This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2721444, IEEE Transactions on Big Data

Achieving Efficient and Privacy-Preserving Cross-Domain Big Data Deduplication in Cloud

Xue Yang, Rongxing Lu, *Senior Member, IEEE,* Kim Kwang Raymond Choo, *Senior Member, IEEE,* Fan Yin, and Xiaohu Tang, *Member, IEEE*

Abstract—Secure data deduplication can significantly reduce the communication and storage overheads in cloud storage services, and has potential applications in our big data-driven society. Existing data deduplication schemes are generally designed to either resist brute-force attacks or ensure the efficiency and data availability, but not both conditions. We are also not aware of any existing scheme that achieves accountability, in the sense of reducing duplicate information disclosure (e.g., to determine whether plaintexts of two encrypted messages are identical). In this paper, we investigate a three-tier cross-domain architecture, and propose an efficient and privacy-preserving big data deduplication in cloud storage (hereafter referred to as EPCDD). EPCDD achieves both privacy-preserving and data availability, and resists brute-force attacks. In addition, we take accountability into consideration to offer better privacy assurances than existing schemes. We then demonstrate that EPCDD outperforms existing competing schemes, in terms of computation, communication and storage overheads. In addition, the time complexity of duplicate search in EPCDD is logarithmic.

Index Terms—Secure data deduplication, big data, brute-force attacks, data availability, accountability.

1 Introduction

Cloud storage usage is likely to increase in our big data-driven society. For example, IDC predicts that the amount of digital data will reach 44 ZB in 2020 [1]. Other studies have also suggested that about 75% of digital data are identical (or duplicate) [2], and data redundancy in backup and archival storage system is significantly more than 90% [3]. While cost of storage is relatively cheap and advances in cloud storage solutions allow us to store increasing amount of data, there are associated costs for the management, maintenance, processing and handling of such big data [4], [5]. It is, therefore, unsurprising that efforts have been made to reduce overheads due to data duplication. The technique of *data deduplication* is designed to identify and eliminate duplicate data, by storing only a single copy of redundant data. In other words, data deduplication technique can significantly reduce storage and bandwidth requirements [6]. However, since users and data owners may not fully trust cloud storage providers, data (particularly sensitive data) are likely to be encrypted prior to outsourcing. This complicates data deduplication efforts, as identical data encrypted by different users (or even the same user using different keys) will result in different ciphertexts [7], [8]. Thus, how to efficiently perform data deduplication on encrypted data is a topic of ongoing research interest.

In recent times, a number of data deduplication schemes have been proposed in the literature. These schemes are designed to realize encrypted data deduplication (see [9], [10], [11]). For example, the μR-MLE2 (Dynamic) scheme proposed in [12] seeks to improve the efficiency of duplicate ciphertext identification. However, the scheme suffers from brute-force attacks, the most popular attack in secure data deduplication schemes (see [9], [10]). Zhou et al. [13] proposed another efficient secure deduplication scheme SecDep to resist brute-force attacks. However, this scheme only deals with small-sized data, and is not suitable for big data deduplication. To solve this problem, Yan et al. [14] proposed a scheme to deduplicate encrypted big data stored in the cloud based on ownership challenge and proxy re-encryption. Although this scheme is efficient, it is vulnerable to brute-force attacks.

While there are schemes in the literature that are resilient to brute-force attacks, we are not aware of any scheme that is resilient to brute-force attacks and provides both efficiency and data availability at the same time. Data encryption alone is insufficient to ensure privacy in existing data deduplication schemes. For example, *duplicate information* (e.g., to determine whether plaintexts of two encrypted messages are identical) of the outsourced data left unprotected may have serious privacy implications. A number of incidents have shown that such information may be more invasive to one's privacy than the core data itself (e.g. NSA PRISM [15]). However, such information disclosure is inevitable in existing deduplication schemes. Therefore, we aim to minimize information leakage as much as practical, in the sense that only the entity (the cloud storage provider) that operates the deduplication knows it. Furthermore, if the duplicate information is leaked, then the cloud storage provider will be held accountable [16], [17].

It is clear that designing an efficient deduplication scheme that achieves privacy-preserving, availability and accountability, while resisting brute-force attacks remains challenging. Therefore, in this paper, using a three-tier cross-domain architecture, we propose an efficient and privacy-preserving big data deduplication in cloud storage, hereafter referred to as EPCDD. The EPCDD scheme achieves privacy-preserving, data availability and accountability, as well as resisting brute-force attacks. We then construct a

- *X. Yang, F. Yin and X. Tang are with the Information Security and National Computing Grid Laboratory, Southwest Jiaotong University, Chengdu, 610031, China.*
  E-mail: xueyang.swjtu@gmail.com (X. Yang), yinfan519@gmail.com (F. Yin), xhutang@swjtu.edu.cn (X. Tang).
- *R. Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, E3B 5A3, Canada.*
  E-mail:rlu1@unb.ca
- *K.-K. R. Choo is with the Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249-0631, USA.*
  E-mail:raymond.choo@fulbrightmail.org

deduplication decision tree based on the *binary search tree* to improve the time complexity of duplicate search. This *deduplication decision tree* is a dynamic tree that supports data update such as data insertion, deletion and modification.

Next, we will introduce the system model, threat model and design goals in Section 2, before describing notations and bilinear groups of composite order in Section 3. We then present the proposed EPCDD scheme in Section 4, analyze the scheme's privacy-preserving capability and security strength (data availability, accountability, and brute-force attack resilience) of the EPCDD scheme in Section 5, and demonstrate the efficiency of our EPCDD scheme in comparison to state-of-the-art schemes of [12], [14] in terms of computation, communication and storage overheads in Section 6. Related work is discussed in Section 7. We conclude this paper in Section 8.

2 Models and Design Goals

In this section, we formalize the system model and threat model used in this paper, and identify our design goals.

2.1 System Model

The system model (see Fig.1) is a three-tier cross-domain big data deduplication system, which comprises a key distribution center (KDC), a cloud service provider (CSP), clients from different domains and the corresponding local managers, denoted as LMA and LMB.

- **KDC:** The trusted KDC is tasked with the distribution and management of private keys for the system.
- **CSP:** The first tier is a CSP, which offers data storage services for clients. While the CSP is capable of supporting the storage needs of clients, it is financially vested to reduce the expensive big data management and maintenance overheads. Therefore, the CSP needs to perform inter-deduplication, which means that messages for deduplication are from different domains, to decrease the corresponding overhead.
- **LMA (LMB):** The second tier consists of domains (e.g., organizations such as companies or universities), which have cloud storage contracts with the CSP. Each domain maintains a local manager (e.g., LMA or LMB), which is responsible for intra-deduplication, and forwarding messages from clients in domain A (or B).
- **Clients:** Every client is affiliated with a domain (e.g., employees in the company or students and faculty members in the university or university network, say University of Texas system). Clients upload and save their data with the CSP. In order to protect their data privacy and help the CSP to complete data deduplication over encrypted data, they encrypt the data and generate the corresponding tags. Finally, clients send message tuples containing encrypted data and the corresponding tags to the LMA or LMB (clients from domains A and B send message tuples to the LMA and LMB, respectively).

For simplicity, Fig. 1 illustrates the system model for two different domains, but our scheme can be easily extended to support multiple domains.

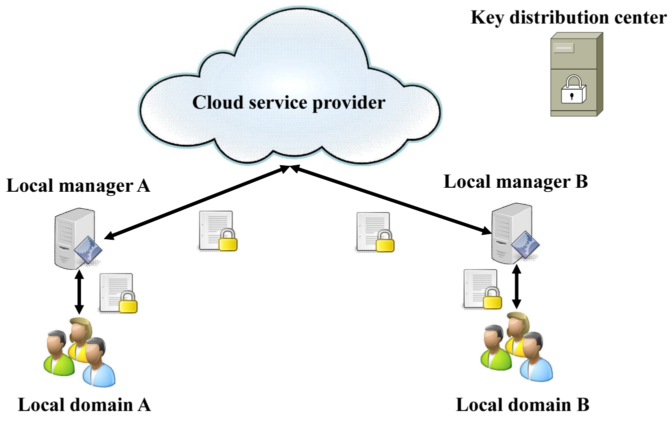


Fig. 1: System model under consideration

2.2 Threat Model

In our threat model, the CSP is considered honest but curious, which is the most common assumption in the literature (see [12], [14], [18]). Specifically, the CSP honestly follows the underlying scheme. However, it is curious about contents of stored data. Because the CSP adopts a pay-as-you-use model, it does not actively modify stored messages due to reputation, financial and legal implications (e.g. a civil litigation can result in significant reputation and financial losses to the provider). Hence, active attacks from the CSP are not considered in this paper. However, due to the significant amount of data stored in the cloud, it may know the plaintext space. Hence, according to the ciphertext and corresponding tags, the CSP (e.g. a malicious CSP employee) can carry out brute-force attacks. Finally, the CSP may obtain the plaintext corresponding to the special ciphertext for other illicit purposes (e.g. information reselling for financial gains).

LMA and LMB are also considered honest but curious. However, these entities have very limited computing and storage capabilities. Therefore, in practice, they do not have sufficient resources to carry out brute-force attacks. LMA or LMB may be curious about its affiliated clients' privacy, even though they may not actively seek to compromise the privacy of their clients. For example, if the domain is a company and LMA (or LMB) is the corresponding information manager. LMA (or LMB) is curious about the data uploaded by the staff. However, to protect the information asset, LMA or LMB does not actively attempt to compromise the privacy of clients, or collude with the CSP.

