FORTIBLOCK: HYBRID BLOCKCHAIN FRAMEWORK FOR SECURE ACCESS CONTROL IN CLOUD ENVIRONMENT

Project Reference No.: *48S_BE_6210*

College : Bangalore Institute Of Technology, Bengaluru
Branch : Department Of Computer Science And Engineering
Guide : Dr. Bhanushree K. J
Student(S): Mr. Sunil R
Mr. Shivakumar
Mr. Surya Bharadwaj B S
Mr. Rohan A Murari

Keywords

Smart Contracts, Decentralization, Direct and Indirect Access, Authorization, Revocation, Blockchain, Cloud.

INTRODUCTION

With the emergence of a computing model, cloud computing technology offers ubiquitous services to users. The cloud option reduces costs associated with user storage and computing, making it more convenient; consequently, an ever-growing number of businesses and individuals opt to store data in the cloud. As cloud computing continues to scale and become more complex, research into computing and edge computing has emerged slowly. Cloud security concerns are increasingly posing a significant obstacle to the advancement of computing technology. In July 2017, the Cloud Security Alliance (CSA) issued a guidance document for cloud computing v4.0, which pinpointed Fourteen cloud security areas, with access control being a fundamental technology in cloud security. Access control is a subject of ongoing research and is primarily designed to prevent unauthorized access to and theft of resources stored in cloud, service models comprise infrastructure, platform, and software as a service. The three processes must safeguard relevant resources by implementing access control measures. Access control is a crucial component in cloud environments.

