Title: Log File Analyzer for Intrusion Detection

Introduction

Modern IT infrastructures produce a vast amount of log data. These logs are essential for monitoring, troubleshooting, and detecting unauthorized access. This project focuses on building a Python-based log file analyzer to detect brute-force attacks, scanning attempts, and DoS patterns using Apache and SSH logs. By analyzing patterns and cross-referencing IPs with known blacklists, this tool helps in identifying potential threats.

Abstract

The project aims to automate the process of detecting suspicious activities by parsing system log files. It identifies critical security patterns, such as repeated failed login attempts or high-frequency access requests. The analyzer highlights these patterns, visualizes access behaviors using graphs, and generates incident reports. The tool is especially useful for beginners in cybersecurity and those preparing for roles such as SOC analysts.

Tools Used

- **Python**: Core programming language for scripting and logic.
- **Regex**: Used for pattern matching to extract relevant fields from logs.
- **Pandas**: For data manipulation and aggregation.
- **Matplotlib**: For generating visual representations like bar charts and line graphs.

Steps Involved in Building the Project

1. **Log Collection**
   Gather sample Apache access logs and SSH authentication logs. These logs typically contain IP addresses, timestamps, request details, and login attempts.
2. **Log Parsing using Regex**
   Use regular expressions to extract IP addresses, timestamps, login statuses (e.g., failed password), and request methods (GET, POST, etc.).
3. **Data Structuring with Pandas**
   Convert parsed log data into structured DataFrames for analysis. Aggregate failed attempts and access frequency per IP.

4. **Suspicious Pattern Detection**
    - Flag IPs with >N failed login attempts (brute-force).
    - Identify IPs accessing >X unique endpoints rapidly (scanning).
    - Detect more than Y requests in Z seconds (DoS behavior).
5. **Cross-reference with Public IP Blacklists**
   Download and use public IP threat lists (like from AbuseIPDB) to cross-check and highlight known attackers.
6. **Visualization**
   Use matplotlib to generate:
    - Bar charts of failed login attempts per IP
    - Line graphs showing traffic volume over time
7. **Incident Reporting**
   Export flagged incidents to a CSV or JSON file with fields such as IP, timestamp, activity type, and threat level.

Conclusion

This log file analyzer project provides a foundational step into the world of threat detection and incident analysis. It simulates the basic tasks performed by a SOC analyst using lightweight and interpretable tools. The project also lays the groundwork for enhancements like email alerts, real-time log streaming, or integrating with SIEM systems in the future.