Secure and Efficient Deduplication for IoT Data in Fog-Assisted Cloud Storage Based on Server-Aided Encryption and Data Locality

Rongxi Wang, Guanxiong Ha, Chunfu Jia, Zhaoyang Liang, and Zhen Su

***Abstract*—Mobile cloud computing is confronting the challenge of an explosive growth in data. This challenge is particularly acute in Internet of Things (IoT) applications, where the massive volumes of data generated place significant pressure on real-time services and storage. To manage massive data more effectively, fog-assisted cloud storage has garnered increasing attention for its capability to provide real-time services. Encrypted deduplication can eliminate duplicate encrypted data, which not only saves storage costs but also preserves data privacy, and thus has high practical value. However, existing encrypted deduplication schemes for fog-assisted cloud storage still have limitations in terms of security and performance. In terms of security, the existing schemes cannot simultaneously resist brute-force attacks (BFAs), side-channel attacks (SCAs), and duplicate faking attacks (DFAs). In terms of performance, the fog nodes in the existing schemes suffer a heavy storage burden, and the fog nodes cannot perform cross-user data deduplication, resulting in high communication overhead.**

This paper proposes a secure and efficient deduplication scheme SEDedup for IoT data in fog-assisted cloud storage. First, we design a server-aided encryption method tailored for fog-assisted cloud storage. Our method enables cross-user data deduplication at the fog nodes to reduce the system's communication overhead, while effectively preventing BFAs, SCAs, and DFAs. Compared with conventional server-aided encryption, our method eliminates the need for IoT devices to interact directly with the remote key server, and the key server does not need to be continuously online. Moreover, we analyze multiple real-world datasets and observe that IoT data exhibits strong data locality. Based on this observation, we develop an epoch-based duplicate detection method, achieving efficient duplicate detection while reducing the storage overhead of fog nodes. SEDedup is further enhanced with generalized deduplication to support similarity-based deduplication, boosting storage efficiency. Security analysis and performance evaluation demonstrate that, compared with the existing state-of-the-art schemes, SEDedup not only enhances security but also exhibits better performance.

***Index Terms*—Internet of Things data, Fog-assisted cloud storage, Encrypted deduplication, Server-aided encryption, Data locality.**

I. Introduction

In recent years, the rapid development of Internet of Things (IoT) technology has led to the widespread adoption of IoT systems. To meet the demand for low-latency data sharing, mobile cloud computing has been proposed [1]–[3], enabling users to outsource their data to the cloud for storage. To cope with the massive volume of data outsourced by users, many prominent companies (e.g., AT&T [4], Verizon [5], and Huawei [6]) have built *fog-assisted cloud storage* to effectively manage IoT data. In a fog-assisted cloud storage system, IoT devices are deployed at the edge of the network, collecting IoT data and transferring data to remote cloud servers via nearby fog nodes (FNs). A FN is deployed for the devices in the same domain, offering real-time service for devices. The adoption of fog-assisted cloud storage has gained significant popularity as it provides advantages such as low latency and location awareness [7]–[9].

Due to the cyclical and continuous nature of IoT device work, there is a significant amount of redundant content in IoT data [9]. For example, in electronic health (eHealth) systems, patients consulting doctors from the same department generate a vast amount of duplicate electronic medical records (EMRs) [10]. The large volume of duplicate IoT data results in severe wastage of storage space. Deduplication is an effective way to identify and remove duplicate data, keeping only one physical copy for all duplicate data to save storage space [11]–[14]. To preserve data privacy, various encrypted deduplication schemes have been developed [15]–[19]. They perform deduplication on encrypted data, achieving both privacy preservation and storage savings. Bellare *et al.* [15] propose message-locked encryption (MLE) to implement encrypted deduplication. It generates keys based on data content, allowing identical plaintexts to be encrypted into the same ciphertext for encrypted deduplication. Unfortunately, MLE is vulnerable to brute-force attacks (BFAs) [16], [20]. Server-aided encryption [16], [20]–[22] introduces a third-party key server (KS) to defend against BFAs. Subsequently, Liu *et al.* [17] achieve encrypted deduplication without third-party servers based on the password authenticated key exchange (PAKE).

Nevertheless, the majority of existing encrypted deduplication schemes are designed for cloud storage with a two-layer architecture consisting of a cloud server and devices, while fog-assisted cloud storage contains a three-layer storage model that includes a cloud server, FNs, and devices. The different architectures result in different design considerations. Compared to encrypted deduplication in cloud storage, the

This work is supported by National Key R&D Program of China (2018YFA0704703) and National Natural Science Foundation of China (62172238). (Corresponding author: Chunfu Jia.)

Rongxi Wang, Guanxiong Ha, Chunfu Jia, Zhaoyang Liang, and Zhen Su are with the College of Cryptology and Cyber Science, Nankai University, Tianjin Key Laboratory of Network and Data Security Technology, and Key Laboratory of Data and Intelligent System Security (DISSec), Tianjin, 300350, China (e-mail: wrx@mail.nankai.edu.cn; hgx1995@mail.nankai.edu.cn; cfjia@nankai.edu.cn; budongjishubu@gmail.com; Suuu826@outlook.com).

counterpart in fog-assisted cloud storage requires additional considerations for *cross-domain deduplication* [23]–[25] and *the limited resources of* FNs [26]. Over the past years, impressive efforts have been devoted to designing encrypted deduplication schemes for fog-assisted cloud storage [7]–[9], [27]–[30]. Fu *et al.* [27] achieve encrypted deduplication for health records based on a fog-assisted cloud storage model. Xie *et al.* [31] propose an encrypted deduplication scheme that provides multi-level security protection for data in fog-assisted cloud storage. These schemes [27], [31] require the device to constantly interact with the remote KS, resulting in high communication complexity. Zhang *et al.* [8] devise a secure deduplication scheme in the fog-assisted cloud storage that resists BFAs and supports cross-domain data deduplication. But their scheme does not support cross-user deduplication, while the duplicate data across different users can lead to heavy storage burden for the system. Gao *et al.* [30] propose an encrypted deduplication scheme based on the relationship between IoT devices in fog-assisted cloud storage. However, their scheme cannot resist BFAs, side-channel attacks (SCAs) [32], [33], and duplicate faking attacks (DFAs) [15]. FCDedup [7] implements both cross-user and cross-domain encrypted deduplication without requiring devices to interact with any remote third-party server, which is the state-of-the-art encrypted deduplication scheme. However, it still has the following limitations in terms of security and performance.

- **Security:** FCDedup is vulnerable to SCAs and DFAs. A malicious device can determine whether data exists in the cloud server and upload inconsistent tags and ciphertexts to break the data integrity.
- **Performance:** FCDedup uses a large number of computationally expensive exponentiations and pairings, resulting in low computation efficiency. Furthermore, in FCDedup, cross-user data deduplication is performed by the cloud server, and FNs are unable to execute cross-user data deduplication. This results in the transmission of duplicate data across users, incurring significant communication overhead.

Additionally, most existing schemes do not take into account the issue of limited storage resources on FNs. A large number of data tags used for duplicate detection can impose a heavy storage burden on FNs. Although Shin *et al.* [26] mitigate the storage overhead of FNs, the introduction of the inefficient PAKE [34] significantly degrades the system performance.

A. Contribution

In this paper, we propose a secure and efficient deduplication scheme SEDedup for IoT data in fog-assisted cloud storage. First, we design a *server-aided encryption method* tailored for fog-assisted cloud storage to enable FNs to perform cross-user data deduplication, which improves the system's communication efficiency. Meanwhile, based on this server-aided encryption method, SEDedup can effectively defend against BFAs, SCAs, and DFAs. Compared to the conventional server-aided encryption [16], [18], [35], [36], our encryption method has the following two advantages. First, KS in our method is only involved during the system initialization and can go offline once the system initialization is complete. The data encryption and upload process does not require the involvement of KS, thereby eliminating the need for frequent interactions between the device and KS. Second, IoT devices do not need to communicate directly with the remote KS. Devices only need to interact with KS through FNs, which avoids the overhead introduced by the device communicating with remote entities.

Furthermore, through analyzing multiple real-world datasets, we observe that *IoT data exhibits strong spatial locality and temporal locality*. The spatial locality indicates that there is a large amount of duplicate data within the same domain, requiring cross-user data deduplication to be performed at the FNs to reduce the system's communication overhead. The temporal locality indicates that there is a large amount of duplicate data within the same epoch. Leveraging the temporal locality of IoT data, we develop an *epoch-based duplicate detection method*, achieving efficient deduplication while reducing the storage burdens of FNs. In our method, FNs maintains only two components: a temporary index that stores data tags within the current epoch, and a cuckoo filter CF that stores tags from all previous epochs. FNs can achieve efficient duplicate detection while storing only a small amount of data. In addition, by integrating *generalized deduplication*, SEDedup enables lossless encrypted deduplication for similar IoT data to improve storage efficiency.

