Linux命令之grep

作为linux中最为常用的三大文本（**awk，sed，grep**）处理工具之一，掌握好其用法是很有必要的。

<https://www.cnblogs.com/flyor/p/6411140.html>

# grep命令使用方法

grep命令的常用格式为：

**grep [选项] “模式” [文件]**

grep家族总共有三个：grep，egrep，fgrep。

下面重点从选项和模式两部分单独介绍。

# 选项部分

## grep命令的常用选项：

-E ：开启扩展（Extend）的正则表达式。

**-i ：忽略大小写（ignore case）。**

　　-v ：反过来（invert），只打印没有匹配的，而匹配的反而不打印。

**-n ：显示行号**

**-w ：被匹配的文本只能是单词**，而不能是单词中的某一部分，如文本中有liker，而我搜寻的只是like，就可以使用-w选项来避免匹配liker

**-c ：显示总共有多少行被匹配到了**，而不是显示被匹配到的内容，注意如果同时使用-cv选项是显示有多少行没有被匹配到。

**-o ：只显示被模式匹配到的字符串。**

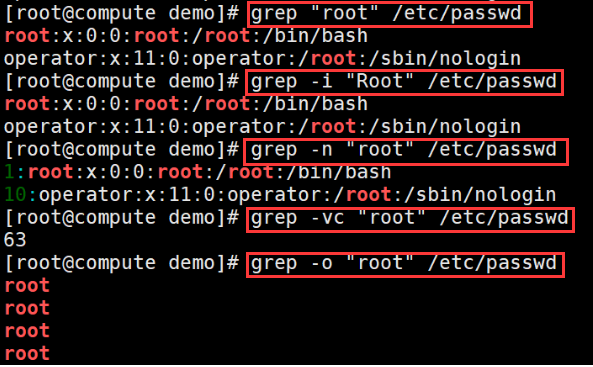
　　--color :将匹配到的内容以颜色高亮显示。

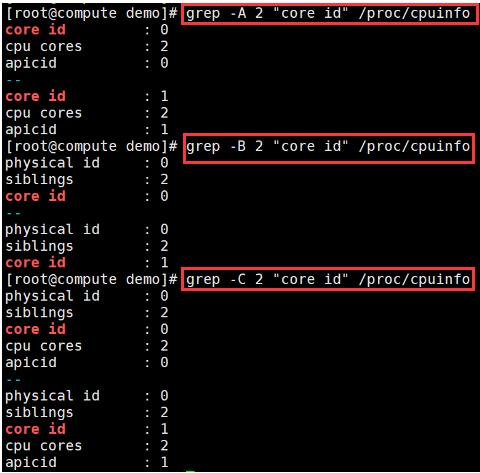
**-A n：显示匹配到的字符串所在的行及其后n行，after**

**-B n：显示匹配到的字符串所在的行及其前n行，before**

**-C n：显示匹配到的字符串所在的行及其前后各n行，context，上下文**

## 选项的使用示例





# 模式部分(重点)：正则表达式

## 匹配普通字符串

直接输入要匹配的字符串，这个可以用**fgrep（fast grep）**代替来**提高查找速度**，

比如我要匹配一下hello.c文件中printf的个数：**grep -c "printf" hello.c**

## 正则表达式基本使用

使用基本正则表达式，下面谈关于基本正则表达式的使用：

　　　　匹配字符：

　　　　　　. ：任意一个字符。

　　　　　　[abc] ：表示匹配一个字符，这个字符必须是abc中的一个。

　　　　　　[a-zA-Z] ：表示匹配一个字符，这个字符必须是a-z或A-Z这52个字母中的一个；

　　　　　　[^123] ：匹配一个字符，这个字符是除了1、2、3以外的所有字符。

　　 对于一些常用的字符集，系统做了定义：

　　　　　　[A-Za-z] 等价于 **[[:alpha:]]**

　　　　　　[0-9] 等价于 [[:digit:]]

　　　　　　[A-Za-z0-9] 等价于 [[:alnum:]]

