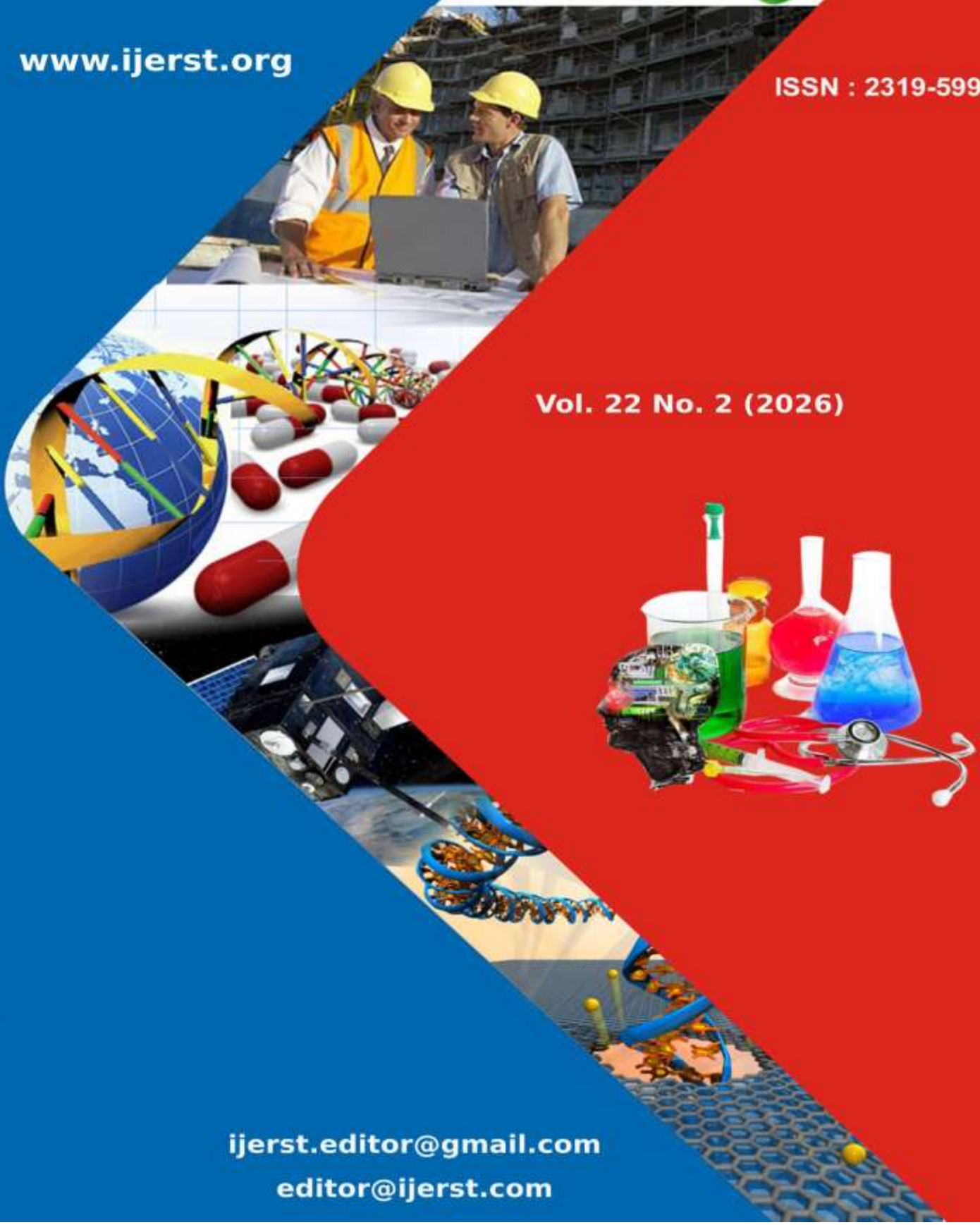
International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 2 (2026)



ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 22, No. 2, 2026
© 2026 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

SECURE CLOUD DATA DEDUPLICATION WITH EFFICIENT RE ENCRYPTION

BORRA TEJA SRI [1], S K ALISHA [2]

1.Student, Dept. of Computer Applications, B. V. Raju College, Bhimavaram. Garagaparru Road, Kovvada, Andhra Pradesh
2.Assistant Professor, Dept. of Computer Applications, B. V. Raju College, Bhimavaram. Garagaparru Road, Kovvada, Andhra Pradesh.

