

A Secure Content-Fingerprint Based Model for Redundancy-Aware Data Placement in Public Cloud Repositories

OPEN ACCESS

Received: 25/05/2026

Accepted: 17/07/2026

Published: 05/08/2026

Citation: Uma G (2026) A Secure Content-Fingerprint Based Model for Redundancy-Aware Data Placement in Public Cloud Repositories. Indian Journal of Science and Technology 19(28): 2042-2049. <https://doi.org/10.17485/IJST/V19I28.824>

* Corresponding author.

venkataraman_uma@yahoo.co.in

Funding: None

Competing Interests: None

Copyright: © 2026 Uma. This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment ([iSee](#))

ISSN

Print: 0974-6846

Electronic: 0974-5645

G Uma^{1*}

¹ Department of Computer Science, Srimad Andavan Arts & Science College (Autonomous), Affiliated to Bharathidasan University, Trichy, Tamil Nadu, India

Abstract

Objectives: To develop a Secure Content-Fingerprint Based Model (SCFBM) for reducing repeated file placement while preserving protected file handling and ownership validation. **Method:** SCFBM was implemented in C#.NET on Microsoft Azure using five benchmark file sizes from 5 MB to 25 MB. The evaluation used computational time, duplicate-upload storage allocation, internal-control checks, and comparison with recent secure deduplication approaches. **Findings:** SCFBM completed secure placement in 130 ms, 250 ms, 377 ms, 498 ms, and 589 ms for 5 MB, 10 MB, 15 MB, 20 MB, and 25 MB files, respectively. Against SecDedup and AC-Dedup, the average computational-time reduction was 40.13% and 11.13%. In repeated-upload trials involving two users with identical file content, storage allocation was reduced by 50%. The results agree with recent evidence that secure deduplication requires ownership verification, key management, and integrity-aware placement, but SCFBM contributes a simpler implementation workflow that connects content-token creation, key-support mapping, encrypted processing, tag verification, and access validation. **Novelty:** SCFBM combines token-level metadata screening, protected-output tag verification, and ownership validation in one redundancy-aware cloud placement workflow.

Keywords: Content fingerprinting; Public cloud; Redundancy-aware placement; Encrypted data; Ownership verification

1 Introduction

Public cloud repositories are widely used for enterprise backup, institutional document repositories, e-learning content storage, research-data sharing, and collaborative file management because storage capacity can be expanded without local infrastructure investment^(1,2). In such shared environments, identical files can be submitted by different users or departments, and ordinary encrypted storage can make repeated-content detection difficult because the same plaintext may produce different ciphertexts under different user keys^(1,3). The existing studies on secure deduplication have evolved beyond mere file matching to provide both data reduction and ownership proof, as well as auditing, access control, dynamic revocation, and key-management facilities⁽³⁻¹⁷⁾.

Although recent milestones reveal some advances in the field, they are accompanied by some outstanding challenges. For example, the scheme SecDedup, combining encrypted deduplication and dynamic auditing, offers protection against attacks of duplicate file creation, fake ownership assertion, and malicious modifications; however, its focus on audits results in additional complexities in routine operations⁽⁴⁾. Another challenge is posed by the schemes proposed by Wang et al.⁽⁵⁾ and Dave et al.^(11,12) for strengthening ownership proof in deduplicated storage; however, they mainly emphasize proof management, not an end-to-end upload-and-storage process. Access control of mobile cloud storage is provided by the scheme AC-Dedup, which is accompanied by difficulties in policy handling in routine operations⁽⁶⁾. Tang et al.⁽³⁾ suggest a dual-layer scheme for access control and key management using hybrid clouds; meanwhile, dualDup focuses on key deduplication of encrypted data⁽⁷⁾. Dynamic data re-encryption, auditing, and authentication schemes by Yuan et al.⁽⁹⁾, Zheng et al.⁽¹⁰⁾, Wang et al.⁽¹³⁾, and Periasamy et al.⁽¹⁴⁾ also demonstrate advancements in the field.

