A Reliable Distributed - Cloud Storage Based on Permissioned Blockchain

Ilanthiraiyan. A[1], Madhan Kumar V. S.[2], Manikandan N.[3], Jayaram P.[4], Kavinmathi B.[5]
[1,2,3,4]*UG student, Dept. of CSE, Tamilnadu College of Engineering, Coimbatore, Tamilnadu.*
[5]*Assistant Professor, Dept. of IT, Tamilnadu College of Engineering, Coimbatore, Tamilnadu.*

***Abstract*—In an era of rapid technological advancement and increasing reliance on interconnected systems, the security of critical infrastructure and sensitive data has become a paramount concern. This paper introduces a novel approach to safeguarding control systems and their communications through the application of Keyed-Homomorphic Public Key Encryption. Traditional encryption techniques provide confidentiality but often fall short in ensuring data integrity, authenticity, and efficient processing for control systems. The proposed method incorporates Keyed- Homomorphic Public Key Encryption, an innovative cryptographic approach that combines the merits of homomorphic encryption and public key cryptography. This approach enables data to be securely processed in an encrypted form while preserving its integrity and authenticity.**

***Index Terms*—Blockchain, Data integrity verification, Merkle tree, Honeypot, Cybersecurity, Keyed-Homomorphic Public Key Encryption, Control Systems, Data Security, Key Management, Encryption, Authentication.**

I. INTRODUCTION

Cloud computing has discovered their wide applications in IT area social organizations and money related exchanges. To achieve the focused- on application and satisfy its functionalities, a Cloud generally stores a lot of information persistently over its lifetime. Information storing and access in Cloud networks essentially follows two methodologies, to be specific, distributed and centralized approaches. Distributed data storage avoids single point of failure, consume less bandwidth etc., as compared to centralized one. Cloud computing is a paradigm that provides massive computation capacity and huge memory space at a low cost. It enables users to get intended services irrespective of time and location across multiple platforms (e.g., mobile devices, personal computers), and thus brings great convenience to cloud users. Among numerous services provided by cloud computing, cloud storage service, such as Apple's iCloud, Microsoft's Azure and Amazon's S3 However, it also suffers from several security threats, which are the primary concerns of cloud users.

II. PROBLEM STATEMENT

A problem statement is important because it examines an issue from multiple angles. It identifies who the problem impacts, what the impacts are, where the problem occurs and why and when it needs to be fixed. Existing Systems or resources which are utilized to just snare, they are not screen and this framework isn't distinguish mistaken demands present inside an organization. They will not change in the cooperation give to the aggressors, from low collaboration to medium and high. Their framework isn't break down, comprehend, watch and track aggressor's way of behaving to make frameworks that are secure as well as handle such traffic. It's anything but a firmly observed registering asset that we need to be examined, went after, or split the difference. Thus, a problem statement ensures individuals investigate the entire scope of an issue and approach the solution with clear goals in mind.

III. EXISTING SYSTEM

In existing system that precisely matches the description of a "Cybersecurity- Enhanced Encrypted Control System Using Keyed-Homomorphic Public Key Encryption". This appears to be a specialized and

potentially research- oriented concept that may not have a direct counterpart in existing, commercial systems. However, components or technologies related to enhancing cybersecurity and encryption in control systems include elements like Public Key Infrastructure (PKI), secure SCADA systems, secure communication protocols, intrusion detection and prevention systems (IDPS), and key management solutions. Please keep in mind that developments in the field of cybersecurity and encryption are ongoing, and specialized systems may have emerged since my last update. For the most up-to-date information, consider reviewing recent research papers, consulting experts in the field, or conducting a targeted search for systems and solutions related to Keyed-Homomorphic Public Key Encryption in control systems.

IV. WORKFLOW

1. Data Provider logs in and uploads files.
2. Files are encrypted using AES and stored in the cloud server.
3. Data User logs in, views files, and sends a request.
4. Authority verifies the request, re-encrypts data, and provides a key with time limit.
5. User downloads and decrypts the file using the key.
6. Honeypot system monitors activity to detect suspicious behavior.
7. If an attacker is detected, the system blocks access and prevents download.

