NSD SHELL DAY06

1 案例1：awk流程控制

1.1 问题

本案例要求了解awk的流程控制操作，可自行设置awk语句来有验证以下操作：

- if分支结构（单分支、双分支、多分支）
- 练习awk数组的使用

1.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：awk过滤中的if分支结构

1）单分支

统计/etc/passwd文件中UID小于或等于1000的用户个数：

```
[root@svr5 ~]# awk -F: '{if($3<=1000){i++}}END{print i}' /etc/passwd
39
```

统计/etc/passwd文件中UID大于1000的用户个数：

```
[root@svr5 ~]# awk -F: '{if($3>1000){i++}}END{print i}' /etc/passwd
8
```

统计/etc/passwd文件中登录Shell是“/bin/bash”的用户个数：

```
[root@svr5 ~]# awk -F: '{if($7~/bash$/){i++}}END{print i}'  /etc/passwd
29
```

2）双分支

分别统计/etc/passwd文件中UID小于或等于1000、UID大于1000的用户个数：

Top

```
[root@svr5 ~]# awk -F: '{if($3<=1000){i++}else{j++}}END{print i,j}' /etc/passwd
39 8
```

分别统计/etc/passwd文件中登录Shell是“/bin/bash”、 登录Shell不是“/bin/bash”的用户个数：

```
[root@svr5 ~]# awk -F: '{if($7~/bash$/){i++}else{j++}} END{print i,j}' /etc/passwd
29 38
```

步骤二：awk数组

1）数组的语法格式

数组是一个可以存储多个值的变量，具体使用的格式如下：

定义数组的格式：数组名[下标]=元素值

调用数组的格式：数组名[下标]

遍历数组的用法：for(变量 in 数组名){print 数组名[变量]}。

```
[root@svr5 ~]# awk 'BEGIN{a[0]=11;a[1]=88;print a[1],a[0]}'
88 11
[root@svr5 ~]# awk 'BEGIN{a++;print a}'
1
[root@svr5 ~]# awk 'BEGIN{a0++;print a0}'
1
[root@svr5 ~]# awk 'BEGIN{a[0]++;print a[0]}'
1
[root@svr5 ~]# awk 'BEGIN{a[0]=0;a[1]=11;a[2]=22; for(i in a){print i,a[i]}}'
0 0
1 11
2 22
```

注意，awk数组的下标除了可以使用数字，也可以使用字符串，字符串需要使用双引号：

```
[root@svr5 ~]# awk 'BEGIN{a["hehe"]=11;print a["hehe"]}'
11
```

Top

2 案例2：awk扩展应用

2.1 问题

本案例要求使用awk工具完成下列两个任务：

- 分析Web日志的访问量排名，要求获得客户机的地址、访问次数，并且按照访问次数排名

2.2 方案

1）awk统计Web访问排名

在分析Web日志文件时，每条访问记录的第一列就是客户机的IP地址，其中会有很多重复的IP地址。因此只用awk提取出这一列是不够的，还需要统计重复记录的数量并且进行排序。

通过awk提取信息时，利用IP地址作为数组下标，每遇到一个重复值就将此数组元素递增1，最终就获得了这个IP地址出现的次数。

针对文本排序输出可以采用sort命令，相关的常见选项为-r、-n、-k。其中-n表示按数字顺序升序排列，而-r表示反序，-k可以指定按第几个字段来排序。

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：统计Web访问量排名

分步测试、验证效果如下所述。

1）提取IP地址及访问量

```
[root@svr5 ~]# awk '{ip[$1]++} \
>  END{for(i in ip) {print ip[i],i }}' /var/log/httpd/access_log
4  127.0.0.1
17 192.168.4.5
13 192.168.4.110
.. ..
```

2）对第1）步的结果根据访问量排名

```
[root@svr5 ~]# awk  '{ip[$1]++} END{for(i in ip) {print i,ip[i]}}' /var/log/httpd/access_lo
17 192.168.4.5
13 192.168.4.110
4 127.0.0.1
.. ..
```

3 案例3：编写监控脚本

3.1 问题

本案例要求编写脚本，实现计算机各个性能数据监控的功能，具体监控项目要求如下Top

- CPU负载
- 网卡流量

- 内存剩余容量
- 磁盘剩余容量
- 计算机账户数量
- 当前登录账户数量
- 计算机当前开启的进程数量
- 本机已安装的软件包数量

3.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：准备工作

1）查看性能数据的命令

```
[root@svr5 ~]# uptime                    //查看CPU负载
[root@svr5 ~]# ifconfig eth0             //查看网卡流量
[root@svr5 ~]# free                     //查看内存信息
[root@svr5 ~]# df                       //查看磁盘空间
[root@svr5 ~]# wc -l /etc/passwd          //查看计算机账户数量
[root@svr5 ~]# who |wc -l                //查看登录账户数量
[root@svr5 ~]# rpm -qa |wc -l             //查看已安装软件包数量
```

步骤二：编写参考脚本

1）脚本内容如下：

```
[root@svr5 ~]# vim test.sh
#!/bin/bash
ip=`ifconfig eth0 | awk '/inet /{print $2}'`
echo "本地IP地址是:"$ip
cpu=`uptime | awk '{print $NF}'`
#awk中NF为当前行的列数，$NF是最后一列
echo "本机CPU最近15分钟的负载是:"$cpu
net_in=`ifconfig eth0 | awk '/RX p/{print $5}'`
echo "入站网卡流量为:"$net_in
net_out=`ifconfig eth0 | awk '/TX p/{print $5}'`
echo "出站网卡流量为:"$net_out
mem=`free | awk '/Mem/{print $4}'`
echo "内存剩余容量为:"$mem
disk=`df | awk '/\/$/{print $4}'`
echo "根分区剩余容量为:"$disk
user=`cat /etc/passwd |wc -l`
echo "本地账户数量为:"$user
login=`who | wc -l`
```

Top

```
19.  echo "当前登陆计算机的账户数量为:"$login
20.  process=`ps aux | wc -l`
21.  echo "当前计算机启动的进程数量为:"$process
22.  soft=`rpm -qa | wc -l`
23.  echo "当前计算机已安装的软件数量为:"$soft
```

4 案例4：编写安全检测脚本

4.1 问题

本案例要求编写脚本，防止远程ssh暴力破解密码，具体监控项目要求如下：

- 检测ssh登录日志，如果远程登陆账号名错误3次，则屏蔽远程主机的IP
- 检测ssh登录日志，如果远程登陆密码错误3次，则屏蔽远程主机的IP

4.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：准备工作

1）过滤帐户名失败的命令(登陆日志文件为/var/log/secure)

```
01.  [root@svr5 ~]# awk '/Invalid user/{print $10}' /var/log/secure
```

2）过滤密码失败的命令

```
01.  [root@svr5 ~]# awk '/Failed password/{print $11}' /var/log/secure
```

步骤二：编写参考脚本

1）脚本内容如下：

```
01.  [root@svr5 ~]# vim test.sh
02.  #!/bin/bash
03.  awk '/Failed password/{print $11}' /var/log/secure  | awk '{ip[$1]++}END{for(i in ip){pr
04.
05.  awk '/Invalid user/{print $10}' /var/log/secure  | awk '{ip[$1]++}END{for(i in ip){print ip
```

Top