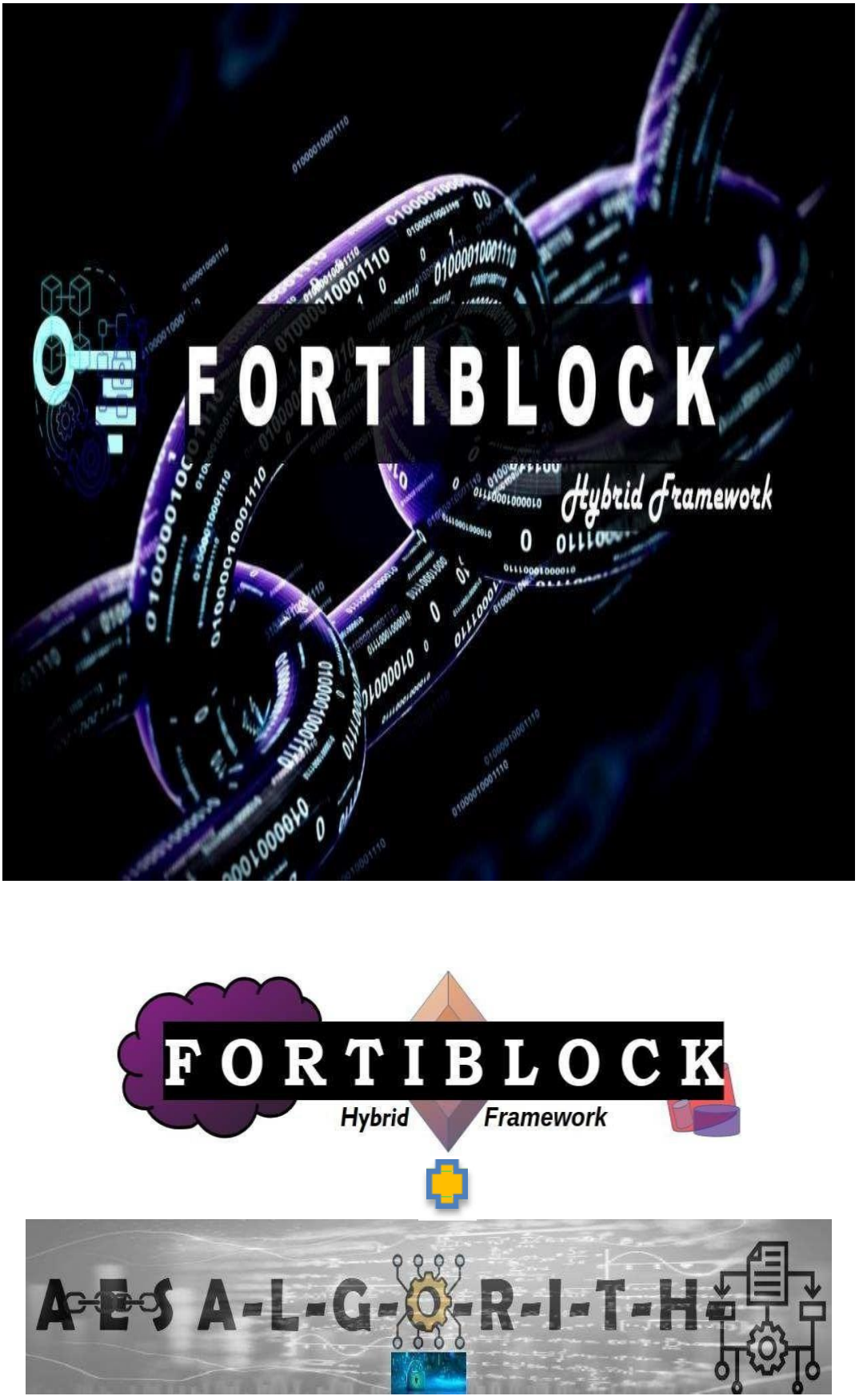

Figure 1: FORTIBLOCK.

- **Purpose**

  - To provide processes of access control, authorization, and access revocation.
  - To secure the data access control from insider and outsider attacks.

- **Scope**

  Access control authenticates users by validating different types of login credentials, such as passwords and usernames, PIN codes, biometric verification, or security tokens. Many access control systems also include multi factor authentication (MFA), a method that requires multiple authentication methods to verify a user's identity.

- **Applicability**

  - The Major aim is to provide a security framework that ensures the data security
  - Providing open-source and efficient access privileges.
  - Important aspect is to reduce the scalability cost. So, every small
  - Industries, organizations, Educational Institutes, etc., are able to access facility.
  - A Decentralized RBCA and Cryptography enabled access control framework.

- **Objectives**

  - Implement a secure web interface for cloud data access management.
  - Enable role-based access control for cloud data users.
  - Detect and prevent unauthorized data access from both internal and external sources.
  - Use blockchain to record and validate access transactions.
  - Ensure data integrity through cryptographic techniques and immutable records.
  - Provide real-time monitoring of access activities in the cloud.
  - Minimize latency in access control using efficient algorithms.
  - Incorporate smart contract-based authorization for automated access decisions.
  - Ensure interoperability with existing cloud services and platforms.

