第5讲 文本文件处理及文件查找

Linux中熟练的使用文本处理工具非常的重要, 因为Linux在设计的时候是采用一切皆文件的哲学的, 系统的配置信息都是以文本方式进行存储的，甚至连计算机中的配置也都使用伪文件系统来表示, 在实际工作中，我们经常需要使用各种文本处理工具来处理各种文本信息，比如我们要查看一个配置文件内容，编辑某一个配置文件等操作。

1 文件内容查看命令：

1）cat (concatenate)命令

[root@study ~]# cat [-AbEnTv]

选项

-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 $ 显示出来以及其它非控制字符显示出来；

-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

为了保证系统文件安全，建议大家在学习时把如下几个文件拷到用户自己的主目录下再去操作

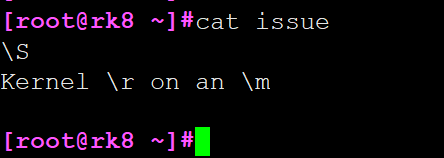
范例

将如下三个文件拷到当前用户主目录下

[root@rk8 ~]#cp /etc/inittab /etc/issue /etc/man\_db.conf /etc/passwd ~/

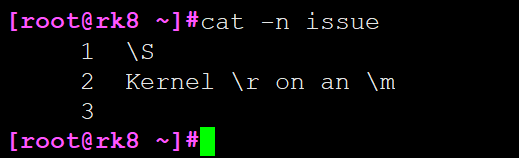
范例1：查看issue 这个文件的内容

[root@study ~]# cat issue



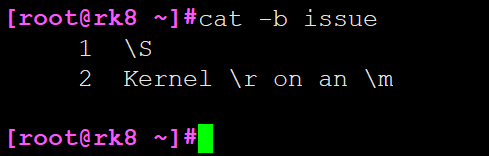
范例2：承上题，如果还要加印行号呢？

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

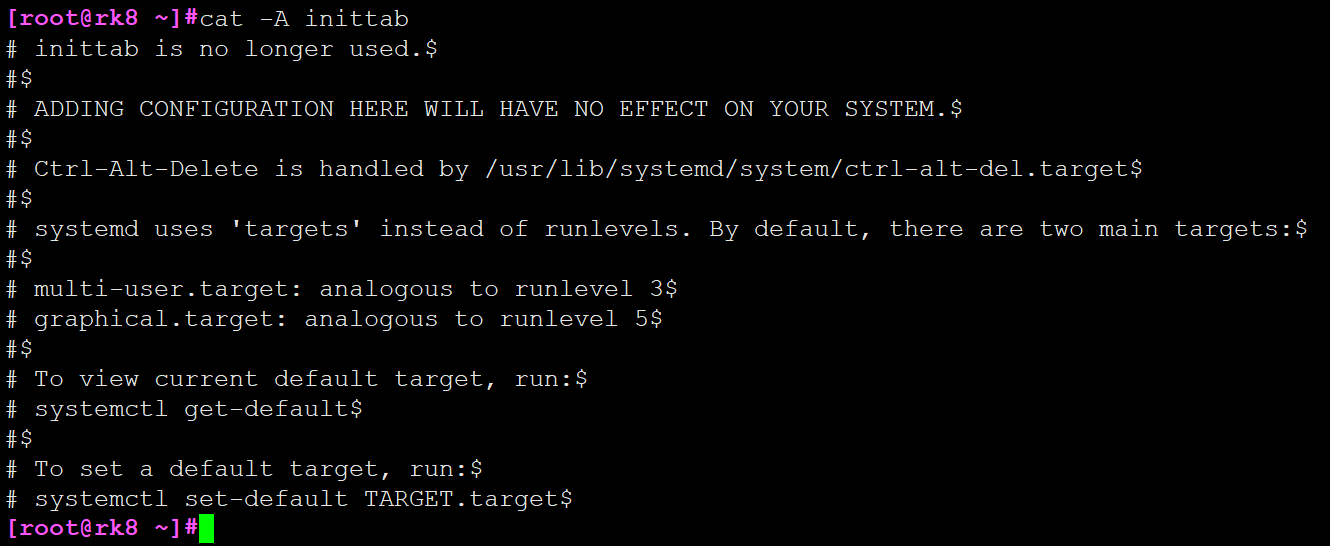


所以这个文件实际有三行！可以印出行号呢，这对于大文件要找某个特定的行时，有点用处！

如果不想要编排空白行的行号，可以使用『cat -b /etc/issue』如下所示：



范例3：将 inittab 的内容完整的显示出来(包含特殊字符)

[root@rk8 ~]#cat -A inittab 此时使用 cat -A 就能够发现那些空白的地方是啥鬼东西了！，断行字符则是以 $ 表示，所以你可以发现每一行后面都是 $ 啊！如果是[tab]的空格则会以 ^I 表示不过断行字符在Windows/Linux则不太相同，Windows的断行字符是 ^M$ 。

当你的文件内容的行数超过 40 行以上，那么一般情况下一屏幕就无法显示完整结果了，因为cat显示文件内容后会直接回到命令提示符状态，所以，这个时候就使用 more 或者是 less 来查看较好！此外，如果是一般的 DOS 文件时或者windows下的文本，就需要特别留意一些奇奇怪怪的符号了， 例如断行与 [tab] 等，要显示出来，就得加入 -A 之类的选项了！

同时也要注意：有些配置文件中不能出现windows这种换行符号的^M。

2）tac (反向列示)

[root@study ~]# tac /etc/issue

Kernel \r on an \m

\S

# 嘿嘿！与刚刚上面的范例一比较，是由最后一行先显示喔！

tac 这个好玩了！怎么说呢？详细的看一下， cat 与 tac ，有没有发现呀！对啦！ tac 刚好是将 cat 反写过来，所以他的功能就跟 cat 相反啦， cat 是由『第一行到最后一行连续显示在荧幕上』，而 tac 则是『 由最后一行到第一行反向在荧幕上显示出来 』，很好玩吧！

3）nl (添加行号打印)

