AZERBAIJAN CYBERSECURITY CENTER

FINAL PROJECT

Log Analysis & CTI

TABLE OF CONTENTS

1. Objectives

The primary objective of this project is to develop a Python-based tool that automates the analysis of web server access logs to identify, enrich, and report on potential security threats. This project simulates a critical function within a Security Operations Center (SOC), where analysts must sift through vast amounts of log data to find malicious activity. By building this tool, you will gain practical experience in file parsing, data correlation, and leveraging external Cyber Threat Intelligence (CTI) to assess risk. The integration of an AI model will challenge you to translate raw technical data into high-level, actionable intelligence, a crucial skill for modern cybersecurity professionals. This tool is intended to assist the cybersecurity team in Azerbaijan by providing a rapid, preliminary analysis of potential threats from log data.

1.1. Requirements

1.1.1. Minimum Requirements

- **Log Parsing:** Your script must parse the provided access.log file (a general web server access log format). From each line, you must accurately extract essential fields including the Source IP Address, Timestamp, HTTP Request Method (e.g., GET), and HTTP Status Code (e.g., 404). **Sample log:**


access_log.txt

- **Threat Intelligence Enrichment:** For each unique IP address found, your tool must perform a reputation check by gathering data from **at least one** of the public CTI sources listed below. Your goal is to extract general CTI information to determine if an IP is associated with malicious activity.

  - **AbuseIPDB**
    - **Example URL:** https://www.abuseipdb.com/check/1.1.1.1 (replace 1.1.1.1 with the target IP).
    - **Data to Extract:** Scrape the page to find the "Abuse Confidence Score," the "Total Reports," and the IP's "Country."

  - **Cisco Talos Intelligence**
    - **Example URL:** https://talosintelligence.com/reputation_center/lookup?search=8.8.8.8 (replace 8.8.8.8 with the target IP).
    - **Data to Extract:** Your script should extract the "Web Reputation" status (e.g., "Trustworthy," "Questionable," "Untrusted") and the "Owner" organization.

- **VirusTotal**
  - **API Documentation:** You can find instructions at https://developers.virustotal.com/reference/ip-object.
  - **Data to Extract:** Use the API to retrieve the analysis statistics (e.g., the number of security vendors that flagged the IP as malicious).

- **Statistical Analysis of Suspicious IPs:** After performing the CTI check, identify the IP addresses that are flagged as malicious or have a high-risk score. For these specific IPs, your script must perform a simple statistical analysis by calculating the total number of requests they made and counting how many of those resulted in a client-side error (4xx code).

- **Minimum AI Integration:** For one of the high-risk IP addresses you identified, perform a simple AI integration task. Provide the AI model with the technical information you gathered (e.g., "This IP has a high abuse score and is categorized under 'Web App Attack'"). Ask the AI to generate a single, plain-English sentence explaining what this type of threat means to a non-technical person.

- **Basic Error Handling:** Your script must be resilient to common errors. Implement try...except blocks to gracefully handle at least two of the following situations: The specified log file cannot be found (FileNotFoundError).
  - A network error occurs while trying to scrape a CTI website.
  - A log line is malformed and cannot be parsed correctly. Instead of crashing, your program should print a user-friendly error message and continue processing where possible.

- **Report Generation:** Your script must generate a report in a .txt file summarizing the findings for the suspicious IP addresses. The report must include the IP, its CTI findings, and its statistical data. The tool must be able to save the report to a file on the filesystem.

1.1.2. Bonus Requirements

- **Advanced AI Anomaly Detection:** Go beyond single-indicator analysis. Provide a statistical summary of the entire log file's activity (e.g., total requests, overall 404/200 ratio, number of unique IPs) to an AI model. Ask the AI to identify and describe anomalous patterns that could represent a coordinated attack, such as a low-and-slow scan or an unusual spike in server errors.

- **Advanced Reporting and System Interaction:** Save the report in a more sophisticated and readable format, such as Markdown (.md) or HTML. Your script should be executable from the command line, accepting the log file path as an argument (using sys.argv). The script must also save the generated report into a dedicated reports/

directory, creating the folder programmatically if it does not already exist (using the os module).

- **Threat Correlation with User-Agents:** Implement a function to analyze User-Agent strings. Identify known malicious or suspicious agents (e.g., "sqlmap," "Nmap," "Hydra"). Suppose a suspicious IP is also using a malicious User-Agent. In that case, this should be flagged as a high-priority indicator in the report, and the AI-generated "Analyst Note" should reflect this higher degree of confidence.

1.2. Deadline

The final project must be submitted by **September 12, 2025,** at **18:00**. No late submissions will be accepted unless extenuating circumstances apply and the instructor has granted prior permission.

1.3. Submission Method

Please submit your project files through the LMS. Ensure that all code, assets, and documentation files are included in a zipped folder. The folder should be named using the following convention:

`[Name].zip`

For example:

`JamesHickie.zip`

1.4. What to Include in your Submission

Your submission should include:

- All Python **`.py`** files that make up the project.
- **`requirements.txt`**: A file listing all the Python packages required for your project.
  - To generate this file, you can run **`pip freeze > requirements.txt`** in your project directory. Ensure you're in the correct virtual environment, if you are using one.
- Any additional assets, such as images or data files, if applicable.
- A README.md file containing:
  - Project name and description.
  - Dependencies
  - Instructions for running the project.
  - Any additional instructions or comments you'd like to add?