International Journal of Basic and Applied Sciences, 14 (3) (2025) 242-257

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/6ekp0r85
Research paper



SECUREDGE: Privacy-Preserving Deduplication with Homomorphic Encryption for Multi-Tenant Cloud Systems

Murala Vijaya *, Dr. Lade Srinivasa Chakravarthy

Department of Computer Science and Engineering, GITAM Deemed to be University, Visakhapatnam, Andhra Pradesh, India
**Corresponding author E-mail: sweetymurala3@gmail.com*

Received: June 16, 2025, Accepted: July 12, 2025, Published: July 24, 2025

Abstract

Cloud computing developments have pushed multi-tenant models to become widely used, allowing enterprises to share computing resources without mixing their data. Even though this approach works, using data deduplication to save space causes serious concerns for privacy. Conventional encryption techniques may not support deduplication since they mask similar data sections and may potentially expose concealed data when processing. Our work offers a novel approach called SECUREDGE. It encrypts data using Fully Homomorphic Encryption (FHE) to ensure security. Data privacy is guaranteed by encrypting information before deduplication. It employs an FHE-based method to identify and remove duplicate data without impacting the security edges between tenants. Businesses facing difficulties in the cloud may rest easy with SECUREDGE's intelligent storage solutions and state-of-the-art cryptography.

Keywords: *Fully Homomorphic Encryption; Multi-Tenant Data Security; Privacy-Preserving Deduplication; Secure Cloud Deduplication; Homomorphic Encryption.*

1. Introduction

Decomposing data allows for better storage in the cloud, which in turn reduces operational costs and maximizes space use. Businesses can improve their data management and cut costs by storing data in a unique way. However, there are risks associated with cloud deduplication, the most significant of which are security breaches and inappropriate access to data. We need trustworthy solutions to prevent sensitive data from being leaked or compromised during deduplication. Because of recent security breaches, it is more critical than ever to make cloud storage more secure. In 2024, hackers used faked identities to gain access to the Snowflake accounts of numerous famous companies, including AT&T and Ticketmaster. The impact caused by these attacks revealed that some security measures are not strong enough [1]. Because Nucor's employee data was compromised, a class-action lawsuit resulted, demonstrating both the real financial and legal consequences of a data breach [2]. Such instances prove that data stored on the cloud without strong deduplication and privacy security can be at risk. When deduplication is done safely, it makes storage more effective, safeguards important files, follows security regulations, and helps maintain the trust of cloud users.

1.1. Security in deduplication models

Although data deduplication saves space and helps the cloud work better, it also raises security concerns. Traditional ways of deduplication typically look at the raw files or their hash codes to find duplicates. This method has the possibility of revealing crucial information that can make a system prone to leaking data, illegal hacking, and attacks via side channels [3]. In this way, attackers may use made-to-order files or analyse duplicate replies to find out about certain data. Multi-tenant cloud settings carry the highest risks due to many users or businesses using the same platform.

One major issue that can happen here is called cross-tenant attack, where insufficient separation or misconfigured deduplication allows someone to access others' data. Hackers can exploit the same way hashing functions work or collisions to access confidential data. When access controls are not implemented enough, malicious or compromised users, including administrators, increase the risk of security problems.

These issues have prompted researchers to build better deduplication models, which now rely on encryption and zero-knowledge proofs. Traditional ways of encryption usually separate encrypted duplicates, which means that some privacy is sacrificed for better deduplication. New technologies are trying to solve this problem with Fully Homomorphic Encryption (FHE) and convergent encryption, which allow for removing duplicate encrypted data without exposing its contents. Given these constant changes, frameworks that both secure, speed up, and expand the system are key. As an example, our SECREDGE architecture relies on Fully Homomorphic Encryption (FHE) to carry

Copyright © Murala Vijaya, Dr. Lade Srinivasa Chakravarthy. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

out deduplication on encrypted information. It improves the problems found in traditional systems and helps set a higher level of data security for users in the cloud.

1.2. Paper organization

It introduces a step-by-step approach to handling privacy-preserving elimination of duplication in systems used by multiple tenants, through the outlined SECUREDGE framework. The Introduction explains what data deduplication means, how it is crucial for efficient cloud storage use, and points out the security threats it introduces in multi-tenant environments. It further explains the need for the creation of the SECUREDGE model. In the background part, we examine the technical aspects of deduplication, including its many applications, security measures, and the challenges that exist with the existing methods. The paper examines the boundaries of homomorphic encryption and highlights Fully Homomorphic Encryption (FHE) as a primary method for performing tasks that protect privacy.
This literature study identifies the issues that SECUREDGE seeks to address by reviewing recent studies on secure deduplication and cloud encryption. Safe deduplication is described in full in the SECUREDGE Model & Design section, including the architecture, algorithm flow, and use of FHE. System Implementation details the cloud platform, datasets, and configurations that will be used for testing. Compare and contrast SECUREDGE with other popular deduplication algorithms in this section, looking at its storage, efficiency, and security performance. This section summarizes what was accomplished in the study, highlights the SECUREDGE project's contributions to secure deduplication, and points out areas that can still be improved. A complete analysis of SECUREDGE's features and their practical role in the cloud can be ensured using this approach.

2. Background

2.1. Deduplication and security

Applying deduplication helps reduce the amount of data saved, which results in lower storage fees. If many employees have the same file, having only one version can boost organizational efficiency. As of now, there are two main categories of deduplication systems in use. Hash-based and delta-based. SHA or MD5 is employed to make hashes for strings of data, and these hashes are then matched to check if any blocks are repeating [4]. In contrast, delta-based (or content-aware) deduplication investigates the content of data to remove any copies. It is possible to perform deduplication on the client or server end. In this method, users check their files against a group index on their computers. It determines the amount of bandwidth required and assigns some jobs to the client, which causes the client to use more resources. A different approach involves collecting all the data on a lead server or device to perform the deduplication process. Since client devices are simpler, they require less processing power, even though the whole dataset is sent over the network.
They point out that it is necessary to ensure resources and the network are not overloaded at any time. When a large group of users connects to a cloud system, handling data sensitively becomes more critical because there are now more security risks. If deduplication is done very efficiently, without proper thought given to security, the data might not be adequately protected. They do this by emphasizing deduplication that helps achieve great effectiveness and ensures the highest level of privacy.

2.1.1. Security implications of deduplication

With traditional deduplication, common data across different users or companies is looked at and removed when it is no longer needed. While storage may be very efficient by using this strategy, your privacy could be at risk in a multi-tenant setting. Doing deduplication on plain data brings the risk of exposing private data. Uploading similar files could have the system connect them to the same block, so the other tenant could find out what the first tenant stores there. Side-channel attacks could also exploit hash values and data pointers from file deduplication to discover information about the stored data, which could lead to a higher number of data breaches among tenants [5].

2.1.2. Effect on deduplication efficiency

Solving these issues is made possible by using new types of encryption like convergent encryption and Fully Homomorphic Encryption (FHE). These systems prevent data from being reused, making sure that privacy and secrecy are ensured. They need high amounts of computing resources. Essentially, FHE helps you work with locked encryption to protect the data, but it takes a lot from the processor and often makes deduplication less effective [6]. Because of this, these approaches could fail when high speed is needed, since the code won't be able to process real-time orders as quickly.

2.1.3. Reconciling security and efficiency

Finding a balance between getting rid of duplicates and securely saving data is the most difficult part. The process of deduplication on the client, while making the network more efficient, can leave the data open to people who should not access it. Alternatively, some storage devices allow deduplication to happen at the server, helping to ensure safety and simplify control. The increased size of the data means it uses more bandwidth as it goes through the entire network [7].

2.1.4. Enhanced solutions

Now, along with deduplication, both Fully Homomorphic Encryption and Private Information Retrieval are also used by experts. The most important aim is to protect data with encryption to ensure it stays as secure as possible from outsiders. More security is usually traded for a slower operating system. Today, technologies need to find a good balance between speed, how much data is stored, and the method of encryption, which is why they work hard to achieve more.

2.2. Security in deduplication

It can be difficult to ensure the safety of data for deduplication when it is done via the cloud. IT involves taking care of many problems connected to technology, daily work, and regulations. It is important to make sure private cloud users' data is kept safe, make efforts to

prevent locking data on devices, and strictly follow both the GDPR and HIPAA rules. You must plan the system, keep the tenant information secure, and maintain good data standards so that deduplication is secure and effective. Make sure you focus on performance as well as security when performing deduplication.
Encryption is essential for safeguarding the confidentiality and integrity of data, particularly in cloud environments where sensitive information is stored and processed. Cloud service providers guarantee privacy and security by encrypting data, thus preventing unwanted access or reading of the information. Encryption can hinder deduplication by converting data into an unreadable format, which may mask commonalities between otherwise identical data objects [8].

2.2.1. Encryption and deduplication conflict

In a standard deduplication procedure, the system detects duplicated data by comparing data blocks, removing duplicates to conserve storage capacity. Encryption alters the look of data, causing identical data blocks to appear distinct post-encryption. For instance, even if two users save identical files, the encrypted representations would probably exhibit distinct ciphertexts owing to the characteristics of the encryption technique and the randomization incorporated into the encryption process (e.g., employing different encryption keys or initialization vectors). Traditional deduplication methods, which depend on comparing data for redundancy, become ineffective when the data is encrypted.