Thus, according to the current literature, secure deduplication can be implemented successfully, but there are still three practical challenges remaining. First, many studies are optimized for just one security service, whether it is dynamic auditing, access control, dynamic revocation, or ownership proof, and not for the entire placement process. Second, many schemes depend on additional trusted components or heavier security operations that may be unnecessary for small and medium public-cloud repositories. Third, repeated-content handling must not automatically create access rights; repository savings must be coupled with user-level verification. These gaps motivate the present work, which is positioned for public-cloud repositories used in institutional backup, academic file sharing, e-learning content stores, and enterprise document repositories where repeated content is common and access control remains essential.

To address these limitations, this study proposes a Secure Content-Fingerprint Based Model (SCFBM) for redundancy-aware data placement. The model creates a content token before encrypted processing, maps the token with a key-support value, produces a verification tag from the protected output, verifies the tag at the repository, and permits later access only after ownership-based validation. Thus, the model addresses three problems: repeated storage allocation, uncontrolled reference creation, and fragmented processing between content identification and authorization. The novelty lies in linking token-based metadata screening, protected-output tag verification, key-support mapping, and proof-based access validation within one cloud placement workflow.

Objectives for this research are: (i) develop a fingerprint-based algorithm that will reduce redundant storage of files in public cloud storages; (ii) implement tokenizing and tagging techniques to identify redundancies in metadata and outputs of sensitive documents; (iii) utilize encryption technology along with ownership checks; and (iv) evaluate the proposed model through computational time, resource usage, control mechanism, and comparative analysis with previous cloud data deduplication studies. Other sections of this paper describe methodology, experimental approach, results, discussion, and conclusion.

2 Methodology

The procedure was then updated to present the system as a cloud security experiment model rather than just an instructional guide for the process itself. The entire process is illustrated through Figure 1. The model involves five stages, all linked together, namely content token creation, mapping support, encryption file processing, repository validation with tags, and access verification according to ownership rights. This process leads to either uploading the content, where the data becomes an uploaded file, or an uploaded reference if it is a duplicate.

2.1 Data source, sample size and experimental parameters

For the experiment, five file-size categories ranging from 5 MB to 25 MB were generated locally and used as benchmark inputs. For each file-size category, one unique upload and one duplicate upload were executed under two different authenticated user profiles. Thus, the evaluation included five unique placement trials and five duplicate-placement trials. Generic binary payloads were used to avoid application-specific compression or file-format effects and to keep the file-size condition uniform across all experiments. The purpose of this setup was to validate the core placement behaviour of SCFBM, namely token lookup, tag lookup, protected-file storage, controlled reference creation, and ownership validation. Since the experiment uses controlled file sizes and two-user duplicate trials, it should be interpreted as a prototype-level validation of the proposed workflow rather than a production-scale benchmark (Table 1). Larger datasets, heterogeneous file formats, higher duplicate ratios, concurrent uploads, and real multi-user cloud workloads are considered as future evaluation extensions. The benchmark package may be provided as supplementary material to support reproducibility.

The selected experimental setting was intentionally controlled to verify whether the proposed workflow correctly distinguishes between a new protected file instance and a duplicate submission from another authenticated user. This design allows the repository decision to be traced clearly from token generation to final storage or reference creation. However, the

