Information Security Management :

Log File Analysis using kali-linux commands

“with explanation of each Command”

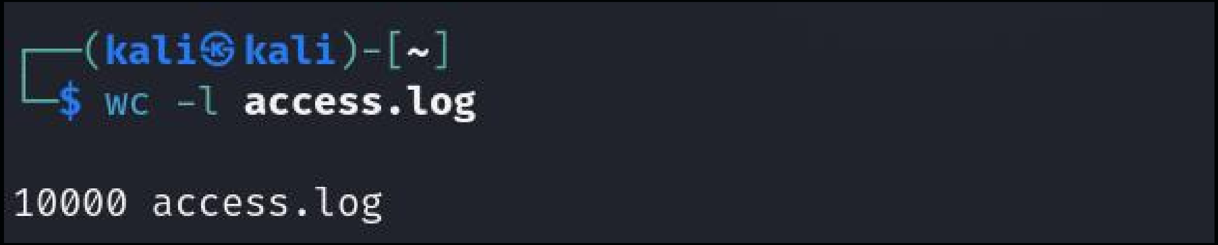
Name : Ali Mohamed Oqab - 2205077
Instructor : Eng.yahya ashraf

This is my LogFile Link :

Press Here : LogFile

Let’s Start With Commands :

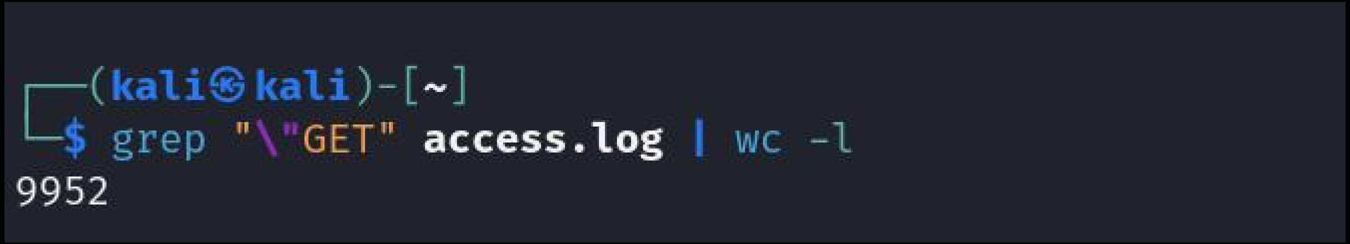
1. wc -l access.log
→ Count total number of requests (lines in the log file)



```
┌──(kali㉿kali)-[~]
└─$ wc -l access.log

10000 access.log
```

2. grep "\"GET" access.log | wc -l

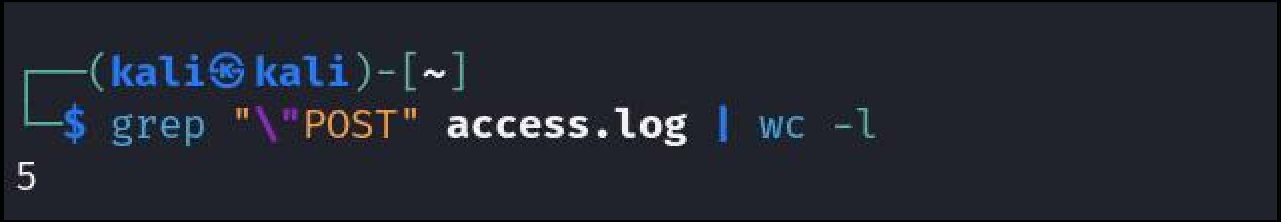
→ Count total number of requests (lines in the log file)



```
┌──(kali㉿kali)-[~]
└─$ grep "\"GET" access.log | wc -l
9952
```

