# Linux三剑客

## 三剑客之一 Grep

## 描述

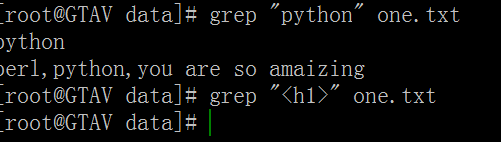
**这个命令全称为是Global Regular Expression Print，即全局正则表达式打印。它使用强大的正则表达式匹配相应的文本。是Linux中最强大的命令之一。它最重要的用途是打印符合匹配模式的文本行；**

**Grep和egrep,fgrep都属于同一命令下的不同分支，egrep相当于grep -e ,fgrep 相当于grep -f，不过后两个命令已经弃用，但依然有效，考虑到向上兼容性；**

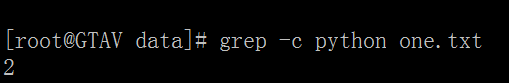
## 格式参数

**Grep +（参数）+正则表达式+ 要过滤的文件**

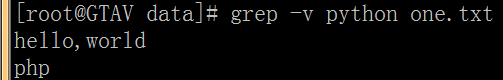
**返回的是过滤后的结果，如果筛选到，则返回筛选到的文本。否则，返回空值。**



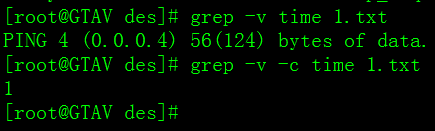
**-c print count of the match line.**



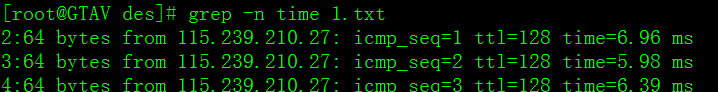
**-v print unmatchable lines**



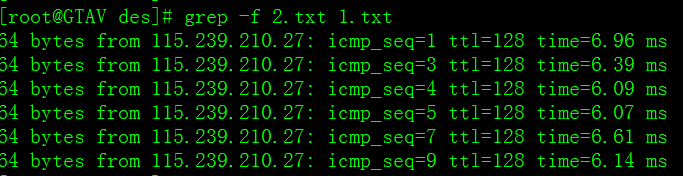
**If -v and -c together, grep will print count the line of unmatchable,**



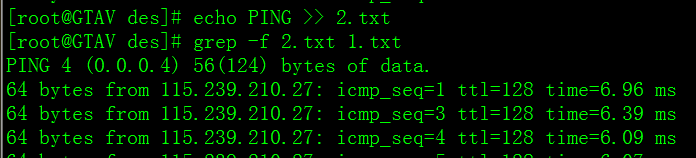
**-n print match lines with line number.**



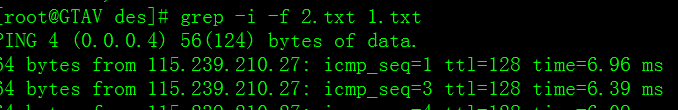
**-f file the match string in file, one per line, if file is empty, there matches nothing. 这个特性可以用来比较两个文件，查找两个文件的相似部分。**



