Advanced Log Analysis Tool for Cybersecurity Threat Detection

Purpose and Overview

This Python-based log analysis tool is designed to assist cybersecurity professionals in detecting suspicious activities from server log files. It identifies patterns such as high-frequency requests, unusual endpoint access, and failed login attempts, which are common indicators of potential cybersecurity threats such as brute force attacks or credential stuffing.

This tool is adaptable and provides the capability to configure suspicious activity thresholds dynamically. Additionally, it generates a detailed report in CSV format for further analysis and archival.

Key Features

1. Comprehensive Log Parsing

- Extracts IP addresses and tracks their request counts.
- Analyzes endpoint access patterns to determine the most frequently accessed resources.
- Identifies failed login attempts based on HTTP status code `401` or specific failure messages like “Invalid credentials.”

2. Configurable Suspicious Activity Threshold

- Dynamically accepts a user-defined threshold for failed login attempts.
- Default threshold (`5 failed attempts`) is applied if no input is provided within the configurable timeout period (3 seconds).

3. Automated Reporting

- Outputs analysis results to a structured CSV file for easy integration into existing cybersecurity workflows.
- Highlights suspicious IPs with high failed login attempts and provides detailed logs for incident response.

Workflow and Methodology

Input

- Log file containing HTTP server activity.
- User-defined threshold for failed login attempts (optional).

Processing

Log Parsing

1. **IP Address Extraction**:
    - Identifies and counts occurrences of each IP address.
2. **Endpoint Analysis**:
    - Tracks requests per endpoint to identify heavily accessed resources.
3. **Failed Login Detection**:
    - Matches patterns indicating login failures, such as HTTP `401` responses or the message “Invalid credentials.”

Suspicious Activity Detection

- Detects IP addresses exceeding the failed login threshold.

Output

- Prints detailed insights into request counts, endpoint access, and suspicious activities.
- Saves findings to a CSV file for further analysis.

Technical Implementation

Configuration Parameters

- **`log_file`**: Path to the input log file.
- **`output_csv`**: Path to save the analysis results.
- **`DEFAULT_FAILED_LOGIN_THRESHOLD`**: Default threshold for failed login attempts.
- **`timeout_input`**: Timeout for user input to configure the failed login threshold.

Core Functions

`timeout_input(prompt, timeout, default)`

- Prompts the user for input with a timeout mechanism.
- Uses threading to allow dynamic configuration while ensuring timely execution.

`parse_log_file(file_path)`

- Reads and parses the log file.
- Extracts:
    - IP addresses and their request counts.
    - Endpoint access patterns.
    - Failed login attempts.

`find_most_accessed_endpoint(endpoint_hits)`

- Identifies the endpoint with the highest number of hits.

`find_suspicious_ips(failed_logins, threshold)`

- Filters IP addresses that exceed the failed login threshold.

`write_to_csv(output_path, ip_requests, most_accessed_endpoint, suspicious_ips)`

- Saves the analysis results into a CSV file with structured sections for:
    - IP request counts.
    - Most accessed endpoints.
    - Suspicious activity details.

Operational Use Cases

Brute Force Detection

Identifies IPs with excessive failed login attempts, a hallmark of brute force attacks.

Resource Abuse Monitoring

Highlights endpoints with unusually high access rates, potentially signaling resource abuse or reconnaissance activities.

Incident Response Facilitation

Provides clear and structured logs for cybersecurity analysts to investigate and respond to detected threats effectively.

Security Best Practices

- Ensure the tool runs on secure and trusted environments.

- Store logs and generated CSVs in secure locations with restricted access.
- Regularly update the tool to incorporate new patterns of malicious behavior.
- Combine with other security measures like firewalls, intrusion detection systems (IDS), and rate-limiting mechanisms for comprehensive protection.

Sample Output

Console Output

IP Address Request Count

192.168.0.101 200

10.0.0.5 150

Most Frequently Accessed Endpoint:

/api/login (Accessed 300 times)

Suspicious Activity Detected:

IP Address Failed Login Attempts

192.168.0.101 15

CSV Structure

Requests per IP

IP Address, Request Count

192.168.0.101, 200

10.0.0.5, 150

Most Accessed Endpoint

Endpoint, Access Count

/api/login, 300

Suspicious Activity

IP Address, Failed Login Count

192.168.0.101, 15

Future Enhancements

1. **Integration with Threat Intelligence**:
    - Cross-reference suspicious IPs with known threat databases.
2. **Real-Time Monitoring**:
    - Adapt the tool for real-time log streaming and analysis.
3. **Visualization Dashboards**:
    - Integrate with tools like Grafana or Kibana for graphical representation of log data.
4. **Enhanced Detection**:
    - Incorporate additional indicators such as unusual geolocations or request patterns.

Conclusion

This log analysis tool is a powerful asset for cybersecurity teams. By providing real-time insights, automated reporting, and dynamic configurability, it empowers organizations to detect and respond to threats promptly. Its extensible design ensures adaptability to evolving cybersecurity challenges, making it a valuable addition to any security operations toolkit.

Developed By: Jalaj Singh

Purpose: Cybersecurity Log Analysis and Proactive Threat Management