Project-week 4: Turn a New Leaf

Submitted by: Navneet kaur

WORKFLOW:

Working as an Access log Analyst at Turn a New Leaf, My role is to monitor the logs for unusual network traffic also to send an alert to manager for unusual number of failed logins on weekly basis and update them via email.

To create an efficient and low maintenance workflow for monitoring access logs and sending alerts for unusual failed logins, I will follow these steps:

- Identify log files on Linux as well on Windows
- Monitor the frequency using Cron jobs or scheduled tasks to run the script once a week every Friday to cover till every Thursday
- Parse the access logs using Python to get information of timestamps, Ip addresses and usernames
- Count the failed logins with the use grep filtering commands
- Generate weekly report with high number of failed logins and relevant information.

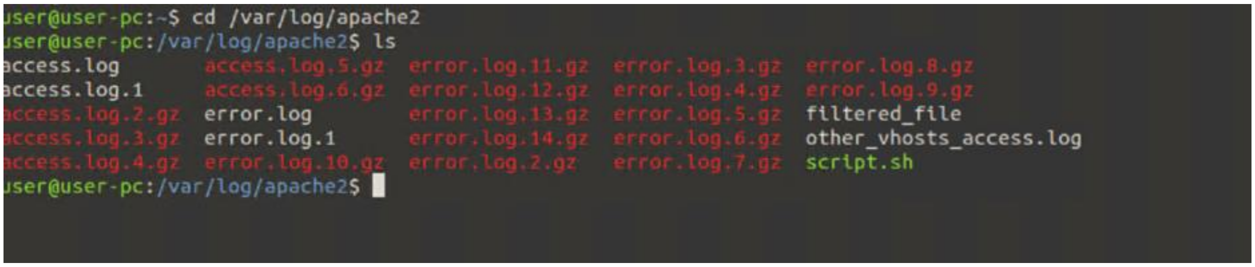
PROGRAMMING:

To complete all these above steps, I'd using:

- Python programming to write a script to parse the logs files, count failed logins
- RegEx to extract the information from logs such as IP address, Timestamps, and status
- Linux terminal to get particular number of lines and filtering according to need.

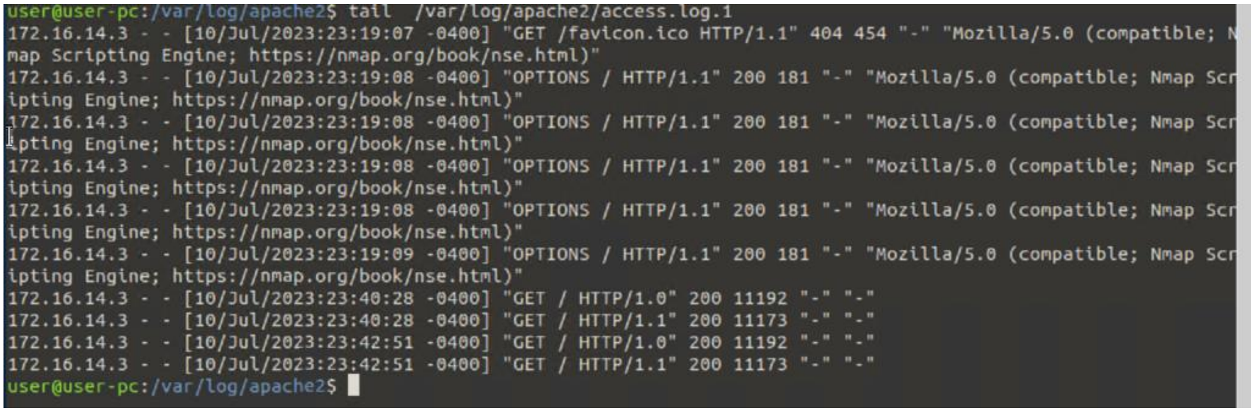
EXPECTED OUTPUT:

To monitor error logs from the Apache server, you can use the Linux tail command to view all the errors as they occur in real-time. My data is in access.log.1 file.