[root@study ~]# 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(预设值)；

-n ：列出行号表示的方法，主要有三种：

-n ln ：行号在荧幕的最左方显示；

-n rn ：行号在自己栏位的最右方显示，且不加 0 ；

-n rz ：行号在自己栏位的最右方显示，且加 0 ；

-w ：行号栏位的占用的字符数。

范例一：用 nl 列出 /etc/issue 的内容

[root@study ~]# nl /etc/issue

1 \S

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@study ~]# nl -b a /etc/issue

1 \S

2 Kernel \r on an \m

3

# 呵呵！行号加上来囉～那么如果要让行号前面自动补上 0 呢？可这样

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

000001 \S

000002 Kernel \r on an \m

000003

# 嘿嘿！自动在自己栏位的地方补上 0 了～预设栏位是六位数，如果想要改成 3 位数？

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

001 \S

002 Kernel \r on an \m

003

# 变成仅有 3 位数囉～

nl 可以将输出的档案内容自动的加上行号！其预设的结果与 cat -n 有点不太一样， nl 可以将行号做比较多的显示设计，包括位数与是否自动补齐 0 等等的功能呢。

2 可翻页查看文件内容

前面提到的 nl 与 cat, tac 等等，都是一次性的将资料一口气显示到荧幕上面，那有没有可以进行一页一页翻动的指令啊？ 让我们可以一页一页的观察，才不会前面的资料看不到啊～呵呵！有的！那就是 more 与 less 囉～

1）more (一页一页向前翻动)

[root@study ~]# more /etc/man\_db.conf

#

#

# This file is used by the man-db package to configure the man and cat paths.

# It is also used to provide a manpath for those without one by examining

# their PATH environment variable. For details see the manpath(5) man page.

#

.....(中间省略).....

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

仔细的给他看到上面的范例，如果 more 后面接的档案内容行数大于荧幕输出的行数时， 就会出现类似上面的图示。重点在最后一行，最后一行会显示出目前显示的百分比， 而且还可以在最后一行输入一些有用的指令喔！在 more 这个程序的运作过程中，你有几个按键可以按的：

**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』这个关键字；

:f ：立刻显示出档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。

要离开 more 这个指令的显示工作，可以按下 q 就能够离开了。而要向下翻页，就使用空白键即可。 比较有用的是搜寻字符串的功能，举例来说，我们使用『 more /etc/man\_db.conf 』来观察该档案， 若想要在该档案内搜寻 MANPATH 这个字符串时，可以这样做：

[root@study ~]# more /etc/man\_db.conf

#

#

# This file is used by the man-db package to configure the man and cat paths.

# It is also used to provide a manpath for those without one by examining

# their PATH environment variable. For details see the manpath(5) man page.

#

....(中间省略)....

/MANPATH <== 输入了 / 之后，游标就会自动跑到最底下一行等待输入！

如同上面的说明，输入了 / 之后，游标就会跑到最底下一行，并且等待你的输入， 你输入了字符串并按下[enter]之后，嘿嘿！ more 就会开始向下搜寻该字符串囉～而重复搜寻同一个字符串， 可以直接按下 n 即可啊！最后，不想要看了，就按下 q 即可离开 more 啦！

2）less (一页一页来回查看)

[root@study ~]# less /etc/man\_db.conf

#

#

# This file is used by the man-db package to configure the man and cat paths.

# It is also used to provide a manpath for those without one by examining

# their PATH environment variable. For details see the manpath(5) man page.

#

.....(中间省略).....

: <== 这里可以等待你输入指令！

**less 的用法比起 more 又更加的有弹性**，怎么说呢？在 more 的时候，我们并没有办法向前面翻， 只能往后面看，但若使用了 less 时，呵呵！就可以使用 [pageup] [pagedown] 等按键的功能来往前往后翻看文件，你瞧，是不是更容易使用来观看一个档案的内容了呢！

除此之外，在 less 里头可以拥有更多的『搜寻』功能喔！不止可以向下搜寻，也可以向上搜寻～ 实在是很不错用～基本上，可以输入的指令有：

空白键 ：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！)

g ：前进到这个资料的第一行去；

G ：前进到这个资料的最后一行去 (注意大小写)；

q ：离开 less 这个程序；

查阅档案内容还可以进行搜寻的动作～瞧～ less 是否很不错用啊！ 其实 less 还有很多的功能喔！详细的使用方式请使用 man less 查询一下啊！ ^\_^

你是否会觉得 less 使用的画面与环境与 man page 非常的类似呢？没错啦！因为man这个指令就是呼叫 less 来显示说明文件的内容的！ 现在你是否觉得 less 很重要呢？ ^\_^

3 资料截取

我们可以将输出的资料作一个最简单的撷取，那就是取出档案前面几行 (head) 或取出后面几行 (tail) 文字的功能。 不过，要注意的是， head 与 tail 都是以『行』为单位来进行资料撷取的喔！

1）head (取出前面几行)

[root@study ~]# 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

/etc/passwd

[root@study ~]# head /etc/man\_db.conf

# 预设的情况中，显示前面十行！若要显示前 20 行，就得要这样：

[root@study ~]# head -n 20 /etc/man\_db.conf

范例：如果后面100行的资料都不打印，只打印/etc/man\_db.conf的前面几行，该如何是好？

[root@study ~]# head -n -100 /etc/man\_db.conf

head 的英文意思就是『头』啦，那么这个东西的用法自然就是显示出一个档案的前几行囉！ 没错！就是这样！若没有加上 -n 这个选项时，预设只显示十行，若只要一行呢？那就加入『 head -n 1 filename 』即可！

另外那个 -n 选项后面的参数较有趣，如果接的是负数，例如上面范例的-n -100时，代表列前的所有行数， 但不包括后面100行。举例来说 CentOS 7.1 的 /etc/man\_db.conf 共有131行，则上述的指令『head -n -100 /etc/man\_db.conf』 就会列出前面31行，后面100行不会打印出来了。这样说，比较容易懂了吧？ ^\_^

2）tail (取出后面几行)

[root@study ~]# 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@study ~]# tail /etc/man\_db.conf