3. grep "\"POST" access.log | wc -l

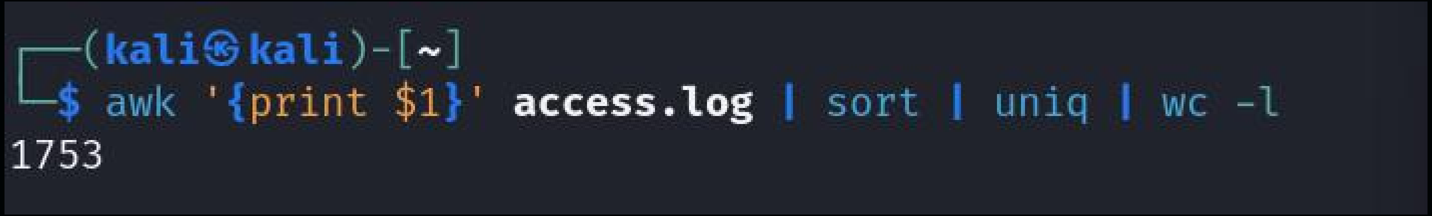
→ Count how many of those are POST requests



```
┌──(kali㉿kali)-[~]
└─$ grep "\"POST" access.log | wc -l
5
```

4. awk '{print $1}' access.log | sort | uniq | wc -l

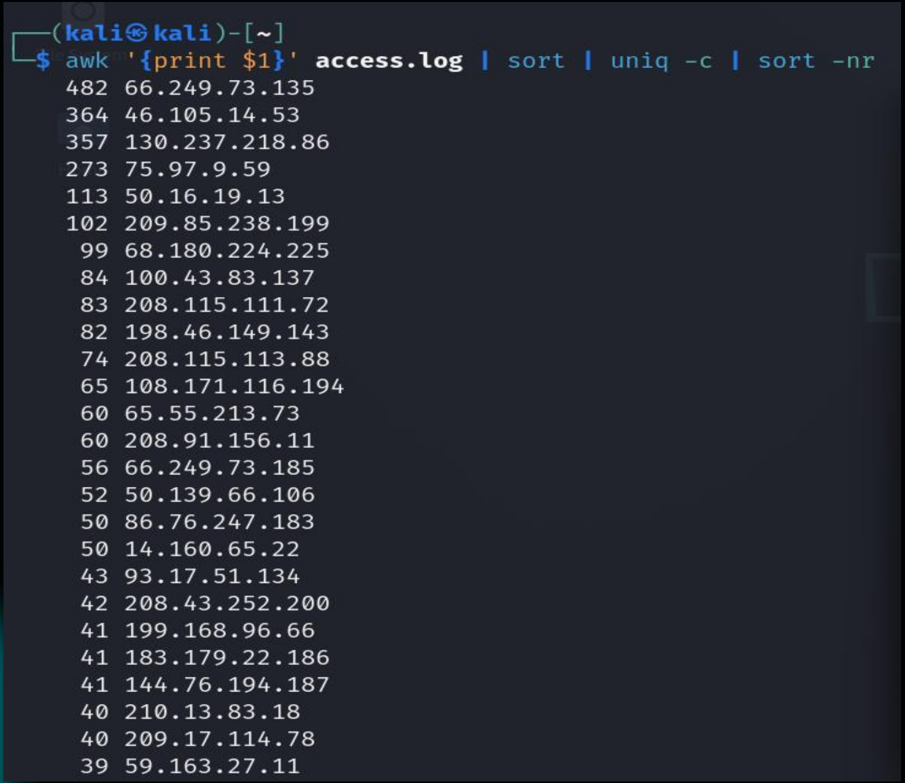
→ Count total number of unique IP addresses