In summary, we make the following contributions:

- We propose SEDedup, a secure and efficient cross-user encrypted deduplication scheme in fog-assisted cloud storage. We design an improved server-aided encryption method, which enables SEDedup to resist BFAs, SCAs, and DFAs simultaneously.
- We devise an epoch-based duplicate detection method based on the temporal locality of IoT data. Based on the temporary index along with the CF, FNs can efficiently detect duplicate data while storing only a small amount of data. In addition, by integrating generalized deduplication, SEDedup can implement lossless encrypted deduplication for similar IoT data, improving storage efficiency.
- We provide a detailed security analysis and comprehensive performance evaluations. The results demonstrate that compared with the existing state-of-the-art schemes, SEDedup not only achieves higher security but also exhibits better performance.

B. Paper Organization

The remainder of this paper is organized as follows. In Section II, we provide the background and motivation for our work, while exploring the locality of IoT data. Section III introduces the preliminaries for this paper. The overview of SEDedup, including the system architecture and threat model, is given in Section IV. In Section V, we present the design of SEDedup, including the design challenges and solutions, and the detailed constructions. The security analysis and performance evaluation are provided in Section VI and

TABLE I: Comparison of SEDedup with the existing encrypted deduplication schemes for fog-assisted cloud storage.

Scheme	Similarity-based deduplication	Cross-domain deduplication	Resistance to BFAs, SCAs and DFAs	Efficient storage of FNs
Xie *et al.* (2023) [31]	✗	✗	✓	✗
Shin *et al.* (2020) [23]	✗	✓	✗	✗
Fu *et al.* (2021) [27]	✗	✓	✓	✗
Zhang *et al.* (2022) [8]	✓	✓	✗	✗
Song *et al.* (2023) [7]	✗	✓	✗	✗
Gao *et al.* (2025) [30]	✓	✓	✗	✗
SEDedup	✓	✓	✓	✓

Section VII, respectively. We conclude this paper in Section VIII.

II. Background and Motivation

This section first introduces conventional encrypted deduplication schemes in cloud storage, and then discusses existing encrypted deduplication schemes in fog-assisted cloud storage, and subsequently analyzes the temporal and spatial locality of IoT data using real-world datasets. The limitations of existing schemes serve as the motivation for our work.

A. Encrypted Deduplication in Cloud Storage

Deduplication saves storage overhead by identifying and removing duplicate data. Encrypted deduplication combines encryption algorithms with deduplication to deduplicate encrypted data, thereby enabling efficient storage and privacy preservation. MLE [15] achieves efficient encrypted deduplication by generating encryption keys based on data content, but it is unable to defend against BFAs [16]. Keelveedhi *et al.* [16] propose server-aided encryption to resist BFAs, wherein the key is derived from both the data content and a master key managed by a key server. As long as the master key is secure, offline BFAs can be prevented. Moreover, server-aided encryption can mitigate online BFAs by enforcing rate limiting on key requests. Subsequently, several encrypted deduplication schemes [23], [35], [37], [38] based on server-aided encryption were proposed. But these schemes are designed for two-layer cloud storage and are not suitable for the three-layer fog-assisted cloud storage.

B. Encrypted Deduplication for Fog-assisted Cloud Storage

Several encrypted deduplication schemes have been proposed for fog-assisted cloud storage in recent years. Xie *et al.* [31] present a secure deduplication scheme for edge-assisted cloud storage systems to resist frequency analysis attacks. Fu *et al.* [27] introduce a fog-to-multicloud cooperative storage model to achieve secure deduplication. But these schemes [27], [31] require continuous interaction between devices and the remote KS to generate keys, resulting in high communication complexity. Shin *et al.* [23] and Yang *et al.* [25] implement cross-domain encrypted deduplication. Nevertheless, these schemes [23], [25] require all data owners within the same domain to share a common secret parameter. As stated in [7], it is not feasible for owners to share the same parameters. Zhang *et al.* [8] achieve cross-domain deduplication and similarity-based deduplication, but it does not support cross-user deduplication, which degrades the communication and storage efficiency. Gao *et al.* [30] propose an encrypted deduplication scheme for fog-assisted cloud storage based on the relationship of IoT devices. However, it is vulnerable to BFAs, SCAs, and DFAs. Song *et al.* [7] propose FCDedup, which supports inter-deduplication across different data owners and has better performance compared to [8]. Nevertheless, FCDedup still has the following limitations.

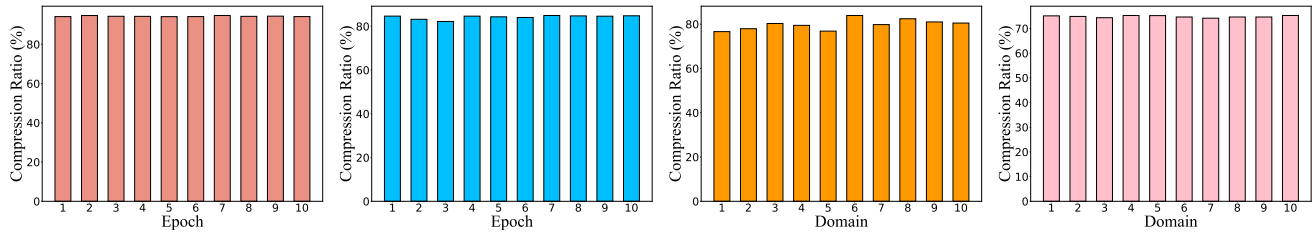
- **Security.** FCDedup is vulnerable to SCAs and DFAs. It employs inconsistent processing flows for duplicate and non-duplicate data, which allows an adversary to infer whether specific data already exists on the cloud server by observing these processing variations. Furthermore, in FCDedup, if the device that uploads data for the first time uploads inconsistent data tags and ciphertext to the cloud server to launch DFAs,the cloud server will perform deduplication based on the inconsistent data tags and delete the subsequently uploaded data. Consequently, the cloud server stores only the inconsistent ciphertext, which renders the data unrecoverable for the subsequent uploaders, thereby compromising data integrity.
- **Performance.** FCDedup relies on extensive bilinear pairings and exponentiation operations, resulting in high computational overhead. Moreover, the requirement to store a large volume of data tags imposes a significant storage burden on the FNs. Besides, FCDedup overlooks the potential for cross-user data duplication at the fog node level, executing cross-user deduplication exclusively on the cloud server. This design leads to high communication overhead, as a large volume of cross-user duplicate data within the same region needs to be uploaded by FN to the cloud server. We verify this in Section II-C.

Table I shows the comparison of SEDedup and several representative encrypted deduplication schemes for fog-assisted cloud storage. SEDedup achieves similarity-based deduplication, cross-domain deduplication, resistance to SCAs and DFAs, and efficient storage of FNs simultaneously for the first time.

C. Locality of IoT Data

We explore the temporal and spatial locality of IoT data, which serves as important design rationales for SEDedup. We analyze data locality using several real-world IoT datasets as outlined below:

- **CMU-MMAC** [39]: This dataset is collected in Carnegie Mellon's Motion Capture Lab. It contains multimodal



(a) CMU-MMAC (b) TSPD (c) ECA&D (d) ASHRAE

Fig. 1: Compression ratios for real-world IoT datasets.

measures of the human activity of subjects performing the tasks involved in cooking and food preparation.

- **Thermal-solar-plant-dataset (TSPD)** [40]: This dataset provides real-time thermal solar plant data logged from 2016-12-28 to 2018-10-10.
- **ECA&D** [41]: This dataset comprises a collection of daily observations from meteorological stations located across Europe and the Mediterranean region.
- **ASHRAE** [42]: This dataset comprises raw data from the last two decades of thermal comfort field studies around the world.

We split the data in the datasets into bases and deviations using generalized deduplication [8], [43]. The details of generalized deduplication are provided in Section III-C. Similar yet distinct data can yield the same base and different deviations. We only deduplicate the bases because there are more duplicates in the bases and fewer duplicates in the deviations. We compute the compression ratio $cr = 1 - \frac{Dv_a}{Dv_b}$, where Dv_a and Dv_b respectively represent the data volume after and before deduplication. The higher the compression ratio, the higher the storage efficiency after deduplication.

Temporal locality. The data in CMU-MMAC and TSPD carry temporal information. So, we use them to assess the temporal locality. The epoch length is set as 5 minutes for CMU-MMAC and 1 day for TSPD. We perform deduplication on the data within the same epoch and compute the compression ratio. As shown in Fig.1(a) and Fig.1(b), consecutive 10 epochs of CMU-MMAC exhibit compression ratios surpassing 80%, whereas those of TSPD surpass 90%. We can observe the strong temporal locality of IoT data. To put it differently, *there is a significant amount of duplicate IoT data within the same epoch*. The temporal locality of IoT data motivates us to design an *epoch-based duplicate detection method* to save storage costs of FNs.

