High Efficient Secure Data Deduplication Method for Cloud Computing

Yongan Guo, Chunlei Jiang

College of Telecommunications & Information Engineering, Nanjing University of Posts and Telecommunications, China
guo@njupt.edu.cn, 1017010208@njupt.edu.cn

Abstract

Deduplication technology helps reduce the overhead of cloud storage services, which clearly benefits the development of the increasingly big-data-driven society. However, data stored in the cloud is usually in an encrypted form since users request high levels of data security and privacy. Thus, data encryption brings new challenges to deduplication in cloud computing, making deduplication for encrypted data more challenging. Because conventional deduplication solutions cannot handle encrypted data, and there are various problems with existing deduplication schemes, this paper proposes an improved scheme based on ownership challenge and proxy re-encryption to enhance its security performance and attain an efficient and secure deduplication. A theoretical analysis and computer simulation is performed to evaluate its performance. The performance analysis shows that the scheme is efficient and has good attack resistance in the big data deduplication for cloud computing.

Keywords: Big data, Cloud computing, Deduplication

1 Introduction

Nowadays, cloud storage has been extensively used and IDC predicted that the size of the world's total digital data will reach 44 ZB by 2020 [1]. Moreover, other studies have shown that around 75% digital data is identical [2] and the repetition rate of which in backup systems exceeds 90% [3]. Although storage costs are relatively low, and with the evolution of cloud storage solutions, more and more data can be stored, the cost associated with managing, maintaining and processing this large amount of data can be significant [4-5].

Deduplication solutions allow cloud storage systems to search for and remove duplicate data as well as ensure data availability. By storing only one copy of a file and replacing the redundant copy with a reference to the saved one, deduplication achieves its goal to store much more data in the same space. That is to say, deduplication technology can significantly help cloud server reduce storage and network burden. Dropbox, Google Drive, Mozy and other cloud storage service providers have adopted deduplication technologies to reduce the storage resource consumption of cloud storage managers, which further reduces the cost of managing and maintaining big data.

However, due to the distrust of cloud users on the cloud service providers, these data holders tend to encrypt their data before uploading. Data encryption is an important method to keep private data safe, but it restricts data re-usability. This makes deduplication much more difficult because the same data encrypted by different keys will derive different ciphertexts [6-7]. Therefore, developing new solutions to coordinate deduplication and client-side encryption and then effectively de-duplicate encrypted data is an important research direction in cloud computing.

2 Related Work

Recently, extensive and in-depth research has been conducted on secure deduplication technologies. Convergence Encryption (CE) is a cryptographic system that produces the same ciphertext from the same plaintext file and has been widely used for secure deduplication [8-10]. Message locked encryption (MLE) proposed in [11] improved the security of the CE. However, these methods have some inherent security restrictions [12]. To cope with the security problem mentioned above and ensure the confidentiality of data, document [12] explained how to ensure the confidentiality of data by converting predictable messages into unpredictable messages. This research esigned a key server which is responsible for repeated inspection in the scheme, but the whole system will break down as long as the attached server fails. Therefore, [13] proposed a solution that can perform encryption at the side of clients without any extra stand-alone server.

In fact, ensuring data confidentiality exclusively is not sufficient to apply deduplication. Efficiency is also an important index to examine the feasibility of a scheme. Therefore, based on R-MLE2 proposed in [15],

*Corresponding Author: Yongan Guo; E-mail: guo@njupt.edu.cn
DOI: 10.3966/160792642020032102023

μR-MLE2 was presented in [14]. This reduces the temporal complexity of the deduplication equivalence test on the data elements, from linear time to logarithmic time.

Recently, more and more researchers took both the reliability and efficiency of their schemes into consideration while designing deduplication solutions. In [16], Li et al. formalized a concept of reliable distributed deduplication system. [17] proposed an efficient secure deduplication solution (SecDep) that possesses resistance to brute-force attacks by using UACE (User-Aware Convergence Encryption) and MLK (Multi-level Key) management methods. Besides, in order to combat document theft attacks, many proposals for data ownership (PoW) have been made, such as the ones in [18] and [19].

Although many deduplication programs are created for efficiency and security guarantees, those programs are not suitably enough in the era of big data. It is clear that to design deduplication solutions for big data environments is an emerging research topic. In 2016, Yan et al. [20] proposed a solution based on ownership challenge and proxy re-encryption to deduplicate encrypted big data stored in the cloud. The solution can flexibly support the control and retrieval of data access through the combination of deduplication and access control features. However, the system cannot resist brute-force attack, which is a common threat in a secure deduplication scenario [11-12].

