

linux指令大全（完整篇）

转载自：<http://linuxfire.dhis.org/~firefox/articles/>

作者：wind521 发表时间：2002/06/28 09:16am

名称：cat

使用权限：所有使用者

使用方式：cat [-AbeEnstTuv] [--help] [--version] fileName

说明：把档案串连接后传到基本输出（萤幕或加 > fileName 到另一个档案）

参数：

-n 或 --number 由 1 开始对所有输出的行数编号

-b 或 --number-nonblank 和 -n 相似，只不过对于空白行不编号

-B 或 --number-nonblank 和 -n 相似，只不过对于空白行不编号

-s 或 --squeeze-blank 当遇到有连续两行以上的空白行，就代换为一行的空白行

-v 或 --show-nonprinting

范例：

cat -n textfile1 > textfile2 把 textfile1 的档案内容加上行号后输入

textfile2 这个档案里

cat -b textfile1 textfile2 >> textfile3 把 textfile1 和 textfile2 的档

案内容加上行号（空白行不加）之后将内容附加到 textfile3

名称：cd

使用权限：所有使用者

使用方式：cd [dirName]

说明：变换工作目录至 dirName。其中 dirName 表示法可为绝对路径或相对路

径。若目录名称省略，则变换至使用者的 home directory（也就是刚 login 时

所在的目录）。

另外，"~" 也表示为 home directory 的意思，"." 则是表示目前所在的目录

，".." 则表示目前目录位置的上一层目录。

范例：跳到 /usr/bin/：

cd /usr/bin

跳到自己的 home directory :
cd ~

跳到目前目录的上上两层 :
cd ../../
cd ../../

指令名称 : chmod
使用权限 : 所有使用者
使用方式 : chmod [-cfvR] [--help] [--version] mode file...

说明 : Linux/Unix 的档案存取权限分为三级 : 档案拥有者、群组、其他。利用

chmod 可以藉以控制档案如何被他人所存取。

mode : 权限设定字串, 格式如下 : [ugoa...][[+|=][rwxX]...]
[,...], 其中 u

表示该档案的拥有者, g 表示与该档案的拥有者属于同一个群体 (group) 者, o 表

示其他以外的人, a 表示这三者皆是。

+ 表示增加权限、- 表示取消权限、= 表示唯一设定权限。

r 表示可读取, w 表示可写入, x 表示可执行, X 表示只有当该档案是子目录

或者该档案已经被设定过为可执行。

-c : 若该档案权限确实已经更改, 才显示其更改动作

-f : 若该档案权限无法被更改也不要显示错误讯息

-v : 显示权限变更的详细资料

-R : 对目前目录下的所有档案与子目录进行相同的权限变更 (即以递归的方式逐

个变更)

--help : 显示辅助说明

--version : 显示版本

范例 :

将档案 file1.txt 设为所有人皆可读取 :

```
chmod ugo+r file1.txt
```

将档案 file1.txt 设为所有人皆可读取 :

```
chmod a+r file1.txt
```

将档案 file1.txt 与 file2.txt 设为该档案拥有者, 与其所属同一个群体者可

写入, 但其他以外的人则不可写入 :

```
chmod ug+w,o-w file1.txt file2.txt
```

将 ex1.py 设定为只有该档案拥有者可以执行 :

```
chmod u+x ex1.py
```

将目前目录下的所有档案与子目录皆设为任何人可读取：

```
chmod -R a+r *
```

此外 chmod也可以用数字来表示权限如 `chmod 777 file`

语法为：`chmod abc file`

其中 a, b, c各为一个数字，分别表示 User、Group、及 Other的权限。

r=4, w=2, x=1

若要 rwx 属性则 4+2+1=7;

若要 rw- 属性则 4+2=6;

若要 r-x 属性则 4+1=7。

```
chmod a=rwx file
```

和

```
chmod 777 file
```

效果相同

```
chmod ug=rwx,o=x file
```

和

```
chmod 771 file
```

效果相同

若用 `chmod 4755 filename` 可使此程式具有 root 的权限

指令名称 : `chown`

使用权限 : `root`

使用方式 : `chown [-cfhvR] [--help] [--version] user[:group]`

file...

说明 : Linux/Unix 是多人多工作业系统，所有的档案皆有拥有者。利用 `chown`

可以将档案的拥有者加以改变。一般来说，这个指令只有是由系统管理者 (root)

所使用，一般使用者没有权限可以改变别人的档案拥有者，也没有权限可以自己

的档案拥有者改设为别人。只有系统管理者 (root) 才有这样的权限。

user : 新的档案拥有者的使用者 IDgroup : 新的档案拥有者的使用者群

体 (group)-c : 若该档案拥有者确实已经更改，才显示其更改动作 -

f : 若该档案

拥有者无法被更改也不要显示错误讯息 -h : 只对于连结 (link) 进行变更，而非该

link 真正指向的档案 -v : 显示拥有者变更的详细资料 -R : 对目前目录下的所有

档案与子目录进行相同的拥有者变更 (即以递回的方式逐个变更) --

help : 显示辅

助说明 `--version` : 显示版本

范例 :

将档案 `file1.txt` 的拥有者设为 `users` 群体的使用者 `jessie` :

```
chown jessie:users file1.txt
```

将目前目录下的所有档案与子目录的拥有者皆设为 `users` 群体的使用者

`lampport` :

```
chown -R lampport:users *
```

名称: `cp`

使用权限: 所有使用者

使用方式:

```
cp [options] source dest
```

```
cp [options] source... directory
```