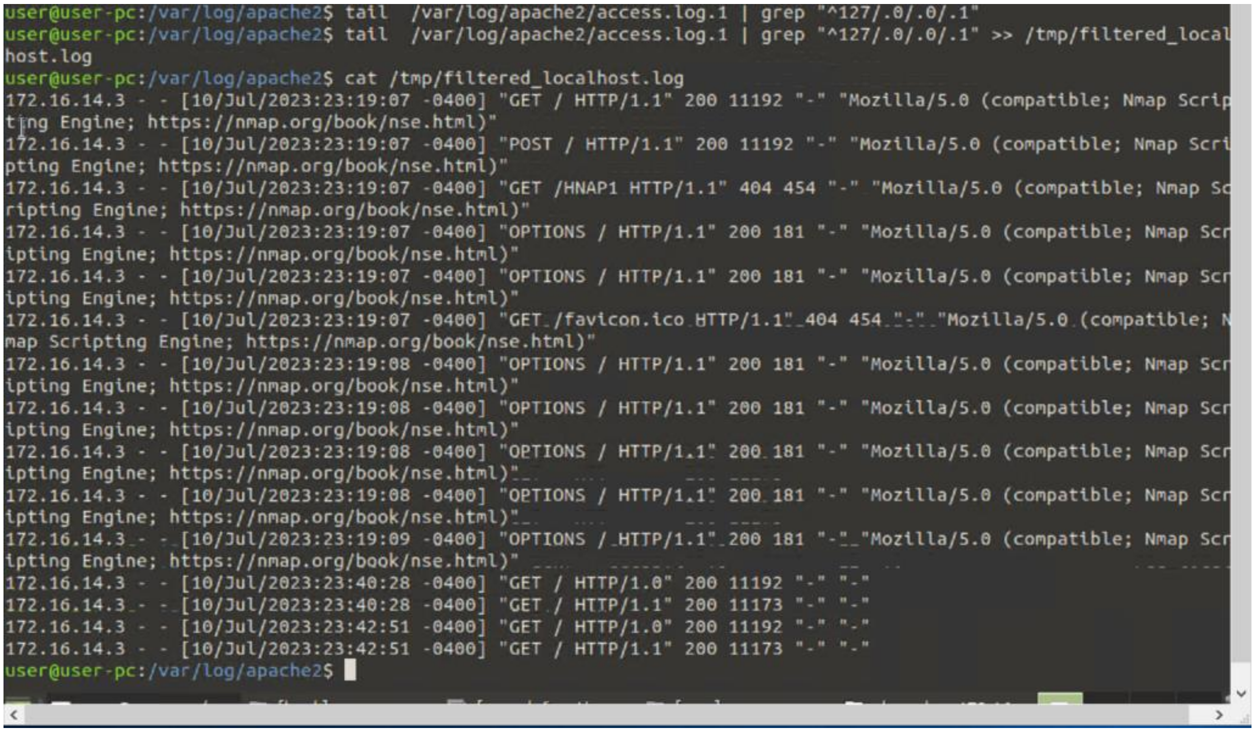
```
user@user-pc:~$ cd /var/log/apache2
user@user-pc:/var/log/apache2$ ls
access.log       access.log.5.gz  error.log.11.gz  error.log.3.gz  error.log.8.gz
access.log.1     access.log.6.gz  error.log.12.gz  error.log.4.gz  error.log.9.gz
access.log.2.gz  error.log        error.log.13.gz  error.log.5.gz  filtered_file
access.log.3.gz  error.log.1      error.log.14.gz  error.log.6.gz  other_vhosts_access.log
access.log.4.gz  error.log.10.gz  error.log.2.gz   error.log.7.gz  script.sh
user@user-pc:/var/log/apache2$ 
```

