A Novel Data Deduplication Scheme for Encrypted Cloud Databases

Jung-Shian Li[1], I-Hsien Liu[1], Chao-Yuan Lee[1], Chu-Fen Li[2], Chuan-Gang Liu[3]

[1] Department of Electrical Engineering, Institute of Computer and Communication Engineering, National Cheng Kung University, Taiwan

[2] Department of Finance, National Formosa University, Taiwan

[3] Department of Applied Informatics and Multimedia, Chia Nan University of Pharmacy & Science, Taiwan

jsli@mail.ncku.edu.tw, ihliu@cans.ee.ncku.edu.tw, cylee@cans.ee.ncku.edu.tw, chufenli@gmail.com, chgliu@mail.cnu.edu.tw

Abstract

As the demand for data sharing and complex access-control policies continues to grow, traditional encryption mechanisms, which are generally established using a Public Key Infrastructure, face the problem of massive processing overheads and huge network bandwidth consumption. Attribute-Based Encryption (ABE) schemes have been proposed as a potential means of addressing these issues and have attracted great attention in recent years. Most previous studies on ABE focus on issues such as the revocation mechanism, multi-authority, the access structure design, and traceability. However, very few studies consider the storage efficiency problem and the present study proposes a novel data deduplication scheme based on ciphertext-policy ABE with convergent encryption and block-level data. The scheme can be deployed in third-party semi-trusted environments, and not only provides flexible, fine-grained access control over encrypted data, but also allows for the in-line elimination of redundancies in order to save cloud storage space. The experimental results show that the proposed scheme has an acceptable computational overhead and provides a promising solution for real-world data cloud storage and access scenarios.

Keywords: Cloud storage, Network security, Cloud security, Data deduplication scheme, Encrypted database

1 Introduction

With the dramatic rise of cloud computing, cloud service vendors are under increasing pressure to provide effective mechanisms for safeguarding data security. Google Cloud Storage has adopted AES-128 as the default encryption mechanism since 2013, while Dropbox employs AES-256 for a higher level of security. However, both encryption mechanisms are executed on the server side. In other words, the encryption keys are in the hands of the vendors, and thus the users just should fully trust the cloud service and believe that the vendors. To address this problem, various local encryption software systems have been proposed and, through integration with popular third-party cloud storage services, provide users with the ability to perform encryption for themselves. However, with the demand for data sharing increasing day-by-day, such systems, which provide only an "encrypted storage" capability, are insufficient to meet many users' requirements. A more sophisticated mechanism, Encrypted data sharing, is required, which is generally performed using Public Key Infrastructure (PKI) technologies [1], in which the sender must first obtain the receiver's public certificate prior to performing encryption. However, with the huge volume of private and confidential data now residing on the internet, such mechanisms face major challenges in controlling the key distribution process, managing the computational overhead, and satisfying the network bandwidth demand.

To tackle these problems, Boneh and Franklin [2] proposed a scheme referred to as Identity-Based Encryption (IBE) in 2003, in which the user identity was taken as part of the public key. Goyal et al., Sahai and Waters [3-4] proposed an Attribute-Based Encryption (ABE) scheme based on IBE with the following features: (1) the senders encrypt the message based only on some unique information and hence there is no need to consider the explicit identity or quantity of the receivers. As a result, the computational overhead is significantly reduced. (2) Only those users whose attributes satisfy the access policy can decrypt the message, and thus data confidentiality is ensured. (3) Collision attacks are prevented. (4) The access policy is based on simple AND and OR operations, and is therefore both expressive and flexible. Due to these advantages, ABE has been used in a broad range of applications nowadays, including encrypted audit

*Corresponding Author: Chuan-Gang Liu; E-mail: chgliu@mail.cnu.edu.tw
DOI: 10.3966/160792642020072104020

loggers [5], pay-per-view TV [6], and electronic healthcare records [7]. Two different variants of ABE have been proposed, namely Ciphertext-Policy ABE (CP-ABE) and Key-Policy ABE (KP-ABE). In CP-ABE, the access policy attached to the ciphertext is determined solely by the encryption party. By contrast, in KP-ABE, the data owner has no say over who can decrypt the data and must therefore place absolute trust in the key issuer. Many ABE schemes have developed in the past decade [8-10], but just few studies have considered the storage efficiency problem in ABE.