说明: 将一个档案拷贝至另一档案, 或将数个档案拷贝至另一目录。

`-a` 尽可能将档案状态、权限等资料都照原状予以复制。

`-r` 若 `source` 中含有目录名, 则将目录下之档案亦皆依序拷贝至目的地。

`-f` 若目的地已经有相同档名的档案存在, 则在复制前先予以删除再行复制。

范例:

将档案 `aaa` 复制 (已存在), 并命名为 `bbb` :

```
cp aaa bbb
```

将所有的 C 语言程式拷贝至 `Finished` 子目录中 :

```
cp *.c Finished
```

名称 : `find`

用法 : `find`

使用说明 :

将档案系统内符合 `expression` 的档案列出来。你可以指要档案的名称、类别、

时间、大小、权限等不同资讯的组合, 只有完全相符的才会被列出来。

`find` 根据下列规则判断 `path` 和 `expression`, 在命令列上第一个 `- ()`, `! -` 之

前的部份为 `path`, 之后的是 `expression`。如果 `path` 是空字串则使用目前路径

, 如果 `expression` 是空字串则使用 `-print` 为预设 `expression?`

`expression` 中可使用的选项有二三十个之多, 在此只介绍最常用的部份。

`-mount`, `-xdev` : 只检查和指定目录在同一个档案系统下的档案, 避免列出其它

档案系统中的档案

-amin n : 在过去 n 分钟内被读取过
-anewer file : 比档案 file 更晚被读取过的档案
-atime n : 在过去 n 天过读取过的档案
-cmin n : 在过去 n 分钟内被修改过
-cnewer file : 比档案 file 更新的档案
-ctime n : 在过去 n 天过修改过的档案
-empty : 空的档案 -gid n or -group name : gid 是 n 或是 group 名称是 name
-ipath p, -path p : 路径名称符合 p 的档案, ipath 会忽略大小写
-iname name, -name name : 档案名称符合 name 的档案。iname 会忽略大小写
-size n : 档案大小是 n 单位, b 代表 512 位元组的区块, c 表示字元数, k 表示 kilo bytes, w 是二个位元组。-type c : 档案类型是 c 的档案。
d: 目录
c: 字型装置档案
b: 区块装置档案
p: 具名贮列
f: 一般档案
l: 符号连结
s: socket
-pid n : process id 是 n 的档案
你可以使用 () 将运算式分隔, 并使用下列运算。
exp1 -and exp2
! expr
-not expr
exp1 -or exp2
exp1, exp2

范例 :

将目前目录及其子目录下所有延伸档名是 c 的档案列出来。

```
# find . -name "*.c"
```

将目前目录其其下子目录中所有一般档案列出

```
# find . -ftype f
```

将目前目录及其子目录下所有最近 20 分钟内更新过的档案列出

```
# find . -ctime -20
```

名称: less

使用权限: 所有使用者

使用方式:

less [Option] filename

说明:

less 的作用与 more 十分相似, 都可以用来浏览文字档案的内容, 不同的是

less 允许使用者来回卷动

以浏览已经看过的部份, 同时因为 less 并未在一开始就读入整个档案, 因此在

遇上大型档案的开启时, 会比一般的文书编辑器 (如 vi) 来的快速。

指令名称 : ln

使用权限 : 所有使用者

使用方式 : ln [options] source dist, 其中 option 的格式为 :
[-bdfinsvF] [-S backup-suffix] [-V {numbered,existing,simple}]
[--help] [--version] [--]

说明 : Linux/Unix 档案系统中, 有所谓的连结 (link), 我们可以将其视为档案

的别名, 而连结又可分为两种 : 硬连结 (hard link) 与软连结 (symbolic link),

硬连结的意思是一个档案可以有多个名称, 而软连结的方式则是产生一个特殊的

档案, 该档案的内容是指向另一个档案的位置。硬连结是存在同一个档案系统中

, 而软连结却可以跨越不同的档案系统。

ln source dist 是产生一个连结 (dist) 到 source, 至于使用硬连结或软连结则

由参数决定。

不论是硬连结或软连结都不会将原本的档案复制一份, 只会占用非常少量的磁碟

空间。

-f : 连结时先将与 dist 同档名的档案删除 -d : 允许系统管理者硬连结自己的

目录 -i : 在删除与 dist 同档名的档案时先进行询问 -n : 在进行软连结时, 将

dist 视为一般的档案 -s : 进行软连结 (symbolic link) -v : 在连结之前显示其

档名 -b : 将在连结时会被覆写或删除的档案进行备份 -S SUFFIX : 将备份的档案

都加上 SUFFIX 的字尾 -V METHOD : 指定备份的方式 --help : 显示辅助说

明 --version : 显示版本

范例：
将档案 yy 产生一个 symbolic link：zz
ln -s yy zz
将档案 yy 产生一个 hard link：zz
ln yy xx

名称：ls
使用权限：所有使用者
使用方式：ls [-alrtAFR] [name...]

说明：
显示指定工作目录下之内容（列出目前工作目录所含之档案及子目录）。
-a 显示所有档案及目录（ls内定将档案名或目录名称开头为"."的视为隐藏档，
不会列出）
-l 除档案名称外，亦将档案型态、权限、拥有者、档案大小等资讯详细列出
-r 将档案以相反次序显示（原定依英文字母次序）
-t 将档案依建立时间之先后次序列出
-A 同 -a，但不列出"."（目前目录）及".."（父目录）
-F 在列出的档案名称后加一符号；例如可执行档则加"*"，目录则加"/"
-R 若目录下有档案，则以下之档案亦皆依序列出

范例：
列出目前工作目录下所有名称是 s 开头的档案，愈新的排愈后面：
ls -ltr s*
将 /bin 目录以下所有目录及档案详细资料列出：
ls -lR /bin
列出目前工作目录下所有档案及目录；目录于名称后加"/"，可执行档于名称后
加"*"：
ls -AF

名称：more
使用权限：所有使用者
使用方式：more [-dlfpcsu] [-num] [+/pattern] [+linenum]
[fileNames..]