METHODOLOGY

- Architecture

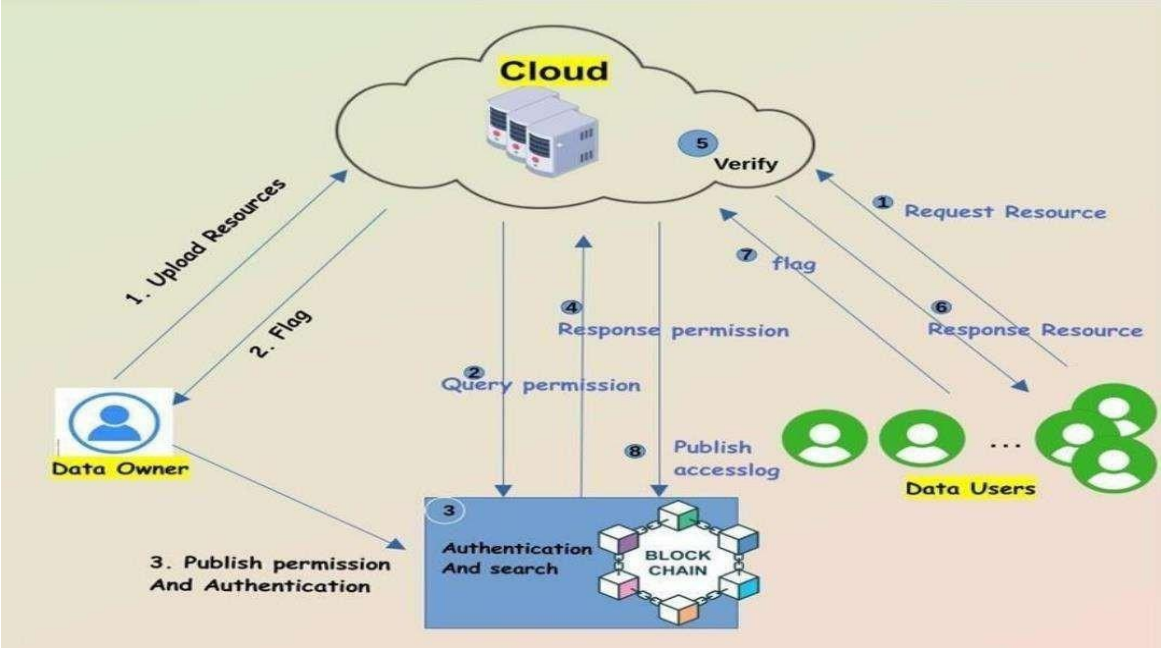

Figure 2: System Architecture.

Figure 2, the system architecture defines the overall structure of the system, including its components, their interactions, and the data flow between them. In this system, we have identified the following components:

Step 1 - **Upload Resources:** The Data Owner (DO) uploads their files or resources to the cloud storage. This is where data is initially stored for further access.

Step 2 - **Flag:** Once the resources are uploaded to the cloud, the cloud verifies them. If there are any issues with the uploaded data (e.g., invalid formats, permissions, etc.), it flags the resource. This flagging mechanism helps in identifying faulty or unauthorized uploads early.

Step 3 - **Publish Permission and Authorization:** After successfully uploading resources, the DO publishes the access permissions and authorizations on the blockchain. These permissions define who can access the data and under what conditions. The blockchain ensures decentralized and tamper-proof records of these access controls.

Step 4 - Request Resource: The Data Users (DU) request access to a resource from the cloud. This step represents users trying to access or download the file uploaded by the DO.

Step 5 - **Flag:** Like step 2, if there's an issue with the user's request (e.g., invalid user credentials or permission issues), the cloud flags the request, preventing unauthorized access.

Step 6 - **Response Resource:** If the request is valid and authorized, the cloud responds by providing access to the requested resource, delivering the file or data to the Data User (DU).

Step 7 - **Query Permission:** Before providing access, the cloud queries the blockchain to verify the permissions. This ensures that only users with the correct authorization (as defined in Step 3) are allowed access to the resource.

Step 8 - Response **Permission:** The blockchain responds to the cloud's query by confirming or denying access based on the permissions and authorizations previously published by the Data Owner. If valid, the blockchain allows the cloud to proceed with the resource request.

Step 9 - **Publish** Access **Log:** Once the file is accessed or denied, the blockchain records this action by publishing an access log. This log keeps track of all access requests, including who accessed what data and when, providing a transparent and secure record of data usage.

Algorithm Design

AESEncryption Algorithm Process

AES Algorithm Design the Advanced Encryption Standard (AES) algorithm is a symmetric encryption algorithm that is widely used for securing data. In the FORTIBLOCK system, AES is used for encrypting the user credentials and access policies, to protect the user's privacy and ensure the security of the system. Here's an overview of the AES algorithm design:

Input Data: The input data for the AES algorithm includes the plaintext message to be encrypted, the encryption key, and the block size (128, 192, or 256 bits).