Clients are considered honest. In theory, it is possible that they would collude with the CSP to obtain other clients' privacy. As mentioned in [14], in practice, such collusion may result in significant risks to the reputation of the CSP, as well as civil litigation or criminal investigations. In addition, if the CSP colludes with client A to compromise the privacy of client B, the CSP is also likely to collude with client B or other clients to compromise the privacy of other existing clients. This would have serious repercussions for the CSP if such collusion is reported or known. Thus, we assume that the CSP does not collude with its clients. Other than brute-force attacks, we do not consider other active attacks.

2.3 Design Goals

The goals of our proposed EPCDD scheme are described as follows:

- *Privacy*. Although the CSP and local managers can obtain the encrypted data along with the corresponding tags, the CSP and LMA (LMB) are not able to obtain plaintexts from these tuples. In addition, the duplicate information disclosure is inevitable in data deduplication, but we seek to minimize such information leakage as much as practical.
- *Resist brute-force attacks*. Suppose the CSP has some background knowledge of the plaintext space. Although the CSP stores encrypted message tuples of all clients and has knowledge of some secret parameters (see section 4.1) received from the KDC, it is not able to obtain the plaintext corresponding to the specific ciphertext through brute-force attacks.
- *Availability*. In order to reduce the big data storage and management overheads, the CSP may attempt to delete the duplicate data, without affecting data availability.
- *Accountability*. Accountability provides citizens a means of holding actors such as politicians and government officials responsible for their actions, particularly those that have an impact on the society, etc [19]. Similarly, in this paper, clients delegate data storage to the CSP by paying associated costs, and disclose the duplicate information to the CSP to help completing the encrypted data deduplication. Hence, the CSP must ensure that clients can be assured that the CSP is accountable in the sense that the CSP will hold true to the contractual obligations (e.g. availability and privacy).
- *Efficiency*. The storage, computation and communication overheads associated with the big data deduplication should be as small as possible, and the cost of searching for duplicated data should also be minimized.

3 Preliminary

In this section, we define some notations and outline the definition of the bilinear groups of composite order, which serves as the building block of the proposed EPCDD scheme.

3.1 Notations

- $a \mid b$: b can be divisible by a.
- $\{0,1\}^n$: set of binary string of length n.
- $\{0,1\}^*$: set of all finite binary strings.
- sk_i : symmetric key for encrypting the data m_i.
- $Enc_{key}(\cdot)$: symmetric encryption algorithm with symmetric key key.
- $|m|$: bit length of m.
- $(C_i, \tau_i^1, \tau_i^2)$: message tuple for the data m_i, where C_i is the ciphertext encrypted using the symmetric encryption algorithm, i.e., $C_i = Enc_{sk_i}(m_i)$, τ_i^1 and τ_i^2 are two tags.
- $current\ node \rightarrow \tau^1(\tau^2)$: first (second) tag stored in the current node, i.e., if the current node stores the message tuple $(C_i, \tau_i^1, \tau_i^2)$, then $current\ node \rightarrow \tau^1 = \tau_i^1$ and $current\ node \rightarrow \tau^2 = \tau_i^2$.

3.2 Bilinear Groups of Composite Order

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic multiplicative groups of composite order N, where $N = pq$, and p, q are two distinct large primes. Let g be a generator of $\mathbb{G}$. A bilinear map groups of composite order is a mapping $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties [20], [21]:

- *Bilinear*: $e(g^a, h^b) = e(g,h)^{ab}$ for all $g, h \in \mathbb{G}$, and $a, b \in \mathbb{Z}_N$.
- *Non-degeneracy*: there exists $g \in \mathbb{G}$ such that $e(g,g)$ has order N in $\mathbb{G}_T$. In other words, $e(g,g)$ is a generator of $\mathbb{G}_T$, where g is a generator of $\mathbb{G}$.
- *Computability*: there exists an efficient algorithm to compute $e(g,h) \in \mathbb{G}_T$ for all $g, h \in \mathbb{G}$.

Definition 1. *(Composite Bilinear Generator) A composite bilinear parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter κ as input, and outputs a 5-tuple $(N, g, \mathbb{G}, \mathbb{G}_T, e)$ where $N = pq$, and p, q are two κ-bit prime numbers. $\mathbb{G}$ and $\mathbb{G}_T$ are two groups with order N, $g \in \mathbb{G}$ is a generator, and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a non-degenerated and efficiently computable composite bilinear map.*

4 Proposed EPCDD Scheme

In this section, we propose an efficient and privacy-preserving cross-domain deduplication scheme for big data storage (EPCDD).

4.1 Key Generation

KDC takes a security parameter κ as input, and outputs a 5-tuple $(N, g, \mathbb{G}, \mathbb{G}_T, e)$ by running the composite bilinear parameter generator algorithm $\mathcal{G}en(\kappa)$. Then, it selects four random numbers $s, t, a, b \in \mathbb{Z}_N$, where $p \mid (as + bt)$, $p \nmid as$ and $p \nmid bt$, and computes $y_A = g^{aq} \in \mathbb{G}$, $y_B = g^{bq} \in \mathbb{G}$. In addition, KDC chooses three cryptographic hash functions $h_1 : \{0,1\}^* \rightarrow \{0,1\}^n$, $h_2 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ and $h_3 : \mathbb{G} \rightarrow \{0,1\}^n$, where n is the bit length of symmetric key. Finally, KDC sends s and t to all members in domains A and B, respectively, and sends y_A and y_B to the CSP by secure channel. KDC publishes parameters $pp = (N, g, \mathbb{G}, \mathbb{G}_T, e, e(g,g)^s, e(g,g)^t, h_1, h_2, h_3)$.

4.2 Data Encryption and Tags Generation

For each client in domain A, after receiving the secret key s, the client encrypts the data m_i and generates corresponding tags for data deduplication as follows.

4.2.1 Data Encryption

With secret key s and parameter $e(g,g)^t$, the client computes the message-dependent symmetric key $sk_i = h_1(m_i \| e(g,g)^{st})$. Then, this client chooses a random number $r_i \in \mathbb{Z}_N$, computes the ciphertext $C_i = Enc_{sk_i}(r_i \| m_i)$, where the symmetric encryption algorithm is the cipher block chaining (CBC) mode, i.e., *AES-CBC*.

4.2.2 Tags Generation

The client generates two tags for data m_i as $\tau_i^1 = g^{s \cdot h_2(m_i)}$, $\tau_i^2 = sk_i \bmod \omega$, where ω is a random integer that not only ensures the security of sk_i but also meets the storage capacity of the CSP. For example, we can set $|\omega| = 128$ bits, thus, sk_i does not be disclosed by guessing attack if we set $n = 256$ bits ($|sk_i|$=256), and it can represent up to 2^{128} data.

Similarly, clients in domain B execute same operations to generate ciphertexts and tags, e.g., for m_j, encrypt it as $C_j = Enc_{sk_j}(r_j \| m_j)$, where $sk_j = h_1(m_j \| e(g,g)^{st})$, and the corresponding tags are computed as $\tau_j^1 = g^{t \cdot h_2(m_j)} \in \mathbb{G}$, $\tau_j^2 = sk_j \bmod \omega$.

4.3 Deduplication Decision Tree (DDT) Initialization

In order to improve the efficiency of finding duplicated data, we construct the *deduplication decision trees* (*DDT*s) based on the popular *binary search tree* (*BST*) [22] for searching duplicate data. As far as we know, the *DDT* initialization is similar to insertion of the *BST*, but this operation begins with the empty tree. Suppose the CSP has received k different message tuples $\{(C_1, \tau_1^1, \tau_1^2), (C_2, \tau_2^1, \tau_2^2), \ldots, (C_k, \tau_k^1, \tau_k^2)\}$ form domain A at the current moment. Hence, CSP constructs a *DDT-A* for this domain to store k message tuples for subsequence deduplication.

Based on the insertion operation of the *BST*, we propose Algorithm 1 to construct *DDT*s. According to Algorithm 1, CSP stores k message tuples in turn at appropriate nodes, as shown in Fig.2. In addition, in order to ensure the time complexity of searching duplicate is $O(\log k)$, we need to balance the tree in the process of the *DDT* construction. The details are introduced in [23].