2.2.2. Encryption challenges in deduplication

To mitigate this issue, specific encryption techniques, such as convergent encryption, have been suggested. In this technique, data is encrypted according to its content, guaranteeing that identical data yields the same ciphertext. This enables the deduplication of encrypted data; however, it still entails some dangers. An adversary could access the unencrypted data if the encryption key is known or exploited. This method compromises encryption security, as identical plaintext consistently produces the same ciphertext, rendering the material susceptible to analysis or pattern recognition [9].
Fully homomorphic encryption (FHE) lets computations free from decryption on encrypted data. This enables deduplication of encrypted data, so preserving privacy during the operation. Though FHE is computationally demanding, this results in notable overhead concerning storage capacity and processing time. Although FHE ensures data security during deduplication, its performance restrictions make it ineffective for large-scale systems or situations needing highest throughput.

2.2.3. Notifying information during deduplication

Sometimes during deduplication encryption inadvertently exposes private information. Should deduplication metadata—such as hash values or pointers—not be securely controlled, attackers may deduce knowledge on the encrypted data by means of side-channel attacks. The deduplication metadata highlights trends or similarities among encrypted objects. This could result in the possible disclosure of sensitive data or unwanted data access.
Encryption is essential for data security, but it can make deduplication more difficult by making comparable data different or by raising the processing load needed for deduplication. Convergent and Fully Homomorphic Encryption (FHE) are two examples of advanced encryption algorithms that have been developed to address these issues; nevertheless, these methods do not always provide the level of security, performance, or scalability that is required. There must be a happy medium between encryption methods and deduplication algorithms for data security and deduplication to work.

2.2.4. Drawbacks of existing deduplication schemes

While current deduplication algorithms do a good job of improving storage and efficiency, they have a number of drawbacks, most notably in regard to privacy, performance, and security. These problems become more apparent in cloud systems that serve several tenants, or businesses or people, using a common infrastructure to store their data. Some of the major problems with current deduplication models are highlighted below:
Privacy and Security Issues: In order to detect redundancies, traditional deduplication methods often compare raw data blocks or file contents; however, this can put sensitive information at risk in situations with several tenants. The deduplication process in a multi-tenant system could lead to shared storage if two tenants store similar data. This could compromise tenant isolation and potentially expose one renter's data to another. Furthermore, deduplication systems generally depend on metadata (such as hash values or references) to monitor duplicates, which may be susceptible to side-channel attacks, allowing an attacker to deduce information about the data [10]. The hazards intensify when encryption is insufficiently included in the deduplication process, as unencrypted data or inadequately protected metadata may be accessible to unauthorized individuals.

2.2.5. Impact of encryption on deduplication efficiency

Whereas encryption provides key support for data privacy, it causes many problems when attempting to deduplicate. AES and other standard algorithms have changed data to make sure that two identical input blocks result in completely different ciphertexts. As a result, software tools cannot detect duplicates, lowering the way data is stored on the system. As a result, convergent encryption is being used, where the same plaintext will always become the same ciphertext, helping eliminate repetitive data during encryption. This method has inherent disadvantages. Certain plaintexts always lead to identical ciphertexts, so attackers can guess the secret information should they discover the pattern. As a result, systems for secure storage have to use more advanced methods to respect privacy while still making sure data is not duplicated.

2.2.5.1. Performance overhead

Fully Homomorphic Encryption (FHE) and similar encryption methods make it possible to perform computations on encrypted data without sacrificing data privacy. However, these solutions frequently necessitate powerful hardware, which in turn increases calculation time and memory usage. When dealing with large datasets, FHE's performance degrades, which is a key drawback of the method. This penalty makes deduplication impractical for applications that demand low latency and great performance, such as cloud systems or real-time

systems. The system's overall efficiency, scalability, and responsiveness are negatively impacted by the addition of encryption, safe computation, and decryption to deduplication. This makes the system work considerably harder.

2.2.5.2. Insufficient tenant isolation

In a multi-tenant cloud system, it is essential to separate the tenants so they do not see one another's data. Existing systems frequently struggle to distinguish data when numerous tenants share the same blocks and data specifics. On occasion, tenant data can be linked by applying the same or a comparable deduplication method to each data split. The potential for a rise in unauthorized glances, hacks, or leaks makes this restriction type risky under certain scenarios. Hackers can more easily access vital data kept in the cloud when deduplication is not applied correctly.

2.2.5.3. Limited scalability

In large, dispersed cloud environments, scalability is an issue for many of the current deduplication algorithms. Finding and removing duplicates takes more time and more processing resources when the amount of data increases. The usage of centralized deduplication indices or databases is a typical limitation, but it can be problematic when dealing with massive data sets or high traffic. Further intricacy is introduced to deduplication applications by convergent encryption, privacy-preserving encryption methods such as FHE, and related technologies. The system's capacity to scale may be compromised due to the burden that the intricate encryption methods place on its computer resources.

When attempting to implement encryption-based deduplication into the current cloud architecture, there are significant operational and architectural obstacles. System developers typically have to make major changes when using FHE or related technologies to secure calculations, encrypt data, and decrypt data. Storage orchestration, key management, and metadata management are already complicated enough without adding further complications. This can lead to increased administration and operational overheads for cloud service providers and organizations, particularly when trying to balance security, performance, and usability in a multi-tenant environment.

2.2.6. Dive into homomorphic mechanisms

It is now possible to compute on encrypted data by using homomorphic encryption, so decryption is not required. Since data is always encrypted, confidential information is always protected and never exposed while being processed. Having this functionality is especially important when data must be private in public or shared cloud environments. With homomorphic encryption, we can keep data hidden and use it safely in cloud environments without sacrificing its effectiveness [11].

2.2.6.1. Categories of homomorphic encryption

Homomorphic encryption exists in different forms, each with a unique amount of computational strength regarding encrypted data.

Partial Homomorphic Encryption (PHE) can only be performed on the encrypted data one type of operation—either addition or multiplication. While suited for some applications, its usability is undermined by its limited functionality [12].

Somewhat Homomorphic Encryption (SHE) extends this capacity to allow a limited number of both addition and multiplication operations. However, it cannot support an unbounded series of calculations since repeated operation accumulates the noise in the ciphertext to a point where decryption is no longer reliable [13].

On the other hand, Fully Homomorphic Encryption (FHE) is the most advanced form, in that uncontrolled computation of both the additive and multiplicative operations on the encrypted data is permitted. FHE can be used in adaptive computational tasks, like machine learning, statistical analytics, and complex data manipulation, while keeping the original plaintext confidential during the entire course of computation [14].

The greatest strength of homomorphic encryption is that it can offer data privacy and support extensive computations over encrypted data sets. Data needs to be encrypted before processing in conventional systems, which presents significant security threats—particularly when sensitive data is processed by unapproved third parties such as public cloud providers. Data exposure during processing can result in possible breaches or unauthorized data access. Homomorphic encryption saves the day when it comes to calculating encrypted data. Software for cloud storage that lets users send encrypted data to the service provider is one example. Even though the provider cannot view the plaintext, they are still able to search, aggregate, and dispose of duplicate data. The user will be given encrypted results after these procedures are finished, which they will have to decrypt independently to obtain the total result. By far, Fully Homomorphic Encryption (FHE) is the gold standard for protecting data duplication in cloud systems that house several tenants. Ensuring data privacy during deduplication is its forte. Protecting sensitive information and ensuring secure data isolation are both crucial in systems with multiple tenants. Cloud providers can optimize storage without access to or knowledge of the source material.

2.2.6.2. Performance overhead

Fast Hash Function (FHE) calculations are slower than plaintext procedures. If you're using your system for large-data or real-time processing, you know how noticeable performance discrepancies may be when there are delays. Implementing FHE deduplication on a broad scale could be difficult due to the significant processing overheads involved, especially for complex or abnormal procedures [15]. An efficient and secure compute environment is crucial for the challenging task of developing FHE, which necessitates certain cryptographic libraries. Bringing legacy FHE up to cloud standards can be a real pain, and major architectural shifts may be necessary for things like storage, data flow, and crucial management controls. Because of these issues, implementing FHE in actual cloud settings is difficult and costly; hence, scalability optimization and careful design are absolutely necessary.

2.2.6.3. Restricted practical implementation

Although FHE could have many applications in the future, it is still a relatively new technology and is in its early stages. There isn't much of a practical use for it. Although its ability to facilitate secure processing of encrypted data has long been established in theory, its actual application has been severely constrained due to its low efficiency and implementation complexity. For certain computational tasks, such as those involving large-scale data transfers or applications that are sensitive to latency, the state-of-the-art Fully Homomorphic Encryption

(FHE) methods may not be sufficient. As methods to reduce these costs emerge, FHE will most likely become more practical with the support of new optimization strategies like as hardware acceleration, batching, and bootstrapping updates. Industries that value customer privacy, such as cloud computing, may embrace this idea.