In fact, as the volume of digital data on the internet continues to grow, data deduplication techniques become essential to optimize the efficiency of mass storage systems. Data deduplication can be regarded as a special case of lossless data compression achieved by eliminating duplicate copies of repeating data. Data deduplication is widely used in many organizations for backup and disaster recovery purposes since over 90% of the data held by such organizations is duplicated in backup datasets [11-13]. However, data deduplication can also be used to improve the utilization of primary storage systems [14]. In fact, it has been estimated that, depending on the data distribution, data deduplication can reduce the occupied storage space by up to 75% compared to the original [15].

According to the principle of ciphertext indistinguishability [16], an adversary should not be able to distinguish pairs of ciphertext based on the message they encrypt. Accordingly, the present study employs an alternative encryption technique known as convergent encryption (CE) [17]. CE has a deterministic encryption characteristic, and therefore produces the same ciphertext for the same files. As a result, it facilitates secure redundancy removing, and therefore provides a feasible solution for optimizing the utilization efficiency of large-scale storage systems. The present study chooses the CP-ABE scheme proposed by Zhen Liu and Duncan S. Wong in 2015 [8] for implementation and analysis purposes. It supports traceability, revocation mechanism and large attribute universes, which are indispensable features for real-world systems. To minimize the CP-ABE storage consumption, this study proposes a new hybrid crypto architecture combined with a secure data deduplication technique. All files are encrypted on the client side and CP-ABE is utilized to achieve secure access control.

Compared to existing systems using CP-ABE in the encryption process, our proposed scheme has the following features:

(1) We propose a novel crypto architecture combined with a secure data deduplication technique.

(2) In our proposed crypto architecture, two encryption technologies is adopted, convergent encryption (CE) technique to encrypt the message, and CP-ABE to encrypt the public and private keys.

(3) With means of block-level in-line data deduplication, the repeating chunks will not be uploaded, our scheme can save the network bandwidth and speeds up the completion time.

(4) Sensitive data can be placed on semi-trusted servers under the assumption of an "honest-but-curious" model [18]. This feature is highly advantageous to large-scale enterprises since it allows both cloud services and data to be migrated to the public cloud, thereby reducing the costs of infrastructure management, hardware maintenance, and human resources, respectively.

The remaining of this paper is as follows. Section 2 introduces related work and then we explain our proposed scheme, a novel data deduplication scheme with CP-ABE in Section 3. Then, we show the implementation of our scheme and evaluate its performance in Section 4. Finally, we conclude this paper in Section 5.

2 Background and Related Work

In this Section, we introduce related works about this study. First, we introduce Ciphertext-Policy Attribute-Based Encryption including syntax of CP-ABE and practical CP-ABE. Then we mention Augmented Revocable CP-ABE (AugR-CP-ABE) in Section 2.2.

2.1 Ciphertext-Policy Attribute-Based Encryption

As shown in Figure 1, CP-ABE consists of four polynomial-time algorithms, namely Setup, Key Generation, Encryption, and Decryption. The functional definitions of each algorithm are provided in the following.

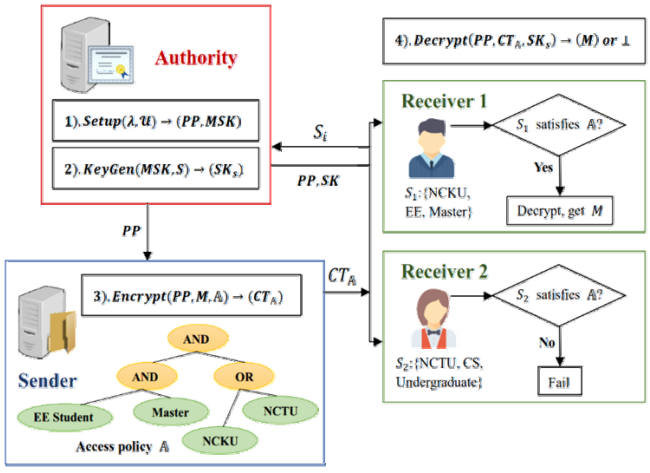


Figure 1. Illustrative example of CP-ABE

Setup.

