链滴

AWK 简明教程

作者：An

原文链接：https://ld246.com/article/1366467646146

来源网站：链滴

有一些网友看了前两天的《必须要道的Linux技巧》希望我能教教他们用awk和sed，所以，出现了这篇文章。我估计这些80后的轻朋友可能对awk/sed这类上古神器有点陌生了，所以需要我这个老家伙来炒炒冷饭。况且，AWK是尔实验室1977年搞出来的文本出现神器，今年是蛇年，是AWK的本命年，而且年纪和我相仿，所以常有必要为他写篇文章。
之所以叫AWK是因为其取了三位创始人 Alfred Aho，Peter Weinberger, 和 Brian Kernighan Family Name的首字符。要学AWK，就得提一提AWK的一本相当经典的书《The AWK Programmin Language》，它在豆瓣上的评分是9.4分！在亚马逊上居然卖1022.30元。
我在这里的教程并不想面面俱到，本文和我之前的Go语言简介一样，全是示例，基本无废话。</
>
我只想达到两个目的：
1）你可以在乘坐公交地铁上下班，或是在坐马桶拉大便时读完（保证是一泡大便的工夫）。</p

2）我只想让这篇博文像一个火辣的脱衣舞女挑起你的兴趣，然后还要你自己去下工夫去撸。</p

废话少说，我们开始脱吧（注：这里只是topless）。
起步上台
我从netstat命令中提取了如下信息作为用例：
<pre>$ cat netstat.txt
Proto Recv-Q Send-Q Local-Address Foreign-Address State
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
tcp 0 0 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
tcp 0 0 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
tcp 0 0 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
tcp 0 0 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
tcp 0 4166 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
tcp 0 1 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
tcp 0 0 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT
tcp 0 1 coolshell.cn:80 208.115.113.92:50601 LAST_ACK
tcp 0 0 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED
tcp 0 0 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2
tcp 0 0 :::22 :::* LISTEN</pre>
下面是最简单最常用的awk示例，其输出第1列和第4例，
其中单引号中的被大括号括着的就是awk的语句，注意，其只能被单引号包含。
其中的$1..$表示第几例。注：$0表示整个行。
<pre>$ awk '{print $1, $4}' netstat.txt
Proto Local-Address
tcp 0.0.0.0:3306
tcp 0.0.0.0:80
tcp 127.0.0.1:9000
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80</pre>

tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp coolshell.cn:80
tcp :::22</pre>
我们再来看看awk的格式化输出，和C语言的printf没什么两样：
<pre>$ awk '{printf "%-8s %-8s %-8s %-18s %-22s %-15s\n",$1,$2,$3,$4,$5,$6}' netstat.txt
Proto Recv-Q Send-Q Local-Address Foreign-Address State
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
tcp 0 0 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
tcp 0 0 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
tcp 0 0 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
tcp 0 0 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
tcp 0 4166 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
tcp 0 1 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
tcp 0 0 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT
tcp 0 1 coolshell.cn:80 208.115.113.92:50601 LAST_ACK
tcp 0 0 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED
tcp 0 0 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2
tcp 0 0 :::22 :::* LISTEN</pre>
脱掉外套
过滤记录
我们再来看看如何过滤记录（下面过滤条件为：第三列的值为0 & 第6列的值为LISTN）
<pre>$ awk '$3==0 & $6=="LISTEN" ' netstat.txt
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 :::22 :::* LISTEN</pre>
其中的“==”为比较运算符。其他比较运算符：!=, >, <, >=, <=
我们来看看各种过滤记录的方式：
<pre>$ awk ' $3>0 {print $0}' netstat.txt
Proto Recv-Q Send-Q Local-Address Foreign-Address State
tcp 0 4166 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
tcp 0 1 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
tcp 0 1 coolshell.cn:80 208.115.113.92:50601 LAST_ACK</pre>
如果我们需要表头的话，我们可以引入内建变量NR：
<pre>$ awk '$3==0 & $6=="LISTEN" || NR==1 ' netstat.txt
Proto Recv-Q Send-Q Local-Address Foreign-Address State
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 :::22 :::* LISTEN</pre>
再加上格式化输出：
<pre>$ awk '$3==0 & $6=="LISTEN" || NR==1 {printf "%-20s %-20s %s\n",$4,$5,

