# linux grep命令详解

2016-07-02 Linux爱好者

（点击上方公众号，可快速关注）

> 来源：ggjucheng
> 链接：http://www.cnblogs.com/ggjucheng/archive/2013/01/13/2856896.html

## 简介

grep (global search regular expression(RE) and print out the line,全面搜索正则表达式并把行打印出来)是一种强大的文本搜索工具，它能使用正则表达式搜索文本，并把匹配的行打印出来。

Unix的grep家族包括grep、egrep和fgrep。egrep和fgrep的命令只跟grep有很小不同。egrep是grep的扩展，支持更多的re元字符， fgrep就是fixed grep或fast grep，它们把所有的字母都看作单词，也就是说，正则表达式中的元字符表示回其自身的字面意义，不再特殊。linux使用GNU版本的grep。它功能更强，可以通过-G、-E、-F命令行选项来使用egrep和fgrep的功能。

## grep常用用法

> [root@www ~]# grep [-acinv] [--color=auto] '搜寻字符串' filename
> 选项与参数：
> -a ：将 binary 文件以 text 文件的方式搜寻数据
> -c ：计算找到 '搜寻字符串' 的次数
> -i ：忽略大小写的不同，所以大小写视为相同
> -n ：顺便输出行号
> -v ：反向选择，亦即显示出没有 '搜寻字符串' 内容的那一行！
> --color=auto ：可以将找到的关键词部分加上颜色的显示喔！

将/etc/passwd，有出现 root 的行取出来

> # grep root /etc/passwd
> root:x:0:0:root:/root:/bin/bash
> operator:x:11:0:operator:/root:/sbin/nologin
> 或

```
# cat /etc/passwd | grep root
root:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
```

将/etc/passwd，有出现 root 的行取出来,同时显示这些行在/etc/passwd的行号

```
# grep -n root /etc/passwd1:root:x:0:0:root:/root:/bin/bash30:operator:x:11:0:operator:/root:/sbin/nologi
```

在关键字的显示方面，grep 可以使用 --color=auto 来将关键字部分使用颜色显示。 这可是个很不错的功能啊！但是如果每次使用 grep 都得要自行加上 --color=auto 又显的很麻烦～ 此时那个好用的 alias 就得来处理一下啦！你可以在 ~/.bashrc 内加上这行：『alias grep='grep --color=auto'』再以『 source ~/.bashrc 』来立即生效即可喔！ 这样每次运行 grep 他都会自动帮你加上颜色显示啦

将/etc/passwd，将没有出现 root 的行取出来

```
# grep -v root /etc/passwdroot:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
```

将/etc/passwd，将没有出现 root 和nologin的行取出来

```
# grep -v root /etc/passwd | grep -v nologin
root:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
```

用 dmesg 列出核心信息，再以 grep 找出内含 eth 那行,要将捉到的关键字显色，且加上行号来表示：

```
[root@www ~]# dmesg | grep -n --color=auto 'eth'247:eth0: RealTek RTL8139 at 0x
# 你会发现除了 eth 会有特殊颜色来表示之外，最前面还有行号喔！
```

在关键字的显示方面，grep 可以使用 --color=auto 来将关键字部分使用颜色显示。 这可是个很不错的功能啊！但是如果每次使用 grep 都得要自行加上 --color=auto 又显的很麻烦～ 此时那个好用的 alias 就得来处理一下啦！你可以在 ~/.bashrc 内加上这行：『alias grep='grep --color=auto'』再以『 source ~/.bashrc 』来立即生效即可喔！ 这样每次运行 grep 他都会自动帮你加上颜色显示啦

用 dmesg 列出核心信息，再以 grep 找出内含 eth 那行,在关键字所在行的前两行与后三行也一起捉出来显示

```
[root@www ~]# dmesg | grep -n -A3 -B2 --color=auto 'eth'245-PCI: setting IRQ 10
# 如上所示，你会发现关键字 247 所在的前两行及 248 后三行也都被显示出来！
# 这样可以让你将关键字前后数据捉出来进行分析啦！
```

根据文件内容递归查找目录

```
# grep  ‘energywise’  *           #在当前目录搜索带'energywise'行的文件

# grep -r  ‘energywise’  *        #在当前目录及其子目录下搜索'energywise'行的文
# grep -l -r  ‘energywise’  *     #在当前目录及其子目录下搜索'energywise'行的文
```

