

# NSD SHELL DAY05

1. [案例1：sed基本用法](#)
2. [案例2：使用sed修改系统配置](#)
3. [案例3：sed多行文本处理](#)
4. [案例4：sed综合脚本应用](#)

## 1 案例1：sed基本用法

### 1.1 问题

本案例要求熟悉sed命令的p、d、s等常见操作，并结合正则表达式，完成以下任务：

- 删除文件中每行的第二个、最后一个字符
- 将文件中每行的第一个、第二个字符互换
- 删除文件中所有的数字
- 为文件中每个大写字母添加括号

### 1.2 方案

sed文本处理工具的用法：

01. 用法1：前置命令 | sed [选项] '条件指令'
02. 用法2：sed [选项] '条件指令' 文件... ..

相关说明如下：

- 条件可以是行号或者/正则/
- 没有条件时，默认为所有条件
- 指令可以是增、删、改、查等指令
- 默认sed会将所有输出的内容都打印出来，可以使用-n屏蔽默认输出
- 选项中可以使用-r选项，让sed支持扩展正则

### 1.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：认识sed工具的基本选项

sed命令的常用选项如下：

- n ( 屏蔽默认输出，默认sed会输出读取文档的全部内容 )
- r ( 让sed支持扩展正则 )
- i ( sed直接修改源文件，默认sed只是通过内存临时修改文件，源文件无影响 )

#### 1) sed命令的 -n 选项

执行p打印等过滤操作时，希望看到的是符合条件的文本。但不使用任何选项时，默认会将原始文本一并输出，从而干扰过滤效果。比如，尝试用sed输出/etc/hosts的第1行：  
[Top](#)

```

01. [ root@svr5 ~] # sed '1p' /etc/hosts
02. 127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
03. 127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
04. ::1      localhost localhost.localdomain localhost6 localhost6.localdomain6

```

可以发现所有的行都被显示出来了（第1行重复2次）。——正确的用法应该添加 -n 选项，这样就可以只显示第1行了：

```

01. [ root@svr5 ~] # sed -n '1p' /etc/hosts
02. 127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

```

行号可以是连续的行号，如打印passwd第3到第6行账户的信息：

```

01. [ root@svr5 ~] # sed -n '3,6p' /etc/passwd
02. bin:x:1:1:bin:/bin:/sbin/nologin
03. daemon:x:2:2:daemon:/sbin:/sbin/nologin
04. adm:x:3:4:adm:/var/adm:/sbin/nologin
05. lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin

```

## 2) sed命令的 -i 选项

正常情况下，sed命令所做的处理只是把操作结果（包括打印、删除等）输出到当前终端屏幕，而不会对原始文件做任何更改：

```

01. [ root@svr5 ~] # sed 'd' /etc/passwd           //删除所有行
02. [ root@svr5 ~] # cat /etc/passwd              //查看原始文本，并未改动

```

**若希望直接修改文件内容，应添加选项 -i。**

比如，直接删除test.txt（自行创建一个任意内容的文件）的第1~4行：

```

01. [ root@svr5 ~] # sed -i '1,4d' test.txt        //删除操作
02. [ root@svr5 ~] # cat test.txt                 //确认删除结果

```

**下文中关于使用sed修改文件的示例中，为了避免大家在练习过程中因误操作导致系统故障，命令省略 -i 选项，不再逐一说明。需要时，大家可自行加上此选项。**

## 3) 多个指令可以使用分号隔离

用分号来隔离多个操作，比如：

```
01. [ root@svr5 ~] # sed - n '1p; 4p' /etc/passwd
02. root:x:0:0:root:/root:/bin/bash
03. adm:x:3:4:adm:/var/adm:/sbin/nologin
```