# 预设的情况中，显示最后的十行！若要显示最后的 20 行，就得要这样：

[root@study ~]# tail -n 20 /etc/man\_db.conf

范例一：如果不知道/etc/man\_db.conf有几行，却只想列出100行以后的资料时？

[root@study ~]# tail -n +100 /etc/man\_db.conf

范例二：持续侦测/var/log/messages的内容

[root@study ~]# 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！

有 head 自然就有 tail ( 尾巴 ) 囉！没错！这个 tail 的用法跟 head 的用法差不多类似，只是显示的是后面几行就是了！预设也是显示十行，若要显示非十行，就加 -n number 的选项即可。

范例一的内容就有趣啦！其实与head -n -xx有异曲同工之妙。当下达『tail -n +100 /etc/man\_db.conf』 代表该档案从100行以后都会被列出来，同样的，在man\_db.conf共有131行，因此第100~131行就会被列出来啦！ 前面的99行都不会被显示出来喔！

至于范例二中，由于/var/log/messages随时会有资料写入，你想要让该档案有资料写入时就立刻显示到荧幕上， 就利用 -f 这个选项，他可以一直侦测/var/log/messages这个档案，新加入的资料都会被显示到荧幕上。 直到你按下[ctrl]-c才会离开tail的侦测喔！由于 messages 必须要 root 权限才能看，所以该范例得要使用 root 来查询喔！

例题：

假如我想要显示 /etc/man\_db.conf 的第 11 到第 20 行呢？

答：

这个应该不算难，想一想，在第 11 到第 20 行，那么我取前 20 行，再取后十行，所以结果就是：『 head -n 20 /etc/man\_db.conf | tail -n 10 』，这样就可以得到第 11 到第 20 行之间的内容了！

这两个指令中间有个管线 (|) 的符号存在，这个管线的意思是：『前面的指令所输出的讯息，请透过管线交由后续的指令继续使用』的意思。 所以， head -n 20 /etc/man\_db.conf 会将档案内的 20 行取出来，但不输出到荧幕上，而是转交给后续的 tail 指令继续处理。 因此 tail 『不需要接档名』，因为 tail 所需要的资料是来自于 head 处理后的结果！这样说，有没有理解？

更多的管线命令，我们会在第三篇继续解释的！

例题：

承上一题，那如果我想要列出正确的行号呢？就是荧幕上仅列出 /etc/man\_db.conf 的第 11 到第 20 行，且有行号存在？

答：

我们可以透过 cat -n 来带出行号，然后再透过 head/tail 来撷取资料即可！所以就变成如下的模样了：

cat -n /etc/man\_db.conf | head -n 20 | tail -n 10

4 非纯文字档： od

我们上面提到的，都是在查阅纯文字档的内容。那么万一我们想要查阅非文字档，举例来说，例如 /usr/bin/passwd 这个执行档的内容时， 又该如何去读出信息呢？事实上，由于执行档通常是 binary file ，使用上头提到的指令来读取他的内容时， 确实会产生类似乱码的资料啊！那怎么办？没关系，我们可以利用 od 这个指令来读取喔！

[root@study ~]# 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 ：利用预设的字符来输出；

c ：使用 ASCII 字符来输出

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

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

o[size] ：利用八进位(octal)来输出资料，每个整数占用 size bytes ；

x[size] ：利用十六进位(hexadecimal)来输出资料，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@study ~]# od -t c /usr/bin/passwd

0000000 177 E L F 002 001 001 \0 \0 \0 \0 \0 \0 \0 \0 \0

0000020 003 \0 > \0 001 \0 \0 \0 364 3 \0 \0 \0 \0 \0 \0

0000040 @ \0 \0 \0 \0 \0 \0 \0 x e \0 \0 \0 \0 \0 \0

0000060 \0 \0 \0 \0 @ \0 8 \0 \t \0 @ \0 035 \0 034 \0

0000100 006 \0 \0 \0 005 \0 \0 \0 @ \0 \0 \0 \0 \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 byes (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@study ~]# od -t oCc /etc/issue

0000000 134 123 012 113 145 162 156 145 154 040 134 162 040 157 156 040

\ S \n K e r n e l \ r o n

0000020 141 156 040 134 155 012 012

a n \ m \n \n

0000027

# 如上所示，可以发现每个字符可以对应到的数值为何！要注意的是，该数值是 8 进位喔！

# 例如 S 对应的记录数值为 123 ，转成十进制：1x8^2+2x8+3=83。

利用这个指令，可以将 data file 或者是 binary file 的内容资料给他读出来喔！ 虽然读出的来数值预设是使用非文字档，亦即是 16 进位的数值来显示的， 不过，我们还是可以透过 -t c 的选项与参数来将资料内的字符以 ASCII 类型的字符来显示， 虽然对于一般使用者来说，这个指令的用处可能不大，但是对于工程师来说， 这个指令可以将 binary file 的内容作一个大致的输出，他们可以看得出东西的啦～ ^\_^

如果对纯文字档使用这个指令，你甚至可以发现到 ASCII 与字符的对照表！非常有趣！ 例如上述的范例二，你可以发现到每个英文字 S 对照到的数字都是 123，转成十进制你就能够发现那是 83 囉！ 如果你有任何程序语言的书，拿出来对照一下 ASCII 的对照表，就能够发现真是正确啊！呵呵！