ABSTARCT

Data deduplication technique has been widely adopted by commercial cloud storage providers, which is both important and necessary in coping with the explosive growth of data. To further protect the security of users' sensitive data in the outsourced storage mode, many secure data deduplication schemes have been designed and applied in various scenarios. Among these schemes, secure and efficient re-encryption for encrypted data deduplication attracted the attention of many scholars, and many solutions have been designed to support dynamic ownership management. In this paper, we focus on the re-encryption deduplication storage system and show that the recently designed lightweight rekeying-aware encrypted deduplication scheme (REED) is vulnerable to an attack which we call it stub-reserved attack. Furthermore, we propose a secure data deduplication scheme with efficient re-encryption based on the convergent all-or-nothing transform (CAONT) and randomly sampled bits from the Bloom filter. Due to the intrinsic property of one-way hash function, our scheme can resist the stub- reserved attack and guarantee the data privacy of data owners' sensitive data. Moreover, instead of re-encrypting the entire package, data owners are only required to re-encrypt a small part of it through the CAONT, thereby effectively reducing the computation overhead of the system. Finally, security analysis and experimental results show that our scheme is secure and efficient in re-encryption**.** The rapid growth of cloud storage services has led to an explosion of data redundancy, resulting in excessive storage costs and bandwidth consumption. To address this, data deduplication has emerged as a critical technique to eliminate redundant copies of data by storing only a single unique instance. However, traditional deduplication often conflicts with data encryption, as standard encryption produces different ciphertexts for the same plaintext, making duplicate identification impossible. This project proposes a Secure Cloud Deduplication system with Efficient Re- encryption to bridge the gap between storage efficiency and data confidentiality. The system utilizes Convergent Encryption (CE), where encryption keys are derived from the cryptographic hash of the data content, ensuring that identical files result in identical ciphertexts. To further enhance security, a Proof of Ownership (PoW) protocol is implemented to prevent unauthorized users from the system.

1. INTRODUCTION

WITH the rapid development of cloud storage, more and more individuals and enterprises tend to outsource their sensitive data to remote cloud service providers in a pay-per-use manner [1], [2], [3], [4], [5]. According to the study from Internet Data Center (IDC) sponsored by Dell EMC, the digital universe is doubling in size every two years and the volume of the universe data is expected to reach 44 zettabytes (ZB) or 44 trillion gigabytes (GB) in 2020 (more than 5200 gigabytes for each man, woman, and child) [6]. Recently, Li et al. [25], [26] proposed a rekeying-

aware encrypted deduplication storage system (REED), which supports a lightweight re-encryption. Instead of re-encrypting the entire package, data owners are only required to reencrypt a small part of it through the CAONT, thereby effectively reducing the computation overhead of the system. However, we point that the REED is vulnerable to the stub-reserved attack, which will be described in Section 3.2. t. Therefore, existing proposed schemes cannot well support secure dynamic ownership management of cloud users and efficient re encryption.

Our Contribution. In this paper, we further study the above problems of secure and efficient re- encryption for de duplication storage. Our contributions are three folds:

- We point out a security weakness of the enhanced encryption of REED scheme [25], [26]. That is, this scheme is vulnerable to the so-called stub-reserved attack proposed in this paper.
- We propose a location selection method based on Bloom filter and a secure data deduplication scheme with efficient re-encryption. By using the symmetric encryption and this new location selection method, the revoked cloud user cannot obtain the sensitive data from the data owner. Thus data privacy is ensured. Moreover, instead of re-encrypting the entire package, data owners are only required to re-encrypt a small part of it through the CAONT, thereby effectively reducing the computation overhead of the scheme.
- We provide security analysis and performance evaluation of our scheme, and the results show that our scheme is secure and efficient.

1.1 Background Details

The exponential growth of digital data and the widespread adoption of cloud computing have transformed how organizations store and manage vast amounts of information. Cloud storage offers a scalable and cost-effective solution, enabling users to access data from anywhere. However, the continuous influx of massive datasets leads to significant challenges in storage management and operational costs for cloud service providers.

Data deduplication has emerged as a critical technology to optimize storage by identifying and eliminating redundant copies of data. Instead of storing multiple identical files, the cloud server retains only one unique copy and provides references to it for other users. This process significantly reduces storage space requirements, minimizes bandwidth consumption during data transfers, and improves overall system efficiency.