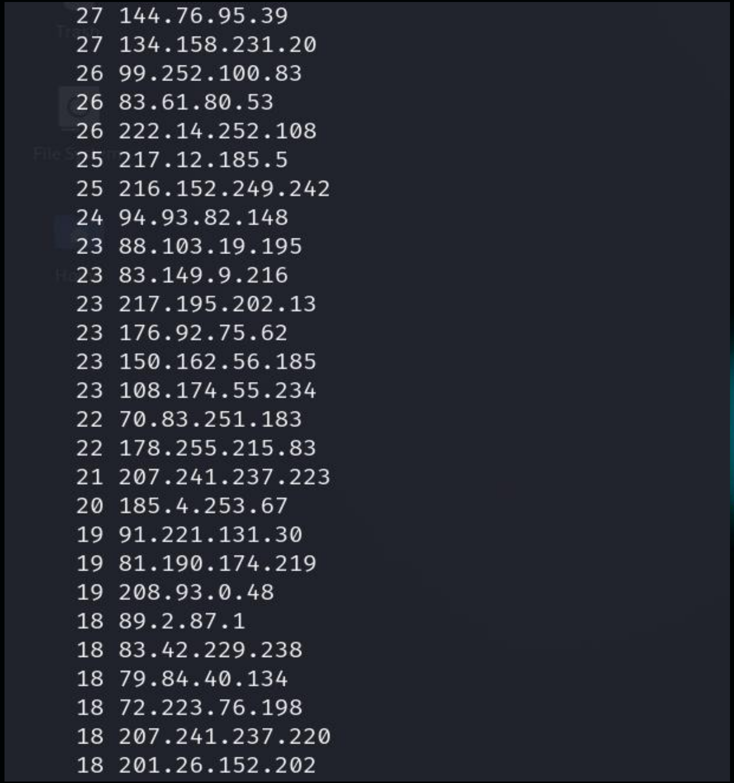
```
┌──(kali㉿kali)-[~]
└─$ awk '{print $1}' access.log | sort | uniq | wc -l
1753
```

5. awk '{print $1}' access.log | sort | uniq -c | sort -nr

→ Show how many requests were made by each IP

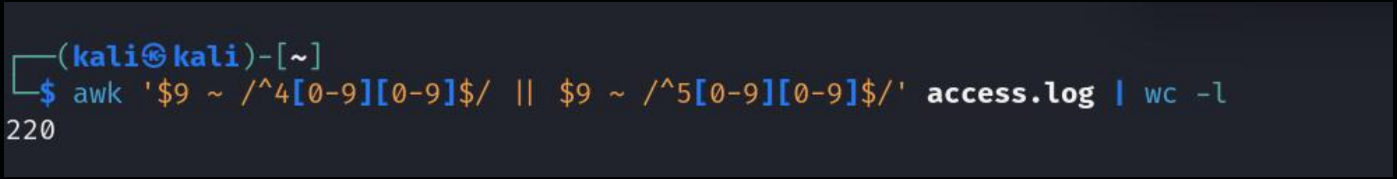


```
┌──(kali㉿kali)-[~]
└─$ awk '{print $1}' access.log | sort | uniq -c | sort -nr
    482 66.249.73.135
    364 46.105.14.53
    357 130.237.218.86
    273 75.97.9.59
    113 50.16.19.13
    102 209.85.238.199
     99 68.180.224.225
     84 100.43.83.137
     83 208.115.111.72
     82 198.46.149.143
     74 208.115.113.88
     65 108.171.116.194
     60 65.55.213.73
     60 208.91.156.11
     56 66.249.73.185
     52 50.139.66.106
     50 86.76.247.183
     50 14.160.65.22
     43 93.17.51.134
     42 208.43.252.200
     41 199.168.96.66
     41 183.179.22.186
     41 144.76.194.187
     40 210.13.83.18
     40 209.17.114.78
     39 59.163.27.11
```



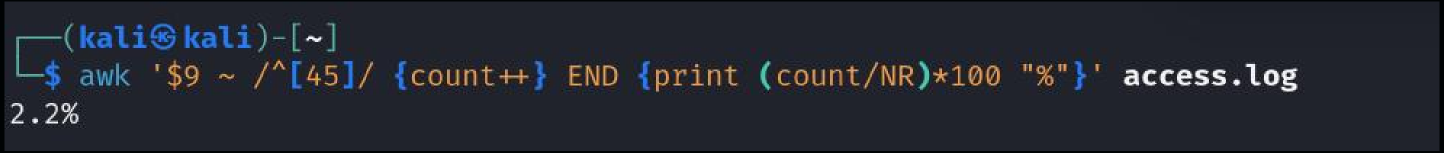
```
     27 144.76.95.39
     27 134.158.231.20
     26 99.252.100.83
     26 83.61.80.53
     26 222.14.252.108
     25 217.12.185.5
     25 216.152.249.242
     24 94.93.82.148
     23 88.103.19.195
     23 83.149.9.216
     23 217.195.202.13
     23 176.92.75.62
     23 150.162.56.185
     23 108.174.55.234
     22 70.83.251.183
     22 178.255.215.83
     21 207.241.237.223
     20 185.4.253.67
     19 91.221.131.30
     19 81.190.174.219
     19 208.93.0.48
     18 89.2.87.1
     18 83.42.229.238
     18 79.84.40.134
     18 72.223.76.198
     18 207.241.237.220
     18 201.26.152.202
```