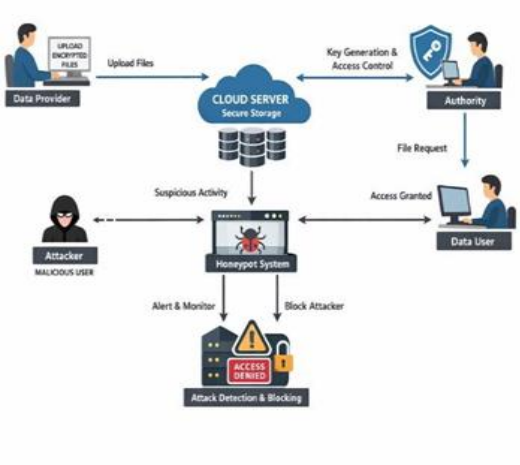

Fig 1: Work Flow

V. PROPOSED SYSTEM

In this we have proposed an approach to safeguarding control systems and their communications through the application of Keyed-Homomorphic Public Key Encryption. Traditional encryption techniques provide confidentiality but often fall short in ensuring data integrity, authenticity, and efficient processing for control systems. The proposed method incorporates Keyed- Homomorphic Public Key Encryption, an innovative cryptographic approach that combines the merits of homomorphic encryption and public key cryptography. This approach enables data to be securely processed in an encrypted form while preserving its integrity and authenticity.

VI. SYSTEM ARCHITECTURE

6.1 Overview

The proposed system architecture is designed to provide secure cloud-based data storage and attack monitoring using encryption techniques and a honeypot mechanism. The system ensures data confidentiality, integrity, and access control, while simultaneously detecting malicious activities through behavioral monitoring.

6.2 Architecture Components

1) Data Provider

The Data Provider is responsible for:

- Registering and authenticating into the system
- Uploading files to the cloud
- Encrypting files using AES algorithm before storage

2) Data User

The Data User interacts with the system by:

- Registering and logging into the system
- Requesting access to encrypted files
- Downloading and decrypting files using the provided key

3) Authority

The Authority acts as a trusted entity responsible for:

- Generating and distributing secure keys
- Re-encrypting data when necessary
- Managing access control policies

4) Cloud Server

The Cloud Server is responsible for:

- Storing encrypted data securely
- Managing communication between system components

5) Honeypot System

The Honeypot System is used for:

- Monitoring user behavior and system activities
- Detecting suspicious or malicious actions

6) Attacker (External Entity)

The Attacker represents an unauthorized user who:

- Attempts to access data using fake credentials
- Sends malicious requests to the system

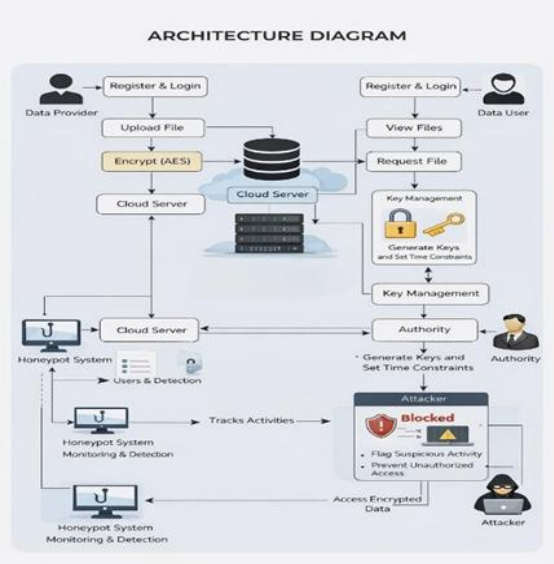

Fig 2: System Architecture

VII. MODULE DESCRIPTION

The proposed system is divided into several functional modules to ensure secure data storage, controlled access, and effective attack detection. Each module performs a specific role in maintaining the integrity and security of the system.