Using Tail command to extract the last log entries



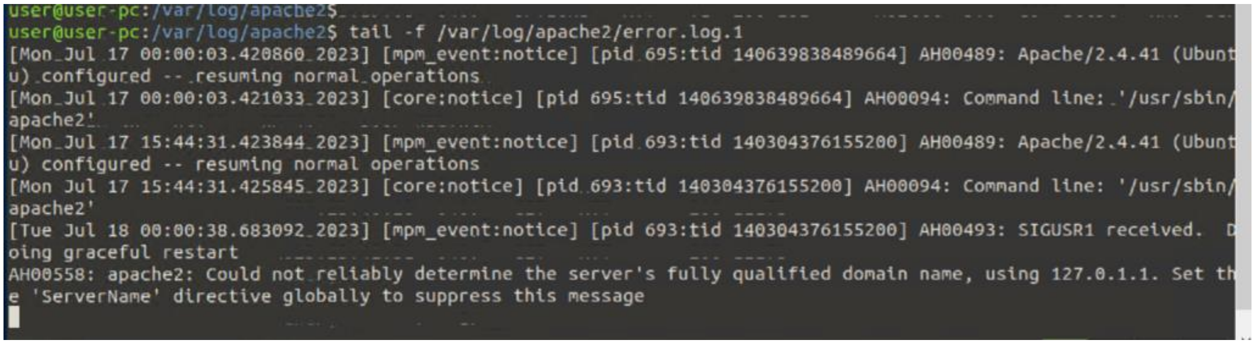
```
user@user-pc:/var/log/apache2$ tail  /var/log/apache2/access.log.1
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "GET /favicon.ico HTTP/1.1" 404 454 "-" "Mozilla/5.0 (compatible; N
map Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:09 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:40:28 -0400] "GET / HTTP/1.0" 200 11192 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:40:28 -0400] "GET / HTTP/1.1" 200 11173 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:42:51 -0400] "GET / HTTP/1.0" 200 11192 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:42:51 -0400] "GET / HTTP/1.1" 200 11173 "-" "-"
user@user-pc:/var/log/apache2$
```

Grep command to Filter the IP address e.g. 127.0.0.1



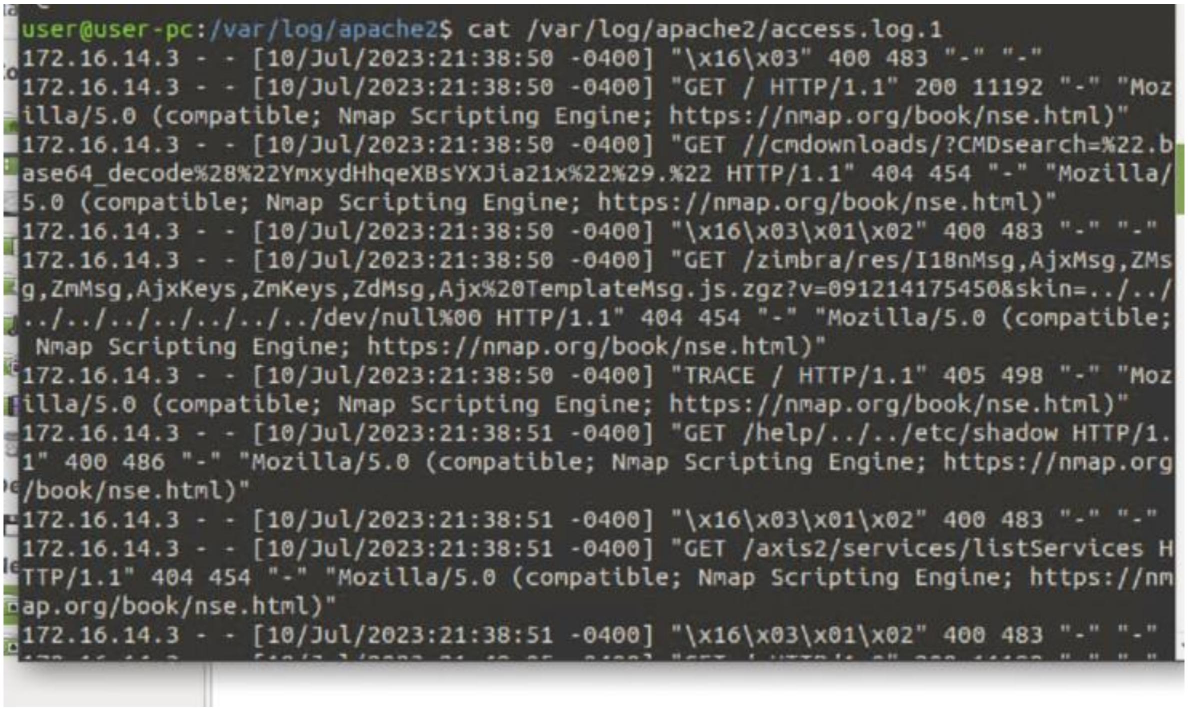
```
user@user-pc:/var/log/apache2$ tail  /var/log/apache2/access.log.1 | grep "^127/.0/.0/.1"
user@user-pc:/var/log/apache2$ tail  /var/log/apache2/access.log.1 | grep "^127/.0/.0/.1" >> /tmp/filtered_local
host.log
user@user-pc:/var/log/apache2$ cat /tmp/filtered_localhost.log
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "GET / HTTP/1.1" 200 11192 "-" "Mozilla/5.0 (compatible; Nmap Scrip
ting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "POST / HTTP/1.1" 200 11192 "-" "Mozilla/5.0 (compatible; Nmap Scri
pting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "GET /HNAP1 HTTP/1.1" 404 454 "-" "Mozilla/5.0 (compatible; Nmap Sc
ripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:07 -0400] "GET /favicon.ico HTTP/1.1" 404 454 "-" "Mozilla/5.0 (compatible; N
map Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:08 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:19:09 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "Mozilla/5.0 (compatible; Nmap Scr
ipting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:23:40:28 -0400] "GET / HTTP/1.0" 200 11192 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:40:28 -0400] "GET / HTTP/1.1" 200 11173 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:42:51 -0400] "GET / HTTP/1.0" 200 11192 "-" "-"
172.16.14.3 - - [10/Jul/2023:23:42:51 -0400] "GET / HTTP/1.1" 200 11173 "-" "-"
user@user-pc:/var/log/apache2$
```

