LINUX FORENSIC ANALYSIS

Tool for Identifying and Detailing Security Breach Attempts on Linux

Project Team:
Deepti Bhat
Harshitha S M
Hemanth Kumar S
Nandish H R
Ramvikas S V

Mentor:
Smita Pawar

College Mentor:
Tanuja Kayarga

HPE-CTY 2024
VIDYAVARDHAKA COLLEGE OF ENGINEERING

Research Summary

Research on Basics of Linux and Cybersecurity

Introduction

In the early stages of our project, we focused on understanding the fundamental aspects of Linux systems and the principles of cybersecurity, particularly the CIA Triad. This foundational knowledge is critical to ensure that our tool is robust, reliable, and secure. Our research encompassed Linux architecture, user authentication mechanisms, and the core principles of cybersecurity, leading to the formulation of the primary requirements and functionalities of our forensic tool.

Linux Architecture and User Authentication

We delved into the Linux operating system to understand its architecture and user authentication processes. Key aspects include:

- **Authentication Mechanisms:** Linux uses various authentication mechanisms, including password-based authentication, public key authentication, and pluggable authentication modules (PAM). These mechanisms ensure that only authorized users can access the system.
- **Authentication Logs:** Linux maintains detailed logs of authentication attempts in files such as /var/log/auth.log. These logs record successful and failed login attempts, password changes, and other authentication-related events, which are crucial for forensic analysis.

Cybersecurity Triads and Essentials

Our tool is designed with a strong focus on the CIA Triad and the five pillars of cybersecurity to ensure comprehensive protection and resilience.

CIA Triad:

1. **Confidentiality:** Ensures that sensitive data is accessible only to authorized individuals.
2. **Integrity:** Ensures that data remains consistent, accurate, and trustworthy throughout its lifecycle.
3. **Availability:** Ensures that data and systems are available to authorized users when needed.

Five Pillars of Cybersecurity:

1. **Risk Management:** Identifying, assessing, and prioritizing cybersecurity risks and vulnerabilities.
2. **Access Control:** Mechanisms to manage and restrict access to systems, networks, and data.
3. **Data Protection:** Safeguarding sensitive information from unauthorised access, disclosure, alteration, or destruction.
4. **Security Awareness and Training:** Educating users about cybersecurity best practices and how to recognize and respond to security incidents.
5. **Security Monitoring and Incident Response:** Continuously monitoring for signs of suspicious activity and responding to security breaches effectively.

Understanding the Problem Statement

The project's primary goal is to develop a reliable and automated tool that can:

- Identify and detail security breach attempts on Linux machines.
- Gather and analyze forensic data to generate comprehensive reports.
- Ensure accessibility of these reports even if the system is isolated or compromised.

The tool aims to assist system administrators and security professionals in promptly identifying and responding to security incidents.

Core Functionality of the Tool

We focused on understanding the types of security breaches in Linux systems, particularly brute force attacks. The core functionalities of our tool include:

- Monitoring login attempts and identifying multiple failed logins, which are indicative of brute force attacks.
- Tracking password changes to detect unauthorized modifications.
- Integrating a malware scanner to identify and respond to suspicious activities, enhancing the tool's scalability and threat detection capabilities.

Choice of Linux OS for the Project

We compared various Linux distributions such as Ubuntu, Parrot OS, and Kali Linux. **Parrot OS** was chosen for its superior forensic analysis capabilities, including:

- A comprehensive suite of pre-installed forensic tools.
- Enhanced security features tailored for penetration testing and forensic analysis.
- A lightweight and efficient environment suitable for continuous monitoring and analysis.

Research on Basic Tools

We explored various tools available in Linux and Parrot OS, including:

- **Wireshark:** For network protocol analysis.
- **Nmap:** For network scanning and discovery.
- **Auditctl:** For auditing and monitoring system activities.

Selection of Appropriate Tools

For identifying failed login attempts, we decided to monitor the auth.log files in Parrot OS using a bash script. The choice of auth.log over journalctl was due to its flexibility and ease of integration. For continuous monitoring, we integrated the bash script with a Python script to automate the process.

To monitor password changes, we implemented a similar approach using auth.log files. For malware scanning, we chose ClamAV along with the Watchdog library due to its advantages over Pynotify, including better performance and more robust monitoring capabilities.