$$Setup(\lambda, \mathcal{U}) \rightarrow (PP, MSK)$$

The Setup algorithm takes a security parameter λ and attribute universe $\mathcal{U}$ as the input and produces a public parameters PP and master secret key MSK as its output.

Key generation.

$$KeyGen(MSK, S) \rightarrow (SK_s)$$

The Key Generation algorithm takes the master secret key MSK and a set of attributes *S* describing the key as its input and produces a secret key corresponding to *S* as the output.

Encryption.

$$Encrypt(PP, M, \mathbb{A}) \rightarrow (CT_{\mathbb{A}}).$$

The Encryption algorithm takes the public parameter PP, a message *M*, and an access structure $\mathbb{A}$ over the universe of attributes as its input. It then encrypts M and produces the ciphertext $CT_{\mathbb{A}}$, which includes access policy $\mathbb{A}$ within it.

Decryption.

$$Decrypt(PP, CT_{\mathbb{A}}, SK_s) \rightarrow (M) \text{ or } \perp.$$

Finally, the Decryption algorithm takes the public parameter PP, ciphertext $CT_{\mathbb{A}}$, and user's secret key SK_s as inputs. The secret key contains the appropriate attributes which satisfy the access structure $\mathbb{A}$. Then it decrypts the ciphertext and returns the message *M*.

In order to implement CP-ABE, Revocation and Large Attribute Universe process should be explained.

2.1.1 Revocation

Most cryptosystems, usually public key infrastructures (PKI), contain a certificate revocation list (CRL), i.e., a list of digital certificates which have been revoked by the Certificate Authority (CA) before their scheduled expiration date. To implement CP-ABE in practical situations, a similar revocation mechanism is required to revoke compromised keys. In the majority of early studies [19-20], the revocation process was executed by an authority. This technique, generally referred to as indirect revocation, requires the authority to update a blacklist and to broadcast the information to the non-revoked system users. These users can then update their secret keys accordingly and use their new keys to perform data decryption. Ostrovsky et al. [21] employed the user identity as one of the encryption attributes such that the ciphertext was intrinsically combined with the identity. Attrapadung et al. [22] proposed a direct revocation mechanism which allowed the revocation list to be enumerated directly during the encryption process. Therefore, even if the attributes satisfy the access policy, the ciphertext cannot be decrypted if the receiver is on the revocation list. Direct revocation mechanisms have several important advantages over indirect mechanisms for CP-ABE applications in that they support revocation on demand and allow the authority to update the revocation list at will rather than waiting for a particular update time.

2.1.2 Large Attribute Universe

In most CP-ABE schemes, the size of the attribute universe is polynomially bounded by the security parameter, λ. Moreover, the attributes need to be enumerated in the initialization phase and cannot be added dynamically after setup. Finally, the size of the public parameter increases linearly with the number of attributes. As a result, CP-ABE is non-scalable and difficult to deploy. Hence, supporting large universes is highly desirable since, in this way, any string can be regarded as an attribute and the attributes need not be explicitly pre-defined during setup. Large universe CP-ABE schemes have been discussed previously [23] and the first truly large universe CP-ABE method, with no restrictions on the access policies or attributes associated with the keys, was proposed by Rouselakis et al. in [24].

2.2 Augmented Revocable CP-ABE (AugR-CP-ABE)

In the past decade, researchers have proposed many variants of CP-ABE, including Revocable CP-ABE [22], Blackbox Traceable CP-ABE [25], Large Universe CP-ABE [24], and Augmented Revocable CP-ABE (AugR-CP-ABE) [8]. Table 1 compares the storage efficiencies and capabilities of the four schemes.

