]# cp -l bashrc bashrc_hlink
```

```
[root@www tmp]# ls -l bashrc*
```

```
-rw-r--r-- 2 root root 176 Sep 24 14:02 bashrc <==与源文件不太一样了！
```

```
-rw-r--r-- 2 root root 176 Sep 24 14:02 bashrc_hlink
```

```
lrwxrwxrwx 1 root root  6 Sep 24 14:20 bashrc_slink -> bashrc
```

范例五：若 ~/.bashrc 比 /tmp/bashrc 新才复制过来

```
[root@www tmp]# cp -u ~/.bashrc /tmp/bashrc
```

# 这个 -u 的特性，是在目标档案与来源档案有差异时，才会复制的。

# 所以，比较常被用于『备份』的工作当中喔！ ^\_^



范例六：将范例四造成的 bashrc\_slink 复制成为 bashrc\_slink\_1 与 bashrc\_slink\_2

```
[root@www tmp]# cp bashrc_slink bashrc_slink_1
```

```
[root@www tmp]# cp -d bashrc_slink bashrc_slink_2
```

```
[root@www tmp]# ls -l bashrc bashrc_slink*
```

```
-rw-r--r-- 2 root root 176 Sep 24 14:02 bashrc
```

```
lrwxrwxrwx 1 root root 6 Sep 24 14:20 bashrc_slink -> bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 24 14:32 bashrc_slink_1    <==与源文件相同
```

```
lrwxrwxrwx 1 root root 6 Sep 24 14:33 bashrc_slink_2 -> bashrc <==是连结档！
```

# 这个例子也是很有趣喔！原本复制的是连结档，但是却将连结档的实际档案复制过来了

# 也就是说，如果没有加上任何选项时，cp 复制的是源文件，而非链接文件的属性！

# 若要复制链接文件的属性，就得要使用 -d 的选项了！如 bashrc\_slink\_2 所示。

范例七：将家目录的 .bashrc 及 .bash\_history 复制到 /tmp 底下

```
[root@www tmp]# cp ~/.bashrc ~/.bash_history /tmp
```

# 可以将多个数据一次复制到同一个目录去！最后面一定是目录！

## 19、移除档案或目录：rm

```
[root@www ~]# rm [-fir] 档案或目录
```

选项与参数：

-f ：就是 force 的意思，忽略不存在的档案，不会出现警告讯息；

-i ：互动模式，在删除前会询问使用者是否动作

-r ：递归删除啊！最常用在目录的删除了！这是非常危险的选项！！

范例一：将刚刚在 cp 的范例中建立的 bashrc 删除掉！

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# rm -i bashrc
```

```
rm: remove regular file `bashrc'? y
```

# 如果加上 -i 的选项就会主动询问喔，避免你删除到错误的档名！

范例二：透过通配符\*的帮忙，将/tmp 底下开头为 bashrc 的档名通通删除：

```
[root@www tmp]# rm -i bashrc*
```

# 注意那个星号，代表的是 0 到无穷多个任意字符喔！很好用的东西！

范例三：将 cp 范例中所建立的 /tmp/etc/ 这个目录删除掉！

```
[root@www tmp]# rmdir /tmp/etc
```

rmdir: etc: Directory not empty <== 删不掉啊！因为这不是空的目录！

```
[root@www tmp]# rm -r /tmp/etc
```

rm: descend into directory `/tmp/etc'? y

....(中间省略)....

# 因为身份是 root，预设已经加入了 -i 的选项，所以你要一直按 y 才会删除！

# 如果不想要继续按 y，可以按下『[ctrl]-c』来结束 rm 的工作。

# 这是一种保护的動作，如果确定要删除掉此目录而不要询问，可以这样做：

```
[root@www tmp]# \rm -r /tmp/etc
```

# 在指令前加上反斜杠，可以忽略掉 alias 的指定选项喔！至于 alias 我们在 bash 再谈！

范例四：删除一个带有 - 开头的档案

```
[root@www tmp]# touch ./-aaa- <==touch 这个指令可以建立空档案！
```

```
[root@www tmp]# ls -l
```

```
-rw-r--r-- 1 root root    0 Sep 24 15:03 -aaa- <==档案大小为 0，所以是空档案
```

```
[root@www tmp]# rm -aaa-
```

```
Try `rm --help' for more information. <== 因为 "-" 是选项嘛！所以系统误判了！
```

```
[root@www tmp]# rm ./-aaa-
```

20、移动档案与目录，或更名：mv

```
[root@www ~]# mv [-fiu] source destination
```

```
[root@www ~]# mv [options] source1 source2 source3 .... directory
```