Spatial locality. The data in ECA&D and ASHRAE carry spatial information. So, we use them to assess the spatial locality. We select data from 10 domains in ECA&D and ASHRAE, respectively, for evaluation. Deduplication is conducted on the data within each domain, and the compression ratio is calculated. As depicted in Fig.1(c) and Fig.1(d), the respective compression ratios for the 10 domains in ECA&D and ASHRAE are approximately 80% and 75%. We can observe the strong spatial locality of IoT data. In other words, *there is a significant amount of duplicate IoT data within the same domain*. The spatial locality of IoT data motivates us to *perform cross-user deduplication for the data in the same domain at* FN*s* to reduce the communication overhead.

III. PRELIMINARIES

A. Cuckoo Filter

A CF [44] is a space-efficient probabilistic data structure used for checking if a data item is a member of a data set. CF may have false positive matches, but false negatives are not possible. CF supports the item insertion, constant-time membership queries, and item deletion.

- $\mathrm{CF.Insert}(x)$: It takes a data item x as input and outputs the CF with x inserted.
- $\mathrm{CF.Check}(x)$: It takes a data item x as input and outputs a flag $f \in \{0, 1\}$, indicating whether x is present in the filter CF. If $f = 1$, it means that CF contains x. If $f = 0$, it means that CF does not include x.
- $\mathrm{CF.Delete}(x)$: It takes a data item x as input and outputs the CF with x removed.

B. Digital Signature

A digital signature is utilized for verifying the authenticity and integrity of digital messages. A digital signature scheme consists of the following three algorithms:

- $\mathrm{Gen_S}(\lambda)$: The key generation algorithm takes a security parameter λ as input and outputs a signature key pair (pk_S, sk_S).
- $\mathrm{Sign_S}(sk_S, m)$: The signature algorithm takes the secret key sk_S and a message $m \in \{0, 1\}^*$ as input and outputs a signature σ.
- $\mathrm{Vrfy_S}(pk_S, m, \sigma)$: The verifying algorithm takes the public key pk_S, the message m, and the signature σ as input and outputs a flag $f \in \{0, 1\}$. If $f = 1$, σ is valid. Otherwise, σ is invalid.

C. Generalized Deduplication

Generalized deduplication [8], [43] is a lossless compression technique applicable to similar data. It employs a transformation function ϕ to transform the data M into a base b and a deviation d. ϕ can map similar yet distinct data into the same base and different deviations. We follow the approach in [43] to implement generalized deduplication. Suppose the message

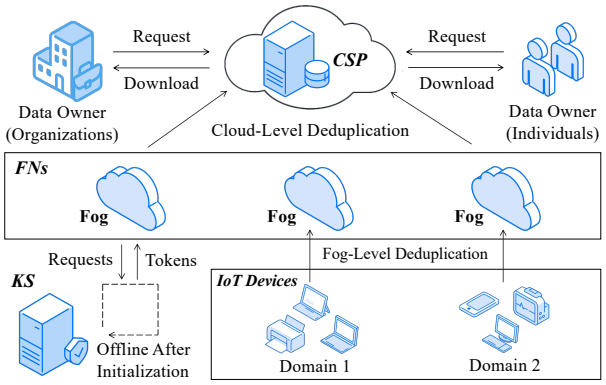


Fig. 2: Architecture of SEDedup.

space is the finite file $\mathbb{F}_2^n$ (the set of binary vectors with length n). ϕ transforms the data $M \in \mathbb{F}_2^n$ into $\phi(M) \triangleq (b, d) = (M_0^{k-1}, M_k^{n-1})$. The first k bits are taken as the base b, while the remaining $n-k$ bits are taken as the deviation d. The inverse function ϕ^{-1} concatenates b and d: $M \leftarrow \phi^{-1}(b, d) \triangleq [b_0, b_1, ..., b_{l_b-1}, d_0, d_1, ..., d_{l_d-1}]$, where l_b and l_d denote the length of b and d, respectively. In order to map similar data to the same base with high probability, we need to construct a permutation π to relocate the indices in the permutation set $I = \{i_1, i_2, ..., i_{n-k}\}$ to the end of M, i.e., $\overline{M} \leftarrow \pi_I(M) \triangleq [..., M_{i_1-1}, M_{i_1+1}, ..., M_{i_2-1}, M_{i_2+1}, ..., M_{i_{n-k}-1}, M_{i_{n-k}+1}, ..., M_{i_1}, M_{i_2}, M_{i_{n-k}}]$. Then, b and d become $\overline{M}_0^{k-1}$ and $\overline{M}_k^{n-1}$, respectively. When recovering the original data M, after concatenating b and d to construct $\overline{M}$, we move the last $n-k$ elements of $\overline{M}$ to their correct positions based on I.

IV. Overview

In this section, we present the system architecture and threat model of SEDedup.

A. System Architecture

Fig. 2 shows the system architecture, which consists of the data owner (D_O), IoT devices, fog nodes (FNs), the cloud service provider (CSP), and the key server (KS).

- *Data owner*: D_O is an individual or organization that deploys multiple IoT devices to collect data. D_O employs CSP to manage collected IoT data.
- *IoT devices*: IoT devices are deployed by D_O and are responsible for collecting IoT data and uploading encrypted data to CSP via FNs.
- *Fog nodes*: FNs are deployed at the edge of networks, serving as intermediaries between IoT devices, KS and CSP. FNs receive data uploaded by IoT devices within their domains, perform inter-owner deduplication (i.e., cross-user deduplication), and only upload the non-duplicate data to CSP to reduce communication overhead.
- *Cloud service provider*: CSP provides data storage services to data owners. It receives data uploaded by FNs and performs inter-deduplication across different FNs to reduce storage overhead.
- *Key server*: KS manages a system-level master key. When initializing an IoT device, KS utilizes the master key to assist FN in generating a token for the device, which is used to achieve secure cross-user encrypted deduplication. After the initialization, KS can go offline.

TABLE II: Major notations used in this paper.

Notation	Description
msk	Master key
pk_{KS}, sk_{KS}	RSA key pair of KS
spk, ssk	Signing key pair of D_O
dsk_i/k_{D_O}	Secret key of D_i/D_O
Δ_i	Token of D_i
β_{D_O}	Secret value for D_O
$\mathrm{Enc}_{\mathrm{SE}}/\mathrm{Dec}_{\mathrm{SE}}$	Symmetric encryption/decryption algorithm
M_i	IoT data
b_i/d_i	Base/deviation of M_i
T_{b_i}	Base tag
I_T, I_F	Temporary index, full index
$k_{b_i}, C_{k_{b_i}}$	Base key, encrypted base key
C_{b_i}, C_{d_i}	Encrypted base, encrypted deviation
h_{b_i}	Hash of base b_i

B. Threat Model

We assume that KS is semi-trusted, similar to the assumption for the key server in existing server-aided encryption [16], [35], [37]. KS honestly follows our predefined protocols, securely stores the master key, and does not collude with other entities. If the master key is compromised by adversaries, the security of SEDedup will degrade to convergent security [15]. The potential security threats are as follows.

- *Honest-but-curious CSP and FNs*: CSP and FNs honestly execute our protocol but are curious about the data contents uploaded by IoT devices. They can arbitrarily access the data stored in them and attempt to launch BFAs to learn plaintext contents.
- *Malicious IoT devices*: Unlike existing schemes [7], [8], we consider threats from the malicious IoT device. It may launch SCAs to infer the storage status of data on the CSP. Specifically, the malicious device uploads data tags to FN and determines whether the data have already been uploaded by another D_O's device through the duplicate response returned by FN. Furthermore, the malicious IoT device may initiate DFAs to break the data integrity of legitimate users. Specifically, it uploads inconsistent tags and ciphertexts to CSP.

We assume that the communication between entities is encrypted and authenticated (e.g., via TLS). Furthermore, following the assumption in existing schemes [7], [8], we assume that the CSP does not collude with the D_O or FNs. This is because, as stated in [45], such collusion could pose significant risks to the CSP's reputation and lead to civil liability or criminal investigation [46]. However, FNs are vulnerable to being compromised by adversaries; therefore, we assume that FNs may collude with the D_O.

V. Design

This section begins by introducing the design challenges and solutions of SEDedup. Subsequently, we provide the detailed construction of SEDedup. The major notations used in this paper are summarized in Table II.

A. Design Challenges and Solutions

Here, we present several challenges (abbreviated as $\mathcal{C}$) that need to be addressed and provide our solutions.

$\mathcal{C}1$: How to achieve efficient cross-user encrypted deduplication while being able to resist BFAs, SCAs, and DFAs? Most of the existing schemes [16], [35], [38] achieve secure cross-user encrypted deduplication based on key transmission among data owners [7], [34] or server-aided encryption [16]. The former incurs high computation overhead and requires data owners to remain online, resulting in poor practicality. Moreover, due to the significant communication overhead caused by frequent interactions between devices and a remote KS, the conventional server-aided encryption is also difficult to apply to fog-assisted cloud storage.

