CNS Project : Design Document

Submitted to

Indian Institute of Technology, Gandhinagar

In partial fulfillment of the requirements for the course

CNS631: Computer & Network Security

by

Mallika Chouhan (24250052)
Shruti Dubey (24250089)
Naveen Kumawat (23120047)
Aniket Asati (24210012)
Jenil Pradipkumar Patel (24210048)
Krishan Kumar Sharma (24210057)

INDEX

Problem Statement

Title: Security Guard at IITGN

Description:

The aim of this project is to design a secure system for tracking the entry, exit, and presence of individuals on campus. The system includes two command-line tools:

1. `logappend` – Appends encrypted entries to a secure log.
2. `logread` – Reads and processes the log entries based on specific user queries.

Key Requirements:

- **Privacy:** Ensure that log data is encrypted, protecting it from unauthorized access.
- **Integrity:** Maintain the accuracy and consistency of log entries by securing the log against tampering.
- **Authentication:** Ensure that only users with the correct key can append to or read from the log.

Deliverables:

- A fully functional system with source code, supporting both logappend and logread functionalities.
- A Makefile to simplify building and setting up the system.
- A design document outlining the architecture, functionality, and security considerations of the system.

Build Requirements:

- Use a private GitHub repository.
- Provide a makefile to build `logappend` and `logread` executables.

Group Members

Name	Role
Mallika Chouhan (24250052)	Project Manager, Testing & Documentation
Shruti Dubey (24250089)	Logappend & Logread Development
Naveen Kumawat (23120047)	Logappend & Logread Development
Krishan Kumar Sharma (24210057)	Encryption & Authentication
Jenil Pradipkumar Patel (24210048)	Encryption & Authentication
Aniket Asati (24210012)	Testing & Documentation

Languages, Frameworks and Tools

The following technologies and frameworks were used in this project:

- **Programming Language:** Python 3.9
- **Encryption Library:** `cryptography` package (Fernet symmetric encryption)
- **Build Tool:** Makefile
- **Version Control:** GitHub for managing source code and project collaboration
- **Text-based Storage:** Plain text files are used for log storage, ensuring simplicity while leveraging encryption for security.

System Architecture

The system consists of the following components:

1. `logappend` **Module**
    - The `logappend.py` script is responsible for appending new entries to the log file.
    - It supports batch processing of entries from a file, allowing multiple log entries to be processed at once.
    - When appending an entry, the following process is followed:
        1. **Input Validation:** The user must specify a valid timestamp, user type, event type, and optionally a room number.
        2. **Authentication:** The token in the log file is compared with the user-provided token to ensure the user is authorized.
        3. **Encryption:** The log entry is encrypted using Fernet before it is appended to the log.
        4. **Append to Log:** The encrypted entry is added to the log file, preserving its integrity.
2. `logread` **Module**
    - The `logread.py` script is responsible for reading and querying the log file.
    - It supports multiple operations, including:
        1. **State Queries:** Printing the current state of the campus, including which employees and guests are currently on campus and in which rooms.
        2. **Time Calculation:** Calculating the total time spent on campus by an employee or guest, based on their arrival and departure times.
        3. **Room Queries:** Listing all rooms visited by an employee or guest.
        4. **Common Room Queries:** Identifying rooms that were occupied by specified individuals at the same time.
    - All log entries are decrypted on the fly to perform the necessary computations.

3. **Authentication and Encryption**
    - A **secret key** is generated by the `setup.py` script, which is stored in a file named `secret.key`.
    - This secret key is used for both encrypting and decrypting log entries.
    - The system uses Fernet symmetric encryption, which is based on AES encryption, ensuring that data remains secure.
    - During log creation and access, the provided token is encrypted and stored within the log file for future verification.

Functionalities

Module	Description	Team Member(s)
logappend	Appends new, encrypted log entries to the log.	Naveen Kumawat, Shruti Dubey
logread	Reads and queries encrypted log entries.	Naveen Kumawat, Shruti Dubey
Authentication	Ensures secure token-based authentication before any operation.	Jenil Patel, Krishan Kumar
Encryption	Encrypts and decrypts log entries using Fernet symmetric encryption.	Jenil Patel, Krishan Kumar
Testing & Documentation	Validates system functionality and ensures proper documentation.	Mallika Chouhan, Aniket Asati

Security

Security is at the core of this system, with a strong focus on ensuring both the privacy and integrity of the logs. The main security measures employed in the system are:

Encryption:

- **Fernet Symmetric Encryption** is used to encrypt log entries before they are written to the log file. Fernet guarantees confidentiality and integrity by using AES in CBC mode with a secure key.
- Only the authorized user, who possesses the correct secret key, can decrypt and access the log entries.

Authentication:

- Each log operation requires an **authentication token/password**. This token is stored encrypted within the log file and is verified against the user-provided token.
- Unauthorized users will be blocked from accessing or modifying the log.

Log Integrity:

- All log entries are encrypted before being appended to the log file, ensuring that the log cannot be tampered with or read without proper decryption.
- The system enforces strict validation rules on timestamps and events to ensure the consistency of the log state.

Secure Communication:

- While the log file is stored locally, all operations (append and read) are performed under encryption. The log file cannot be interpreted by users who do not have the secret key.

Testing and Validation

Extensive testing was performed to ensure that the system meets its functional and security requirements. Testing focused on:

1. **Correctness of Log Operations:** Verifying that all log entries are appended and retrieved correctly after encryption and decryption.
2. **Security Testing:** Ensuring that unauthorized users cannot access the log without the correct key.
3. **Edge Cases:** Testing cases such as log file corruption, invalid timestamps, or unauthorized access attempts.

References

- **Cryptographic Best Practices:** https://cheatsheetseries.owasp.org/cheatsheets/Cryptographic_Storage_Cheat_Sheet.html

Conclusion

The **Security Guard at IITGN** system is a robust, secure logging solution designed for tracking campus activities. Through the use of encryption and authentication, it guarantees the confidentiality and integrity of log data. The design has evolved to use simple text-based storage combined with strong encryption to meet the project's security goals. Testing has confirmed the system's reliability, and all functionalities have been implemented as specified.