**More than 1 line :**



**-i 忽略在模式匹配和文件内容之间的差异，比如大小写等；**

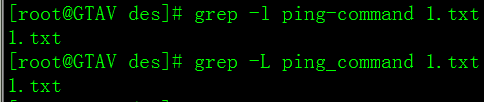


**-x match a whole line**



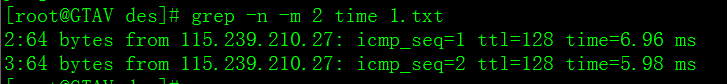
**More than 1 line :**

**-l print the name of input file which have been normally matched .**

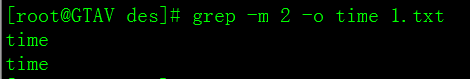


**-L print the name of input file which havn’t been normally matched.**

**-m num 在匹配了num行后停止匹配**



**-m print only mathed part of match line**



**-q 不管是否匹配到，程序立即退出**



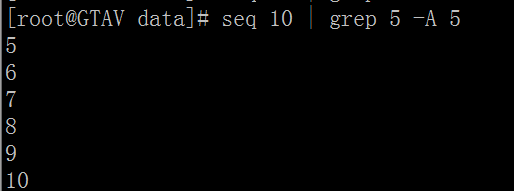
**-d action 如果被搜索文件是目录，则必须使用此参数，action表示该执行的行为，如果是recurse，表示递归读取目录中的所有文件，这相当于使用-r参数；如果是skip ，则跳过此目录；默认是read选项，读取目录就像是普通文件一样，但不会递归。**

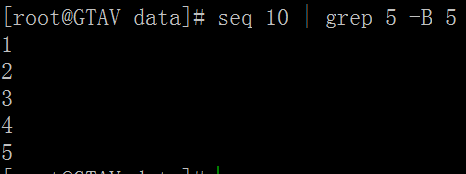
**三个参数：**

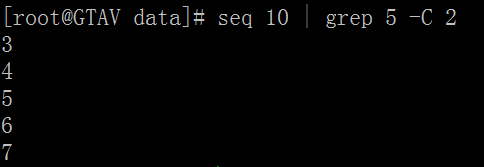
**-A n：显示匹配的这一行，并且显示这一行之后的n行**

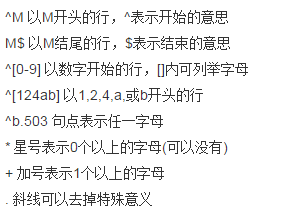
**-B n：显示匹配的这一行，并显示这一行之前的n行**

**-C n：显示匹配的这一行，并显示这一行前后的各n行**

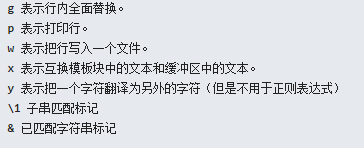








**替换参数**



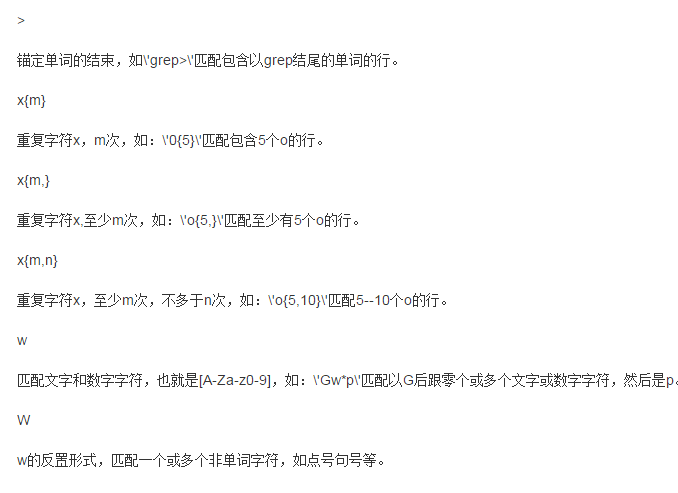
## 描述

**这个命令是Linux中最强大的命令之一，它用来过滤文本内容，有多种形式的过滤，提供了多个参数以满足这种多样化的要求，比如使用-o可以仅仅打印匹配到的字符串，使用-n可以打印匹配到的行的行号，使用-c可以只打印匹配到的行数等等，它把过滤到的结果可以传到下一个命令进行处理；**

**Grep可以被用于shell脚本，因为它返回的是布尔值，如果搜索成功则返回0，搜索不成功，则返回1；如果搜索的文件不存在，则返回2**

**关于正则表达式：http://richier.blog.51cto.com/1447532/1630963，这个博客上有详细的关于正则表达式的讨论**







## 核心内容

**这个命令的核心是过滤筛选文本，然后将过滤到的结果通过管道进行下一步处理，如果说命令执行过程是一个完整的操作过程，grep就是一个过滤器，完成筛选过程。**

**end**

# 三剑客之二 Sed

## 描述

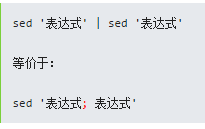
**Sed全称是stream editor 流式编辑器，它用来将一系列的文档字符串经过一种编辑命令以另一种格式输出，sed通常用来匹配一个或多个正则表达式的文本进行处理；使用分号；可以执行多个命令，这在sed中非常常用；尽管sed命令的限制很多，但一连串的sed可以编写类似快打砖块，俄罗斯方块这样的游戏**

