HashLine - A File Integrity Monitor

Submitted in partial fulfillment of the requirements

of the degree

BACHELOR OF ENGINEERING

IN INFORMATION TECHNOLOGY

By

Vinaya Redekar	**22101B2004**
Anam Ansari	**22101B2005**
Arisha Rakhangi	**22101B2006**

Supervisor

Prof. Debarati Ghosal

VIT | Vidyalankar Institute of Technology

Accredited A+ by NAAC

Department of Information Technology

Vidyalankar Institute of Technology

Vidyalankar Educational Campus,

Wadala(E), Mumbai - 400 037

University of Mumbai (AY 2024-25)

CERTIFICATE

This is to certify that the Mini Project entitled "**HashLine - A File Integrity Monitor"** is a bonafide work of **Vinaya Redekar(22101B2004), Anam Ansari(22101B2005), Arisha Rakhangi(22101B2006)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in **"Information Technology" .**

Prof. Debarati Ghosal
Supervisor

Dr. Vidya Chitre
Head of Department

Dr. S. A. Patekar
Principal

Mini Project Approval

This SAD LAB Mini Project entitled **"HashLine - A File Integrity Monitor"** by **Vinaya Redekar(22101B2004), Anam Ansari(22101B2005), Arisha Rakhangi(22101B2006)** is approved for the degree of **Bachelor of Engineering** in **Information Technology.**

Examiners

1.....
(Internal Examiner Name & Sign)

2.....
(External Examiner name & Sign)

Date:

Place:

Contents

Abstract

In today's rapidly evolving digital landscape, ensuring the integrity of critical files and directories is paramount for maintaining system security. HashLine, a Python-based File Integrity Monitoring solution, offers a robust mechanism for validating the integrity of files and directories by detecting unauthorized modifications, additions, or deletions. Developed exclusively for the Windows operating system, HashLine utilizes SHA512 hashing to compare the current state of files with a pre-established baseline, ensuring that any file tampering is promptly identified. The application provides a user-friendly interface built with the tkinter module, allowing users to easily browse directories, update baselines, and check file integrity. HashLine supports all file types and includes comprehensive monitoring of subdirectories. By maintaining a separate baseline for each monitored directory, HashLine offers a reliable, secure, and scalable solution for protecting file integrity in various environments.

Acknowledgments

I would like to take this opportunity to extend our heartfelt gratitude to our esteemed professor, Prof. Debarati Ghosal, for providing us with the invaluable opportunity to undertake this project on " HashLine - A File Integrity Monitor". Her unwavering guidance and mentorship helped us gain precious knowledge and conduct extensive research.

I would also like to express our sincere appreciation to our Head of Department for his/her support and encouragement throughout the project.

Finally, we would like to express our appreciation to our parents for their unrelenting support and encouragement.

STUDENT NAME

Vinaya Redekar(22101B2004)

Anam Ansari(22101B2005)

Arisha Rakhangi(22101B2006)

Chapter 1: Introduction

1.1 Introduction

File Integrity Monitoring (FIM) is a process designed to monitor and detect changes made to files. This process ensures that files, whether system files or application data, remain in a trusted state, unaffected by unauthorized changes. HashLine is a File Integrity Monitor (FIM) developed to ensure that files remain unchanged and are not subject to tampering or corruption. By comparing the current state of a file with a known good state or "baseline," HashLine can alert users to any discrepancies, ensuring the integrity of the system's file structure.

1.2 Motivation

In an era where data integrity is critical for personal, business, and institutional information, there is a growing demand for reliable tools that can help users monitor their files. Traditional file integrity monitoring tools are either too complex for the average user or not flexible enough to meet modern needs. The goal behind HashLine is to create a user-friendly, efficient, and lightweight FIM solution that can easily be implemented on Windows platforms, catering to both technical and non-technical users. HashLine's intuitive graphical user interface (GUI) ensures that users can monitor file integrity with minimal effort, making it accessible for everyday use.