## 步骤二：认识sed工具的条件

# sed [选项] '条件指令' 文件.. ..

sed命令可以使用行号或正则做为条件匹配：

1) 行号案例

打印第3行：

```
01. [ root@svr5 ~] # sed - n '3p' /etc/passwd
```

打印第3到5行：

```
01. [ root@svr5 ~] # sed - n '3,5p' /etc/passwd
```

打印第3和5行：

```
01. [ root@svr5 ~] # sed - n '3p; 5p' /etc/passwd
```

打印第3以及后面的10行：

```
01. [ root@svr5 ~] # sed - n '3,+10p' /etc/passwd
```

打印奇数行：

```
01. [ root@svr5 ~] # sed - n '1~2p' /etc/passwd
```

打印偶数行：

[Top](#)

```
01 [root@svr5 ~]# sed -n '2~2p' /etc/passwd
```

## 2) 正则案例

打印包含root的行：

```
01 [root@svr5 ~]# sed -n '/root/p' /etc/passwd
```

打印bash结尾的行：

```
01 [root@svr5 ~]# sed -n '/bash$/p' /etc/passwd
```

## 3) 没有条件，则表示匹配所有行

```
01 [root@svr5 ~]# sed -n 'p' /etc/passwd
```

## 步骤三：sed工具的p、d、s操作指令案例集合

1) 下面看看sed工具的p指令案例集锦（自己提前生成一个a.txt文件）

```
01 [root@svr5 ~]# sed -n 'p' a.txt //输出所有行，等同于cat a.txt
02 [root@svr5 ~]# sed -n '4p' a.txt //输出第4行
03 [root@svr5 ~]# sed -n '4,7p' a.txt //输出第4~7行
04 [root@svr5 ~]# sed -n '4,+10p' a.txt //输出第4行及其后的10行内容
05 [root@svr5 ~]# sed -n '/^bin/p' a.txt //输出以bin开头的行
06 [root@svr5 ~]# sed -n '$=' a.txt //输出文件的行数
```

2) 下面看看sed工具的d指令案例集锦（自己提前生成一个a.txt文件）

```
01 [root@svr5 ~]# sed '3,5d' a.txt //删除第3~5行
02 [root@svr5 ~]# sed '/xml/d' a.txt //删除所有包含xml的行
03 [root@svr5 ~]# sed '/xml/!d' a.txt //删除不包含xml的行，!符号表示取反
04 [root@svr5 ~]# sed '/^install/d' a.txt //删除以install开头的行
05 [root@svr5 ~]# sed '$d' a.txt //删除文件的最后一行
06 [root@svr5 ~]# sed '/^$/d' a.txt //删除所有空行
```

[Top](#)

### 3) sed命令的s替换基本功能 ( s/旧内容/新内容/选项 ) :

```

01. [ root@svr5 ~] # vim test.txt           //新建素材
02. 2017 2011 2018
03. 2017 2017 2024
04. 2017 2017 2017
05.
06. [ root@svr5 ~] # sed 's/2017/xxxx/'    test.txt
07. [ root@svr5 ~] # sed 's/2017/xxxx/g'    test.txt
08. [ root@svr5 ~] # sed 's/2017/xxxx/2'    test.txt
09. [ root@svr5 ~] # sed 's/2017//2'        test.txt
10. [ root@svr5 ~] # sed -n 's/2017/xxxx/p' test.txt

```

### 4) 下面看看sed工具的s指令案例集锦 ( 自己提前生成一个a.txt文件 )

注意：替换操作的分隔 "/" 可改用其他字符，如#、&等，便于修改文件路径

```

01. [ root@svr5 ~] # sed 's/xml/XML/' a.txt //将每行中第一个xml替换为XML
02. [ root@svr5 ~] # sed 's/xml/XML/3' a.txt //将每行中的第3个xml替换为XML
03. [ root@svr5 ~] # sed 's/xml/XML/g' a.txt //将所有的xml都替换为XML
04. [ root@svr5 ~] # sed 's/xml//g' a.txt //将所有的xml都删除 (替换为空串)
05. [ root@svr5 ~] # sed 's#/bin/bash#/sbin/sh#' a.txt //将/bin/bash替换为/sbin/sh
06. [ root@svr5 ~] # sed '4,7s/^/#/' a.txt //将第4~7行注释掉 (行首加#号)
07. [ root@svr5 ~] # sed 's/^#an/an/' a.txt //解除以#an开头的行的注释 (去除行首的#号)

```

### 步骤四：利用sed完成本例要求的任务

参考数据文件内容如下：

```

01. [ root@svr5 ~] # cat nssw.txt
02. Hello the world
03. ni hao ma beijing