To achieve security, availability and efficiency in data de-duplication solutions as well as resist brute-force attacks, Yang et al. proposed an efficient and privacy-preserving big data deduplication in cloud storage (EPCDD) based on a three-tier cross-domain architecture [21]. EPCDD realizes several design goals, including privacy protection, data availability and resistance against brute-force attacks. In addition, the complexity of EPCCD searching duplication is logarithmic.

Before presenting our improvement plan, we will briefly introduce two kinds of deduplication schemes mentioned above: (1) the scheme based on ownership challenge and proxy re-encryption proposed by Yan et al. [20], and (2) the EPCCD proposed by Yang et al. [21].

2.1 A Scheme Based on Ownership Challenge and Proxy Re-encryption

The proposed solution consists of three types of entities, as shown in Figure 1:

Cloud service provider (CSP) that provides the storage service. CSP is not completely reliable, because of its curiosity of the data content stored on its server. However, it has to honestly conduct data storage for commercial profits and assume the work of proxy re-encryption (PRE) [22];

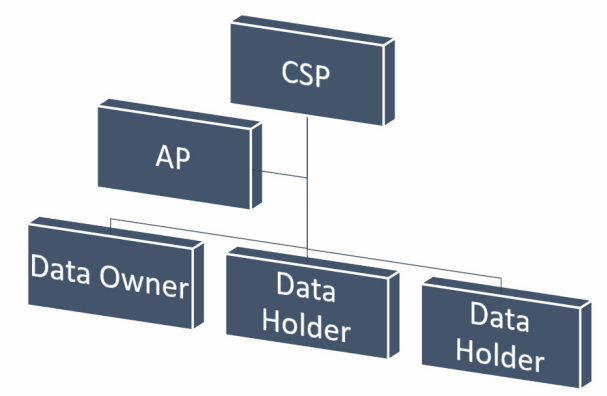

Figure 1. System model of the scheme based on ownership challenge and proxy re-encryption

Data holders who temp to save their data on CSP servers. In this system, there may be multiple qualified data holders who want to upload same encrypted raw data so that the deduplication scheme can play its due role. Data holders who create files are considered as data owners and therefore have higher priorities than common data holders;

Authorized party (AP) which has considerable computing resources. This entity is in charge of verifying data ownership and carrying out deduplication function. By definition, neither AP nor CSP knows the data stored on the other party's server.

The advantage of this scheme is that it realized flexible management over access control. Besides, this scheme can perform deduplication in the era of big data with high efficiency, expending only some additional calculation overhead and extra communication cost. The scheme is cost-efficient compared to big data upload and storage costs.

The disadvantage of this scheme lies in its inability to resist brute-force attacks from malicious CSPs. In fact, SPs inevitably have some plaintext space information. That is, some users may store unencrypted data on the CSP servers. Using these plaintext information, the CSP may find some relevance between ciphertext and plaintext, resulting in information disclosure. This is also a common problem that needs to be solved in many encryption deduplication solutions.

2.2 EPCDD

The architecture of EPCDD contains four types of entities. They constitute a three-layer cross-domain deduplication framework as shown in Figure 2. The four types of entities are listed as follows:

KDC. Key distribution center (KDC) is an extra party which distributes and manages system keys.

CSP. The first layer of the model is the cloud service provider (CSP), which provides customers with data storage services. The CSP is capable of supporting the customers' storage needs. Also, it can reduce expenses associated with big data management by removing duplicate data.

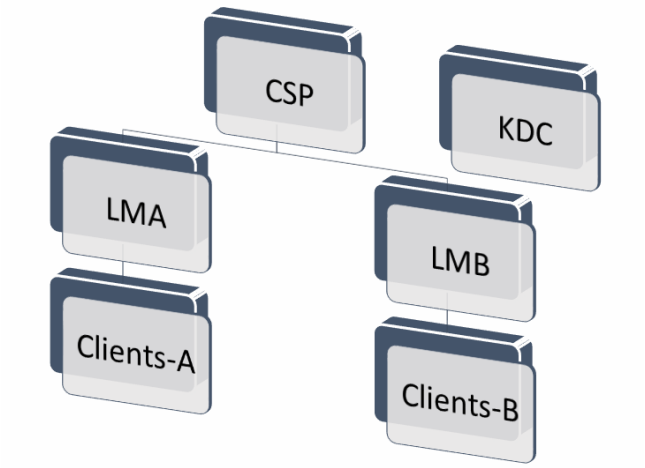

Figure 2. The system model of the EPCDD scheme

LM. The second layer refers to several domains (such as companies or universities) that have service contract with the CSP. The local manager (LM) of each domain (e.g., LMA and LMB in Figure 2) is responsible for internal deduplication and message forwarding from local clients or CSP. For simplicity, Figure 2 illustrates the model of two domains. But the model can be extended to three or more domains as well.