3. Literature survey

The growing body of research on encrypted and secure cloud storage systems highlights the vital need for privacy-preserving methods in shared databases. Using a combination of Blowfish encryption and dynamic ownership management, deduplication improves processing efficiency and privacy (Bharath Babu and J. R., 2022) [16]. The deduplication architecture outlined in Fan et al. (2019) [17] successfully counters certain ciphertext and plaintext assaults while preserving performance through the integration of convergent encryption with user permission settings.

Yang et al. (2018) [18] addressed the issue of privacy-preserving deduplication in multi-domain big data circumstances by balancing deduplication performance and secrecy using randomized tags and ciphertexts. This minimized brute force intrusions. In their analysis of privacy issues connected to multi-tenancy, Goyal et al. (2019) [19] emphasized the need for extra security measures and cited the example of cross-tenant data leaking. Given these considerations, Wang et al. (2023) [20] put out a plan to improve privacy with little effort and expense by doing away with middlemen through distributed key management and lightweight encryption.

In their extensive research on hidden computing, Novković et al. (2021) [21] emphasized the significance of managidetectionng internal and external vulnerabilities in systems with several tenants. Systems that depend on SaaS were also given a comprehensive threat model. To improve deduplication accuracy in multiuser scenarios, Shri and Srinivas (2017) [22] employed a homomorphic authenticated tree. At the same time, this improved security, performance, and integrity. El Ghazouani et al. (2024) [23] suggested a multi-agent system for secure deduplication in distributed clouds with confirmed data integrity, in contrast to Madasu et al. (2024) [24], which used ECDH encryption for cross-domain deduplication with limited resources.

By combining deduplication with encryption and indexed hash trees, Rashid et al. (2012) [25] improved cloud data security and privacy in cloud environments. These articles follow the history of safe deduplication from its earliest days of static data structures and simple encryption to its modern days of homomorphic encryption, secure computing, and blockchain technology. These works are telling of the persistent challenges of privacy and security in cloud storage and also support further developments such as the one undertaken in SECUREDGE, which seeks to implement Fully Homomorphic Encryption (FHE) for secure, efficient duplicate data detetion in a multi-tenant system.

Improving the safety of WBAN-based patient health monitoring is the primary emphasis of T. N. P. Madhuri et al. (2022) [26]. By suggesting a two-pronged strategy that uses Elliptic Curve Cryptography (ECC) for encryption and Diffie-Hellman key generation with adjustable key sizes, it tackles the crucial problem of safe data transmission between nodes. Also included is a biometric authentication system that uses a person's unique set of fingerprints or other biometric features to confirm their identity before granting access to healthcare providers or patients. In order to achieve better performance in the wireless healthcare setting employing ECC, the system utilizes an asymmetric encryption approach to guarantee data security.

3.1. Identified gaps and opportunities

The present study reports large gaps in present secure deduplication solutions that we put forth as a call for the SECUREDGE architecture. Today's practices at times fall short in scale, especially in multi-tenant and cross-domain cloud environments in which we see an increase in data and in the variety of users that complicate the system. Also, many of the present solutions rely on hardware security elements like Trusted Execution Environments (TEEs) that, in turn, present issues with respect to what is deployed and which bring up questions of hardware trust and portability. Also, at present, we see in many cases that there are performance issues which come from the crypto processing requirements and from storage inefficiency in the care of very large data sets. Also, although the use of blockchain for its audit and immutability features is very promising, we report that in practice it does not scale well and has issues with latency and limited throughput.

Besides, ciphertext growth with encryption-intensive methods detracts from system efficiency as a whole. A major limitation is the lack of quantum-resistant cryptographic techniques, which are growing in importance with the impending threats of quantum computing. SECUREDGE addresses these limitations by employing Fully Homomorphic Encryption (FHE) to enable secure deduplication without compromising plaintext data confidentiality. SECUREDGE offers robust privacy protection, maintains tenant isolation, and avoids cross-domain data leakage threats by performing operations on ciphertext data natively. SECUREDGE offers a scalable, cryptographically secure, and quantum-forward-compatible solution specifically tailored for cloud storage privacy-sensitive scenarios.

Table 1: Identified Gaps and Opportunities in Prior Studies.

Study Reference	Identified Gaps	Opportunities
Bharath Babu & J. R. (2022)	Limited focus on multi-tenant cross-domain data security.	Extend dynamic ownership models to address cross-domain security and tenant isolation in large-scale environments.
Fan et al. (2019)	Relies on TEE, which may not scale efficiently in high-performance multi-tenant systems.	Explore more scalable cryptographic methods like Fully Homomorphic Encryption (FHE) for privacy-preserving deduplication without reliance on hardware-based TEEs.
Yang et al. (2018)	Does not address high computational overhead for large-scale deduplication in cloud systems.	Optimize encryption schemes (e.g., hybrid FHE approaches) to balance computational efficiency and privacy in big data environments.
Goyal et al. (2019)	Emphasizes security concerns but lacks implementation of practical deduplication schemes.	Implement and evaluate practical, scalable privacy-preserving deduplication frameworks for resource-sharing models in multi-tenant clouds.
Wang et al. (2023)	Focuses on distributed key management but does not address ciphertext expansion challenges.	Introduce techniques like pre-encryption compression or hybrid encryption to reduce ciphertext size and improve overall storage efficiency.

Novković et al. (2021)	Primarily addresses confidentiality risks but overlooks deduplication-specific performance impacts in multi-tenant SaaS systems.	Combine confidential computing with advanced deduplication mechanisms like FHE to ensure both security and efficiency in SaaS environments.
Shri & Srinivas (2017)	Lacks advanced encryption methods for ensuring robust privacy in multiuser environments.	Integrate Fully Homomorphic Encryption with deduplication to achieve stronger privacy guarantees while maintaining performance.
Madasu et al. (2024)	Limited applicability to large-scale, diverse datasets due to resource constraints.	Enhance cross-domain deduplication models by incorporating scalable encryption techniques, enabling efficient deduplication for heterogeneous cloud datasets.
El Ghazouani et al. (2024)	Blockchain-based deduplication introduces high latency and storage overhead for metadata management.	Develop lightweight blockchain-based deduplication solutions integrated with cryptographic techniques like FHE to reduce latency and overhead.
Rashid et al. (2012)	Framework does not account for evolving threats, such as quantum computing, that could compromise encryption methods.	Design future-proof deduplication frameworks leveraging quantum-resistant encryption techniques and homomorphic encryption for long-term data security.

4. Methodology

The SECUREDGE architecture begins with a full Authentication and Authorization process, which is put in place to only allow authenticated and authorized users who wish to use the system. This module we seen to be that of multi-factor authentication (MFA), which is a combination of elements like passwords, biometrics, and crypto tokens for secure data owners and authorized entities' sign-on. Post authentication, users are given which is they have been proven to present proper authorization to do so access to basic features like data encryption, upload, and deduplication. That is a key phase for us to be protected from illegal entry, internal security threats, and also -- to prevent -- security breaches at the initial stage. The authentication module is also designed for easy plug-in to present-day cloud identity and access management (IAM) systems that, in turn, give us a wide reach across many multitenant cloud environments, and at the same time which also improve the access control at the level of the user and the tenant.

4.1. Data ingestion

In the Data Ingestion phase, data owners encrypt their files or data chunks using Fully Homomorphic Encryption (FHE) before uploading them to the cloud. FHE ensures that any future activity, including deduplication, is conducted on the encrypted data (out of sight of the actual plaintext). Fully Homomorphic Encryption (FHE) provides a provably secure foundation for performing computations directly on ciphertext, and is especially appropriate for privacy-preserving deduplication in multi-tenant cloud environments. In the Data Ingestion phase, several preparatory actions are completed before FHE encryption; the incoming files are first divided into smaller data blocks (or chunks), with each chunk subsequently encrypted with the FHE algorithm. As a reference to facilitate efficient duplication detection, a unique cryptographic hash is also produced for every encrypted chunk. These hashes will serve as identifiers during the deduplication phase. A critical benefit of this process is that since the encryption precedes client data privacy, leaving the client environment, cloud service providers do not see or have access to the underlying plaintext, thereby ensuring tenant privacy and data isolation.

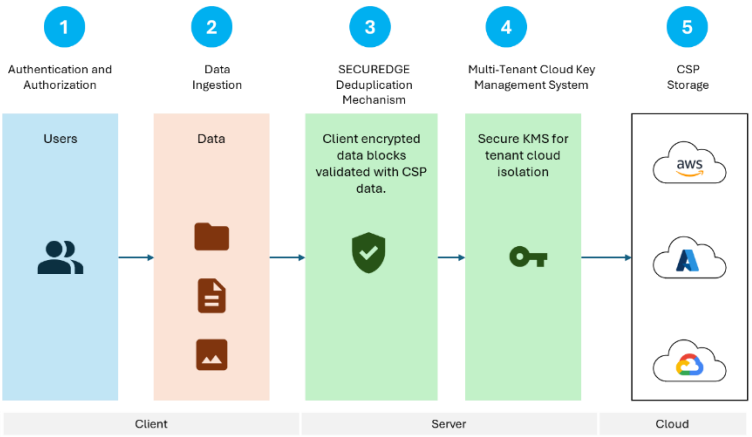


Fig. 1: SECUREDGE Architecture.