## 参数

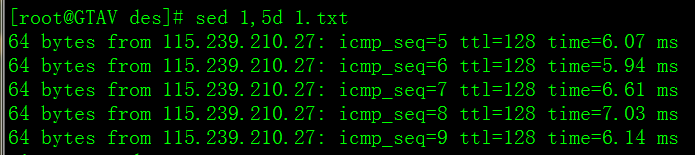
**格式 sed option command file**

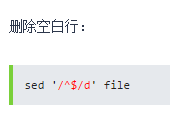
**Sed option -f scriptfile file (适用于脚本模式匹配)**

**-n 使用安静模式。默认的，每次sed命令执行，所有的来自STDIN的字符都会被打印一次，使用这个参数可以只打印选取的结果**

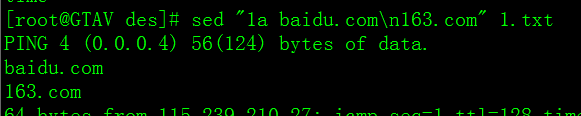


**（n，m）d : 删除行，用来忽略n到m之间的行，不包括m**

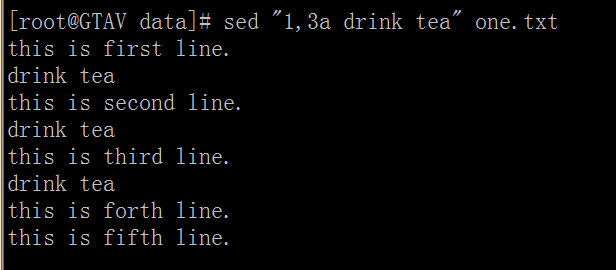




**a 用来添加文本到某一行中，格式：sed ‘(n)a addtext’ file**

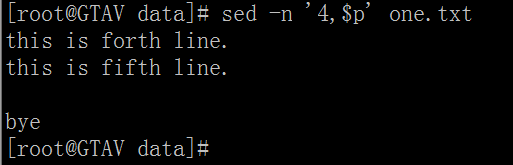


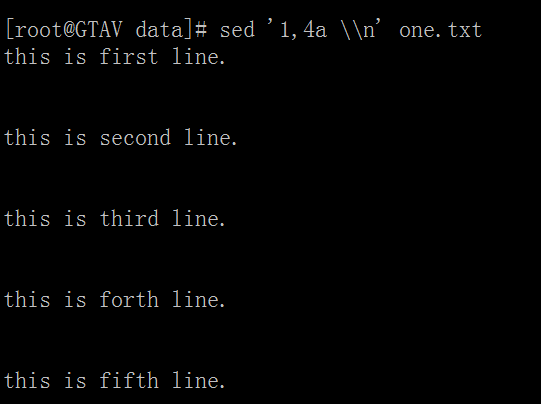
**还可以添加到多行中**



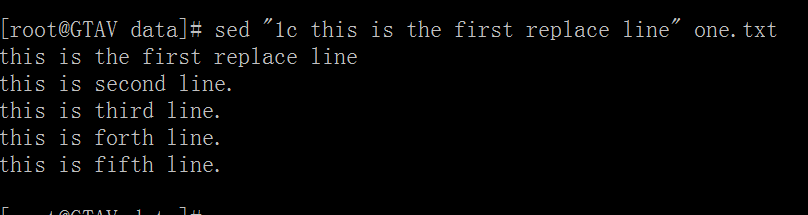