```

本小节的操作使用nssw.txt作为测试文件。

#### 1) 删除文件中每行的第二个、最后一个字符

分两次替换操作，第一次替换掉第2个字符，第二次替换掉最后一个字符：

[Top](#)

```
01 [root@svr5 ~]# sed 's/.//2; s/.$//' nssw.txt
```

2) 将文件中每行的第一个、倒数第1个字符互换

每行文本拆分为“第1个字符”、“中间的所有字符”、“倒数第1个字符”三个部分，然后通过替换操作重排顺序为“3-2-1”：

```
01 [root@svr5 ~]# sed -r 's/^(.)(.*)(.)$/\3\2\1/' nssw.txt
```

3) 删除文件中所有的数字

因原文件内没有数字，行首也没有空格，这里稍作做一点处理，生成一个新测试文件：

```
01 [root@svr5 ~]# sed 's/[0-9]//' nssw.txt
```

以nssw2.txt文件为例，删除所有数字、行首空格的操作如下：

```
01 [root@svr5 ~]# sed -r 's/[0-9]//g; s/^( )+//' nssw2.txt
```

4) 为文件中每个大写字母添加括号

使用“( )”可实现保留功能，所以可参考下列操作解决：

```
01 [root@svr5 ~]# sed -r 's/([A-Z])/[\1]/g' nssw.txt
```

## 2 案例2：使用sed修改系统配置

### 2.1 问题

本案例要求熟悉课上的sed应用案例，并编写脚本anonftp.sh，实现以下功能：

- 通过yum安装vsftpd软件包
- 修改vsftpd服务配置，开启匿名上传
- 调整/var/ftp/pub目录权限，允许写入
- 启动vsftpd服务，并设置开机自运行

### 2.2 步骤

实现此案例需要按照如下步骤进行。

[Top](#)

**步骤一：编写anonftp.sh脚本，用来装配匿名FTP服务**

### 1) 任务需求及思路分析

vsftpd服务的安装、改目录权限、起服务等操作可以直接写在脚本中。

修改vsftpd.conf配置的工作可以使用sed命令，根据默认配置，只需要定位到以#anon开头的行，去掉开头的注释即可。

### 2) 根据实现思路编写脚本文件

```

01. [ root@svr5 ~] # vim anonftp.sh
02.  #! /bin/bash
03.  yum -y install vsftpd                //安装vsftpd软件
04.  cp /etc/vsftpd/vsftpd.conf{,.bak}    //备份默认的配置文
05.  sed -i "s/^#anon/anon/" /etc/vsftpd/vsftpd.conf //修改服务配置
06.  chmod 777 /var/ftp/pub                //调整目录权限
07.  systemctl start vsftpd              //启动服务
08.  systemctl enable vsftpd             //设为自动运行
09.
10. [ root@svr5 ~] # chmod +x anonftp.sh
11. [ root@svr5 ~] # ./anonftp.sh

```

## 3 案例3：sed多行文本处理

### 3.1 问题

本案例要求使用sed工具来完成下列任务操作：

- 修改主机名配置文件
- 修改hosts文件，添加两条映射记录：192.168.4.5 与 svr5.tarena.com、svr5，还有119.75.217.56与www.baidu.com

### 3.2 方案

# sed [选项] '条件指令' 文件..

sed工具的多行文本处理操作：

- i：在指定的行之前插入文本
- a：在指定的行之后追加文本
- c：替换指定的行

### 3.3 步骤

基本语法格式案例：

注意：系统默认没有a.txt文件，需要自己创建一个测试文件！！

```

01. [ root@svr5 ~] # sed '2a XX' a.txt    //在第二行后面，追加XX
02. [ root@svr5 ~] # sed '2i XX' a.txt    //在第二行前面，插入XX

```

[Top](#)

```
03. [root@svr5 ~] # sed '2c XX' a.txt //将第二行替换为XX
```

实现此案例需要按照如下步骤进行。

### 步骤一：修改主机名配置文件

#### 1) 确认修改前的配置

```
01. [root@svr5 ~] # cat /etc/hostname
02. svr5.tarena.com
```

#### 2) 使用sed修改主机名配置所在行的内容（c整行替换）

```
01. [root@svr5 ~] # sed '1c mysvr.tarena.com' /etc/hostname
```

### 步骤二：修改hosts文件，添加新的记录

#### 1) 确认修改前的配置

```
01. [root@svr5 ~] # cat /etc/hosts
02. 127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
03. ::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
```