To extraxt error log file



```
user@user-pc:/var/log/apache2$
user@user-pc:/var/log/apache2$ tail -f /var/log/apache2/error.log.1
[Mon Jul 17 00:00:03.420860 2023] [mpm_event:notice] [pid 695:tid 140639838489664] AH00489: Apache/2.4.41 (Ubunt
u) configured -- resuming normal operations
[Mon Jul 17 00:00:03.421033 2023] [core:notice] [pid 695:tid 140639838489664] AH00094: Command line: '/usr/sbin/
apache2'
[Mon Jul 17 15:44:31.423844 2023] [mpm_event:notice] [pid 693:tid 140304376155200] AH00489: Apache/2.4.41 (Ubunt
u) configured -- resuming normal operations
[Mon Jul 17 15:44:31.425845 2023] [core:notice] [pid 693:tid 140304376155200] AH00094: Command line: '/usr/sbin/
apache2'
[Tue Jul 18 00:00:38.683092 2023] [mpm_event:notice] [pid 693:tid 140304376155200] AH00493: SIGUSR1 received.  D
oing graceful restart
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set th
e 'ServerName' directive globally to suppress this message
```

Cat used to see the content of access.log.1 file to check the logins and failed logins



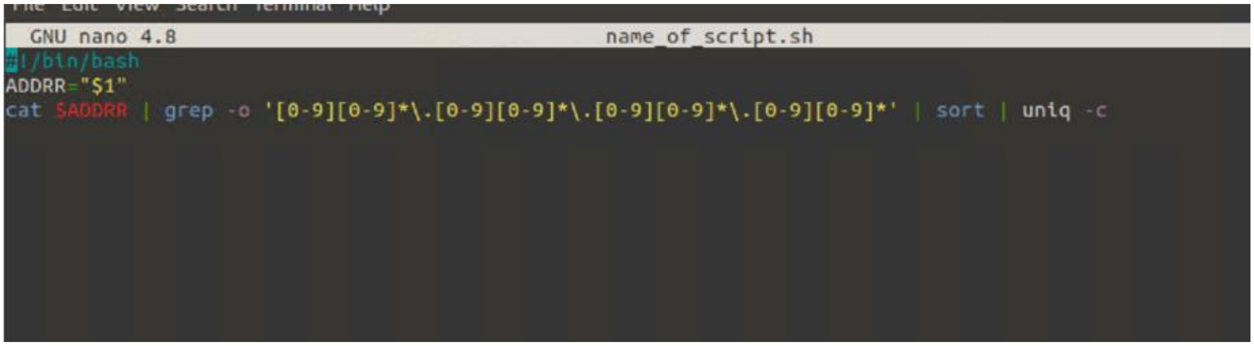
```
user@user-pc:/var/log/apache2$ cat /var/log/apache2/access.log.1
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "\x16\x03" 400 483 "-" "-"
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "GET / HTTP/1.1" 200 11192 "-" "Moz
illa/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "GET //cmdownloads/?CMDsearch=%22.b
ase64_decode%28%22YmxydHhqeXBsYXJia21x%22%29.%22 HTTP/1.1" 404 454 "-" "Mozilla/
5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "\x16\x03\x01\x02" 400 483 "-" "-"
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "GET /zimbra/res/I18nMsg,AjxMsg,ZMs
g,ZmMsg,AjxKeys,ZmKeys,ZdMsg,Ajx%20TemplateMsg.js.zgz?v=091214175450&skin=../../
../../../../../../../dev/null%00 HTTP/1.1" 404 454 "-" "Mozilla/5.0 (compatible;
 Nmap Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:50 -0400] "TRACE / HTTP/1.1" 405 498 "-" "Moz
illa/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:51 -0400] "GET /help/../../etc/shadow HTTP/1.
1" 400 486 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org
/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:51 -0400] "\x16\x03\x01\x02" 400 483 "-" "-"
172.16.14.3 - - [10/Jul/2023:21:38:51 -0400] "GET /axis2/services/listServices H
TTP/1.1" 404 454 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nm
ap.org/book/nse.html)"
172.16.14.3 - - [10/Jul/2023:21:38:51 -0400] "\x16\x03\x01\x02" 400 483 "-" "-"
```

Cron is used to schedule the monitoring every Thursday

CRON jobs: 0 0 * * 4 * /bin/sh script.sh //

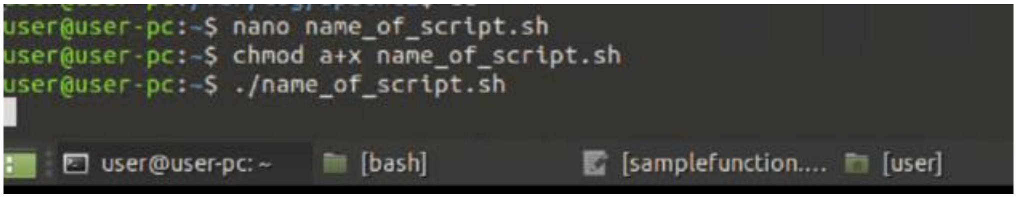
RegEx is used to filter logs and to sort it out by using RegEx. I have a done by using bash.

I have created name_of_script.sh file and wrote followed code:

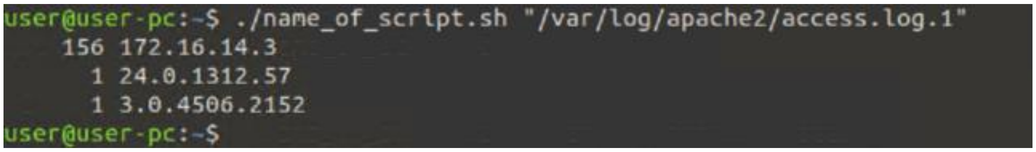


```
  GNU nano 4.8                          name_of_script.sh
#!/bin/bash
ADDRR="$1"
cat $ADDRR | grep -o '[0-9][0-9]*\.[0-9][0-9]*\.[0-9][0-9]*\.[0-9][0-9]*' | sort | uniq -c
```

Giving permission using chmod a+x name_of_ script.sh and to run it ./name_of_script.sh