Client. The last layer consists of customers affiliated with a domain (e.g., company employees or students in university). Clients take responsibility for encrypting data to protect privacy and generating tags to enable data deduplication.

EPCDD provides an efficient and confidential solution of big data deduplication in the cloud by introducing a three-tier system. This scheme realized its design goals including privacy protection, accountability and resistance against brute-force attacks. Moreover, it outperforms other recent deduplication schemes in total overhead and time complexity of duplication search.

However, EPCDD requires the CSP to construct a deduplication decision tree for each domain and this will bring about some additional costs. Moreover, the time complexity of searching for duplicate data may no longer be a strict logarithmic time when domains are extended to a considerable number.

3 An Enhanced Scheme Based On Ownership Challenge and Proxy Re-Encryption

In this section, we will introduce the enhanced version of the scheme mentioned in section 2, including the detailed system model and procedures.

3.1 System Model

Similar to the model mentioned in section 2, the entire system consists of three types of entities:

Cloud Service Provider (CSP). CSP provides users with data storage services and performs agent re-encryption. The hypothesis of CSP is that it is honest but curious, which was widely used in other studies [12-14]. Specifically, since CSPs obtain commercial profits by providing storage services, they need to take the influence of reputation and laws into account (for example, civil lawsuits may bring about significant reputation and financial losses), and therefore they will not actively modify the stored information. However, because of huge amounts of data in CSP servers, it may have some plain text information. CSP (for example, malicious CSP employees) may use brute-force attacks to obtain some relationships between plaintext and ciphertext (or tags), and then achieve certain illegal purposes using these information (for example, selling users' data for pelf).

Authorized Party (AP). AP is responsible for verifying data ownership and handling deduplication. The AP is completely trusted by users and does not collude with CSP. In theory, the AP may collude with the CSP to obtain the user's private information. However, once the collusion is disclosed, it will bring unpredictable losses to the CSP (reputation, financial, etc.).

Data holders, or users. In this system, there may be more than one legitimate data holder for a specific file, so that the deduplication program can play its due role. Users are assumed to be honest, although they may also collude with CSPs to obtain data that does not belong to them. However, the one who invaded privacy of others also faces the risk of being invaded by other ones. And as mentioned above, once the collusion is disclosed, it will bring huge losses to the CSP, and users will not have the chance to enjoy the convenient cloud storage service any more.

3.2 Procedures

3.2.1 Encrypted Data Upload

User u_1 attempts to save its own data M in the cloud. Assuming that the same data does not exist in the CSP, the system performs the encrypted data upload program, as shown in Figure 3:

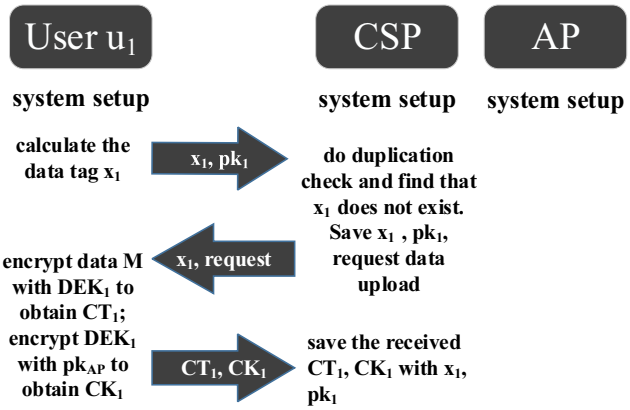

Figure 3. Encrypted Data Upload

System setup. In the system setup process, each user u_i in the system generates sk_i and pk_i ($\mathrm{sk}_i = a_i$, $\mathrm{pk}_i = g^{a_i}$, $\mathrm{a}_i \in Z_p$) for PRE. The public key pk_i is used to generate the re-encryption key for u_i on the AP. Then, u_i generates another key pair (s_i, V_i) based on Elliptic Curve Cryptography (ECC), where s_i is the private key and $V_i = -s_i * P$ is the public one. P is the shared base point of elliptic curve Eq(a, b) among all entities. AP independently generates a random ω, key pair (pk_{AP} , sk_{AP}) and parameter σ, and broadcasts ω, σ and pk_{AP} to users.