选项与参数：

-f : force 强制的意思，如果目标档案已经存在，不会询问而直接覆盖；

-i : 若目标档案 (destination) 已经存在时，就会询问是否覆盖！

-u : 若目标档案已经存在，且 source 比较新，才会更新 (update)

范例一：复制一档案，建立一目录，将档案移动到目录中

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# cp ~/.bashrc bashrc
```

```
[root@www tmp]# mkdir mvtest
```

```
[root@www tmp]# mv bashrc mvtest
```

# 将某个档案移动到某个目录去，就是这样做！

范例二：将刚刚的目录名称更名为 mvtest2

```
[root@www tmp]# mv mvtest mvtest2 <== 这样就更名了！简单~
```

# 其实在 Linux 底下还有个有趣的指令，名称为 rename，

# 该指令专职进行多个档名的同时更名，并非针对单一档名变更，与 mv 不同。

请 man rename。

范例三：再建立两个档案，再全部移动到 /tmp/mvtest2 当中

```
[root@www tmp]# cp ~/.bashrc bashrc1
```

```
[root@www tmp]# cp ~/.bashrc bashrc2
```

```
[root@www tmp]# mv bashrc1 bashrc2 mvtest2
```

# 注意到这边，如果有多个来源档案或目录，则最后一个目标文件一定是『目录！』

# 意思是说，将所有数据移动到该目录的意思！

## 21、取得路径的文件名与目录名：basename, dirname

```
[root@www ~]# basename /etc/sysconfig/network
network      <== 很简单！就取得最后的档名~
[root@www ~]# dirname /etc/sysconfig/network
/etc/sysconfig <== 取得的变成目录名了！
```

```
[root@www ~]# basename /etc/sysconfig/network
network      <== 很简单！就取得最后的档名~
[root@www ~]# dirname /etc/sysconfig/network
/etc/sysconfig <== 取得的变成目录名了！
```

## 22、由第一行开始显示档案内容：cat



```
[root@www ~]# cat [-AbEnTv]
```

选项与参数：

- A : 相当于 -vET 的整合选项，可列出一些特殊字符而不是空白而已；
- b : 列出行号，仅针对非空白行做行号显示，空白行不标行号！
- E : 将结尾的断行字符 \$ 显示出来；
- n : 打印出行号，连同空白行也会有行号，与 -b 的选项不同；
- T : 将 [tab] 按键以 ^I 显示出来；
- v : 列出一些看不出来的特殊字符

范例一：检阅 /etc/issue 这个档案的内容

```
[root@www ~]# cat /etc/issue
```

```
CentOS release 5.3 (Final)
```

```
Kernel \r on an \m
```

范例二：承上题，如果还要加印行号呢？

```
[root@www ~]# cat -n /etc/issue
```

- 1 CentOS release 5.3 (Final)
- 2 Kernel \r on an \m
- 3

# 看到了吧！可以印出行号呢！这对于大档案要找某个特定的行时，有点用处！

# 如果不想编排空白行的行号，可以使用『cat -b /etc/issue』，自己测试看看：

范例三：将 /etc/xinetd.conf 的内容完整的显示出来(包含特殊字符)

```
[root@www ~]# cat -A /etc/xinetd.conf
```

```
#$
```

```
....(中间省略)....
```

23、从最后一行开始显示：tac（可以看出 tac 是 cat 的倒着写）

```
[root@www ~]# tac /etc/issue
```

```
Kernel \r on an \m
```

```
CentOS release 5.3 (Final)
```

```
# 嘿嘿！与刚刚上面的范例一比较，是由最后一行先显示喔！
```

24、显示的时候，顺道输出行号：nl

```
[root@www ~]# nl [-bnw] 档案
```

选项与参数：

-b ：指定行号指定的方式，主要有两种：

-b a ：表示不论是否为空行，也同样列出行号(类似 cat -n)；

-b t ：如果有空行，空的那一行不要列出行号(默认值)；

-n ：列出行号表示的方法，主要有三种：

-n ln ：行号在屏幕的最左方显示；

-n rn ：行号在自己字段的最右方显示，且不加 0 ；

-n rz ：行号在自己字段的最右方显示，且加 0 ；

-w ：行号字段的占用的位数。

范例一：用 nl 列出 /etc/issue 的内容

```
[root@www ~]# nl /etc/issue
```

```
1 CentOS release 5.3 (Final)
```

```
2 Kernel \r on an \m
```

# 注意看，这个档案其实有三行，第三行为空白(没有任何字符)，  
# 因为他是空白行，所以 nl 不会加上行号喔！如果确定要加上行号，可以这样做：