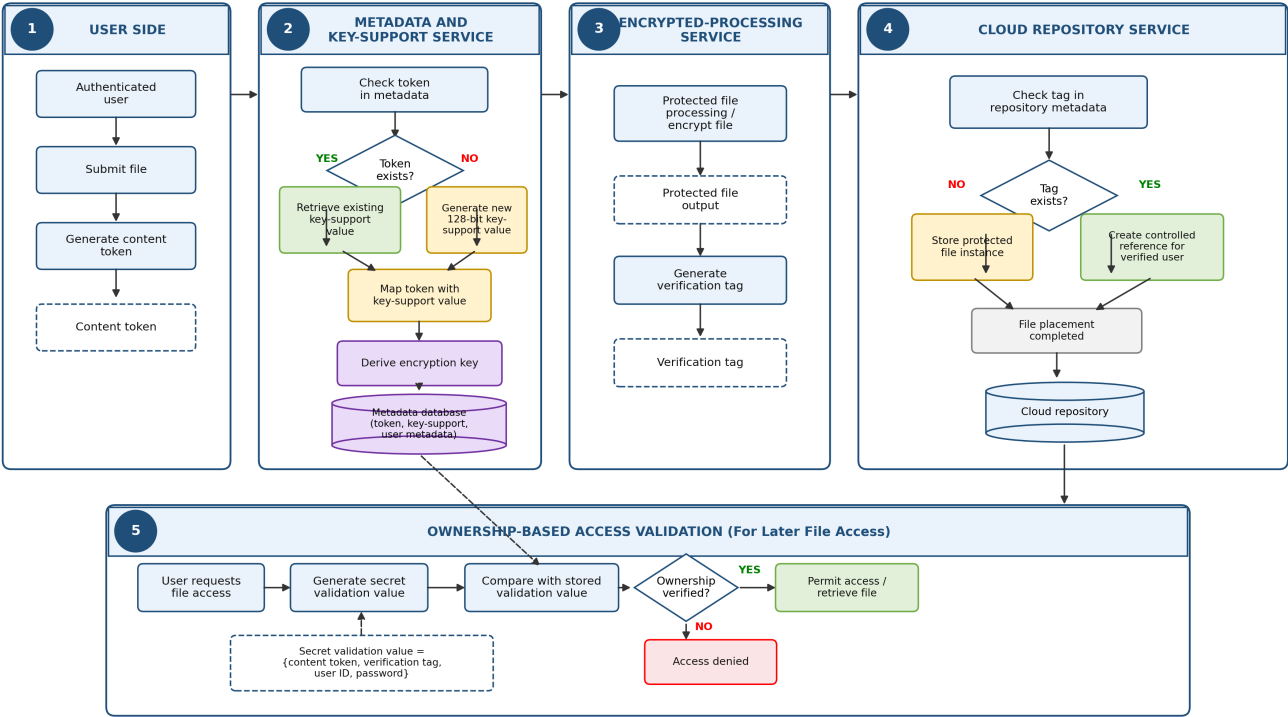


Fig 1. Proposed secure content-fingerprint based cloud placement workflow.

Table 1. Experimental dataset and parameter configuration

Parameter	Configuration used in the study
Implementation platform	C#.NET application deployed on Microsoft Azure
Cloud instance	Windows Server 2008 micro instance, 30 GB storage, 1 GB RAM
Sample size	Five file-size classes; one unique upload and one duplicate upload per size class
File sizes	5 MB, 10 MB, 15 MB, 20 MB, 25 MB
User condition	Two authenticated users for duplicate-upload validation
Measured metric	Computational time in milliseconds from token generation to repository decision
Storage control	Ordinary repeated storage without SCFBM
Comparison methods	SecDedup ⁽⁴⁾ and AC-Dedup ⁽⁶⁾ using identical file-size scale and execution-time criteria
Validation checks	Token lookup, tag lookup, protected-file placement, controlled reference creation, ownership validation

setup does not cover all production conditions, such as mixed file formats, simultaneous upload requests, large-scale user groups, and varying duplicate distributions. Therefore, the results are used to demonstrate feasibility and initial performance, while broader workload testing is identified as a required future extension.

2.2 Baseline controls and validation standards

Three controls were used to improve reproducibility. First, all timing observations used the same file-size scale and the same endpoint definition: elapsed time from content-token generation to final repository decision. Second, storage allocation was tested against ordinary repeated storage, where two users uploading the same file would consume two complete storage allocations. Third, the duplicate-upload condition was validated by submitting the same file content from a second authenticated user and checking whether SCFBM created a reference rather than a second stored object. SecDedup⁽⁴⁾ and AC-Dedup⁽⁶⁾ were

used as recent baseline methods for computational-time comparison because they represent auditing-oriented and access-control-oriented secure deduplication, respectively.