Tag generation. User u_1 generates a tag of data M: $x_1 = H(H(M) * P) \bmod \omega$ and then sends x_1 and pk_1 to CSP.

Duplication check. CSP searches for duplicate data by checking if x_1 exists. Here, we assume that the result is negative. CSP returns x_1 to u_1 and requests u_1 for data upload.

Data upload. User u_1 encrypts M with symmetric key DEK_1 to obtain ciphertext CT_1, and then encrypts DEK_1 with pk_{AP} to obtain CK_1 . u_1 sends CT_1 and CK_1 to the CSP. CSP saves them with x_1 and pk_1 .

3.2.2 Data Deduplication

The data holder u_2 also wants to save M in the cloud. However, the CSP server has already saved ciphertext of M (CT_1), and the system executes the deduplication program, as shown in Figure 4:

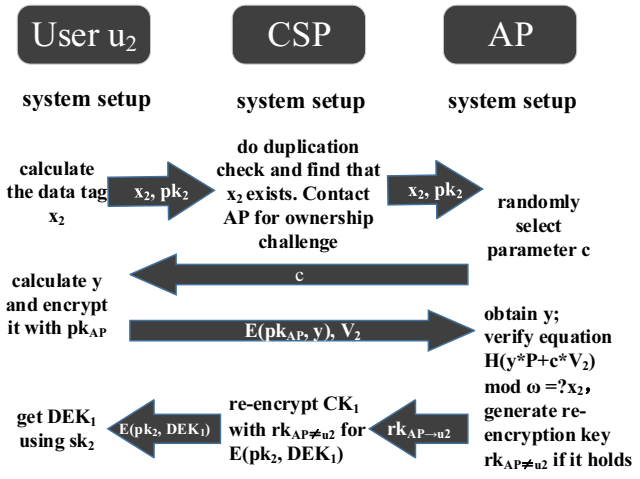

Figure 4. Data Deduplication

System setup. This process is similar to the one described above, so we skip this part.

Tag generation. User u_2 generates a data tag $x_2 = H(H(M) * P) \bmod \omega$, which will be sent to CSP hereafter with its public key pk_2 .

Duplication check. CSP finds the same record in its tag database, indicating the existence of duplicate data. So, CSP forwards x_2 and pk_2 to the AP for ownership challenge.

Ownership challenge. AP issues ownership challenge to u_2. First, AP chooses a number $c \in \{0,...,2^{\sigma} - 1\}$ randomly and sends it to u_2 . The data holder u_2 calculates $y = H(M) + s_2 * c$ after receiving c and encrypts y using pk_{AP} . Then u_2 sends $E(pk_{AP}, y)$ and V_2 to the AP. After the AP receives the data sent by u_2, it verifies the equation $H(y * p + c * V_2) \bmod \omega = x_2$. If the equation holds, the re-encryption key $rk_{AP \to u_2}$ will be calculated and sent to CSP.

Proxy re-encryption. After receiving the re-encryption key, the CSP re-encrypts CK_1 to obtain $E(pk_2, DEK_1)$ (DEK_1 encrypted with pk_2), then sends it to u_2 and allows u_2 to visit CT_1 . At this time, u_2 can use its private key sk_2 to decrypt $E(pk_2, DEK_1)$ to obtain DEK_1 , so as to obtain M included in CT_1 without uploading its own data.

3.2.3 Data Deletion

Sometime, the data holder u_2 wants to delete useless data M from the CSP to free up storage space. After the CSP receives the request sent by u_2 , it deletes the deduplication record of u_2 and prevents u_2 from accessing CT_1 . The encrypted data CT_1 will also be deleted if CSP finds that the deduplication record is empty, which indicates that no one needs M any more.

3.2.4 Encrypted Data Update

CSP is able to support the function of data update. As shown in Figure 5, u_1 wants to update its encrypted data stored in the CSP to better assure the privacy security. It generates updated ciphertexts $CT_1^{'}$ and $CK_1^{'}$ with the new symmetric key $DEK_1^{'}$, and sends them together with an update request to CSP. CSP saves $CT_1^{'}$, $CK_1^{'}$ with x_1 , pk_1 . Next, CSP must update re-encrypted keys to ensure data availability. AP generates and sends re-encryption keys (e.g., $rk_{AP \to u_2}$) to CSP, which is used to perform re-encryption on $CK_1^{'}$. Afterwards, CSP generates and distributes re-encrypted keys (e.g., $E(pk_2, DEK_1)$) to other eligible data holders.