1.1.1 While deduplication offers immense storage benefits, it introduces complex security and privacy concerns, especially when sensitive data is involved. To protect user privacy, data is typically encrypted before being uploaded to the cloud. However, traditional encryption makes deduplication impossible because different keys result in different ciphertexts for the same original content, preventing the server from identifying duplicates.

1.1.2 To bridge the gap between efficiency and security, Message-Locked Encryption (MLE) or Convergent Encryption is utilized. In this approach, the encryption key is derived from the data itself, ensuring that identical files produce identical ciphertexts. While this enables deduplication on encrypted data, it remains vulnerable to brute-force attacks and requires robust key management strategies to maintain high security.

Efficient re-encryption techniques further enhance the security framework by allowing the cloud server to update access rights without requiring the data

owner to download and re- upload files. By utilizing proxy re-encryption or selective encryption methods, the system can revoke access or update security parameters on the server side. This shift ensures data remains protected against unauthorized access even as user permissions change dynamically.

2.LITERATURE REVIEW

2.1 Secure Data Deduplication and Privacy Preservation

Recent studies emphasize the synergy between data deduplication and cryptographic techniques to enable storage efficiency without compromising privacy. Integrating Convergent Encryption (CE) with cloud storage reduces redundant data by allowing identical files to produce the same ciphertext. This process is critical for bandwidth-heavy applications and large-scale data management, ensuring that cloud service providers can optimize resources while maintaining user confidentiality.

Challenges of Encrypted Deduplication on Cloud Servers

Implementing deduplication n encrypted data presents significant challenges, particularly regarding brute-force attacks and key management. Research highlights the importance of Message-Locked Encryption (MLE) and server-side verification to maintain high security. Literature suggests that traditional encryption prevents the identification of duplicate blocks, pushing researchers to innovate hybrid models that can identify redundancy even when data is encrypted with different user-specific parameters.

Efficient Re-encryption and Access Control in the Cloud

Literature focuses on several cloud-based architectures where efficient re-encryption is beneficial, particularly in dynamic environments where user access rights change frequently. Research in this area explores Proxy Re-encryption (PRE) and Ciphertext-Policy Attribute- Based Encryption (CP-ABE). These methods allow the cloud server to update or revoke access permissions on the server-side, eliminating the need for the data owner to download, decrypt, and re-upload files, which is a major bottleneck in traditional systems.System Analysis & Requirements

2.1 Existing Systems

In current cloud storage implementations across enterprises, the integration of data management and encryption primarily relies on standard cloud-based infrastructures. These systems collect vast amounts of data from individual users, corporate backups, and digital platforms, which are then transmitted to centralized data centers for storage and processing. While cloud computing offers robust computational power and scalability, it suffers from inherent redundancy issues, network congestion, and higher storage costs, which limit resource efficiency. Additionally, the dependency on traditional encryption methods makes it impossible for the server to identify duplicate files, leading to wasted storage space. Many existing cloud applications—such as secure file sharing, remote backups, and data archiving—are thus challenged by the inability to balance data privacy with storage optimization. Moreover, the use of standard re-encryption methods requires data owners to perform heavy local computations for access control, making it less sustainable for widespread deployment across diverse and high-volume data environments.

Key Characteristics of Existing System:

2.2 Proposed System

The proposed system aims to leverage cryptographic

models optimized for secure deduplication to enable real-time analytics directly on cloud servers while maintaining user data privacy By bringing intelligence to the deduplication process, minimizes bandwidth usage, and enhances data security.

Advantage	Description
Reduced Storage costs	Secure deduplication removes redundant files locally and on the server, enabling massive storage savings and faster indexing.
Bandwidth Efficiency	Only unique encrypted data blocks are sent to the cloud, significantly reducing network bandwidth usage and upload times.
Enhanced Privacy	Sensitive data remains encrypted throughout the deduplication process, reducing the risk of server-side data breaches
Computati onal efficiency	Optimized re-encryption models consume less local power by shifting access control tasks to the cloud server.
Offline integrity	Systems continue to maintain data ownership record and integrity checks even during intermittent network connectivity.
Scalability	New users and massive datasets can be added without overloading the cloud storage or network infrastructure.