说明：
类似 cat，不过会以一页一页的显示方便使用者逐页阅读，而最基本的指

令就是按空白键（ space）就往下一页显示，按 b 键就会往回（ back）一页显示，而且还有搜寻字串的功能（与 vi 相似），使用中的说明文件，请按 h 。

参数： -num 一次显示的行数

-d 提示使用者，在画面下方显示 [Press space to continue, q to quit.]，

如果使用者按错键，则会显示 [Press h for instructions.] 而不是哔声

-l 取消遇见特殊字元 ^L（送纸字元）时会暂停的功能

-f 计算行数时，以实际上的行数，而非自动换行过后的行数（有些单行字数太长

的会被扩展为两行或两行以上）

-p 不以卷动的方式显示每一页，而是先清除萤幕后再显示内容

-c 跟 -p 相似，不同的是先显示内容再清除其他旧资料

-s 当遇到有连续两行以上的空白行，就代换为一行的空白行

-u 不显示下引号（根据环境变数 TERM 指定的 terminal 而有所不同）

+/ 在每个档案显示前搜寻该字串（pattern），然后从该字串之后开始显示

+num 从第 num 行开始显示

fileNames 欲显示内容的档案，可为复数个数

范例：

more -s testfile 逐页显示 testfile 之档案内容，如有连续两行以上空白行则

以一行空白行显示。

more +20 testfile 从第 20 行开始显示 testfile 之档案内容。

名称： mv

使用权限：所有使用者

使用方式：

mv [options] source dest

mv [options] source... directory

说明：

将一个档案移至另一档案，或将数个档案移至另一目录。

参数：

-i 若目的地已有同名档案，则先询问是否覆盖旧档。

范例：

将档案 aaa 更名为 bbb：

mv aaa bbb

将所有的 C 语言程式移至 Finished 子目录中：


```
mv -i *.c
```

名称: rm

使用权限: 所有使用者

使用方式: rm [options] name...

说明:

删除档案及目录。

-i 删除前逐一询问确认。

-f 即使原档案属性设为唯读, 亦直接删除, 无需逐一确认。

-r 将目录及以下之档案亦逐一删除。

范例:

删除所有 C 语言程式档; 删除前逐一询问确认 :

```
rm -i *.c
```

```
rm -i *.c
```

将 Finished 子目录及子目录中所有档案删除 :

```
rm -r Finished
```

名称: rmdir

使用权限: 于目前目录有适当权限的所有使用者

使用方式: rmdir [-p] dirName

说明:

删除空的目录。

参数:

-p 是当子目录被删除后使它也成为空目录的话, 则顺便一并删除。

范例:

将工作目录下, 名为 AAA 的子目录删除 :

```
rmdir AAA
```

在工作目录下的 BBB 目录中, 删除名为 Test 的子目录。若 Test 删除后, BBB

目录成为空目录, 则 BBB 亦予删除。

```
rmdir -p BBB/Test
```

名称: touch

使用权限: 所有使用者

使用方式:

```
touch [-acfm]
```

```
[-r reference-file] [--file=reference-file]
```

```
[-t MMDDhhmm[[CC]YY][.ss]]
[-d time] [--date=time] [--time={atime, access, use, mtime, modify}]
[--no-create] [--help] [--version]
file1 [file2 ...]
```

说明:

touch 指令改变档案的时间记录。 `ls -l` 可以显示档案的时间记录。

参数:

- a 改变档案的读取时间记录。
 - m 改变档案的修改时间记录。
 - c 假如目的档案不存在，不会建立新的档案。与 `--no-create` 的效果一样。
 - f 不使用，是为了与其他 unix 系统的相容性而保留。
 - r 使用参考档的时间记录，与 `--file` 的效果一样。
 - d 设定时间与日期，可以使用各种不同的格式。
 - t 设定档案的时间记录，格式与 `date` 指令相同。
- `--no-create` 不会建立新档案。
`--help` 列出指令格式。
`--version` 列出版本讯息。
`--version` 列出版本讯息。

范例:

最简单的使用方式，将档案的时间记录改为现在的时间。若档案不存在，系统会

建立一个新的档案。

```
touch file
```

```
touch file1 file2
```

将 `file` 的时间记录改为 5 月 6 日 18 点 3 分，公元两千年。时间的格式可以

参考 `date` 指令，至少需输入 `MMDDHHmm`，就是月日时与分。

```
touch -c -t 05061803 file
```

```
touch -c -t 050618032000 file
```

将 `file` 的时间记录改变成与 `referencefile` 一样。

```
touch -r referencefile file
```

将 `file` 的时间记录改成 5 月 6 日 18 点 3 分，公元两千年。时间可以使用

`am`, `pm` 或是 24 小时的格式，日期可以使用其他格式如 `6 May 2000`

。

```
touch -d "6:03pm" file
```

```
touch -d "05/06/2000" file
```

```
touch -d "6:03pm 05/06/2000" file
```

名称 : `at`

使用权限 : 所有使用者

使用方式 : at -V [-q queue] [-f file] [-mldbv] TIME

说明 :

at 可以让使用者指定在 TIME 这个特定时刻执行某个程式或指令, TIME 的格式是 HH:MM 其中的 HH 为小时, MM 为分钟, 甚至你也可以指定 am, pm,

midnight, noon, teatime(就是下午 4 点钟)等口语词。

midnight, noon, teatime(就是下午 4 点钟)等口语词。

如果想要指定超过一天内的时间, 则可以用 MMDDYY 或者 MM/DD/YY 的格式, 其

中 MM 是分钟, DD 是第几号, YY 是指年份。另外, 使用者甚至也可以使用像是

now + 时间间隔来弹性指定时间, 其中的时间间隔可以是 minutes, hours,

days, weeks?