Table 1. Efficiency comparison of different CP-ABE schemes

Scheme	Public Parameter Size	Private Key Size	Ciphertext Size
[22]	$7+\lvert S\rvert_{\max}+l_{\lvert S\rvert_{\max}}$	$4+\lvert S\rvert$	$2+l+2\lvert R\rvert$
[25]	$3+4\sqrt{N}+\mathcal{U}$	$4+\lvert S\rvert$	$17\sqrt{N}+2l$
[24]	6	$2+2\lvert S\rvert$	$2+3l$
[8]	$5+5\sqrt{N}$	$2+\sqrt{N}+2\lvert S\rvert$	$16\sqrt{N}+3l$
Scheme	Revocation	Traceability	Large Universe
[22]	V	X	--
[25]	X	V	X
[24]	X	X	V
[8]	V	V	V

Referring to Table 1, *N* is the total number of users in the system, $|\mathcal{U}|$ is the size of the attribute universe, *l* is the number of LSSS matrix for an access policy, |*S*| is the size of the attribute set for the secret key, and $|R|$ is the number of revoked users in the revocation list. Note that the scheme in [22] is not truly large universe since the maximum size of the attribute set $|S|_{\max}$ must be fixed in the setup phase. In the present case, $l_{|S_{\max}|}$ is set as the maximum number of rows in the LSSS matrix. The overhead of AugR-CP-ABE is just $O(\sqrt{N})$. Furthermore, it is the first scheme to

support all of the desirable features. However, the storage overhead of AugR-CP-ABE may be still present problems in large-scale systems with very large numbers of users and attributes. Consequently, as described in the following section, this study proposes a new secure data deduplication technique which renders CP-ABE more feasible for cloud storage services and similar large-scale systems.

2.3 Cloud Data Deduplication with CP-ABE

Recently, cloud service provider tries to provide secure cloud storage service and access control to prevent illegitimate user from accessing cloud data. Hence, CP-ABE is usually widely used to ensure secure cloud storage service. However, as described above, data deduplication is also getting critical for efficient cloud data storage service. Hence, many researches [11-15, 27, 29] take much effort to develop deduplication scheme for cloud storage system. But, the cloud storage system with ABE usually does not support secure deduplication [31]. There are still few research works focusing on the topic, improving the efficiency of clould data deduplication with CP-ABE. Recently, in 2016, the authors in [30] focus on ciphertexts duplication based on ABE and they claim that their paper is the first to take effort on this topic. In their paper, they design a secure ciphertext deduplication scheme based on a classical CP-ABE scheme, which eliminates the duplicated secrets and adding additional randomness to some certain ciphertext. The idea of their ciphertext deduplication scheme is to modify the construction with a recursive algorithm. In 2017, an outstanding paper [31] also proposed an attribute-based storage system with secure deduplication, which utilizes a hybrid cloud setting in which a private cloud and a public cloud do their corresponding jobs, duplication detection and the storage management, respectively. The goal of those studies is to develop a secure and efficient cloud data duplication system.

In this paper, we also develop a novel deduplication scheme for cloud encrypted data, which ensure a secure and efficient cloud storage service.

3 A Novel Data Deduplication Scheme with CP-ABE

In this Section, we firstly introduce secure data deduplication and show the adversary model in this paper. We also explain the participants in our proposed scheme. Then, we show our proposed scheme and explain the all process of it in details.

3.1 Secure Data Deduplication

Meyer and Bolosky [26] collected 162 terabytes of data from 857 desktop computers over a span of 4 weeks and found that the storage consumption could be reduced to as little as 32% of the original requirement. Accordingly, the present study proposes a novel scheme based on CP-ABE and a block-level data design to perform data deduplication in cloud storage systems, thereby significantly improving their space utilization efficiency. That is, the data on the client device are chunked by chunking algorithm and convergently encrypted prior to transmission to the cloud. In our proposed framework, the user files are encrypted using convergent encryption rather than CP-ABE, and CP-ABE is used only to encrypt the convergent keys in order to protect them from adversaries. We describe convergent encryption and chunking algorithm as below.