Algorithm 1 The Construction for Deduplication Decision Tree (*DDT*)

```
Suppose the CSP needs to store message tuple $(C_i, \tau_i^1, \tau_i^2)$ at
some point. (Initially, the current node is the root of the DDT)
while $current\ node \neq null$ do
    if $\tau_i^2 < current\ node \rightarrow \tau^2$ then
        move tuple $(C_i, \tau_i^1, \tau_i^2)$ to the left subtree of current node.
    else
        move the tuple to the right subtree of current node.
    end if
end while
Store the message tuple $(C_i, \tau_i^1, \tau_i^2)$ at the current node, where
$current\ node \rightarrow \tau^1 = \tau_i^1$, $current\ node \rightarrow \tau^2 = \tau_i^2$ and
$current\ node \rightarrow C = C_i$.
```

4.4 Uploading Data

Suppose a client in domain B wishes to upload the data m_* to the CSP. This client computes two tags $\tau_*^1 = g^{t \cdot h_2(m_*)}$ and $\tau_*^2 = h_1(m_* \| e(g,g)^{st}) \bmod \omega$, and sends them to the LMB. Upon receiving tags, LMB computes the hash value for τ_*^1, i.e., $T_* = h_3(\tau_*^1)$, and then searches the hash table that records hash values of the first tag for all different data from domain B. If the same hash value has already been recorded, LMB returns this client "*duplication find*", and does not need to forward any message to the CSP. Otherwise, LMB sends the tags (τ_*^1, τ_*^2) to the CSP. After receiving it, CSP checks the duplication on the *DDT-A*. If the duplicated data exist, CSP sends "*duplication find*" to the LMB. Otherwise, it sends "*upload data*" to the LMB. After receiving the feedback, LMB forwards it to the client. Once receiving the message "*upload data*", the client encrypts m_* as $C_* = Enc_{sk_*}(r_* \| m_*)$, and then sends it to the CSP via the LMB. After receiving the ciphertext C_*, CSP leverages Algorithm 1 to insert the message tuple $(C_*, \tau_*^1, \tau_*^2)$ into the appropriate node in the *DDT-B*. See Algorithm 2 for details. It is worth noting that the process of uploading data from clients in domain A is identical to B. Hence, we ignore this process. Fig. 2 shows an example of searching the duplicate data.

Algorithm 2 Data Deduplication over *DDT-A*

```
1: Client → LMB: a client sends two tags $(\tau_*^1, \tau_*^2)$ to the LMB.
2: LMB: LMB computes $T_* = h_3(\tau_*^1)$, and then checks the
   hash table. If the same hash value has already been recorded,
   sends the "duplication find" back to this client. Otherwise,
   LMB stores $T_*$ at the hash table, and transmits tags $(\tau_*^1, \tau_*^2)$
   to the CSP.
3: CSP: After receiving tags, it starts checking for the duplicate
   data from the root node. (Initially, the current node is the root
   of DDT-A)
   3.1 CSP determines whether $current\ node \rightarrow \tau^2 = \tau_*^2$,
       if not, it will proceed to step 3.2. Otherwise, it verifies
       whether
           $e(current\ node \rightarrow \tau^1, g^{aq}) \cdot e(\tau_*^1, g^{bq}) = 1.$   (1)
       If the equation (1) holds, then the duplicate data has been
       found. CSP deletes tags $(\tau_*^1, \tau_*^2)$. Otherwise, it means that
       there is no duplicate data. After that, CSP stops the search
       and goes to step 4.
   3.2 CSP compares $current\ node \rightarrow \tau^2$ and $\tau_*^2$. If $\tau_*^2 <$
       $current\ node \rightarrow \tau^2$, then CSP moves the pointer to the
       left subtree of the current node. Otherwise, it moves the
       pointer to the right subtree. Then, it returns to step 3.1.
       This process is repeated until the duplicated data is found
       or the remaining subtree is null.
4: CSP → client: CSP returns 1 when the duplication is found,
   and 0 otherwise.
5: Client: when the client receives 0 from the CSP, it encrypts
   the data $m_*$ as $C_* = Enc_{sk_*}(r_* \| m_*)$, and sends it to the
   CSP via the LMB. Otherwise, it does not need to upload $m_*$.
6: CSP: After receiving the ciphertext $C_*$, CSP leverages Al-
   gorithm 1 to inset the message tuple $(C_*, \tau_*^1, \tau_*^2)$ into the
   appropriate node in DDT-B.
```

Corollary 1. *(Correctness of the inter-deduplication) Suppose two message tuples $(C_i, \tau_i^1, \tau_i^2)$ and $(C_j, \tau_j^1, \tau_j^2)$ are from domains A and B, respectively. The equation*

$$e(\tau_i^1, g^{aq}) \cdot e(\tau_j^1, g^{bq}) = 1 \tag{2}$$

holds, if and only if $m_i = m_j$.

Proof. As noted in Section 4.1, since $p \mid (as + bt)$, there is an integer k such that $as + bt = kp$. In addition, $e(g,g)$ is a generator of $\mathbb{G}_T$ with order N, so $e(g,g)^N = 1$. We first prove the sufficient condition. Suppose $m_i = m_j$, then $h_2(m_i) = h_2(m_j)$, we obtain:

$$\begin{aligned} & e(\tau_i^1, g^{aq}) \cdot e(\tau_j^1, g^{bq}) \\ =& e(g^{s \cdot h_2(m_i)}, g^{aq}) \cdot e(g^{t \cdot h_2(m_j)}, g^{bq}) \\ =& e(g,g)^{asqh_2(m_i) + btqh_2(m_j)} \\ =& e(g,g)^{(as+bt)q \cdot h_2(m_i)} \end{aligned} \tag{3}$$

$$\xrightarrow{\because\ as+bt=kp,\ e(g,g)^N=1}$$

$$= e(g,g)^{kN \cdot h_2(m_i)} = 1 \tag{4}$$

Hence, when $m_i = m_j$, Eq. (2) always holds.

For the necessary condition, we use *reductio ad absurdum*. If Eq. (2) holds, but $m_i \neq m_j$, as shown in Eq. (3) and Eq. (4), the condition $N \mid (asqh_2(m_i) + btqh_2(m_j))$ must be satisfied,

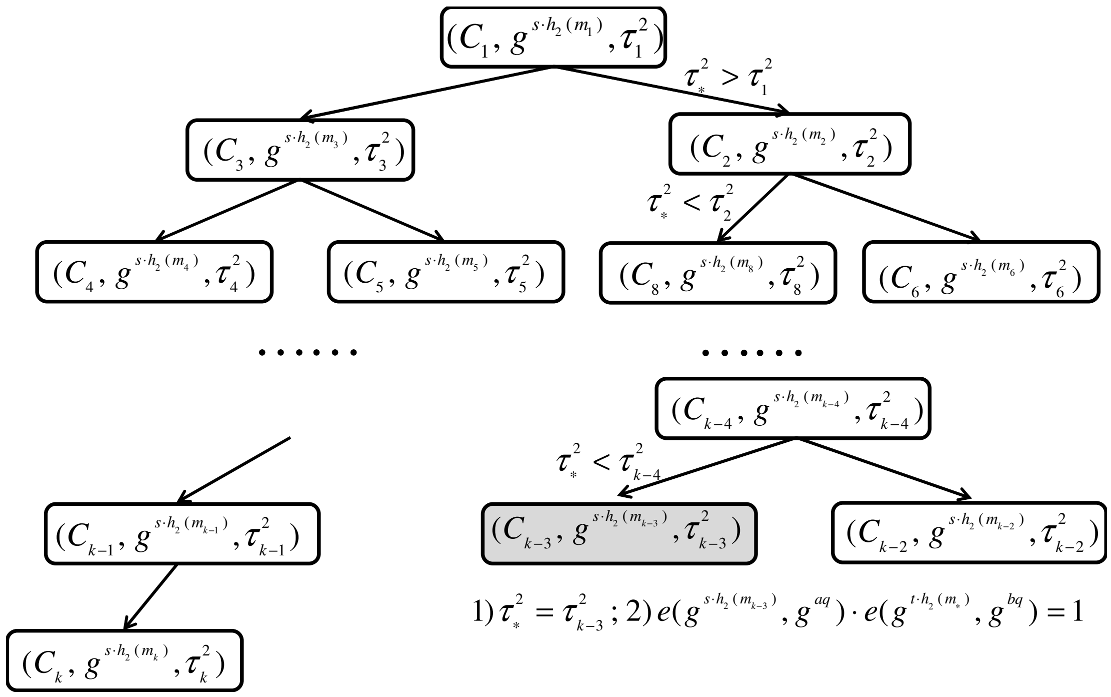

$(C_1, g^{s\cdot h_2(m_1)}, \tau_1^2)$
$\tau_*^2 > \tau_1^2$
$(C_3, g^{s\cdot h_2(m_3)}, \tau_3^2)$
$(C_2, g^{s\cdot h_2(m_2)}, \tau_2^2)$
$\tau_*^2 < \tau_2^2$
$(C_4, g^{s\cdot h_2(m_4)}, \tau_4^2)$
$(C_5, g^{s\cdot h_2(m_5)}, \tau_5^2)$
$(C_8, g^{s\cdot h_2(m_8)}, \tau_8^2)$
$(C_6, g^{s\cdot h_2(m_6)}, \tau_6^2)$
$(C_{k-4}, g^{s\cdot h_2(m_{k-4})}, \tau_{k-4}^2)$
$\tau_*^2 < \tau_{k-4}^2$
$(C_{k-1}, g^{s\cdot h_2(m_{k-1})}, \tau_{k-1}^2)$
$(C_{k-3}, g^{s\cdot h_2(m_{k-3})}, \tau_{k-3}^2)$
$(C_{k-2}, g^{s\cdot h_2(m_{k-2})}, \tau_{k-2}^2)$
$1)\ \tau_*^2 = \tau_{k-3}^2;\ 2)\ e(g^{s\cdot h_2(m_{k-3})}, g^{aq})\cdot e(g^{t\cdot h_2(m_*)}, g^{bq}) = 1$
$(C_k, g^{s\cdot h_2(m_k)}, \tau_k^2)$

Fig. 2: The deduplication decision tree for domain A (*DDT-A*) and an example of searching the duplicate data. Given the message tuple $(\tau_*^1 = g^{t\cdot h_2(m_*)}, \tau_*^2)$ from domain B, according to Algorithm 2, CSP finds that the same data has already been stored, i.e., $(C_{k-3}, \tau_{k-3}^1, \tau_{k-2}^2)$.

which is equivalent to $p \mid (ash_2(m_i) + bth_2(m_j))$. Therefore, we obtain:

$$\begin{aligned}
&p \mid (ash_2(m_i) + bth_2(m_j)) \\
\overset{(a)}{\Leftrightarrow}&p \mid (kp\cdot h_2(m_j) + as(h_2(m_i) - h_2(m_j))) \\
\Leftrightarrow&p \mid as(h_2(m_i) - h_2(m_j)) \\
\overset{(b)}{\Leftrightarrow}&p \mid h_2(m_i) - h_2(m_j) \qquad (5)
\end{aligned}$$

where (a) follows from the condition $bt = kp - as$, and (b) satisfies constrains that $p \nmid as$ and p is the prime number.