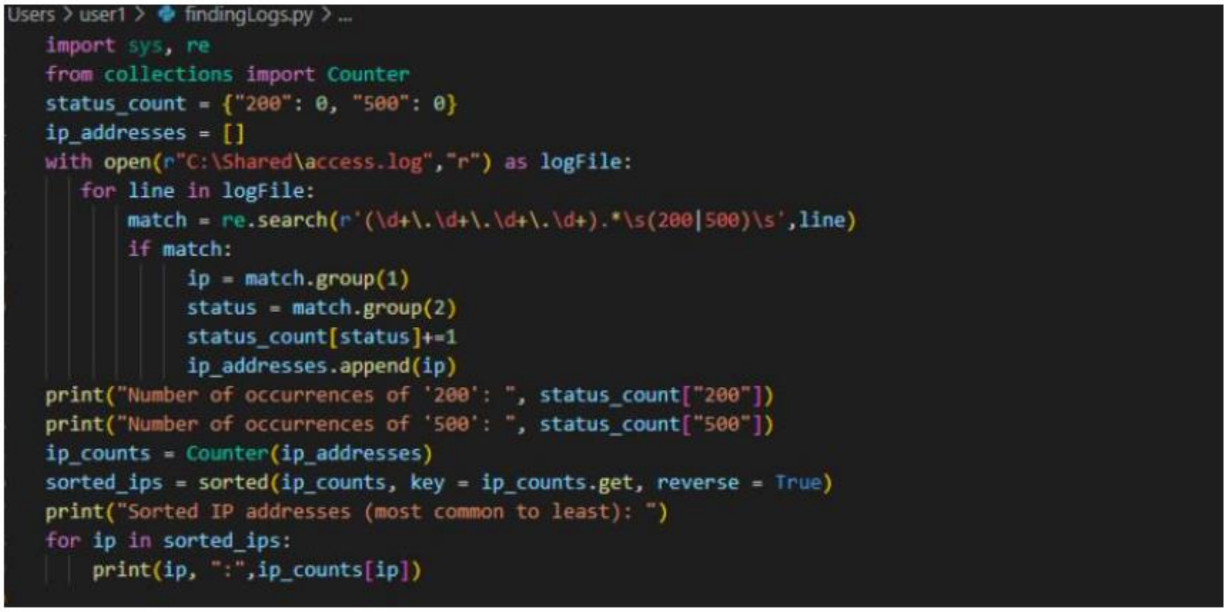


```
user@user-pc:~$ nano name_of_script.sh
user@user-pc:~$ chmod a+x name_of_script.sh
user@user-pc:~$ ./name_of_script.sh
```

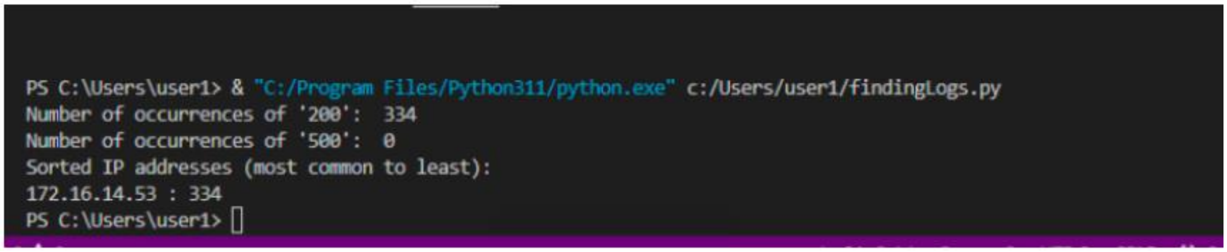


```
user@user-pc:~$ ./name_of_script.sh "/var/log/apache2/access.log.1"
    156 172.16.14.3
      1 24.0.1312.57
      1 3.0.4506.2152
user@user-pc:~$
```

Python: I have used python to filter logs also. The following code is filtering the occurrences of HTTP status and also sorting IP addresses.