7.1 Data Provider Module

The Data Provider module is responsible for managing data upload and encryption processes. Initially, the data provider registers and logs into the system using valid credentials. After successful authentication, the provider can upload files to the cloud server.

7.2 Data User Module

The Data User module enables users to access and download files from the cloud. The user must first register and log into the system. Once authenticated, the user can browse available files and send a request to access specific data. Upon approval from the authority, the user receives the decryption key and can download and decrypt the file securely.

7.3 Authority Module

The Authority module acts as a trusted third party responsible for access control and key management. It verifies user requests and ensures that only authorized users can access the requested data.

7.4 Honeypot Module

The Honeypot module is designed to detect and analyze malicious activities within the system. It acts as a decoy environment to attract attackers and study their behavior.

7.5 Attacker Module (External Entity)

The Attacker module represents unauthorized users attempting to gain access to the system. These attackers may use fake credentials or perform malicious activities such as unauthorized file access. The system identifies such activities through the honeypot mechanism and prevents further access by blocking the attacker.

Fig 3: Use Case Diagram.

VIII. IMPLEMENTATION

8.1 Overview

The proposed system is implemented as a secure cloud-based application that integrates encryption techniques with a honeypot-based attack detection mechanism. The implementation focuses on ensuring data confidentiality, controlled access, and real- time monitoring of malicious activities.

8.2 Development Environment

The implementation environment consists of the following technologies:

- Front-End: HTML, CSS, JavaScript
- Back-End: Java (JSP and Servlets)
- Database: MySQL
- IDE: Eclipse
- Server: Apache Tomcat

8.3 System Implementation Modules

1) Data Provider Implementation

The Data Provider module is implemented using JSP-based web interfaces. The provider can register, log in, and upload files through a secure interface.

- Files are encrypted using the AES algorithm before being stored in the database.
- Encryption ensures that data remains secure even if unauthorized access occurs.

2) Data User Implementation

The Data User module allows users to:

- Register and log in securely
- View available files
- Send requests to access files

3) Authority Implementation

The Authority module is implemented as a control interface that:

- Verifies user access requests
- Generates secure keys
- Manages access permissions

4) Cloud Server Implementation

The Cloud Server module is implemented using:

- JSP pages for interface
- JDBC for database connectivity

5) Honeypot Implementation

The Honeypot module is implemented as a monitoring system that:

- Tracks user behavior
- Identifies suspicious patterns
- Logs malicious activities

8.4 Database Design

The system uses a MySQL database with the following key tables:

- User Table: Stores user credentials and details
- File Table: Stores file metadata and encryption keys
- Request Table: Tracks file access requests
- Log Table: Records user activities and detected attacks

The database ensures efficient data management and supports secure operations.

8.5 Security Implementation

The system incorporates multiple security mechanisms:

- AES Encryption: Protects file data
- SHA-256 Hashing: Ensures data integrity
- Authentication: Secure login system
- Access Control: Authority-based permissions
- Honeypot Detection: Identifies and blocks attackers

These mechanisms work together to provide a multi-layered security architecture.

8.6 Workflow Execution

- Data Provider uploads encrypted files to the cloud
- Cloud Server stores the encrypted data
- Data User requests file access
- Authority verifies request and generates key
- User downloads and decrypts file
- Honeypot monitors all activities
- Attacker is detected and blocked if malicious behavior occurs

8.7 Implementation Outcome

The system successfully:

- Secures cloud-stored data using encryption
- Controls access through authority-based permissions
- Detects and prevents unauthorized access using honeypot
- Provides a scalable and efficient cloud- based solution

IX. SYSTEM TESTING

System testing is a critical phase in the software development lifecycle, aimed at validating the complete and integrated system against specified requirements. The proposed system is tested to ensure that all modules function correctly, securely, and efficiently under various conditions. The testing process verifies that the system meets both functional and non-functional requirements, including security,

performance, and reliability. Special emphasis is placed on validating the encryption mechanisms and honeypot-based attack detection system.