6}' netstat.txt
Local-Address Foreign-Address State
0.0.0.0:3306 0.0.0.0:* LISTEN
0.0.0.0:80 0.0.0.0:* LISTEN
127.0.0.1:9000 0.0.0.0:* LISTEN
:::22 :::* LISTEN</pre>
内建变量
说到了内建变量，我们可以来看看awk的一些内建变量：

$0 当前记录（这个变量中存放着整个行的内容）
$1~$n 当前记录的第n个字段，字段间由FS分隔
FS 输入字段分隔符 默认是空格或Tab
NF 当前记录中的字段个数，就是有多少列
NR 已经读出的记录数，就是行号，从1开始，如果有多个文件话，这个值也是不断累加中。</li

FNR 当前记录数，与NR不同的是，这个值会是各个文件自己的行号
RS 输入的记录分隔符， 默认为换行符
OFS 输出字段分隔符， 默认也是空格
ORS 输出的记录分隔符，默认为换行符
FILENAME 当前输入文件的名字

怎么使用呢，比如：我们如果要输出行号：
<pre>$ awk '$3==0 && $6=="ESTABLISHED" || NR==1 {printf "%02s %s %-20s %-
0s %s\n",NR, FNR, $4,$5,$6}' netstat.txt
01 1 Local-Address Foreign-Address State
07 7 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
08 8 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
10 10 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
14 14 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
17 17 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED</pre>
指定分隔符
<pre>$ awk 'BEGIN{FS=":"} {print $1,$3,$6}' /etc/passwd
root 0 /root
bin 1 /bin
daemon 2 /sbin
adm 3 /var/adm
lp 4 /var/spool/lpd
sync 5 /sbin
shutdown 6 /sbin
halt 7 /sbin</pre>
上面的命令也等价于：（-F的意思就是指定分隔符）
<pre>$ awk -F: '{print $1,$3,$6}' /etc/passwd</pre>
注：如果你要指定多个分隔符，你可以这样来：
<pre>awk -F '[:]'</pre>
再来看一个以\t作为分隔符输出的例子（下面使用了/etc/passwd文件，这个文件是以:分隔的）

<pre>$ awk -F: '{print $1,$3,$6}' OFS="\t" /etc/passwd
root 0 /root
bin 1 /bin
daemon 2 /sbin
adm 3 /var/adm
lp 4 /var/spool/lpd
sync 5 /sbin</pre>
脱掉衬衫

字符串匹配
我们再来看几个字符串匹配的示例：
<pre>$ awk '$6 ~ /FIN/ || NR==1 {print NR,$4,$5,$6}' OFS="\t" netstat.txt
1 Local-Address Foreign-Address State
6 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
9 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
13 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
18 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2
$ $ awk '$6 ~ /WAIT/ || NR==1 {print NR,$4,$5,$6}' OFS="\t" netstat.txt
1 Local-Address Foreign-Address State
5 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
6 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
9 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
11 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
13 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
15 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT
18 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2 </pre>
上面的第一个示例匹配FIN状态， 第二个示例匹配WAIT字样的状态。其实 ~ 表示模式开始。/ /是模式。这就是一个正则表达式的匹配。
其实awk可以像grep一样的去匹配第一行，就像这样：
<pre>$ awk '/LISTEN/' netstat.txt
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 :::22 :::* LISTEN</pre>
我们可以使用 “/FIN|TIME/” 来匹配 FIN 或者 TIME :
<pre>$ awk '$6 ~ /FIN|TIME/ || NR==1 {print NR,$4,$5,$6}' OFS="\t" netstat.txt
1 Local-Address Foreign-Address State
5 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
6 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
9 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
11 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
13 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
15 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT
18 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2</pre>
再来看看模式取反的例子：
<pre>$ awk '$6 !~ /WAIT/ || NR==1 {print NR,$4,$5,$6}' OFS="\t" netstat.txt
1 Local-Address Foreign-Address State
2 0.0.0.0:3306 0.0.0.0:* LISTEN
3 0.0.0.0:80 0.0.0.0:* LISTEN
4 127.0.0.1:9000 0.0.0.0:* LISTEN
7 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
8 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
10 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
12 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
14 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
16 coolshell.cn:80 208.115.113.92:50601 LAST_ACK
17 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED
19 :::22 :::* LISTEN</pre>
或是：
<pre>awk '!/WAIT/' netstat.txt</pre>
折分文件
awk拆分文件很简单，使用重定向就好了。下面这个例子，是按第6例分隔文件，相当的简单（中的NR!=1表示不处理表头）。