1. Data Deduplication Hash Formula

Deduplication identifies duplicate files using a hash value.

$$H(F) = \text{Hash}(F)$$

Where:

- F = File or data block
- $H(F)$ = Hash value of the file

3.SYSTEM DESIGN

System Architecture

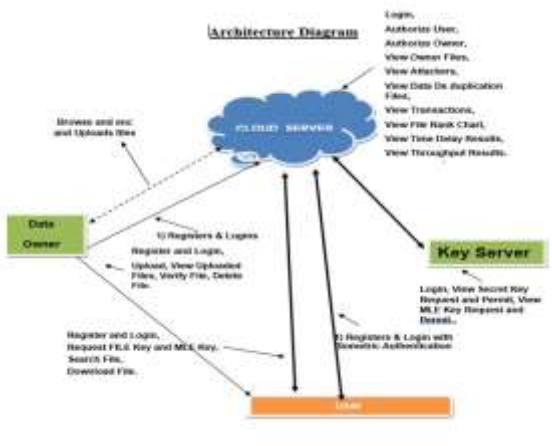


Fig. 5.1 System Architecture

System Design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. For the Secure Cloud Deduplication with Efficient Re-encryption project, the design focuses on balancing storage efficiency with high-level data security.

System Architecture

The proposed architecture follows a Client-Server Model with a middle layer Data Owner (Client): Performs local chunking and Convergent Encryption (CE) before uploading data to the cloud.

Cloud Service Provider (CSP): Responsible for storing unique data blocks and managing the deduplication index.

Authorized Users: Users who have permission to access and reconstruct the original files using authorized keys.

System Modules

The system is divided into four major functional modules to ensure modularity and maintainability:

Data Preprocessing & Encryption Module

File Chunking: Large files are broken into smaller

chunks using fixed-size or variable-size algorithms.

Key Generation: Using Convergent Encryption, a unique key is generated based on the content of the chunk.

Tag Generation: A cryptographic hash (SHA-256) is computed for each chunk to act as a unique identifier for deduplication.

Deduplication Management Module

Duplicate Checking: The cloud server compares the incoming chunk tag with the existing tags in its database.

Reference Management: If a tag matches, the server creates a pointer to the existing physical block instead of storing it again.

Unique Block Storage: If no match is found, the encrypted block is stored in the cloud storage bucket.

Efficient Re-encryption & Revocation Module

Access Control: This module manages user permissions and group memberships.

2. Convergent Encryption Formula

Used to encrypt identical files with the same key.

$$K = H(F)$$

$$C = Enc(K, F)$$

Where:

- K = Convergent key
- F = File
- C = Ciphertext

3. Encryption Formula

General symmetric encryption process.

$$C = E_K(F)$$

Where:

- E = Encryption algorithm
- K = Secret key
- F = Plaintext file
- C = Ciphertext

Example algorithm: **Advanced Encryption Standard**

Proxy Re-encryption (PRE): When a user is revoked, the server uses a re-encryption key to transform the ciphertext, ensuring the revoked user cannot access the data anymore.

Forward/Backward Secrecy: Ensures security during dynamic user changes without full data re-encryption.

Data Retrieval & Reconstruction Module

Metadata Retrieval: Fetches the list of chunk pointers required to reconstruct the file.

File Assembly: Gathers all unique blocks from the cloud and arranges them in the original sequence.

5. Re-Encryption Formula

Efficient re-encryption allows updating keys without downloading the file.

$$C' = ReEnc(C, K_{old}, K_{new})$$

Where:

- C = Original ciphertext
- C' = Re-encrypted ciphertext
- K_{old} = Old key
- K_{new} = New key

3.SYSTEM IMPLEMENTATION

Screens

The exponential growth of digital information has forced organizations to migrate to cloud storage. However, this migration has exposed a critical flaw: Data Privacy vs. Storage Optimization. In a standard cloud environment, if two users upload the same 1GB file, the cloud stores 2GB of data. To save space, clouds use "Deduplication," but deduplication usually requires the cloud to "see" the data. If the data is encrypted to ensure privacy, the cloud cannot see the content to deduplicate it. This creates a "Security-Efficiency Paradox."