Output Data: The output data for the AES algorithm is the encrypted ciphertext. Logic of Processes: The AES algorithm consists of several processes that are repeated in rounds. The number of rounds depends on the block size and the key size. Here's a summary of the processes involved:

Key Espansion: The encryption key is expanded into a set of round keys, one for each round of encryption.

Initial Round: The plaintext message is XORed with thefirst-round key.

Rounds: The plaintext and round key are processed through a series of substitution, permutation, and mixing operations, which are repeated for each round of encryption. **Final Round:** The last round of encryption is a modified version of the previous rounds, and includes a final substitution operation.
Output: The resulting ciphertext is the output of the AES encryption algorithm.

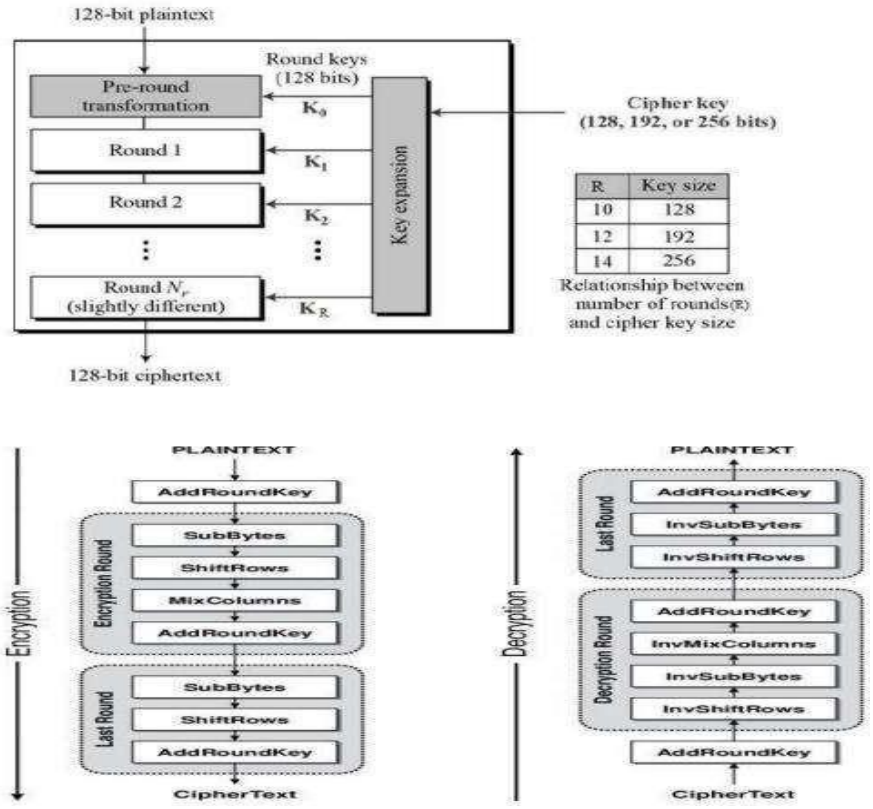

Figure 3: AESEncryption Process.

Figure 3, the design of the AES algorithm utilizes complex mathematical operations, such as substitution, permutation, and mixing, to transform the plaintext into ciphertext. These operations are designed to be resistant to various attacks, such as

differential and linear cryptanalysis, and to ensure the security of the encrypted data. The working of the AES algorithm involves the use of the encryption key to transform the plaintext message is broken up into a sequence of encrypted blocks. The key is expanded into a block of round keys, used to perform the multiple operations in the round. The resulting ciphertext is transmitted or stored, and can only be decrypted using the proper key.

In the FORTIBLOCK system, the use of the AES algorithm aims to guarantee the Security and Privacy of User Data. By encrypting the user credentials and access policies with AES, the systemcan protectuserprivacy and be surethatalldataare kept secure.