6. `awk '$9 ~ /^4[0-9][0-9]$/ || $9 ~ /^5[0-9][0-9]$/' access.log | wc -l`
→ Count how many requests failed (status codes 4xx or 5xx)



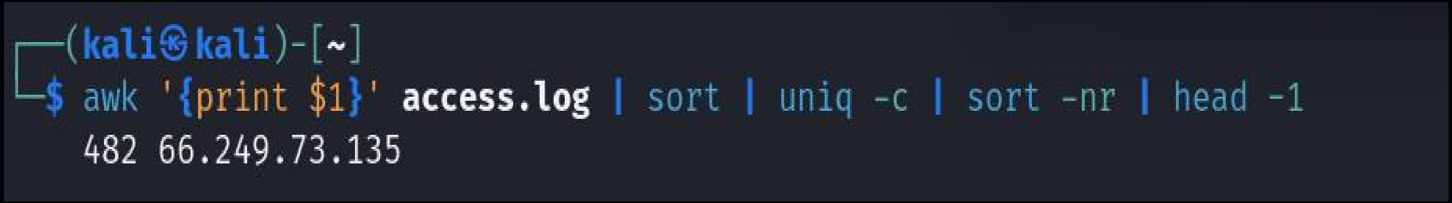
```
┌──(kali㉿kali)-[~]
└─$ awk '$9 ~ /^4[0-9][0-9]$/ || $9 ~ /^5[0-9][0-9]$/' access.log | wc -l
220
```

7. awk '$9 ~ /^[45]/ {count++} END {print (count/NR)*100 "%"}' access.log
→ Calculate the percentage of failed requests



```
┌──(kali㉿kali)-[~]
└─$ awk '$9 ~ /^[45]/ {count++} END {print (count/NR)*100 "%"}' access.log
2.2%
```

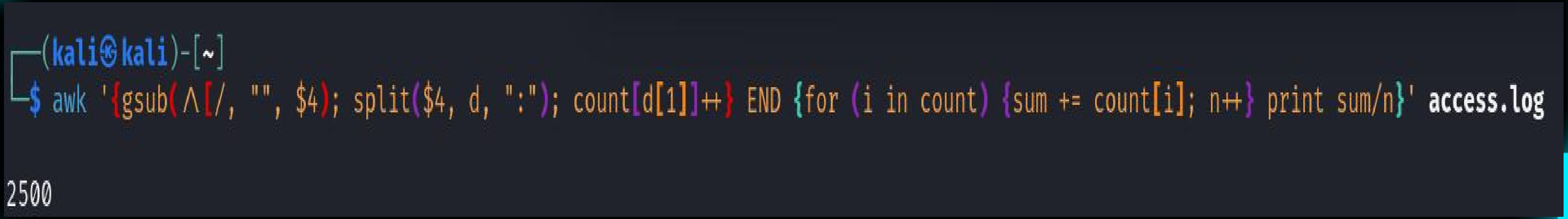
8. awk '{print $1}' access.log | sort | uniq -c | sort -nr | head -1
→ Find the most active IP address (Top User)



```
┌──(kali㉿kali)-[~]
└─$ awk '{print $1}' access.log | sort | uniq -c | sort -nr | head -1
    482 66.249.73.135
```

9. awk '{gsub(/\[/, "", $4); split($4, d, ":"); count[d[1]]++} END {for (i in count) {sum += count[i]; n++} print sum/n}' access.log