这几个命令很使用，是查找文件的利器。

## grep与正规表达式

### 字符类

字符类的搜索：如果我想要搜寻 test 或 taste 这两个单字时，可以发现到，其实她们有共通的 't?st' 存在～这个时候，我可以这样来搜寻：

```
[root@www ~]# grep -n 't[ae]st' regular_express.txt8:I can't finish the test.9:Oh! The soup taste good.
```

其实 [] 里面不论有几个字节，他都谨代表某『一个』字节， 所以，上面的例子说明了，我需要的字串是『tast』或『test』两个字串而已！

字符类的反向选择 [^] ：如果想要搜索到有 oo 的行，但不想要 oo 前面有 g，如下

```
[root@www ~]# grep -n '[^g]oo' regular_express.txt2:apple is my favorite food.3:Football game is not use
```

第 2,3 行没有疑问，因为 foo 与 Foo 均可被接受！

但是第 18 行明明有 google 的 goo 啊～别忘记了，因为该行后面出现了 tool 的 too 啊！所以该行也被列出来～ 也就是说， 18 行里面虽然出现了我们所不要的项目 (goo) 但是由於有需要的项目 (too) ， 因此，是符合字串搜寻的喔！

至於第 19 行，同样的，因为 gooooooogle 里面的 oo 前面可能是 o ，例如：go(ooo)oogle ，所以，这一行也是符合需求的！

字符类的连续：再来，假设我 oo 前面不想要有小写字节，所以，我可以这样写 [^abcd....z]oo ， 但是这样似乎不怎么方便，由於小写字节的 ASCII 上编码的顺序是连续的， 因此，我们可以将之简化为底下这样：

```
[root@www ~]# grep -n '[^a-z]oo' regular_express.txt3:Football game is not use
```

也就是说，当我们在一组集合字节中，如果该字节组是连续的，例如大写英文/小写英文/数字等等， 就可以使用[a-z],[A-Z],[0-9]等方式来书写，那么如果我们的要求字串是数字与英文呢？ 呵呵！就将他全部写在一起，变成：[a-zA-Z0-9]。

我们要取得有数字的那一行，就这样：

```
[root@www ~]# grep -n '[0-9]' regular_express.txt5:However, this dress is about
```

**行首与行尾字节 ^ $**

行首字符：如果我想要让 the 只在行首列出呢？ 这个时候就得要使用定位字节了！我们可以这样做：

```
[root@www ~]# grep -n '^the' regular_express.txt12:the symbol '*' is represente
```

此时，就只剩下第 12 行，因为只有第 12 行的行首是 the 开头啊～此外， 如果我想要开头是小写字节的那一行就列出呢？可以这样：

```
[root@www ~]# grep -n '^[a-z]' regular_express.txt
2:apple is my favorite food.
4:this dress doesn't fit me.
10:motorcycle is cheap than car.
12:the symbol '*' is represented as start.
18:google is the best tools for search keyword.
19:gooooooogle yes!
20:go! go! Let's go.
```

如果我不想要开头是英文字母，则可以是这样：

```
[root@www ~]# grep -n '^[^a-zA-Z]' regular_express.txt1:"Open Source" is a good
```

^ 符号，在字符类符号(括号[])之内与之外是不同的！ 在 [] 内代表『反向选择』，在 [] 之外则代表定位在行首的意义！

那如果我想要找出来，行尾结束为小数点 (.) 的那一行：

```
[root@www ~]# grep -n '\.$' regular_express.txt1:"Open Source" is a good mechan
```

特别注意到，因为小数点具有其他意义(底下会介绍)，所以必须要使用转义字符(\)来加以解除其特殊意义！

找出空白行：

```
[root@www ~]# grep -n '^$' regular_express.txt22:
```

因为只有行首跟行尾 (^$)，所以，这样就可以找出空白行啦！

**任意一个字节 . 与重复字节 ***

这两个符号在正则表达式的意义如下：

```
. (小数点)：代表『一定有一个任意字节』的意思；* (星号)：代表『重复前一个字符，
```

假设我需要找出 g??d 的字串，亦即共有四个字节， 起头是 g 而结束是 d ，我可以这样做：

```
[root@www ~]# grep -n 'g..d' regular_express.txt1:"Open Source" is a good mecha
```