另外, 使用者也可指定 today 或 tomorrow 来表示今天或明天。当指定了时间并

按下 enter 之后, at 会进入交谈模式并要求输入指令或程式, 当你输入完后按

下 ctrl+D 即可完成所有动作, 至于执行的结果将会寄回你的帐号中。

-V : 印出版本编号

-q : 使用指定的伫列(Queue)来储存, at 的资料是存放在所谓的 queue 中, 使

用者可以同时使用多个 queue, 而 queue 的编号为 a, b, c... z 以及 A, B,

... Z 共 52 个

-m : 即使程式 / 指令执行完成后没有输出结果, 也要寄封信给使用者

-f file : 读入预先写好的命令档。使用者不一定要使用交谈模式来输入, 可以

先将所有的指定先写入档案后再一次读入

-l : 列出所有的指定 (使用者也可以直接使用 atq 而不用 at -l)

-d : 删除指定 (使用者也可以直接使用 atrm 而不用 at -d)

-v : 列出所有已经完成但尚未删除的指定

例子 :

三天后的下午 5 点钟执行 /bin/ls :

```
at 5pm + 3 days /bin/ls
```

三个星期后的下午 5 点钟执行 /bin/ls :

三个星期后的下午 5 点钟执行 /bin/ls :

```
at 5pm + 2 weeks /bin/ls
```

明天的 17:20 执行 /bin/date :

```
at 17:20 tomorrow /bin/date
```

1999 年的最后一天的最后一分钟印出 the end of world !

at 23:59 12/31/1999 echo the end of world !

名称 : crontab

使用权限 : 所有使用者

使用方式 :

crontab [-u user] filecrontab [-u user] { -l | -r | -e }

说明 :

crontab 是用来让使用者在固定时间或固定间隔执行程式之用, 换句话说, 也就

是类似使用者的时程表。 -u user 是指设定指定 user 的时程表, 这个前提是你

必须要有其权限 (比如说是 root) 才能够指定他人的时程表。如果不使用 -u

user 的话, 就是表示设定自己的时程表。

参数 :

-e : 执行文字编辑器来设定时程表, 内定的文字编辑器是 VI , 如果你想用别的

文字编辑器, 则请先设定 VISUAL 环境变数来指定使用那个文字编辑器 (比如说

setenv VISUAL joe)

-r : 删除目前的时程表

-l : 列出目前的时程表

时程表的格式如下 :

f1 f2 f3 f4 f5 program

其中 f1 是表示分钟, f2 表示小时, f3 表示一个月份中的第几号, f4 表示月份

, f5 表示一个星期中的第几天。 program 表示要执行的程式。

当 f1 为 * 时表示每分钟都要执行 program, f2 为 * 时表示每小时都要执行程

式, 其余类推

当 f1 为 a-b 时表示从第 a 分钟到第 b 分钟这段时间内要执行, f2 为 a-b 时

表示从第 a 到第 b 小时都要执行, 其余类推

当 f1 为 */n 时表示每 n 分钟个时间间隔执行一次, f2 为 */n 表示每 n 小时

个时间间隔执行一次, 其余类推

当 f1 为 a, b, c, ... 时表示第 a, b, c, ... 分钟要执行, f2 为 a, b,

c, ... 时表示第 a, b, c, ... 个小时要执行, 其余类推

使用者也可以将所有的设定先存放在档案 file 中, 用 crontab file 的方式来

设定日程表。

例子：

每月每天每小时的第 0 分钟执行一次 /bin/lis：

```
0 7 * * * /bin/lis
```

在 12 月内，每天的早上 6 点到 12 点中，每隔 20 分钟执行一次

```
/usr/bin/backup：
```

```
0 6-12/3 * 12 * /usr/bin/backup
```

周一到周五每天下午 5:00 寄一封信给 alex@domain.name：

```
0 17 * * 1-5 mail -s "hi" alex@domain.name < /tmp/maildata
```

每月每天的午夜 0 点 20 分，2 点 20 分，4 点 20 分

执行 echo "haha"

```
20 0-23/2 * * * echo "haha"
```

注意：

当程式在你所指定的时间执行后，系统会寄一封信给你，显示该程式执行的内容

，若是不希望收到这样的信，请在每一行空一格之后加上 >

```
/dev/null 2>&1
```

即可。

名称：date

使用权限：所有使用者

使用方式：

```
date [-u] [-d datestr] [-s datestr] [--utc] [--universal]
  [--date=datestr] [--set=datestr] [--help] [--version] [+FORMAT]
  [MMDDhhmm[[CC]YY][.ss]]
```

说明：

date 可以用来显示或设定系统的日期与时间，在显示方面，使用者可以设定欲显

示的格式，格式设定为一个加号后接数个标记，其中可用的标记列表如下

：

时间方面：

%：印出 %

%n：下一行

%t：跳格

%H：小时 (00..23)