Because $h_2 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, the values of $h_2(m_i)$ and $h_2(m_j)$ satisfy $1 \leq h_2(m_i), h_2(m_j) \leq p-1$. Hence, we can obtain $\mid h_2(m_i) - h_2(m_j) \mid \leq p-2$, it means that $p \nmid (h_2(m_i) - h_2(m_j))$, which contradicts the formula (5). It is worth noting that when $h_2(m_i) - h_2(m_j) = 0$, the formula (5) holds. However, as h_2 is the cryptographic hash function, the probability of collision is negligible. In other words, if $m_i \neq m_j$, $h_2(m_i) - h_2(m_j) \neq 0$, that is, the equation (2) does not hold. Thus, we prove that Eq. (2) holds, if and only if $m_i = m_j$. □

Correctness of the Eq. (1). Suppose the message tuple $(C_i, \tau_i^1, \tau_i^2)$ is stored at the current node of *DDT-A*. Because all message tuples stored in *DDT-A* are from domain A, we obtain:

$$\begin{aligned}
&e(current\ node \rightarrow \tau^1, g^{aq}) \cdot e(\tau_*^1, g^{bq}) \\
=&e(\tau_i^1, g^{aq}) \cdot e(\tau_*^1, g^{bq}) \\
=&e(g^{s\cdot h_2(m_i)}, g^{aq}) \cdot e(g^{t\cdot h_2(m_*)}, g^{bq}) \\
=&e(g,g)^{asqh_2(m_i)+btqh_2(m_*)} \qquad (6)
\end{aligned}$$

According to Corollary 1, Eq. (6) equals to 1 if and only if $h_2(m_i) = h_2(m_*)$, i.e., $m_i = m_*$.

As described in [22], *BST* is a dynamic tree, which supports lookup, insertion and deletion operations. Hence, our *DDT-A* and *DDT-B* also support these operations. Clearly, the process of finding the duplicated data is equivalent to the lookup operation, and the process of adding a new data is equivalent to the insertion operation. In addition, if the client needs to delete a specific data stored in the CSP, then the CSP can use the deletion operation of *BST*, while ensuring the balance of *DDTs*.

5 Security Analysis

In this section, we analyze the security properties of the proposed EPCDD scheme. In particular, following design goals illustrated in Section 2.3, our analysis will focus on explaining how the proposed EPCDD scheme can achieve privacy-preserving, data availability and accountability, while resisting brute-force attacks.

5.1 Privacy Analysis

We analyze that our EPCDD scheme can protect the privacy of sensitive data from disclosure, and minimize the duplicate information disclosure. In order to cooperate with the CSP to process data deduplication, clients need to not only upload the encrypted data, but also provide two corresponding tags. Because C_i is encrypted by the symmetric encryption algorithm, i.e., *AES-CBC*, the security of C_i is based on the symmetric encryption algorithm. Moreover, if the CSP and LMA (LMB) want to obtain m_i from the tag $\tau_i^1 = g^{s\cdot h_2(m_i)}$ ($\tau_i^1 = g^{t\cdot h_2(m_i)}$), it means that they need to deal with the discrete logarithm (DL) problem and the one-way hash function, which have been proved to be computationally infeasible difficult problems. Therefore, CSP or LMA (LMB) cannot obtain m_i from the τ_i^1. Because $\tau_i^2 = sk_i \bmod \omega$, there is an integer k such that $sk_i = k\omega + \tau_i^2$. One equation has two unknown numbers, CSP or LMA (LMB) only can obtain

sk_i by guessing attack. However, as described in section 4.2, if $|sk_i| = 256$ bits, we can set $|\omega| = 128$ bits, which can sufficiently resist the guessing attack. Therefore, data confidentiality can be achieved in this paper. In addition, to verify whether the different ciphertexts correspond to the identical plaintext, it needs to verify whether the Eq. (1) holds. As designed in our EPCDD scheme, only the CSP has the secret parameters g^{aq} and g^{bq}, so that only it can perform this verification. In other words, only the CSP knows the duplicate information. Thus, Our scheme can reduce the disclosure of duplicate information as much as possible.

5.2 Brute-force Attack Resilience

As mentioned in Section 2.3, CSP has some background knowledge of plaintext space $\mathcal{M}$, and stores the message tuples of all clients, but our proposed EPCDD scheme is still able to resist brute-force attacks. Specifically, although the CSP possesses two secret parameters g^{aq} and g^{bq}, it cannot obtain aq or bq due to the hardness of the DL problem, let alone s or t ($p \mid as + bt$). Hence, given $e(g,g)^s$ and $e(g,g)^t$, it is hard to compute $e(g,g)^{st}$ without s or t, which is the CDH problem. Similarly, it is hard to get g^s or g^t.

For the specific message tuple $(C_i, \tau_i^1, \tau_i^2)$, although the CSP has the plaintext space $\mathcal{M}$ and can try all data $m_i \in \mathcal{M}$, it cannot generate the valid symmetric key $sk_i = h_1(m_i \| e(g,g)^{st})$ without $e(g,g)^{st}$. Thereby, it cannot decrypt C_i, let alone generate the same ciphertext C_i without knowing sk_i and the random number r_i. Since τ_i^2 is determined by sk_i, it cannot launch brute-force attacks through τ_i^2 without knowing sk_i. Moreover, for all $m_i \in \mathcal{M}$, CSP can compute the corresponding hash value $h_2(m_i)$. However, it still cannot compute $\tau_i^1 = g^{s \cdot h_2(m_i)}$ ($\tau_i^1 = g^{t \cdot h_2(m_i)}$) without g^s (g^t). Therefore, CSP can not find the correlation between the plaintext and the specific message tuple by brute-force attacks.

5.3 Availability

As discussed in Section 2.3, whichever the duplicated data has been deleted, as long as the client has uploaded the ciphertext corresponding to the specific data, it must ensure that this client can download and decrypt the stored ciphertext to obtain this data. More specifically, suppose client A from domain A needs to store m_i. Firstly, he sends the message tuple $(C_i, \tau_i^1, \tau_i^2)$ to the CSP. Then, CSP discovers C_i^* that has the identical data m_i has been stored previously. Hence, it does not need to store $(C_i, \tau_i^1, \tau_i^2)$. After a period of time, client A sends the request to download the encrypted data C_i, however, the CSP returns C_i^*. As shown in 4.2, $C_i^* = Enc_{sk_i}(r_i^* \| m_i)$, where $sk_i = h_1(m_i \| e(g,g)^{st})$. Obviously, client A has the symmetric key sk_i of m_i, so it can decrypt C_i^*. In other words, although the message tuple $(C_i, \tau_i^1, \tau_i^2)$ has been deleted by the CSP, client A can still obtain the data m_i.

As discussed in Section 2.2, CSP would not collude with its clients so that clients cannot generate sk_i without m_i. That is, clients cannot obtain the data that does not belong to them. Hence, our EPCDD scheme satisfies not only data availability but also the proofs-of-ownership (PoW) [14].

5.4 Accountability

Under the aforementioned design goals, accountability occurs at two cases in this paper: data unavailability and duplicate information disclosure. If the data can not be available due to the data deduplication, clients can conclude that the responsibility belongs to the CSP. Since only the CSP has secret parameters g^{aq} and g^{bq}, only it can verify whether the Eq. (1) holds. In other words, only the CSP can perform encrypted data deduplication. Moreover, if the duplicate information is leaked, we can know that the CSP should be accountable since only the CSP knows this information. Thus, although the CSP is honest but curious, we can still control the behavior of the CSP to a certain extent to ensure data availability and improve privacy-preserving (protect duplicate information from disclosure).

6 Performance Evaluation

In this section, we evaluate the performance of the proposed EPCDD scheme in terms of the computational, communication and storage overheads. Moreover, we give a comparison with the μR-MLE2 (Dynamic) scheme [12] and Yan's scheme [14].

6.1 Computational Overheads

There are four entities in our EPCDD scheme, namely: clients, KDC, CSP and LMA (LMB). Under the aforementioned system model, KDC is responsible for generation of system parameters, which does not participate in the data deduplication. Thus, the computational overhead of the KDC can be ignored. We analyze the computational overhead of uploading one data in two cases: the data is duplicate and the data is new.