To this end, we design *an improved server-aided encryption* suitable for fog-assisted cloud storage to achieve efficient cross-user encrypted deduplication while defending BFAs, SCAs, and DFAs. We deploy a key server KS that manages a system-level master key msk. When initializing a new IoT device D for a data owner D_O within the domain of the fog node FN, FN interacts with D, KS and CSP to generate a token $\Delta \leftarrow \frac{msk \cdot \beta_{D_O}}{dsk}$ for D, where dsk is the secret key of D, and β_{D_O} is the secret value for data owner D_O generated by CSP. During key generation, D interacts with FN and CSP. FN and CSP assist D in generating an encryption key based on the token Δ. The encryption key is generated based on msk and data content, so devices deployed by different data owners can produce the same encryption key and ciphertext when encrypting identical plaintext. Meanwhile, server-aided encryption enables FNs in SEDedup to perform cross-user data deduplication. As stated in Section II-C, real-world data exhibits strong spatial locality, so there is a large amount of duplicate data within the same domain. Performing cross-user data deduplication at FNs can significantly reduce the communication cost from FNs to CSP, thus improving the communication efficiency of the system.

Furthermore, to resist DFAs, data tag is generated by FN based on the ciphertext, which ensures the consistency between the ciphertext and the tag. To prevent SCAs, SEDedup implements two-stage deduplication [47]. That is, for inter-owner duplicate data (i.e., cross-user duplicate data), SEDedup performs target-based deduplication, requiring devices to upload the whole data content. For intra-owner duplicate data, SEDedup performs source-based deduplication, devices do not need to upload duplicate data. Therefore, adversaries cannot use the duplicate responses returned by FNs to infer the data existence information of other owners. Besides, due to the introduction of the master key msk, FN or CSP cannot launch BFAs for ciphertext to learn data information.

Our server-aided encryption only requires KS to be involved in the initialization of new devices, while the key generation does not need the involvement of KS, which avoids the frequent interactions with KS. Moreover, our method does not require devices to communicate directly with KS. Devices, KS and CSP collaboratively generate tokens via FNs, avoiding complex communication between devices and remote entities.

$\mathcal{C}2$: How to achieve efficient duplicate detection while reducing the storage overhead of FNs? Inspired by the temporal locality of IoT data (Section II-C), we develop *an epoch-based duplicate detection method*. SEDedup evolves over epochs. During epoch e_i, FN only stores a temporary index I_T that stores data tags within e_i, and a CF that stores tags from all previous epochs. CSP stores a complete index I_F that records data tags from all epochs and a CF that records information of all tags. Upon receiving a tag T, FN first retrieves T from I_T. Due to the temporal locality of IoT data, there is a large amount of duplicate data within the same epoch. This characteristic of IoT data ensures that FN has a high probability of identifying duplicates in I_T. If T cannot be found in I_T, FN checks if CF contains T. If it does, FN submits T to CSP, which retrieves T in the full index I_F for duplicate detection. Otherwise, this indicates that there is no duplicate, and the device needs to upload the data. After each data upload, FN and CSP update I_T and I_F for subsequent duplicate detection, respectively. At the end of the epoch, FN deletes CF and updates I_T, and CSP inserts all tags in the updated I_F into CF to generate an updated cuckoo filter CF'. Based on I_T and CF, FN can achieve efficient duplicate detection while storing only a small amount of data.

$\mathcal{C}3$: How to deduplicate highly similar but distinct IoT data? By introducing generalized deduplication [8], [43], SEDedup implements encrypted deduplication for similar IoT data. Specifically, the device D leverages generalized deduplication to transform the collected IoT data into bases and deviations. Similar but different data will be divided into the same base and different deviations. Here, for simplicity, we assume that all IoT data is of the same type. When dealing with multiple types of IoT data, we can perform generalized deduplication for each type of data separately to obtain the base and deviation [8]. Bases contain a majority of information and have a large amount of duplicates, whereas deviations have a smaller size and very few duplicates. To this end, we adopt the improved server-aided encryption for bases, which enables secure deduplication, thus reducing storage overhead. While we deploy the randomized encryption for deviations and do not perform deduplication on deviations. Therefore, we implement lossless encrypted deduplication for similar IoT data.

B. Construction

The construction of SEDedup is as follows.

Initialization. KS takes a security parameter λ as input and generates a multiplicative cyclic group $\mathbb{G}$ with generator g and prime order q. KS then generates a master key $msk \in \mathbb{Z}_q$ and an RSA key pair (pk_{KS}, sk_{KS}). KS selects the hash functions $\mathrm{H}_1 : \{0,1\}^* \rightarrow \{0,1\}^\lambda$, $\mathrm{H}_2 : \{0,1\}^* \rightarrow \mathbb{G}$. KS publicizes the public parameter $\mathrm{pp} \leftarrow \{q, G, g, \mathrm{H}_1, \mathrm{H}_2\}$ and its public key pk_{KS}. The master key msk and secret key sk_{KS} are stored in KS securely. Following registration with the CSP, D_O deploys IoT devices to collect IoT data. The CSP subsequently generates a secret value $\beta_{D_O} \in \mathbb{Z}_q$ for D_O and ensures its secure storage within the CSP. D_O generates a signature key pair $(spk, ssk) \leftarrow \mathrm{Gen_S}(\lambda)$ and a secret key $k_{D_O} \leftarrow \{0,1\}^\lambda$. For its device D_i, D_O selects a secret key

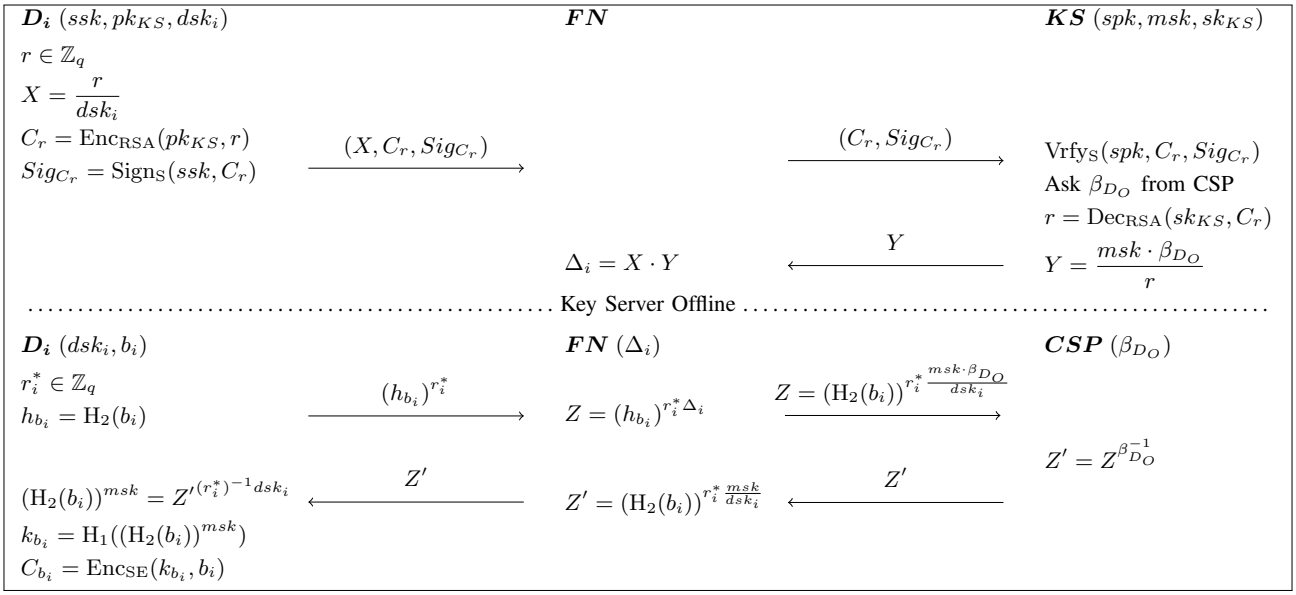


Fig. 3: Initialization, key generation, and data encryption

$dsk_i \in \mathbb{Z}_q$, inputs (ssk, k_{D_O}, dsk_i) into D_i. D_O manages (k_{D_O}, ssk) and publicizes spk. When initializing a new device D_i within the domain of the fog node FN, FN interacts with D_i, CSP and KS to generate a token Δ_i for D_i by following steps (see Fig. 3):

- D_i chooses a random value $r \in \mathbb{Z}_q$, computes $X \leftarrow \frac{r}{dsk_i}$, $C_r \leftarrow \mathrm{Enc_{RSA}}(pk_{KS}, r)$, and a signature $Sig_{C_r} \leftarrow \mathrm{Sign_S}(ssk, C_r)$. Then, it sends (X, C_r, Sig_{C_r}) to FN.
- FN forwards (C_r, Sig_{C_r}) to KS, which computes $f \leftarrow \mathrm{Vrfy_S}(spk, C_r, Sig_{C_r})$. If $f = 0$, KS aborts. Otherwise, KS informs CSP of the new device registration. CSP checks whether D_O is registered with it. If D_O is not registered, the device registration is rejected. If D_O is registered, CSP sends the secret value β_{D_O} to the KS. Subsequently, KS decrypts $r \leftarrow Dec_{RSA}(sk_{KS}, C_r)$ and returns $Y \leftarrow \frac{msk \cdot \beta_{D_O}}{r}$ to FN.
- FN computes a token $\Delta_i \leftarrow X \cdot Y = \frac{msk \cdot \beta_{D_O}}{dsk_i}$ for D_i.

