Lab 1- Shell Programming Report

DV1457- Programming in UNIX Environment

Chen Hang,dong
19960112-1297
Blekinge Institute of Technology
chinatony112@outlook.com

Peng Rong
19951120-7004
Blekinge Institute of Technology
ropeaudrey@gmail.com

1. How to limit the number of results to N

   First, we use condition statement to judge whether the inputted parameters containing "**-n N**", if true, using "**head -$N**" to output the limited result list, otherwise, output the whole result list.

2. How to find the most number of connection attempts

```
cat $FILENAME|awk '{print $1}'|sort|uniq -c > data/c.txt
cat data/c.txt | sort -b -n -k 1,1 -r | awk '{print $2,"\t",$1}'
```

   First we catch the user inputted log file, then using "**awk '{print $1}'**" for selecting the first column(IP address column).Secondly we using command "**uniq –c**" to count the number of each IP's connection attempts times, then using "**sort –b –n –k 1,1 -r**" to sort the number of times and output the most number of connection attempts list, the number of results will be presented according to the user input.

3. How to find the most number of successful connection attempts

```
cat $FILENAME|awk '{if(match($9,/^2[0-9][0-9]/)) print $1,$9}'
      |sort|uniq -c > data/2.txt
cat data/2.txt | sort -b -n -k 1,1 -r | awk '{print $2,"\t",$1}'
```

   According to the RFC 1945[1], we believe the HTTP Status Code in the log file which starting with 2 considered to be successful attempt. So, we are using "**match()**" methods to pick and match our pattern of HTTP code, then count the number of times of each IP's successful attempts, sort the results and print the result list.

4. How to find the most common results codes and corresponding IP address

```
cat $FILENAME | awk '{print $9,"\t",$1}' | sort -r -n -t "." -k1,1n -k2,2n -
k3,3n -k4,4n > data/r.txt
#count for result code and select most common one (code)
code=$(cat data/r.txt | awk '{print $1}' | uniq -c | sort -r -n -k 1,1 | awk
'{print $2}' |head -1)
```

   First, we use command "**sort –r –n –t "." –k1,1n –k2,2n –k3,3n –k4,4n**" to sort the IP addresses, because we hope these IP addresses which with the same status code can be collected together. Secondly, we use "**uniq –c"** and "**sort –r –n –k 1,1**"to find the most common status codes and save it in a variable. Finally, we use "match()" to pick those IP addresses which with the status code equals to the value of the variable.

5. How to find the most common failure (no auth, not found etc.) codes and the corresponding IP

```
#sort by results code and IP,output into file (prepare for print on screen)
cat $FILENAME | awk '{print $9,"\t",$1}' | sort -r -n -t "." -k1,1n -k2,2n -
k3,3n -k4,4n > data/F.txt
# select 4**, 5** , count for result code and select most common one (code)
code=$(cat data/F.txt | awk '{print $1}' | egrep  "[4-5][0-9]{2}" | uniq -c |
sort -r -n -k 1,1 | awk '{print $2}' |head -1)
```

   This question is similar to question 4, but it need to select all failure codes before running the step of question 4. According to the RFC 1945[1], the failure code including all start with 4 and 5, which means all 4** and 5** codes are failure code, so we use "**egrep [4-5][0-9]{2}**" to grep these failure codes. Count how many times does one code repeat and select most common failure result code. Finally using "**awk**" command to select IP address and print on screen

6. How to get the sum of bytes of each IP and to know which IP number sent the most bytes

   In this question, there are 1 challenges to deal with.

   1) How to count the sum of bytes of each IP. In this challenge, we use loop to read lines and using array to output the calculated result. Because of the IP has been sorted, when reading lines, if it gets the same IP, the bytes will be added, if the IP is different, the array tag will move to next one and reset the bytes.

```
while read line
do
        t=($line)
        if [ "${t[0]}" == "$tempIP" ];then
                let bytes=$[$bytes+${t[1]}]
                cal[$tag]="$tempIP $bytes"
        else
                if [ "$tag" -ne "-1" ];then
                        cal[$tag]="$tempIP $bytes"
                fi
                let tag++
                tempIP="${t[0]}"
                let bytes=$[${t[1]}]
        fi
done < data/t.txt
```

7. Blacklist check

   In this function, there are two challenges.

   1) Analysis the web name and get IP address. In this part, we use "**nslookup $name**" command, by using loop to read each name in the blacklist and using "grep Add" command to get IP Address.

   2) Compare the result IP (result from previous command E.g. -c, -t etc.) with the IP in the blacklist. In this part, we use double loop to deal with it. The first loop is to read result IP line by line, the second loop is to read IP in blacklist. Each result IP will run and scan all IP address in blacklist.

   If the result IP is in the blacklist, it will print "blacklist" after result line.

8. Command line check

   Command line check is used for checking if user input correct parameters and formats. For all input that does not match the format, it will print correct format on the screen.

```
log_sum.sh [-n N] (-c|-2|-r|-F|-t) [-e] <filename>
```

   First, the number of parameters should more than 1 and less than 6, and we discuss each situation according to the number of input parameters. In this discuss if-then judgement, the function will check If there is "-n", check if the "N" is a digit, check if there is "-e", check if the file is existing. If format is correct, based on check result, it will analysis and select result from file.

9. Reference

[1] https://tools.ietf.org/html/rfc1945#section-9