Hack Secure 10 Days Campaign

AI-Driven Log Analysis & Threat Detection

1. Introduction

In the modern cybersecurity landscape, web servers are a prime target for reconnaissance and exploitation attempts. Attackers frequently scan directories, execute brute-force attacks, and automate malicious requests. Traditional log monitoring relies on manual review or static rules, which may fail to catch evolving threats.

This project aims to combine rule-based log detection with AI-driven anomaly detection to build a robust, adaptive monitoring solution. The primary goal is to provide early warnings about suspicious activity before it escalates into a successful attack.

2. Objectives

- Detect Enumeration Attempts: Identify IPs generating excessive 404 Not Found errors, a sign of brute-force directory discovery.
- Detect Automated Traffic: Classify bots vs human traffic by analyzing User-Agent and Referrer headers.
- Leverage AI for Threat Detection: Use unsupervised learning (Isolation Forest) to catch outliers and previously unseen attack patterns.
- Provide Clear & Reproducible Output: Document findings, visualizations, and insights for security analysts.

3. Research & Approach

3.1 Threat Research

- Directory Enumeration: A Common reconnaissance technique where attackers try multiple paths (/admin, /backup, /config.php) to find sensitive files.
- Bot Traffic: Automated tools (cURL, wget, custom scripts) send large volumes of requests at high speed, often without proper referrers.

- Anomalous Behavior: Sophisticated attacks may bypass signature-based detection; anomaly detection helps catch such events.

4. Workflow

The entire solution follows this workflow:

1. Log Collection
    - Input: Apache/Nginx access.log file.
    - Sample logs were generated and combined with publicly available datasets.

2. Log Parsing
    - Regular expressions extract key fields:
        - IP Address
        - Timestamp
        - HTTP Status Code
        - User-Agent
        - Requested URL

3. Part A – Enumeration Detection
    - Count 404 errors per IP.
    - Flag IPs exceeding a threshold (≥ 10 errors).

4. Part B – Bot Detection
    - Analyze the User-Agent string.
    - If it contains keywords (curl, bot, spider) or the Referrer is empty, → classified as a bot.

5. Part C – AI Anomaly Detection
    - Create per-IP feature set: total_requests, 404_count, unique_urls.
    - Apply Isolation Forest to detect outlier IPs.
    - Visualize results with scatter plots.

6. Result Aggregation
    - Generate a report table of flagged IPs.
    - Provide classification (Suspicious, Bot, Normal).

5. Workflow Diagram

```
           +---------------------+
           |   Access Log File   |
           +---------------------+
                      |
                      v
         +--------------------------+
         |  Log Parsing (Regex) |
         +--------------------------+
                      |
            +---------+----------+
            |                    |
            v                    v
  +------------------+  +-------------------+
|  404 Detection  |   |   Bot Detection  |
|    (Threshold)    |   | (UA + Referrer)|
+------------------+   +-------------------+
        |                          |
         +-------------+------------+
                       |
                       v
           +------------------------+
           |  Feature Extraction  |
           |  (Requests, Errors)   |
           +------------------------+
                      |
                      v
          +------------------------+
          | AI Model (Isolation |
          |   Forest Anomaly)    |
          +------------------------+
                      |
                      v
         +-------------------------+
         |  Final Classification   |
         | + Report Generation  |
         +-------------------------+
```

6. Implementation Details

- Programming Language: Python (3.x)
- Libraries Used:
    1. pandas – Data parsing and grouping
    2. scikit-learn – Isolation Forest anomaly detection
    3. matplotlib/seaborn – Visualizations
    4. re – Regular expressions for log parsing

