二，Linux三剑客

文章目录

- 为什么使用三剑客
- 三剑客是什么
  - 第一剑客：grep
  - 第二剑客：awk
  - 第三剑客：sed
- 总结

为什么使用三剑客

Linux给人的印象是黑黑乎乎的窗口，操作数据没有windows方便，其实不然，Linux也有自己的宝贝，称之为三剑客：grep，awk和sed。你可以用这三件法宝很方便的处理数据：查找，分段，修改，而这三个功能对应着我们今天的主角：grep，awk，sed。
总结来说：

形象一点比喻，如果把数据比作人群，那么grep就是照妖镜，用来找出妖精，awk就是尺子，给人群分门别类，而sed就是宝剑，用来除掉妖精。当你明白为什么要用三剑客时，现在你就要拿着这三把剑去斩妖除魔。

三剑客是什么

下面我会分别介绍这三剑客，你需要做的，是拿出电脑和我一同演练剑术。

第一剑客：grep

grep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。这把剑可以分辨妖魔，从人群中找出害人的妖精。这可是非常重要的供武器，请一定要尝试着掌控它。

```
curl https://testerhome.com \
| grep -o 'http://[a-zA-Z0-9\.\-]*'
```

我们的人群从哪里来？从curl中来，上面的`curl https://testerhome.com`表示展示testerhome的**html**信息，我把这些信息比做茫茫人群：

```
<div class="links"  data-no-turbolink="">
<span style="font-size:14px; color:#666;">友情链接</span><span style="margin-le
          <a style="color:#317DDA;" href="http://wetest.qq.com/?from=links_test
          <a style="color:#317DDA;" href="http://www.infoq.com/cn" target="_bla
          <a style="color:#317DDA;" href="http://www.testtao.com/portal.php" ta
          <a style="color:#317DDA;" href="https://www.testwo.com/" target="_bla
```

```
            <a style="color:#317DDA;" href="http://tieba.baidu.com/f?ie=utf-8&kw=
            <a style="color:#317DDA;" href="http://www.itdks.com/" target="_blan
  </span>
```

这是我截取的部分内容，你会看到许多信息，如果我想找出妖精（所有的链接），你可以用grep的-o参数，打印匹配到的部分，你可以理解为妖精现形，其后的`http://[a-zAZ0-9\.\-]*`是正则表达式，它匹配以http://开头，其后跟着字母(a-zA-Z)或数字(0-9)或.(\.)或-(\-)，这里不懂的不要紧，后面我会讲正则这个强大的工具。正则的意思很明显，妖精的样子是以http://开头的连接

> 因为上面的连接以"结尾，比如`href="http://wetest.qq.com/?from=links_testerhome"`，当遇到"时，会跳出匹配，

请跟着我敲出上面的命令，以下面是它匹配到的内容：

```
http://wetest.qq.com
http://www.infoq.com
http://www.testtao.com
http://tieba.baidu.com
http://www.itdks.com
```

就像本例子一样，妖精的长相千奇百怪，不光是连接，还有可能是一段文本，一句话，甚至是一个单词，要想掌握grep，你还要了解下面表格中的信息，表格是剑术手册，至关重要：

命令	解释
-A n	追加显示结果行后面n行
-B n	追加显示结果行前面n行
-C n	追加显示结果行前后n行，默认 -C 2，表示 -A 2 -B 2
-c, –count	显示搜索结果数量
-o	只打印匹配结果部分
-E	使用扩展版正则表达式匹配，相当于执行 egrep
-F	使用静态字符串匹配，相当于执行 fgrep
-z	先解包再搜索，相当于执行 zgrep

命令	解释
-e pattern	表示后续字符串为目标正则表达式，主要用于同时使用多个 -e匹配多个模式，也用于正则表达式开头为 – 的场合（消除歧义）
-v	反向搜索，打印不匹配搜索模式的结果
-q,–quiet,–silent	非贪婪式搜索，搜到一个结果时停止继续搜索该文件
–exclude,–exclude-dir pattern	排除非目标搜索文件或目录，高优先级
–include,–include-dir pattern	仅搜索目标文件或目录，低优先级
-f file	指定搜索某个文件
-H,-h	打印、不打印文件名
-n	显示行号，从 1开始
-I	忽略二进制文件
-i	忽略大小写，默认大小写敏感
-L	仅列出打印不含目标搜索结果的文件
-l	仅列出打印包含目标搜索结果的文件
-m n	仅显示前n个结果
-R	递归扫描子文件夹
-S	递归扫描时，追踪扫描符号链接文件、目录，默认不追踪（等同于指定 -p）
-s	静默模式，忽略错误文件
-w,-x	正则模式作为单个完整单词、完整行进行搜索