2.3 Proposed SCFBM workflow

SCFBM begins when an authenticated user submits a file. The content-token generator produces a compact content token from the input file and sends the token to the metadata and key-support service. If the token is already present, the previously mapped key-support value is retrieved. If the token is new, a fresh 128-bit key-support value is generated and mapped with the token. The encrypted-processing service then derives the encryption key, produces the protected file output, and passes that output to the tag-generation stage. The repository service checks the tag against stored repository metadata. A new tag results in storage of a protected file instance, while an existing tag results in a controlled reference for the validated user. Later access requests are verified using an ownership-validation value derived from the content token, verification tag, user ID, and password.

2.4 Algorithmic modification and processing logic

The proposed modification is the separation of early content screening and post-encryption repository verification. Earlier secure deduplication designs often rely on one dominant mechanism such as proof of ownership, dynamic auditing, re-encryption, or access-policy control^(3–10). SCFBM separates the process into two lightweight identifiers: a content token for early metadata screening and a verification tag for protected-output checking. This design reduces unnecessary repeated placement while still ensuring that the duplicate decision is made after protected processing.

Algorithm 1. Secure content-fingerprint based cloud placement

Input: File F , user identity U , password credential P

Output: Store protected file instance or create controlled reference

1. Authenticate U and receive F .
2. Generate content token T from F using binary conversion and repeated XOR reduction.
3. Search T in the metadata service.
4. If T exists, retrieve the mapped key-support value; otherwise generate a new 128-bit key-support value and map it with T .
5. Derive the encryption key and create protected file output EF .
6. Generate verification tag G from EF using binary reduction and XOR-based compaction.
7. Search G in repository metadata.
8. If G is new, store EF as a protected file instance; otherwise create a controlled reference for U .
9. For later access, compute $\text{Encoded}(T, G, U, P)$ and compare it with the stored validation value. Permit retrieval only when both values match.

2.5 Content-token, tag and ownership validation

The content token is generated by converting the submitted file into ASCII and binary representations, padding the binary stream where required, dividing the stream into two equal parts, and applying repeated XOR reduction until the required compact token is obtained. This token is not treated as the stored file; it acts as a metadata-screening value. The verification tag is generated after encrypted processing. The protected output is converted into decimal and binary form, repeated binary patterns are reduced, and XOR operations are applied across blocks until the final tag size is reached. The tag is then used for repository-side duplicate checking.

Ownership authentication prevents someone from accessing the system just by claiming that the identical file is present. The ownership authentication value is derived by $\text{Encoded}(\{\text{content token, authenticity, user id, password}\})$ and compared against the pre-stored authentication value. A proof-of-ownership authentication control must be enforced in deduplication-based cloud storage systems; otherwise, the deduplication process will enable hackers to gain access to stored content^(15,16).

2.6 Security and reproducibility considerations

SCFBM maintains confidentiality through protected file processing, enhances storage capacity through repository selection based on tags, and manages access control by confirming ownership. Dynamic auditing, blockchain auditing, and revocation management have not been incorporated into the model at this time, but their integration is planned because current studies

have proven that they are vital in managing dynamic cloud data and ownership^(9–14). Reproducibility involves using identical files, the same time boundary, same cloud environment, and identical conditions for duplicates upload.

2.7 Threat Model and Security Analysis

The proposed SCFBM assumes a public cloud repository where authenticated users may upload identical or different files, and the cloud service may receive repeated placement requests from multiple users. The threat model considers three major risks: false duplicate claims, replay of previously observed validation values, and unauthorized ownership assertion. In a false duplicate claim, a malicious user may attempt to obtain a reference to an already stored file without genuinely possessing the same content. SCFBM reduces this risk by separating content-token screening from protected-output tag verification and by permitting later access only after ownership-based validation. The content token alone is not sufficient to retrieve data; the user must also pass validation using the token, verification tag, user identity, and password-derived credential.