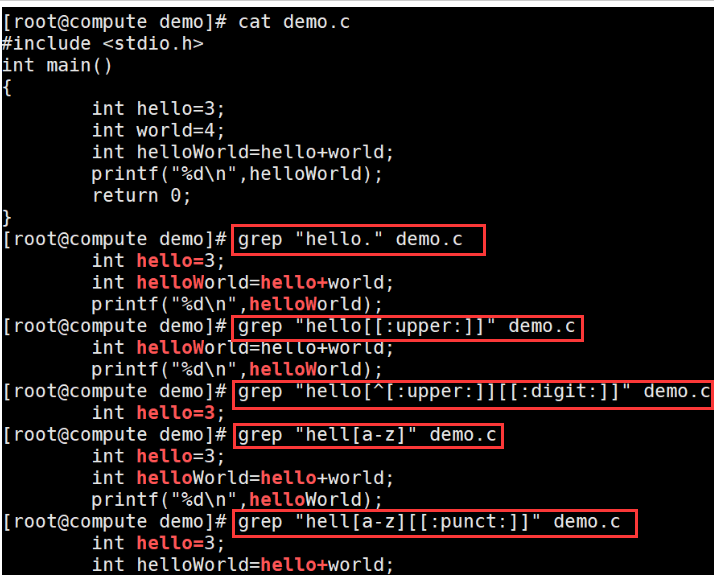
　　　　　　tab,space 等空白字符 [[:space:]]

　　　　　　[A-Z] 等价于 [[:upper:]]

　　　　　　[a-z] 等价于 [[:lower:]]

　　　　　　标点符号 [[:punct:]]