例题：

我不想找 google，想要立刻找到 password 这几个字的 ASCII 对照，该如何透过 od 来判断？

答：

其实可以透过刚刚上一个小节谈到的管线命令来处理！如下所示：

echo password | od -t oCc

echo 可以在荧幕上面显示任何信息，而这个信息不由荧幕输出，而是传给 od 去继续处理！就可以得到 ASCII code 对照囉！

/etc/inittab 系统启动进入文本还是图形，配置文件（系统启动进到哪一个级别）

/etc/man.config 帮助文档的配置文件

/etc/rc.d/rc.sysinit shellscripts shell的脚本，是系统中非常重要的一个文件，

**/etc/httpd/conf/httpd.conf apache的配置文件(www)**

相反

#cat 建议查看内容不超过一屏的文件

同这个相反一个命令：tac 倒着显示文件内容

#more 空格后翻,内容看完就回到命令提示符

#less 查看文件功能要强一些，同more相比较，它可以前翻，即使文件内容显示已经结束，也不会回到命令提示符，**需要按q退出**

**#head 显示文件前10行**

**/etc/passwd 系统中用户信息文件**

#head –n 5 文件 显示前5行

**#tail 同head的用法一样,显示后10行 还有一个非常重要的选项,-f**

/etc/passwd

上面命令查看的都是ASCII 文件🡪记事本文件

利用编辑器vim

[root@localhost cpp]# cat /etc/rc.sysinit

说明：

Less用q退出

Vi 退出 shift+zz

[root@localhost cpp]# cat /etc/rc.sysinit

[root@localhost cpp]#more /etc/rc.sysinit

[root@localhost cpp]#less /etc/rc.sysinit

[root@localhost cpp]#vi /etc/rc.sysinit

复制时是否包含目录本身

#ls pwd cd mkdir rmdir cp mv rm

5 文件查找命令

一、命令简介

  Linux whereis命令用于查找文件。该指令会在特定目录中查找符合条件的文件。这些文件应属于原始代码、二进制文件，或是帮助文件。该指令只能用于查找二进制文件、源代码文件和man手册页，一般文件的定位需使用locate命令。

  Linux which命令用于查找文件。which指令会在环境变量$PATH设置的目录里查找符合条件的文件。

二、使用示例

1、查找bash命令位置

[bdsc@centos7 ~]$ which bash

/bin/bash

[bdsc@centos7 ~]$ whereis bash

bash: /usr/bin/bash /usr/share/man/man1/bash.1.gz

2、whereis命令只查找二进制文件路径

[bdsc@centos7 ~]$ whereis -b bash

bash: /usr/bin/bash

3、whereis命令只查找帮助文档路径

[bdsc@centos7 ~]$ whereis -m bash

bash: /usr/share/man/man1/bash.1.gz

4、which查找并显示所有匹配项

[test@s101 ~]$ which -a bash

/bin/bash

/usr/bin/bash

三、使用语法及参数说明

1、whereis用法

用法：whereis [选项] 文件

2、which用法

用法：which [选项] 命令

### **3、whereis参数说明**

| **参数** | **参数说明** |
| --- | --- |
| -b | 只搜索二进制文件 |
| -B <目录> | 定义二进制文件查找路径 |
| -m | 只搜索 man 手册 |
| -M <目录> | 定义 man 手册查找路径 |
| -s | 只搜索源代码 |
| -S <目录> | 定义源代码查找路径 |
| -f | 终止 <目录> 参数列表 |
| -u | 搜索不常见记录 |
| -l | 输出有效查找路径 |

### **4、which参数说明**

| **参数** | **参数说明** |
| --- | --- |
| –version, -[vV] | 查看命令版本 |
| –help, | 查看命令帮助信息 |
| –skip-dot | 跳过路径中以.点开头的目录 |
| –skip-tilde | 跳过路径中以~波浪号开头的目录。 |
| –show-dot | 不要在输出中将点扩展到当前目录 |
| –show-tilde | 为非根目录的主目录输出波浪号 |
| –tty-only | 如果不在tty上，请停止处理右侧的选项 |
| –all, -a | 打印路径中的所有匹配项，而不只是第一个匹配项 |
| –read-alias, -i | 从stdin读取别名列表 |
| –skip-alias | 忽略选项-read-alias；不从标准输入读 |
| –read-functions | 从stdin读取shell函数 |
| –skip-functions | 忽略选项–read-functions; 不从标准输入读 |

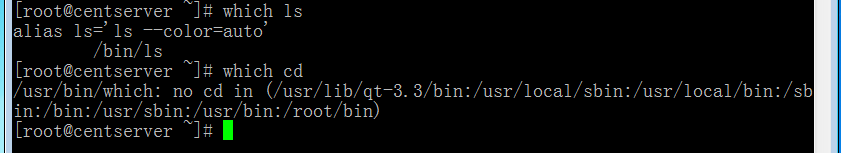
### 1）which

功能：只能查可执行命令，查找的依据是**PATH**,**常用来查找我们经常执行的命令在哪儿,也就是查找到的是外部命令，同时要查找这个系统中是否存在同名称的别名**

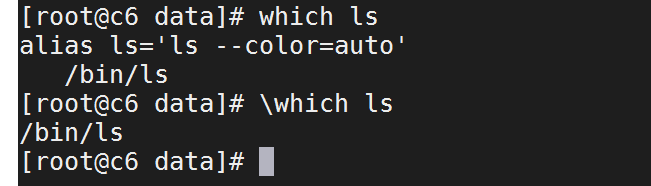
which命令用于查找可执行文件的位置，它会根据文件名搜索系统路径中的文件，并返回第一个找到的可执行文件的位置信息，如果没有找到匹配的文件，which命令会返回一个错误信息。

**内部命令cd**

**ls**



Which本身在系统中是一个别名，如果搜索可执行文件只想得到路径，则可以使用原命令

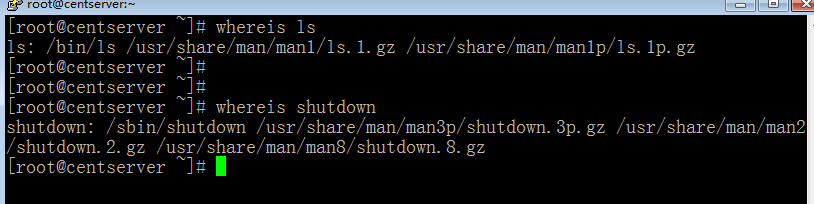


### 2）whereis

功能：也是查找命令的,

1. 这个搜索的是命令的位置,
2. 但是它不搜索别名,
3. 搜索这个命令的帮助文档的位置
4. 搜索配置文件

whereis命令用于查找二进制、源代码、手册页等类型的文件，它会根据文件名搜索系统路径中的文件，并返回文件的位置信息，whereis命令支持多种选项，如-b、-m、-s等，可以满足用户不同的需求。



3）locate

locate命令用于查找文件或目录。locate命令要比find -name快得多，原因在于它不搜索具体目录，而是搜索一个数据库/var/lib/mlocate/mlocate.db。这个数据库中含有本地所有文件信息。Linux系统自动创建这个数据库，并且每天自动更新一次。因此，我们在用whereis 和 locate查找文件时，有时会找到已经被删除的数据；或者刚刚建立的文件，却无法查找到，原因就是因为数据库文件没有被更新。为了避免这种情况，可以在使用locate之前，先使用updatedb命令，手动更新数据库。整个locate工作其实是由4部分组成的：

