**Log File Analyzer for Intrusion Detection**

**Project Title : Log File Analyzer for Intrusion Detection using Python.**

**Done By : Vaishnavi Nagarkar**

**Tools Used : Python, Regex, Pandas, Matplotlib, VS Code.**

# 1. Introduction

In today's rapidly evolving cyber threat landscape, it is essential to have efficient Security Operations Center (SOC) mechanisms in place. This project focuses on building a Defensive SOC tool using Python to analyze server log files and detect suspicious activities. The tool simulates a mini-SOC environment by leveraging Python’s data processing capabilities to analyze logs such as SSH and Apache access logs.

# 2. Objectives

- To simulate a basic SOC environment using Python scripting.  
- To detect brute-force attacks, failed login attempts, and other suspicious activities in log files.  
- To visualize data patterns and trends from logs using graphs.  
- To gain hands-on experience with Python for cybersecurity purposes.

# 3. Tools & Technologies

- Python  
- Regex  
- Pandas  
- Matplotlib  
- VS Code

# 4. Implementation Overview

- Step 1: Loaded and parsed raw log files using regular expressions.  
 - Used logs: auth.log (for SSH), access.log (for Apache).  
- Step 2: Extracted relevant fields such as IP addresses, timestamps, request types, and status codes.  
- Step 3: Counted login attempts, failed logins, and unusual access patterns.  
- Step 4: Identified anomalies such as:  
 - Multiple failed login attempts from the same IP (brute-force).  
 - Suspicious user-agent strings or HTTP status codes.  
- Step 5: Generated visualizations (bar graphs, line charts) to represent:  
 - Number of failed logins by IP.  
 - Traffic by time of day.  
 - Top accessing IPs.

# 5. Sample Code Snippet

import re  
from collections import Counter  
  
with open("auth.log") as file:  
 log\_data = file.read()  
  
failed\_attempts = re.findall(r"Failed password for .\* from (\d+\.\d+\.\d+\.\d+)", log\_data)  
ip\_counter = Counter(failed\_attempts)  
  
for ip, count in ip\_counter.items():  
 if count > 5:  
 print(f"Suspicious IP: {ip} with {count} failed login attempts")

# 6. Results

- Successfully identified top suspicious IP addresses based on failed SSH logins.  
- Visualized attack trends over time to detect peak attack periods.  
- Discovered misconfigurations and abuse attempts in web server logs.

Sample Graphs:  
- Top 5 IPs with Failed Logins (Bar Graph)  
- Login Attempts by Hour (Line Graph)

# 7. Conclusion

This project demonstrated how Python can be effectively used for log analysis and basic threat detection. It provided insights into defensive cybersecurity techniques and laid the foundation for more advanced SOC tools. Future improvements can include:  
- Real-time alerting with emails or system logs.  
- Integration with SIEM platforms like Splunk or ELK.  
- Use of machine learning for anomaly detection.

# 8. Learnings & Future Scope

- Learned how to use Python libraries for cybersecurity purposes.  
- Understood the importance of log files in detecting security incidents.  
- Next steps include implementing this in real-time environments using Splunk, Wireshark, or Syslog tools.