1.3 Problem Statement & Objectives

Problem Statement:

HashLine aims to solve the problem of detecting unauthorized file changes by offering a system that can:

- Detect file tampering by comparing file hashes with previously stored baselines.
- Work effectively with all types of files, including documents, presentations, and spreadsheets.
- Allow users to monitor entire directories, including their subdirectories, for any changes.
- Provide an easy-to-use interface for users with minimal technical expertise. The objective is to provide a lightweight, GUI-based tool for Windows that offers comprehensive file integrity monitoring without the complexities seen in

larger enterprise-level solutions.

Objectives:

1. **Ensure File Integrity**: To develop a solution that monitors and detects any unauthorized changes, deletions, or additions to files and directories by comparing the current file state with a baseline using SHA512 hashing.
2. **Provide a User-Friendly Interface**: To offer a simple and intuitive graphical user interface (GUI) using tkinter that allows users to easily browse directories, update baselines, and check file integrity.
3. **Support a Wide Range of File Types**: To ensure that the solution can handle and monitor various file formats, including .doc, .txt, .rtf, .ppt, .xlsx, and others.
4. **Enable Subdirectory Monitoring**: To implement functionality that monitors both directories and their subdirectories, allowing users to track changes across an entire file structure.
5. **Baseline Management**: To create and maintain separate baseline files for each monitored directory, storing file paths and their respective SHA512 hashes, ensuring efficient integrity checking.
6. **Real-time Tampering Detection**: To provide timely identification of any file tampering or changes, including file modification, deletion, or addition, based on baseline comparisons.
7. **Scalability Across Directories**: To enable integrity checking for multiple directories with independent baselines, offering flexibility in monitoring different sections of a file system.

Chapter 2: Literature Survey

2.1 Survey of Existing/Similar System

1. **Tripwire:** Tripwire is a widely used file integrity monitoring solution that tracks changes in file systems and directories. It creates cryptographic hashes of the monitored files and compares them to known baselines to detect unauthorized changes. Tripwire is highly configurable and suitable for both Linux and Windows systems [1].

   Limitation: While comprehensive, Tripwire can be resource-heavy and difficult to configure for users who are not familiar with security or system administration.

2. **AIDE (Advanced Intrusion Detection Environment):** AIDE is an open-source file integrity checker commonly used on Linux systems. Similar to Tripwire, AIDE builds a database of cryptographic checksums for all monitored files and directories, then checks them against the current state of the files [2].

   Limitation: AIDE lacks GUI support, making it less user-friendly for non-technical users. It is also predominantly Linux-focused and not well-suited for Windows users.

3. **OSSEC**
   OSSEC is an open-source security monitoring tool that includes file integrity monitoring as part of its feature set. It monitors files for changes by checking cryptographic hashes, among other security features like intrusion detection and log analysis [3].

   Limitation: OSSEC is mainly designed for enterprise-level systems with extensive security requirements, making it complex and difficult to set up for smaller environments or personal use.

2.2 Limitation Existing/Similar system or research gap

Despite the reliability and effectiveness of existing file integrity monitoring systems, there are several limitations:

- **Complexity in Setup and Usage:** Most existing systems, such as Tripwire and OSSEC, require significant configuration effort, making them difficult for non-technical users to implement effectively. This complexity can be a barrier for small businesses or individual users who need simple solutions.
- **Resource Intensive:** Many FIM solutions are built for large-scale environments and can be resource-intensive, impacting system performance.

- **Lack of User-friendly GUI:** Systems like AIDE lack graphical interfaces, which makes it harder for non-expert users to operate and monitor file integrity.
- **Limited Windows Support:** Some solutions, such as AIDE, are predominantly Linux-focused, limiting their usefulness for Windows-based environments.

How HashLine Addresses These Limitations:

- **Ease of Use:** HashLine offers a simple and intuitive graphical interface (built using the tkinter module) that makes file integrity monitoring accessible for users without advanced technical knowledge.
- **Lightweight Solution:** HashLine is designed to be lightweight and resource-efficient, ensuring it can be used on smaller-scale systems without impacting performance.
- **Support for Windows:** Unlike some other FIM tools, HashLine is built specifically for the Windows operating system, making it more relevant for users in that environment.