**可以让每一行增加空行**

**筛选指定的行到底部最后一行，$表示最后一行**



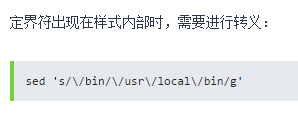


**使用c来替换成目标内容**

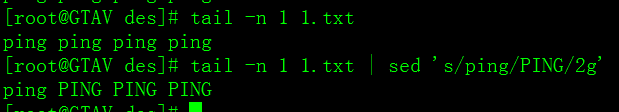


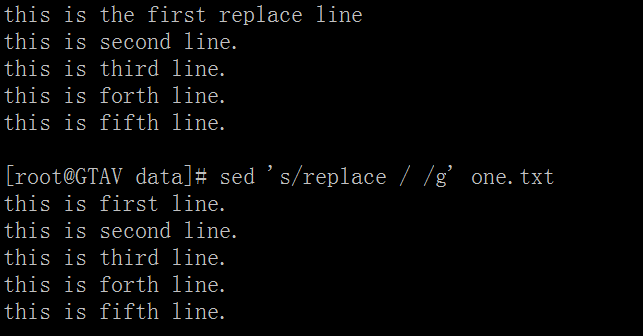
**替换字符串 : ‘s/要替换的字符串/替换成什么字符串/g’，使用-i来替换源文件，否则只在当前回话中输出替换后的文本，源文件不变。**

**可以使用///来作为定界符，也可以使用# ，@等，如果定界符出现在文本中，需要转移**



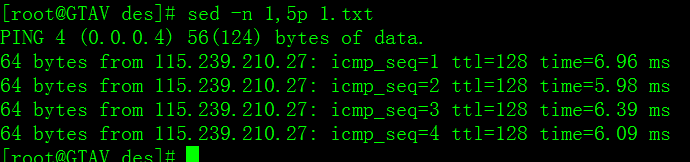
**如果从某一行中第n个开始匹配，使用ng**



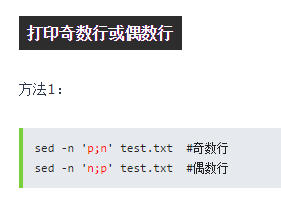


**把replace替换为空字符串**

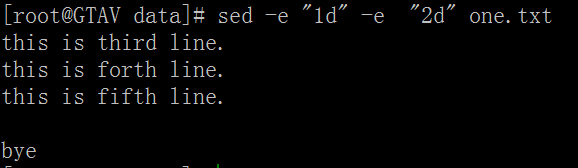
**P 参数用来输出指定的行，它必须和-n参数一起使用，关闭默认输出，仅仅输出筛选到的行**