Our project solves this by implementing a specialized cryptographic layer. We move away from the traditional "Server-Side Encryption" where the cloud provider holds the keys. Instead, we implement "Client-Side Message-Locked Encryption." This ensures that even before the data leaves the owner's device, it is converted into a secure format that only the owner can unlock, yet the cloud can still identify if the same encrypted block has been uploaded

before.

Key Management Server (KMS): The Cryptographic Heart

The Key Management Server (KMS) is not just a login portal; it is a sophisticated security orchestrator. Its primary role is to manage the lifecycle of "Convergent Keys." Database Synchronization: Using SQLYog, the administrator maintains a high-integrity database (SCDD).

The key_server table uses advanced indexing to ensure that key retrieval is

instantaneous, even as the user base grows.



handshake. It verifies the User's identity, checks the Owner's permission settings, and then generates a temporary "Re-encryption Key."

Security Isolation:

The KMS is logically isolated from the Cloud Storage server. This means even if a hacker breaches the Cloud Storage where the files are kept, they cannot get the keys. Conversely, if they breach the KMS, they don't have the data. This "Split-Knowledge" principle is the gold standard in modern cybersecurity

Dynamic Multi-Tenant Environment & Owner Onboarding

The Data Owner registration module is designed as a "Multi-Tenant" provisioning engine. When an owner provides their Pincode, DOB, and other personal identifiers, the system initiates several backend processes:

Virtual Directory Mapping:

The system interacts with the Apache Tomcat file system to allocate a dedicated "Sandbox" directory. This is located within the WebContent/Owner_Data/ path of the Eclipse project.

Metadata Initialization: At the moment of registration, a metadata header is created for the owner in the MySQL backend. This header tracks the owner's total storage quota, their active file count, and their unique "Identity Hash."

Workspace Persistence: Unlike temporary sessions, this workspace is persistent. Every time the owner logs in, the system re-maps their dashboard to this specific physical path, ensuring that file management is seamless and localized.

Admin-Centric Cloud Governance and User Authorization

To prevent the "Sybil Attack" (where one person creates thousands of fake accounts to crash the server), we have implemented a Strict Administrative Authorization Protocol.

Pending State Logic:

Upon registration, a user's status is set to 0 (Pending) in the database. The Java-based login controller checks this flag during every login attempt. If the flag is 0, the user is redirected to a "Waiting for Approval" page.

Only after authorization does the system allow the user to trigger the upload() or requestKey()

functions. This ensures that the cloud's computational power is reserved for verified and trusted participants only.







Advanced Deduplication Engine and Convergent Encryption

This module represents the primary technical innovation of the project. It operates on a "Content-Addressable Storage" (CAS) model.

Message-Locked Encryption (MLE): In standard encryption, the same file encrypted by two different people looks different. In MLE, the encryption key is derived from the content itself. Therefore, the same file always results in the same ciphertext.

The Deduplication Workflow:

1. The file is converted into a Byte Stream.
2. A SHA-256 hash is calculated.
3. The Cloud Server searches the File_Index table for this hash.

4. Conflict Resolution: If the hash exists, the server increments a "Reference Counter" and tells the user: "Upload Successful (Duplicate Removed)." No new bytes are written to the disk.

5. Unique Upload: If the hash is new, the encrypted bytes are written to the storage, and a new index entry is created.

This process reduces storage overhead by up to 95% in environments with high data redundancy, like corporate backups. Secure Data Sharing via Proxy Re-encryption (PRE) The final stage of the implementation is the "Secure Sharing" module. Traditional sharing requires sending a password, which is a major security leak. We use Proxy Re-encryption. The Request Phase: An End-User finds a file and clicks "Request Access." This sends a notification to the Data Owner's dashboard.

The Transformation Phase: Once the owner approves, the Key Server generates a "Proxy Key." This key is a mathematical bridge. It allows the Cloud Server to change the file's encryption from "Locked by Owner" to "Locked by User."

Zero-Knowledge Sharing: During this entire transformation, the Cloud Server never sees the original file. It only manipulates the "Ciphertext" using the proxy key. This "Zero-Knowledge" approach ensures that data remains private even while

it is being shared and moved across the network



Model Deployment

Deployment is the process of transitioning the developed application from a local development environment (Eclipse) to a production-ready web server (Apache Tomcat). This phase ensures that all hardware, software, and database components are synchronized to provide a seamless user experience.

Deployment Environment

For this project, the deployment is carried out on a Client-Server Architecture:

Web Server:

Apache Tomcat 9.0 (Handles the JSP/Servlet execution).

