Log Analysis

Given a log find what ip sent the most requests:
$ awk '{print $1}' access.log | sort | uniq -c | sort -nr | head -n 1

FInd what ip sent exactly 41 requests:
$ awk '{ count[$1]++ } END { for (ip in count) { if (count[ip] == 41) { print "IP address:", ip } } }' access.log

Find # of POST requests made to a server:
$ awk '$6 == "\"POST" {count++} END {print count}' access.log

Find the day the server had the most traffic:

```
from collections import defaultdict
from datetime import datetime
# Open the log file
log_file_path = 'access.log'  # Replace with the actual path to your log file
# Dictionary to store traffic counts for each day
traffic_by_day = defaultdict(int)
with open(log_file_path, 'r') as file:
    # Iterate through each line in the file
    for line in file:
        # Extract the timestamp from the log entry
        timestamp_str = line.split()[3]  # Assuming timestamp is at index 3, adjust if necessary
        # Parse the timestamp into a datetime object
        timestamp = datetime.strptime(timestamp_str, '[%d/%b/%Y:%H:%M:%S')
        # Extract the day from the timestamp
        day = timestamp.strftime('%Y-%m-%d')
        # Increment the traffic count for the corresponding day
        traffic_by_day[day] += 1
# Find the day with the maximum traffic
max_traffic_day = max(traffic_by_day, key=traffic_by_day.get)
max_traffic_count = traffic_by_day[max_traffic_day]
print("Day with the most traffic:", max_traffic_day)
print("Traffic count on that day:", max_traffic_count)
```

How many requests were sent on the day with the most traffic:

```
from datetime import datetime
# Open the log file
log_file_path = 'access.log'  # Replace with the actual path to your log file
requests_count = 0
# Date to filter
target_date = '2024-03-27'  # Change to the desired date
with open(log_file_path, 'r') as file:
    # Iterate through each line in the file
    for line in file:
        # Extract the timestamp from the log entry
        timestamp_str = line.split()[3]  # Assuming timestamp is at index 3, adjust if necessary
        # Parse the timestamp into a datetime object
        timestamp = datetime.strptime(timestamp_str, '[%d/%b/%Y:%H:%M:%S')
        # Extract the date from the timestamp
```

```
        date = timestamp.strftime('%Y-%m-%d')
        # Check if the date matches the target date
        if date == target_date:
            requests_count += 1
print("Number of requests on", target_date + ":", requests_count)
```

Enumeration and Exploitation

What is the flag?

```
ENC = [95, 91, 68, 33, 67, 83, 73, 91, 48, 56, 39, 43, 58]
def main():
    dec = ''
    rotate = [12, 16, 29]
    count = 0
    for byt in ENC:
        dec += chr(byt ^ rotate[count])
        count = (count + 1) % 2
    print(dec)

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

ENC list contains encrypted bytes.
Rotate list contains the rotation values used for XOR operation.
Script iterates through each byte in ENC, performs an XOR operation with the corresponding value in rotate, and appends the result to dec.
The count variable is used to cycle through the rotation values.
%2 never allows rotate[2] to be used so change to %3 to solve

```
ENC = [95, 91, 68, 33, 67, 83, 73, 91, 48, 56, 39, 43, 58]
def main():
    dec = ''
    rotate = [12, 16, 29]
    count = 0
    for byt in ENC:
        dec += chr(byt ^ rotate[count])
        count = (count + 1) % 3
    print("Decrypted Flag:", dec)
if __name__ == '__main__':
    main()
```

Given a DLL file, what runtime does it use?
When loaded into dotPeek, we see the second file on the left is mscorlib.dll. Look it up, see it is part of the .NET runtime environment. We can also see in the header of one of the decompiled files that the location is in ….\Microsoft.NET\Framework64\v4.0.30319\mscorlib.dll.
So the answer is “.NET” runtime.

What is the flag?

Password Cracking
What is the md5 hash of this password `thelastofus2202`
https://www.md5hashgenerator.com/

What is the sha1 hash of this password `1476succession`
https://www.md5hashgenerator.com/

What is the sha256 hash of this password `64gameofthrones71`
https://tools.keycdn.com/sha256-online-generator

Crack the passwords:
Elyse: 7f9d62c839dfec068f18bd0e255a3a0a
$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
$ hashcat -m 0 hash1.txt /usr/share/wordlists/rockyou.txt
Pass: Ryan0797

Zoe: 229e7e733f89213922919b6729ff8593
$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
$ hashcat -m 0 hash2.txt /usr/share/wordlists/rockyou.txt
Pass: TAYLOR15

Joe: 174a27e8802b3f37274a696c65075a32
$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
$ hashcat -m 0 hash3.txt /usr/share/wordlists/rockyou.txt
Pass: sam1080

Forensics
Given fs.img
What type of filesystem
$ fsstat fs.img
What is the flag
Was in one of the deleted jpg files

Web
How many seconds in between each update interval
Go on site > Inspect > Go to sources > In the code see the following line of code:
updateInterval = setInterval(update, 5000);
So answer is 5 seconds.