```
[root@www ~]# nl -b a /etc/issue
```

```
1 CentOS release 5.3 (Final)
```

```
2 Kernel \r on an \m
```

```
3
```

# 呵呵！行号加上来啰~那么如果要让行号前面自动补上 0 呢？可这样

```
[root@www ~]# nl -b a -n rz /etc/issue
```

```
000001 CentOS release 5.3 (Final)
```

```
000002 Kernel \r on an \m
```

```
000003
```

# 嘿嘿！自动在自己字段的地方补上 0 了~预设字段是六位数，如果想要改成 3 位数？

```
[root@www ~]# nl -b a -n rz -w 3 /etc/issue
```

```
001  CentOS release 5.3 (Final)
```

```
002  Kernel \r on an \m
```

```
003
```

```
# 变成仅有 3 位数啰~
```

25、一页一页的显示档案内容： more

```
[root@www ~]# more /etc/man.config
```

```
#
```

```
# Generated automatically from man.conf.in by the
```

```
# configure script.
```

```
#
```

```
# man.conf from man-1.6d
```

```
....(中间省略)....
```

```
--More--(28%) <== 重点在这一行喔！你的光标也会在这里等待你的指令
```

- 空格键 (space) : 代表向下翻一页；
- Enter : 代表向下翻『一行』；
- /字符串 : 代表在这个显示的内容当中，向下搜寻『字符串』这个关键词；
- :f : 立刻显示出文件名以及目前显示的行数；
- q : 代表立刻离开 more，不再显示该档案内容。
- b 或 [ctrl]-b : 代表往回翻页，不过这动作只对档案有用，对管线无用。

26、与 more 类似，但是比 more 更好的是，他可以往前翻页：less



```
[root@www ~]# less /etc/man.config
#
# Generated automatically from man.conf.in by the
# configure script.
#
# man.conf from man-1.6d
....(中间省略)....
: <== 这里可以等待你输入指令！
```

- 空格键 : 向下翻动一页；
- [pagedown] : 向下翻动一页；
- [pageup] : 向上翻动一页；
- /字符串 : 向下搜寻『字符串』的功能；
- ?字符串 : 向上搜寻『字符串』的功能；
- n : 重复前一个搜寻 (与 / 或 ? 有关！)
- N : 反向的重复前一个搜寻 (与 / 或 ? 有关！)
- q : 离开 less 这个程序；

## 27、只看头几行：head

```
[root@www ~]# head [-n number] 档案
```

选项与参数：

-n ：后面接数字，代表显示几行的意思

```
[root@www ~]# head /etc/man.config
```

# 默认的情况中，显示前面十行！若要显示前 20 行，就得要这样：

```
[root@www ~]# head -n 20 /etc/man.config
```

范例：如果后面 100 行的数据都不打印，只打印/etc/man.config 的前面几行，该如何是好？

```
[root@www ~]# head -n -100 /etc/man.config
```

28、只看尾几行: tail

```
[root@www ~]# tail [-n number] 档案
```

选项与参数：

-n ：后面接数字，代表显示几行的意思

-f ：表示持续侦测后面所接的档名，要等到按下[ctrl]-c 才会结束 tail 的侦测

```
[root@www ~]# tail /etc/man.config
```

# 默认的情况中，显示最后的十行！若要显示最后的 20 行，就得要这样：

```
[root@www ~]# tail -n 20 /etc/man.config
```

范例一：如果不知道/etc/man.config 有几行，却只想列出 100 行以后的数据时？

```
[root@www ~]# tail -n +100 /etc/man.config
```

范例二：持续侦测/var/log/messages 的内容

```
[root@www ~]# tail -f /var/log/messages
```

<==要等到输入[ctrl]-c 之后才会离开 tail 这个指令的侦测！

29、以二进制的放置读取档案内容：od

```
[root@www ~]# od [-t TYPE] 档案
```

选项或参数：

-t : 后面可以接各种『类型 (TYPE)』的输出，例如：

a : 利用默认的字符来输出；

c : 使用 ASCII 字符来输出

d[size] : 利用十进制(decimal)来输出数据，每个整数占用 size bytes ；

f[size] : 利用浮点数(floating)来输出数据，每个数占用 size bytes ；

o[size] : 利用八进制(octal)来输出数据，每个整数占用 size bytes ；

x[size] : 利用十六进制(hexadecimal)来输出数据，每个整数占用 size bytes ；

范例一：请将/usr/bin/passwd 的内容使用 ASCII 方式来展现！