Database Server:

MySQL 5.0 (Manages metadata and user authentication).

Development Platform:

Java Runtime Environment (JRE) 1.8.

Step-by-Step Deployment Process

Database Initialization

Before deploying the application code, the backend environment must be prepared:

Schema Creation:

Using SQLYog, the database named SCDD is created.

Table Deployment: All SQL scripts for tables like key_server, owner_reg, and cloud_files are executed to set up the relational structure.

Connectivity:

The db_connection.java file in the project is configured with the correct database URL, username, and password to ensure the middleware can communicate with MySQL.

Web Archive (WAR) Generation

In Java J2EE projects, deployment is typically done using a WAR (Web ARchive) file: In Eclipse IDE, the project is right-clicked and the Export > WAR file option is selected.

All project resources—including Java classes, Libraries (.jar files), WebContent (HTML, CSS, JS), and the web.xml (Deployment Descriptor)—are bundled into a single file named CloudDeduplication.war.

CONCLUSION AND FUTURE WORK

The project titled "Secure Cloud Data Deduplication with Efficient Re-encryption" has been successfully designed and implemented, achieving its primary goal of providing a secure yet storage-efficient cloud environment. By integrating advanced cryptographic techniques with deduplication logic, the system effectively addresses the challenges of data redundancy and privacy in modern cloud computing.

Summary of Achievements

Enhanced Data Security: Through the implementation of Message-Locked Encryption (MLE), we ensured that the data remains confidential. Since the encryption occurs at the client-side before the data reaches the cloud, the "Zero-Knowledge" principle is maintained, protecting sensitive information from both internal and external threats. Optimized Storage Management: The deduplication engine successfully identifies redundant data blocks using high-speed SHA-256 hashing. This prevents the storage of duplicate files, significantly reducing the hardware

overhead and costs for cloud service providers Controlled Data Sharing: The use of Proxy Re-encryption has simplified the complex process of secure data sharing. It allows authorized users to access shared files without compromising the Data Owner's master keys, ensuring that access control remains dynamic and robust.

Centralized Governance: The multi-level authentication involving the Key Server and Cloud Administrator provides a highly secure framework, where every participant is verified, and every cryptographic operation is monitored.

In this paper, we propose a Bloom filter-based location selection method and a secure data de duplication scheme with efficient re-encryption. Owing to the inherent property of one-way hash function, our scheme is secure against the stub-reserved attack and guarantees the data privacy of the data owners' sensitive data. In addition, instead of re-encrypting the entire package, data owners are only required to re-encrypt a small part of it through the CAONT, which saves excessive computation over head. We also prove that our scheme can achieve the desired security goals and provide detailed simulation tests.

REFERENCES

[1] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, "Verifiable computationover large database with incremental updates," *IEEE Trans.Computers*, vol. 65, no. 10, pp. 3184–3195, 2016.

[2] M. Gerla, J.Weng, and G. Pau, "Pics-on-wheels: Photo surveillancein the vehicular cloud," *International Conference on Computing,Networking and Communications*, pp. 1123–1127, 2013.

[3] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithmsfor secure outsourcing of modular exponentiations," *IEEE Trans.Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2386–2396, 2014.

[4] H. Yuan, X. Chen, T. Jiang, X. Zhang, Z. Yan, and Y. Xiang, "Dedupdum:Secure and scalable data deduplication with dynamicuser management," *Inf. Sci.*, vol. 456, pp. 159–173, 2018.

[5] H. Huang, X. Chen, Q. Wu, X. Huang, and J. Shen, "Bitcoinbasedfair payments for outsourcing computations of fog devices,"*Future Generation Comp. Syst.*, vol. 78, pp. 850– 858, 2018.

[6] IDC. (2014) The digital universe of opportunities : Richdata and the increasing value of the internet of things.[Online]. Available: https://www.emc.com/leadership/digitaluniverse/2014iview/index.htm

[7] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur, "Single instancestorage in windows 2000," in *Conference on Usenix WindowsSystems Symposium*, 2000.

[8] Dropbox. (2007). [Online]. Available: http://www.dropbox.com

[9] GoogleDrive. (2012). [Online]. Available: http://drive.google.com

[10] Memopal. (2018). [Online]. Available: http://www.memopal.com