SHA-256/512 for Credential Protection (Blockchain): Algorithm

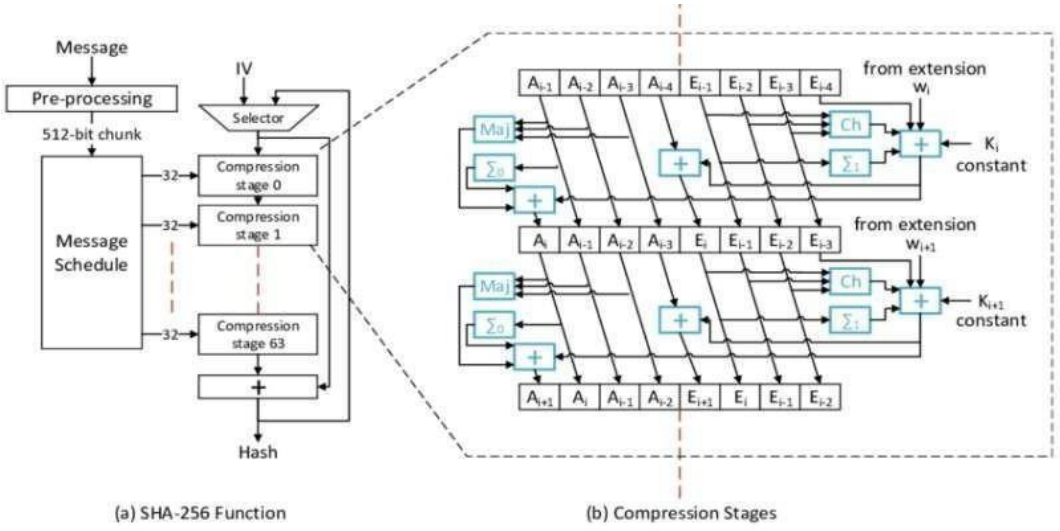

Figure 4: SHA — 256 Encryption Algorithm.

Figure 4, Ethereum bloekchain is used to implement decentralized access control and store user credentials securely. Smart contracts, written in Solidity, are used to handle user registration and authentication. The SHA-256/512 hashing algorithms ensure that user credentials (like passwords) are stored in an encrypted form, making it difficult for unauthorized access.

Ethereum Implements and Stores User Credentials:

User Registration: When a user registers, their password is hashed using SHA- 256/512 before it is stored on the blockchain.

Smart Contracts: Smart contracts store these hashed credentials and manage

access control functions (e.g., login, access permissions, revocation).

Security: SHA-256/512 generates a unique, irreversible hash of the password, ensuring that even if someone gains access to the blockchain, they cannot retrieve the original password. Authentication: During login, the entered credentials are hashed again, and the hashed value is compared with the stored hash. If the two hashes match, the user is authenticated.
Input: User credentials (username/password)
Output: Hashed credentials stored on the
blockchain
Step 1: Take User Credentials: Collect username and password.

Step 2: Apply SHA-256/512: Hash the credentials using SHA-256 or SHA-

512. Step 3: Generate Hash: Obtain a unique, irreversible hash of the

credentials.

Step 4: Create Smart Contract: Implement a smart contract to handle storing and comparing hashes.
Step 5: Store Hash on Blockchain: Store the hashed credentials immutably on the blockchain. Step 6: Verify on Login: When a user logs in, hash the entered credentials and compare them with the stored hash.

Step 7: Grant/Reject Access: If the hashes match, access is granted; otherwise, it's denied.
Step 8: Log Transaction: Log the access attempt (successful or failed) on the blockchain.

RESULTS

Figure 5, the cloud and specify the access control for different users. The entire process is saved in the blockchain as a secured hash key (Trans_id) is stored, making it decentralized for authorized users with access permission.