For data m_i, the client first generates two tags τ_i^1 and τ_i^2 to help the CSP to complete the deduplication. This operation requires one exponentiation in $\mathbb{G}$, one multiplication in $\mathbb{Z}_N$, and one module in $\mathbb{Z}_\omega$. Since the multiplication in $\mathbb{Z}_N$ and module in $\mathbb{Z}_\omega$ are considered negligible compared to the exponentiation and pairing, the computational overhead of tag generation requires one exponentiation in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. If the duplication is found, the client does not need to do anything else. Otherwise, it needs to encrypt the data m_i by symmetric encryption, i.e., *AES-CBC*. As shown in [14], the computational overhead of data encryption using symmetric encryption depends on the size of data, which is inevitable for protecting the data. Therefore, we can ignore the computational overhead of the encryption in the comparison of these three schemes. In addition, regardless of whether duplicate data exist, the client needs to generate the message-dependent key sk_i for data availability, which costs one hash and one exponentiation in $\mathbb{G}_T$. It is worth noting that the computational overhead of hash depends on the size of data, but it is very fast, which can be ignored. Therefore, this computation only needs one exponentiation in $\mathbb{G}_T$.

After receiving tags τ_i^1 and τ_i^2, CSP compares τ_i^2 and $current\ node \rightarrow \tau^2$ according to the *DDT* shown in Fig. 2. If $\tau_i^2 = current\ node \rightarrow \tau^2$, it needs to verify the Eq. (1), which requires two pairings and one multiplication in $\mathbb{G}_T$. As shown in Algorithm 2, the CSP needs to verify the Eq. (1) if and only if $\tau_i^2 = current\ node \rightarrow \tau^2$, which is independent of the search complexity. Hence, the computational complexity for finding duplication is $O(1)$. Moreover, for the intra-deduplication, LMA and LMB just compute the hash value $h_3(\tau_i^1)$, and then check whether the same hash value exists (by comparing the value with the records in LMA or LMB). Obviously, the computational overhead for the LMA and LMB can be ignored.

In Yan's scheme [14], the client u_i first computes the tag x_i to help the CSP to check the duplication. If the duplicate data does not exist, u_i encrypts the data m_i with *AES*, and encrypts the symmetric key $CK_i = E(pk_{AP}, sk_i)$ with proxy re-encryption (PRE) proposed in [24]. Except for the overhead of symmetric encryption, it requires four exponentiation operations in $\mathbb{G}$ and one exponentiation in $\mathbb{G}_T$. If the duplicate data exists, u_i would not encrypt m_i and sk_i, but u_i, CSP and AP need to verify data ownership to ensure data availability for authorized clients. Specifically, AP selects c and sends it to u_i. After receiving c, u_i computes y, and then performs $E(pk_{AP}, y)$ to protect $H(m_i)$ from disclosure. After that, it sends $E(pk_{AP}, y)$ to the AP. AP first decrypts $E(pk_{AP}, y)$ to obtain y, and verifies whether $H(yp + cV_i) = x_i$. If it holds, AP issues the re-encryption key for authorized client u_i by computing $rk_{AP \to u_i} = RG(pk_{AP}; sk_{AP}; pk_j)$, and then sends it to the CSP. After receiving $rk_{AP \to u_i}$, CSP has to finish the re-encryption operation $E(pk_i, sk_j) = R(rk_{AP \to u_i}; E(pk_{AP}, sk_j))$, and sends $E(pk_i, sk_j)$ to u_i. Finally, u_i decrypts the re-encryption ciphertext $E(pk_i, sk_j)$ to obtain sk_j selected by the client u_j. Therefore, the total computational overheads include eight exponentiation operations in $\mathbb{G}$, five exponentiation operations in $\mathbb{G}_T$ and one pairing. Likewise, in the μR-MLE2 (Dynamic) scheme [12], regardless of whether there is the duplicated data, the client needs to compute the tag $\tau_i = (g^{r_i}, g^{r_i h(m_i)})$ and the CSP needs to check the duplication by verifying whether $e(g^{r_*}, g^{r_i \cdot h(m_i)}) = e(g^{r_i}, g^{r_* \cdot h(m_*)})$. The computational overheads include two exponentiation operations in $\mathbb{G}$ and $2 \cdot O(h)$ pairing operations, where $O(h)$ denotes the time complexity of the search and h is the depth of the *DDT*. Furthermore, if the duplicated data does not exist, it still takes two exponentiation operations in $\mathbb{G}$ to encrypt the symmetric key.

We present the comparison of computational overheads in our EPCDD scheme, the μR-MLE2 (Dynamic) scheme [12] and Yan's scheme [14] in Table. 1. From the table, it is shown that the complexity of duplication search for both our EPCDD and μR-MLE2 (Dynamic) scheme [12] is $O(h)$ (logarithmic level). Actually, for k different data stored in the CSP, our EPCDD can ensure that the depth h of the *DDT* is always $\log k$ based on the balanced *BST*. However, μR-MLE2 (Dynamic) scheme [12] has been proved that $h \geq \log k$. Thus, the complexity of duplication search in our EPCDD scheme is better than μR-MLE2 (Dynamic) scheme [12]. Note that T_e, T_{et}, T_m and T_p represent the computational costs of an exponentiation in $\mathbb{G}$, an exponentiation in $\mathbb{G}_T$, a multiplication in $\mathbb{G}_T$ and a pairing, respectively. Based on the Table. 1, when k data need to be uploaded by different clients, wherein the duplication ratio is δ, the total computational overheads of EPCDD scheme, μR-MLE2 (Dynamic) scheme [12] and Yan's scheme [14] are $k \cdot (3T_e + 3T_{et} + 2T_p + T_m)$, $k \cdot (4T_e + 2T_p \cdot O(h) - 2T_e \cdot \delta)$ and $k \cdot ((4T_e + 4T_{et} + T_p) \cdot \delta + 4T_e + T_{et})$, respectively. Furthermore, we conduct the experiments with PBC [25] and OpenSSL [26] libraries running on a 2.6 GHz-processor 2 GB-memory computing machine to study the operation costs. To this end, we choose the security parameter $\kappa = 512$ bits, so that the bit-length of N is 1024 bits. The experimental results indicate that $T_p = 30$ ms, $T_m = 0.003$ ms, $T_{et} = 2.6$ ms and $T_e = 24$ ms.

With the exact operation costs, we depict the variation of computational overheads in terms of the number of uploads k and duplication ratio δ ($0 \leq \delta \leq 1$) in Fig. 3. Note that the depth of the *DDT* is set as $h = 60$, which can store up to 2^{60} data and be enough to meet the up-to-date data storage demand (44 zettabytes in 2020 [1]). From the figure, we can see that the computational overheads of three schemes increase as k increases. Actually, since our EPCDD scheme uses the shared message-dependent symmetric key, it inherently satisfies data availability and data ownership (i.e., the PoW). Regardless of whether the duplicated data exist, it only needs to generate tags and search the duplication, which represent the basic deduplication operations. Hence, the computational overheads of our EPCDD are independent of δ as shown in Fig. 3(a). However, for Yan's scheme [14], if duplicated data exist, it needs more operations to verify the ownership to ensure the data availability besides the basic deduplication operations. Thus, as shown in Fig. 3(c), the computational overheads increase as duplicated data (i.e., δ) increase. Because Yan's scheme [14] supports big data storage and deduplication, our EPCDD can better support big data deduplication compared with it. Actually, the computational overheads of μR-MLE2 (Dynamic) scheme [12] decrease as δ increases, i.e., $k \cdot (4T_e + 2T_p \cdot O(h) - 2T_e \cdot \delta)$, but the maximum value of δ is 1, which is too small compared to the explosively growth of big data ($O(h)$). Hence, the effect of δ is negligible, as shown in Fig. 3(b), but the computational overheads would rapidly increase as $O(h)$ increases, which makes this scheme can not support big data deduplication. In summary, it is shown that our EPCDD scheme significantly reduces the computational overheads compared with the μR-MLE2 (Dynamic) scheme [12] and Yan's scheme [14].

6.2 Communication Overheads

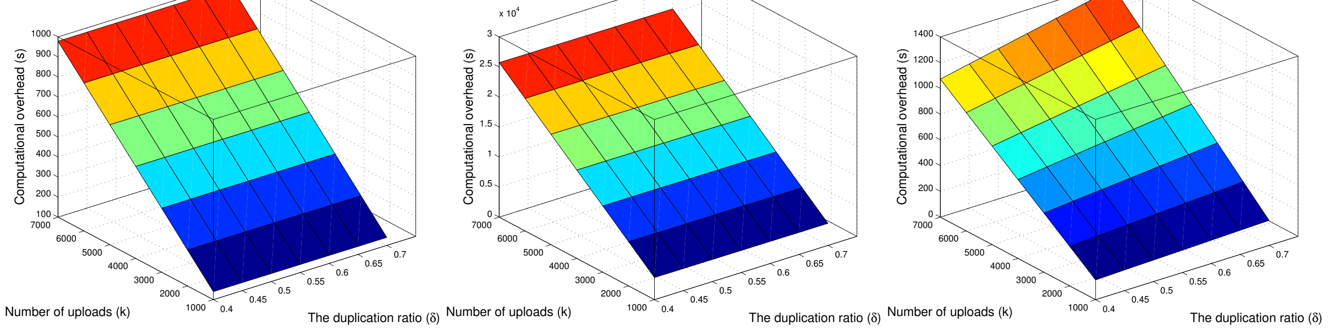
As described above, we omit communication overheads of encrypted data C_i and discuss the overheads in two cases: without the duplication and with the duplication. In our EPCDD scheme, regardless of whether duplicated data exist, the client needs to send $\tau_i^1 \| \tau_i^2$ to the CSP via the LMA or LMB, which costs $|\tau_i^1| + |\tau_i^2|$ bits. Because the length of symmetric key for *AES-CBC* is 256 bits ($n = 256$ bits), we set $|\omega| = 128$ bits, which is sufficient for the security of τ_i^2. Thus, the size of the message tuple is $|\tau_i^1| + |\tau_i^2| = 1152$ bits. Consider k data from different clients need to be uploaded, wherein the duplication ratio is δ, the whole communication overheads are $1152k$ bits. In addition, KDC needs to send secret parameters to k clients and the CSP, the messages are in the form of $s \| e(g, g)^t$ ($t \| e(g, g)^s$) and $y_A \| y_B$, respectively. Thus, its size should be $2048k$ and 2048 bits, respectively. In summary, the total communication overheads of our EPCDD are $(3200 \cdot k + 2048)$ bits.