4.1.1. SECUREDGE deduplication mechanism

The SECUREDGE Deduplication Framework, which is the base function of the system that performs deduplication on encrypted data without breaking confidentiality. At upload of encrypted data blocks by the user, the system performs encrypted hash comparisons to identify and remove duplicates at the same time, which also protects the underlying plaintext from access. We use Full Homomorphic Encryption (FHE) in this to ensure safety, and at the same time reduce the compute load associated with encrypted processes. Also during this process, the client's encrypted data is methodically put up against what we have in storage with the Cloud Service Provider (CSP), of which we have, of course already indexed. Upon a match, the system does not create a new copy of the data; instead, it puts in a reference which ties the data owner to the existing encrypted block. This approach significantly reduces storage utilization without sacrificing the integrity, confidentiality, and isolation of the tenant data within a shared environment. The deduplication is maximally achieved in terms of privacy protection and storage optimality, thus being extremely compatible with security-critical cloud environments.

4.1.2. Multi-tenant cloud key management system

Ensuring tenant isolation and security of encryption keys in a multi-tenant environment demands the Multi-Tenant Cloud Key Management System (MTKMS). Every tenant receives a different set of encryption keys depending on hardware security modules (HSMs) or cloud-based safe key vaults under control by the MTKMS to stop illegal access. The system makes sure that tenants and CSP managers never know about encryption keys. The secure key rotation and revocation of the MTKMS guarantees the conformity to data security standards. Even in the case of a cloud service provider intrusion, tightly separating access restrictions helps to achieve tenant isolation thus it becomes challenging for one tenant to read or decode another tenant's data.

4.1.3. Cloud service provider data storage

Retained in the Cloud Service Provider (CSP) Data Storage are encrypted, last phase deduplicates data blocks. Strict security criteria let this storage layer be fault-tolerant and redundant most effectively. The data is kept encrypted using homomorphic encryption (FHE); metadata is carefully controlled to link data owners to their matching encrypted blocks. This guarantees, even in the event of a data breach, the assailant cannot access private data without the required keys. Following GDPR, HIPAA, ISO 27001, and applying tenant isolation techniques, CSP storage. The SECUREDGE framework guarantees safe and efficient storage even in big, multi-tenant cloud systems.

4.2. Algorithm and scheme design

4.2.1. Evolution of FHE algorithm

We investigated secure deduplication in multi-tenant cloud services using Brakerski-Gentry-Vaikuntanathan (BGV), one of the possibilities for fully homomorphic encryption (FHE) techniques. The secure deduplication industry is an appropriate match for BGV because of its ability to execute mathematical operations on encrypted data. Cloud systems that handle large amounts of data benefit from batch processing because it allows for the simultaneous processing of several data items, which increases their speed and scalability. Not only that, BGV ensures strong data confidentiality by allowing calculations on encrypted data and minimizes security threats like unauthorized access in multi-tenant systems. Proven applications of BGV, such as IBM's HELib, prove its value and lay the groundwork for its incorporation into live systems. In light of our research objectives, BGV outperforms the SECUREDGE architecture in terms of safety and efficiency.

4.2.2. Scheme preliminary and notations

Brakerski-Gentry-Vaikuntanathan (BGV) is a lattice-based method that enables fully homomorphic encryption (FHE), a methodology that enables computations on encrypted data without decryption. We will go over the key notations and preparatory schemes of the SECUREDGE design, the main operations of the algorithm, and the general workflow of the BGV algorithm below.

Working Flow

Key Generation:

The key generation process involves creating three keys: the secret key (sk), the public key (pk), and the evaluation key (evk).

- The secret key is used for decryption and remains private.
- The public key is used for encryption and is shared with entities needing to encrypt data.
- The evaluation key allows homomorphic operations on encrypted data.

Process:

- Randomly generate a secret key $sk \in Rq$ where Rq is a polynomial ring modulo q.
- Compute pk and evk based on sk and lattice-based cryptographic assumptions.

Encryption:

- Beginning with the public key pk, the BGV method constructs a polynomial representation of the plaintext message m. Consequently, we may deduce the ciphertext c by solving the equation:

$$c=pk\cdot m+e \pmod q \tag{1}$$

- Here, e is a small noise Poisson that encrypts the plaintext within the ciphertext, guaranteeing cryptographic security. The ring is defined by the modulus q in Poisson arithmetic.

Evaluation (Homomorphic Operations):

- By providing homomorphic operations with the evaluation key evk, the BGV approach gives algebraic computations on ciphertexts without decryption. T:

$$\text{Addition: } c_{add}= (c1+c2) \pmod q \tag{2}$$

$$\text{Multiplication: } c_{mul}=(c1\cdot c2) \pmod q \tag{3}$$

Decryption:

- The encrypted result is decrypted using the secret key (sk).
- The decryption formula is

$$m=c\cdot sk^{-1} \pmod q \tag{4}$$

- The output is the plaintext result of the computations performed on encrypted data.

Table 2: Scheme Preliminaries and Notations

Notation	Description
R_q	Polynomial ring modulo q, $R_q=Z[x]/(x^n+1)$
Sk	Secret key used for decryption
Pk	Public key used for encryption
Evk	Evaluation key used for performing homomorphic operations
M	Plaintext message or data to be encrypted
C	Ciphertext, the encrypted form of m
Q	Modulus for noise and polynomial coefficients determines encryption security
E	Noise polynomial added during encryption to enhance security
C_{add}	Resultant ciphertext from homomorphic addition
C_{mul}	Resultant ciphertext from homomorphic multiplication
Δ	Scaling factor used to map plaintext into ciphertext space
LWE	Learning With Errors, the lattice-based hardness assumption underlying the security of BGV
Ciphertext Slot	A unit of plaintext data encoded in the ciphertext, allowing batching of multiple plaintext values.

4.2.3. Proposed algorithm

Offering deduplication in multi-tenant cloud systems while protecting user confidentiality is the aim of the Secure Deduplication with Encryption using Fully Homomorphic Encryption project. Using Fully Homomorphic Encryption (FHE) to conduct calculations on encrypted data, SECUREDGE safeguards sensitive tenant data during deduplication. Secure processing, comparison, and storage of encrypted data blocks while protecting the privacy of unencrypted data are the main goals of the SECUREDGE system, which is detailed below along with its operational algorithm and architectural framework.

Scheme A: FHE File Does Not Exist

Objective:

To ensure privacy and make deduplication easier in the future, it's best to save newly created files in the cloud securely if they aren't already encrypted.

Steps on Working Flow

1) File encryption

By utilizing the public key pk and the fully homomorphic encryption (FHE) approach, the data owner encrypts the plaintext file F:

$$C_F=Enc_{pk}(F) \tag{5}$$

The ciphertext of the file can be securely processed and stored by assuming the value CF.

2) Generating file metadata

The original file F generates a unique hash HF using a secure cryptographic hash function:

$$H_F=Hash(F) \tag{6}$$

This hash not only identifies the file uniquely for deduplication but also checks if it is stored on the cloud.

3) Cloud Query

The Cloud Service Provider (CSP) verifies if another file with the same contents already exists after receiving the encrypted file hash HF.

4) Store the File

Should no matching hash come across in the CSP storage ($H_F \notin CSP$):

- The cloud securely stores the ciphertext C_F.
- For future comparison and reference, the hash H_F is included in the deduplication index of the CSP.

Using FHE, this method ensures that no copies of encrypted files are ever stored, reducing data duplication while protecting user privacy.

Scheme B: FHE File Exists with Same Content

Objective:

To prevent wasting space and avoid uploading identical content that already exists encrypted in the cloud, it is best to skip the upload.

Working Flow Procedures

1) File Encryption and Hash Generation

The owner of the data encrypts file F: using the Fully Homomorphic Encryption (FHE) method and the public key pk.:

$$C_F=Enc_{pk}(F) \tag{7}$$

Then, a one-of-a-kind cryptographic hash HF is generated from the contents of the file:

$$H_F=Hash(F) \tag{8}$$

2) Cloud Query

In order to check if the identical file has previously been stored, the encrypted file hash HF is given to the Cloud Service Provider (CSP).

3) Match for Hash

If HF is a member of CSP, meaning the hash is already in the CSP's deduplication index, then an encrypted file that is identical to it already exists.

4) Skip Upload

By avoiding the upload altogether, the system preserves data secrecy, uses less bandwidth, and reduces storage capacity requirements because the file is already in encrypted form.

The use of FHE-enabled encrypted hash comparisons ensures storage efficiency while protecting data privacy.

Scheme C: FHE File Exists with Different Content

Objective:

Store a file in the cloud as a new version or unique entity to ensure data integrity when its name matches an existing file but its content differs.
Steps within the working flow
i) File Encryption and Hash Generation
The data owner encrypts file F with fully homomorphic encryption (FHE) and the public key pk:

$$C_F=Enc_{pk}(F) \quad (9)$$

A cryptographic hash H_F is calculated from the original file content:

$$H_F=Hash(F) \quad (10)$$

ii) Cloud Query
The Cloud Service Provider (CSP) gets the encrypted hash H_F to check for an already-match.
iii) Hash Mismatch
If HF ∉ CSP, a file with the same name but different contents might exist. The file is thus considered to be unique.
iv) Upload the File.
The encrypted file CF is saved in the cloud in an encrypted version after it has been uploaded as a new entity. The next step is for the CSP to govern future deduplication searches by adding the hash HF to its deduplication index.
This approach ensures multiple cloud-based backups by using FHE to protect file data. This makes sure the data is still real and can be checked.