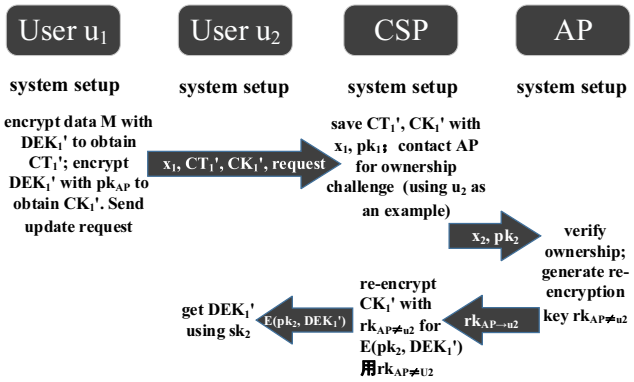

Figure 5. Encrypted Data Update

In practice, CSP has the authority to decide whether such an update request from users can be fulfilled according to user agreement.

3.2.5 Valid Data Control

Not only the storage cost cut through deduplication can benefit the CSP, but also users can directly or indirectly enjoy the benefits of reduced storage costs. However, users sometimes prefer the complete control over data for cost reduction. That means users always want to upload their own data and store it in the cloud, no matter whether the same data exists in the CSP server or not.

As shown in Figure 6, user u_i generates a tag with its ECC secret key s_i to deduplicate the data it uploaded before, and data M can be uploaded as long as u_i never stores the same data before. Notice that unlike the operation aforementioned, u_i uses pk_i to generate CK_i instead of pk_{AP}. This makes u_i liberate from deduplication program. In addition, u_i can use its data freely: sharing, deleting, or updating them. For instance, u_i can independently generate $rk_{u_i \to u_j}$ so that u_i can share data with u_j.

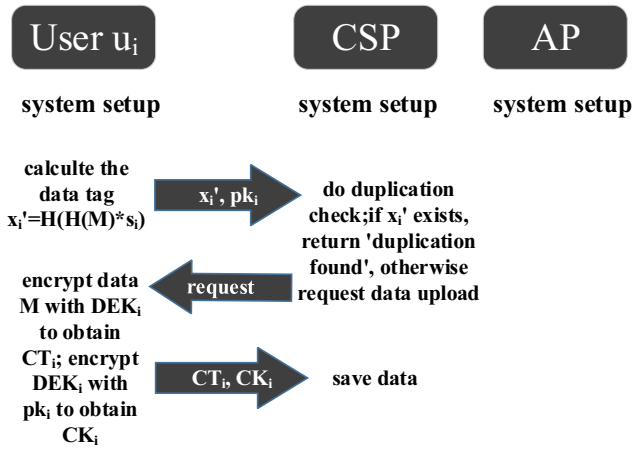

Figure 6. Valid Data Control

4 Security Analysis

This section describes the security features of our scheme, focusing on how to implement privacy protection, brute force attacks resistance, and data availability.

4.1 Privacy Protection

Our scheme protects users' sensitive information from disclosure.

To enable our scheme, besides the encrypted data , user u_i needs to upload the encrypted secret key CK_i, the data tag x_i and the public key pk_i to CSP. In our system model, CSP does not collude with AP, so CSP cannot obtain sk_{AP}, and thus cannot obtain the key DEK_i that the user uses to encrypt the data. In other words, the CSP cannot obtain the user's private data from CT_i and CK_i, so that they are safe. CSP can only obtain information about M from data tag x_i.

Since $x_i = H(H(M)*P) \bmod \omega$, there is an integer k such that $H(H(M)*P) = k*\omega + x_i$. There are two unknown in one equation, thus CSP can only obtain $H(H(M)*P)$ by guessing attacks. However, if the size of $H(H(M)*P)$ is 256 bits, ω can be set to 128 bits so that x_i can fully withstand such attacks while representing up to 2^{128} different messages. To take a step back, even if the CSP obtains $H(H(M)*P)$, information of M contained therein will not be disclosed due to the irreversible nature of the hash cryptographic function.

For AP, although it has the key that can decrypt CK_i, CSP prevents it from accessing data in the cloud. Therefore, among all messages available to AP, only the parameter y used for ownership challenge and data tag x_i contains data M. Since the hash cryptographic function is irreversible, AP cannot obtain the data M from them.

Finally, assuming that malicious users want to acquire data M that does not belong to them, they can never succeed because $H(M)$ can be correctly generated only when data M is actually presented. Then, $y = H(M) + s_i * c$ can be calculated to pass the ownership challenge. The data tag x_i obtained by illegal means is useless. In a word, our scheme can ensure that users' privacy information will not be leaked.