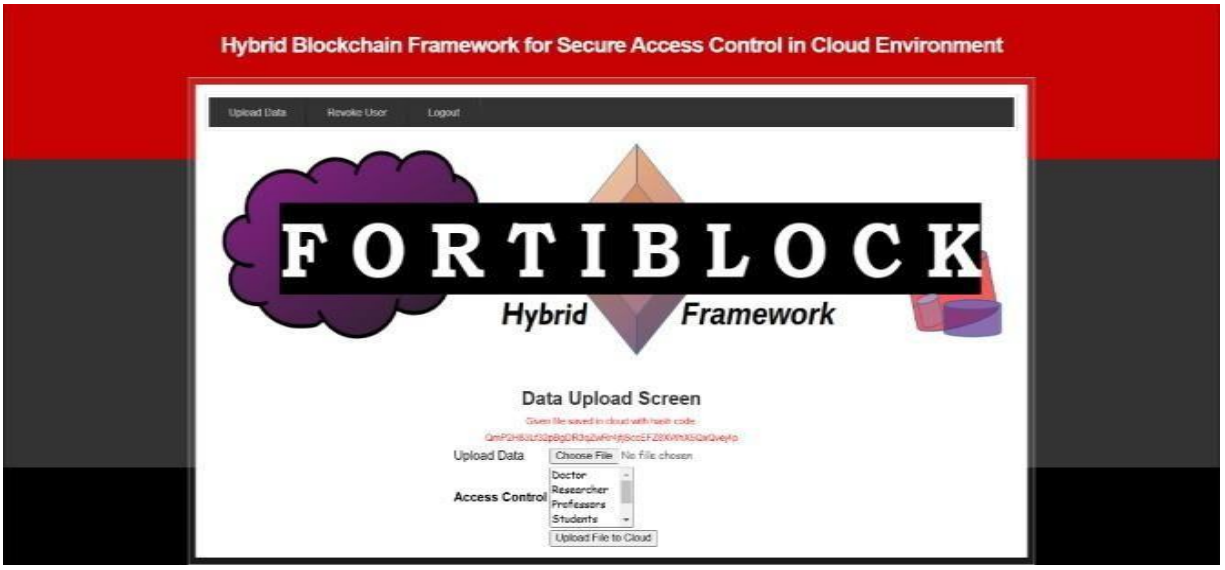

Figure 5: DataOwner upload the data.