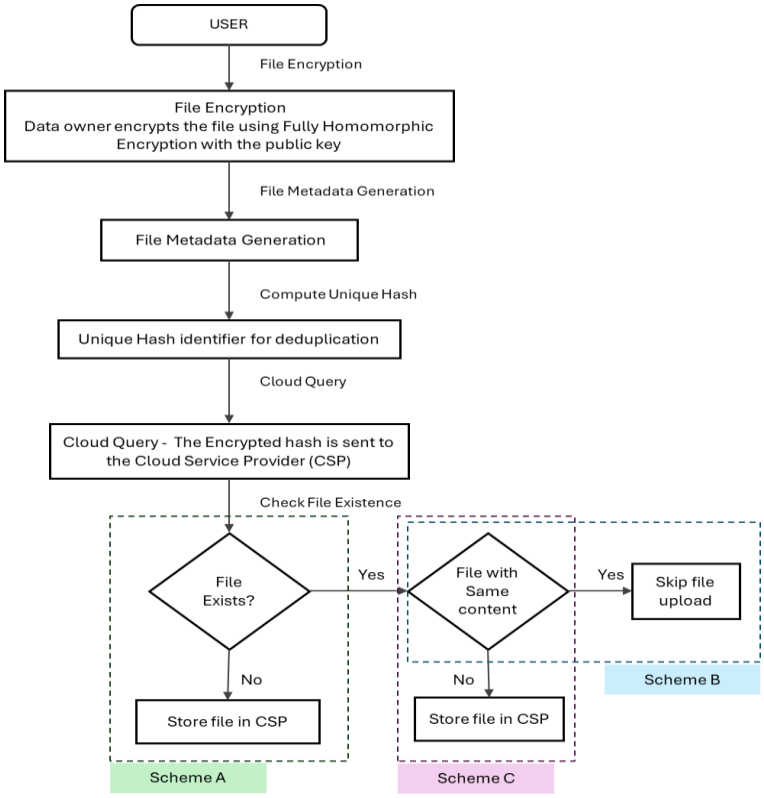


Fig. 2: Flow of SECUREDGE Schemes.

Table 3: Notations and Formulae

Notation	Description
F	File in plaintext
C_F	Encrypted file using FHE: $CF=Enc_{pk}(F)$
pk	Public key used for encryption
sk	Secret key used for decryption
H_F	Hash of the file, unique identifier for deduplication: $H_F=Hash(F)$
CSP storage	Cloud Service Provider's storage containing encrypted hashes of stored files
$H_F \in$ CSP storage	Indicates the file exists in the cloud storage
$H_F \notin$ CSP storage	Indicates the file does not exist in the cloud storage

4.2.3.1. SECUREDGE scheme design

The Multi-Tenant Cloud Key Management module of the SECUREDGE architecture enables users to privately encrypt, decrypt, and isolate keys for many cloud tenants. By creating distinct cryptographic and logical key spaces for each tenant, this module prevents key sharing and the acquisition of keys by other tenants or Cloud Service Providers (CSPs). Encryption, decryption, and key management are all essential cryptographic activities that can be safely executed in a trusted environment. Data communication is made safe by enabling a secure key distribution method. Following the principles of zero-trust cloud security, this approach mitigated internal risks, tenant attacks,

and the possibility of sensitive data slipping into the wrong hands. Tenants are able to connect with dependable key management systems, which aids in scalability, and the solution also ensures the security criteria of organizations. One of the most notable solutions is Azure Key Vault, while others include HashiCorp Vault and AWS KMS.

Workflow Steps

Step 1: Tenant Key Generation

Each tenant T_i generates a unique public-private key pair (pk_i,sk_i) using a Key Generation algorithm:

$$(pk_i,sk_i) = KeyGen(\lambda) \quad (11)$$

Where:

- λ: Security parameter (e.g., key size).
- pk_i: Public key for encryption.
- sk_i: Private key for decryption.
- These keys are managed independently for each tenant to ensure tenant isolation in a multi-tenant cloud.

Step 2: File Encryption

The tenant encrypts the file, Fusing their unique public key pki with Fully Homomorphic Encryption (FHE):

$$C_F=Enc_{pki}(F) \quad (12)$$

Where:

- CF: Encrypted ciphertext of the file F.
- Encpki(F): FHE encryption function.
- The encryption ensures that tenant Ti data remains secure even if other tenants attempt unauthorized access.

Step 3: Secure Key Distribution

The CSP does not store sk_i (private key) to prevent key exposure.

A Secure Key Management System (SKMS) is used to facilitate secure distribution of sk_i to authorized entities using cryptographic protocols like Diffie-Hellman or Public Key Infrastructure (PKI).

The SKMS maintains a mapping of tenant identifiers T_i and their respective public keys pk_i:

$$KeyMapping: \{T_1 \rightarrow pk_1, T_2 \rightarrow pk_2, \ldots\} \quad (13)$$

Step 4: Tenant Isolation

Each tenant T_i's keys and encrypted data are isolated logically within the cloud infrastructure.

Data for tenant T_i can only be decrypted using sk_i, ensuring that:

$$F=Dec_{ski}(C_F) \quad (14)$$

Cross-tenant access is restricted since the private key ski is not shared or accessible by other tenants or the CSP.

Step 5: Secure Data Sharing

If tenant T_i wants to share encrypted data C_F with tenant T_j, a Re-Encryption Key (REK) REKi→j is generated:

$$REKi \rightarrow j=GenReKey(sk_i,pk_j) \quad (15)$$

The encrypted file is re-encrypted for tenant T_j:

$$C_{Fj}=ReEnc(REK_{i \rightarrow j},C_F) \quad (16)$$

Tenant T_j can now decrypt the file using their private key sk_j:

$$F=Decsk_j(C_{Fj}) \quad (17)$$

Step 6: Key Revocation

If a tenant T_i leaves the cloud or their keys are compromised, the SKMS invalidates pk_i and sk_i. A new key pair is generated, and data is re-encrypted.

Table 4: Notations and Formulae

Notation	Description
λ	Security parameter (e.g., key size)
Ti	Tenant identifier
(pki,ski)	Public-private key pair for tenant Ti
F	File in plaintext
CF	Encrypted file using FHE: CF=Encpki(F)
Encpki(F)	Fully Homomorphic Encryption function
Decski(CF)	Decryption function using private key ski
KeyMapping	Mapping of tenant identifiers to public keys: {T1→pk1,T2→pk2,… }
REKi→j	Re-Encryption Key for sharing data between Ti and Tj
ReEnc(REKi→j,CF)	Re-encryption function using REKi→j

5. System implementation

5.1. Experimental setup

Table 5: Experimental Setup for SECUREDGE Framework

Component	Specifications	Purpose
Cloud Platform	AWS (Amazon Web Services)	To simulate a real-world multi-tenant cloud storage environment.
Tenant Simulations	10-50 virtual tenants, each representing different data owners.	To test tenant isolation, deduplication efficiency, and security measures.
Encryption Library	Microsoft SEAL	Implements Fully Homomorphic Encryption (FHE) for encrypted deduplication.
Data Sets	- Public datasets (e.g., Enron email dataset, synthetic datasets) - Sizes: 10GB, 50GB, 100GB	To evaluate deduplication and encryption performance on diverse data types and scales.
Computing Resources	- VM instances: 4 vCPUs, 16GB RAM, SSD storage - Multi-node cluster for distributed deduplication testing	To handle encryption and deduplication computations efficiently and test scalability.
Key Management System	Secure Key Management System (e.g., Azure Key Vault)	For managing encryption keys and tenant isolation securely.
Deduplication Mechanism	SECUREDGE deduplication module integrated with FHE	To validate encrypted data blocks for deduplication.
Performance Monitoring Tools	- Grafana, Prometheus - Built-in metrics (CPU usage, memory, I/O performance)	To monitor the computational overhead introduced by FHE and deduplication.
Evaluation Metrics	-Encryption time -Deduplication efficiency - Storage space saved - Key management overhead	To measure the performance and security of the proposed system.
Security Analysis Tools	Penetration testing tools (e.g., OWASP ZAP)	To test the system's resistance against security vulnerabilities, including key breaches.
Test Environment Configuration	Windows Server 2019 and above	Provides a consistent environment for running experiments.
Comparison Models	Existing deduplication models (e.g., Convergent Encryption, Privacy-Preserving Deduplication schemes)	To benchmark SECUREDGE against current state-of-the-art deduplication systems.
Simulation Tools	- Python (NumPy, Pandas) for data simulation	To simulate deduplication scenarios and analyze results systematically.

5.2. Proposed model execution

By means of a rigorous and comprehensive experimental framework, the proposed SECUREDGE model evaluates its security, scalability, and effectiveness within a multi-tenant cloud environment. The operation consists of the following subsequent phases and elements:

1. Cloud Platform Simulation

The SECUREDGE model replicates a real multi-tenant cloud environment using AWS (Amazon Web Services). The platform offers scalability, resource allocation, and storage capacity to fairly depict events from the real world.