Forensic Report Format

We researched various forensic report formats to design a structured report that meets user needs. The report includes detailed information about breach attempts, system activities, and recommendations for mitigating identified threats.

Delivering Forensic Reports to Users

To deliver forensic reports to end users, we decided to send email via SMTP with the help of python script. This script allows seamless integration with our tool, enabling the automated generation and delivery of reports via email, ensuring that system administrators receive timely and actionable insights.

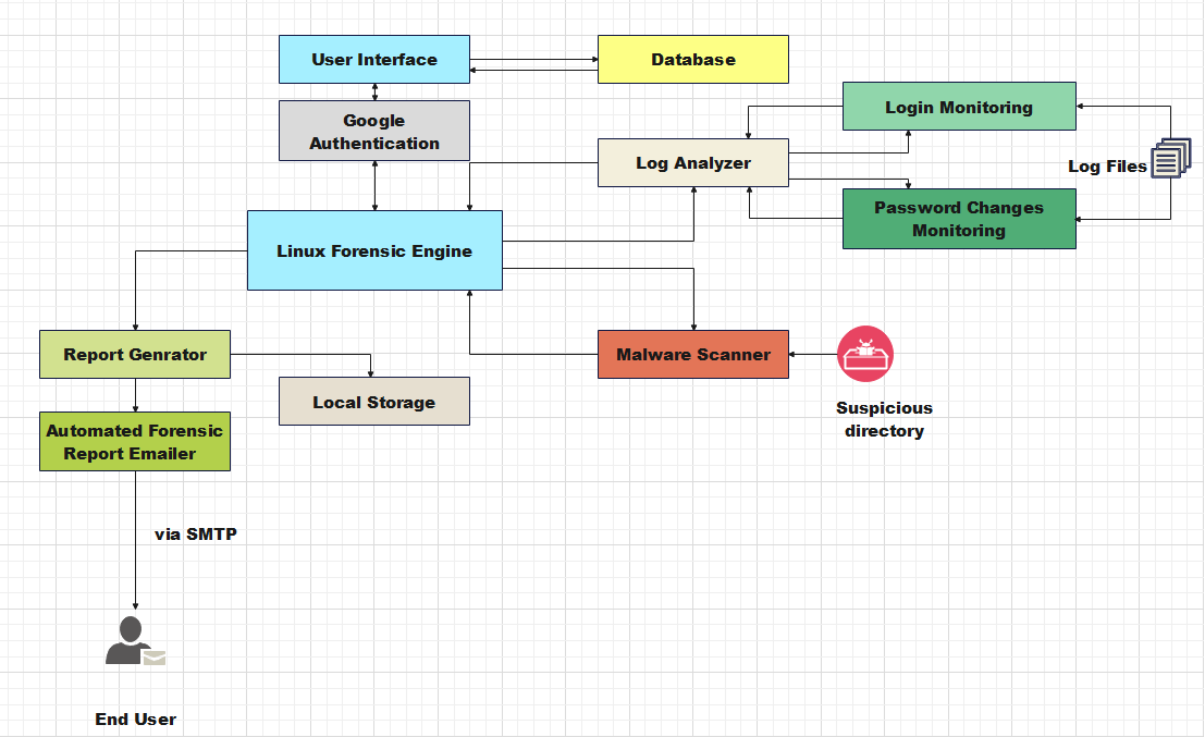
Why Our Tool is Essential

Analyzing log files is crucial but challenging due to the sheer volume of data generated by modern systems. This complexity is especially pronounced in forensic investigations, where quick and accurate log analysis is vital for understanding security breaches and mitigating their impacts. Our Linux Forensic Analysis Tool addresses this need by automating the examination of log files and enhancing malware detection capabilities. This dual functionality ensures swift and precise analysis, enabling the creation of comprehensive forensic reports that help reconstruct events leading to security incidents and identify unauthorized access attempts.

Integrating a malware analyzer plugin further bolsters the tool's effectiveness. Malware analysis is essential for identifying software that could compromise system integrity. Our tool's real-time monitoring of login attempts and password changes, coupled with its ability to scan for malware, provides a holistic approach to system security. This comprehensive solution is crucial for system administrators and security professionals, allowing them to quickly identify and respond to threats, ensuring a secure and resilient computing environment.

High-Level Design Overview

The high-level design diagram provides an architectural view of the Linux Forensic Analysis Tool, illustrating the flow of data and interactions between various components.