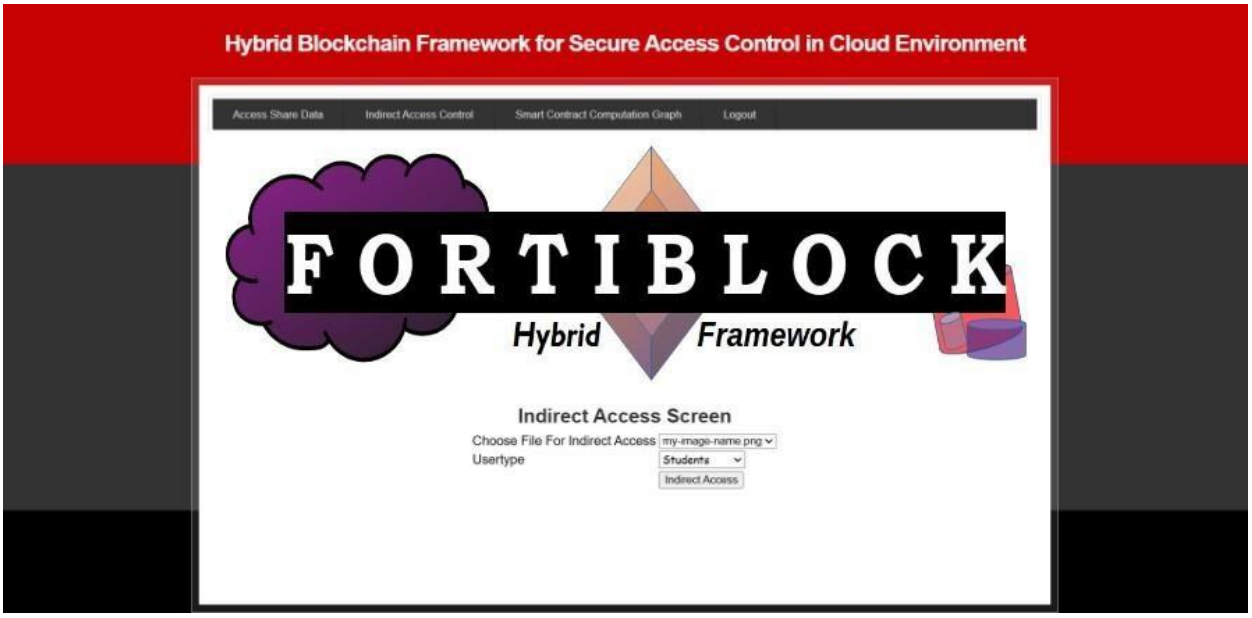

Figure 6: Indirect Access.

Figure 6, is also secured in the blockchain; not every random user can access the data; only verified users are always identified and then are allowed to access the file, This user interface allows a data user to select the users or categories to whom they should be given indirect access. Through indirect access, enabling of controlled sharing and collaboration is enhanced as other authorized user ability to access shared resources is determined by the permissions provided by the data user.

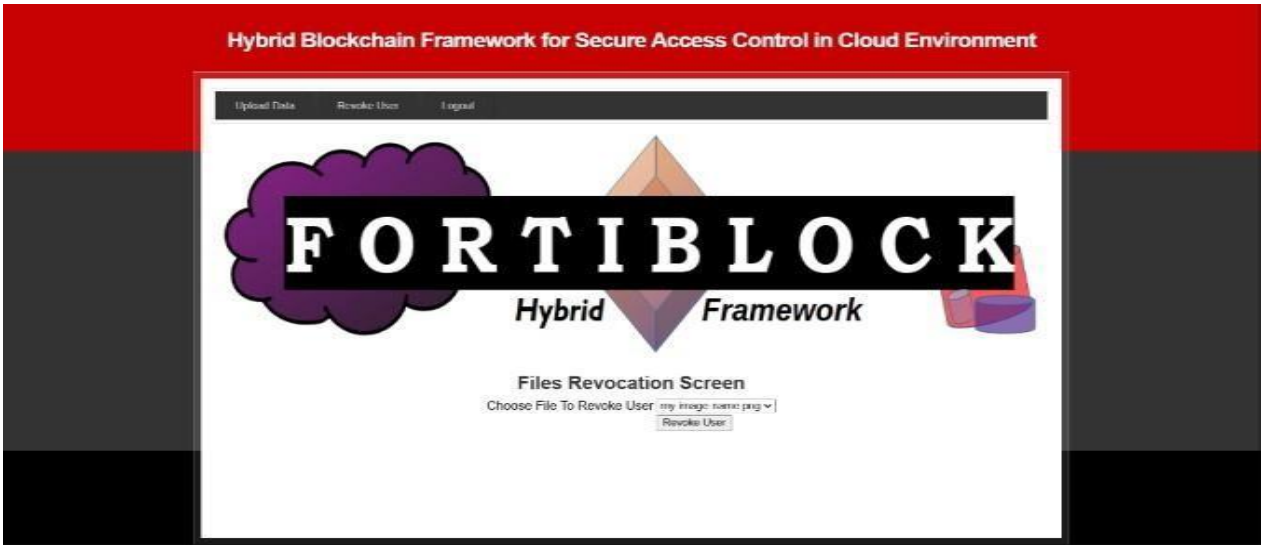

Figure 7: File Revocation Access.

Figure 7, a unique property granted to the Data owner to withdraw file access permissions, with the revoke action functioning similarly to **ON DELETE CASCADE**, where it erases existing file access permissions and revokes any associated indirect access.