During the initialization, D_i collaborates with KS through FN and CSP to generate the token Δ_i. D_i does not need to interact directly with the remote KS, which reduces system communication overhead. Besides, from the perspective of security, as D_i has signed C_r, FN cannot use a forged C_r' to learn msk.

Key generation and data encryption (Fig. 3). Suppose that M_i is the IoT data collected by the device D_i. D_i first utilizes generalized deduplication to transform M_i into base b_i and deviation d_i. D_i encrypts b_i and d_i with server-aided encryption and standard symmetric encryption, respectively. D_i performs symmetric encryption with secret key k_{D_O} to get the encrypted deviation $C_{d_i} \leftarrow \mathrm{Enc_{SE}}(k_{D_O}, d_i)$ and computes a base hash $h_{b_i} \leftarrow \mathrm{H_2}(b_i)$. Subsequently, D_i runs the following key generation protocol with FN to get the base key k_{b_i}.

- D_i selects a random value $r_i^* \in \mathbb{Z}_q$, computes a blinded hash $(h_{b_i})^{r_i^*}$, and sends $(h_{b_i})^{r_i^*}$ to FN.
- FN computes $Z \leftarrow ((h_{b_i})^{r_i^*})^{\Delta_i} = (\mathrm{H_2}(b_i))^{r_i^* \frac{msk \cdot \beta_{D_O}}{dsk_i}}$ and sends Z to CSP.
- CSP computes $Z' \leftarrow Z^{\beta_{D_O}^{-1}} = ((h_{b_i})^{r_i^*})^{\Delta_i \cdot \beta_{D_O}^{-1}} = (\mathrm{H_2}(b_i))^{r_i^* \frac{msk}{dsk_i}}$ and sends Z' to D_i via FN.
- D_i computes $(\mathrm{H_2}(b_i))^{msk} \leftarrow Z'^{(r_i^*)^{-1} dsk_i}$ and gets the base key $k_{b_i} \leftarrow \mathrm{H_1}((\mathrm{H_2}(b_i))^{msk})$.

After obtaining k_{b_i}, D_i computes the encrypted base $C_{b_i} \leftarrow \mathrm{Enc_{SE}}(k_{b_i}, b_i)$ and the encrypted key $C_{k_{b_i}} \leftarrow \mathrm{Enc_{SE}}(k_{D_O}, k_{b_i})$. Note that the base key k_{b_i} contains only the base content and the master key. Therefore, when devices deployed by different data owners upload the same base, the same base key, and the same encrypted base will be generated, and then the cross-user encrypted deduplication for bases can be realized. If a malicious device frequently initiates key generation requests to FN for online BFAs, FN can perform rate limiting as a defense.

Duplicate detection and data upload. D_i computes a tag $T_{b_i} \leftarrow \mathrm{H_1}(C_{b_i})$ for base b_i and then uploads $(id_{D_O}, T_{b_i}, C_{d_i}, C_{k_{b_i}})$ to FN, where id_{D_O} denotes the id of D_O. FN performs the following duplicate detection based on T_{b_i}.

FN first retrieves T_{b_i} in the temporary index I_T, which records the tags within the current epoch along with the id of their data owners. Due to the temporal locality, most duplicate bases can be found in I_T. If $< T_{b_i}, id_{D_O} >$ can be found in I_T, it indicates that b_i has been uploaded by D_O and is an intra-owner duplicate. Then, FN sets the result of duplicate detection to $dr \leftarrow 1_I$. If only T_{b_i} can be found in I_T, it indicates that b_i has been uploaded by other data owners and is a cross-user duplicate. Then, FN sets $dr \leftarrow 1_C$. If I_T does not include T_{b_i}, FN executes $f \leftarrow \mathrm{CF.Check}(T_{b_i})$ to check if CF contains T_{b_i}. If $f = 0$, it indicates that the base b_i is non-duplicate, and FN sets $dr \leftarrow 0$. Otherwise, FN sends T_{b_i} to CSP for further duplicate detection.

Upon receiving T_{b_i}, CSP retrieves it in the full index

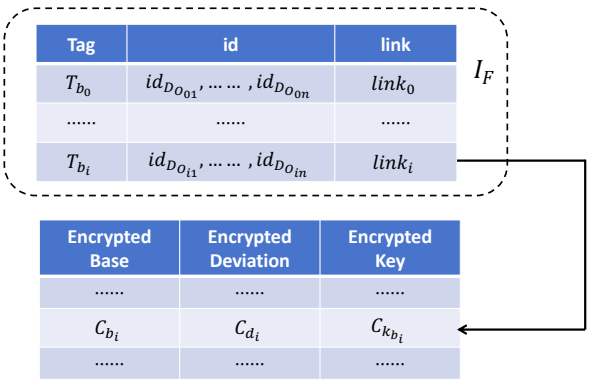


Fig. 4: Storage structure of CSP.

I_F, which records the tags in all previous epochs and their corresponding owner id. The storage structure of CSP is shown in Fig. 4. Similarly, if $< T_{b_i}, id_{D_O} >$ can be retrieved from I_F, CSP returns the duplicate response $dr = 1_I$ to FN. If only T_{b_i} can be retrieved from I_F, CSP returns $dr = 1_C$ to FN. If I_F does not include T_{b_i}, it indicates a false positive of CF occurred, and CSP returns $dr = 0$ to FN.

After the aforementioned duplication detection, FN can obtain the duplicate response dr. If $dr = 1_I$, then C_{b_i} is an intra-owner duplicate, FN only needs to upload C_{d_i} to CSP, which stores C_{d_i} for data recovery. If $dr = 1_C$ or $dr = 0$, FN requires D_i to upload C_{b_i}. If $dr = 0$, FN computes the tag $T_{b_i} \leftarrow \mathrm{H}_1(C_{b_i})$, records T_{b_i} and id_{D_O} in I_T, and uploads $(T_{b_i}, C_{b_i}, C_{d_i}, id_{D_O}, C_{k_{b_i}})$ to CSP. If $dr = 1_C$, FN records (T_{b_i}, id_{D_O}) in I_T and uploads $(id_{D_O}, C_{k_{b_i}}, C_{d_i})$ to CSP.

Data access. When D_O accesses M_i uploaded by its device D_i, it initiates a request to CSP. CSP retrieves the encrypted base C_{b_i}, encrypted deviation C_{d_i}, and encrypted key $C_{k_{b_i}}$, and then returns them to D_O. D_O decrypts $C_{k_{b_i}}$ and C_{d_i} using its secret key k_{D_O} to restore the key k_{b_i} and deviation d_i, and decrypts C_{b_i} with k_{b_i} to obtain b_i. Finally, D_O recovers the original data M_i with b_i and d_i.

VI. SECURITY ANALYSIS

In this section, we provide a security analysis for SEDedup based on our threat model (see Section IV-B).

A. Security Analysis for Token Generation

The security of token generation is based on the unforgeability of the digital signature and the semantic security of the RSA encryption algorithm.

Theorem 1. *If the signature scheme satisfies unforgeability and RSA is semantically secure, then the fog node FN cannot learn msk, β_{D_O} or dsk_i during token generation.*

Proof. During the token generation, FN receives $X \leftarrow \frac{r}{dsk_i}$, $C_r \leftarrow \mathrm{Enc}_{\mathrm{RSA}}(pk_{KS}, r)$, $Sig_{C_r} \leftarrow \mathrm{Sign}_{\mathrm{S}}(ssk, C_r)$, and $Y \leftarrow \frac{msk \cdot \beta_{D_O}}{r}$. Since RSA is semantically secure, FN cannot restore r from C_r, then it also cannot restore dsk_i and msk, β_{D_O} from X and Y. If FN attempts to select a random value $r' \in \mathbb{Z}_q$ and construst a fake RSA ciphertext $C'_r \leftarrow \mathrm{Enc}_{\mathrm{RSA}}(pk_{KS}, r')$, it cannot forge a valid signature $Sig_{C'_r} \leftarrow \mathrm{Sign}_{\mathrm{S}}(ssk, C'_r)$ since it has no knowledge of the signature key ssk. In summary, FN cannot learn msk, β_{D_O} or dsk_i during the token generation. □

B. Data Confidentiality

Here, we prove that the honest-but-curious CSP and FN cannot recover any data information.