In Yan's scheme [14], if duplicated data exist, the client u_i sends token $x_i \| pk_i$ to the CSP, which costs 1152 bits if hash value is 128 bits. Then, CSP forwards this token to the AP, after that AP sends c with 512 bits in length to u_i for verifying data ownership. After receiving c, u_i returns $E(pk_{AP}, y) \| V_i$ with 3072 bits in length. Finally, AP sends the re-encryption key $rk_{AP \to u_i}$ with 1024 bits in length to the CSP for authorization of the legitimate client u_i. Hence, the communication overheads with duplication are 6912 bits. In the case of without duplication, each client u_i just needs to upload $x_i \| pk_i \| CK_i$ to the CSP, so the communication overheads are 3200 bits. For k data ($k\delta$ duplicated data), the total communication overheads are $k(3712 \cdot \delta + 3200)$ bits. Similarly, the total communication overheads for the μR-MLE2 (Dynamic) scheme [12] are $k(2 \cdot O(h) + 6144 - 4096 \cdot \delta)$ bits.

We further plot the comparison of communication overhead in terms of the number of uploads k and the duplication ratio δ in Fig. 4. It is shown that our EPCDD scheme reduces the communication overhead compared with the other two schemes. Specifically, in Fig. 4(a), our EPCDD scheme does not vary with

TABLE 1: A Comparative Summary: Computational Overheads

Scheme	Without duplicated data	With duplicated data	Complexity of duplication search
EPCDD	$3T_e + 3T_{et} + 2T_p + T_m$		$O(h)$
μR-MLE2 (Dynamic) [12]	$4T_e + 2T_p \cdot O(h)$	$2T_e + 2T_p \cdot O(h)$	$O(h)$
Yan's scheme [14]	$4T_e + T_{et}$	$8T_e + 5T_{et} + T_p$	$O(1)$



(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

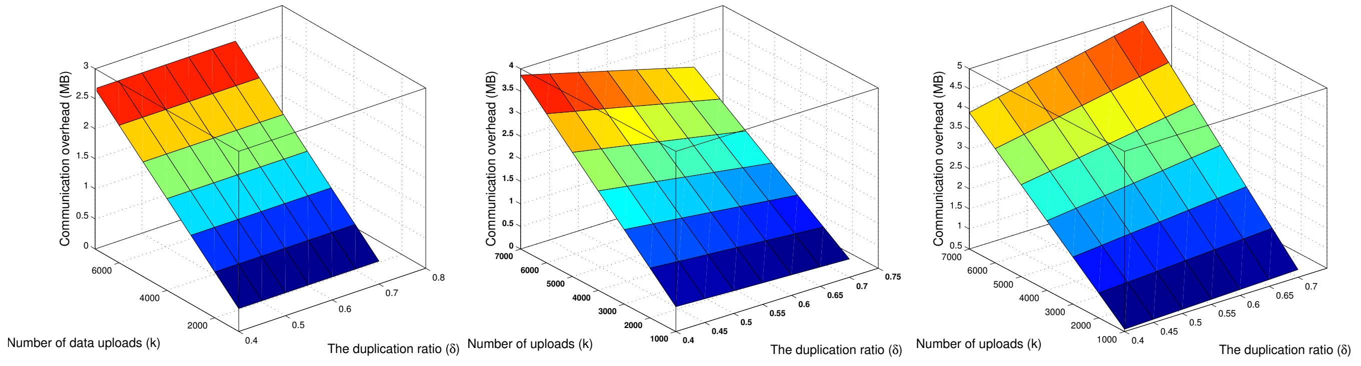
Fig. 3: A Comparative Summary: Computational Overheads

δ since it does not need any other communication except for the basic communication for deduplication. However, in Fig. 4(c), because the client needs to communicate with AP to verify the ownership besides the basic communication, the communication overheads of Yan's scheme [14] increase as δ increases. For μR-MLE2 (Dynamic) scheme [12], although the communication overheads decrease as δ increases, the communication efficiency is not better than our EPCDD because of the growth of $O(h)$. Moreover, the μR-MLE2 (Dynamic) scheme [12] uses the client-assistant way to complete duplication search, which means that the number of communication rounds between the CSP and clients is $O(h)$. Hence, this scheme brings about the communication delay, which may sharply reduce the whole efficiency of the system in the case of network congestion. Obviously, the duplication search is performed by the CSP itself, so the number of communication rounds in our EPCDD and Yan's scheme [14] are both $O(1)$. Hence, our EPCDD scheme is more efficient than other two schemes in terms of communication overhead.

6.3 Storage Costs

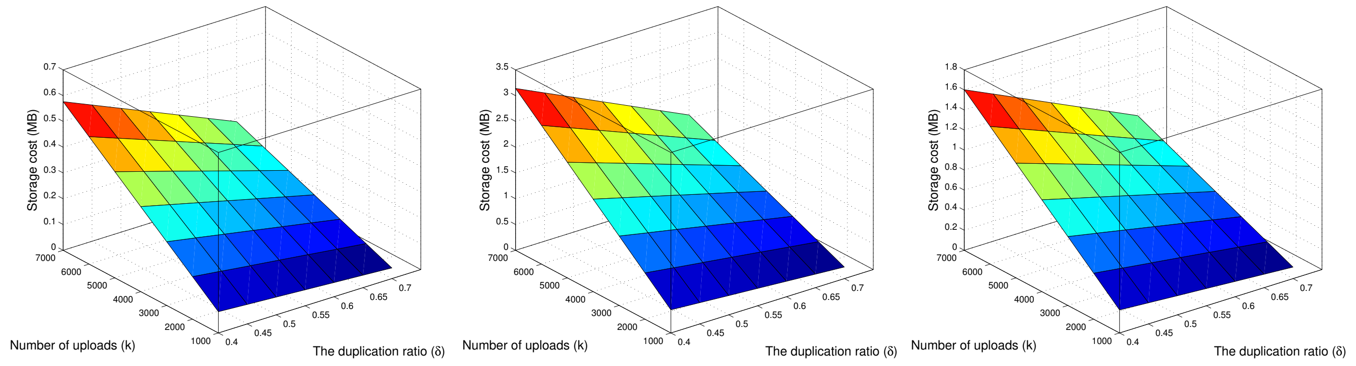
As mentioned above, CSP does not store any message of duplicated data. Thus, for k data with $k\delta$ duplicate data, CSP just needs to store $k(1-\delta)$ ciphertexts and the corresponding tags. Similarly, we omit the storage costs of ciphertexts as the same part in the three schemes. Hence, the storage costs of our EPCDD, μR-MLE2 (Dynamic) and Yan's schemes are $1152k(1-\delta)$ bits, $6272k(1-\delta)$ bits and $3200k(1-\delta)$ bits, respectively. Fig. 5 shows the comparison of storage costs for these three schemes in terms of k and δ. From the figures, we can see that the storage costs for these three schemes increase as k increases and decrease as δ increases. Further, it can be obviously shown that our EPCDD scheme reduces storage costs compared with the others.

In order to prove the advantages of deduplication in our EPCDD scheme, we compare the computational, communication and storage overheads with and without data deduplication technique. Similarly, suppose k data needs to be uploaded from different clients and the duplicate ratio is δ, if the deduplication technique is not used, it is obvious that clients only need to encrypt the data with symmetric encryption and upload the ciphertext to the CSP for data storage. Hence, computational overheads are $k \cdot T_{Enc}$, communication and storage costs are both $k \cdot |m|$. When the deduplication technique is used, for $k \cdot \delta$ duplicate data, our EPCDD scheme only needs to compute and upload two tags for duplicate searching, and does not need to store any information. As analyzed in Section 6.1 and 6.2, the computational and communication overheads are $k \cdot \delta \cdot 0.14$ seconds and $3200 \cdot k$ bits, respectively. For $k(1-\delta)$ new data, clients needs to encrypt these data besides tag generation. Therefore, computational overheads are $k(1-\delta) \cdot (T_{Enc} + 0.14)$ seconds. Likewise, the communication and storage overheads are $k(1-\delta) \cdot (3200 + |m|)$ bits and $k(1-\delta) \cdot (1152 + |m|)$ bits, respectively. Finally, the total overheads of computation, communication and storage are $k((1-\delta) \cdot T_{Enc} + 0.14)$ seconds, $k((1-\delta) \cdot |m| + 3200) + 2048$ bits and $k(1-\delta) \cdot (1152 + |m|)$ bits, respectively. We summarise the comparison of computational, communication and storage overheads with and without data deduplication technique in Table. 2. Furthermore, we test the operation time of data encryption with *AES-CBC* by applying 256-bit AES key, the average operation time is about 89 MB/s. Thus, the encryption time of data m is about $T_{Enc} = \frac{|m|}{89}$ seconds. For big data ($|m|$ is several hundred megabytes to several gigabytes), T_{Enc} is about ten seconds. Hence, the computational costs with deduplication technique are approximately equal to $k((1-\delta) \cdot T_{Enc})$ seconds. Likewise, with the deduplication technique, both communication and storage overheads are approximately equal to $k(1-\delta) \cdot |m|$ bits. Based on the analysis of Table. 2, we can see that the data deduplication technique significantly reduces the computational, communication and storage costs compared with the case of without the deduplication. Further, the effect brought by deduplication becomes better with the increase in variables k, δ and $|m|$.



