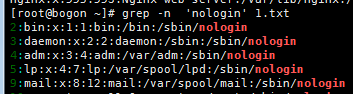
**一、正则（grep）**

1、grep -n 'nologin' 1.txt //过滤出文件中含'nologin'关键字的行，并且显示行号，centos7自带关键字颜色



2、grep -c 'nologin' 1.txt //匹配到的关键字有多少行



3、grep -ni 'nologin' 1.txt // -i 不区分大小写



4、grep -niv 'nologin' 1.txt // -v 取反，不包含关键字的行



5、grep -r 'root' /etc/ //遍历所有子目录下

6、grep -nA2 'root' 1.txt //列出含关键字的行和它下面的2行

grep -nB2 'root' 1.txt //列出含关键字的行和它上面的2行

grep -nC2 'root' 1.txt //列出含关键字的行和它上下的2行

7、grep '[0-9]' 1.txt //匹配包含数字的行

8、grep -nv '[0-9]' 1.txt //不包含数字的行



9、grep -n '^#' /etc/inittab //过滤出以‘#’开头的行

10、grep -nv '^#' /etc/inittab // 不以‘#’开头的行

11、grep '[^0-9]' 1.txt //匹配非数字字符，不是匹配的行



12、grep -v '[0-9]' 1.txt //匹配不含数字的行

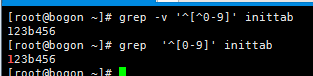


13、grep '^[^0-9]' inittab // 匹配不以非数字字符开头的行，也就是过滤掉以数字开头的行



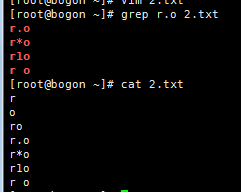
14、grep -v '^[^0-9]' inittab

grep '^[0-9]' inittab //匹配以数字开头的行的内容



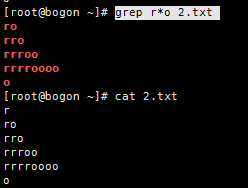
15、grep r.o 2.txt //r和o之间的任意一个字符

cat 2.txt

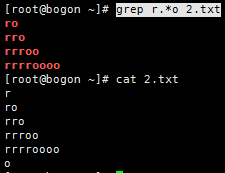


16、grep r\*o 2.txt //\*号左边的任意多个字符包含零个

当是零个r时，匹配单独的o



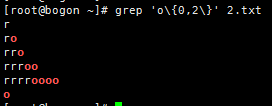
17、grep r.\*o 2.txt //r和o之间的任意字符



18、grep 'o\{2\}' 2.txt //匹配2倍的o，2.txt的内容同上



19、匹配区间



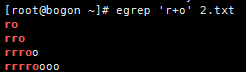
20、egrep 'o{2}' 2.txt



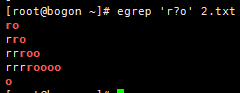


21、grep -E '(oo{2})' 2.txt //匹配2倍‘oo’

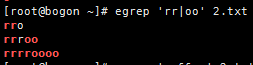
22、egrep 'r+o' 2.txt //1个或多个‘+’前面的字符，单独的r和单独的o都不能匹配



23、egrep 'r?o' 2.txt //1个或0个‘？’前面的字符



24、egrep 'rr|oo' 2.txt //‘|’或者

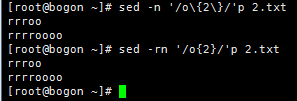


**二、sed**

1、sed -n '/root/'p 1.txt //匹配包含‘root’关键字的行



2、sed适用于'.';'.\*';'?';'+';'.\*'; sed -r 不用脱意



3、sed -n '1'p 1.txt //打印第一行的内容

4、sed -n '1，5'p 1.txt //打印第1行到第5行的内容

5、sed -n '5，$'p 1.txt //打印第5到最后

6、sed -n '/o$/'p 2.txt //打印末尾匹配o的行

7、sed -n '/^o/'p 2.txt //打印匹配以o开头的行

8、sed -e '1'p -e '/t/'p -n 2.txt //2个动作，打印第一行的同时，匹配't'的行

9、sed中，使用\u表示大写，\l表示小写  
 把每个单词的第一个小写字母变大写：  
sed 's/\b[a-z]/\u&/g' filename  
 把所有小写变大写：  
sed 's/[a-z]/\u&/g' filename  
大写变小写：  
sed 's/[A-Z]/\l&/g' filename

10、sed -n '/jz/'Ip 5.txt //I 不区分大小写



11、sed '1,30'd 1.txt //过滤掉除1到30行以外的行

12、sed -i '1,30'd 1.txt //删除1到30行



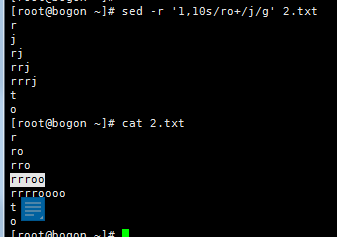
13、sed -i '/^$/'d 2.txt //删除空行

14、sed -i '/jz/'Id 5.txt //删除带关键字的行，不区分大小写

15、sed '1,10s/root/jz/g' 1.txt //将1到10行的root变成jz，源文件不变

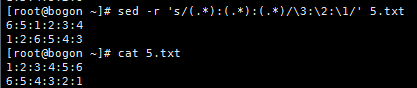
16、sed -i '1,10s/nologin/jz/g' 1.txt //真实替换，源文件改变