Monitoring Subdirectories: HashLine efficiently monitors not only files but also subdirectories within a selected directory, providing comprehensive coverage.

2.3 Mini Project Contribution

In our collaborative project, each team member played a distinct role to enhance different aspects of the website. One member took charge of the 'About Us' and 'Contact' pages, ensuring that these sections effectively communicated our company's identity and provided seamless ways for users to connect with us. Another member dedicated their efforts to the 'Products' and 'Careers' pages, focusing on showcasing our product offerings and creating an engaging platform for potential job seekers. The third member, on the other hand, was instrumental in revamping the 'Home' page and strengthening our 'Market Presence,' where their design and content contributions significantly improved our website's initial impact and industry visibility. This division of responsibilities allowed us to collectively create a well-rounded and comprehensive web presence.

Chapter 3: Proposed System

3.1 Introduction

The proposed system, HashLine, is designed to detect and prevent unauthorized tampering of files and directories within a Windows-based environment. The system uses SHA512 hashing to verify the integrity of files by comparing their current state with a known good state, also called the baseline. The system is designed to be user-friendly, providing a graphical interface for managing file integrity checks.

3.2 Architecture/ Framework

The HashLine system is structured around three core functions: Directory Browsing, Baseline Management, and Integrity Checking. The system architecture consists of the following components:

- **Frontend (User Interface)**: Built using Python's tkinter library, the graphical user interface (GUI) provides a simple and intuitive way for users to interact with the system. The GUI allows users to browse directories, update the baseline, and check file integrity.
- **Backend (Hashing and File Management)**: The backend handles the file system interaction (using the os module) and performs hashing using the hashlib library (SHA512 algorithm). This component is responsible for storing and retrieving baseline data and performing integrity checks.

HashLine operates based on a predefined workflow that ensures the integrity of files and directories. The following explains the step-by-step operations:

3.2.1. Directory Browsing

- **Objective**: Allow the user to select a directory for monitoring.
- **Implementation**: The "Browse" button triggers a directory selection prompt, allowing the user to select the directory to be monitored. This directory path is stored temporarily for further actions.
- **User Interaction**: When the user clicks on the 'Browse' button, a directory selection window is opened. The selected directory is displayed in the GUI and is used for baseline updates and integrity checks.

3.2.2. Updating the Baseline

- **Objective**: Create or update a baseline of the selected directory that stores the current file structure and hashes.
- **Implementation**: When the user selects a directory and clicks the "Update

Baseline" button, the system first checks if the user has selected a directory. If not, an error message is displayed. If a directory is selected, the system reads each file in the directory (including subdirectories), calculates its SHA512 hash, and stores the file path and hash in a baseline file.

- **Baseline File**: A simple text file that stores the file paths and their respective hashes in the following format:

  makefile

  FilePath1 = Hash1

  FilePath2 = Hash2

  FilePath3 = Hash3

- **User Interaction**: If the user clicks "Update Baseline", HashLine verifies if a directory was selected. If a directory is selected, the system creates or updates a baseline, ensuring the baseline accurately reflects the current state of files in the directory.