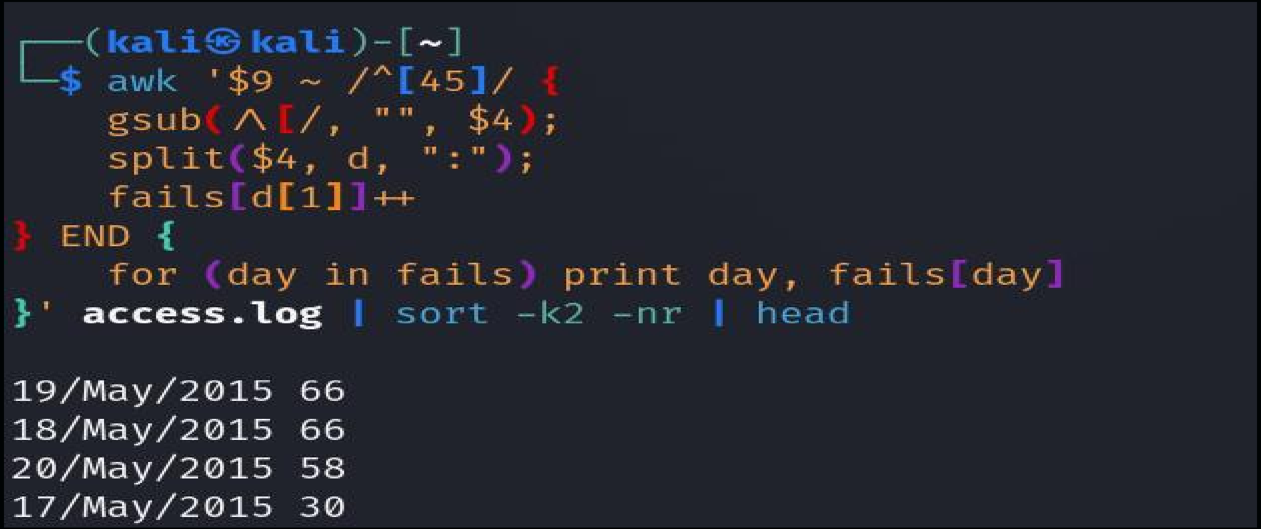
→ Calculate average number of requests per day



```
┌──(kali㉿kali)-[~]
└─$ awk '{gsub(/\[/, "", $4); split($4, d, ":"); count[d[1]]++} END {for (i in count) {sum += count[i]; n++} print sum/n}' access.log

2500
```

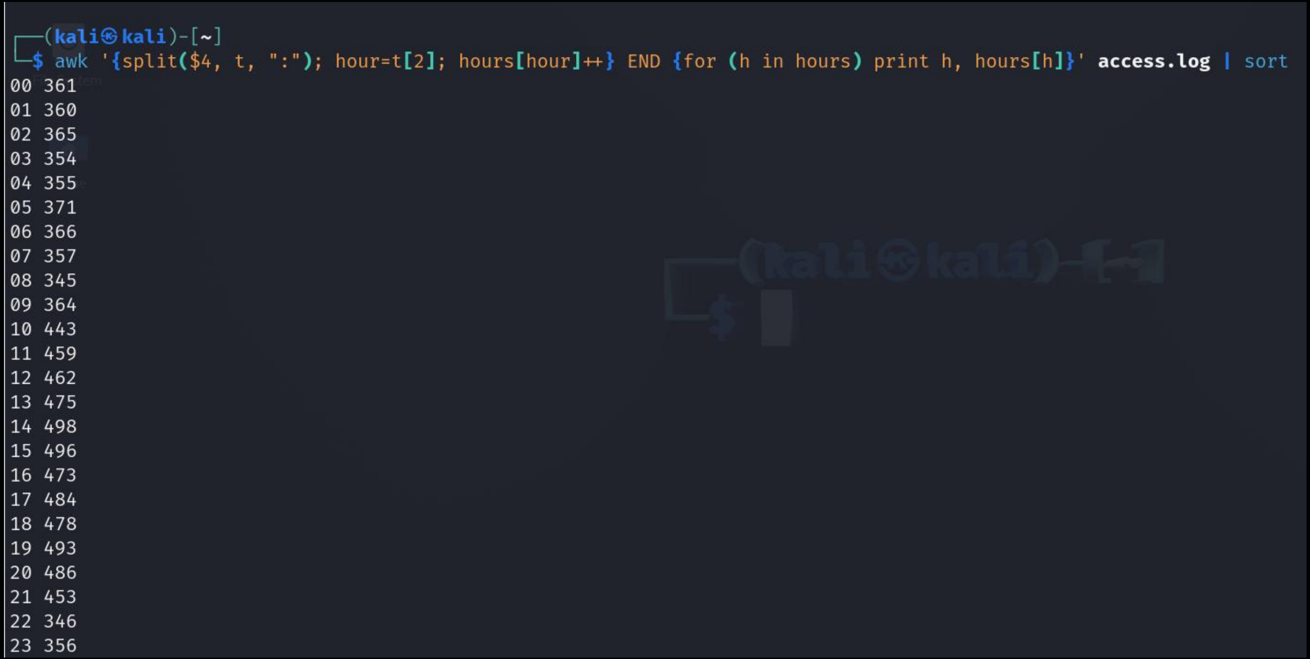
10. awk '$9 ~ /^[45]/ {gsub(/\[/, "", $4);
split($4, d, ":");
 fails[d[1]]++} END
{for (day in fails) print day, fails[day] }' access.log | sort -k2 -nr | head