3.1.1 Convergent Encryption

Convergent encryption (CE) uses the hash value of plaintexts as the secret keys. That is, given a message M, hash function $h()$, and symmetric encryption algorithm $E()$, the ciphertext C is obtained as

$$C = E(key, message) = E(h(M), M) \quad \textbf{(1)}$$

This approach guarantees a convergence property. That is, any client encrypting the same plaintext generates the same secret keys and thus produces the same ciphertext. As a result, CE provides a feasible solution for data data deduplication [27] and is already employed by many commercial cloud storage providers, including Bitcasa and GNUnet. In the present study, CE is similarly employed to ensure the privacy of the user data when implemented on semi-honest cloud storage servers.

3.1.2 Chunking Algorithms

The present framework adopts the Rabin-Karp rolling hash algorithm to implement variable size chunking. The algorithm requires four parameters to be predefined, namely the window size W, the shifting length L, an integer divisor D, and an integer remainder R, where $0 \le R < D$. The sliding window W shifts L bytes at a time from the beginning to the end. In every shift, the Rabin-Karp Rolling Hash algorithm calculates a hash value $h = has(W)$ and checks if $(h \bmod D) = R$. Referring to the illustrative example shown in Figure 2, the integer remainder R has a specific pattern, And only when the data in the sliding window matches this pattern, the position is set as the breakpoint for the chunk boundary. Notably, this sliding window approach ensures that modifications to the file affect only the current chunk.

3.2 Adversaries Model

The proposed data deduplication strategy is determined under the following premises:

(1) Data on the cloud must be encrypted;

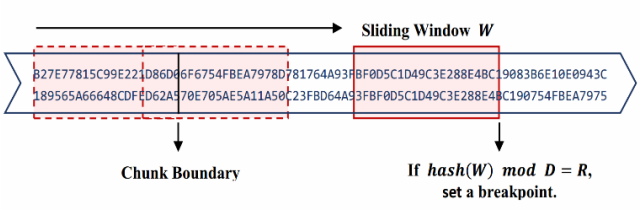


Figure 2. Variable size chunking using Rabin-Karp rolling hash algorithm

(2) Encryption and decryption should both be performed on the client side.

(3) The third-party storage provider is an honest-but-curious server [28].

In other words, the server does not tamper with the data, and honestly executes the proposed scheme in accordance with the prescribed protocol, but may try to learn information from the stored files. Based on these premises and assumptions, chunking and encryption should both be performed on the client side such that the data are not transmitted in plaintext form and are protected against internal adversaries on the data storage provider.

3.3 System Participants

The system considered in the present study consists of five primary participants, namely the Data Owner and Data User on the client side, and the Authority, Metadata Manager and Cloud Storage Provider on the server side. The five participants are defined as follows.

- Data Owner: The individual who actually possesses the data.
- Data User: The person or entity that wishes to access the data shared by the data owner.
- Authority: This party is responsible for generating the public parameter, master secret key and user's secret key (private key). According to the set of attributes from different users, the authority issues a corresponding secret key to them. In this study, the authority server must be fully trusted.

3.3.1 Cloud Storage Provider

As described above, the data storage provider is assumed to be a semi-trusted server with an "honest-but-curious" model. Due to the deployment of the data deduplication technique, the data chunks stored on the storage node are de-duplicated on the client side. Moreover, with the purpose of data confidentiality, all of the data chunks are encrypted by CE, and hence it is impossible to deduce the content from the data chunk itself.

3.3.2 Metadata Manager

The party is responsible for maintaining all the information the users require to reconstruct the original files from the encrypted data chunks. The CE keys in the present framework are additionally encrypted by AugR-CP-ABE. Hence, the metadata manager can also be regarded as a semi-trusted server, and if necessary, placed with the data storage provider in the same cluster.