4.2 Brute-force Attacks Resistance

As mentioned in section 2, due to huge amounts of data saved in cloud servers, CSP inevitably owns some plaintext data (sometimes users are not willing to encrypt non-sensitive data before uploading it).

Despite CSP's attempt to derive the hash value

$H(M)$ of all plaintext data, data tag x_i contains a random number ω. As long as CSP does not know this random number, it can never test the correlation between the plaintext data and data tag. In addition, the CSP does not know the key pk_{AP} that AP distributes to users, so it cannot launch attacks to CK_i, let alone the cipher text CT_i.

Assuming that CSP wants to impersonate a user to pass ownership challenge for stealing users' data. It sends data tag x_i and public key pk_i to AP and obtains the parameter c needed for ownership challenge. Then, CSP tries the hash value $H(M)$ of all plaintext data, assuming that it contains the correct hash value. Nevertheless, the CSP does not know the ECC-based key pair (s_i, V_i) of the user u_i. As a result, y cannot be calculated correctly, and CSP cannot pass the ownership challenge.

In summary, the CSP is unable to get the relation between encrypted data and plaintext through brute-force attacks. In addition, because users do not have sufficient resources and AP with sufficient resources cannot obtain the data stored in the CSP, the criteria are not met to launch brute-force attacks.

4.3 Data Availability

Data deduplication schemes must ensure the availability of data. As long as the user has uploaded his own data, no matter which duplicate copy is deleted by deduplication, it must be ensured that the client can obtain their own data.

Specifically, user u_i wants to store its own data M in the cloud. CSP finds that M has already been stored, so it performs deduplication. After the ownership challenge, CSP sends the re-encrypted key $E(pk_j, DEK_i)$ to u_j and authorizes u_j to access the original ciphertext CT_i in CSP server. The user does not need to upload his own data. After a period of time, the user u_j wants to download his own data, and the CSP returns to the user the CT_i. As described in section 3.2.1, $CT_i = E(DEK_i, M)$, the user u_j can decrypt $E(pk_j, DEK_i)$ with its own private key sk_j, to get DEK_i so that it can obtain the data M from CT_i. To sum up, our scheme can ensure the availability of data.

If the data has been updated before user u_j downloads the data, our solution can guarantee the data availability, as described in section 3.

5 Efficiency Analysis

This section measures the efficiency of our scheme in terms of computational overhead and communication overhead, and compares the result with another efficient data deduplication scheme, EPCCD.

5.1 Computational Overhead

In this section, we analyze the computational cost of the whole system during uploading data M in the following two cases: no data is duplicated, and data is duplicated.

In the first case, user u_i and AP first set the system, and each generates a key pair (pk_i, sk_i) and (pk_{AP}, sk_{AP}), requiring 2 exponential calculations. User u_i needs to generate another key pair (s_i, V_i), which requires a point multiplication. Besides, in order to help the execution of deduplication, u_i needs to calculate data tag x_i, which in turn requires a point multiplication. Notice that the calculation of hash value is affected by file size, but it is very fast and is negligible compared to the exponential or point multiplication operations mentioned above. In the same way, the cost of checking duplications in the server using x_i by CSP can also be ignored. Moreover, since the computational overhead of encrypting data M is unavoidable in all encrypted data deduplication schemes, the overhead of this part is skipped. Finally, encrypting CK_i using the PRE method requires 2 exponential operations. In summary, the total computational overhead of the system is: 4 exponential operations and 2 point multiplications.

For the second case, user u_i also sets the system first. This requires 1 exponent operation and 1 point multiplication. User u_i computes the data tag x_i which needs 1 point multiplication. After the CSP receives the tag x_i, it finds a duplication and forwards the tag x_i and the public key pk_i to AP for ownership challenge. In the course of ownership challenge, u_i calculates y that requires 1 point multiplication and $E(pk_{AP}, y)$ that requires 2 exponential operations. AP requires 1 exponent operation to decrypt y and 2 point multiplications to verify ownership. Generating a re-encryption key requires 1 exponent operation. The CSP requires 1 bilinear pairing operation for re-encryption operation. Finally, user u_i needs 1 exponential operation to decrypt and obtain DEK. In summary, the overall computational cost of the system is: 6 exponential operations, 5 point multiplications and 1 bilinear pairing operation.

If there are n users who need to upload data, and the repetition rate of the data is k ($0 < k < 1$), the total cost is $n*(4T_e + 2T_m + k*(2T_e + 4T_m + T_p))$, where T_e, T_m, T_p represent the time taken for exponential, point multiplication and bilinear pairing operations, respectively. For the EPCCD scheme used for comparison, the computational cost of the entire

system is $n*(6T_e + T_m + T_p)$ [21].