```
[root@www ~]# od -t c /usr/bin/passwd
```

```
0000000 177  E  L  F 001 001 001  \0 \0 \0 \0 \0 \0 \0 \0 \0
```

```
0000020 002  \0 003  \0 001  \0 \0 \0 260 225 004  \b  4  \0 \0 \0
```

```
0000040 020  E  \0 \0 \0 \0 \0 \0  4  \0  \0 \a \0  (  \0
```

```
0000060 035  \0 034  \0 006  \0 \0 \0  4  \0 \0 \0  4 200 004  \b
```

```
0000100  4 200 004  \b 340  \0 \0 \0 340  \0 \0 \0 005  \0 \0 \0
```

.....(后面省略)....

# 最左边第一栏是以 8 进位来表示 bytes 数。以上面范例来说，第二栏

0000020 代表开头是

# 第 16 个 bytes (2x8) 的内容之意。

范例二：请将/etc/issue 这个档案的内容以 8 进位列出储存值与 ASCII 的对照表

```
[root@www ~]# od -t oCc /etc/issue
```

```
0000000 103 145 156 164 117 123 040 162 145 154 145 141 163 145 040
065
```

```
    C e n t O S   r e l e a s e   5
```

```
0000020 056 062 040 050 106 151 156 141 154 051 012 113 145 162 156
145
```

```
    . 2   ( F i n a l ) \n K e r n e
```

```
0000040 154 040 134 162 040 157 156 040 141 156 040 134 155 012 012
    l   \ r   o n   a n   \ m \n \n
```

```
0000057
```

# 如上所示，可以发现每个字符可以对应到的数值为何！

# 例如 e 对应的记录数值为 145，转成十进制： $1 \times 8^2 + 4 \times 8 + 5 = 101$ 。

30、修改档案时间或新建档案: touch



```
[root@www ~]# touch [-acdm] 档案
```

选项与参数：

-a : 仅修订 access time ;

-c : 仅修改档案的时间，若该档案不存在则不建立新档案；

-d : 后面可以接欲修订的日期而不用目前的日期，也可以使用 --date="日期或时间"

-m : 仅修改 mtime ;

-t : 后面可以接欲修订的时间而不用目前的时间，格式为[YYMMDDhhmm]

范例一：新建一个空的档案并观察时间

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# touch testtouch
```

```
[root@www tmp]# ls -l testtouch
```

```
-rw-r--r-- 1 root root 0 Sep 25 21:09 testtouch
```

# 注意到，这个档案的大小是 0 呢！在预设的状态下，如果 touch 后面有接档案，

# 则该档案的三个时间 (atime/ctime/mtime) 都会更新为目前的时间。若该档案不存在，

# 则会主动的建立一个新的空的档案呢！例如上面这个例子！

范例二：将 ~/.bashrc 复制成为 bashrc，假设复制完全的属性，检查其日期

```
[root@www tmp]# cp -a ~/.bashrc bashrc
```

```
[root@www tmp]# ll bashrc; ll --time=atime bashrc; ll --time=ctime  
bashrc
```

```
-rw-r--r-- 1 root root 176 Jan  6 2007 bashrc <==这是 mtime
```

```
-rw-r--r-- 1 root root 176 Sep 25 21:11 bashrc <==这是 atime
```

```
-rw-r--r-- 1 root root 176 Sep 25 21:12 bashrc <==这是 ctime
```

范例三：修改案例二的 bashrc 档案，将日期调整为两天前

```
[root@www tmp]# touch -d "2 days ago" bashrc
```

```
[root@www tmp]# ll bashrc; ll --time=atime bashrc; ll --time=ctime  
bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 23 21:23 bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 23 21:23 bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 25 21:23 bashrc
```

# 跟上个范例比较看看，本来是 25 日的变成了 23 日了 (atime/mtime) ~

# 不过，ctime 并没有跟着改变喔！

范例四：将上个范例的 bashrc 日期改为 2007/09/15 2:02

```
[root@www tmp]# touch -t 0709150202 bashrc
```

```
[root@www tmp]# ll bashrc; ll --time=atime bashrc; ll --time=ctime  
bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 15 2007 bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 15 2007 bashrc
```

```
-rw-r--r-- 1 root root 176 Sep 25 21:25 bashrc
```

# 注意看看，日期在 atime 与 mtime 都改变了，但是 ctime 则是记录目前的时间！

31、档案预设权限：umask

```
[root@www ~]# umask
0022      <==与一般权限有关的是后面三个数字！
[root@www ~]# umask -S
u=rwx,g=rx,o=rx
```

```
[root@www ~]# umask
0022
[root@www ~]# touch test1
[root@www ~]# mkdir test2
[root@www ~]# ll
-rw-r--r-- 1 root root  0 Sep 27 00:25 test1
drwxr-xr-x 2 root root 4096 Sep 27 00:25 test2
```

```
[root@www ~]# umask 002
[root@www ~]# touch test3
[root@www ~]# mkdir test4
[root@www ~]# ll
-rw-rw-r-- 1 root root  0 Sep 27 00:36 test3
drwxrwxr-x 2 root root 4096 Sep 27 00:36 test4
```