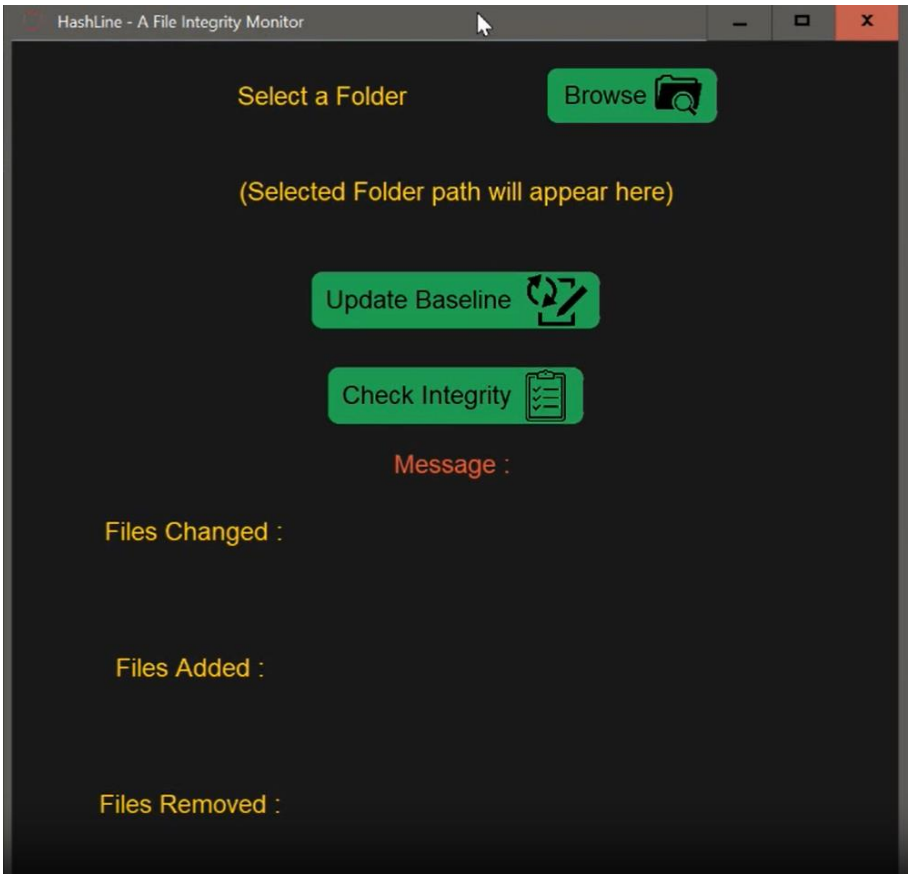
3.2.3. Checking File Integrity

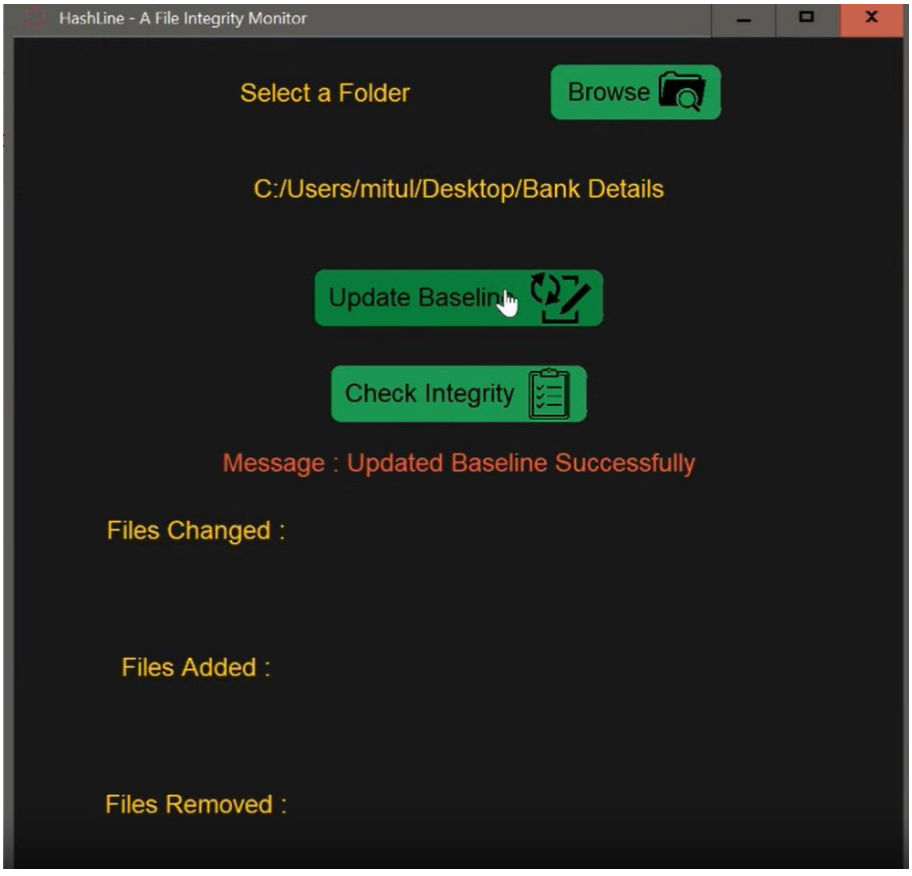
- **Objective**: Compare the current state of files in a directory with the baseline to detect any tampering or unauthorized changes.
- **Implementation**: When the user clicks the "Check Integrity" button, the system again checks whether a directory has been selected. If not, an error message is shown. The system then checks if a baseline exists for the selected directory. If no baseline is found, an error message prompts the user to update the baseline. If a baseline exists, the system reads the baseline and compares the current file paths and hashes with those in the baseline.
  - **Hash Comparison**: If a file's current hash differs from the stored hash, it indicates tampering or modification.
  - **File Additions/Deletions**: If new files are found that were not in the baseline, the system flags them as additions. If files from the baseline are missing, the system flags them as deletions.
- **User Interaction**: Upon clicking "Check Integrity", HashLine performs a comparison with the baseline and alerts the user about any changes, additions, or deletions in the files and subdirectories.