%I：小时 (01..12)

%k：小时 (0..23)

%l：小时 (1..12)

%M：分钟 (00..59)

%p : 显示本地 AM 或 PM
%r : 直接显示时间 (12 小时制, 格式为 hh:mm:ss [AP]M)
%s : 从 1970 年 1 月 1 日 00:00:00 UTC 到目前为止的秒数
%S : 秒 (00..61)
%T : 直接显示时间 (24 小时制)
%X : 相当于 %H:%M:%S
%Z : 显示时区

日期方面 :

%a : 星期几 (Sun.. Sat)
%A : 星期几 (Sunday.. Saturday)
%b : 月份 (Jan.. Dec)
%B : 月份 (January.. December)
%c : 直接显示日期与时间
%d : 日 (01.. 31)
%D : 直接显示日期 (mm/dd/yy)
%h : 同 %b
%j : 一年中的第几天 (001.. 366)
%m : 月份 (01.. 12)
%U : 一年中的第几周 (00.. 53) (以 Sunday 为一周的第一天的情形)
%w : 一周中的第几天 (0.. 6)
%W : 一年中的第几周 (00.. 53) (以 Monday 为一周的第一天的情形)
%x : 直接显示日期 (mm/dd/yy)
%y : 年份的最后两位数字 (00.. 99)
%Y : 完整年份 (0000.. 9999)

若是不以加号作为开头, 则表示要设定时间, 而时间格式为
MMDDhhmm[[CC]YY][.ss], 其中 MM 为月份, DD 为日, hh 为小时,
mm 为分钟

, CC 为年份前两位数字, YY 为年份后两位数字, ss 为秒数
把计 :

-d datestr : 显示 datestr 中所设定的时间 (非系统时间)
--help : 显示辅助讯息
-s datestr : 将系统时间设为 datestr 中所设定的时间
-u : 显示目前的格林威治时间
--version : 显示版本编号

例子 :

显示时间后跳行, 再显示目前日期 :

```
date +%T%n%D
```

显示月份与日数 :

```
date +%B %d
```

显示日期与设定时间 (12:34:56) :

```
date --date 12:34:56
```

注意 :

当你不希望出现无意义的 0 时（比如说 1999/03/07），则可以在标记中插入 - 符号，比如说 `date +%H:%M:%S` 会把时分秒中无意义的 0 给去掉，像是原本的

08:09:04 会变为 8:9:4。另外，只有取得权限者（比如说 root）才能设定系统时间。

当你以 root 身分更改了系统时间之后，请记得以 `clock -w` 来将系统时间写入

CMOS 中，这样下次重新开机时系统时间才会持续抱持最新的正确值。

名称：sleep

使用权限：所有使用者

使用方式：sleep [--help] [--version] number[smhd]

说明：sleep 可以用来将目前动作延迟一段时间

参数说明：

--help：显示辅助讯息

--version：显示版本编号

number：时间长度，后面可接 s、m、h 或 d

其中 s 为秒，m 为分钟，h 为小时，d 为日数

例子：

显示目前时间后延迟 1 分钟，之后再次显示时间：
`date;sleep 1m;date`

名称：time

使用权限：所有使用者

使用方式：time [options] COMMAND [arguments]