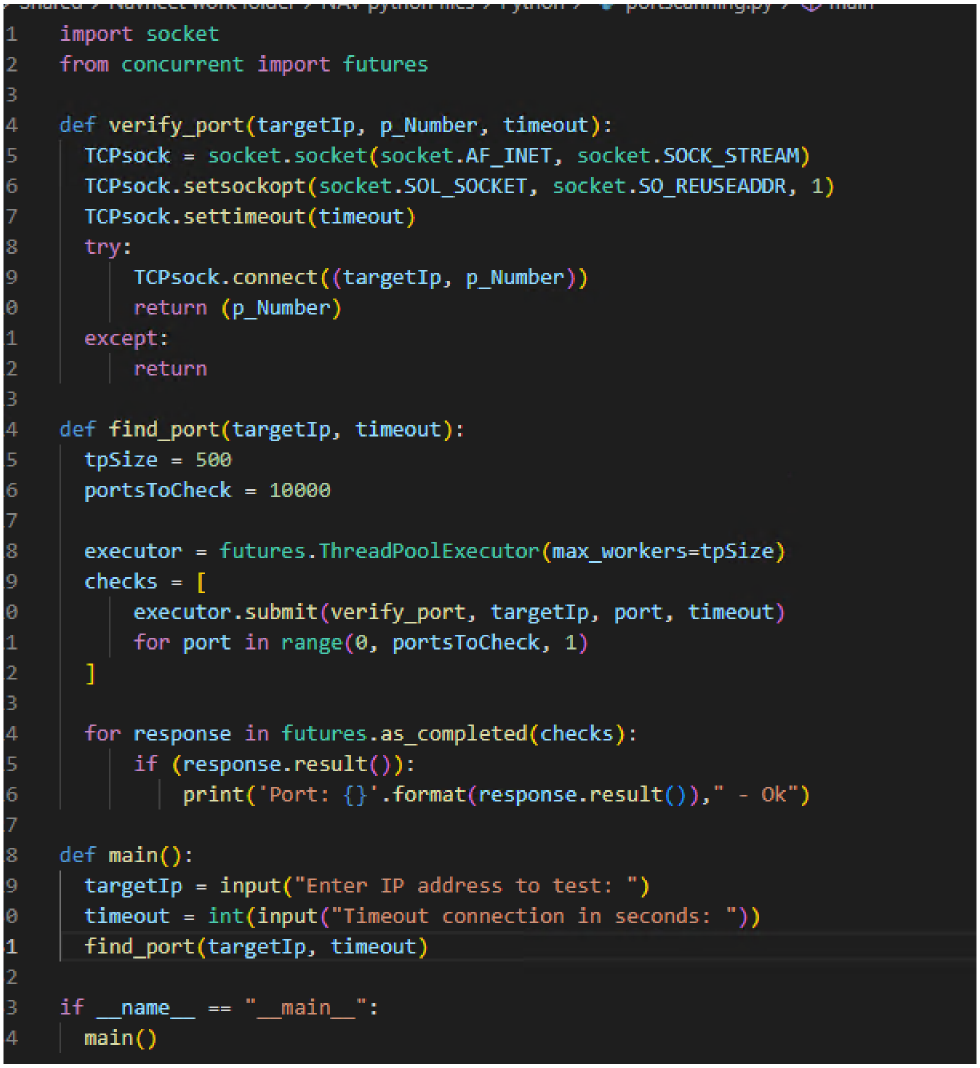


```python
import sys, re
from collections import Counter
status_count = {"200": 0, "500": 0}
ip_addresses = []
with open(r"C:\Shared\access.log","r") as logFile:
    for line in logFile:
        match = re.search(r'(\d+\.\d+\.\d+\.\d+).*\s(200|500)\s',line)
        if match:
            ip = match.group(1)
            status = match.group(2)
            status_count[status]+=1
            ip_addresses.append(ip)
print("Number of occurrences of '200': ", status_count["200"])
print("Number of occurrences of '500': ", status_count["500"])
ip_counts = Counter(ip_addresses)
sorted_ips = sorted(ip_counts, key = ip_counts.get, reverse = True)
print("Sorted IP addresses (most common to least): ")
for ip in sorted_ips:
    print(ip, ":",ip_counts[ip])
```

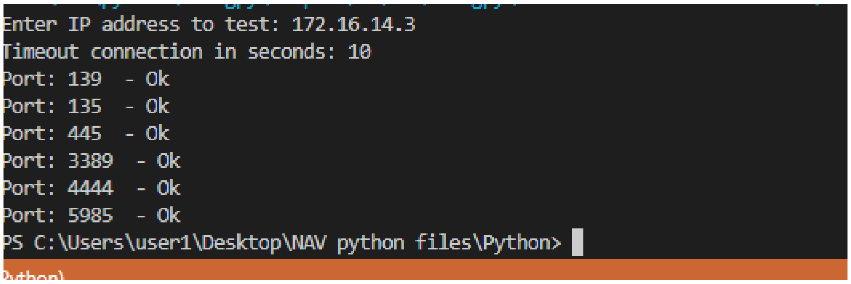


```
PS C:\Users\user1> & "C:/Program Files/Python311/python.exe" c:/Users/user1/findingLogs.py
Number of occurrences of '200':  334
Number of occurrences of '500':  0
Sorted IP addresses (most common to least):
172.16.14.53 : 334
PS C:\Users\user1> 
```