因为强调 g 与 d 之间一定要存在两个字节，因此，第 13 行的 god 与第 14 行的 gd 就不会被列出来啦！

如果我想要列出有 oo, ooo, oooo 等等的数据， 也就是说，至少要有两个(含) o 以上，该如何是好？

因为 * 代表的是『重复 0 个或多个前面的 RE 字符』的意义， 因此，『o*』代表的是：『拥有空字节或一个 o 以上的字节』，因此，『 grep -n 'o*' regular_express.txt 』将会把所有的数据都列印出来终端上！

当我们需要『至少两个 o 以上的字串』时，就需要 ooo* ，亦即是：

```
[root@www ~]# grep -n 'ooo*' regular_express.txt1:"Open Source" is a good mecha
```

如果我想要字串开头与结尾都是 g，但是两个 g 之间仅能存在至少一个 o ，亦即是 gog, goog, gooog.... 等等，那该如何？

```
[root@www ~]# grep -n 'goo*g' regular_express.txt18:google is the best tools fo
```

如果我想要找出 g 开头与 g 结尾的行，当中的字符可有可无

```
[root@www ~]# grep -n 'g.*g' regular_express.txt1:"Open Source" is a good mecha
```

因为是代表 g 开头与 g 结尾，中间任意字节均可接受，所以，第 1, 14, 20 行是可接受的喔！ 这个 .* 的 RE 表示任意字符是很常见的.

如果我想要找出『任意数字』的行？因为仅有数字，所以就成为：

```
[root@www ~]# grep -n '[0-9][0-9]*' regular_express.txt5:However, this dress is
```

**限定连续 RE 字符范围 {}**

我们可以利用 . 与 RE 字符及 * 来配置 0 个到无限多个重复字节， 那如果我想要限制一个范围区间内的重复字节数呢？

举例来说，我想要找出两个到五个 o 的连续字串，该如何作？这时候就得要使用到限定范围的字符 {} 了。 但因为 { 与 } 的符号在 shell 是有特殊意义的，因此， 我们必须要使用字符 \ 来让他失去特殊意义才行。 至於 {} 的语法是这样的，假设我要找到两个 o 的字串，可以是：

```
[root@www ~]# grep -n 'o\{2\}' regular_express.txt1:"Open Source" is a good mec
```

假设我们要找出 g 后面接 2 到 5 个 o ，然后再接一个 g 的字串，他会是这样：

```
[root@www ~]# grep -n 'go\{2,5\}g' regular_express.txt18:google is the best too
```

如果我想要的是 2 个 o 以上的 goooo....g 呢？除了可以是 gooo*g ，也可以是：

```
[root@www ~]# grep -n 'go\{2,\}g' regular_express.txt18:google is the best tool
```

**扩展grep(grep -E 或者 egrep)：**

使用扩展grep的主要好处是增加了额外的正则表达式元字符集。

打印所有包含NW或EA的行。如果不是使用egrep，而是grep，将不会有结果查出。

```
# egrep 'NW|EA' testfile
      northwest           NW         Charles Main             3.0      .98      3
      eastern              EA          TB Savage                4.4      .84      5
```

对于标准grep，如果在扩展元字符前面加\，grep会自动启用扩展选项-E。

```
#grep 'NW\|EA' testfile
northwest           NW         Charles Main             3.0      .98      3        3
```

搜索所有包含一个或多个3的行。

```
# egrep '3+' testfile
# grep -E '3+' testfile
# grep '3\+' testfile
#这3条命令将会
northwest           NW         Charles Main             3.0      .98      3
```

搜索所有包含0个或1个小数点字符的行。

```
# egrep '2\.?[0-9]' testfile
# grep -E '2\.?[0-9]' testfile
# grep '2\.\?[0-9]' testfile
#首先含有2字符，其后紧跟着0个或1个点，后面再是0和9之间的数字。
western             WE          Sharon Gray              5.3      .97      5
```

搜索一个或者多个连续的no的行。

```
# egrep '(no)+' testfile
# grep -E '(no)+' testfile
# grep '\(no\)\+' testfile    #3个命令返回相同结果，
northwest           NW         Charles Main             3.0      .98      3        3
```

## 不使用正则表达式

fgrep 查询速度比grep命令快，但是不够灵活：它只能找固定的文本，而不是规则表达式。

如果你想在一个文件或者输出中找到包含星号字符的行

```
fgrep   '*' /etc/profile
for i in /etc/profile.d/*.sh ; do

或
grep -F '*' /etc/profile
for i in /etc/profile.d/*.sh ; do
```