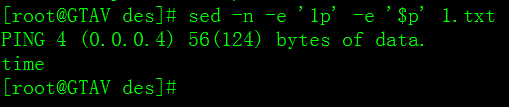


**打印奇数行和偶数行**

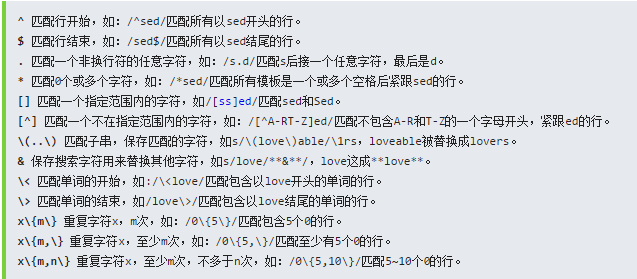


**-e用来在一行执行多个命令**



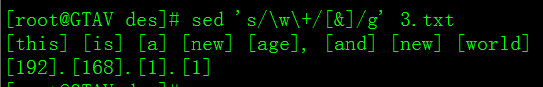


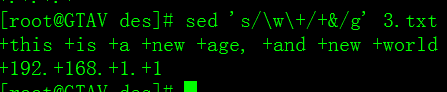
**可以使用正则表达式来筛选替换的内容，支持的正则表达式如下：**



**使用’\w\+’来替换每个单词的风格样式，**

**&代表匹配到的每个单词**

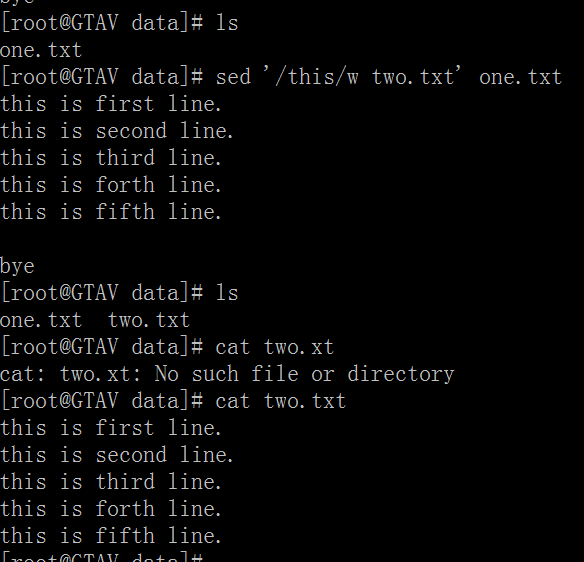




**子串匹配，\(..\)用来匹配子串，第一个匹配到的就标记为\1**



**W，写入命令，sed ‘/要写入的字符串/w file’ finename ，把filname中的和匹配的字符串能匹配到的行写入到file中。**



## 核心内容

**这个命令的核心是以多种方式处理文本，这些方式包括但不限于添加，删除，修改，精确查询，它返回的文本作为下一阶段的输入或者是直接打印到屏幕上；它的工作流程是先把文件内容读入到缓冲区，然后一点一点从缓冲区中处理。**

**<http://man.linuxde.net/sed> 这篇博客详细介绍了这个命令，还可以编写脚本以让sed来执行的方法。这里不再讨论，在后面会用到的时候再次研究；**

# AWK命令

## 三剑客之awk命令

## 描述

**Awk是Linux中最强大的文本处理工具之一。相比于grep文本的查找，sed对行的操作，AWK更多的是对数据进行详细的分析并生成报告，它可以把数据按行分析，然后按照分隔符进行切片，切开的部分进行分析处理等。它是Linux现有的最强大的数据处理引擎之一**

**Awk现在已经成为一种新的编程语言：awk语言--样式扫描与处理语言。它可以实现正则表达式的匹配，样式装入，流控制，进程控制等，甚至内置的条件语句也非常强大。Awk有三个版本，awk，nawk，gawk，通常所指的是gawk，即gun版本。**

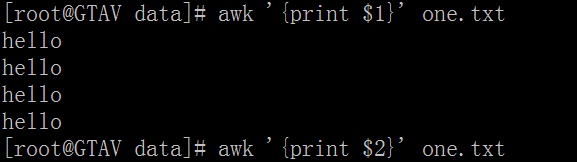
**基本用法：awk =+ ‘{pattern + action}’ filename：pattern是模式，可以是正则表达式，也可以是awk内置的函数。Action是要执行的动作**