Key Component	Function	Interaction
User Interface	Allows users to interact with the tool, enabling features such as login attempts monitoring, password changes monitoring, and malware security threat monitoring.	Communicates with the Linux Forensic Engine and Database.
Google Authentication	Provides user authentication using Google's authentication services.	Ensures secure access to the tool and integrates with the User Interface and Linux Forensic Engine.
Linux Forensic Engine	Acts as the core processing unit of the tool, coordinating data collection, analysis, and report generation.	Interfaces with the Log Analyzer, Malware Scanner, and Report Generator.
Log Analyzer	Analyzes log files for suspicious activities such as frequent failed login attempts and suspicious password changes.	Works with Login Monitoring and Password Changes Monitoring components to gather relevant data from log files.
Login Monitoring	Monitors log files for frequent failed login attempts and other suspicious login activities.	Sends data to the Log Analyzer for further processing.
Password Changes Monitoring	Monitors log files for suspicious password changes.	Sends data to the Log Analyzer for further processing.
Malware Scanner	Scans the system for malware threats using tools like watchdog and clamscan.	Monitors directories for malware files and sends result to the Linux Forensic Engine.
Report Generator	Generates detailed forensic reports based on the analyzed data.	Receives processed data from the Linux Forensic Engine and prepares comprehensive reports.
Automated Forensic Report Emailer	Automatically emails the generated forensic reports to the user.	Utilizes SMTP to send reports generated by the Report Generator to the end user.

Key Component	Function	Interaction
Database	Stores data related to user configurations such as thresholds and user contact details.	Interacts with the User Interface and Log Analyzer to retrieve and store necessary data (thresholds + email).

Components Breakdown

1. User Interface

Function:

The User Interface (UI) allows users to interact with the tool. It provides features for monitoring login attempts, password changes, and malware threats.

Technical Particulars:

Front-end Framework: Implemented using a NW.js framework with HTML/CSS and JavaScript for responsive design.

Endpoints: Includes GET, PUT and POST methods to handle user requests and display data fetched from the backend.

Authentication Integration: Interfaces with Google Authentication services to ensure secure access.

Data Display: Presents data retrieved from the Database and processed by the Linux Forensic Engine in a user-friendly manner.

Interaction:

With Google Authentication: Verifies user identity before granting access.

With Linux Forensic Engine: Sends user commands and data.

With Database: Fetches user-specific configurations and preferences.

2. Google Authentication

Function:

Provides secure user authentication using Google's authentication services.

Interaction:

With User Interface: Facilitates login and session management.

With Linux Forensic Engine: Ensures only authenticated requests are processed.

3. Linux Forensic Engine

Function:

Acts as the core processing unit, coordinating data collection and report generation.

Technical Particulars:

Core Components: Comprises various modules like Log Analyzer, Malware Scanner, and Report Generator.

Data Processing and generation: The above mentioned blocks use Python scripting for handling bash scripts, which monitor `auth.log` files in authentication cases. The Python scripts, `loginstart.py`, `passstart.py`, and `malware.py`, trigger the functions for monitoring and thus enable the functionality of the bash script and mailing script recursively. Additionally, the `malware.py` script integrates with the ClamScan utility to generate malware scan reports.

Thread Management: Manages concurrent processing of logs and malware scanning.

Interaction:

With Log Analyzer: Receives log data of failed login attempts and password changes.

With Malware Scanner: Directs malware scanning tasks and receives results.

With Report Generator: Passes analyzed data for report creation.

With Database: Coordinates with data stored in db which is responsible for performing tasks under various scenarios or user requested cases.

4. Log Analyzer

Function:

Analyzes log files for suspicious activities such as frequent failed login attempts and suspicious password changes.

Technical Particulars:

Log Parsing: Utilizes tools like grep and awk for efficient log parsing.

Interaction:

With Login Monitoring and Password Changes Monitoring: Collects relevant log data.

With Linux Forensic Engine: Sends analyzed results for further processing.

5. Login Monitoring

Function:

Monitors log files for frequent failed login attempts and other suspicious login activities.

Technical Particulars:

Log Files: Monitors /var/log/auth.log and similar files for login attempts.

Thresholds: Configurable thresholds for identifying brute force attacks.

Features:

If the number of break-in attempts exceeds the threshold within a specified time set by the user, further action will be taken to detail the report and gather all additional information.

Interaction:

With Log Analyzer: Feeds data into the Log Analyzer for detailed examination.