2. Tenant Simulations

The cloud environment can house 10 to 50 virtual tenants, each of which stands for a different data owner. This architecture guarantees, among several users using the same infrastructure, the evaluation of tenant isolation, deduplication efficacy, and security protocols.

3. Data Encryption Employing Fully Homomorphic Encryption

Data Encipherment made possible by Fully Homomorphic Encryption (FHE). The Microsoft SEAL library fully homomorphicizes tenant data before upload. This guarantees encrypted data stays while processing, so enabling deduplication operations and lowering security and privacy issues.

4. Data Ingestion and Dataset Configuration

Analyzing public data, including the Enron email gathering, the system generates datasets ranging 10GB to 100 GB. These datasets allow one to evaluate the efficiency of the deduplication and encryption techniques over several data sizes.

5. Distributed Computing and Resource Allocation

Virtual machines tuned with four vCPUs, 16GB of RAM, and SSD storage process data in concert with a multi-node cluster to test distributed deduplication. This guarantees effective treatment of chores needing resources, including encryption and data validation.

6. Key Management System

Powerful key management systems such as Azure Key Vault track encryption keys. The system guarantees tenant isolation by blocking illegal access and securely tying keys to assigned tenants.

7. Deduplication Mechanism

Against current cloud data, the FHE encryption method and the SECUREDGE deduplication module find and verify duplicate encrypted data blocks. This guarantees effective deduplication free from the data decryption needed.

8. Performance Evaluation

Prometheus and Grafana let real-time system performance monitoring take advantage. We track important indicators including CPU use, memory consumption, and input/output performance to assess computational overhead and efficiency.

9. Evaluation Metrics:

We assess the model applying the following standards.

Encryption Duration: The length required for completely homomorphic encryption of data.

Deduplication Efficiency: Over the upload process, exactly found and removed percentage of duplicate data blocks.

Storage Space Saved: Reducing storage use brought about by deduplication helps to save space.

Key Management Overhead: Multi-tenant systems' computational and resource cost of distributing encryption keys.

10. Security Evaluation
The SECUREDGE platform has been thoroughly tested for vulnerabilities using recognized tools in the industry, such as OWASP ZAP and Burp Suite. This technology can mimic a variety of possible attack vectors, such as breaches of tenant isolation, illegal access, and massive extraction operations. These tests assess the framework's resilience to both internal and external threats, proving that it successfully implements access control and safeguards data confidentiality in a multi-tenant cloud environment.
11. Comparative Analysis
We may evaluate SECUREDGE's efficacy and safety by comparing it to other privacy-preserving deduplication systems and existing deduplication techniques, like convergent encryption. Among the many benefits of SECUREDGE that are emphasized in this comparison are its enhanced encryption strength, accurate encryption data duplication detection, and storage space optimization. Based on the results, SECUREDGE is the superior choice where privacy, scalability, and security are of the utmost importance.
12. Simulative Tools
The utilization of libraries like NumPy and Pandas for processing and visualizing data allows analytical and simulation activities to be performed using Python-based technologies. By simulating various deduplication situations across a number of parameters, including dataset size, duplication ratio, and encryption complexity, these tools provide valuable insights into system behavior, performance trends, and future enhancement options.

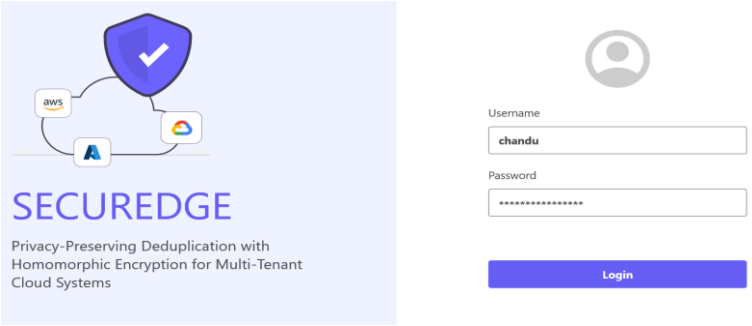


Fig. 3: User Authentication Screen.

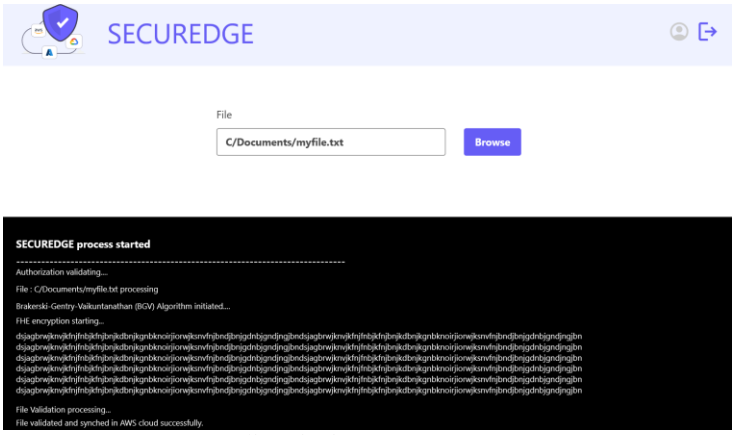


Fig. 4: File Upload & SECUREDGE Process.

6. Results and discussion

6.1. Cloud service provider configurations

Large-scale multi-tenant deduplication activities depend on optimal configurations, especially concerning the SECUREDGE framework on big datasets (e.g., 100GB). These configurations enable researchers to investigate the scalability, computational overhead, and responsiveness of the system in practical cloud environments by means of thorough evaluation of encryption and deduplication efficacy at maximum load conditions. Showcasing the resilience and applicability of SECUREDGE in corporate-level environments calls for premium configurations. Conversely, low-end models are fit for smaller-scale simulations and initial model validation—that is, for testing on 10GB datasets. These settings are efficient for evaluating fundamental deduplication logic, verifying algorithm accuracy, and tracking initial performance indicators without the resources of large-scale installations. This work gives high priority to high-configuration scenarios to demonstrate how well SECUREDGE manages complex, privacy-sensitive deduplication jobs on multi-tenant cloud platforms, validating the relevance of the framework to contemporary cloud architecture.

Table 6: CSP Configurations

CSP	Configuration	CPU	CPUs	Memory (GB)
AWS Elastic Cloud Computing™	Low	Intel Xeon E5-2666 v3	4	16
	High	Intel Xeon Platinum	72	192
Azure Virtual Machine™	Low	Intel Xeon Platinum 8168	4	16
	High	Intel Xeon Platinum 8168	72	144
Google Cloud Compute Engine™		Intel Xeon Scalable	4	16
	High	Intel Xeon Scalable	60	240

6.1.1. Encryption time

Encryption time is defined as the overall timeframe necessary to transform plaintext data
Encryption time is the total time required using a fully homomorphic encryption (FHE) method to convert plaintext data into ciphertext. This statistic enables one to assess the security module performance in the architecture of SECUREDGE. Particularly in multi-tenant cloud systems handling large databases, the encryption process results in appreciable processing load.
Tenc, the encryption time, has a mathematical form as:

$$Tenc = f(S,C,R) \tag{18}$$

Where:

- S: Size of the dataset (in GB or MB)
- C: Complexity of the FHE algorithm (e.g., polynomial or exponential based on BGV operations)
- R: Resources used, including CPU, memory, and parallelism

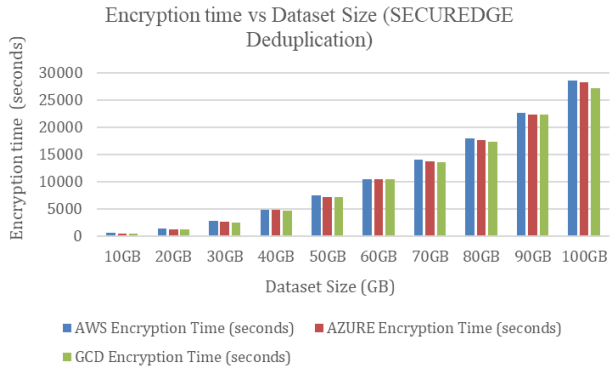


Fig. 5: Encryption Time vs Dataset Size.

Using fully homomorphic encryption (FHE), the graph exposes the relationship between encryption duration and dataset size, so guiding the encryption time with increasing data volume. The rather linear increase in the encryption time concerning dataset size suggests the major computational load FHE generates. Smaller datasets (e.g., 10GB) fit low-latency or moderate-scale use since their fast ending of the encryption process is appropriate. Still, the encryption length rises sharply with increasing dataset size—reaching 100GB—exposing a scalability problem for FHE-based systems in high-volume environments. This trend underlines in safe deduplication systems the natural trade-off between strong data privacy and system efficiency. Practicality and responsiveness in real-world multi-tenant cloud systems are underlined by efficient homomorphic encryption solutions, improved resource allocation, or hybrid cryptographic approaches.