Collision attacks are also considered because token and tag values are compact representations generated from file content and protected output. A collision may occur if two different files produce the same token or tag. SCFBM reduces the practical effect of such a collision by using two levels of verification: early token-based metadata screening and post-encryption tag-based repository checking. Even if a token-level collision occurs, the final placement decision is checked again using the verification tag generated from the protected file output. For stronger deployment, the token and tag generation stages may be strengthened by integrating standard cryptographic hash functions such as SHA-256 or SHA-3 as an additional verification layer.

Replay attacks may occur when an attacker attempts to reuse previously captured validation information to gain access. SCFBM limits this risk by deriving the access-validation value from user-specific credentials along with the content token and verification tag. Therefore, a stored validation value is not treated as a universal access token. However, in large-scale deployment, replay resistance can be further improved by adding nonce-based challenge-response verification, timestamp checking, and session-bound access tokens.

Malicious ownership claims are handled through proof-based access validation. A user is not allowed to access stored content only because the repository identifies a duplicate tag. Instead, the system creates a controlled reference and verifies ownership before retrieval. This prevents duplicate detection from becoming an unauthorized access channel. The proposed model therefore supports confidentiality, duplicate-aware placement, and ownership-based access control. Nevertheless, the present implementation does not include dynamic auditing, blockchain-based logging, or revocation-aware re-encryption. These security extensions are considered future enhancements for large-scale multi-user cloud deployment.

3 Results and Discussion

3.1 Computational time analysis

Computational time was measured for file sizes from 5 MB to 25 MB. The recorded time includes token generation, key-support mapping, encrypted processing, tag generation, repository verification, and final storage or reference decision. Table 2 compares SCFBM with SecDedup⁽⁴⁾ and AC-Dedup⁽⁶⁾.

Table 2. Computational time comparison of SCFBM with recent baseline methods

File size	SecDedup ⁽⁴⁾ (ms)	AC-Dedup ⁽⁶⁾ (ms)	SCFBM (ms)	Reduction vs SecDedup (%)	Reduction vs AC-Dedup (%)
5 MB	210	142	130	38.10	8.45
10 MB	418	280	250	40.19	10.71
15 MB	610	416	377	38.20	9.38
20 MB	815	548	498	38.90	9.12
25 MB	1027	689	589	42.65	14.51
Average	616	415	368.8	40.13	11.13

SCFBM recorded the lowest computational time for every file size. At 5 MB, SCFBM required 130 ms, compared with 210 ms for SecDedup and 142 ms for AC-Dedup. At 25 MB, SCFBM required 589 ms, compared with 1027 ms and 689 ms. The average time of SCFBM was 368.8 ms, giving average reductions of 40.13% over SecDedup and 11.13% over AC-Dedup. The stronger reduction against SecDedup is consistent with the difference in design emphasis: SecDedup includes dynamic auditing and attack-resistance mechanisms⁽⁴⁾, whereas SCFBM focuses on lightweight placement decision and ownership-controlled reference creation. The smaller but consistent advantage over AC-Dedup indicates that dynamic access-control support may introduce additional processing requirements in comparison with the proposed token-tag workflow⁽⁶⁾.

3.2 Storage allocation analysis

Storage allocation was evaluated using a duplicate-upload scenario involving two authenticated users submitting identical file content. Table 3 compares ordinary repeated storage with the proposed redundancy-aware placement decision. In ordinary repeated storage, each user upload is treated as a separate stored object. In SCFBM, the first upload is stored as a protected file instance, while the second upload with the same content is converted into a controlled reference after repository-side tag verification and ownership validation.