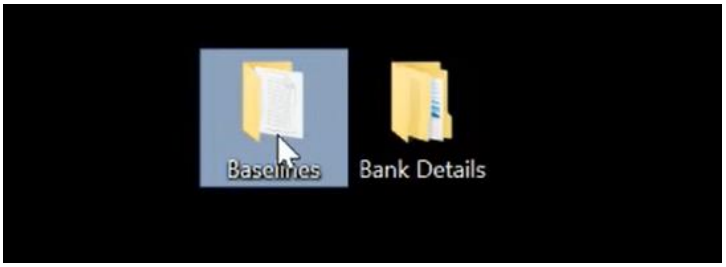
3.3 Details of Hardware & Software

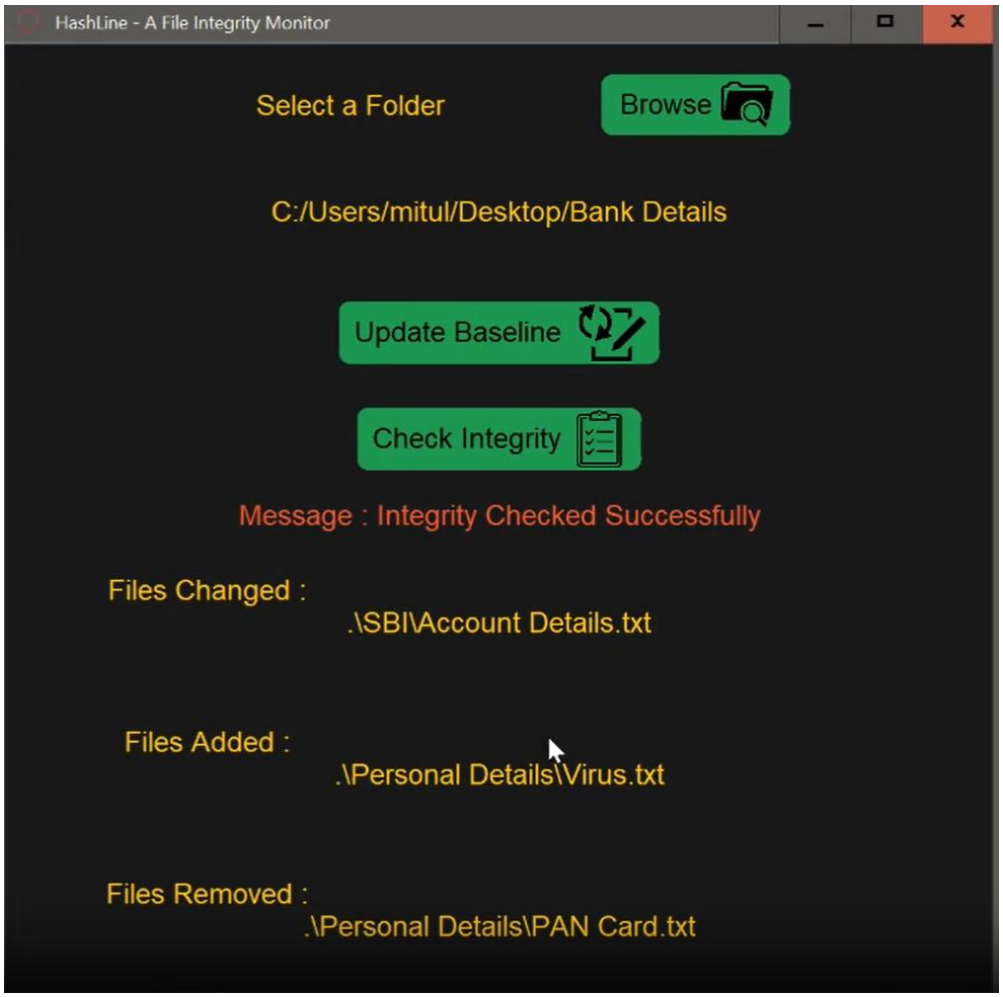
- **Hardware Requirements**
  - **Processor**: Intel Core i3 or higher
  - **RAM**: 4 GB or more
  - **Disk Space**: 500 MB free space
  - **Operating System**: Windows 10/11
  - **Display**: 1024x768 resolution or higher
  - **Input Devices**: Mouse, Keyboard
- **Software Requirements**
  - **Operating System**: Windows 10/11
  - **Programming Language**: Python 3.7 or above
  - **Python Libraries**:
    - os (built-in)
    - hashlib (built-in)
    - tkinter (built-in)
  - **IDE**: VSCode / PyCharm / Sublime Text
  - **Additional Tools**: Pip (for managing Python libraries)
- **Installation Steps**
  1. Install Python 3.7+ and ensure it's added to PATH.
  2. No extra libraries needed (tkinter and hashlib are built-in).
  3. Use IDE like VSCode or PyCharm for development.
  4. Run HashLine using: python hashline.py.

3.4 Experiment and Results









3.5 Conclusion and Future

Conclusion

HashLine is a secure, efficient, and user-friendly File Integrity Monitoring solution developed in Python for the Windows operating system. It provides a reliable method for detecting unauthorized file modifications, additions, and deletions by comparing the current state of files with a known baseline using the SHA512 hashing algorithm. The easy-to-use graphical interface, built with tkinter, allows users to effortlessly browse directories, update baselines, and check file integrity. With support for all file types and subdirectory monitoring, HashLine serves as an essential tool for protecting sensitive data in various environments, from corporate IT systems to personal use.