32、配置文件档案隐藏属性: `chattr`

[root@www ~]# chattr [+ -=][ASacdistu] 档案或目录名称

选项与参数：

+ ：增加某一个特殊参数，其他原本存在参数则不动。

- ：移除某一个特殊参数，其他原本存在参数则不动。

= ：设定一定，且仅有后面接的参数

A ：当设定了 A 这个属性时，若你有存取此档案(或目录)时，他的访问时间  
atime

将不会被修改，可避免 I/O 较慢的机器过度的存取磁盘。这对速度较慢的计算机有帮助

S ：一般档案是异步写入磁盘的(原理请参考第五章 sync 的说明)，如果加上 S 这个

属性时，当你进行任何档案的修改，该更动会『同步』写入磁盘中。

a ：当设定 a 之后，这个档案将只能增加数据，而不能删除也不能修改数据，只有 root

才能设定这个属性。

c : 这个属性设定之后, 将会自动的将此档案『压缩』, 在读取的时候将会自动解压缩,

但是在储存的时候, 将会先进行压缩后再储存(看来对于大档案似乎蛮有用的!)

d : 当 dump 程序被执行的时候, 设定 d 属性将可使该档案(或目录)不会被 dump 备份

i : 这个 i 可就很厉害了! 他可以让一个档案『不能被删除、改名、设定连结也无法

写入或新增资料!』对于系统安全性有相当大的帮助! 只有 root 能设定此属性

s : 当档案设定了 s 属性时, 如果这个档案被删除, 他将会被完全的移除出这个硬盘

空间, 所以如果误删了, 完全无法救回来了喔!

u :与 s 相反的,当使用 u 来配置文件案时,如果该档案被删除了,则数据内容其实还

存在磁盘中,可以使用来救援该档案喔!

注意:属性设定常见的是 a 与 i 的设定值,而且很多设定值必须要身为 root 才能设定

范例:请尝试到/tmp 底下建立档案,并加入 i 的参数,尝试删除看看。

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# touch attrtest <==建立一个空档案
```

```
[root@www tmp]# chattr +i attrtest <==给予 i 的属性
```

```
[root@www tmp]# rm attrtest <==尝试删除看看
```

```
rm: remove write-protected regular empty file `attrtest'? y
```

```
rm: cannot remove `attrtest': Operation not permitted <==操作不许可
```

# 看到了吗?呼呼!连 root 也没有办法将这个档案删除呢!赶紧解除设定!

范例:请将该档案的 i 属性取消!

```
[root@www tmp]# chattr -i attrtest
```



### 33、显示档案隐藏属性：lsattr

```
[root@www ~]# lsattr [-adR] 档案或目录
```

选项与参数：

-a : 将隐藏文件的属性也秀出来；

-d : 如果接的是目录，仅列出目录本身的属性而非目录内的文件名；

-R : 连同子目录的数据也一并列出来！

```
[root@www tmp]# chattr +aij attrtest
```

```
[root@www tmp]# lsattr attrtest
```

```
----ia---j--- attrtest
```

## 34、观察文件类型：file

```
[root@www ~]# file ~/.bashrc
/root/.bashrc: ASCII text <==告诉我们是 ASCII 的纯文本档啊！
[root@www ~]# file /usr/bin/passwd
/usr/bin/passwd: setuid ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared libs), for
GNU/Linux 2.6.9, stripped
# 执行文件的数据可就多的不得了！包括这个档案的 suid 权限、兼容于 Intel
386
# 等级的硬件平台、使用的是 Linux 核心 2.6.9 的动态函数库链接等等。
[root@www ~]# file /var/lib/mlocate/mlocate.db
/var/lib/mlocate/mlocate.db: data <== 这是 data 档案！
```

35、寻找【执行挡】: which

```
[root@www ~]# which [-a] command
```

选项或参数：

-a ：将所有由 PATH 目录中可以找到的指令均列出，而不止第一个被找到的指令名称

范例一：分别用 root 与一般账号搜寻 ifconfig 这个指令的完整文件名

```
[root@www ~]# which ifconfig
```

```
/sbin/ifconfig      <==用 root 可以找到正确的执行档名喔！
```

```
[root@www ~]# su - vbird <==切换身份成为 vbird 去！
```

```
[vbird@www ~]$ which ifconfig
```

```
/usr/bin/which: no ifconfig in
```

```
(/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin
```

```
:/home/vbird/bin)      <==见鬼了！竟然一般身份账号找不到！
```

# 因为 which 是根据用户所设定的 PATH 变量内的目录去搜寻可执行文件的！

所以，

# 不同的 PATH 设定内容所找到的指令当然不一样啦！因为 /sbin 不在 vbird 的

# PATH 中，找不到也是理所当然的啊！瞭乎？