(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

Fig. 4: A Comparative Summary: Communication Overheads



(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

Fig. 5: A Comparative Summary: Storage Costs

From the above analysis, the proposed EPCDD scheme largely reduces overheads for clients and the CSP compared with the case of without the deduplication technique, especially when the uploaded data number k and data size $|m|$ are large and the duplication ratio δ is high, i.e., for big data. Further, our EPCDD scheme is indeed efficient in terms of computational, communication and storage costs compared with the up-to-date works, which shows the significance and practical potential of our EPCDD scheme to support big data deduplication and storage.

7 Related work

Data deduplication technique has increasingly been used in cloud storage services, such as Dropbox [27], Google Drive [28], Mozy [29], Sipderoak [30], to reduce storage space and the associated costs. Such technique can be broadly categorized based on the level of granularity, namely: file-level and chunk-level. We refer interested reader to [31], [32] for a performance comparison between file-level and chunk-level deduplication approaches.

In recent times, secure data deduplication has been studied by the research community [33]. Convergent encryption (CE), also known as content hash keying, is a cryptosystem that produces identical ciphertexts from identical plaintext files and has been widely applied in secure data deduplication [34], [35], [36]. Bellare et al. [10] formalize a new cryptographic primitive, message-locked encryption (MLE), to improve the security of CE. However, because of the use of a deterministic and message-dependent symmetric key [9], these approaches suffer from inherent security limitations shown in [7]. To enhance the security of deduplication and ensure data confidentiality, Keelveedhi et al. [9] explained how one can ensure data confidentiality by transforming the predictable message into an unpredictable message. In their system, a key server is introduced to generate the file tag for duplication check. However, the third party server suffers from the single point of failure. Thus, Liu et al. [37] proposed the first secure cross-user deduplication scheme that supports client-side encryption without requiring any additional independent servers.

Generally speaking, efficiency is an important indicator to measure whether a scheme can be applied in practice. Hence, just ensuring data confidentiality is insufficient to apply data deduplication in the real-world. Secure data deduplication techniques can also be classified, based on their underlying architecture (i.e. client-side deduplication and server-side deduplication). Client-side deduplication acts on the data at the client before it is transferred. As shown in [38], [39], such an approach can result in bandwidth and storage savings. However, client-side deduplication scheme increases the overhead of computation and key management for clients. Thus, Li et al. [40] attempted to address the problem of achieving efficient and reliable key management in secure deduplication. In server-side deduplication, the target storage server handles deduplication, and the client is unaware of

TABLE 2: Performance Comparison (With and Without Data Deduplication Technique) in EPCDD

	Computational overhead (s)	Communication overhead (bits)	Storage costs (bits)
Without deduplication	$k * T_{Enc}$	$k * \lvert m \rvert$	$k * \lvert m \rvert$
With deduplication	$k * (T_{Enc}(1-\delta) + 0.14)$	$k * (3200 + \lvert m \rvert * (1-\delta)) + 2048$	$k * (1-\delta) * (1152 + \lvert m \rvert)$

any deduplication that might occur. As shown in [41], server-side deduplication can improve storage utilization, but does not result in any bandwidth. In order to save on bandwidth and storage, as well as reducing computation overheads, a client-side cross-user deduplication scheme was proposed in [42]. Jiang et al. [12] introduced a new primitive, μR-MLE2, which is based on the fully randomized scheme (R-MLE2) proposed by Abadi et.al [11]. It reduces the time complexity of deduplication equality test from linear [11] to efficient logarithmic time over the whole data items.

As observed in [8], in addition to achieving efficiency in storage, communication and computation, reliability, security and privacy should also be taken into consideration when designing a deduplication scheme. Li et al. [43] attempted to formalize the notion of distributed reliable deduplication system, and Zhou et al. [13] proposed an efficient secure deduplication scheme (SecDep), which employs User-Aware Convergent Encryption (UACE) and Multi-Level Key management (MLK) approaches to resist brute-force attacks. To resist file-stealing attacks [7] and achieve the access authorization [44], a number of proofs-of-ownership (PoW) schemes have been presented in the literature, such as those in [45], [46]. While there are a number of deduplication schemes designed to improve efficiency and achieve security and privacy assurances, these schemes are not adequate to handle big data. Predictably, designing deduplication schemes to handle big data is an ongoing research topic. For example, Yan et al. [14] proposed a scheme to deduplicate encrypted big data stored in cloud based on ownership challenge and proxy re-encryption. The scheme integrates data deduplication with access control to flexibly support data access control and revocation. However, this scheme is unable to resist brute-force attacks.

As we have previously remarked, existing schemes do not achieve both brute-force attack resilience, privacy and efficiency. This is the gap we contributed to in this paper.

8 Conclusion

Cloud storage adoption, particularly by organizations, is likely to remain a trend in the foreseeable future. This is, unsurprising, due to the digitization of our society. One associated research challenge is how to effectively reduce cloud storage costs due to data duplication.

In this paper, we proposed an efficient and privacy-preserving big data deduplication in cloud storage for a three-tier cross-domain architecture. We then analyzed the security of our proposed scheme and demonstrated that it achieves improved privacy-preserving, accountability and data availability, while resisting brute-force attacks. We also demonstrated that the proposed scheme outperforms existing state-of-the-art schemes, in terms of computation, communication and storage overheads. In addition, the time complexity of duplicate search in our scheme is an efficient logarithmic time.

Future research includes extending the proposed scheme to fully protect the duplicate information from disclosure, even by a malicious CSP, without affecting the capability to perform data deduplication. Future research agenda will also include extending the scheme to be resilient against a wider range of security threats by external attackers, as well as improving the time complexity of duplicate search.

Acknowledgment

The authors would like to thank the Editor and the three anonymous reviewers for providing constructive and generous feedback. X. Yang and X. Tang are supported in part by the National Science Foundation of China under Grant 61325005 and the Major Frontier Project of Sichuan Province under Grant 2015JY0282, R. Lu is supported in part by Natural Sciences and Engineering Research (NSERC) Discovery Grants (no. Rgpin 04009), NBIF Start-Up Grant (Rif 2017-012), and URF Grant (no. 124419), and K.-K.R. Choo is supported by the Cloud Technology Endowed Professorship.

References

[1] IDC, "Executive summary: Data growth, business opportunities, and the it imperatives,," http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm, 2014.

[2] J. Gantz and D. Reinsel, "The digital universe decade-are you ready," https://hk.emc.com/collateral/analyst-reports/idc-digital-universe-are-you-ready.pdf.

[3] H. Biggar, "Experiencing data de-duplication: Improving efficiency and reducing capacity requirements," *The Enterprise Strategy Group.*, 2007. [Online]. Available: http://journals.sagepub.com/doi/abs/10.1177/000944550704300309

[4] D. Quick and K.-K. R. Choo, "Impacts of increasing volume of digital forensic data: A survey and future research challenges," *Digital Investigation*, vol. 11, no. 4, pp. 273–294, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.diin.2014.09.002

[5] ——, "Big forensic data reduction: digital forensic images and electronic evidence," *Cluster Computing*, vol. 19, no. 2, pp. 723–740, 2016. [Online]. Available: http://dx.doi.org/10.1007/s10586-016-0553-1

[6] M. Dutch, "Understanding data deduplication ratios," http://www.chinabyte.com/imagelist/2009/222/l3pm284d8r1s.pdf, 2009.

[7] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," *IEEE Security & Privacy*, vol. 8, no. 6, pp. 40–47, 2010. [Online]. Available: http://dx.doi.org/10.1109/MSP.2010.187

[8] J. Paulo and J. Pereira, "A survey and classification of storage deduplication systems," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 11:1–11:30, 2014. [Online]. Available: http://doi.acm.org/10.1145/2611778

[9] 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*, 2013, pp. 179–194. [Online]. Available: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/bellare

[10] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, 2013, pp. 296–312. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38348-9_18

[11] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, 2013, pp. 374–391. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40041-4_21

[12] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, "Secure and efficient cloud data deduplication with randomized tag," *IEEE Trans. Information Forensics and Security*, vol. PP, no. 99, pp. 1–1, 2016. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2016.2622013

[13] Y. Zhou, D. Feng, W. Xia, M. Fu, F. Huang, Y. Zhang, and C. Li, "Secdep: A user-aware efficient fine-grained secure deduplication scheme with multi-level key management," in *IEEE 31st Symposium on Mass Storage Systems and Technologies, MSST 2015, Santa Clara, CA, USA, May 30 - June 5, 2015*, 2015, pp. 1–14. [Online]. Available: http://dx.doi.org/10.1109/MSST.2015.7208297

[14] 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. [Online]. Available: http://dx.doi.org/10.1109/TBDATA.2016.2587659

[15] "Prism (surveillance program)," https://www.theguardian.com/us-news/prism.

