**shell day 05**

================================================

**sed中的替换，使用 s/旧内容/新内容/**

vim test.txt //准备素材,写入下列内容

2017 2011 2018

2017 2017 2024

2017 2017 2017

sed 's/2017/6666/' test.txt //替换所有行第1个2017

sed '1s/2017/6666/' test.txt //替换第1行第1个2017

sed '2s/2017/6666/' test.txt //替换第2行第1个2017

sed '2s/2017/6666/2' test.txt //替换第2行第2个2017

sed 's/2017/6666/g' test.txt //替换所有行所有2017

sed '/2024$/s/2017/6666/' test.txt //替换以2024结尾的行的

第1个2017

head -5 /etc/passwd > user

思考：如何把user文档中的/bin/bash替换成/sbin/sh

sed 's//bin/bash//sbin/sh/' user //直接替换，报错

sed 's/\/bin\/bash/\/sbin\/sh/' user //使用转义符号，可以成功，但读起来难识别

sed 's#/bin/bash#/sbin/sh#' user //更换替换符号，最佳方案

**sed中替换的高级应用**

vim nssw //创建素材，写入下列两行文本

Hello the world

ni hao ma beijing

**1）删除文档中每行的第2个、最后1个字符**

sed 's/.//2' nssw //替换所有行的第2个任意字符为空

sed 's/.$//' nssw //替换所有行的最后1个任意字符为空

sed 's/.//2;s/.$//' nssw //连在一起写中间加分号隔开

**2）将文本中所有数字删除**

sed 's/[0-9]//g' nssw //使用正则表达式找到任意数字，然后都替换为空

**3）给文档中所有的大写字母加个括号**

( ) 保留，相当于复制 \1代表粘贴

sed -r 's/([A-Z])/(\1)/g' nssw //找到文档中所有行的所有

大写字母，每找到一个就保留(复制)，然后替换的时候粘贴

该字母并套上个括号

**4）将文档中每行的第1个字符和最后一个字符互换**

echo abc > abc //准备素材

sed -r 's/(a)(b)(c)/\1\2\3/' abc //替换时多用几个括号，按顺序粘贴内容不变

sed -r 's/(a)(b)(c)/\3\2\1/' abc //粘贴时调换顺序可以实现首尾字符互换

echo xyz >> abc //追加一行xyz

sed -r 's/(a)(b)(c)/\3\2\1/' abc //这样替换就不灵了，因为只匹配abc的行

sed -r 's/(.)(.)(.)/\3\2\1/' abc //换成点，可以匹配任意三个字符的行

echo abcd >> abc //再追加abcd

sed -r 's/(.)(.)(.)/\3\2\1/' abc //这样替换就不灵了，因为匹配任意三个字符的行

sed -r 's/^(.)(.\*)(.)$/\3\2\1/' abc //中间加.\*代表匹配任意即可

sed -r 's/^(.)(.\*)(.)$/\3\2\1/' nssw //可以匹配任意文档实现首位互换

**编写脚本，安装httpd，启动时使用82号端口**

#!/bin/bash

yum -y install httpd &> /dev/null //此处要保证yum正常

#sed -i '42s/0/2/' /etc/httpd/conf/httpd.conf //方法一，找

到42行，将0换成2

sed -i '/Listen 80/s/0/2/' /etc/httpd/conf/httpd.conf //方法

二，找到有Listen 80的行，将0换成2

systemctl restart httpd

setenforce 0

systemctl stop firewalld //关闭selinux和防火墙

使用火狐浏览器访问192.168.4.7:82

a行下追加 i 行上添加 c 替换整行

sed 'a 666' user //所有行的下面追加666

sed '1a 666' user //第1行的下面追加666

sed '/root/a 666' user //有root的行的下面追加666

sed 'i 666' user //所有行的上面添加666

sed '1i 666' user //第1行的上面添加666

sed '2i 666' user //第2行的上面添加666

sed '/root/i 666' user //有root的行的上面添加666

sed 'c 666' user //替换所有行为666

sed '1c 666' user //替换第1行为666

sed '/^bin/c 666' user //替换以bin开头的行为666

-------------------------------------------------------------------

**找到使用bash的用户，然后按照"用户名 --> 密码"的格式存储在**

**一个文件中。**

**第一步：找到使用bash作登录Shell的本地用户**

sed -n '/bash$/s/:.\*//p' /etc/passwd //首先使用替换功能，找到

使用bash的用户，然后将用户信息后面的所有信息替换为空，留下

的就仅仅时用户的名字

**第二步：列出这些用户的shadow密码记录**

sed -n '/bash$/s/:.\*//p' /etc/passwd //可以找到几个用户

grep root /etc/shadow //使用root测试寻找密码的过程