3.4 Proposed Scheme

This section describes the proposed framework and explains how it simultaneously supports both securedata deduplication and fine-grained access control in encrypted cloud databases. Figure 3 presents a schematic overview of the overall system architecture. We define the used parameters as follows:

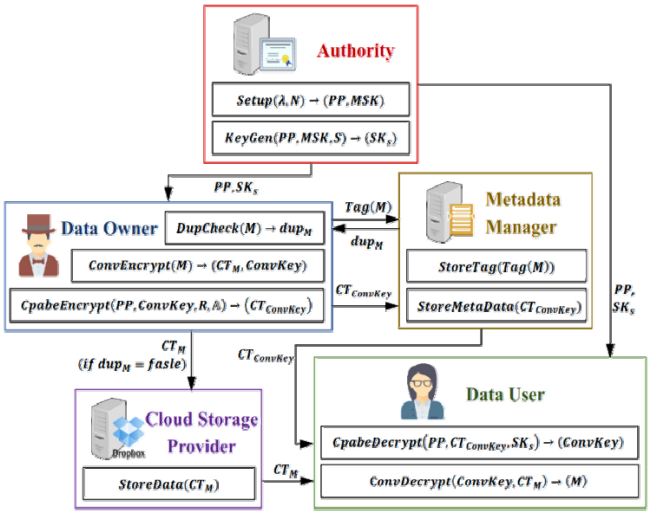


Figure 3. System architecture (file-level data deduplication)

λ : Security Parameter — N: Total System User
PP : Public Parameter — **MSK:** Master Secret Key
M : Message — S: Set of Attribute
R : Revocation List — $\mathbb{A}$: Access policy
SK_s : Secret Key — CT_M : Encrypted Message
$ConvKey$: Convergent Encryption Key
$CT_{convKey}$: Encrypted ConvKey

Note that in order to simplify the presentation, the data chunking and file reconstruction processes are deliberately omitted from the figure. As shown in the figure (and described above), the system consists of five participants, namely the Data Owner, the Data User, the Authority, the Metadata Manager, and the third-party Cloud Storage Provider. Moreover, the proposed framework consists of four phases, namely (1) Setup and Key Generation; (2) Chunking and Duplicate Checking; (3) Encryption and Upload; and (4) Download and Decryption. The details of each phase are described in the following.

3.4.1 Setup and Key Generation Phase

In the system initialization phase, the security parameter and number of users are set in accordance with the security requirements and scale of deployment,

respectively. In inital Setup Phase, The authority then computes the public parameter and master secret key, and broadcasts the public parameter to every user (Data user and owner) in the system. In the Key Generation phase, depending on different attributes, the authority generates secret keys for the authorized users using the master secret key. Above procedures can be observed in Figure 3 and Figure 4.

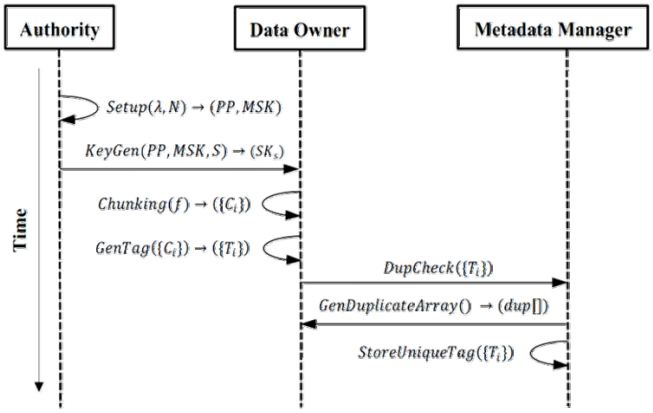


Figure 4. Key generation and duplicate checking

3.4.2 Chunking and Duplicate Checking Phase

In the chunking stage, the user file is broken into blocks using the Rabin-based variable-size chunking algorithm and duplicates are detected via an inspection of the records on the metadata manager (see Figure 4). Since hashing is CPU intensive, a weak hashing algorithm is first used to identify potentially duplicate data. The suspected data are then rehashed using a much stronger hashing algorithm to verify whether or not they truly are duplicate.