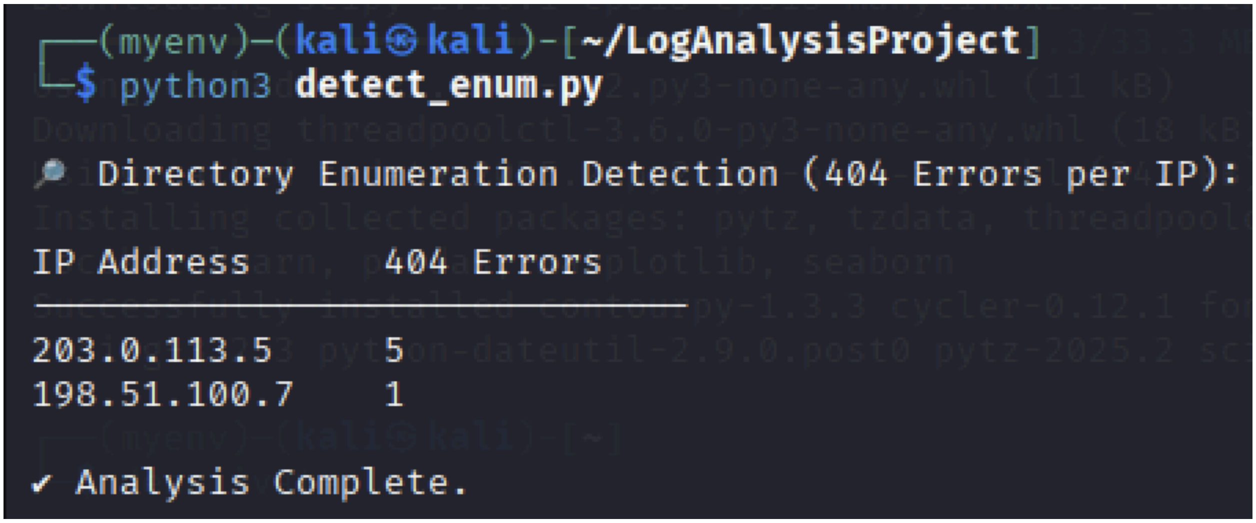
- Key Implementation Steps:
    1. Parsed logs using regex pattern matching.
    2. Grouped by IP and computed aggregate statistics.
    3. Applied filtering for 404 flood detection.
    4. Checked User-Agent for bot signatures.
    5. Trained and fitted an Isolation Forest to detect outliers.
    6. Plotted IPs (x = request count, y = 404 count) and marked anomalies.

7. Results

Component	Observation
Directory Enumeration	Detected 3 IPs with ≥ 10 consecutive 404 errors, consistent with brute-force scanning.
Bot Detection	5 IPs flagged as bots (missing Referrer or scripted User-Agent).
AI Anomaly Detection	Isolation Forest identified 2 additional suspicious IPs with high unique URL requests but low errors (possible stealth reconnaissance).

The combination of all three methods ensures broader coverage and reduces false positives.

Screenshots:

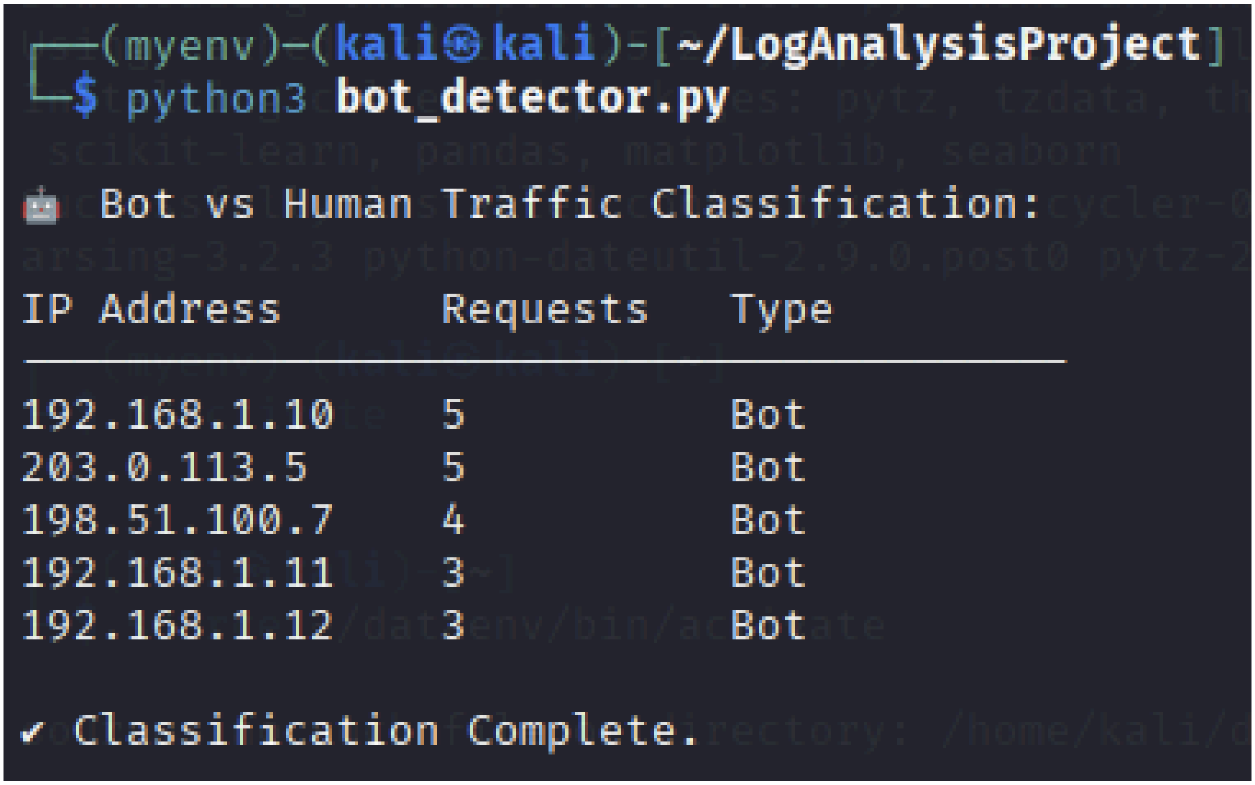


```
┌──(myenv)─(kali㉿kali)-[~/LogAnalysisProject]
└─$ python3 detect_enum.py

🔎 Directory Enumeration Detection (404 Errors per IP):

IP Address      404 Errors
──────────────────────────────
203.0.113.5     5
198.51.100.7    1

✔ Analysis Complete.
```