In order to measure the computational overhead of the system more intuitively, we tested specific values for T_e, T_m and T_p. The test environments are: 3.2GHz processor 4GB memory computer, Ubuntu 16.04, and PBC Library. We did 100 tests each and averaged them. The results are shown in Table 1: T_e = 17.574 ms、T_m = 0.013 ms、T_p = 22.633 ms.

Table 1. Basic operation costs

Types	T_e	T_m	T_p
Costs (ms)	17.574	0.013	22.633

5.2 Communication Overhead

In this section, we also analyze the communication cost while uploading data M under two conditions: no data is duplicated, and data is duplicated. Since the uploading of encrypted data is unavoidable, we ignore the communication overhead.

If no data is duplicated in the CSP server, in addition to the communication overhead of sending the ciphertext, only $2\,x_i$, pk_i and CK_i need to be sent. If our scheme chooses the SHA-256 as hash function, the total communication overhead is 2*128+1024+256= 1536 bits.

If data is duplicated, the communication contents apart from the ciphertext include: $2\,x_i$, $2\,pk_i$, c, $E(pk_{AP}, y)$, V_i, $rk_{AP\to u_i}$, and $E(pk_i, DEK)$. If we set DEK size to 256 bits, the ownership challenge parameter c and ECC key V_i to 160 bits, the total overhead is 5952 bits.

According to the above, assuming that there are n users who need to upload data, where the data repetition rate is k (0 < k <1), the total communication overhead is $n*(1536 + k*4416)$ bits. And the overhead of EPCDD scheme is $n*3200 + 2048$ bits [21].

6 Conclusion

The widespread use of cloud storage remains a constant trend in our big data-driven society. With the explosive growth of data volume, duplicate data will inevitably increase. In order to reduce the burden of cloud servers and also effectively clean up duplicate data under complex conditions (the presence of encrypted data), it is significant and very important to research the deduplication scheme of encrypted data.

In this paper, we improved a deduplication scheme based on ownership challenge and proxy re-encryption to enhance its security performance, specifically to increase its resistance to brute-force attacks from CSP.

Our scheme can ease the burden on cloud storage servers because it allows servers not to maintain multiple copies of the same data. The cost is to take up some extra storage space to save the user's public key and the data tag that helps the CSP search for duplicate data, and the encrypted DEK. However, this storage cost is acceptable compared to large amounts of duplicate data.

Our scheme performs excellently in big data environment. In this scenario, it is not complicated to perform data deduplication, which means that it takes only a short time to complete the work. The extra computational and communication burden introduced in this process (as described in section 5) has nothing to do with the size of the data itself and has very little impact on the result. Compared to the cost of encrypting and uploading big data, these extra burdens can be said to be insignificant.

In the process of efficiency analysis, we compared the system overhead with the EPCDD scheme (see Table 2). The comparison shows that our scheme is slightly inferior, and EPCCD is indeed a superior deduplication scheme. However, as described in section 2, EPCDD may encounter some difficulties in practical applications. In addition, our scheme possesses the flexibility that EPCCD does not have. With ownership challenges, our scheme has the flexibility to control access to encrypted data. And to support the update of encrypted data, new keys can be easily issued by CSPs to qualified users at low cost.

Table 2. Efficiency comparison of the two schemes

	Computing Overhead	Communication Overhead
Our Scheme	$n*(4T_e + 2T_m + k*(2T_e + 4T_m + T_p))$	$n*(1536 + k*4416)$
EPCDD Scheme	$n*(6T_e + T_m + T_p)$	$n*3200 + 2048$

Further researches on this scenario include enhancements to its security capabilities to ensure security in the face of malicious CSPs or more diversified attacks, while simultaneously improving the performance of its search duplication and making it more efficient.

Acknowledgments

The work is supported by the National Key R&D Program of China (2018YFC1314900), the Natural Science Foundation of China under Grant 61871446.

References

[1] IDC, *Executive Summary: Data Growth, Business Opportunities, and the IT Imperatives*, #IDC_1672, April, 2014.

[2] J. Gantz, D. Reinsel, *The Digital Universe Decade – Are You Ready?*, #IDC_1672, May, 2010.

[3] H. Biggar, *Experiencing Data De-duplication: Improving Efficiency and Reducing Capacity Requirements*, ESG White Paper, February, 2007.

[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, December, 2014.