Table 3. Storage allocation with and without SCFBM

File size	With SCFBM (MB)	Without SCFBM (MB)	Storage reduction (%)
5 MB	5	10	50
10 MB	10	20	50
15 MB	15	30	50
20 MB	20	40	50
25 MB	25	50	50

The storage result confirms that SCFBM stores only one protected file instance when two authenticated users submit identical content. For example, in the 25 MB case, ordinary repeated storage requires 50 MB because two complete copies are stored. In contrast, SCFBM requires only 25 MB because the second user receives a controlled reference after validation. This produces a 50% storage reduction in the tested two-user duplicate-upload condition.

However, this result should be interpreted within the scope of the experiment. The 50% reduction is obtained because the test scenario contains two users uploading identical content. In real cloud repositories, the actual storage saving will depend on the duplicate ratio, number of users, file-size distribution, file type, and access pattern. If more users upload the same file, the relative saving may increase because only one protected instance is stored and additional users receive controlled references. If the duplicate ratio is low, the storage saving will be lower. Therefore, the present result validates the core duplicate-placement logic of SCFBM, while larger datasets and multi-user workload experiments are needed to measure production-scale storage efficiency.

3.3 Validation with recent literature

The discussion was expanded to compare the present findings with recent favoring and contrasting reports. Table 4 summarizes the relation between SCFBM and selected recent secure deduplication studies.

Table 4. Literature validation and positioning of SCFBM

Study	Major focus	Relevant limitation/gap	Relation to SCFBM
Peng et al. ⁽⁴⁾	Encrypted deduplication with dynamic auditing	Audit-heavy design may increase processing cost	SCFBM is lighter but lacks dynamic auditing
Wang et al. ⁽⁵⁾	Auditing and ownership management	Strong ownership focus, less emphasis on upload-to-placement sequence	SCFBM adds token-tag placement workflow
Qi et al. ⁽⁶⁾	Dynamic access control in mobile cloud storage	Access-policy handling increases complexity	SCFBM uses simpler ownership validation
Tang et al. ⁽³⁾	Hybrid-cloud access control and key management	Requires private-cloud support	SCFBM is deployable as a public-cloud workflow
Yuan et al. ⁽⁹⁾	Efficient re-encryption for revocation	Focuses on ownership changes and re-encryption	SCFBM does not yet include revocation re-encryption
Zheng et al. ⁽¹⁰⁾	Integrity auditing with data dynamics	Designed for dynamic audit support	SCFBM can adopt auditing as future extension
Dave et al. ^(11,12)	Ownership verification for deduplicated systems	Main contribution is proof mechanism	SCFBM uses proof as part of end-to-end placement
Periasamy et al. ⁽¹⁴⁾	Cryptographic authentication and deduplication efficiency	Authentication and security mechanisms are broader	SCFBM provides a narrower but implementable placement model

The comparison shows that the present result is favorable where an implementation-oriented placement workflow is required. It is also consistent with studies emphasizing ownership control^(5,11,12,15,16) and key/security handling^(3,7,14). At the

same time, SCFBM does not contradict audit-oriented studies; rather, it addresses a different deployment goal. SecDedup and DIADD are stronger for dynamic auditing^(4,10), and re-encryption based designs are stronger for user revocation⁽⁹⁾. SCFBM contributes a compact workflow for detecting repeated content before final allocation, reducing storage use, and preventing duplicate detection from becoming an unauthorized access channel.

3.4 Functional comparison

Table 5 gives a concise feature-level comparison. The table is intended to position SCFBM and not to claim superiority in every security dimension, because the compared studies differ in environment, dataset, and threat assumptions.