#### 2) 使用sed修改hosts文件，添加两行新纪录（a追加）

```
01. [root@svr5 ~] # sed -i '$a 192.168.4.5 svr5.tarena.com svr5' /etc/hosts
02. 127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
03. ::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
04. 192.168.4.5 svr5.tarena.com svr5
```

## 4 案例4：sed综合脚本应用

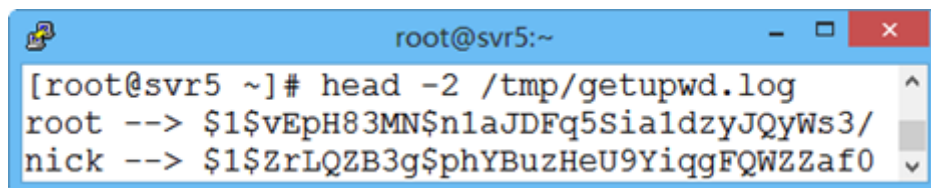
### 4.1 问题

本案例要求编写脚本getupwd.sh，实现以下需求：

- 找到使用bash作登录Shell的本地用户
- 列出这些用户的shadow密码记录
- 按每行“用户名 --> 密码记录”保存到getupwd.log，如图-1所示

[Top](#)





```

root@svr5:~
[ root@svr5 ~]# head -2 /tmp/getupwd.log
root --> $1$vEpH83MN$nlajDFq5SialdzyJQyWs3/
nick --> $1$ZrLQZB3g$phYBuzHeU9YiqgFQWZZaf0

```

图 - 1

## 4.2 方案

基本思路如下：

1. 先用sed工具取出登录Shell为/bin/bash的用户记录，保存为临时文件/tmp/urec.tmp，并计算记录数量
2. 再结合while循环遍历取得的账号记录，逐行进行处理
3. 针对每一行用户记录，采用掐头去尾的方式获得用户名、密码字符串
4. 按照指定格式追加到/tmp/getupwd.log文件
5. 结束循环后删除临时文件，报告分析结果

## 4.3 步骤

实现此案例需要按照如下步骤进行。

### 步骤一：编写getupwd.sh脚本

```

01. [ root@svr5 ~]# vim ./getupwd.sh
02.  #/bin/bash
03.  A=$( sed -n '/bash$/s/:.*//p' /etc/passwd)      ## 提取符合条件的账号记录
04.  for i in $A                                     ##遍历账号记录
05.  do
06.      pass1=$( grep $i /etc/shadow)
07.      pass2=${ pass1#*:}
08.      pass=${ pass2%%\* *}
09.      echo "$i  --> $pass"
10.  done
11.
12. [ root@svr5 ~]# chmod +x ./getupwd.sh

```

### 步骤二：测试、验证执行结果

```

01. [ root@svr5 ~]# ./getupwd.sh
02. 用户分析完毕，请查阅文件 /tmp/getupwd.log
03.
04. [ root@svr5 ~]# less /tmp/getupwd.log
05. root --> $6$IWgMYmRA Cwdbf wBo $dr8Yn983nswiVwOdT MjzbDv SLeCd1GMjYjbv sDiFEkL8jnXOLcocB
06. zengy e --> $6$Qb37LOdzRl5995Pl $L0zT OgnhGz8ihWkW81J.5XhPp/I7x2./Me2ag0S8tRndCBL9nljHIK

```

[Top](#)

07. clamav -->!!
08. my sql -->!!
09. abc -->!!
10. ....



从上述参考脚本可以发现，使用sed来实现字段提取会比较复杂。下一章课程将会学到awk命令，届时可以通过更简单的方法来改进此脚本内容。

总结知识点：

#sed [选项] '条件指令' 文件

选项:

-n 屏蔽默认输出

-r 支持扩展正则

-i 修改源文件

条件：

行号 4 4,5 4~2 4,+10

/正则/

指令：

p 打印

d 删除

s 替换s/旧/新/g

a 追加

i 插入

c 替换行

[Top](#)