17、sed -r '1,10s/ro+/j/g' 2.txt

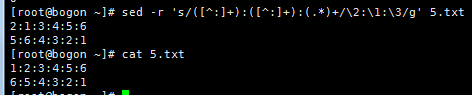


18、sed -r 's/([^:]+):(.\*):([^:])+/\3:\2:\1/g' 5.txt //第一段和最后一段调换

sed -r 's/(.\*):(.\*):(.\*)/\3:\2:\1/' 5.txt //同上



19、sed -r 's/([^:]+):([^:]+):(.\*)+/\2:\1:\3/g' 5.txt //将第1段和第2段替换



20、sed 's@/sbin/nologin@555@g' 1.txt ，将/sbin/nologin 替换成555

21、sed 's/[a-zA-Z]//g' 1.txt //删除字母

22、sed -r 's/(.\*)/aaa:&/' 1.txt //在最前面加上aaa:,&表示小括号里面

sed 's/^/hello:&/g' 1.txt //同上

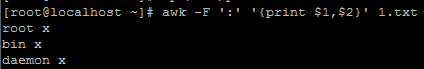
sed 's/$/&:hello/g' 1.txt //在每行的最后添加

**三、awk**

1、awk -F ':' '{print $1}' 1.txt //以‘：’为分隔符打印文件第一段

2、awk -F ':' '{print $0}' 1.txt //打印所有内容

3、awk -F ':' '{print $1,$2}' 1.txt //打印第1段和第2段



4、awk -F ':' '{print $1"&"$3"&"$4}' 1.txt //打印出来的内容以'&'分隔，默认是空格



5、awk '/oo/' 1.txt //打印匹配'oo'的行

6、awk -F ':' '$1 ~ /oo/' 1.txt //第一段匹配‘oo’的行

7、awk -F ':' '$1 ~ /o+/' 1.txt //第一段匹配‘o’的行

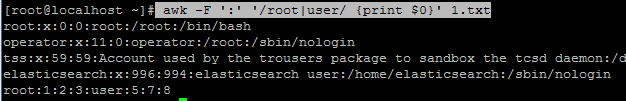
8、awk -F ':' '/root/ {print $1,$3} /user/ {print $7,$8}' 11.txt //打印匹配root的第1段和第3段，打印匹配user的第7段和第8段

cat 11.txt

root:1:2:3:user:5:7:8



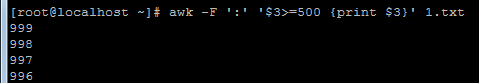
9、awk -F ':' '/root|user/ {print $0}' 1.txt //打印包含root或者user的行



10、awk -F ':' '$3==0 {print $1}' 1.txt //第三段等于0，并且显示第一段内容

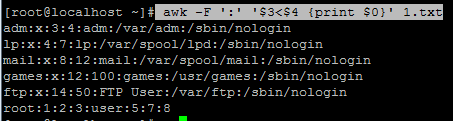


12、awk -F ':' '$3>=500 {print $3}' 1.txt //第3段打印500的行，并打印第3段内容



13、awk -F ':' '$3!=500 {print $3}' 1.txt //打印第3段不等于500的行

14、awk -F ':' '$7!=="/sbin/nologin" {print $0}' 1.txt //注意这里第七段不匹配"sbin/nologin"是有双引号的，如果是数字不要加

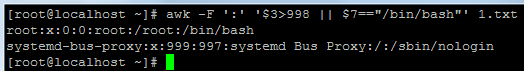
15、awk -F ':' '$3<$4 {print $0}' 1.txt //打印出第3段小于第4段的行

16、awk -F ':' '$3!=$4 {print $0}' 1.txt

awk -F ':' '$3==$4 {print $0}' 1.txt

17、awk -F ':' '$3>5 && $3<10' 1.txt //第3段大于5且小于10的行

18、awk -F ':' '$3>998 || $7=="/bin/bash"' 1.txt //第三段大于900或者第7段匹配/bin/bash





19、awk -F ':' '{OFS="#"} $3>998 || $7 ~ /bash/ {print $1,$3,$7}' 1.txt //打印匹配条件的，匹配出来的分隔符为‘#’



20、awk -F ':' '{OFS="#"} {if ($3>998) {print $1,$2,$3,$7}}' 1.txt



21、awk -F ':' '$3>998 || $7 ~ /\/bin\/bash/ {print NR}' 1.txt //打印出匹配到的是第几行



22、awk -F ':' '$3>998 || $7 ~ /\/bin\/bash/ {print NF}' 1.txt //匹配的内容有几段



23、awk -F ':' 'NR<=3' 1.txt打印前三行

24、awk -F ':' 'NR<=3 && $1 ~ /root|bin/' 1.txt //匹配前三行并且第一段需要匹配到root或者bin



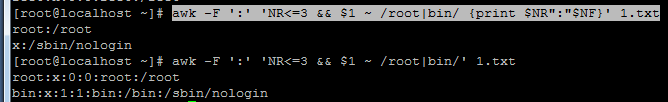
25、awk -F ':' 'NF==6 && $1 ~ /root|bin/' 1.txt //匹配只有6段的行，并且第一段匹配root或者bin



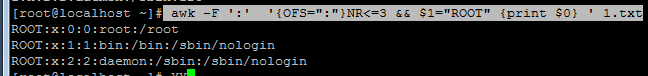
26、awk -F ':' 'NR<=3 && $1 ~ /root|bin/ {print $NR":"$NF}' 1.txt

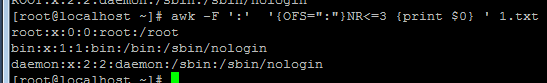
$NR：打印第n行的第n段

$NF: 打印每行的最后一段



27、awk -F ':' '{OFS=":"}NR<=3 && $1="ROOT" {print $0} ' 1.txt //前三行的第一段改成ROOT





28、awk -F ':' '{(tot=tot+$1)}; END {print tot}' 2.txt //打印第三段的和，必须是tot

http://www.cnblogs.com/emanlee/p/3327576.html