→ Identify which days had the highest number of failure requests



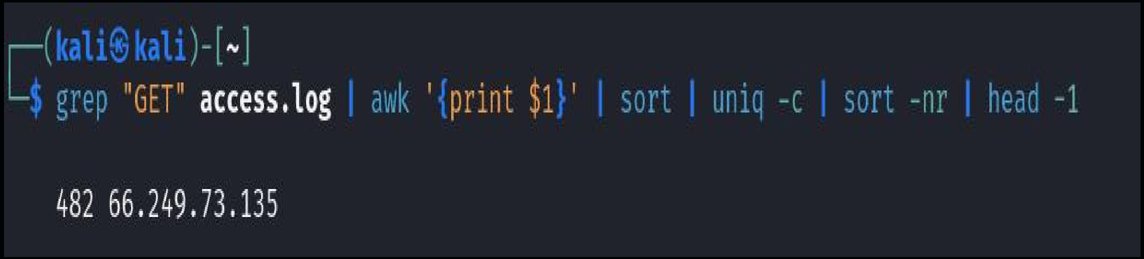
```
┌──(kali㉿kali)-[~]
└─$ awk '$9 ~ /^[45]/ {
    gsub(/\[/, "", $4);
    split($4, d, ":");
    fails[d[1]]++
} END {
    for (day in fails) print day, fails[day]
}' access.log | sort -k2 -nr | head

19/May/2015 66
18/May/2015 66
20/May/2015 58
17/May/2015 30
```

11. awk '{split($4, t, ":"); hour=t[2]; hours[hour]++} END {for (h in hours) print h, hours[h]}' access.log | sort
→ Calculate number of requests made each hour of the day



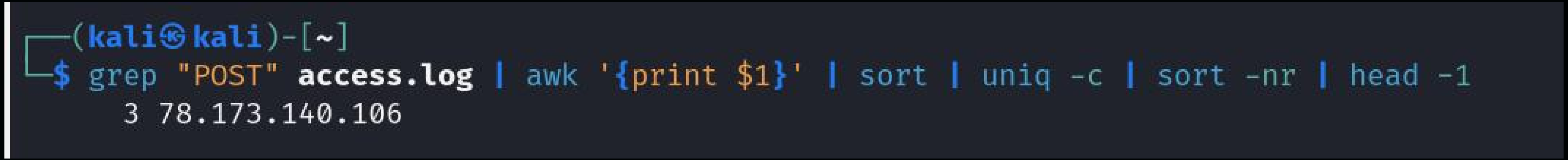
```
┌──(kali㉿kali)-[~]
└─$ awk '{split($4, t, ":"); hour=t[2]; hours[hour]++} END {for (h in hours) print h, hours[h]}' access.log | sort
00 361
01 360
02 365
03 354
04 355
05 371
06 366
07 357
08 345
09 364
10 443
11 459
12 462
13 475
14 498
15 496
16 473
17 484
18 478
19 493
20 486
21 453
22 346
23 356
```