[5] D. Quick, K. K. R. Choo, Big Forensic Data Reduction: Digital Forensic Images and Electronic Evidence, *Cluster Computing*, Vol. 19, No. 2, pp. 723-740, June, 2016.

[6] D. Harnik, B. Pinkas, A. Shulman-Peleg, Side Channels in Cloud Services: Deduplication in Cloud Storage, *IEEE Security & Privacy*, Vol. 8, No. 6, pp. 40-47, November-December, 2010.

[7] J. Paulo, J. Pereira, A Survey and Classification of Storage Deduplication Systems, *ACM Computing Surveys*, Vol. 47, No. 1, pp. 1-30, July, 2014.

[8] J. R. Douceur, A. Adya, W. J. Bolosky, P. Simon, M. Theimer, Reclaiming Space from Duplicate Files in a Serverless Distributed File System, *Proceedings 22nd International Conference on Distributed Computing Systems*, Vienna, Austria, 2002, pp. 617-624.

[9] M. W. Storer, K. M. Greenan, D. D. E. Long, E. L. Miller, Secure Data Deduplication, *Proceedings of the 2008 ACM Workshop On Storage Security And Survivability, StorageSS 2008*, Alexandria, VA, USA, 2008, pp. 1-10.

[10] A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, J. C. S. Lui, A Secure Cloud Backup System with Assured Deletion and Version Control, *2011 40th International Conference on Parallel Processing Workshops*, Taipei City, Taiwan, 2011, pp. 160-167.

[11] M. Bellare, S. Keelveedhi, Interactive Message-Locked Encryption and Secure Deduplication, *18th IACR International Conference on Practice and Theory in Public-Key Cryptography*, Gaithersburg, MD, USA, 2015, pp. 516-538.

[12] M. Bellare, S. Keelveedhi, T. Ristenpart, DupLESS: Server-aided Encryption for Deduplicated Storage, *22nd USENIX conference on Security Symposium*, Washington, D.C., USA, 2013, pp. 179-194.

[13] J. Liu, N. Asokan, B. Pinkas, Secure Deduplication of Encrypted Data without Additional Independent Servers, *22nd ACM SIGSAC Conference on Computer and Communications Security*, Denver, Colorado, USA, 2015, pp. 874-885.

[14] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, W. Lou, Secure and Efficient Cloud Data Deduplication With Randomized Tag, *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 3, pp. 532-543, March, 2017.

[15] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, G. Segev, Message-Locked Encryption for Lock-Dependent Messages, *Annual Cryptology Conference*, Santa Barbara, CA, USA, 2013, pp. 374-391.

[16] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang, M. M. Hassan, A. Alelaiwi, Secure Distributed Deduplication Systems with Improved Reliability, *IEEE Transactions on Computers*, Vol. 64, No. 12, pp. 3569-3579, December, 2015.

[17] Y. Zhou, D. Feng, W. Xia, M. Fu, F. Huang, Y. Zhang, C. Li, SecDep: A User-Aware Efficient Fine-Grained Secure Deduplication Scheme with Multi-Level Key Management, *31st International Conference on Massive Storage Systems and Technology (MSST2015)*, Santa Clara, CA, USA, 2015, pp. 1-14.

[18] R. D. Pietro, A. Sorniotti, Boosting Efficiency and Security in Proof of Ownership for Deduplication, *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, Seoul, Korea, 2012, pp. 81-82.

[19] J. Hur, D. Koo, Y. Shin, K. Kang, Secure Data Deduplication with Dynamic Ownership Management in Cloud Storage, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 28, No. 11, pp. 3113-3125, November, 2016.

[20] Z. Yan, W. Ding, X. Yu, H. Zhu, R. H. Deng, Deduplication on Encrypted Big Data in Cloud, *IEEE Transactions on Big Data*, Vol. 2, No. 2, pp. 138-150, June, 2016.

[21] X. Yang, R. Lu, K. R. Choo, F. Yin, X. Tang, Achieving Efficient and Privacy-Preserving Cross-Domain Big Data Deduplication in Cloud, *IEEE Transactions on Big Data*, pp. 1-1, June, 2017.

[22] G. Ateniese, K. Fu, M. Green, S. Hohenberger, Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage, *ACM Transactions on Information and System Security*, Vol. 9, No. 1, pp. 1-30, February, 2006.

Biographies

Yongan Guo, senior engineer of College of Communications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, China. His research interests include network virtualization, future Internet, and Internet of Things.

Chunlei Jiang, post graduate student of College of Communications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, China. His research interests include wireless communication and Internet of Things.