<pre>$ awk 'NR!=1{print > $6}' netstat.txt
$ ls
ESTABLISHED FIN_WAIT1 FIN_WAIT2 LAST_ACK LISTEN netstat.txt TIME_WAIT
$ cat ESTABLISHED
tcp 0 0 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
tcp 0 4166 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
tcp 0 0 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED
$ cat FIN_WAIT1
tcp 0 1 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1
$ cat FIN_WAIT2
tcp 0 0 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
tcp 0 0 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
tcp 0 0 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2
$ cat LAST_ACK
tcp 0 1 coolshell.cn:80 208.115.113.92:50601 LAST_ACK
$ cat LISTEN
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 :::22 :::* LISTEN
$ cat TIME_WAIT
tcp 0 0 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
tcp 0 0 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
tcp 0 0 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT </pre>
你也可以把指定的列输出到文件：
<pre>awk 'NR!=1{print $4,$5 > $6}' netstat.txt</pre>
再复杂一点：（注意其中的if-else-if语句，可见awk其实是个脚本解释器）
<pre>$ awk 'NR!=1{if($6 ~ /TIME|ESTABLISHED/) print > "1.txt";
else if($6 ~ /LISTEN/) print > "2.txt";
else print > "3.txt" }' netstat.txt
$ ls ?.txt
1.txt 2.txt 3.txt
$ cat 1.txt
tcp 0 0 coolshell.cn:80 124.205.5.146:18245 TIME_WAIT
tcp 0 0 coolshell.cn:80 110.194.134.189:1032 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49809 ESTABLISHED
tcp 0 0 coolshell.cn:80 123.169.124.111:49829 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.215.36:36970 TIME_WAIT
tcp 0 4166 coolshell.cn:80 61.148.242.38:30901 ESTABLISHED
tcp 0 0 coolshell.cn:80 110.194.134.189:4796 ESTABLISHED
tcp 0 0 coolshell.cn:80 183.60.212.163:51082 TIME_WAIT
tcp 0 0 coolshell.cn:80 123.169.124.111:49840 ESTABLISHED
$ cat 2.txt
tcp 0 0 0.0.0.0:3306 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:9000 0.0.0.0:* LISTEN
tcp 0 0 :::22 :::* LISTEN
$ cat 3.txt
tcp 0 0 coolshell.cn:80 61.140.101.185:37538 FIN_WAIT2
tcp 0 0 coolshell.cn:80 116.234.127.77:11502 FIN_WAIT2
tcp 0 1 coolshell.cn:80 124.152.181.209:26825 FIN_WAIT1