x=$(grep root /etc/shadow) //将找到的结果存在变量x中

x2=${x#root:} //掐头，然后赋值给另外一个变量x2

echo ${x2%%:\*} //去尾，得到密码

**最后，按照上述思路编写脚本：**

#!/bin/bash

u=$(sed -n '/bash$/s/:.\*//p' /etc/passwd) //找到使用bash的用户名

for i in $u //把上一步找到的用户放入循环

do

x=$(grep $i: /etc/shadow) //将每个人的密码信息找出

x2=${x#\*:} //掐头

x3=${x2%%:\*} //去尾，得到密码

echo "$i --> $x3" //按 "用户-->密码" 格式喊出，如果要存到文档可以用追加重定向

done

-----------------------------------------------------------------------

**grep 易于使用，主要用来搜索**

**sed 非交互式增删改查**

**awk 精确搜索，逐行处理**

**使用方法：**

**1，前置指令 | awk 选项 条件 指令**

**2 ，awk 选项 条件 指令 被处理文档**

**选项 -F 定义分隔符**

**指令 print**

**awk内置变量 $1第1列 $2 $3 ... NR行号 NF列号**

awk '{print}' test.txt //输出所有行的内容

awk '/beijing/{print}' test.txt //输出有beijing的行的内容

awk '/beijing$/{print}' test.txt //输出以beijing结尾的行的内容

awk '{print $1}' test.txt //输出所有行的第1列

awk '{print $3}' test.txt //输出所有行的第3列

awk '/the/{print $3}' test.txt //输出有the的行的第3列

awk '/the/{print NR}' test.txt //输出有the的行的行号

awk '{print NR}' test.txt //输出所有行的行号

awk '{print NF}' test.txt //输出所有行的列号

awk '{print $0,NR}' test.txt //输出所有列然后空格输出行号

awk -F: '{print $1}' user //使用-F:修改分隔符为冒号，输出第1列

awk -F: '{print $3}' user

awk -F: '{print $1" 的id号是 "$3}' user //输出常量时使用双引号

awk -F: '{print $1" 的解释器是 "$7}' user

**利用awk提取本机的网络流量信息**

ifconfig ens33 | awk '/RX p/{print "ens33网卡的接收流量是"$5"字节"}'

ifconfig ens33 | awk '/TX p/{print "ens33网卡的发送流量是"$5"字节"}'

**利用awk提取本机的根分区剩余容量**

df -h |awk '/\/$/{print "根分区剩余容量是"$4}'

--------------------------------------------------------------------------------

**awk的处理时机，可以利用该功能，安排额外的任务**

**BEGIN{ } 执行1次，逐行任务之前的任务，任务内容可以自定义**

**{print}逐行任务，执行n次，任务内容可以自定义**

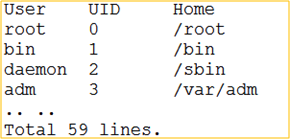
**END{ } 执行1次，逐行任务之后可以再次安排的任务，任务内容可以自定义**

awk 'BEGIN{print "666"}' // BEGIN任务，单独输出一次666

awk '{print "666"}' user //逐行任务，逐行输出666，user文档有几行就输出几次

awk 'END{print "666"}' user // END任务，单独输出一次666

利用上述方式输出以下表格



awk 'BEGIN{print "User\tUID\tHome"}' //BEGIN任务，其中\t是可以实现tab键(空格、对其)的效果

awk -F: '{print $1"\t"$3"\t"$6}' user //逐行任务，输出每行的第1列(用户名)，第3列(id号)，第6列(家目录)，这里要注意由于中间要用\t实现空格效果，而\t是常量所以需要用双引号包围

awk 'END{print "总计"NR"行"}' user //END任务，输出内容是常量与变量，NR是行号，在END任务中就是最后一行的行号

**最后，将3个任务整合在一起**

awk -F: 'BEGIN{print "User\tUID\tHome"}{print $1"\t"$3"\t"$6}END{print "总计"NR"行"}' user

----------------------------------------------------------------------------

**运维课程需要的环境**

创建虚拟机，用最小化方式安装，按要求配置好ip，同网段之间要能互通，配置好yum，修改主机名

proxy 192.168.2.5(vmnet2) 192.168.4.5(vmnet4) //必须，这台机器要2快网卡，配置2个网段

web1 192.168.2.100(vmnet2) //必须

web2 192.168.2.200(vmnet2) //必须

client 192.168.4.10(vmnet4) //可选，主要作为测试，如果没有该主机将来可以用真机测试

火狐浏览器

上述需求可以参考视频：

链接：https://pan.baidu.com/s/1d8H9bqsJ83eQOSsvNeYWOA

提取码：11ie