Theorem 2. *If the symmetric encryption algorithm is semantically secure, the probability advantage that CSP or FN restores data information is negligible.*

Proof. During the data upload, both FN and CSP learn C_{d_i}, C_{b_i}, $C_{k_{b_i}}$, T_{b_i}. Since msk is kept confidential, neither CSP nor FN can learn the base key $k_{b_i} \leftarrow \mathrm{H}_1(\mathrm{H}_2(b_i)^{msk})$. C_{b_i}, C_{d_i}, $C_{k_{b_i}}$ are all symmetric ciphertexts, and the keys k_{b_i} and k_{D_O} are kept secret from CSP and FN. Therefore, if the symmetric encryption algorithm is semantically secure, the probability advantage that CSP or FN restores data information from $T_{b_i} \leftarrow \mathrm{H}_1(C_{b_i})$, C_{b_i}, C_{d_i} or $C_{k_{b_i}}$ is negligible. □

C. Attack Resistance

We prove that SEDedup can combat potential attacks launched by the adversary.

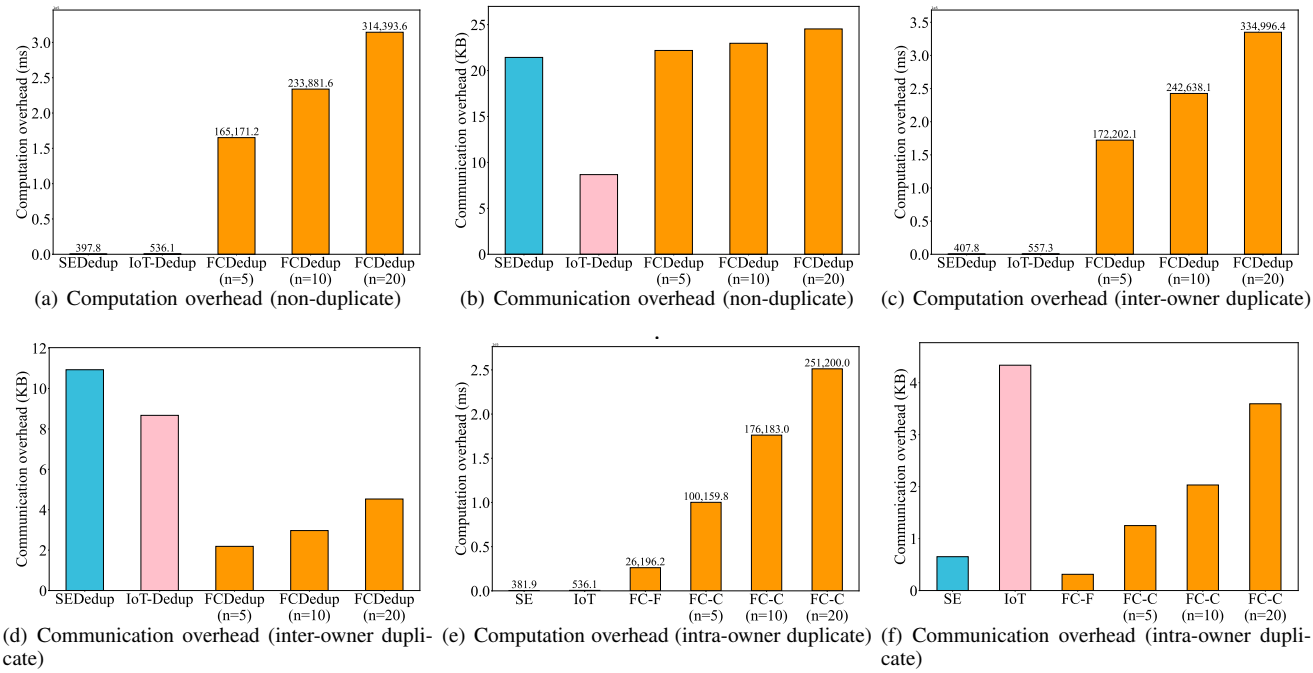
Side-channel attacks (SCAs). SEDedup implements two-stage deduplication to resist SCAs. Since FNs perform target-based deduplication on duplicate data across different owners, adversaries cannot determine the data existence information of other owners based on the results of duplicate detection, thereby effectively defending against SCAs.

Duplicate-faking attacks (DFAs). SEDedup performs cross-user deduplication on encrypted bases. A malicious device may upload inconsistent tags and encrypted bases to launch DFAs.

Theorem 3. *If the hash function H_1 is collision-resistant, SEDedup provides resistance to DFAs.*

Proof. In SEDedup, FN computes the hash of the encrypted base C_{b_i} as the data tag $T_{b_i} \leftarrow \mathrm{H}_1(C_{b_i})$ for duplicate detection. This can ensure the consistency between the tag and the ciphertext, preventing DFAs. If a malicious device uploads the inconsistent tag and encrypted base, successfully launching DFAs, then it needs to construct a pair of ciphertexts (C'_{b_i}, C_{b_i}) such that $C'_{b_i} \neq C_{b_i}$ and $\mathrm{H}_1(C'_{b_i}) = \mathrm{H}_1(C_{b_i})$, which breaks our assumption. □

Brute-force attacks (BFAs). The base key k_{b_i} is derived from both the data content and the master key msk. Since msk is kept secret, neither FN nor CSP can obtain a potential base key through BFAs. If a device initiates a large number of key generation requests in an attempt to obtain a large number of base keys, FN and CSP can perform rate limiting as a defense. Besides, the encrypted deviation C_{d_i} and encrypted key $C_{k_{b_i}}$ are encrypted with the secret key k_{D_O}. As k_{D_O} is kept secret, BFAs for C_{d_i} and $C_{k_{b_i}}$ can be prevented.



(a) Computation overhead (non-duplicate) (b) Communication overhead (non-duplicate) (c) Computation overhead (inter-owner duplicate)

(d) Communication overhead (inter-owner duplicate) (e) Computation overhead (intra-owner duplicate) (f) Communication overhead (intra-owner duplicate)

Fig. 5: Computation and communication overheads when a device uploads non-duplicate data, inter-owner duplicate data, or intra-owner duplicate data. SE and IoT denote SEDedup and IoT-Dedup, respectively. FC-F and FC-C denote the overhead of FNs and CSP performing intra-owner deduplication, respectively.

Collusion Attacks (CAs). An adversary may compromise IoT devices and a fog node FN within their domain simultaneously, thereby obtaining dsk and $\Delta = \frac{msk \cdot \beta_{D_O}}{dsk}$. The adversary attempts to use dsk and Δ to learn the data hash h', then recovers data information by performing BFAs on h'. Specifically, the adversary computes $msk \cdot \beta_{D_O} \leftarrow dsk \cdot \Delta$ from dsk and Δ and subsequently tries to recover h' from $(h')^{msk}$ using $msk \cdot \beta_{D_O}$. However, since the adversary cannot obtain the secret value β_{D_O}, it is unable to recover msk from $msk \cdot \beta_{D_O}$. Furthermore, given that the Discrete Logarithm Problem is assumed to have no feasible probabilistic polynomial-time solution, the adversary cannot recover h' from $(h')^{msk}$. Therefore, SEDedup is resistant to CAs between FNs and malicious devices.

VII. Evaluation

We implement a prototype of SEDedup based on OpenSSL [48], and GMP [49]. The prototype consists of IoT devices, fog nodes FNs, the key server KS, and the cloud service provider CSP. The hash function and symmetric encryption used in the prototype are SHA256 and AES-256, respectively. The signing algorithm is implemented based on RSA. The key length of RSA encryption is 2048 bits. Additionally, we implement the state-of-the-art encrypted deduplication schemes FCDedup [7] and IoT-Dedup [30] for performance comparison. Our experiments run on a machine equipped with a quad-core 2.5GHz Intel Core-i5-13400, 8GB RAM, and installed with 64-bit Ubuntu 20.04.6. The evaluation results in this section are the average of more than 100 runs.

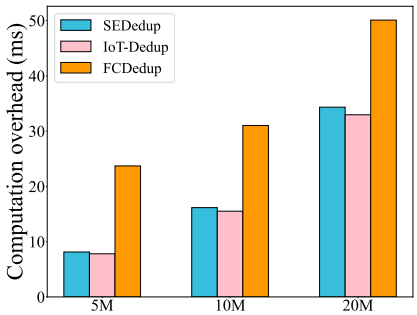
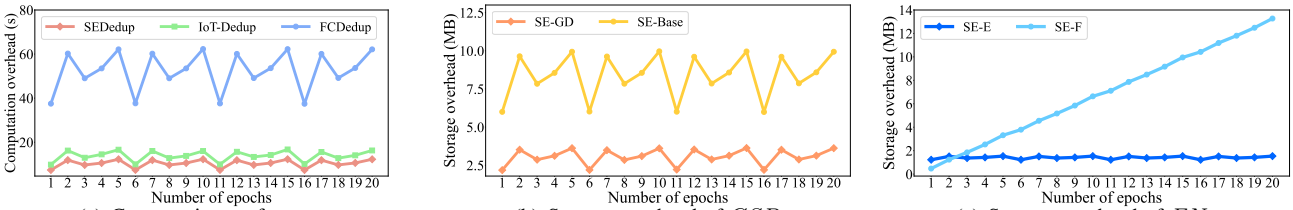


Fig. 6: Overhead of data access.