/usr/bin/updatedb: 主要用来更新数据库，通过crontab自动完成的

/usr/bin/locate: 查询文件位置

/etc/updatedb.conf: updatedb的配置文件

/var/lib/mlocate/mlocate.db: 存放文件信息的文件

locate会读取由updatedb准备好的一个或多个数据库，然后将满足匹配PATTERN的文件写到标准输出，每行一个文件名。假如并未指定--regex选项，则PATTERN可以包含通配符。假如PATTERN中并未包含任何通配符，则locate命令以\*PATTERN\*模式进行查找。

默认情况下，locate命令并不会检查数据库中的文件是否仍然存在，也不会报告在上一次更新数据库之后产生的文件。

#### 1) 用法

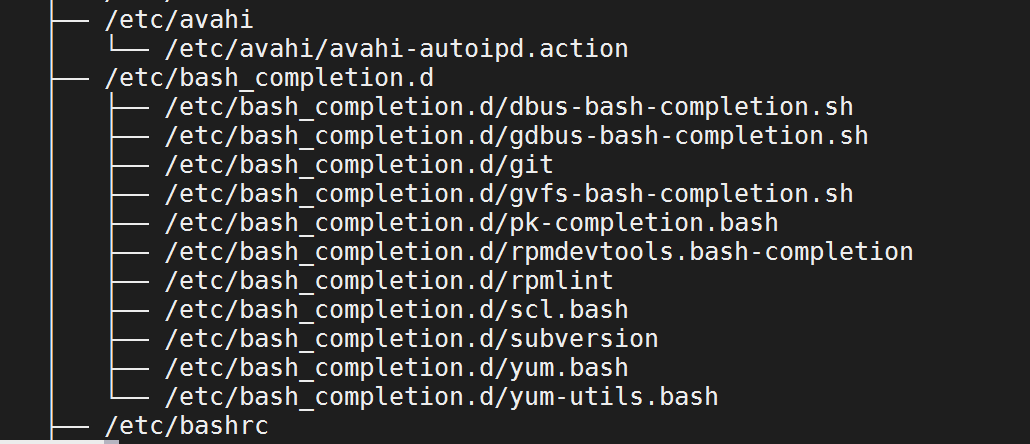
locate [OPTION]... [PATTERN]...

功能：默认情况下会搜索整个完整路径中包含有搜索的字符串的文件及目录

即dirname+basename

Fullname

/var/spool/mail/tom



#### 选项

 -A, --all               打印所有匹配PATTERNs的文件

 -b, --basename          只名称与PATTERNs匹配的文件（例如/test/test.txt，则basename为test.txt)

 -c, --count             只打印匹配到的文件数目，而不是文件名本身

 -d, --database DBPATH   用DBPATH来代替默认的数据库。DBPATH是一个以：（冒号）分割的数据库名称列表

 -e, --existing          只打印在调用locate时存在的文件

 -L, --follow            当和-e选项一起使用时，如果该文件是一个软链接文件，则会获取到该软链接对应的实际文件（此为locate命令默认行文）。

 -h, --help              打印帮助文档信息

**-i, --ignore-case       当进行文件模式匹配时，忽略大小写**

 -l, --limit, -n LIMIT   当成功匹配到LIMIT数量的文件时，退出

 -P, --nofollow, -H      当和-e选项一起使用时，如果该文件是一个软链接文件，此时不跟随链接的实际文件

 -0, --null              输出时以ASCII NUL作为分隔符

 -S, --statistics        获得每一个搜索数据库的统计信息

 -q, --quiet             当在读取和处理数据库时，遇到错误也不打印相关信息

 -r, --regexp REGEXP     查找一个基本的模式匹配REGEXP。此选项情况下，不能出现PATTERNs

 -V, --version           打印locate的版本信息和license信息

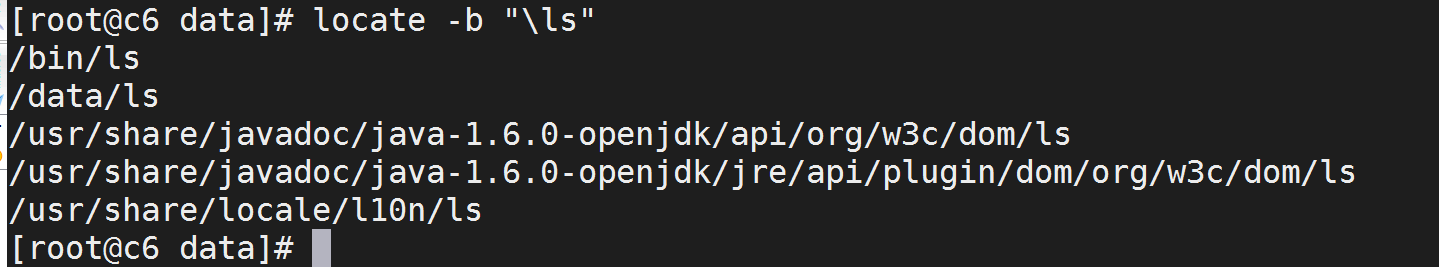
 -w, --wholename         匹配完整路径名（默认）

例如精确查找一个文件名称为NAME（并不是\*NAME\*)的文件,可以使用如下：

 #**locate -b '\NAME'**

应为\本身是一个通配字符，因此这里会禁止隐式的转化为\*NAME\*.

如果不带\ ,则表示模糊搜索文件名中包含中搜索字符串



#### 使用范例

**1) 搜索目录下所有以sh开头的文件**

 # locate /etc/sh

 /etc/shadow

 /etc/shadow-

 /etc/shells

**2) 包含某个字符串的相关文件**

 # locate passwd

 /etc/passwd

 /etc/passwd-

 ....