Future Scope

1. **Cross-Platform Support**: Extend HashLine's compatibility to other operating systems like Linux and macOS to make it a universally applicable file integrity solution.
2. **Real-Time Monitoring**: Implement real-time file integrity monitoring that automatically detects changes without manual checks, providing instant alerts on any file tampering.
3. **Notification System**: Introduce an alerting system that sends notifications (via email or SMS) when any unauthorized changes are detected in the monitored files or directories.
4. **Encrypted Baseline Files**: Add security features to encrypt and protect the baseline file from tampering, ensuring only authorized users can modify or access it.
5. **Cloud-Based Monitoring**: Enable HashLine to monitor cloud storage systems and remote directories, expanding its utility for modern cloud environments.
6. **Enhanced Reporting**: Develop detailed log reports of file integrity checks and changes, allowing users to track historical data and perform in-depth audits.

REFERENCES

[1] Qiaoyun Gu, Anxin Li, The Design and Implementation of File Integrity Monitoring System Based on Windows [J], Computer Engineering, 2004, 30(12)

[2] Da Su, Virtualization: The Key of High Effeciency IT [J], Computer Users of China, 2009(9)

[3] Daniel P B. Marco Cesat. Understanding the Linux Kernel [M], 3rd Edition, Sebastopol: O'Reilly, 2005