说明：time 指令的用途，在于量测特定指令执行时所需消耗的时间及系统资源

等资讯。例如 CPU 时间、记忆体、输入输出等等。需要特别注意的是，部分资讯

在 Linux 上显示不出来。这是因为在 Linux 上部分资源的分配函式与 time 指

令所预设的方式并不相同，以致于 time 指令无法取得这些资料。

-o or --output=FILE

设定结果输出档。这个选项会将 time 的输出写入所指定的档案中。如果档案已

经存在，系统将覆写其内容。

-a or --append

配合 -o 使用，会将结果写到档案的末端，而不会覆盖掉原来的内容。

-f FORMAT or --format=FORMAT

以 FORMAT 字串设定显示方式。当这个选项没有被设定的时候，会用系统预设的

格式。不过你可以用环境变数 time 来设定这个格式，如此一来就不必每次登入

系统都要设定一次。

一般设定上，你可以用

t

表示跳栏，或者是用

n

表示换行。每一项资料要用 % 做为前导。如果要在字串中使用百分比符号，就用

。（学过 C 语言的人大概会觉得很熟悉）

time 指令可以显示的资源有四大项，分别是：

Time resources

Memory resources

IO resources

Command info

详细的内容如下：

Time Resources

E 执行指令所花费的时间，格式是： [hour]:minute:second。请注意这个数字并

不代表实际的 CPU 时间。

e 执行指令所花费的时间，单位是秒。请注意这个数字并不代表实际的 CPU 时间。

S 指令执行时在核心模式（kernel mode）所花费的时间，单位是秒。

U 指令执行时在使用者模式（user mode）所花费的时间，单位是秒。

P 执行指令时 CPU 的占用比例。其实这个数字就是核心模式加上使用者模式的

CPU 时间除以总时间。

Memory Resources

M 执行时所占用的实体记忆体的最大值。单位是 KB

t 执行时所占用的实体记忆体的平均值，单位是 KB

K 执行程序所占用的记忆体总量（stack+data+text）的平均大小，单位是 KB

D 执行程序的自有资料区（unshared data area）的平均大小，单位是 KB

p 执行程序的自有堆叠（unshared stack）的平均大小，单位是 KB

X 执行程序间共享内容（shared text）的平均值，单位是 KB

Z 系统记忆体页的大小，单位是 byte。对同一个系统来说这是个常数

IO Resources

F 此程序的主要记忆体页错误发生次数。所谓的主要记忆体页错误是指某一记忆

体页已经置换到置换档 (swap file) 中, 而且已经分配给其他程序。此时该页的

内容必须从置换档里再读出来。

R 此程序的次要记忆体页错误发生次数。所谓的次要记忆体页错误是指某一记忆

体页虽然已经置换到置换档中, 但尚未分配给其他程序。此时该页的内容并未被

破坏, 不必从置换档里读出来

W 此程序被交换到置换档的次数

c 此程序被强迫中断 (像是分配到的 CPU 时间耗尽) 的次数

w 此程序自愿中断 (像是在等待某一个 I/O 执行完毕, 像是磁碟读取等等) 的次

数

l 此程序所输入的档案数

o 此程序所输出的档案数

r 此程序所收到的 Socket Message

s 此程序所送出的 Socket Message

k 此程序所收到的信号 (Signal) 数量

Command Info

C 执行时的参数以及指令名称

x 指令的结束代码 (Exit Status)

-p or --portability

这个选项会自动把显示格式设定成为:

real %e

user %U

sys %S

这么做的目的是为了与 POSIX 规格相容。

-v or --verbose

这个选项会把所有程式中用到的资源通通列出来, 不但如一般英文语句, 还有说

明。对不想花时间去熟习格式设定或是刚刚开始接触这个指令的人相当有用。

范例:

利用下面的指令

time -v ps -aux

我们可以获得执行 ps -aux 的结果和所花费的系统资源。如下面所列的资料:

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.4 1096 472 ? S Apr19 0:04 init
root 2 0.0 0.0 0 0 ? SW Apr19 0:00 [kflushd]
root 3 0.0 0.0 0 0 ? SW Apr19 0:00 [kpiod]
```

```
.....
root 24269 0.0 1.0 2692 996 pts/3 R 12:16 0:00 ps -aux
Command being timed: "ps -aux"
User time (seconds): 0.05
System time (seconds): 0.06
Percent of CPU this job got: 68%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.16
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 0
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 238
Minor (reclaiming a frame) page faults: 46
Voluntary context switches: 0
Involuntary context switches: 0
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

名称 :login

这个命令都不会就不要干算了！呵呵我也不在这里多费笔墨耽误大家美好青春了 ^_^

名称: passwd

使用权限: 所有使用者

使用方式: passwd [-k] [-l] [-u [-f]] [-d] [-S] [username]

说明: 用来更改使用者的密码

参数:

-k
-l
-u
-f

-d 关闭使用者的密码认证功能，使用者在登入时将可以不用输入密码，只有具备

root 权限的使用者方可使用。

-S 显示指定使用者的密码认证种类，只有具备 root 权限的使用者方可使用。

[username] 指定帐号名称。

名称：who

使用权限：所有使用者都可使用

使用方式：who - [husfV] [user]

说明：显示系统中有那些使用者正在上面，显示的资料包含了使用者 ID，使用

的终端机，从那边连上来的，上线时间，呆滞时间，CPU 使用量，动作等等。

-h：不要显示标题列

-u：不要显示使用者的动作 / 工作

-s：使用简短的格式来显示

-f：不要显示使用者的上线位置

-V：显示程式版本

名称：mail

使用权限：所有使用者

使用方式：mail [-iInV] [-s subject] [-c cc-addr] [-b bcc-addr]
user1

[user 2 ...]

说明：

mail 不仅只是一个指令，mail 还是一个电子邮件程式，不过利用 mail 来读信

的人应该很少吧！对于系统管理者来说 mail 就很有用，因为管理者可以用

mail 写成 script，定期寄一些备忘录提醒系统的使用者。

参数：

i 忽略 tty 的中断讯号。(interrupt)

I 强迫设成互动模式。(Interactive)

v 列印出讯息，例如送信的地点、状态等等。(verbose)

n 不读入 mail.rc 设定档。

s 邮件标题。

c cc 邮件地址。

b bcc 邮件地址。

范例：

将信件送给一个或以上的电子邮件地址，由于没有加入其他的选项，使用者必须

输入标题与信件的内容等。而 user2 没有主机位置，就会送给邮件服务器的

user2 使用者。

```
mail user1@email.address
```

```
mail user1@email.address user2
```

将 mail.txt 的内容寄给 user2 同时 cc 给 user1 。如果将这一行指令设成

cronjob 就可以定时将备忘录寄给系统使用者。

```
mail -s 标题 -c user1 user2 < mail.txt
```

名称 : talk

使用权限 : 所有使用者

使用方式 :

```
talk person [ttyname]
```

说明 :

与其他使用者对谈

person : 预备对谈的使用者帐号，如果该使用者在其他机器上，则可输入 person@machine.name

ttyname : 如果使用者同时有两个以上的 tty 连线，可以自行选择合适的 tty

传讯息

例子 .1 :

与现在机器上的使用者 Rollaend 对谈，此时 Rollaend 只有一个连线

:

```
talk Rollaend
```

接下来就是等 Rollaend 回应，若 Rollaend 接受，则 Rollaend 输入

```
`talk jzlee`即
```

可开始对谈，结束请按 ctrl+c

例子 .2 : 与 linuxfab.cx 上的使用者 Rollaend 对谈，使用 pts/2 来对谈 :

```
talk Rollaend@linuxfab.cx pts/2
```