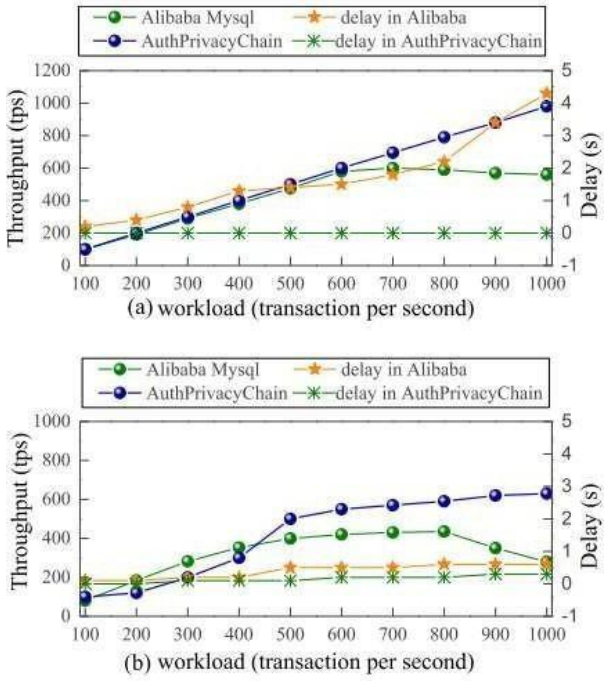

Figure 8: Comparison of throughput and delay, Traditional and AuthPrivacyChain access control performance comparison.

Finally, the overhead of access control needs to be analysis too. We compare FORTIBLOCK to the traditional cloud access control platform. FORTIBLOCK uses Kylin's node Api- kylin.eos1aomao.com. Assume that the user has logged in to the

corresponding device, ignoring the interference as depicted, only the time of access control is considered, accounting for both network factors and the time of user input.

CONCLUSION

Traditional cloud and blockchain-based access control systems face significant security and privacy challenges, often relying on centralized authorities that introduce vulnerabilities to both internal and external threats. To address these issues, we propose FORTIBLOCK, a hybrid blockchain framework designed to enhance privacy-focused access control in cloud environments. This framework leverages public and private blockchains, ensuring transparency while securing sensitive access control operations.

In FORTIBLOCK, every access-related transaction is immutably recorded on the blockchain, preventing unauthorized modifications and ensuring accountability. User authentication is secured using SHA-256 hashing, while AES encryption protects data before cloud storage, ensuring confidentiality and integrity. The framework implements Role-Based Access Control (RBAC) through smart contracts, automating permission grants, revocations, and access monitoring.

Experimental results demonstrate that only authorized users can access the resources, effectively preventing unauthorized access. FORTIBLOCK upholds key security principles, including confidentiality, integrity, availability, authenticity, and accountability, while mitigating both external attacks and insider threats. By integrating decentralized trust, cryptographic security, and transparent auditing, FORTIBLOCK establishes a scalable, privacy- preserving, and secure cloud access control system, making it a robust solution for modern cloud-based applications.

FUTURE SCOPE OF WORK

1. Multi-Cloud Integration

- Enable seamless access control across different cloud platforms (AWS, Azure, Google Cloud).
- Implement inter-cloud data synchronization while maintaining security and privacy.

2. AI-Powered Access Control

- Use machine learning for real-time anomaly detection and dynamic permission adjustments.

- Develop behavior-based access control models to enhance security automation.

3. Quantum-Resistant Cryptography

- Upgrade encryption algorithms to post-quantum cryptographic standards.
- Implement lattice-based and hash-based cryptography for long-term data protection.

4. Cross-Blockchain Interoperability

- Integrate with Hyperledger, Binance Smart Chain, or Polkadot for multi-blockchain compatibility.
- Facilitate secure cross-chain data sharing without compromising privacy.

5. Zero-Knowledge Proof (ZKP) for Enhanced Privacy

- Enable identity verification without revealing user credentials.
- Implement ZKP-based permission validation to enhance confidentiality in access control.