A. Evaluation on Synthetic Data

We use synthetic data to evaluate the computation and communication overheads. In FCDedup, the number n of tags matched by the same short hash affects performance. We evaluate the overhead of FCDedup when n is set to 5, 10, and 20. The size of the test file is set to 10 MB. Data uploads can be divided into three cases: uploading non-duplicate data ($dr = 0$), uploading inter-owner ($dr = 1_C$) duplicate data, and uploading intra-owner duplicate data ($dr = 1_I$). While IoT-Dedup supports both fuzzy and exact deduplication, our evaluation focuses exclusively on its exact deduplication performance to ensure fairness in the experiment.

Uploading non-duplicate data. Fig. 5(a) and Fig. 5(b) illustrate the computation and communication overheads when uploading non-duplicate data. SEDedup has significantly lower computation overhead compared to FCDedup and IoT-Dedup. This is because FCDedup performs multiple expensive exponentiation and pairing operations during data upload, whereas



(a) Computation performance (b) Storage overhead of CSP (c) Storage overhead of FNs

Fig. 7: Evaluation on real-world data. SE-Base and SE-GD denote the schemes before and after using generalized deduplication, respectively. SE-F denotes the scheme where FNs store all tags, while SE-E denotes the scheme where FNs only store tags from the current epoch.

SEDedup adopts efficient improved server-aided encryption. Moreover, FCDedup requires computationally expensive pairing operations, its prototype must be implemented based on the less efficient PBC [50]. In contrast, SEDedup only requires exponentiation operations, and its prototype is implemented based on the more efficient GMP. Therefore, this further increases the computational overhead of FCDedup. As n decreases, the overhead of FCDedup reduces. However, even when $n = 5$, SEDedup still achieves more than 400$\times$ speedups over FCDedup. Moreover, compared to FCDedup, SEDedup reduces the communication overhead significantly. This is because FCDedup requires the use of short hashes to retrieve some candidate tags during duplicate detection, and the transmission of these candidate tags introduces additional communication costs. The communication overhead of SEDedup is higher than that of IoT-Dedup because IoT-Dedup only transmits tags, which is vulnerable to BFAs, SCAs, and DFAs.

Uploading inter-owner and intra-owner duplicate data. Fig. 5(c) and Fig. 5(d) present the overhead of uploading inter-owner duplicates. SEDedup remains significantly faster than FCDedup in computation overhead. The reason is that, similarly, SEDedup reduces the number of exponentiation of pairing operations by employing improved server-aided encryption. SEDedup introduces higher communication overhead compared to FCDedup because inter-owner duplicates need to be uploaded to defend against DFAs and SCAs. In IoT-Dedup, the communication overhead remains unchanged because both inter-user duplicate data and non-duplicate data need be uploaded to the CSP.

Fig. 5(e) and Fig. 5(f) present the overhead of uploading intra-owner duplicates. In FCDedup, the intra-owner deduplication can be performed by FN or CSP. Because of the different costs, we use FC-F and FC-C to denote the overhead of FN and CSP performing intra-owner deduplication, respectively. When uploading intra-owner duplicates, benefiting from the efficient improved server-aided encryption, compared to FC-F, SEDedup reduces the computation overhead significantly. FC-C has a significantly higher computation overhead since it needs to convert all candidate tags to data tags for duplicate detection, whose overhead increases as n increases. Compared to FC-F, SEDedup introduces a 25% increase in communication overhead as it involves an additional step of generating server-aided encryption key. IoT-Dedup incurs the highest communication overhead because it still requires additional meta data to be fetched from the FN.

Data access. We evaluate the performance of data access using test files of various sizes. As shown in Fig. 6, compared to FCDedup, SEDedup improves computation performance by 31.5%-65.7%. This is because the key recovery in SEDedup is more efficient than FCDedup. Compared to SEDedup, IoT-Dedup only eliminates one key decryption operation; consequently, their total computation overhead of data access is similar.

B. Evaluation on Real-world Data

Computation performance. To simulate the realistic scenario, we assume there are 2 domains, with 2 data owners in each domain, and each owner deploys 2 devices. The data within the epoch is evenly distributed among these devices, which then upload the data to CSP. Fig. 7(a) presents the computation overhead for uploading data in 20 consecutive epochs. Similar to the evaluation on synthetic data, SEDedup exhibits better performance compared to FCDedup, since it reduces the use of computationally expensive exponentiations and pairings.

Storage efficiency. We first evaluate the improvement in storage efficiency through generalized deduplication. Fig. 7(b) shows the storage overhead before and after using generalized deduplication in SEDedup. Due to the presence of a large amount of highly similar yet distinct IoT data in the real-world dataset, utilizing generalized deduplication can significantly enhance storage efficiency. With generalized deduplication, the storage overhead of CSP can be reduced by 63.1-63.8%.

Then, we evaluate the storage overhead of FNs. We compare the storage overhead of FNs before and after using the epoch-based duplicate detection method. SE-F represents the scheme where FNs store all tags, while SE-E represents the scheme where FNs only store tags from the current epoch. As shown in Fig. 7(c), initially, the storage overhead of SE-E is higher because FN needs to maintain a CF, which introduces some additional storage overhead. However, as the epoch evolves, the storage overhead of SE-F increases linearly, while the overhead of SE-E remains stable. Compared to SE-F, SE-E has significantly lower storage overhead, which verifies the effectiveness of the epoch-based duplicate detection method.

VIII. Conclusion

In this paper, we propose SEDedup, a cross-user encrypted deduplication scheme for fog-assisted cloud storage. An im-

proved server-aided encryption for fog-assisted cloud storage is designed to achieve efficient c ross-user e ncrypted deduplication and provide resistance to BFAs, SCAs, and DFAs. Moreover, we develop an epoch-based duplicate detection method based on the temporal locality of IoT data, which enables SEDedup to implement efficient d uplicate detection while mitigating the storage burden of FNs. SEDedup can also be combined with generalized deduplication to realize lossless encrypted deduplication for similar data, improving storage efficiency. W e p rove t he s ecurity o f S EDedup and conduct the performance evaluation. Compared to the state-of-the-art schemes FCDedup and IoT-Dedup, SEDedup exhibits better performance.

REFERENCES

[1] A. ur Rehman Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 393–413, 2014.

[2] F. Liu, P. Shu, H. Jin, L. Ding, J. Yu, D. Niu, and B. Li, "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wirel. Commun.*, vol. 20, no. 3, pp. 1–0, 2013.

[3] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 337–368, 2014.

[4] "At&t," https://www.att.com.

[5] "Verizon," https://www.verizon.com/.

[6] "Huawei," :https://www.huawei.com/en.

[7] M. Song, Z. Hua, Y. Zheng, T. Xiang, and X. Jia, "Fcdedup: A two-level deduplication system for encrypted data in fog computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 10, pp. 2642–2656, 2023.

[8] C. Zhang, Y. Miao, Q. Xie, Y. Guo, H. Du, and X. Jia, "Privacy-preserving deduplication of sensor compressed data in distributed fog computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4176–4191, 2022.

[9] Y. Gao, L. Chen, J. Han, S. Yu, and H. Fang, "Similarity-based secure deduplication for iiot cloud management system," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 2242–2256, 2024.

[10] Y. Zhang, C. Xu, H. Li, K. Yang, J. Zhou, and X. Lin, "Healthdep: An efficient and secure deduplication scheme for cloud-assisted ehealth systems," *IEEE Trans. Ind. Informatics*, vol. 14, no. 9, pp. 4101–4112, 2018.

[11] Y. Shin, D. Koo, and J. Hur, "A survey of secure data deduplication schemes for cloud storage systems," *ACM Comput. Surv.*, vol. 49, no. 4, pp. 74:1–74:38, 2017.

[12] P. Prajapati and P. Shah, "A review on secure data deduplication: Cloud storage security issue," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, pp. 3996–4007, 2022.

[13] W. Xia, H. Jiang, D. Feng, F. Douglis, P. Shilane, Y. Hua, M. Fu, Y. Zhang, and Y. Zhou, "A comprehensive study of the past, present, and future of data deduplication," *Proc. IEEE*, vol. 104, no. 9, pp. 1681–1710, 2016.

[14] Q. Huang, Z. Zhang, and Y. Yang, "Privacy-preserving media sharing with scalable access control and secure deduplication in mobile cloud computing," *IEEE Trans. Mob. Comput.*, vol. 20, no. 5, pp. 1951–1964, 2021.