3.4.3 Encryption and Upload Phase

After chunking and duplicate checking, the remaining chunks are encrypted by CE and are uploaded to the cloud storage provider. Notably, in contrast to conventional secure data deduplication systems, the CE keys are not passed to the metadata manager directly in plaintext form. Rather, the CE keys for the convergently encrypted file are archived and the package is then encrypted by AugR-CP-ABE under a given access policy (see Figure 5). In other words, the system adopts a hybrid encryption technique, which combines symmetric encryption (CE) with asymmetric encryption (CP-ABE).

3.4.4 Download and Decryption Phase

In the Download and Decryption phase, the authorized data user first requests the encrypted chunks and encrypted package of CE keys from the cloud storage provider and metadata manager, respectively (see Figure 6). If the user has the appropriate secret key satisfying the access policy embedded within the

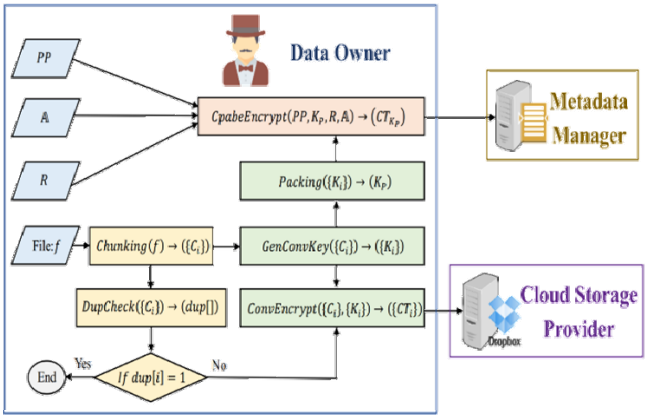


Figure 5. Encryption and upload phase

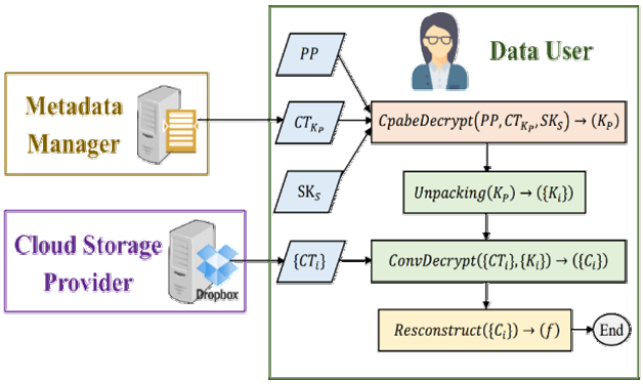


Figure 6. Download and decryption phase

ciphertext, he or she is able to decrypt the encrypted CE keys. Having obtained these keys, the data user can then decrypt all of the data chunks and reconstruct the original file.

In summary, the proposed framework proposes a client-side data deduplication technique. That is, the data on the client device are chunked and convergently encrypted prior to transmission to the cloud. Under these conditions, in-line data deduplication provides a more intuitive approach than post-process data deduplication since, given the use of client-side data deduplication, the repeating data chunks will be found in the first place and can therefore be directly eliminated during transmission.

The proposed strategy has the further advantage of providing data confidentiality over the entire process. In other words, the semi-trusted servers have no opportunity to ever manipulate the data in plaintext form. Finally, the proposed method consumes less storage space than existing post-process data deduplication schemes.

4 Implementation and Evaluation

An experimental testbed was constructed on a Linux Ubuntu system with a 1.7 GHz Intel Core 4 processor, 6 GB of RAM, and a 5400 RPM Western Digital 500

GB drive. All of the implementations were conducted using Java. To improve the computation performance, we use the PBC Native Extension and GNU Multiple Precision Arithmetic Library (GMP), which were both written in C.

