Sec 06 linux 文件与目录管理

目录与路径

- 相对路径
- 绝对路径

特殊的目录

.	此层目录	所有目录底下都存在
..	上一层目录	所有目录底下都存在
-	前一个工作目录	
~	目前用户身份所在的家目录	
~<account>	<account>这个用户的家目录	

注：根目录下也存在 .. ，其 .. 与 . 是同个目录。

处理目录的指令

cd = change directory

pwd = print working directory

[-P] 显示出确实的路径，而非链接(link)路径

mkdir = make directory

[-m] 配置目录的权限，而不是使用预设权限(umask)

[-p] 递归地建立 parent 目录（如果本来不存在的话）

rmdir = remove directory 删除空的目录 （对比 rm -r <…>）

[-p] 连同上层、空的目录也一起删除 e.g. rmdir -p test1/test2/test3/test4

关于执行文件路径的变量：$PATH

ls 指令的完整文件名为 /bin/ls（有时）

系统会按照PATH的设定去每个PATH定义的目录下搜寻文件名为 ls 的可执行文件，如果在PATH定义的目录中会有多个文件名为 ls 的可执行文件，那么先搜寻到的同名指令会被执行。

```
root@VSA11779357:~ # echo $PATH
/sbin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/usr/bin:/bin:
/usr/games:/sbin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/usr/bin
i542016@VSA11779357:~ > echo $PATH
/home/i542016/bin:/usr/local/bin:/usr/bin:/bin:/usr/games:/sbin:/usr/sbin:
/usr/local/sbin:/root/bin:/usr/local/bin:/usr/bin
```

注：以":"作为分隔符，有顺序之分。 `which ls` ⇒ /usr/bin/ls，且 /bin/ls 是link到 /usr/bin/ls

若想将 /root 加入PATH当中，只需 `PATH="${PATH}:/root"`。

上例中，`ls` = `/usr/bin/ls`。

不同身份者预设的PATH不同，默认的能够"随意"执行的指令也不同。

文件与目录管理

文件与目录的检视：ls

ls [-aAdfFhilnrRSt] <文件或目录名称>
[--color={never, auto, always}]
[--full-time]

-a：全部的文件与目录（含隐藏文件与目录）
-A：参考 -a，但不包含 . 与 ..
-F：列出类型，如 * 代表可执行文件，/ 代表目录 等
-h：须与 -l 一起用，将文件大小以人类易读的方式（kB、GB）列出（不能用来查看文件夹的占用空间）
`du -sh <…>`
-l：长数据串，包含属性与权限
-R：连同子目录内容一起列出来
-r：将排序结果反向输出

略

复制、删除与移动：cp，rm，mv

cp [-adfilprsu] <来源文件 source><目标文件 destination>
[options] source1 source2 ... sourceN <directory/>
目录！

-p：连同文件的属性（权限、用户、时间）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
-i：若目标文件已经存在，在覆盖前会先进行询问
-a：相当于 -dr --preserve=all
注：在预设的条件中，cp 的 source 与 destination 的权限是不同的，destination 的拥有者通常是指令操作者
-u：destination 比 source 旧才更新 destination，或 destination 不存在的情况下才复制（备份）
-d：若来源文件为链接文件，则复制链接文件而非文件本体

rm [-fir] <文件或目录>
-f：force
-i：互动模式，在删除前会询问使用者
-r：递归删除，用于目录的删除
若要删除一个叫 -aaa- 的文件，则可以 `rm -- -aaa-` 或者 `rm ./-aaa-`

mv [-fiu] <source><destination>
[options] source1 source2 ... sourceN <directory>
-f：force，若目标文件已经存在，则不会询问而直接覆盖
-i：若目标文件已经存在，就会进行询问
-u：update，若目标文件已经存在，且 source 比较新时，才会更新

取得路径的文件名与目录名称

```
i542016@vsa11779357: ~ > dirname /home/i542016/desks
/home/i542016
i542016@vsa11779357: ~ > basename /home/i542016/desks
desks
```

文件内容查询

cat
tac 反向 cat
nl 显示的时候，顺便输出行号
more 一页一页地显示文件内容
less 与more类似，但它可以往前翻页
head 只看头几行
tail 只看后几行
od 以二进制的方式读取文件内容

cat := concatenate

cat [-AbEnTv] <文件>

[-b] 列出行号，仅针对非空白行做行号显示，空白行不标行号
[-n] 列出行号，连同空白行也会有行号
[-T] 将 tab 按键以 ^I 显示出来
[-E] 将结尾的断行字符 $ 显示出来
[-A] 相当于 -vET

nl [-bnw] <文件>

[-b]
-b a：不论是否为空行，都列出行号（cat -n）
-b t：(default) 空行不列出行号

[-n]
-n ln：行号显示在屏幕的左侧
-n rn：行号显示在字段的右侧，不加0
-n rz：行号显示在字段的右侧，且加0

[-w]
-w <num>：行号有 <num> 位

more

space	向下翻一页	:+f	显示出文件名与目前显示的行数
enter	向下翻一行	q	退出
/+<string>	按 Enter 开始搜索下面的内容，按 n 搜索下一个	b	往回翻一页