Table 5. Functional comparison with recent secure cloud deduplication studies

Method	Year	Encrypted storage	Access control	Ownership verification	Dynamic auditing	Key management	Redundancy-aware placement
SecDedup ⁽⁴⁾	2023	Yes	Limited	Yes	Yes	Yes	Yes
Ownership-auditing scheme ⁽⁵⁾	2023	Yes	Yes	Yes	Yes	Yes	Yes
AC-Dedup ⁽⁶⁾	2024	Yes	Yes	Yes	Limited	Yes	Yes
Lightweight hybrid scheme ⁽³⁾	2024	Yes	Yes	Limited	Limited	Yes	Yes
dualDup ⁽⁷⁾	2022	Yes	Limited	Limited	No	Yes	Yes
Efficient re-encryption scheme ⁽⁹⁾	2022	Yes	Yes	Yes	No	Yes	Yes
DIADD ⁽¹⁰⁾	2025	Yes	Limited	Yes	Yes	Yes	Yes
SCFBM	2026	Yes	Yes	Yes	No	Yes	Yes

The distinguishing element of SCFBM is not the introduction of an entirely new cloud-storage problem, but the integration of content-token screening, protected-output tag verification, key-support mapping, and ownership validation within one deployable workflow. This directly responds to the identified gap that existing methods often optimize particular security services without presenting a simple public-repository placement process.

3.5 Practical Applicability and Evaluation Limitation

The experimental results demonstrate the feasibility of SCFBM as a lightweight redundancy-aware placement workflow. However, the present evaluation is limited to five file-size categories and duplicate-upload validation involving two authenticated users. This setting is useful for validating the basic placement logic, but it does not fully represent real-world cloud repositories where thousands of users may upload mixed file types such as documents, images, videos, compressed archives, and database backups. Therefore, the present results should be interpreted as prototype-level validation rather than full production-scale benchmarking. A larger evaluation using heterogeneous datasets, different duplicate ratios, concurrent uploads, and real multi-user access patterns would provide stronger evidence of scalability and practical applicability.

4 Conclusion

This study proposed the Secure Content-Fingerprint Based Model (SCFBM) for redundancy-aware file placement in public cloud repositories. The model integrates content-token based early screening, key-support mapping, encrypted file processing, protected-output tag verification, repository-side placement decision, and ownership-based access validation in one workflow. The experimental results showed that SCFBM completed secure placement in 130 ms, 250 ms, 377 ms, 498 ms, and 589 ms for 5 MB, 10 MB, 15 MB, 20 MB, and 25 MB files, respectively. The average computational time was 368.8 ms, showing reductions of 40.13% over SecDedup and 11.13% over AC-Dedup under the same file-size scale. In duplicate-upload validation with two authenticated users, SCFBM reduced storage allocation by 50% by storing one protected file instance and creating a controlled reference for the second verified user. These results confirm that the model can reduce repeated storage while maintaining controlled access. However, the present evaluation is limited to five file-size categories and two-user duplicate-upload testing in a controlled cloud setting. It does not fully represent large-scale repositories with heterogeneous file formats, concurrent uploads, and varying duplicate ratios. The current version also does not include formal cryptographic proof, nonce-based replay prevention, dynamic auditing, or revocation-aware re-encryption. Future work will extend the evaluation to larger datasets, real

multi-user workloads, mixed file types, and stronger security mechanisms such as hash-based collision resistance, challenge-response verification, dynamic integrity auditing, and blockchain-assisted audit trails.