**3) 限定显示数量**

如果显示的内容过多，可以使用-n选项来限定显示数量。

 # locate -n 5 passwd

 /etc/passwd

 /etc/passwd-

 /etc/pam.d/passwd

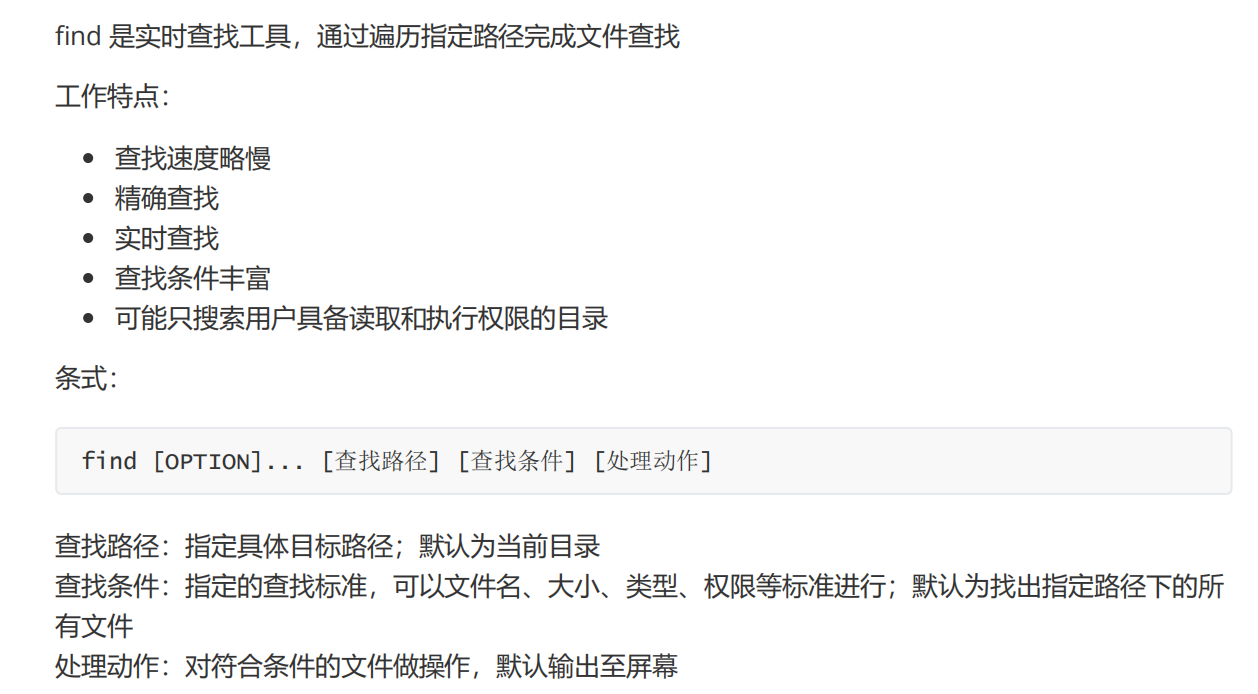
 /etc/security/opasswd

 /usr/bin/gpasswd

1. find

**老老实实找硬盘**

**按basename查找**



#### 1 根据文件名和inode查找



**选项与参数：**

#### 2 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@study ~]# 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@study ~]# find /etc -newer /etc/passwd

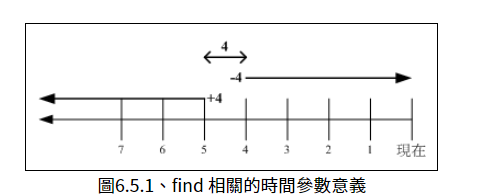
# -newer 用在分辨两个档案之间的新旧关系是很有用的！

时间参数真是挺有意思的！我们现在知道 atime, ctime 与 mtime 的意义，如果你想要找出一天内被更动过的文件名称， 可以使用上述范例一的作法。但如果我想要找出『4天内被更动过的档案档名』呢？那可以使用『 find /var -mtime -4 』。那如果是『4天前的那一天』就用『 find /var -mtime 4 』。有没有加上『+, -』差别很大喔！我们可以用简单的图示来说明一下：

find 相关的时间参数意义

图6.5.1、find 相关的时间参数意义

图中最右边为目前的时间，越往左边则代表越早之前的时间轴啦。由图 6.5.1 我们可以清楚的知道：



+4代表大于等于5天前的档名：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

非常有趣吧！你可以在 /var/ 目录下搜寻一下，感受一下输出档案的差异喔！再来看看其他 find 的用法吧！

选项与参数：

#### 3. 根据属主、属组查找

-user USERNAME：查找属主为指定用户(UID)的文件

-group GRPNAME: 查找属组为指定组(GID)的文件

-uid UserID：查找属主为指定的UID号的文件

-gid GroupID：查找属组为指定的GID号的文件

-nouser：查找没有属主的文件

-nogroup：查找没有属组的文件

范例三：搜寻 /home 底下属于 dmtsai 的档案

[root@study ~]# find /home -user dmtsai

# 这个东西也很有用的～当我们要找出任何一个使用者在系统当中的所有档案时，

# 就可以利用这个指令将属于某个使用者的所有档案都找出来喔！

范例四：搜寻系统中不属于任何人的档案

[root@study ~]# find / -nouser

# 透过这个指令，可以轻易的就找出那些不太正常的档案。如果有找到不属于系统任何人的档案时，

# 不要太紧张，那有时候是正常的～尤其是你曾经以原始码自行编译软件时。

如果你想要找出某个使用者在系统底下建立了啥咚咚，使用上述的选项与参数，就能够找出来啦！ 至于那个 -nouser 或 -nogroup 的选项功能中，除了你自行由网络上面下载档案时会发生之外， 如果你将系统里面某个账号删除了，但是该账号已经在系统内建立很多档案时，就可能会发生无主孤魂的档案存在！ 此时你就得使用这个 -nouser 来找出该类型的档案囉！