9.1 Testing Methodology
The system follows a structured testing approach based on the Waterfall Model, where testing is conducted at different stages of development. The methodology includes:
Designing test cases based on system requirements
Executing tests for each module independently

9.2 Types of Testing
Unit testing is performed on individual modules such as Data Provider, Data User, Authority, Cloud Server, and Honeypot. Each unit is tested to ensure that it performs its intended function correctly.

9.3 Integration Testing
Integration testing verifies the interaction between different modules. It ensures that data flows correctly across the system components.

9.4 Functional Testing
Functional testing ensures that the system performs all intended operations as specified.

X. SECURITY MECHANISMS

Security is a fundamental requirement in cloud-based systems, especially when handling sensitive data and preventing unauthorized access. The proposed system incorporates a multi-layered security architecture that combines encryption, authentication, access control, and attack detection mechanisms. These mechanisms ensure data confidentiality, integrity, availability, and protection against malicious activities within the cloud environment.

10.1 Data Encryption Mechanism
The system employs the Advanced Encryption Standard (AES) to secure data before it is stored in the cloud.

10.2 Hashing and Data Integrity
To ensure data integrity, the system utilizes the SHA-256 hashing algorithm.

Benefits:
- Detects unauthorized data modifications
- Maintains trustworthiness of stored data

10.3 Authentication Mechanism
The system implements a secure authentication process for all users.
- Users must register and log in using valid credentials
- Credentials are verified before granting access
- Prevents unauthorized users from accessing the system

Features:
- Username and password validation
- Session management
- Secure login process

XI. SYSTEM FEATURES

The proposed system incorporates a set of advanced features designed to ensure secure data storage, efficient access control, and real-time attack monitoring in a cloud environment. These features collectively enhance the system's reliability, scalability, and security.

11.1 Key Features
1) Secure Data Encryption
The system uses the Advanced Encryption Standard (AES) to encrypt files before storing them in the cloud.
- Ensures data confidentiality
- Protects sensitive information from unauthorized access
- Supports efficient encryption and decryption

2) User Authentication and Authorization
The system provides a robust authentication mechanism.
- Secure user registration and login
- Verification of user credentials
- Role-based access control (Data Provider, Data User, Authority)

This feature ensures that only legitimate users can access system resources.

3) Controlled Data Access
Access to files is strictly controlled by the Authority module.
- Users must request permission to access files

- Authority verifies requests and provides secure keys
- Supports time-based access restrictions

4) Cloud-Based Storage System

The system utilizes a centralized cloud server for data storage.

- Stores only encrypted data
- Provides scalable and flexible storage
- Enables remote access from multiple locations

5) Honeypot-Based Attack Detection

A key feature of the system is the integration of a honeypot mechanism.

- Monitors user behavior continuously
- Detects suspicious and malicious activities

XII. RESULTS AND DISCUSSION

The proposed system was evaluated to assess its security, performance, and attack detection capability. The results confirm the effectiveness of combining encryption with honeypot-based monitoring.

1) Data Security: Files were successfully encrypted using AES and securely stored in the cloud.
2) Access Control: Only authorized users could access and decrypt files through the Authority module.
3) Performance: The system showed efficient response time for upload and download operations.
4) Attack Detection: The honeypot effectively identified and blocked malicious users.
5) Data Integrity: SHA-256 ensured detection of any data modification.

The system provides a secure and reliable cloud solution by preventing unauthorized access and detecting malicious behavior. The integration of encryption and honeypot significantly enhances overall security compared to existing systems. The results confirm the effectiveness of combining encryption with honeypot- based monitoring.