```
[vbird@www ~]$ exit      <==记得将身份切换回原本的 root
```

范例二：用 which 去找出 which 的档名为何？

```
[root@www ~]# which which
```

```
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot '  
/usr/bin/which
```

# 竟然会有两个 which，其中一个 alias 这玩意儿呢！那是啥？

# 那就是所谓的『命令别名』，意思是输入 which 会等于后面接的那串指令啦！

# 更多的数据我们会在 bash 章节中再来谈的！

范例三：请找出 cd 这个指令的完整文件名

```
[root@www ~]# which cd
```

```
/usr/bin/which: no cd in
```

```
(/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin
```

```
:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin)
```

# 瞎密？怎么可能没有 cd，我明明就能够用 root 执行 cd 的啊！

36、寻找特定档案: whereis

范例一：请用不同的身份找出 ifconfig 这个档名

```
[root@www ~]# whereis ifconfig
```

```
ifconfig: /sbin/ifconfig /usr/share/man/man8/ifconfig.8.gz
```

```
[root@www ~]# su - vbird      <==切换身份成为 vbird
```

```
[vbird@www ~]$ whereis ifconfig <==找到同样的结果喔！
```

```
ifconfig: /sbin/ifconfig /usr/share/man/man8/ifconfig.8.gz
```

```
[vbird@www ~]$ exit          <==回归身份成为 root 去！
```

# 注意看，明明 which 一般使用者找不到的 ifconfig 却可以让 whereis 找到！

# 这是因为系统真的有 ifconfig 这个『档案』，但是使用者的 PATH 并没有加入 /sbin

# 所以，未来你找不到某些指令时，先用档案搜寻指令找找看再说！

范例二：只找出跟 passwd 有关的『说明文件』档名(man page)

```
[root@www ~]# whereis -m passwd
```

```
passwd: /usr/share/man/man1/passwd.1.gz
```

```
/usr/share/man/man5/passwd.5.gz
```

37、寻找特定档案: locate



```
[root@www ~]# locate [-ir] keyword
```

选项与参数：

-i ：忽略大小写的差异；

-r ：后面可接正规表示法的显示方式

范例一：找出系统中所有与 passwd 相关的档名

```
[root@www ~]# locate passwd
```

```
/etc/passwd
```

```
/etc/passwd-
```

```
/etc/news/passwd.nntp
```

```
/etc/pam.d/passwd
```

```
....(底下省略)....
```

38、寻找特定档案：find

```
[root@www ~]# find [PATH] [option] [action]
```

选项与参数：

1. 与时间有关的选项：共有 `-atime`, `-ctime` 与 `-mtime`，以 `-mtime` 说明  
`-mtime n`：n 为数字，意义为在 n 天之前的『一天之内』被更动过内容的档案；  
`-mtime +n`：列出在 n 天之前(不含 n 天本身)被更动过内容的档案档名；  
`-mtime -n`：列出在 n 天之内(含 n 天本身)被更动过内容的档案档名。  
`-newer file`：file 为一个存在的档案，列出比 file 还要新的档案档名

范例一：将过去系统上面 24 小时内有更动过内容 (mtime) 的档案列出

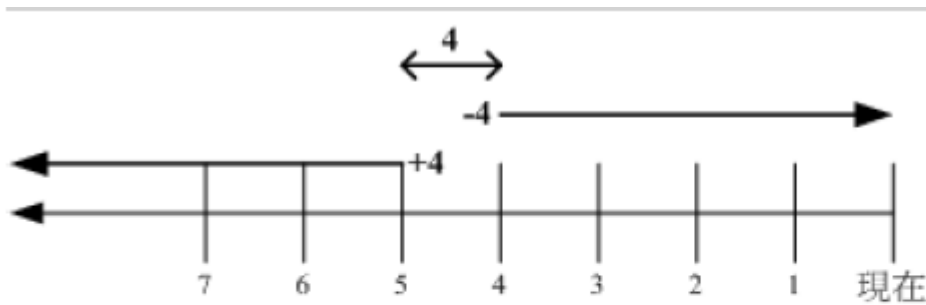
```
[root@www ~]# find / -mtime 0
```

# 那个 0 是重点！0 代表目前的时间，所以，从现在开始到 24 小时前，  
# 有变动过内容的档案都会被列出来！那如果是三天前的 24 小时内？  
# `find / -mtime 3` 有变动过的档案都被列出的意思！

范例二：寻找 /etc 底下的档案，如果档案日期比 /etc/passwd 新就列出