选项与参数：

#### 4 与档案权限及名称有关的参数：

-size [+-]SIZE： 搜寻比 SIZE 还要大(+)或小(-)的档案。这个 SIZE 的规格有：

c: 代表 byte， k: 代表 1024bytes。所以，要找比 50KB

还要大的档案，就是『 -size +50k 』,这个表示小于50K文件，

需要注意的是：目录还是按照目录文件来处理的

-type TYPE 搜寻档案的类型为 TYPE 的，类型主要有：

一般正规档案 (f),

设备文件 (b, c),

目录 (d),

软链接文件 (l),

socket (s), 及 FIFO (p) 等属性。

-perm mode ：搜寻档案权限『**刚好等于**』 mode 的档案，这个 mode 为类似 chmod

的属性值，举例来说， -rwsr-xr-x 的属性为 4755 ！

-perm -mode ：搜寻档案权限『**必须要全部囊括 mode 的权限**』的档案，举例来说，

我们要搜寻 -rwxr--r-- ，亦即 0744 的档案，使用 -perm -0744，

当一个档案的权限为 -rwsr-xr-x ，亦即 4755 时，也会被列出来，

因为 -rwsr-xr-x 的属性已经囊括了 -rwxr--r-- 的属性了。

-perm /mode ：搜寻档案权限『包含任一 mode 的权限』的档案，举例来说，我们搜寻

-rwxr-xr-x ，亦即 -perm /755 时，但一个档案属性为 -rw-------

也会被列出来，因为他有 -rw.... 的属性存在！

范例五：找出档名为 passwd 这个档案

[root@study ~]# find / -name passwd

范例五-1：找出档名包含了 passwd 这个关键字的档案

[root@study ~]# find / -name "\*passwd\*"

# 利用这个 -name 可以搜寻档名啊！预设是完整档名，如果想要找关键字，

# 可以使用类似 \* 的任意字符来处理

范例六：找出 /run 目录下，档案类型为 Socket 的档名有哪些？

[root@study ~]# find /run -type s

# 这个 -type 的属性也很有帮助喔！尤其是要找出那些怪异的档案，

# 例如 socket 与 FIFO 档案，可以用 find /run -type p 或 -type s 来找！

范例七：搜寻档案当中含有 SGID 或 SUID 或 SBIT 的属性

[root@study ~]# find / -perm /7000

# 所谓的 7000 就是 ---s--s--t ，那么只要含有 s 或 t 的就列出，所以当然要使用 /7000，

# 使用 -7000 表示要同时含有 ---s--s--t 的所有三个权限。而只需要任意一个，就是 /7000 ～了乎？

上述范例中比较有趣的就属 -perm 这个选项啦！他的重点在找出特殊权限的档案囉！ 我们知道 SUID 与 SGID 都可以设定在二进制程序上，假设我想要找出来 /usr/bin, /usr/sbin 这两个目录下， 只要具有 SUID 或 SGID 就列出来该档案，你可以这样做：

[root@study ~]# find /usr/bin /usr/sbin -perm /6000

因为 SUID 是 4 分，SGID 2 分，总共为 6 分，因此可用 /6000 来处理这个权限！ 至于 find 后面可以接多个目录来进行搜寻！另外， find 本来就会搜寻次目录，这个特色也要特别注意喔！ 最后，我们再来看一下 find 还有什么特殊功能吧！

*选项与参数：*

选项与参数：

#### 5 额外可进行的动作：

**处理动作**

-print：默认的处理动作，显示至屏幕

-ls：类似于对查找到的文件执行“ls -l”命令

-delete：删除查找到的文件

-flfls fifile：查找到的所有文件的长格式信息保存至指定文件中

-ok COMMAND {} \; 对查找到的每个文件执行由COMMAND指定的命令，对于每个文件执行命令之

前，都会交互式要求用户确认

-exec COMMAND {} \; 对查找到的每个文件执行由COMMAND指定的命令

{}: 用于引用查找到的文件名称自身

#备份配置文件，添加.orig这个扩展名

find -name ".conf" -exec cp {} {}.orig \;

#提示删除存在时间超过３天以上的joe的临时文件

find /tmp -ctime +3 -user joe -ok rm {} \;

#在主目录中寻找可被其它用户写入的文件

find ~ -perm -002 -exec chmod o-w {} \;

#查找/etc下的权限为644，后缀为conf的普通文件，并显示出来

find /data –type f -perm 644 -name ".sh" –exec chmod 755 {} \;

#find /etc -type f -perm 644 -name "\*.conf" -exec ls -l {} \;

#find /etc -type f -perm 644 -name "\*g.conf" -exec ls -l {} \;

-exec command ：command 为其他指令，-exec 后面可再接额外的指令来处理搜寻到的结果。

-print ：将结果打印到荧幕上，这个动作是预设动作！

范例八：将上个范例找到的档案使用 ls -l 列出来～

[root@study ~]# find /usr/bin /usr/sbin -perm /7000 -exec ls -l {} \;

# 注意到，那个 -exec 后面的 ls -l 就是额外的指令，指令不支援命令别名，

# 所以仅能使用 ls -l 不可以使用 ll 喔！注意注意！

范例九：找出系统中，大于 1MB 的档案

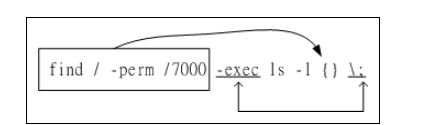
[root@study ~]# find / -size +1M

find 的特殊功能就是能够进行额外的动作(action)。我们将范例八的例子以图解来说明如下：

find 相关的额外动作

图6.5.2、find 相关的额外动作

该范例中特殊的地方有 {} 以及 \; 还有 -exec 这个关键字，这些东西的意义为：



**{} 代表的是『由 find 找到的内容』，如上图所示，find 的结果会被放置到 {} 位置中；**

**-exec 一直到 \; 是关键字，代表 find 额外动作的开始 (-exec) 到结束 (\;) ，在这中间的就是 find 指令内的额外动作。 在本例中就是『 ls -l {} 』囉！**

**因为『 ; 』在 bash 环境下是有特殊意义的，因此利用反斜线来跳脱。**

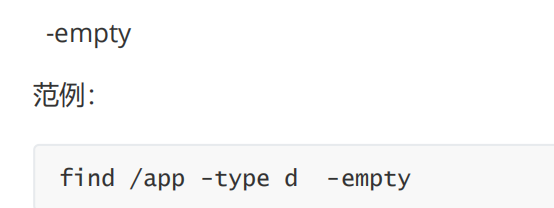
**透过图 6.5.2 你应该就比较容易了解 -exec 到 \; 之间的意义了吧！**

如果你要找的档案是具有特殊属性的，例如 SUID 、档案拥有者、档案大小等等， 那么利用 locate 是没有办法达成你的搜寻的！此时 find 就显的很重要啦！ 另外，find 还可以利用万用字元来找寻档名呢！举例来说，你想要找出 /etc 底下档名包含 httpd 的档案， 那么你就可以这样做：

[root@study ~]# find /etc -name '\*httpd\*'

不但可以指定搜寻的目录(连同次目录)，并且可以利用额外的选项与参数来找到最正确的档名！真是好好用！ 不过由于 find 在寻找资料的时后相当的操硬盘！所以没事情不要使用 find 啦！有更棒的指令可以取代呦！那就是上面提到的 whereis 与 locate 囉！

**空文件或者目录**



**组合条件**

与：-a

或：-o

非：-not !