[15] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2013, pp. 296–312.

[16] S. Keelveedhi, M. Bellare, and T. Ristenpart, "Dupless: Server-aided encryption for deduplicated storage," in *Proceedings of the 22th USENIX Security Symposium, Washington, DC, USA, August 14-16, 2013*. USENIX Association, 2013, pp. 179–194.

[17] J. Liu, N. Asokan, and B. Pinkas, "Secure deduplication of encrypted data without additional independent servers," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, I. Ray, N. Li, and C. Kruegel, Eds. ACM, 2015, pp. 874–885. [Online]. Available: https://doi.org/10.1145/2810103.2813623

[18] Y. Ren, J. Li, Z. Yang, P. P. Lee, and X. Zhang, "Accelerating encrypted deduplication via sgx." in *USENIX Annual Technical Conference*, 2021, pp. 957–971.

[19] Z. Yang, J. Li, and P. P. Lee, "Secure and lightweight deduplicated storage via shielded deduplication-before-encryption," in *2022 USENIX Annual Technical Conference (USENIX ATC 22)*, 2022, pp. 37–52.

[20] G. Ha, C. Jia, X. Ge, Y. Chen, and X. Shan, "Similarity-aware defense scheme for online brute-force attacks in encrypted deduplication," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–14, 2025.

[21] G. Ha, X. Ge, C. Jia, Y. Chen, and Z. Su, "Revisiting sgx-based encrypted deduplication via pow-before-encryption and eliminating redundant computations," *IEEE Transactions on Dependable and Secure Computing*, vol. 22, no. 3, pp. 2037–2053, 2025.

[22] G. Ha, Y. Chen, C. Jia, K. Chen, R. Wang, and Q. Jia, "Scalable encrypted deduplication based on location-hiding secret sharing of data keys," *IEEE Transactions on Computers*, vol. 74, no. 11, pp. 3710–3721, 2025.

[23] Y. Shin, D. Koo, J. Yun, and J. Hur, "Decentralized server-aided encryption for secure deduplication in cloud storage," *IEEE Trans. Serv. Comput.*, vol. 13, no. 6, pp. 1021–1033, 2020.

[24] X. Yang, R. Lu, K. K. R. Choo, F. Yin, and X. Tang, "Achieving efficient and privacy-preserving cross-domain big data deduplication in cloud," *IEEE transactions on big data*, vol. 8, no. 1, pp. 73–84, 2017.

[25] X. Yang, R. Lu, J. Shao, X. Tang, and A. A. Ghorbani, "Achieving efficient and privacy-preserving multi-domain big data deduplication in cloud," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1292–1305, 2018.

[26] H. Shin, D. Koo, and J. Hur, "Secure and efficient hybrid data deduplication in edge computing," *ACM Transactions on Internet Technology (TOIT)*, vol. 22, no. 3, pp. 1–25, 2022.

[27] Y. Fu, N. Xiao, T. Chen, and J. Wang, "Fog-to-multicloud cooperative ehealth data management with application-aware secure deduplication," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3136–3148, 2021.

[28] S. Jiang, J. Liu, Y. Zhou, and Y. Fang, "Fvc-dedup: A secure report deduplication scheme in a fog-assisted vehicular crowdsensing system," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, pp. 2727–2740, 2022.

[29] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. S. Shen, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, pp. 581–594, 5 2020.

[30] Y. Gao, L. Chen, J. Lai, T. Wang, X. Wu, and S. Yu, "Iot-dedup: Device relationship-based iot data deduplication scheme," *IEEE Transactions on Parallel and Distributed Systems*, vol. 36, no. 5, pp. 847–860, 2025.

[31] Q. Xie, C. Zhang, and X. Jia, "Security-aware and efficient data deduplication for edge-assisted cloud storage systems," *IEEE Trans. Serv. Comput.*, vol. 16, no. 3, pp. 2191–2202, 2023.

[32] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," *IEEE Secur. Priv.*, vol. 8, no. 6, pp. 40–47, 2010.

[33] G. Ha, Y. Chen, Z. Cai, C. Jia, and X. Shan, "Random coding responses for resisting side-channel attacks in client-side deduplicated cloud storage," *IEEE Trans. Serv. Comput.*, vol. 18, no. 3, pp. 1697–1710, 2025.

[34] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," 1992.

[35] C. Qin, J. Li, and P. P. C. Lee, "The design and implementation of a rekeying-aware encrypted deduplication storage system," *ACM Trans. Storage*, vol. 13, no. 1, pp. 9:1–9:30, 2017.

[36] S. Qi, W. Wei, J. Wang, S. Sun, L. Rutkowski, T. Huang, J. Kacprzyk, and Y. Qi, "Secure data deduplication with dynamic access control for mobile cloud storage," *IEEE Trans. Mob. Comput.*, vol. 23, no. 4, pp. 2566–2582, 2024. [Online]. Available: https://doi.org/10.1109/TMC.2023.3263901

[37] G. Ha, C. Jia, Y. Chen, H. Chen, and M. Li, "A secure client-side deduplication scheme based on updatable server-aided encryption," *IEEE Trans. Cloud Comput.*, vol. 11, no. 4, pp. 3672–3684, 2023.

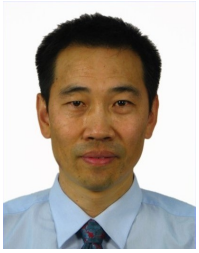
[38] Y. Zhang, C. Xu, N. Cheng, and X. Shen, "Secure password-protected encryption key for deduplicated cloud storage systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2789–2806, 2021.

[39] "Carnegie mellon university multimodal activity (cmu-mmac) database." http://kitchen.cs.cmu.edu.

[40] "Thermal solar plant dataset." https://github.com/stritti/thermal-solar-plant-dataset.
[41] "The European Climate Assessment & Dataset project," https://www.ecad.eu.
[42] "ASHRAE global database of thermal comfort field measurements," https://doi.org/10.6078/D1F671.
[43] R. Vestergaard, D. E. Lucani, and Q. Zhang, "A randomly accessible lossless compression scheme for time-series data," in *39th IEEE Conference on Computer Communications, INFOCOM 2020, Toronto, ON, Canada, July 6-9, 2020*. IEEE, 2020, pp. 2145–2154.
[44] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: Practically better than bloom," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, 2014, pp. 75–88.
[45] Z. Yan, W. Ding, X. Yu, H. Zhu, and R. H. Deng, "Deduplication on encrypted big data in cloud," *IEEE Trans. Big Data*, vol. 2, no. 2, pp. 138–150, 2016.
[46] X. Yang, R. Lu, K. R. Choo, F. Yin, and X. Tang, "Achieving efficient and privacy-preserving cross-domain big data deduplication in cloud," *IEEE Trans. Big Data*, vol. 8, no. 1, pp. 73–84, 2022.
[47] M. Li, C. Qin, and P. P. C. Lee, "Cdstore: Toward reliable, secure, and cost-efficient cloud storage via convergent dispersal," in *Proceedings of the 2015 USENIX Annual Technical Conference, USENIX ATC 2015, July 8-10, Santa Clara, CA, USA*, S. Lu and E. Riedel, Eds. USENIX Association, 2015, pp. 111–124.
[48] "Openssl," https://www.openssl.org/.
[49] "Gmp," http://gmplib.org/.
[50] "Pbc," http://crypto.stanford.edu/pbc/, 2016.

Rongxi Wang is currently working toward the Ph.D. degree from the College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China. His current research interests focus on security and privacy issues in cloud computing, storage security, and applied cryptography.

Guanxiong Ha received the PhD degree from the computer science and technology from Nankai University, Tianjin, China, in 2025. He is currently a lecturer with the College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China. His research interests include encrypted deduplication, cloud data security, and applied cryptography. He has authored or coauthored several papers at premium international journals and conferences, including the IEEE TC, IEEE TDSC, IEEE TIFS, IEEE TSC, IEEE TCC, Information Sciences, and AsiaCCS. He has been invited as a reviewer for international journals and conferences such as IEEE TDSC, IEEE TSC, and IEEE ICC.



Chunfu Jia is a Ph.D. supervisor, a professor, and the Head of Department in the Department of Cyber Sciences, Nankai University. His main research interests include network and system security, applied cryptography, and malware analysis. His work has appeared in prestigious venues such as CCS, NDSS, S&P, USENIX Security, ESORICS, IEEE TDSC, and IEEE TIFS.

Zhaoyang Liang is currently pursuing a bachelor's degree in cryptoscience and technology at the College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China. His research interests include applied cryptographic protocols, password-based cryptography, and post-quantum cryptography. He focuses on the design of secure protocols and privacy-preserving techniques.

Zhen Su is currently pursuing a master's degree in College of Cryptology and Cyber Science, Nankai University, Tianjin, P. R. China. His research interests include cloud applied cryptography and Network Security.