```
[root@www ~]# find /etc -newer /etc/passwd
```

# `-newer` 用在分辨两个档案之间的新旧关系是很有用的！



- +4 代表大于等于 5 天前的檔名 : `ex> find /var -mtime +4`
- -4 代表小于等于 4 天内的档案档名 : `ex> find /var -mtime -4`
- 4 则是代表 4-5 那一天的档案档名 : `ex> find /var -mtime 4`

### 39、压缩文件和读取压缩文件: gzip, zcat

```
[root@www ~]# gzip [-cdtv#] 檔名
```

```
[root@www ~]# zcat 檔名.gz
```

选项与参数：

-c ：将压缩的数据输出到屏幕上，可透过数据流重导向来处理；

-d ：解压缩的参数；

-t ：可以用来检验一个压缩文件的一致性~看看档案有无错误；

-v ：可以显示出原档案/压缩文件案的压缩比等信息；

-# ：压缩等级，-1 最快，但是压缩比最差、-9 最慢，但是压缩比最好！预设是-6

范例一：将 /etc/man.config 复制到 /tmp ，并且以 gzip 压缩

```
[root@www ~]# cd /tmp
```

```
[root@www tmp]# cp /etc/man.config .
```

```
[root@www tmp]# gzip -v man.config
```

```
man.config: 56.1% -- replaced with man.config.gz
```

```
[root@www tmp]# ll /etc/man.config /tmp/man*
```

```
-rw-r--r-- 1 root root 4617 Jan 6 2007 /etc/man.config
```

```
-rw-r--r-- 1 root root 2684 Nov 10 17:24 /tmp/man.config.back.Z
```

```
-rw-r--r-- 1 root root 2057 Nov 10 17:14 /tmp/man.config.gz  <==gzip 压
```

缩比较佳

范例二：由于 man.config 是文本文件，请将范例一的压缩文件的内容读出来！

```
[root@www tmp]# zcat man.config.gz
```

# 由于 man.config 这个原本的档案是文本文件，因此我们可以尝试使用 zcat 去读取！

# 此时屏幕上会显示 man.config.gz 解压缩之后的档案内容！

范例三：将范例一的档案解压缩

```
[root@www tmp]# gzip -d man.config.gz
```

# 不要使用 gunzip 这个指令，不好背！使用 gzip -d 来进行解压缩！

# 与 gzip 相反，gzip -d 会将原本的 .gz 删除，产生原本的 man.config 档案。

范例四：将范例三解开的 man.config 用最佳的压缩比压缩，并保留原本的档案

```
[root@www tmp]# gzip -9 -c man.config > man.config.gz
```

#### 40、压缩文件和读取压缩文件: bzip2, bzip2

```
[root@www ~]# bzip2 [-cdkzv#] 檔名
```

```
[root@www ~]# bzipcat 檔名.bz2
```

选项与参数：

-c ：将压缩的过程产生的数据输出到屏幕上！

-d ：解压缩的参数

-k ：保留源文件，而不会删除原始的档案喔！

-z ：压缩的参数

-v ：可以显示出原档案/压缩文件案的压缩比等信息；

-# ：与 gzip 同样的，都是在计算压缩比的参数， -9 最佳， -1 最快！

范例一：将刚刚的 /tmp/man.config 以 bzip2 压缩

```
[root@www tmp]# bzip2 -z man.config
```

# 此时 man.config 会变成 man.config.bz2 ！

范例二：将范例一的档案内容读出来！

```
[root@www tmp]# bzipcat man.config.bz2
```

# 此时屏幕上会显示 man.config.bz2 解压缩之后的档案内容！！



范例三：将范例一的档案解压缩

```
[root@www tmp]# bzip2 -d man.config.bz2
```

范例四：将范例三解开的 man.config 用最佳的压缩比压缩，并保留原本的档案

```
[root@www tmp]# bzip2 -9 -c man.config > man.config.bz2
```

41、压缩文件和读取压缩文件：tar