Wc 统计 文件中有多少行，多少个单词，多少个字符

Wc

## Linux文本处理三剑客 grep sed awk

【ps】 这个大家必须要知道有这么个东东

参见第6章的bash shell

awk、grep、sed是linux操作文本的三大利器，合称文本三剑客，也是必须掌握的linux命令之一。三者的功能都是处理文本，但侧重点各不相同，其中属awk功能最强大，但也最复杂。grep更适合单纯的查找或匹配文本，sed更适合编辑匹配到的文本，awk更适合格式化文本，对文本进行较复杂格式处理。

正则表达式

1.正则表达式主要分为两类：

基本的正则表达式

扩展的正则表达式

下面我们来看一下基本的正则表达式来感受一下正则表达式的强大

$ :用来匹配模式的最右则如"ksy$"是用来匹配

^ :用来匹配模式的最左侧如 "^ksy"是用来匹配以单词ksy开头的行。

$^:表示匹配空格行

\*:用来匹配前一个字符0次或者1次以上

.\*:组合符用来匹配任意长度的字符串

^.\*:用来匹配任意多个字符开头的内容

.\*$:用来匹配任意多个字符结尾的内容

[abc]:用来匹配集合里面任意一个字符

[^abc]:表示用来匹配除了集合当中的任意字符

. :用来匹配单个任意字符并且有且只能匹配一个字符不能匹配空行

\:用来转义让有特殊含义的字符现出原型比如说.就代表小数点

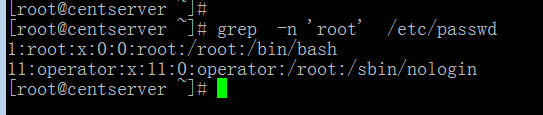
1. grep

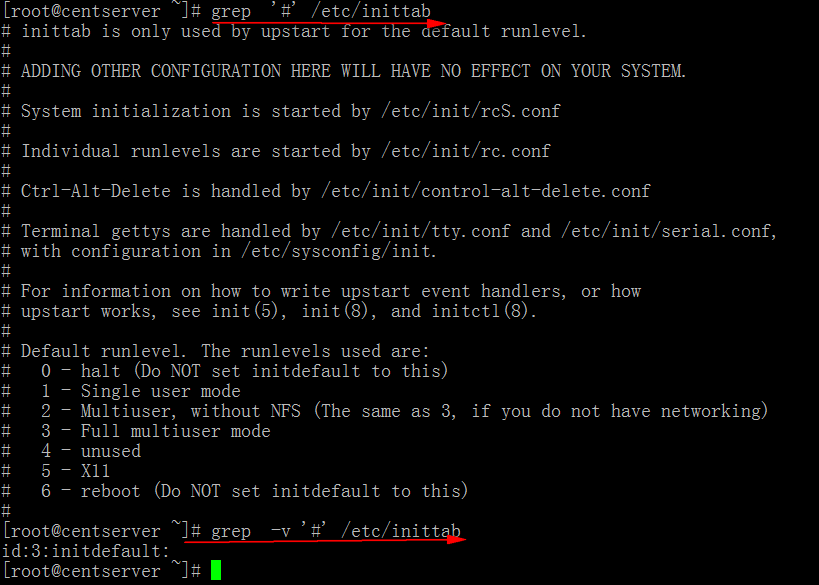
grep cut awk sed

文件内容查找

是在指定的文件里面去找,相当于word文件中的ctrl+F,

它是以行为搜索单位,就是说结果是查找在这个文件中包含了搜索关键字的行显示出来





**实例：**

将vsftpd的配置文件进行处理，只保留有效的行

-v 表示反选

-w 表示单词了

^ 表示以什么开头 ^#表示以#号开头

$ 表示以什么结尾

“^$” 表示空行，因为中间什么都没有

#grep -v "#" /etc/vsftpd/vsftpd.conf | grep -v "^$"

1. wc命令

统计一个文件单词，多少行，多少字符

经常结合管道命令一起使用来统计结果

别名命令alias unalias

显示系统时间date

更改系统时间 hwclock

文件内容查看命令

# cat (tac) more less head tail od(用来查看二进制文件滴)

利用编辑器查看文件内容 vim 退出shift + zz

/etc/rc.sysinit

/etc/inittab

/etc/passwd

6 实验安排

1 分别利用这几个命令 #cat more less head tail，

查看 /etc/inittab /etc/rc.d/rc.sysinit /etc/passwd

[root@centserver bin]# cat /etc/rc.d/rc.sysinit

#od /bin/ls

**利用nano将/etc/inittab最后一行的3改成5保存退出**

2、分别利用which whereis 两个命令查找， ls mkdir cd比较结果有什么不同

#which ls

#which mkdir

#whereis ls

#whereis mkdir

**如果利用which查找不显示别名应该怎么查找？**

1. 利用grep检索在/etc/passwd中检索root所在行

#grep “root” /etc/passwd

思考

如果要把下面文件有效行找出来，利用grep怎么找

/etc/vsftpd/vsftpd.conf

1. 利用locate直接查找如下三个关键字

ls inittab ifcfg

（updatedb,更新数据库）

再利用locate按名字查找,[提示带-b选项 ] “\名字” 精确查找

1. 利用find在/etc/目录下查找网卡配置文件以及passwd关键字，提示网卡配置文件在rhel7以后是ifcfg-en开头的

6 利用find查找/etc/目录下的所有以 .conf结尾的文件

6 显示/etc/passwd前11到第20行内容

#head -n 20 /etc/passwd | tail -n 10