这些信息是掌握第一剑客grep的重要资料，当然你需要多加练习才能更好的使用它，表格供你不明时进行查询。
除此之外，如果你想查看更多有关grep的内容，可以用：

```
grep --help
```

第二剑客：awk

如果为三剑客排个地位，那么awk绝对是**老大**，他甚至可以代替grep，awk不光支持正则，还可以对字段进行分段处理，它的功能是辨认妖精和给妖精分类。

```
echo $PATH | awk 'BEGIN{RS=":"}{print $0}' \
 | awk -F/ '{print $1,$2}'
```

不同于上面的curl命令，echo $PATH打印本地的环境变量，这是一个新人群：

```
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

人群太乱了，它们都堆在一排，我根本没法操作，我现在用法宝awk把他们分行`awk 'BEGIN{RS=":"}{print $0}'`，其中RS的意思是用:作为记录分割符，将上面的内容**分行**，BEGIN的意思是**初始化代码块**，在对每一行进行处理之前，先执行BEGIN中的内容，这就是施法前摇：

```
/usr/local/sbin
/usr/local/bin
/usr/sbin
/usr/bin
/sbin
/bin
```

这样看起来舒服很多，大家都被分门别类，现在我发现了妖精，符号/太累赘，想阻挡我们的去路，于是用awk去掉这个拦路虎`awk -F/ '{print $1,$2}'`命令用-F/表示用/作为分隔符，$1和$2表示提取每行前2个内容，print进行打印：

```
 usr
 usr
 usr
 usr
 sbin
 bin
```

有的同学很迷惑，为啥每行只提取出一个字符（usr），而不是两个（usr locl），这是因为$1是指/前的字符，比如/usr/local/sbin，第一个/前是空白，第二个/前是usr，因此打印只显示了**空白usr**

其实参数FS与F的使用相同，上面的剑客也可以这么用：

```
echo $PATH | awk 'BEGIN{RS=":"}{print $0}' \
| awk 'BEGIN{FS="/"}{print $1,$2}'
```

就像本例子一样，人群会很复杂，你要先精简人群，才能找出妖精，要想掌握awk，你还要了解下面表格中的信息，表格是剑术手册，至关重要：
awk的语法是这样的：

```
awk [-F|-f|-v] ‘BEGIN{} //{command1; command2} END{}’ file
```

意思可以对照下面的表格：

选项	解释
[-F|-f|-v]	-F指定分隔符，-f调用脚本，-v定义变量 var=value
' '	引用代码块
BEGIN	初始化代码块，在对每一行进行处理之前，初始化代码，主要是引用全局变量，设置FS分隔符
//	匹配代码块，可以是字符串或正则表达式
{}	命令代码块，包含一条或多条命令
;	多条命令使用分号分隔
END	结尾代码块，在对每一行进行处理之后再执行的代码块，主要是进行最终计算或输出结尾摘要信息

其他命令	解释
$0	表示整个当前行
$1	每行第一个字段
NF	字段数量变量
NR	每行的记录号，多文件记录递增

其他命令	解释
FNR	与NR类似，不过多文件记录不递增，每个文件都从1开始
\t	制表符
\n	换行符
FS	BEGIN时定义分隔符
RS	输入的记录分隔符， 默认为换行符(即文本是按一行一行输入)
~	匹配，与==相比不是精确比较
!~	不匹配，不精确比较
==	等于，必须全部相等，精确比较
!=	不等于，精确比较
&&	逻辑与
+	匹配时表示1个或1个以上
/[0-9][0-9]+/	两个或两个以上数字
/[0-9][0-9]*/	一个或一个以上数字
FILENAME	文件名
OFS	输出字段分隔符， 默认也是空格，可以改为制表符等
ORS	输出的记录分隔符，默认为换行符,即处理结果也是一行一行输出到屏幕
-F'[:#/]'	定义三个分隔符

除此之外，如果你想查看更多有关awk的内容，可以用：