less

space 向下翻一页

pagedown 向下翻一页

pageup 向上翻一页

/ + <string> 向下搜寻 <string>

? + <string> 向上搜寻 <string>

n 重复前一个搜寻

N 反向地重复前一个搜寻

g 到第一行去

G 到最后一行去

q 退出

注：man这个指令就是呼叫less来显示说明内容的。

head [-n <num>] <文件>

head -n -10 <文件> ：不显示尾部的10行

tail [-n <num>] <文件>

tail -n +10 <文件> ：只显示10行以后的内容

tail -f <文件> ：持续follow，按 ctrl + c 离开

文件与目录的默认权限与隐藏权限

文件预设权限 umask

umask就是指定目前用户在建立文件或目录时候的默认权限

```
i542016@vsa11779357: ~> umask
0022
i542016@vsa11779357: ~> umask -S
u=rwx,g=rx,o=rx
```

umask的分数指的是该默认值需要减去的权限

建立文件时，默认"没有x权限"，即只有rw权限，最大为666分(-rw-rw-rw-)

建立目录时，默认"所有权限均开放"，最大为777分(drwxrwxrwx)

上例中，

∵ r=4, w=2, x=1 且 umask=022

∴ 当建立文件时，u= 6-0=6=rw，g=6-2=4=r，o=6-2=4=r

即 -rw-r--r--

当建立目录时，u=7-0=7=rwx，g=7-2=5=rx，o=7-2=5=rx 即 drwxr-xr-x

若需要设置新建的文件的权限为 -rw-rw-r--，
则只需执行 `umask 002`，
此时，
当新建文件时，u=6-0=6=rw，g=6-0=6=rw，o=6-2=4=r
即 -rw-rw-r--
当新建目录时，u=7-0=7=rwx，g=7-0=7=rwx，o=7-2=5=rx
即 drwxrwxr-x

文件隐藏属性

配置文件与目录的隐藏属性用 chattr 命令，
显示文件与目录的隐藏属性用 lsattr 命令。

文件特殊权限

```
root@vsa1779357:~# ls -ld /tmp
      drwxrwxrwt ...
root@vsa1779357:~# ls -l /usr/bin/passwd
      -rwsr-xr-x ...
```

s与t 和"系统的帐号"及"系统的程序"较为相关。

△ 当s出现在文件拥有者的x权限上时，此时就被称为SUID(:=Set UID)的特殊权限，

① SUID 权限仅对二进制程序有效
② 执行者对于该程序需要具有 x 权限
③ 本权限仅在执行该程序的过程中有效（run-time）
④ 执行者将具有该程序的拥有者(owner)的权限

以修改密码的指令 /usr/bin/passwd 为例，
j542016 对于 /usr/bin/passwd 这个程序来说是有 x 权限的，
/usr/bin/passwd 的拥有者是 root，
但是 j542016 在执行 passwd 的过程中，会"暂时"获得 root 的权限，/etc/shadow 便可以被修改。
然而，j542016 不能使用 cat 去读取 /etc/shadow，因为 cat 没有 SUID 的权限。

注：
SUID 仅可用在 binary program 上，不能用在 shell script 上！
因为 shell script 只是将很多的 binary program "叫进来执行"而已。
且 SUID 的权限部分得看 shell script 呼叫进来的程序的设定，而非 shell script 本身。

△ 当s出现在群组的x权限上时，此时称为SGID(:=Set GID)的特殊权限，
例如 /usr/bin/write，SGID 可以针对文件或目录来设定。

对文件来说，
① SGID 对二进制程序有用
② 程序执行者对于该程序来说，需具有 x 权限
③ 执行者在执行的过程中将会获得该程序群组的支持

对于目录来说，当一个目录设定了SGID的权限后，它将具有如下的功能：

① 用户若对于此目录具有r与x的权限时，该用户能够进入此目录

② 用户在此目录下的有效群组（effective group）将会变成该目录的群组

③ 用途：若用户在此目录下具有w的权限（可以新建文件），则使用者所建立的新文件的群组与此目录的群组相同

△ SBIT (:= Sticky Bit)

只针对目录有效，

假如用户对于此目录具有w, x权限，即具有写入的权限时，

当用户在该目录下建立文件或目录时，仅有自己与root才有权力删除之。

例如 /tmp，(drwxrwxrwt)。

如何设定 SUID/SGID/SBIT ?

在chmod时，在代表权限的三个数字之前再加上一个数字即可。（4为SUID，2为SGID，1为SBIT）

观察文件类型：file

略

指令与文件的搜寻

脚本文件名的搜寻

which [-a] <command> 寻找执行档

[-a]：将所有在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

注：`which history`会失效，因为history是"bash内建的命令"，而which是去PATH目录下找。(type)

文件档名的搜寻

whereis [-bmsu] <文件或目录名> 在一些特定的目录中寻找文件档名

[-l] 可以列出 whereis 会去查询的目录

[-b] 只找 binary 格式的文件

[-m] 只找在说明文件 manual 路径下的文件

[-s] 只找 source 来源文件

find [PATH] [option] [action]

option
- 与时间有关的选项：n天前、n天内、比某个文件新 ...
- 与 user、group name 有关的参数
- 与文件权限及名称有关的参数：-name、-size、-type、-perm
- 额外可进行的动作：-exec <command>

例：find /usr/bin /usr/sbin -perm /7000 -exec ls -l {} \;

{}："find"找到的内容

\;：转义，;在bash环境下有特殊意义