使用示例：

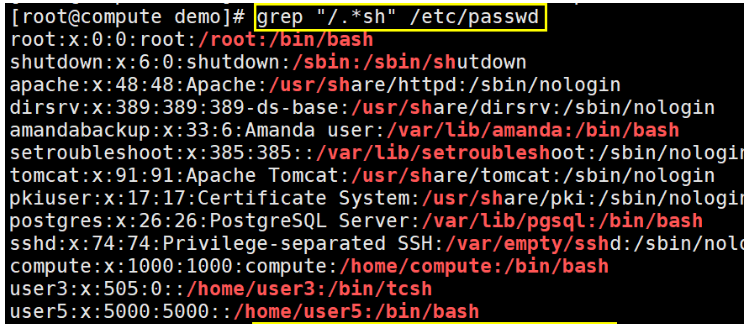


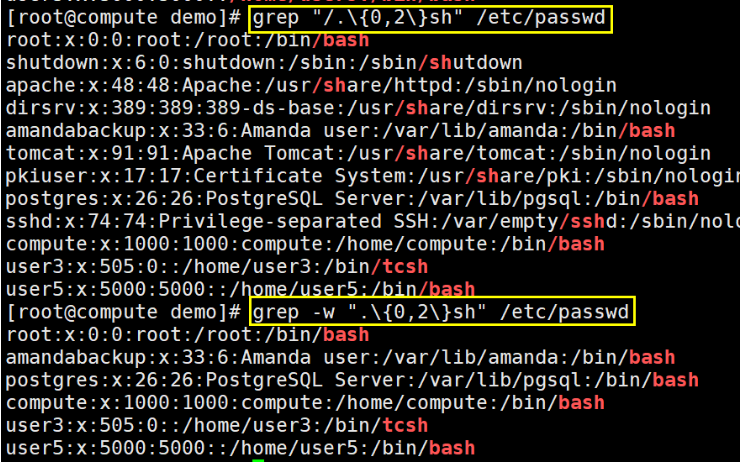
## 匹配次数：

**\{m,n\}** ：匹配其前面出现的字符至少m次，至多n次。

**\?** ：匹配其前面出现的内容0次或1次，等价于\{0,1\}。

　　　　　　\* ：匹配其前面出现的内容任意次，等价于\{0,\}，所以 ".\*" 表述任意字符任意次，即无论什么内容全部匹配。





## 位置锚定：行首或行尾、单词开头或结尾

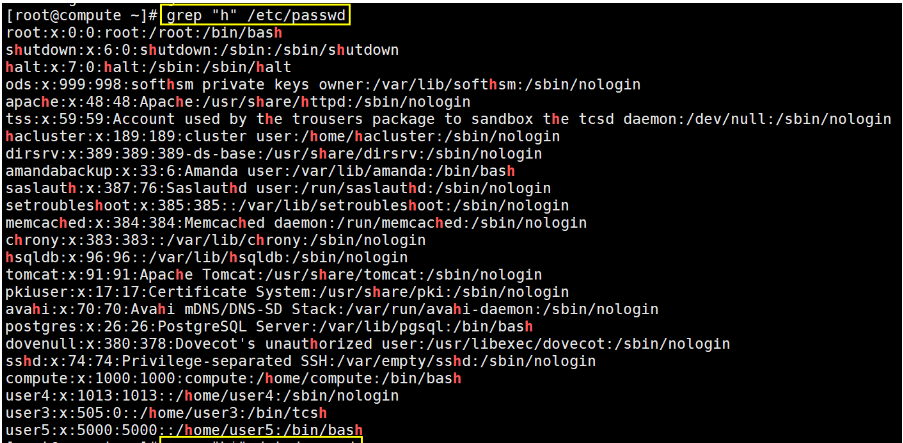
**^** ：锚定行首

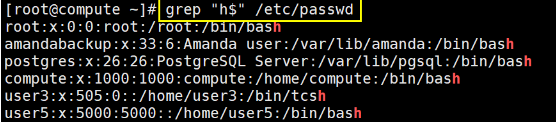
**$** ：锚定行尾。技巧：**"^$"用于匹配空白行。**

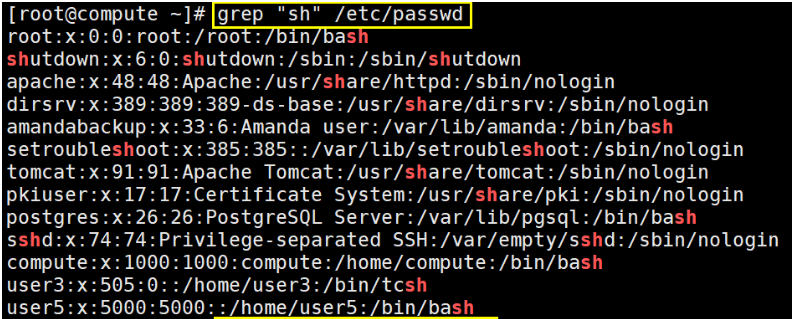
**\b或\<**：锚定单词的词首。如"\blike"不会匹配alike，但是会匹配liker

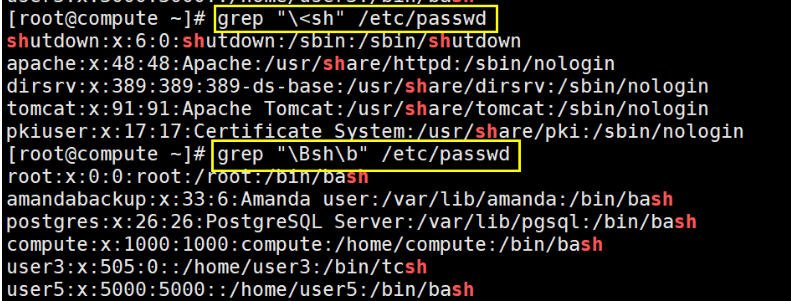
**\b或\>**：**锚定单词的词尾**。如"\blike\b"不会匹配alike和liker，只会匹配like