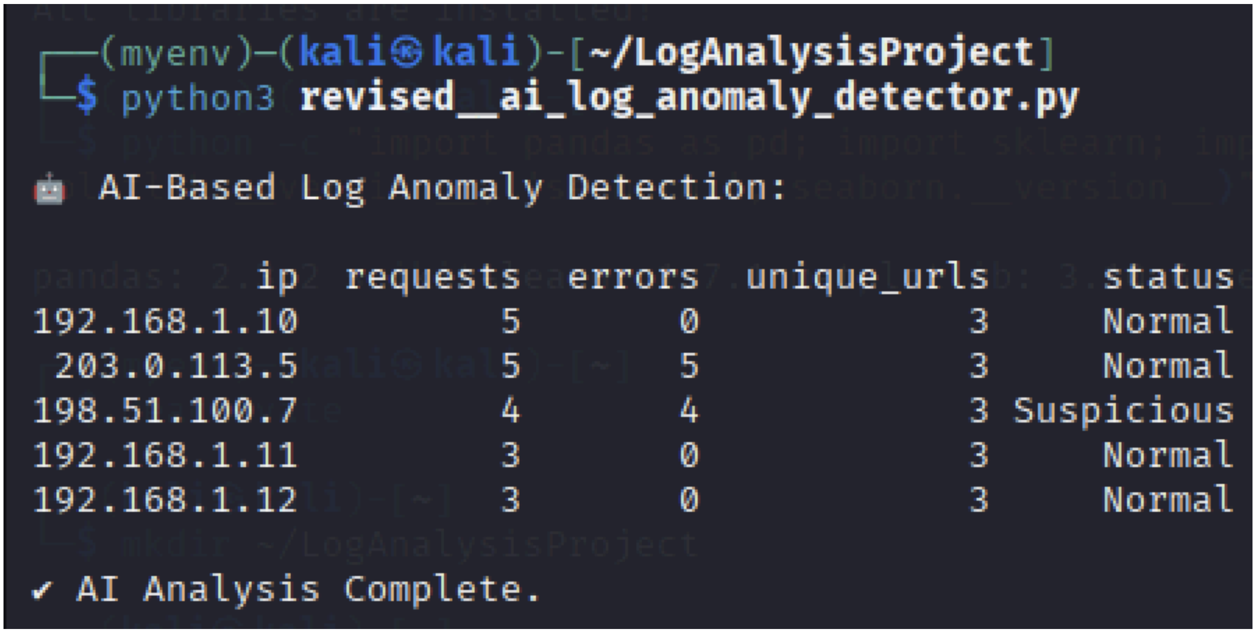


```
┌──(myenv)─(kali㉿kali)-[~/LogAnalysisProject]
└─$ python3 bot_detector.py

🤖 Bot vs Human Traffic Classification:

IP Address        Requests    Type
────────────────────────────────────────
192.168.1.10      5           Bot
203.0.113.5       5           Bot
198.51.100.7      4           Bot
192.168.1.11      3           Bot
192.168.1.12      3           Bot

✔ Classification Complete.
```



```
┌──(myenv)-(kali㉿kali)-[~/LogAnalysisProject]
└─$ python3 revised__ai_log_anomaly_detector.py

🤖 AI-Based Log Anomaly Detection:

          ip  requests  errors  unique_urls      status
192.168.1.10         5       0            3      Normal
 203.0.113.5         5       5            3      Normal
198.51.100.7         4       4            3  Suspicious
192.168.1.11         3       0            3      Normal
192.168.1.12         3       0            3      Normal

✔ AI Analysis Complete.
```

9. Conclusion

This project demonstrates that AI-augmented log analysis can significantly improve threat visibility.

Key takeaways:

- Rule-based detection (404 floods, bot UA detection) is effective for known attacks.
- AI-based anomaly detection identifies unknown patterns, improving detection coverage.
- The combined approach provides a layered defense and reduces reliance on manual monitoring.

10. Future Work

- Real-time log streaming and dashboard visualization.
- Integration with SIEM solutions like ELK Stack or Splunk.
- GeoIP lookup for attacker attribution and visualization.
- Use deep learning (LSTM/Autoencoders) for sequential log anomaly detection.