tcp 0 1 coolshell.cn:80 208.115.113.92:50601 LAST_ACK
tcp 0 0 coolshell.cn:80 117.136.20.85:50025 FIN_WAIT2 </pre>
统计
下面的命令计算所有的C文件，CPP文件和H文件的文件大小总和。
<pre>$ ls -l *.cpp *.c *.h | awk '{sum+=$5} END {print sum}'
2511401</pre>
我们再来看一个统计各个connection状态的用法：（我们可以看到一些编程的影子了，大家都程序员我就不解释了。注意其中的数组的用法）
<pre>$ awk 'NR!=1{a[$6]++;} END {for (i in a) print i ", " a[i];}' netstat.txt
TIME_WAIT, 3
FIN_WAIT1, 1
ESTABLISHED, 6
FIN_WAIT2, 3
LAST_ACK, 1
LISTEN, 4</pre>
再来看看统计每个用户的进程的占了多少内存（注：sum的RSS那一列）
<pre>$ ps aux | awk 'NR!=1{a[$1]+=$6;} END { for(i in a) print i ", " a[i]"KB";}'
dbus, 540KB
mysql, 99928KB
www, 3264924KB
root, 63644KB
hchen, 6020KB</pre>
脱掉内衣
awk脚本
在上面我们可以看到一个END关键字。END的意思是“处理完所有的行的标识”，即然说到了END就有必要介绍一下BEGIN，这两个关键字意味着执行前和执行后的意思，语法如下：
BEGIN{ 这里面放的是执行前的语句 }
END {这里面放的是处理完所有的行后要执行的语句 }
{这里面放的是处理每一行时要执行的语句}
为了说清楚这个事，我们来看看下面的示例：
假设有这么一个文件（学生成绩表）：
<pre>$ cat score.txt
Marry 2143 78 84 77
Jack 2321 66 78 45
Tom 2122 48 77 71
Mike 2537 87 97 95
Bob 2415 40 57 62</pre>
我们的awk脚本如下（我没有写有命令行上是因为命令行上不易读，另外也在介绍另一种用法）
<pre>$ cat cal.awk
#!/bin/awk -f
#运行前
BEGIN {
math = 0
english = 0
computer = 0
printf "NAME NO. MATH ENGLISH COMPUTER TOTAL\n"
printf "---\n"
}
#运行中
{
math+=$3
english+=$4
computer+=$5
printf "%-6s %-6s %4d %8d %8d %8d\n", $1, $2, $3,$4,$5, $3+$4+$5

}
#运行后
END {
printf "---\n"
printf " TOTAL:%10d %8d %8d \n", math, english, computer
printf "AVERAGE:%10.2f %8.2f %8.2f\n", math/NR, english/NR, computer/NR
} </pre>
我们来看一下执行结果：（也可以这样运行 ./cal.awk score.txt）
<pre>$ awk -f cal.awk score.txt
NAME NO. MATH ENGLISH COMPUTER TOTAL

Marry 2143 78 84 77 239
Jack 2321 66 78 45 189
Tom 2122 48 77 71 196
Mike 2537 87 97 95 279
Bob 2415 40 57 62 159

TOTAL: 319 393 350
AVERAGE: 63.80 78.60 70.00</pre>
环境变量
即然说到了脚本，我们来看看怎么和环境变量交互：（使用-v参数和ENVIRON，使用ENVIRON环境变量需要export）
<pre>$ x=5
$ y=10
$ export y
 echo x $y
5 10
 awk -v val=x '{print $1, $2, $3, $4+val, $5+ENVION["y"]}' OFS="\t" score.txt
Marry 2143 78 89 87
Jack 2321 66 83 55
Tom 2122 48 82 81
Mike 2537 87 102 105
Bob 2415 40 62 72 </pre>
几个花活
最后，我们再来看几个小例子：
<pre>#从file文件中找出长度大于80的行
awk 'length>80' file
#按连接数查看客户端IP
netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr
#打印 99 乘法表
seq 9 | sed 'H;g' | awk -v RS='' '{for(i=1;i<=NF;i++)printf("%dx%d=%d%s", i, NR, i*NR, i==R?"\n":"\t")}' </pre>
自己撸吧
关于其中的一些知识点可以参看gawk的手册：
内建变量，参看：http://www.gnu.org/software/gawk/manual/gawk.html#Built_002din-Variables
流控方面，参看：http://www.gnu.org/software/gawk/manual/gawk.html#Statement
内建函数，参看：http://www.gnu.org/software/gawk/manual/gawk.html#Built_002din
正则表达式，参看：http://www.gnu.org/software/gawk/manual/gawk.html#Regexp
（文完）
转自：http://coolshell.cn/articles/9070.html