　　　　　　\B ：与\b作用相反。









## 分组及引用：

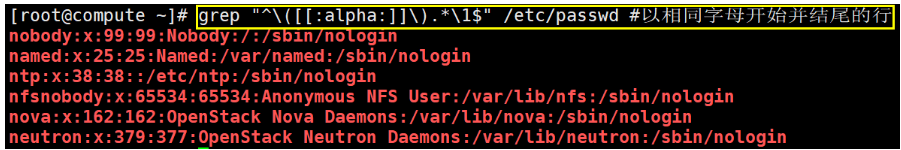
　　　　　　\(string\) ：将string作为一个整体方便后面引用

　　　　　　　　\1 ：引用第1个左括号及其对应的右括号所匹配的内容。

　　　　　　　　\2 ：引用第2个左括号及其对应的右括号所匹配的内容。

　　　　　　　　\n ：引用第n个左括号及其对应的右括号所匹配的内容。

示例：匹配出以相同字符开头且以该字符结尾的行



## 扩展的正则表达式

扩展的（Extend）正则表达式（注意要使用扩展的正则表达式要加-E选项，或者直接使用egrep）：

　　　　匹配字符：这部分和基本正则表达式一样

　　　　匹配次数：

　　　　　　\* ：和基本正则表达式一样

　　　　　　? ：基本正则表达式是\?，二这里没有\。

　　　　　　{m,n} ：相比基本正则表达式也是没有了\。

　　　　　　+ ：匹配其前面的字符至少一次，相当于{1,}。

　　　　位置锚定：和基本正则表达式一样。

　　　分组及引用：

　　　　　　(string) ：相比基本正则表达式也是没有了\。

　　　　　　　　\1 ：引用部分和基本正则表达式一样。

　　　　　　　　\n ：引用部分和基本正则表达式一样。

　　　　或者：

　　　　　　a|b ：匹配a或b，注意a是指 | 的左边的整体，b也同理。比如 C|cat 表示的是 C或cat，而不是Cat或cat，如果要表示Cat或cat，则应该写为 (C|c)at 。记住(string)除了用于引用还用于分组。

**注1**：默认情况下，正则表达式的匹配工作在**贪婪模式**下，也就是说它会尽可能长地去匹配，比如某一行有字符串 abacb，如果搜索内容为 "a.\*b" 那么会直接匹配 abacb这个串，而不会只匹配ab或acb。

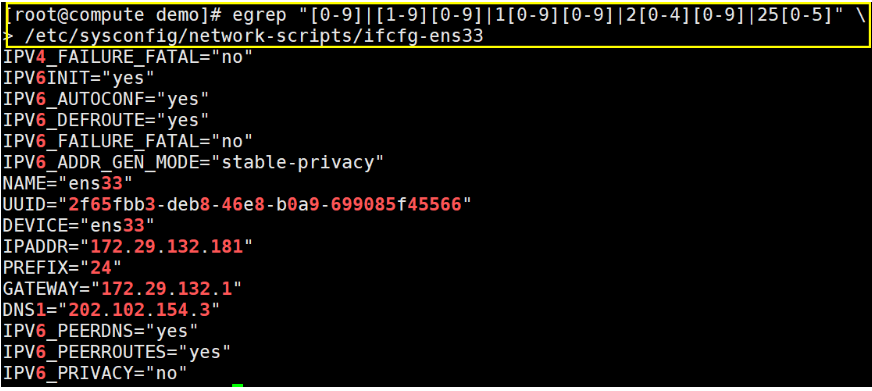
**注2**：所有的正则字符，如 [ 、\* 、( 等，若要搜索 \* ，而不是想把 \* 解释为重复先前字符任意次，可以使用 \\* 来转义。

## 练习:检索出所有的IP地址

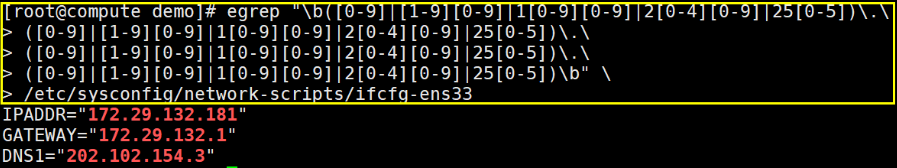
下面用一个练习来结束本次grep的学习：

在网络配置文件 /etc/sysconfig/network-scripts/ifcfg-ens33 中检索出所有的 IP

### 检索出 0-255的范围



### 由0-255的数字组合成IP



### 简化

