-n -r -i

p d s

a行后追加 i行前添加 c替换行

sed 'a 888' txt5 //每行下添加888

sed '2a 888' txt5 //第二行下添加888

sed '1,2a 888' txt5 //第1到2行下添加888

sed 'i 888' txt5 //每行上添加888

sed '3i 888' txt5 //第3行上添加888

sed '/A/a 888' txt5 //有A的行下添加888

sed 'c 888' txt5 //所有行替换成888

sed '1c 888' txt5 //第1行替换成888

如何使用sed和字符串处理判断yum仓库好坏?

vim

grep

sed 非交互式编辑文本

awk 搜索

awk '{print}' txt //输出文档所有内容

awk '{print $1}' txt //输出文档每行第1列,默认用空格做分隔符

awk '{print $1 $3}' txt //输出文档每行第1列,第3列

head -1 /etc/passwd > txt2 //准备新素材

awk '{print $2}' txt2 //输出第2列,无显示,没有空格做默认分隔符,当前文档一整行算一列

awk -F: '{print $5}' txt2 //使用:做分隔符,显示第5列

awk -F: '{print $7}' txt2 //使用:做分隔符,显示第5列

awk内置变量

$1 $2 $3 ..... NR行 NF列

awk -F[:/] '{print $9}' txt2 //使用多个分隔符后显示第9列

awk -F[:/] '{print NF}' txt2 //显示有几列

awk -F[:/] '{print NR}' txt2 //显示第几行

awk -F[:/] '{print NR NF}' txt2 //都显示,看不清

awk -F[:/] '{print NR,NF}' txt2 //加逗号有空格效果

awk -F[:/] '{print NR" "NF}' txt2 //或把空格引起来

awk -F[:/] '{print "当前行有"NF"列"}' txt2 //常量配合变量输出

awk -F[:/] '{print "当前行有NF列"}' txt2 //变量不能在双引号内,否则变常量输出

ifconfig eth0 |awk '/RX p/{print "当前网卡接收的流量是"$5"字节"}'

ifconfig eth0 |awk '/TX p/{print "当前网卡发送的流量是"$5"字节"}'

df -h | awk '/vda1/{print "磁盘可用空间是"$4}'

awk '/Failed/{print $11}' /var/log/secure //在安全日志中查找访问本机失败的ip地址记录

awk流程控制:

BEGIN任务 执行1次

逐行任务 执行多次,与文本有关

END任务 执行1次

awk [选项] ' BEGIN{指令} {指令} END{指令}' 文件

awk -F: 'BEGIN{print "User\tUID\tHome"}{print $1"\t"$3"\t"$6}END{print "总计",NR,"行"}' txt3 //按需求的格式输出文档内容, 有表头,有结尾统计

awk处理条件

1,正则 ~包含 !~不包含

awk -F: '$1~/root/' /etc/passwd //输出第1列包含root的行

awk -F: '$7!~/nologin$/{print $1,$7}' /etc/passwd //输出第7列不包含nologin的行的第1列和第7列

2,使用数字或字符做条件 == != > < >= <=

awk 'NR==5{print}' txt3 //输出第5行

awk 'NR!=5' txt3 //输出的结果不要第5行

awk -F: '$3==0' txt3 //输出第三列等于0的行

awk -F: '$1==root' txt3 //输出第1列等于root的行,没加引号无效

awk -F: '$1=="root"' txt3 //要求同上,加引号可以正常输出

awk -F: '$1!="root"' txt3 //输出第一列不是root的行

awk -F: '$3>1000' /etc/passwd //输出第3列大于1000的行

awk -F: '$3>=1000' /etc/passwd //大于等于

awk -F: '$3<=10' /etc/passwd //小于等于

3使用逻辑符号 && ||

awk -F: '$3<=10 && $3>1000' /etc/passwd //找小于等于10并且大于1000的,逻辑错误,找不到

awk -F: '$3<=10 || $3>1000' /etc/passwd //改成或者可以找到

awk -F: '$3>10 && $3<1000' /etc/passwd //找大于10并且小于1000的

awk -F: '$3<10 && $3>1000' /etc/passwd //逻辑错误找不到

awk 'NR>=2 && NR<=4' txt3

使用awk按需求输出文档内容 用户名 --> 密码

#!/bin/bash

u=`awk -F: '/bash$/{print $1}' /etc/passwd` //首先找到哪些用户使用的解释器是bash,找到这些用户后把名字赋予变量u

for i in $u //把这些找到的用户名轮番交给for循环处理

do

grep $i /etc/shadow | awk -F: '{print $1" --> "$2}' //如果第一次循环到的账户是root,那么grep root 可以找到shadow文档中对应的行,包含用户和密码,再使用awk筛选输出第1列用户名-->第2列密码

done