With Database: Updates thresholds and configuration settings.

6. Password Changes Monitoring

Function:

Monitors log files for suspicious password changes.

Technical Particulars:

Log Files: Tracks /var/log/auth.log for password change events.

Change Detection: Uses time-based and event-based triggers to detect changes.

Alert Mechanism: Notifies the Log Analyzer upon detecting suspicious activities.

Features:

If the number of password changes exceeds the threshold within a specified time set by the user, further action will be taken to detail the report and gather all additional information.

Interaction:

With Log Analyzer: Provides log data related to password changes.

With Database: Stores configuration settings and user preferences.

7. Malware Scanner

Function:

Scans the system for malware threats using tools like Watchdog and ClamAV.

Technical Particulars:

Watchdog Library: Monitors file system changes in real-time.

ClamAV Integration: Uses ClamAV for scanning and identifying malware.

Real Time Scanning: Implements scans when suspicious activity occurs (in our case file creation).

Interaction:

With Linux Forensic Engine: Reports scanning results for further action(email delivery).

8. Logging for Local Storage

Function:

Store reports in log format and implement log rotation.

Technical Particulars:

Python logging library: Converts received reports to log format and saves them in the local system, facilitating log configuration without the need for additional configuration files.

Log Configuration: Maintains 5 backup log files, each up to 1MB, and rotates logs.

Interaction:

With Linux Forensic Engine: Receives reports from python scripts and handles log rotation.

Client-Centric Tool Workflow

Entry Point

The user accesses the tool via a standalone application

User Authentication

Login Screen: The user is presented with a login screen where they enter their email and password.
Registration (if applicable): New users can register by providing necessary details such as email, password, and other required information.

Dashboard

Welcome Screen: Upon successful login, the user is directed to the dashboard, displaying an overview of the tool's features.
Navigation Menu: The dashboard includes a navigation menu with options such as Account and Services.

Feature Configuration

After logging in, the user can configure various security features:

Login Fail Frequency: The user sets the frequency at which login failures are monitored.
Login Fail Duration: The user specifies the duration within which failed logins are tracked.
Password Change Frequency: The user sets how often password changes are monitored.
Password Change Duration: The user specifies the duration within which password changes are tracked.
Malware Monitoring Path: The user sets the directory path for malware monitoring.

Enabling Services

Once the features are configured, the user can enable various security services:

Login Monitoring: This service monitors the Linux system for failed login attempts. It provides alerts and reports on unusual login activity.
Password Change Monitoring: This service tracks suspicious password changes, alerting the user to potential security breaches.
Malware Detection: This service scans the specified directory for malware files, identifying and isolating potential threats.

Exiting the Tool

Logging Out: The user can log out by clicking on the profile icon and selecting 'Log Out'.

Future Scope

1. Behavioural Analysis and Anomaly Detection:

By integrating machine learning algorithms, the tool can be enhanced to perform advanced behavioural analysis and anomaly detection. This would allow the system to learn from historical data and identify unusual patterns that may indicate potential security threats or breaches.

2. Automated Incident Response:

Developing automated incident response capabilities can streamline the process of handling detected security incidents. This feature could include predefined actions such as isolating affected systems, alerting administrators, and initiating recovery protocols to minimize the impact of a security breach.

3. Enhanced User Interface and Reporting:

Improving the user interface to provide more intuitive and customizable dashboards can help users better visualize and understand security data. Additionally, offering more detailed and configurable reporting options will allow for tailored insights that meet specific organizational needs.

Conclusion

The development of this Linux Forensic Analysis Tool aims to enhance the security of Linux-based systems. By automating the detection and analysis of security breaches, the tool provides insights and actionable intelligence that assist system administrators and security professionals in responding more effectively to threats.

Our research and implementation have demonstrated the tool's ability to monitor login attempts, track password changes, and scan for malware, ensuring comprehensive coverage of critical security aspects.

Moving forward, the proposed enhancements, such as integrating machine learning for anomaly detection and incorporating real-time threat intelligence, will elevate the tool's capabilities to address evolving security challenges. By continuing to innovate and expand its features, we aim to provide a robust, scalable, and user-friendly solution that meets the dynamic needs of Linux users.

In conclusion, this tool not only addresses immediate security needs but also lays a foundation for future advancements in forensic analysis and incident response, contributing to a more secure and resilient computing environment.