**因为awk已经成为一门独立的语言，这里只做基本的了解，不做深入的研究**

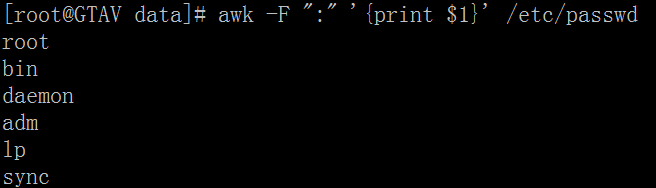
## 参数

**筛选第一列，然后输出**

**筛选文本的每行的第一列，分隔符为空格**

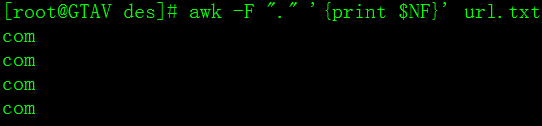


**使用-F来强制指定分隔符**

****

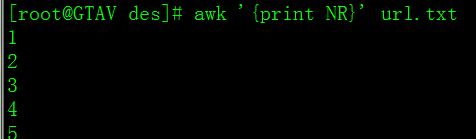
**筛选最后一列，使用$NF，不能使用双引号包含{}。$0表示整个文本**

**$NF-1表示倒数第二列**



**Awk是有一系列常量，**

**NR表示行号**

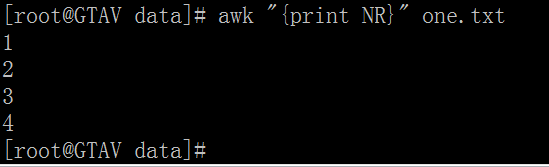


**$0 表示当前所有行**

**$1 表示第一列，也就是第二个参数；**

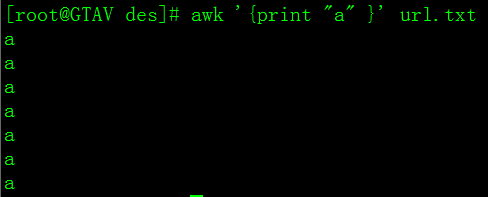
**$2 表示第二列，以此类推**

**$NF是表示读到的最后一个参数，也就是最后一列**



### {print var} 输出命令

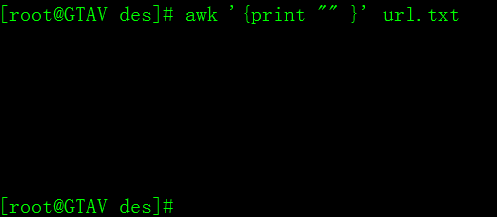
**{print “a”} filename : 输出同等行数的a**



**{print $1;print $2} file 将每一行中第一列和第二列的内容分行输出**



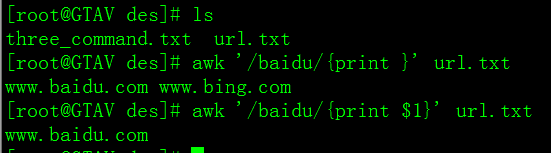
**{print “”} filename : 输出和filename同样多的空行，表明这个命令是逐行处理的。**



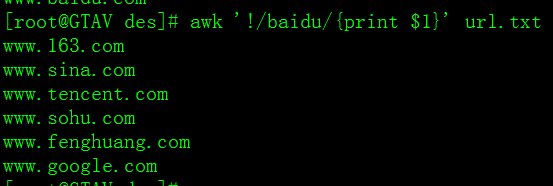
**OFS ‘symbol’ 以symbol来分割字符串，-F用来以已有的分隔符进行分割，返回的字符串会去除分隔符。**



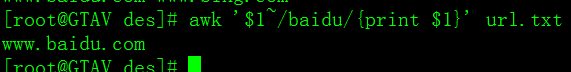
**//纯字符匹配**



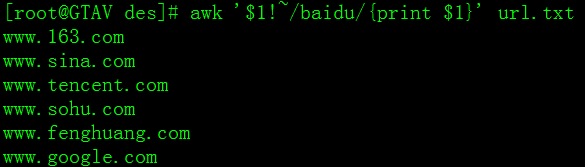
**!//纯字符不匹配，输出不匹配的字符**



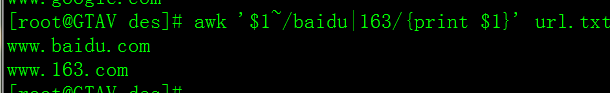
**~//字段值匹配，输出匹配的字符串**



**!~//字段值不匹配，输出不匹配的字符串**



**~/a1|a2/字段值匹配a1或a2**



**end**

## 核心内容

## 这个命令最重要的部分是它对文本的处理更加精细化，可以具体到每个单词的改变，文本的迭代，条件判断等，已经成为一门独立的语言

**<http://www.cnblogs.com/xudong-bupt/p/3721210.html> 这个博客详细介绍了awk的基本语法知识，包括变量的定义，基本条件语句，循环语句等。这里不做研究**

## 总结

**Linux三剑客，grep，sed，awk，这三个命令非常类似，因为它们都用来对文本进行处理，从最开始的文本按行过滤，到对文本的内容进行控制，然后以想要的方式进行输出，再到最后使用awk语言，直接对文本进行迭代控制，适用于自动化处理。这三个命令是Linux中对文本处理的主力军，几乎囊括了文本处理会遇到的一切问题。**