Port Scanning: To find vulnerabilities, I have done port scan using a python code. The code is as follows:



```python
import socket
from concurrent import futures

def verify_port(targetIp, p_Number, timeout):
    TCPsock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    TCPsock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    TCPsock.settimeout(timeout)
    try:
        TCPsock.connect((targetIp, p_Number))
        return (p_Number)
    except:
        return

def find_port(targetIp, timeout):
    tpSize = 500
    portsToCheck = 10000

    executor = futures.ThreadPoolExecutor(max_workers=tpSize)
    checks = [
        executor.submit(verify_port, targetIp, port, timeout)
        for port in range(0, portsToCheck, 1)
    ]

    for response in futures.as_completed(checks):
        if (response.result()):
            print('Port: {}'.format(response.result())," - Ok")

def main():
    targetIp = input("Enter IP address to test: ")
    timeout = int(input("Timeout connection in seconds: "))
    find_port(targetIp, timeout)

if __name__ == "__main__":
    main()
```



```
Enter IP address to test: 172.16.14.3
Timeout connection in seconds: 10
Port: 139  - Ok
Port: 135  - Ok
Port: 445  - Ok
Port: 3389  - Ok
Port: 4444  - Ok
Port: 5985  - Ok
PS C:\Users\user1\Desktop\NAV python files\Python>
```

UNUSUAL BEHAVIOUR:

Number of failed login attempts for any user exceeds the threshold, unrecognized IP addresses, changing passwords; it should be flagged as unusual behaviour and send alert to manager

POTENTIAL ITERATION:

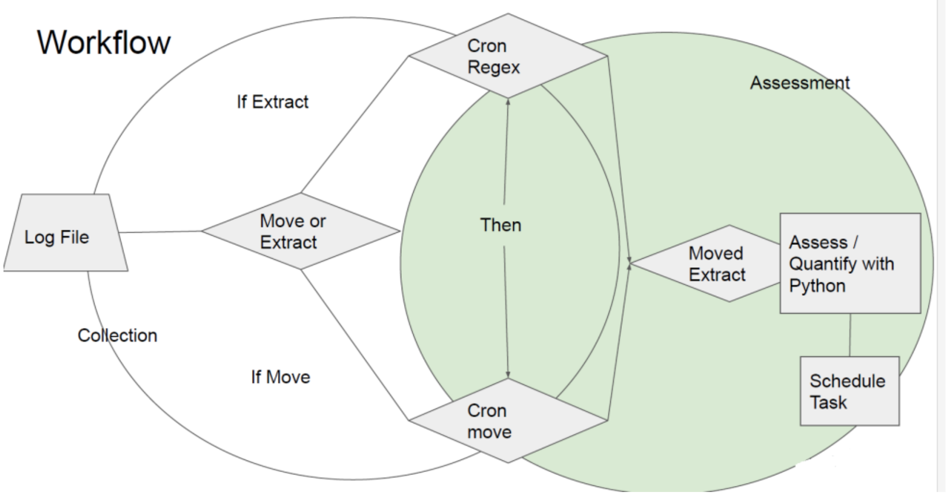
- Continuously monitoring of log files instead of once a week, schedule Task using cron jobs.

- Enhance the reporting using graphs, charts to provide clear view
- Use of network monitoring tool like PRTG to monitor the traffic and locate reconnaissance attacks
- Automate the alerts using sensors

Citations:

https://www.pythonforbeginners.com/code-snippets-source-code/port-scanner-in-python#htoc-writing-a-program-using-python-sockets
https://cyber.compass.lighthouselabs.ca/p/2/days/w03d5/activities/2868
https://devhints.io/bash
https://crontab-generator.org/
https://www.pythonforbeginners.com/code-snippets-source-code/port-scanner-in-python
https://linuxhint.com/var-log-messages/

Flow chart:


Workflow
If Extract
Cron Regex
Assessment
Log File
Move or Extract
Then
Moved Extract
Assess / Quantify with Python
Collection
If Move
Cron move
Schedule Task