6.1.2. Deduplication efficiency

The ratio of duplicated data that has been effectively found and removed from the whole uploaded collection is known as deduplication efficiency. This statistic is especially important in data-intensive cloud environments since it shows how well a deduplication system lowers storage consumption. It is calculated as:

$$\text{Deduplication Efficiency (\%)} = \frac{\text{Data Reduced (GB)}}{\text{Total Uploaded Data}} \text{X } 100 \tag{19}$$

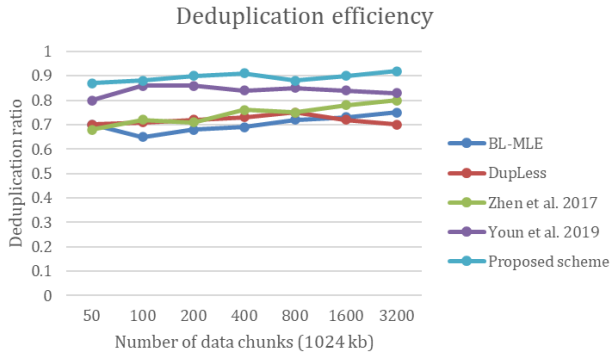


Fig. 6: Deduplication Efficiency.

6.1.3. Storage space efficiency

Efficiency of storage space measures the proportion of stored space preserved using a deduplication technique. This metric shows the system's ability to eliminate duplicate data, so improving storage capacity in cloud computing systems. It serves as a good indicator of how well the deduplication framework minimizes storage costs while maintaining data security and access.

$$\text{Storage Space Efficiency (\%)} = \frac{(\text{Storagewithoutdeduplication} - \text{Storagewithdeduplication})}{\text{Storage without deduplication}} \text{X } 100 \tag{20}$$

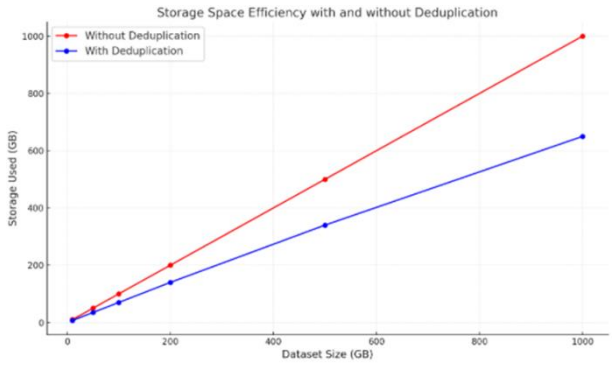


Fig. 7: Storage Space efficiency.

Deduplication allows the graph to show the storage efficiency attained in the secure multi-tenant cloud environment. As the dataset increases, the deduplication method shows clear improvement in storage optimization; efficiency percentages are always rising. This rising trend shows the algorithm's capacity in identifying and removing duplicate data blocks, especially in bigger datasets where duplication is more common. The results highlight the proposed SECUREDGE platform's scalability and efficiency. SECUREDGE satisfies the needs for strong privacy protections and best use of resources by guaranteeing outstanding storage efficiency and data security, so demonstrating fit for pragmatic cloud implementations. The results confirm that deduplication using FHE can scale effectively with increasing data volumes in multi-tenant systems.

6.1.4. Key management overhead

Very much the scalability and efficiency of encryption systems in multi-tenant cloud environments depends on the management overhead of keys. It relates to the computational and resource costs of securely producing, distributing, rotating, deleting, managing encryption keys among many tenants. This overhead in the SECUREDGE architecture is much influenced by the use of secure key storage systems such as HashiCorp Vault and fully homomorphic encryption (FHE). Unlike systems using simpler encryption techniques—such as Convergent Encryption, which usually results in reduced overhead—FHE-based architectures like SECUREDGE exchange increased computational complexity for better security, encompassing increased tenant isolation and greater resilience against key compromise. More importantly influences overhead in encryption and decryption processes the efficiency of major rotation and revocation methods, as well as the latency related to key retrieval. Reaching equilibrium between strict security measures and operational efficiency will help to guarantee that major management processes are scalable, responsive, and fit for pragmatic implementation in privacy-sensitive cloud services.

$$T_{KMO} = T_{Encryption} + T_{KeyGeneration} + T_{KeyStorage} + T_{KeyDistribution} \quad (21)$$

Where:
T_{KMO}: Total key management overhead latency (in milliseconds).
$T_{Encryption}$: Time taken for encryption operations.
$T_{KeyGeneration}$: Time required to generate encryption keys.
$T_{KeyStorage}$: Time taken to securely store the keys.
$T_{KeyDistribution}$: Time required to securely distribute keys to tenants or systems

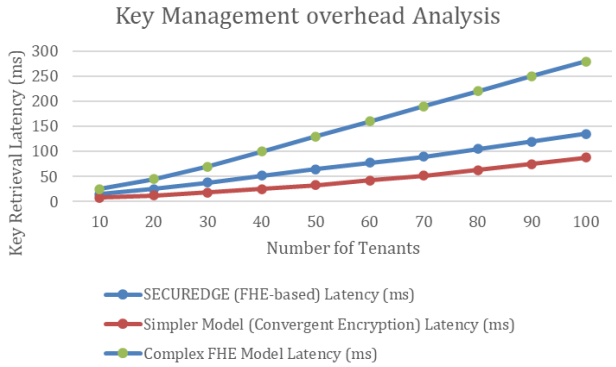


Fig. 8: Key Management Overhead.

The graph displays SECUREDGE's key management overhead compared to other deduplication systems, using metrics like key retrieval latency (ms) on the y-axis and the number of tenants on the x-axis. SECUREDGE demonstrates a moderate but scalable overhead due to its advanced key isolation and encryption mechanisms. Simpler systems show lower overhead but are less secure, while more complex FHE systems have higher acceptable costs. The graph underscores SECUREDGE's ability to balance security and computational efficiency practically.

6.1.5. Security analysis

The SECUREDGE framework was tested using the OWASP ZAP to assess its efficacy in a multi-tenant cloud setting. The major objectives of the research were to address data breach defenses, cross-tenant data leaks, and encryption key management security. Data duplication and fully homomorphic encryption (FHE) work hand in hand to prevent computation or administration from exposing plaintext, thus

delivering robust data confidentiality. By analyzing tenant isolation solutions in multi-tenant cloud platforms, we show that the system successfully stops tenants from sharing keys and unauthorized access. The framework also passed a battery of synthetic penetration tests that included a wide range of external and internal threats, including injection assaults, brute-force attempts, key extraction scenarios, and more. According to the findings, SECUREDGE's privacy-preserving deduplication, key protection, and complete isolation make it a top choice for highly secure cloud environments.

6.1.6. Comparative analysis

The article compares and contrasts SECUREDGE with a number of other methods. Some of these techniques include ciphertext-based multi-domain deduplication, TEE-integrated privacy-preserving frameworks, blowfish-based dynamic deduplication, and hybrid blockchain-multi-agent systems. To find the best architecture, we analyzed its security time, deduplication efficiency, storage capacity conservation, and control over keys overhead. Using SECUREDGE's fully homomorphic encryption (FHE), deduplication on encrypted data might be accomplished without disclosing the plaintext. We achieved parity in deduplication efficiency while staying within strict privacy requirements, all because of storage optimization for SECUREDGE. The model's security was bolstered by incorporating a secure multi-tenant key management solution, which improved tenant isolation and key compromise protection. These results validate SECUREDGE as a powerful, scalable, privacy-first deduplication solution for multi-tenant cloud infrastructures that prioritize efficiency and privacy.

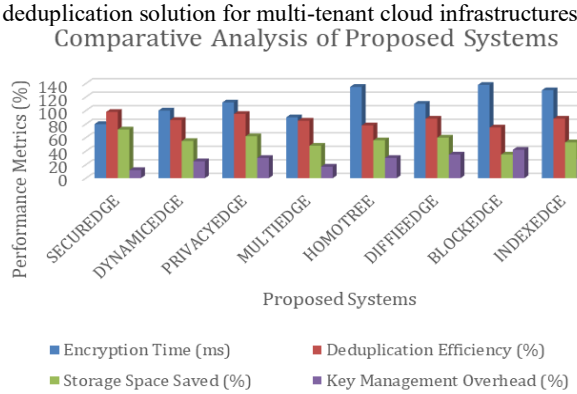


Fig. 9: Comparative Analysis.

The graph displays various deduplication systems compared based on four key performance metrics: time to encrypt, efficiency of deduplication, storage space saved, and overhead in key management.
X-axis: Proposed Systems
Y-axis: Metric Values
For encryption time: Time in milliseconds (ms)
For deduplication efficiency, storage saved, and key overhead: Percentage (%)
As seen in the figure, SECUREDGE indicates that Fully Homomorphic Encryption (FHE) can improve encryption security while keeping deduplication efficiency competitive and optimized storage. Renter isolation is superior in all studied systems, but key management overhead is marginally more than lightweight alternatives. Based on these results, SECUREDGE is the best option for multi-tenant cloud settings that prioritize user privacy.