接下来就是等 Rollaend 回应，若 Rollaend 接受，则 Rollaend 输入

```
`talk
```

```
jzlee@jzlee.home`即可开始对谈，结束请按 ctrl+c
```

注意：若萤幕的字会出现不正常的字元，试着按 `ctrl+l` 更新萤幕画面。

名称： `kill`

使用权限：所有使用者

使用方式：

```
kill [ -s signal | -p ] [ -a ] pid ...
```

```
kill -l [ signal ]
```

说明：

`kill` 送出一个特定的信号 (`signal`) 给行程 `id` 为 `pid` 的行程根据该信号

而做特定的动作，若没有指定，预设是送出终止 (`TERM`) 的信号把计？

`-s (signal)`：其中可用的讯号有 `HUP` (1), `KILL` (9), `TERM` (15), 分别代表着

重跑，砍掉，结束；详细的信号可以用 `kill -l`

`-p`：印出 `pid`，并不送出信号

`-p`：印出 `pid`，并不送出信号

`-l (signal)`：列出所有可用的信号名称

范例：

将 `pid` 为 `323` 的行程砍掉 (`kill`)：

```
kill -9 323
```

将 `pid` 为 `456` 的行程重跑 (`restart`)：

```
kill -HUP 456
```

名称： `ps`

使用权限：所有使用者

使用方式： `ps [options] [--help]`

说明：显示瞬间行程 (`process`) 的动态

参数：

`ps` 的参数非常多，在此仅列出几个常用的参数并大略介绍含义

`-A` 列出所有的行程

`-w` 显示加宽可以显示较多的资讯

`-au` 显示较详细的资讯

`-aux` 显示所有包含其他使用者的行程

`au(x)` 输出格式：

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
```

USER: 行程拥有者
PID: pid
%CPU: 占用的 CPU 使用率
%MEM: 占用的记忆体使用率
VSZ: 占用的虚拟记忆体大小
RSS: 占用的记忆体大小
TTY: 终端的次要装置号码 (minor device number of tty)
STAT: 该行程的状态 :
D: 不可中断的静止 (通常□□值 b 进行 I/O 动作)
R: 正在执行中
S: 静止状态
T: 暂停执行
Z: 不存在但暂时无法消除
W: 没有足够的记忆体分页可分配
N: 低优先序的行程
L: 有记忆体分页分配并锁在记忆体内 (即时系统或握 A I/O)
START: 行程开始时间
TIME: 执行的时间
COMMAND: 所执行的指令

范例:

```
ps
PID TTY TIME CMD
2791 tty0 00:00:00 tcsh
3092 tty0 00:00:00 ps
% ps -A
PID TTY TIME CMD
1 ? 00:00:03 init
2 ? 00:00:00 kflushd
3 ? 00:00:00 kpiod
4 ? 00:00:00 kswapd
5 ? 00:00:00 mdrecoveryd
.....
% ps -aux
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.7 1096 472 ? S Sep10 0:03 init [3]
root 2 0.0 0.0 0 0 ? SW Sep10 0:00 [kflushd]
root 3 0.0 0.0 0 0 ? SW Sep10 0:00 [kpiod]
root 4 0.0 0.0 0 0 ? SW Sep10 0:00 [kswapd]
.....
```

名称: top
使用权限: 所有使用者

使用方式: top [-] [d delay] [q] [c] [S] [s] [i] [n] [b]

说明: 即时显示 process 的动态

d : 改变显示的更新速度, 或是在交谈式指令列 (interactive command)
按 s

q : 没有任何延迟的显示速度, 如果使用者是有 superuser 的权限, 则
top 将
会以最高的优先序执行

c : 切换显示模式, 共有两种模式, 一是只显示执行档的名称, 另一种是
显示完

c : 切换显示模式, 共有两种模式, 一是只显示执行档的名称, 另一种是
显示完

整的路径与名称 S : 累积模式, 会将已完成或消失的子行程 (dead
child

process) 的 CPU time 累积起来

s : 安全模式, 将交谈式指令取消, 避免潜在的危机

i : 不显示任何闲置 (idle) 或无用 (zombie) 的行程

n : 更新的次数, 完成后将会退出 top

b : 批次档模式, 搭配 "n" 参数一起使用, 可以用来将 top 的结果
输出到档案

内

范例:

显示更新十次后退出 ;

```
top -n 10
```

使用者将不能利用交谈式指令来对行程下命令 :

```
top -s
```

将更新显示二次的结果输入到名称为 top.log 的档案里 :

```
top -n 2 -b < top.log
```

名称: expr

使用权限: 所有使用者

字符串长度

```
shell>> expr length "this is a test"
```

```
14
```

数字商数

```
shell>> expr 14 % 9
```

```
5
```

从位置处抓取字符串

```
shell>> expr substr "this is a test" 3 5
```

```
is is
```

字符串 only the first character

```
shell>> expr index "testforthegame" e
2
### 字符串真实重现
shell>> expr quote thisisatestformela
thisisatestformela
```

名称 : tr

1. 比方说要把目录下所有的大写档名换为小写档名?
似乎有很多方式, "tr" 是其中一种 :

```
#!/bin/sh
dir="/tmp/testdir";
files=`find $dir -type f`;
for i in $files
do
dir_name=`dirname $i`;
ori_filename=`basename $i`
new_filename=`echo $ori_filename | tr [:upper:] [:lower:]` >
/dev/null;
#echo $new_filename;
mv $dir_name/$ori_filename $dir_name/$new_filename
done
```

2. 自己试验中 ... lowercase to uppercase