Fig 4: Screenshot of home page

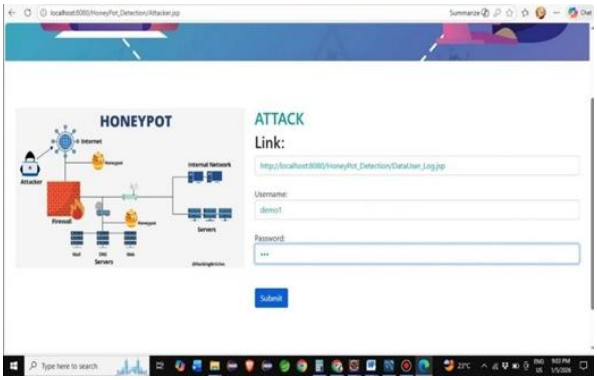

Fig 5: Screenshot of Attacker page

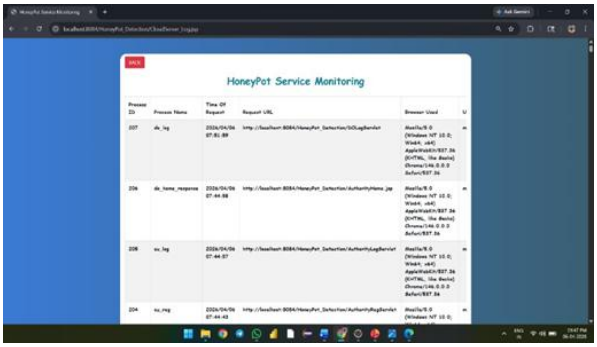

Fig 6: Screenshot of Honeypot Server Upload page

XIII. FEARURE ENHANCEMENTS

The proposed system provides a secure framework for cloud data storage and attack detection. However, with the rapid evolution of cloud technologies and cybersecurity threats, several enhancements can be incorporated to further improve the system's security, scalability, and performance. This section outlines potential future improvements that can extend the capabilities of the system.

XIV. CONCLUSION

A protected certain semantic looking through plan that treats coordinating among questions and archives as a word transportation ideal coordinating undertaking. Accordingly, we research the crucial hypotheses of straight programming to plan the word transportation issue and a result check system.

REFERENCES

[1] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proc. IEEE Symp. Security and Privacy, 2000, pp. 44–55.

[2] Z. Fu, J. Shu, X. Sun, and N. Linge, "Smart cloud search services: Verifiable keyword-based semantic search over encrypted cloud data," IEEE Trans. Consumer Electronics, vol. 60, no. 4, pp. 762–770, 2014.

[3] Z. J. Fu, X. M. Sun, N. Linge, and L. Zhou, "Achieving effective cloud search services: Multi-keyword ranked search over encrypted cloud data supporting synonym query," IEEE Trans. Consumer Electronics, vol. 60, no. 1, pp. 164–172, 2014.

[4] T. S. Moh and K. H. Ho, "Efficient semantic search over encrypted data in cloud computing," in Proc. IEEE Int. Conf. High Performance Computing and Simulation, 2014, pp. 382–390.

[5] N. Jadhav, J. Nikam, and S. Bahekar, "Semantic search supporting similarity ranking over encrypted private cloud data," Int. J. Emerging Engineering Research and Technology, vol. 2, no. 7, pp. 215–219, 2014.

[6] Z. H. Xia, Y. L. Zhu, X. M. Sun, and L. H. Chen, "Secure semantic expansion-based search over encrypted cloud data supporting similarity ranking," J. Cloud Computing, vol. 3, no. 1, pp. 1–11, 2014.

[7] Z. Fu, L. Xia, X. Sun, A. X. Liu, and G. Xie, "Semantic-aware searching over encrypted data for cloud computing," IEEE Trans. Information Forensics and Security, vol. 13, no. 9, pp. 2359–2371, Sep. 2018.

[8] Z. J. Fu, X. L. Wu, Q. Wang, and K. Ren, "Enabling central keyword-based semantic extension search over encrypted outsourced data," IEEE Trans. Information Forensics and Security, vol. 12, no. 12, pp. 2986–2997, 2017.

[9] Y. G. Liu and Z. J. Fu, "Secure search service based on Word2Vec in the public cloud," Int. J. Computer Science and Engineering, vol. 18, no. 3, pp. 305–313, 2019.

[10] E. J. Goh, "Secure indexes," IACR Cryptology e-Print Archive, vol. 2003, pp. 216–234, 2003.

[11] "Algorithms, Equity, and Bias," AI Now Institute, 2018.