7. Conclusion

The present research verifies that the SECUREDGE concept is reliable for managing privacy-preserving deduplication in cloud environments with multiple tenants. To accomplish secure deduplication using fully homomorphic encryption (FHE), SECUREDGE is an option. This approach safeguards user privacy while surpassing competing deduplication techniques on critical assessment parameters. Fully Homomorphic Encryption (FHE) is computationally complex, yet SECUREDGE demonstrates decent responsiveness and scalability in comparison to popular cloud systems like AWS, Azure, and Google Cloud. Notable deduplication approaches such as BL-MLE and DupLess are surpassed by the model, according to Zhen et al. (2017) and Youn et al. (2019). The significance of SECUREDGE in large-scale applications is demonstrated by its consistently improving deduplication ratios as dataset quantities increase. The ability of SECUREDGE to provide secure, low-latency, low-key retrieval with excellent tenant isolation has been proven by a recent study on key management overhead. Despite using more resources than systems that use convergent encryption, SECUREDGE guarantees user privacy and performs as well as current FHE-based systems. The SECUREDGE solution achieves optimal performance, storage efficiency, and security in privacy-sensitive, multi-tenant cloud systems by offering a scalable and consistent method for safe deduplication.

8. Future work

There is much need for improvement in the field of secure deduplication in multi-tenant cloud systems; nevertheless, the SECUREDGE architecture has greatly improved it. Make better use of fewer processing resources by employing Fully Homomorphic Encryption (FHE). Batch encryption methods and hardware acceleration (GPUs, FPGAs, etc.) can enhance encryption throughput and system responsiveness for large datasets. Through the utilization of hybrid encryption methods and the merging of two lightweight systems, Fully Homomorphic Encryption (FHE) is achieved, which offers an improved secure strategy. The privacy-performance trade-off may, however, become less stringent. More effort to improve SECUREDGE to enable cross-cloud deduplication in all clouds (federated or hybrid) could be a solution to the trust management and inter-cloud key coordination problems. A more robust and adaptable key management system can be achieved by using algorithms that are impervious to quantum computing or by employing key auditing methods that are based on blockchain technology. It is critical to do research into adaptive deduplication algorithms so that SECUREDGE can perform better in various cloud deployment scenarios. The systems are designed to adapt on the fly to changes in data sensitivity and redundancy levels.

Acknowledgement

I would like to thank my guide, Dr. Lade Srinivasa Chakravarthy, for his support in my research work, and also thankful to GITAM University for providing sufficient laboratory support for this work.

Conflict of interest

The authors declare that there is no conflict of Interest.

References

[1] Newman, L. H., "The worst hacks of 2024", WIRED, 2024. https://www.wired.com/story/worst-hacks-2024/
[2] Mettela, T., "Americans in line to get $8,250 checks from 'steel' data breach settlement – and you only need a single recei. The US Sun, 2024.
[3] Shin, Y., Koo, D., & Hur, J., "A Survey of Secure Data Deduplication Schemes for Cloud Storage Systems", ACM Computing Surveys (CSUR), 49, 1 – 38, 2017. https://doi.org/10.1145/3017428.
[4] Kaur, R., Chana, I., & Bhattacharya, J., "Data deduplication techniques for efficient cloud storage management: a systematic review", The Journal of Supercomputing, 74, 2035 – 2085, 2017. https://doi.org/10.1007/s11227-017-2210-8.
[5] Prajapati, P., & Shah, P., "A Review on Secure Data Deduplication: Cloud Storage Security Issue", J. King Saud Univ. Comput. Inf. Sci., 34, 3996-4007, 2020. https://doi.org/10.1016/j.jksuci.2020.10.021.
[6] Xie, Q., Zhang, C., & Jia, X., "Security-Aware and Efficient Data Deduplication for Edge-Assisted Cloud Storage Systems", IEEE Transactions on Services Computing, 16, 2191-2202, 2023. https://doi.org/10.1109/TSC.2022.3195318.
[7] Jiang, S., Jiang, T., & Wang, L., "Secure and Efficient Cloud Data Deduplication with Ownership Management", IEEE Transactions on Services Computing, 13, 1152-1165, 2020. https://doi.org/10.1109/TSC.2017.2771280.
[8] Song, M., Hua, Z., Zheng, Y., Xiang, T., & Jia, X., "Enabling Transparent Deduplication and Auditing for Encrypted Data in Cloud", IEEE Transactions on Dependable and Secure Computing, 21, 3545-3561, 2024. https://doi.org/10.1109/TDSC.2023.3334475.
[9] Wang, L., Wang, B., Song, W., & Zhang, Z., "A key-sharing based secure deduplication scheme in cloud storage", Inf. Sci., 504, 48-60, 2019. https://doi.org/10.1016/j.ins.2019.07.058.
[10] Fan, Y., Lin, X., Liang, W., Tan, G., & Nanda, P., "A secure privacy preserving deduplication scheme for cloud computing", Future Gener. Comput. Syst., 101, 127-135, 2019. https://doi.org/10.1016/j.future.2019.04.046.
[11] G. Crihan, M. Crăciun, and L. Dumitriu, "A comparative assessment of homomorphic encryption algorithms applied to biometric information," Inventions, vol. 8, no. 4, 102, 2023. https://doi.org/10.3390/inventions8040102.
[12] Ryu, J., Kim, K., & Won, D., "A Study on Partially Homomorphic Encryption", 2023 17th International Conference on Ubiquitous Information Management and Communication (IMCOM), 1-4, 2023. https://doi.org/10.1109/IMCOM56909.2023.10035630.
[13] Alaya, B., Laouamer, L., & Msilini, N. (2020). Homomorphic encryption systems statement: Trends and challenges. Comput. Sci. Rev., 36, 100235. https://doi.org/10.1016/j.cosrev.2020.100235.
[14] Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F., & Aaraj, N., "Survey on Fully Homomorphic Encryption, Theory, and Applications", Proceedings of the IEEE, 110, 1572-1609, 2022. https://doi.org/10.1109/JPROC.2022.3205665.
[15] J. S. Rauthan, "Fully homomorphic encryption: A case study," Journal of Intelligent & Fuzzy Systems, vol. 43, no. 6, pp. 8417–8437, 2022. https://doi.org/10.3233/JIFS-221454.
[16] Bharath Babu, S., & J. R., "Secure deduplication with dynamic updates in multi-tenant cloud environment", 2022 International Conference on Advanced Computing Technologies and Applications (ICACTA), 1-4, 2022. https://doi.org/10.1109/ICACTA54488.2022.9752987.
[17] Fan, Y., Lin, X., Liang, W., Tan, G., & Nanda, P., "A secure privacy-preserving deduplication scheme for cloud computing", Future Generation Computer Systems, 101, 127-135, 2019. https://doi.org/10.1016/j.future.2019.04.046.
[18] Yang, X., Lu, R., Shao, J., Tang, X., & Ghorbani, A., "Achieving efficient and privacy-preserving multi-domain big data deduplication in cloud", IEEE Transactions on Services Computing, 14(6), 1292-1305, 2018. https://doi.org/10.1109/TSC.2018.2881147.
[19] Goyal, N., Pandey, A. K., Gupta, S. K., & Pandey, R., "Suppleness of multi-tenancy in cloud computing", ERN: Other Econometrics, 2019. https://doi.org/10.2139/ssrn.3358249.
[20] Wang, J., He, J., Li, W., Lan, X., Liu, Q., & Li, T., "A secure duplicate data sharing method against untrusted cloud service providers", 2023 IEEE 12th International Conference on Cloud Networking (CloudNet), 352-359, 2023. https://doi.org/10.1109/CloudNet59005.2023.10490032.
[21] Novković, B., Bozic, A., Golub, M., & Groš, S., "Confidential computing as an attempt to secure service provider's confidential client data", 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), 1213-1218, 2021. https://doi.org/10.23919/MIPRO52101.2021.9597198.
[22] Shri, M., & Srinivas, P., "Enhanced protection deduplicate data storage scheme for multiuser environments in cloud", International Journal of Advance Research in Science and Engineering, 2138-2142, 2017.
[23] M. El Ghazouani, A. Ikidid, C. Ait Zaouiat, L. Aziz, M. Y. Ichahane, and L. Er-Rajy, "Optimal Method Combining Blockchain and Multi-Agent System to Ensure Data Integrity and Deduplication in the Cloud Environment", Int. J. Interact. Mob. Technol., vol. 18, no. 10, pp. pp. 90–105, May 2024. https://doi.org/10.3991/ijim.v18i10.43305.
[24] S. Madasu, P. Murugesan, H. V. Jaganathan and S. Pamulaparthyvenkata, "Elliptic Curve Diffie-Hellman based Privacy-Preserving Deduplication for Big Data in Cloud Systems," 2024 International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 2024, pp. 1-4, https://doi.org/10.1109/IACIS61494.2024.10721723.
[25] Rashid, F., Miri, A., & Woungang, I., "A secure data deduplication framework for cloud environments", 2012 Tenth Annual International Conference on Privacy, Security and Trust, 81-87, 2012. https://doi.org/10.1109/PST.2012.6297923.
[26] T. N. P. Madhuri, M. S. Rao, P. S. Santosh, P. Tejaswi, and S. Devendra, "Data Communication Protocol using Elliptic Curve Cryptography for Wireless Body Area Network," *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 133–139, Mar. 2022, https://doi.org/10.1109/ICCMC53470.2022.9753898.