```
tr abcdef... [del] ABCDE... [del]
tr a-z A-Z
tr [:lower:] [:upper:]
shell>> echo "this is a test" | tr a-z A-Z > www
shell>> cat www
```

THIS IS A TEST

3. 去掉不想要的字符串

```
shell>> tr -d this ### 去掉有关 t.e.s.t
```

this

man

man

test

e

4. 取代字符串

```
shell>> tr -s "this" "TEST"
```

this

TEST

th

TE

指令: `clear`
用途: 清除萤幕用。
使用方法: 在 `console` 上输入 `clear`。

名称: `compress`
使用权限: 所有使用者
使用方式: `compress [-dfvcV] [-b maxbits] [file ...]`

说明:
`compress` 是一个相当古老的 `unix` 档案压缩指令, 压缩后的档案会加上一个 `.Z`
延伸档名以区别未压缩的档案, 压缩后的档案可以以 `uncompress` 解压。
若要将
数个档案压成一个压缩档, 必须先将档案 `tar` 起来再压缩。由于
`gzip` 可以产生
更理想的压缩比例, 一般人多已改用 `gzip` 为档案压缩工具。

参数:
`c` 输出结果至标准输出设备 (一般指萤幕)
`f` 强迫写入档案, 若目的档已经存在, 则会被覆盖 (`force`)
`v` 将程式执行的讯息印在萤幕上 (`verbose`)
`b` 设定共同字符串数的上限, 以位元计算, 可以设定的值为 `9` 至 `16`
`bits`。由于
值越大, 能使用的共同字符串就越多, 压缩比例就越大, 所以一般使用预设值 `16`
`bits (bits)`
`d` 将压缩档解压缩
`V` 列出版本讯息

范例:
将 `source.dat` 压缩成 `source.dat.Z`, 若 `source.dat.Z` 已经存在, 内容则会
被压缩档覆盖。
`compress -f source.dat`
将 `source.dat` 压缩成 `source.dat.Z`, 并列印出压缩比例。
`-v` 与 `-f` 可以一起使用
`compress -vf source.dat`
将压缩后的资料输出后再导入 `target.dat.Z` 可以改变压缩档名。
`compress -c source.dat > target.dat.Z`
`-b` 的值越大, 压缩比例就越大, 范围是 `9-16`, 预设值是 `16`。
`compress -b 12 source.dat`

将 `source.dat.Z` 解压成 `source.dat`，若档案已经存在，使用者按 `y` 以确定覆盖档案，若使用 `-df` 程式则会自动覆盖档案。由于系统会自动加入 `.Z` 为延伸档名，所以 `source.dat` 会自动当作 `source.dat.Z` 处理。

```
compress -d source.dat
compress -d source.dat.Z
```

名称: `lpd`
使用权限: 所有使用者
使用方式: `lpd [-l] [#port]`

`lpd` 是一个常驻的印表机管理程式，它会根据 `/etc/printcap` 的内容来管理本地

或远端的印表机。`/etc/printcap` 中定义的每一个印表机必须在 `/var/lpd` 中有

一个相对应的目录，目录中以 `cf` 开头的档案表示一个等待送到适当装置的印表

工作。这个档案通常是由 `lpr` 所产生。

`lpr` 和 `lpd` 组成了一个可以离线工作的系统，当你使用 `lpr` 时，印表机不需要

能立即可用，甚至不用存在。`lpd` 会自动监视印表机的状况，当印表机上线后，

便立即将档案送交处理。这个得所有的应用程式不必等待印表机完成前一工作。

参数:

`-l`: 将一些除错讯息显示在标准输出上。

`#port`: 一般而言，`lpd` 会使用 `getservbyname` 取得适当的 TCP/IP port，你可

以使用这个参数强迫 `lpd` 使用指定的 port。

范例:

这个程式通常是由 `/etc/rc.d` 中的程式在系统启始阶段执行。

名称 `lpq`

-- 显示列表机贮列中未完成的工作 用法

```
lpq [l] [P] [user]
```

说明

`lpq` 会显示由 `lpd` 所管理的列表机贮列中未完成的项目。

范例

1. 显示所有在 lp 列表机队列中的工作

```
# lpq -PlpRank Owner Job Files Total Size1st root 238 (standard  
input)
```

```
1428646 bytes
```

相关函数

lpr, lpc, lpd

名称: lpr

使用权限: 所有使用者

使用方式: lpr [-P printer]

将档案或是由标准输入送进来的资料送到印表机队列之中，印表机管理程式 lpd

会在稍后将这个档案送给适当的程式或装置处理。 lpr 可以用来将资料送给本地

或是远端的主机来处理。

参数:

-p Printer: 将资料送至指定的印表机 Printer, 预设值为 lp。

范例:

将 www.c 和 kkk.c 送到印表机 lp。

```
lpr -Plp www.c kkk.c
```

名称 : lprm

-- 将一个工作由印表机队列中移除 用法

```
/usr/bin/lprm [P] [file...]
```

说明

尚未完成的印表机工作会被放在印表机队列之中，这个命令可用来将常未送到印

表机的工作取消。由于每一个印表机都有一个独立的队列，你可以用 -P 这个命

令设定想要作用的印列机。如果没有设定的话，会使用系统预设的印表机。

这个命令会检查使用者是否有足够的权限删除指定的档案，一般而言，只有档案

的拥有者或是系统管理员才有这个权限。

范例

将印表机 hpprinter 中的第 1123 号工作移除

```
lprm -Phpprinter 1123
```

将第 1011 号工作由预设印表机中移除

```
lprm 1011
```