5 Acknowledgements

NIL

Author details: Dr. G. Uma serves as an Associate Professor

References

- 1) Prajapati P, Shah P. A review on secure data deduplication: cloud storage security issue. *Journal of King Saud University - Computer and Information Sciences*. 2022;34(7):3996–4007. <https://doi.org/10.1016/j.jksuci.2020.10.021>.
- 2) Sabeerath K, Manikandasaran SS. BTEDD: block-level tokens for efficient data deduplication in public cloud infrastructures. *The Scientific Temper*. 2024;15(3):2507–2514. <https://doi.org/10.58414/SCIENTIFICTEMPER.2024.15.3.16>.
- 3) Tang X, Guo C, Choo KKR, Jiang X, Liu Y. A secure and lightweight cloud data deduplication scheme with efficient access control and key management. *Computer Communications*. 2024;222:209–219. <https://doi.org/10.1016/j.comcom.2024.05.003>.
- 4) Peng L, Yan Z, Liang X, Yu X. SecDedup: secure data deduplication with dynamic auditing in the cloud. *Information Sciences*. 2023;644:119279. <https://doi.org/10.1016/j.ins.2023.119279>.
- 5) Wang M, Xu L, Hao R, Yang M. Secure auditing and deduplication with efficient ownership management for cloud storage. *Journal of Systems Architecture*. 2023;142:102953. <https://doi.org/10.1016/j.sysarc.2023.102953>.
- 6) Qi S, Wei W, Wang J, Sun S, Rutkowski L, Huang T, et al. Secure data deduplication with dynamic access control for mobile cloud storage. *IEEE Transactions on Mobile Computing*. 2024;23(4):2566–2582. <https://doi.org/10.1109/TMC.2023.3263901>.
- 7) Chouhan V, Peddoju SK, Buyya R. dualDup: a secure and reliable cloud storage framework to deduplicate the encrypted data and key. *Journal of Information Security and Applications*. 2022;69:103265. <https://doi.org/10.1016/j.jisa.2022.103265>.
- 8) Balaji K, Manikandasaran SS. Data security and deduplication framework for securing and deduplicating users' data in public and private cloud environment. *Journal of Scientific Research*. 2022;14(1):153–165. <http://dx.doi.org/10.3329/jsr.v14i1.54063>.
- 9) Yuan H, Chen X, Li J, Jiang T, Wang J, Deng RH. Secure cloud data deduplication with efficient re-encryption. *IEEE Transactions on Services Computing*. 2022;15(1):442–456. <https://doi.org/10.1109/TSC.2019.2948007>.
- 10) Zheng X, Shen W, Su Y, Gao Y. DIADD: secure deduplication and efficient data integrity auditing with data dynamics for cloud storage. *IEEE Transactions on Network and Service Management*. 2025. <https://doi.org/10.1109/TNSM.2025.3535708>.
- 11) Dave J, Ram KR, Patil P, Patil H, Borole S, Panda S. Secure and efficient ownership verification for deduplicated cloud computing systems. *Journal of Cloud Computing*. 2025;14(1):20. <https://doi.org/10.1186/s13677-025-00743-y>.
- 12) Dave J, Ram KR, Patil P, Patil H, Borole S, Chinni VK, et al. Blind and bidirectional ownership verification for deduplicated cloud computing systems. *Future Generation Computer Systems*. 2026;175:108082. <https://doi.org/10.1016/j.future.2025.108082>.
- 13) Wang C, Zhang G, Qi H, Chen B. A CAE model-based secure deduplication method. *Scientific Reports*. 2025;15(1):24605. <https://doi.org/10.1038/s41598-025-09788-0>.
- 14) Periasamy JK, Prabhakar S, Vanathi A, Yu L. Enhancing cloud security and deduplication efficiency with SALIGP and cryptographic authentication. *Scientific Reports*. 2025;15:30112. <https://doi.org/10.1038/s41598-025-14972-3>.
- 15) Dave J, Bhatt M, Pancholi D. Secure proof of ownership for deduplicated cloud storage system. *International Journal of Information and Computer Security*. 2023;21(1/2):205–228. <https://doi.org/10.1504/IJICS.2023.131097>.
- 16) Huang K, Zhang X, Mu Y, Rezaeiabagha F, Du X. Bidirectional and malleable proof-of-ownership for large file in cloud storage. *IEEE Transactions on Cloud Computing*. 2022;10(4):2351–2365. <https://doi.org/10.1109/TCC.2021.3054751>.
- 17) Sabeerath K, Manikandasaran SS. ESPoW: efficient and secured proof of ownership method to enable authentic deduplicated data access in public cloud storage. *The Scientific Temper*. 2024;15(4):3165–3172. <https://doi.org/10.58414/SCIENTIFICTEMPER.2024.15.4.25>.