[16] R. Bhaskar, S. Guha, S. Laxman, and P. Naldurg, "Verito: A practical system for transparency and accountability in virtual economies," in *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*, 2013. [Online]. Available: http://internetsociety.org/doc/verito-practical-system-transparency-and-accountability-virtual-economies

[17] D. Boyd, K. Crawford, S. Shaikh, and V. Ravishankar, "Six provocations for big data," http://www.ils.albany.edu/wordpress/wp-content/uploads/2016/01/Six-provocations-for-Big-Data1.pdf.

[18] X. Yang, R. Lu, H. Liang, and X. Tang, "SFPM: A secure and fine-grained privacy-preserving matching protocol for mobile social networking," *Big Data Research*, vol. 3, pp. 2–9, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.bdr.2015.11.001

[19] A. K. Mohan, E. Cutrell, and B. Parthasarathy, "Instituting credibility, accountability and transparency in local service delivery?: helpline and aasthi in karnataka, india," in *International conference on information and communication technologies and development, ICTD 2013, Cape Town, South Africa, December 7-10, 2013, Volume 1: Papers*, 2013, pp. 238–247. [Online]. Available: http://doi.acm.org/10.1145/2516604.2516622

[20] D. Boneh, E. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, 2005, pp. 325–341. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30576-7_18

[21] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, 2007, pp. 535–554. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-70936-7_29

[22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms (3. ed.)*. MIT Press, 2009. [Online]. Available: http://mitpress.mit.edu/books/introduction-algorithms

[23] D. E. Knuth, *The Art of Computer Programming, Volume III: Sorting and Searching*. Addison-Wesley, 1973.

[24] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006. [Online]. Available: http://doi.acm.org/10.1145/1127345.1127346

[25] B. Lynn, "Pbc library," https://crypto.stanford.edu/pbc/.

[26] "Openssl," https://www.openssl.org/.

[27] "Dropbox, a file-storage and sharing service," http://www.dropbox.com.

[28] "Google drive," http://drive.google.com.

[29] "Mozy: A file-storage and sharing service." http://mozy.com/.

[30] "Spideroak," https://www.spideroak.com/.

[31] D. T. Meyer and W. J. Bolosky, "A study of practical deduplication," *TOS*, vol. 7, no. 4, p. 14, 2012. [Online]. Available: http://doi.acm.org/10.1145/2078861.2078864

[32] C. Policroniades and I. Pratt, "Alternatives for detecting redundancy in storage systems data," in *Proceedings of the General Track: 2004 USENIX Annual Technical Conference, June 27 - July 2, 2004, Boston Marriott Copley Place, Boston, MA, USA*, 2004, pp. 73–86. [Online]. Available: http://www.usenix.org/publications/library/proceedings/usenix04/tech/general/policroniades.html

[33] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. R. Weippl, "Dark clouds on the horizon: Using cloud storage as attack vector and online slack space," in *20th USENIX Security Symposium, San Francisco, CA, USA, August 8-12, 2011, Proceedings*, 2011. [Online]. Available: http://static.usenix.org/events/sec11/tech/full_papers/Mulazzani6-24-11.pdf

[34] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in *ICDCS*, 2002, pp. 617–624. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2002.1022312

[35] M. W. Storer, K. M. Greenan, D. D. E. Long, and E. L. Miller, "Secure data deduplication," in *Proceedings of the 2008 ACM Workshop On Storage Security And Survivability, StorageSS 2008, Alexandria, VA, USA, October 31, 2008*, 2008, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/1456469.1456471

[36] A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, and J. C. S. Lui, "A secure cloud backup system with assured deletion and version control," in *2011 International Conference on Parallel Processing Workshops, ICPPW 2011, Taipei, Taiwan, Sept. 13-16, 2011*, 2011, pp. 160–167. [Online]. Available: http://dx.doi.org/10.1109/ICPPW.2011.17

[37] 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 12-6, 2015*, 2015, pp. 874–885. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813623

[38] J. Xu, E. Chang, and J. Zhou, "Weak leakage-resilient client-side deduplication of encrypted data in cloud storage," in *8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013*, 2013, pp. 195–206. [Online]. Available: http://doi.acm.org/10.1145/2484313.2484340

[39] N. Kaaniche and M. Laurent, "A secure client side deduplication scheme in cloud storage environments," in *6th International Conference on New Technologies, Mobility and Security, NTMS 2014, Dubai, United Arab Emirates, March 30 - April 2, 2014*, 2014, pp. 1–7. [Online]. Available: http://dx.doi.org/10.1109/NTMS.2014.6814002

[40] J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 6, pp. 1615–1625, 2014. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2013.284

[41] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou, "A hybrid cloud approach for secure authorized deduplication," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1206–1216, 2015. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2014.2318320

[42] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," *IACR Cryptology ePrint Archive*, vol. 2011, p. 207, 2011. [Online]. Available: http://eprint.iacr.org/2011/207

[43] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang, M. M. Hassan, and A. Alelaiwi, "Secure distributed deduplication systems with improved reliability," *IEEE Trans. Computers*, vol. 64, no. 12, pp. 3569–3579, 2015. [Online]. Available: http://dx.doi.org/10.1109/TC.2015.2401017

[44] K. Zhang, X. Liang, J. Ni, K. Yang, and X. S. Shen, "Exploiting social network to enhance human-to-human infection analysis without privacy leakage," *IEEE Trans. Dependable Sec. Comput.*, vol. PP, no. 99, pp. 1–1, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7738429/

[45] R. D. Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication," in *7th ACM Symposium on Information, Compuer and Communications Security, ASIACCS '12, Seoul, Korea, May 2-4, 2012*, 2012, pp. 81–82. [Online]. Available: http://doi.acm.org/10.1145/2414456.2414504

[46] J. Hur, D. Koo, Y. Shin, and K. Kang, "Secure data deduplication with dynamic ownership management in cloud storage," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 11, pp. 3113–3125, 2016. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2016.2580139

Xue Yang received the B.S. degree in information security from the Southwest Jiaotong University, Chengdu, China, in 2012. She is currently working towards a Ph.D. degree in information and communication engineering, Southwest Jiaotong University. Her research interests include big data security and privacy, applied cryptography and network security.

Rongxing Lu (S'09-M'10-SM'15) has been an assistant professor at the Faculty of Computer Science, University of New Brunswick (UNB), Canada, since August 2016. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious Governor General's Gold Medal, when he received his PhD degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is presently a senior member of IEEE Communications Society. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. Dr. Lu currently serves as the Secretary of IEEE ComSoc CIS-TC.

Kim-Kwang Raymond Choo (SM'15) received the Ph.D. in Information Security from Queensland University of Technology, Australia. He currently holds the Cloud Technology Endowed Professorship at The University of Texas at San Antonio, and is a Fellow of the Australian Computer Society. He was named Cybersecurity Educator of the Year – APAC (2016 Cybersecurity Excellence Awards produced in cooperation with the Information Security Community on LinkedIn) in 2016. In 2015, he and his team won the Digital Forensics Research Challenge organized by Germany's University of Erlangen-Nuremberg. He was named one of 10 Emerging Leaders in the Innovation category of The Weekend Australian Magazine/Microsofts Next 100 series in 2009, and his other awards include ESORICS 2015 Best Research Paper Award, Highly Commended Award from Australia New Zealand Policing Advisory Agency (2014), Fulbright Scholarship (2009), 2008 Australia Day Achievement Medallion, and British Computer Societys Wilkes Award (2007).

Fan Yin received the B.S. degree in information security from the Southwest Jiaotong University, Chengdu, China, in 2012. He is currently working toward the Ph.D. degree in information and communication engineering, Southwest Jiaotong University.

His research interests include privacy-preserving and security for cloud security and network security.

Xiaohu Tang (M'04) received the B.S. degree in applied mathematics from the Northwest Polytechnic University, Xi'an, China, the M.S. degree in applied mathematics from the Sichuan University, Chengdu, China, and the Ph.D. degree in electronic engineering from the Southwest Jiaotong University, Chengdu, China, in 1992, 1995, and 2001 respectively.

From 2003 to 2004, he was a research associate in the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology. From 2007 to 2008, he was a visiting professor at University of Ulm, Germany. Since 2001, he has been in the School of Information Science and Technology, Southwest Jiaotong University, where he is currently a professor. His research interests include coding theory, network security, distributed storage and information processing for big data.

Dr. Tang was the recipient of the National excellent Doctoral Dissertation award in 2003 (China), the Humboldt Research Fellowship in 2007 (Germany), and the Outstanding Young Scientist Award by NSFC in 2013 (China). He serves as Associate Editors for several journals including IEEE Transactions on Information Theory and IEICE Trans on Fundamentals, and served on a number of technical program committees of conferences.