5_Shell07 高级 awk 流程控制数组

一 几种查找方式对比

1.1 vim:交互式 增删改查

1.2 grep:模糊查找

1.3 sed:非交互式 增删改查

1.4 awk:精确查找 行列查找 对查找的数据进行 2 次处理

1.4.1 格式:

前置指令 | awk [选项] '[条件]{指令}'

awk [选项] '[条件]{指令}' 文件

选项: -F 指定分隔符

指令: print

1.4.2 条件:

正则表达式;~ !~(包含与不包含)

字符串与数值比较: == != > < >= <=

逻辑符号 && ||(逻辑或 逻辑与)

1.4.3 awk 简单流程控制任务

BEGIN{指令} 执行 1 次

{指令} 执行 n 次

END{指令} 执行 1 次

以上 3 种方式可单独使用,也可匹配使用

二 awk 流程控制

2.1 awk 分支结构

2.1.1 awk 单分支 if

格式:awk [选项] ‘{if(条件){执行指令}}’

2.1.2 awk 双分支 if

格式:awk [选项] ‘{if(条件){执行指令 1}else{执行指令 2}}’

2.1.3 awk 多分支 if

格式:awk [选项] ‘{if(条件){执行指令 1}else if{执行指令 2}...else{执行指令 n}}’

```
]# awk -F: '{if($3>=1000){x++;}}END{print x}' /etc/passwd
    #awk 单分支 if
2    #输出 UID 大于等于 1000 的用户数,END{print x}表示在最后一行执行
]# awk -F: '{if($3>=1000){x++}else{y++}}END{print x,y}'
/etc/passwd    #awk 双分支 if
2 37    #输出 UID 大于等于 1000 的用户数,和 UID 小于 1000 的用户数
```

三 awk 数组

3.1 数组的定义及使用

3.1.1 定义格式: **数组名[下标]=元素值** #实现 1 个名称存多个值,加强版变量

下标和元素值可以为除整数以外的其他类型的值,使用时加””

3.1.2 调用数组格式: **数组名[下标]**

3.1.3 遍历数组用法: **for(变量 in 数组名){print 数组名[变量]}**

```
]# awk 'BEGIN{a[1]=10;a[2]=20;a[3]=30;print a[1],a[2],a[3]}'
10 20 30    #数组定义与调用输出
[root@desktop0 opt]# head -5 /etc/passwd > user
[root@desktop0 opt]# awk '{a[1]++}END{print a[1]}' user
5  #因为 user 有 5 行,{a[1]++执行了 5 次},最后 END{print a[1]}输出为 5
```

3.2 遍历数组

3.2.1 格式: **for(变量 in 数组名){print 数组名[变量]}**

for 循环 循环的是数组的下标

```
]# 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  #定义 1 个数组 a,有 3 个下标,对应 3 个值,用 for 循环循环显示下标和值
]# cat abc
abc
abc
xyz
]# awk '{a[$1]++}END{for(i in a){print i,a[i]}}' abc
abc 2
xyz 1
```

#文档 abc 第 1 行,a[$1]为 a[abc],{a[$1]++}使 a[abc]=1

#文档 abc 第 2 行,a[$1]为 a[abc],此时 a[abc]=1,{a[$1]++}使 a[abc]=2

#文档 abc 第 3 行,a[$1]为 a[xyz],{a[$1]++}使 a[xyz]=1

#最后数组 a 为 a[abc]=2,a[xyz]=1

#for(i in a)使 i 分别为 abc xyz,a[i]分别为 2 1

```
[root@desktop0 opt]# cat abc
abc 192.168.0.1
abc 192.168.0.1
xyz 192.168.0.2
opq 192.168.0.3
opq 192.168.0.3
opq 192.168.0.3
[root@desktop0  opt]#  awk  '{ip[$2]++}END{for(i  in  ip){print
i,ip[i]}}' abc
192.168.0.1 2
192.168.0.2 1
192.168.0.3 3
```