12. grep "GET" access.log | awk '{print $1}' | sort | uniq -c | sort -nr | head -1
→ Find IP that used GET the most



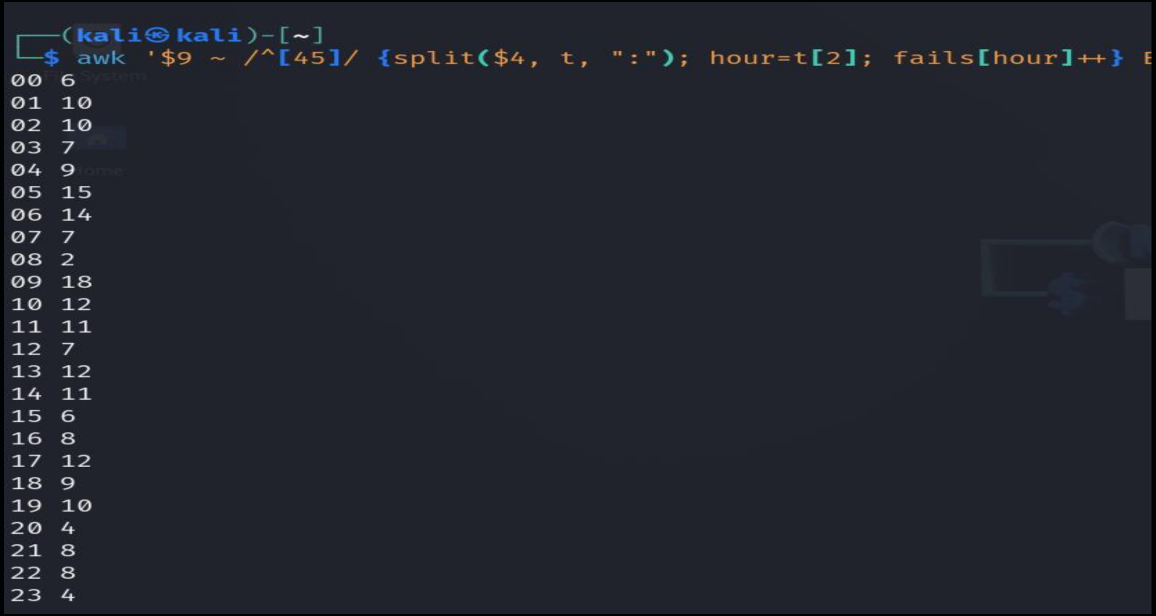
```
┌──(kali㉿kali)-[~]
└─$ grep "GET" access.log | awk '{print $1}' | sort | uniq -c | sort -nr | head -1

    482 66.249.73.135
```

13. grep "POST" access.log | awk '{print $1}' | sort | uniq -c | sort -nr | head -1
→ Find IP that used POST the most



```
┌──(kali㉿kali)-[~]
└─$ grep "POST" access.log | awk '{print $1}' | sort | uniq -c | sort -nr | head -1
      3 78.173.140.106
```

14. awk '$9 ~ /^[45]/ {split($4, t, ":"); hour=t[2]; fails[hour]++} END {for (h in fails) print h, fails[h]}' access.log | sort
→ Identify if failure requests occur more during specific hours



```
┌──(kali㉿kali)-[~]
└─$ awk '$9 ~ /^[45]/ {split($4, t, ":"); hour=t[2]; fails[hour]++} E
00 6
01 10
02 10
03 7
04 9
05 15
06 14
07 7
08 2
09 18
10 12
11 11
12 7
13 12
14 11
15 6
16 8
17 12
18 9
19 10
20 4
21 8
22 8
23 4
```