```
[root@www ~]# tar [-j|-z] [cv] [-f 建立的檔名] filename... <==打包与压缩
```

```
[root@www ~]# tar [-j|-z] [tv] [-f 建立的檔名] <==察看檔名
```

```
[root@www ~]# tar [-j|-z] [xv] [-f 建立的檔名] [-C 目录] <==解压缩
```

选项与参数：

-c ：建立打包档案，可搭配 -v 来察看过程中被打包的档名(filename)

-t ：察看打包档案的内容含有哪些档名，重点在察看『档名』就是了；

-x ：解打包或解压缩的功能，可以搭配 -C (大写) 在特定目录解开

特别留意的是， -c, -t, -x 不可同时出现在一串指令列中。

-j ：透过 bzip2 的支持进行压缩/解压缩：此时档名最好为 \*.tar.bz2

-z ：透过 gzip 的支持进行压缩/解压缩：此时档名最好为 \*.tar.gz

-v ：在压缩/解压缩的过程中，将正在处理的文件名显示出来！

-f filename ：-f 后面要立刻接要被处理的档名！建议 -f 单独写一个选项啰！

-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。

其他后续练习会使用到的选项介绍：

-p ：保留备份数据的原本权限与属性，常用于备份(-c)重要的配置文件

-P ：保留绝对路径，亦即允许备份数据中含有根目录存在之意；

--exclude=FILE：在压缩的过程中，不要将 FILE 打包！

其实最简单的使用 tar 就只要记忆底下的方式即可：

- 压 缩：tar -jcv -f filename.tar.bz2 要被压缩的档案或目录名称
- 查 询：tar -jtv -f filename.tar.bz2
- 解压缩：tar -jxv -f filename.tar.bz2 -C 欲解压缩的目录

```
[root@www ~]# tar -zpcv -f /root/etc.tar.gz /etc
tar: Removing leading '/' from member names <==注意这个警告讯息
/etc/
...中间省略...
/etc/esd.conf
/etc/crontab
# 由于加上 -v 这个选项，因此正在作用中的文件名就会显示在屏幕上。
# 如果你可以翻到第一页，会发现出现上面的错误讯息！底下会讲解。
# 至于 -p 的选项，重点在于『保留原本档案的权限与属性』之意。
```

```
[root@www ~]# tar -jpcv -f /root/etc.tar.bz2 /etc
# 显示的讯息会跟上面一模一样啰！
```

```
[root@www ~]# ll /root/etc*  
-rw-r--r-- 1 root root 8740252 Nov 15 23:07 /root/etc.tar.bz2  
-rw-r--r-- 1 root root 13010999 Nov 15 23:01 /root/etc.tar.gz  
[root@www ~]# du -sm /etc  
118    /etc  
# 为什么建议您使用 -j 这个选项？从上面的数值你可以知道了吧？^_^
```

```
[root@www ~]# tar -jtv -f /root/etc.tar.bz2  
....前面省略....  
-rw-r--r-- root/root 1016 2008-05-25 14:06:20 etc/dbus-1/session.conf  
-rw-r--r-- root/root 153 2007-01-07 19:20:54 etc/esd.conf  
-rw-r--r-- root/root 255 2007-01-06 21:13:33 etc/crontab
```

```
[root@www ~]# tar -jxv -f /root/etc.tar.bz2  
[root@www ~]# ll  
....(前面省略)....  
drwxr-xr-x 105 root root 12288 Nov 11 04:02 etc  
....(后面省略)....
```

```
[root@www ~]# tar -jxv -f /root/etc.tar.bz2 -C /tmp
```

```
[root@www ~]# ll /tmp
```

....(前面省略)....

```
drwxr-xr-x 105 root root 12288 Nov 11 04:02 etc
```

....(后面省略)....

# 1. 先找到我们要的档名，假设解开 shadow 档案好了：

```
[root@www ~]# tar -jtv -f /root/etc.tar.bz2 | grep 'shadow'
```

```
-r----- root/root 1230 2008-09-29 02:21:20 etc/shadow-
```

```
-r----- root/root 622 2008-09-29 02:21:20 etc/gshadow-
```

```
-r----- root/root 636 2008-09-29 02:21:25 etc/gshadow
```

```
-r----- root/root 1257 2008-09-29 02:21:25 etc/shadow <==这是我们  
要的！
```

# 先搜寻重要的档名！其中那个 grep 是『撷取』关键词的功能！我们会在第三篇说明！

# 这里您先有个概念即可！那个管线 | 配合 grep 可以撷取关键词的意思！

# 2. 将该档案解开！语法与实际作法如下：

```
[root@www ~]# tar -jxv -f 打包檔.tar.bz2 待解开档名
```

```
[root@www ~]# tar -jxv -f /root/etc.tar.bz2 etc/shadow  
etc/shadow
```

```
[root@www ~]# ll etc
```

```
total 8
```

```
-r----- 1 root root 1257 Sep 29 02:21 shadow <==哟喝！只有一个档案  
啦！
```

# 很有趣！此时只会解开一个档案而已！不过，重点是那个档名！你要找到正确的档名。

# 在本例中，你不能写成 /etc/shadow ！因为记录在 etc.tar.bz2 内的档名之故！