3.2.2 案例:统计网站访问量

server 安装 httpd 服务,设置防火墙为 trusted,设置 setenforce 0

desktop 真机: firefox 172.25.0.11

web 日志文件:/var/log/httpd/access_log

server 执行命令:

```
]# awk '{ip[$1]++}END{for(i in ip){print "IP 为"i"的主机访问本机 web 服务"ip[i]"次!"}}' /var/log/httpd/access_log
IP 为 172.25.0.10 的主机访问本机 web 服务 16 次!
IP 为 172.25.0.11 的主机访问本机 web 服务 12 次!
IP 为 172.25.0.250 的主机访问本机 web 服务 12 次!
```

server 测试:]# curl 172.25.0.11 &> /dev/null #执行 3 次

server 再执行命令:

```
]# awk '{ip[$1]++}END{for(i in ip){print "IP 为"i"的主机访问本机 web 服务"ip[i]"次!"}}' /var/log/httpd/access_log
IP 为 172.25.0.10 的主机访问本机 web 服务 16 次!
IP 为 172.25.0.11 的主机访问本机 web 服务 15 次!
IP 为 172.25.0.250 的主机访问本机 web 服务 12 次!
```

3.2.3 对访问量进行排序 sort -n 表示对数字排序 -r 降序排列

```
[root@server0 httpd]# awk '{ip[$1]++}END{for(i in ip){print ip[i],i}}' /var/log/httpd/access_log | sort -nr
16 172.25.0.10
15 172.25.0.11
12 172.25.0.250
```

3.2.4 使用 ab 命令对网页进行压力测试 -n 访问次数 -c 并发数量

ab 随 httpd-tools(httpd 依赖包)安装完成

server:

```
]# ab -n 1000 -c 1 172.25.0.11/

]# awk '{ip[$1]++}END{for(i in ip){print ip[i],i}}'
/var/log/httpd/access_log | sort -nr

1015 172.25.0.11

16 172.25.0.10

12 172.25.0.250
```

3.2.5 编写脚本,实现计算机各个性能数据的监控功能,具体监控项目要求如下:

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

```
#!/bin/bash

while :                    #不停执行

do

#uptime                    #查看 CPU 负载

uptime | awk '{print "CPU 平均负载是"$8,$9,$10}'

#ifconfig eth0             #查看网卡流量

ifconfig eth0 | awk '/RX p/{print "网卡 eth0 接收的数据量为"$5"字节
"}'
```

```
ifconfig eth0 | awk '/TX p/{print "网卡 eth0 发送的数据量为"$5"字节
"}'
#free                          #查看内存信息
free -m | awk '/Mem:/{print "物理内存可用余量为"$4"MB"}'
#df                            #查看磁盘空间
df -h | awk '/vda1/{print "根分区可用余量是"$4"G"}'
#wc -l /etc/passwd    #查看计算机账户数量
user=$(cat /etc/passwd | wc -l)
echo "计算机账户数量为$user 个"
#who |wc -l                    #查看登录账户数量
u=$(who | wc -l)
echo "当前服务器登录的用户数是$u 个"
p=$(ps aux | wc -l)   #查看计算机进程数
echo "当前计算机运行的进程数为$p 个"
#rpm -qa |wc -l        #查看已安装软件包数量
ins=$(rpm -qa | wc -l)
echo "当前主机安装软件包数量是$ins 个"
sleep 3                        #等待 3 秒
clear                          #清屏,为下一次输出做准备
done
```

3.2.6 编写脚本,监控服务器,如果发现有人尝试登录失败 5 次,则发邮件给管理员,告诉管理员是那个 IP 尝试登录.

安全日志路径:**/var/log/secure**

```
#!/bin/bash
x=$(awk '/Failed/{ip[$11]++}END{for(i in ip){print ip[i]","i}}' /var/log/secure)    #输出格式为”次数,IP 地址”
for i in $x
   do
      n=${i%,*}    #去尾,获取次数
      p=${i#*,}    #掐头,获取 IP
      [ $n -ge 5 ] && echo "$p 访问本机失败了$n 次!" | mail -s test root    #将登录失败次数与 5 比较,若大于等于 5 则执行发邮件代码
   done
```

删除之前的邮件: `rm -rf /var/spool/mail/root`