```
awk --help
```

拓展：RS、ORS、FS、OFS的区别和联系

第三剑客：sed

这是最后的剑客sed，如果说grep用来分辨妖精，awk用来规则人群，那么sed就是用来斩除妖精的武器。

严格来说：sed是一种流编辑器，它是文本处理中非常有用的工具，能够完美的配合正则表达式使用，功能不同凡响。

```
echo $PATH | awk 'BEGIN{RS=":"}{print $0}' \
| sed 's#/ #----#g'
```

本次我将同时使用两件法宝：awk和sed。
人群依旧是echo $PATH：

```
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

用第二件法宝awk进行分割后如下（不懂的可以往上看）：

```
/usr/local/sbin
/usr/local/bin
/usr/sbin
/usr/bin
/sbin
/bin
```

接下来我要做的，是把拦路虎/放平，将/替换成—-sed 's#/ #----#g'其中的s表示进行替换，注意到其后有三个#，#之间的内容分别是要替换的内容/，替换成什么 —-，开启法宝后如下：

```
----usr----local----sbin
----usr----local----bin
----usr----sbin
----usr----bin
----sbin
----bin
```

你会发现，\全部被替换成了—-
就像本例子一样，斩除的功能不言而喻，要想掌握sed，你还要了解下面表格中的信息，表格是剑术手册，至关重要.
sed的命令格式如下：

```
sed [options] 'command' file(s)
sed [options] -f scriptfile file(s)
```

选项

参数	完整参数	说明
-e	script –expression=script	以选项中的指定的script来处理输入的文本文件
-f	script –files=script	以选项中的指定的script文件来处理输入的文本文件
-h	–help	显示帮助
-n	–quiet –silent	仅显示script处理后的结果
-V	–version	显示版本信息

命令

命令	说明
d	删除，删除选择的行
D	删除模板块的第一行
s	替换指定字符
h	拷贝模板块的内容到内存中的缓冲区
H	追加模板块的内容到内存中的缓冲区
g	获得内存缓冲区的内容，并替代当前模板块中文本
G	获得内存缓冲区的内容，并追加到当前模板块文本的后面
l	列表不能打印字符的清单
n	读取下一个输入行，用下一个命令处理新的行而不是第一个命令
N	追加下一个输入行到模板块后面并在二者间嵌入一个新行，改变当前行号码
p	打印模板块的行

命令	说明
P	打印模板块的第一行
q	退出sed
b label	分支到脚本中带有标记的地方，如果分支不存在则分支到脚本的末尾
r file	从file中读行
t label	if分支，从最后一行开始，条件一旦满足或者T，t命令，将导致分支到带有标号的命令处，或者到脚本的末尾
T label	错误分支，从最后一行开始，一旦发生错误或者T，t命令，将导致分支到带有标号的命令处，或者到脚本的末尾
w file	写并追加模板块到file末尾
W file	写并追加模板块的第一行到file末尾
!	表示后面的命令对所有没有被选定的行发生作用
=	打印当前行号
#	把注释扩展到第一个换行符以前

替换

命令	说明
g	表示行内全面替换
p	表示打印行
w	表示把行写入一个文件
x	表示互换模板块中的文本和缓冲区中的文本
y	表示把一个字符翻译为另外的字符（但是不用于正则表达式）
\1	子串匹配标记
&	已匹配字符串标记

扩展：

sed处理数据时，把当前处理的行存储在临时缓冲区中，称为『模式空间』（pattern space），接着用sed命令处理**缓冲区**中的内容，处理完成后，把缓冲区的内容送往屏幕。接着处理下一行，这样不

断重复，直到文件末尾。**文件内容并没有改变**，除非你使用重定向存储输出。sed主要用来自动编辑一个或多个文件，简化对文件的反复操作，编写转换程序等。

更多用例请参考：http://www.runoob.com/linux/linux-comm-sed.html
除此之外，如果你想查看更多有关awk的内容，可以用：

```
awk --help
```

总结

Linux三剑客是常识，因为主流服务器都是Linux，服务器测试必然成为刚需。在本次学习中，我们以几个小例子入门，对三剑客和正则有了初步了解，在接下来的学习中，我希望你要勤查，勤练，勤动手，命令是敲出来的，不是看出来的。

vscode小技巧：https://code.visualstudio.com/Docs/editor/codebasics

更多

霍格沃兹测试学院官网首页:https://testing-studio.com