The metadata of the files and hash values of the data chunks were stored in Redis, which is a lightweight, high-performance, in-memory, key value database. The database provides a number of data structures, including strings, hashes, lists, sets, sorted sets, bitmaps, and HyperLogLogs. Moreover, it supports permanent storage, which can synchronize the data in memory to the disk for persistence. These reasons make Redis applicable to our metadata manager. The cloud storage provider was implemented using Dropbox. Then, we evaluate our scheme in term of four topics, Duplicate Checking Evaluation, Chunking Algorithm Comparison, Convergent Encryption Performance and AugR-CP-ABE Encryption Performance.

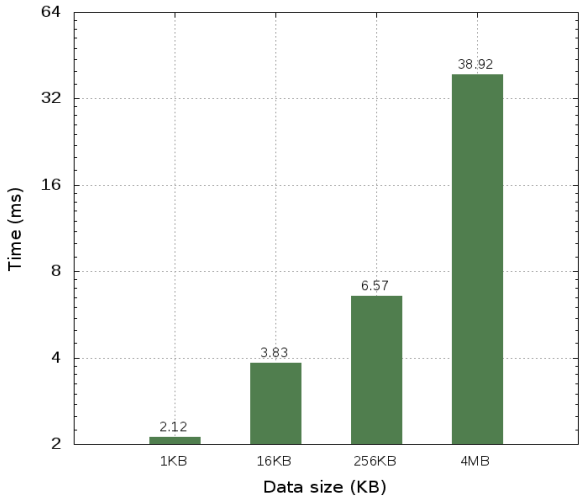
4.1 Duplicate Checking Evaluation

We choose "client-side"data deduplication scheme, which the analysis of detecting duplicates occurs in client device. Figure 7(a) shows the execution time of the duplicate checking process for input files of different sizes. Figure 7(b) shows the de-duplication time for a 5 MB file processed using different chunking sizes. It is seen that even though it is faster to process smaller chunks, the resulting increase in the number of duplicate checks results in a significant performance degradation. Nonetheless, a smaller chunk size is advantageous in improving the storage efficiency. In practice, there is no absolute standard for the option of chunk size. Data deduplication process must associate performance overhead to a certain extent, whereas smaller chunk size enhances the space efficiency. We need to tradeoff between performance and storage efficiency depend on the demand. In general, with a greater amount of data stored by the cloud storage provider, a file has more chance to be de-duplicated. Consequently, significant savings in the transmission time can be made. For example, Zhang et al. [29] presented an image management system which not only reduced the virtual machine image storage by 80%, but also reduced the transmission time by 30%.

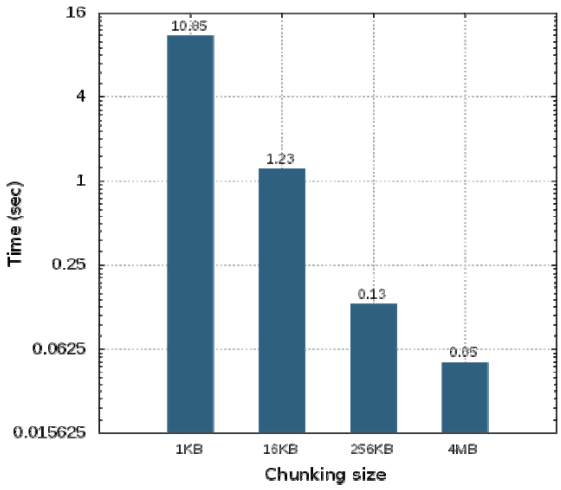
4.2 Chunking Algorithm Comparison

In this paper, file-based chunking, fixed-size chunking and variable-size chunking using Rabin-Karp Rolling Hash algorithm have been implemented. In performing the comparison, the rolling hash was implemented using the following simple polynomial function:

$$H = c_1 a^{k-1} + c_2 a^{k-2} + c_3 a^{k-3} + c_4 a^{k-4} + \cdots + c_k a^0 \quad \textbf{(2)}$$



(a) Duplicate checking time vs. data size



(b) De-duplication time for 5M file given different chunking sizes

Figure 7. Evaluation of duplicate checking performance (using SHA-512)

where a is a prime constant and $c_1, c_2, c_3, \ldots, c_k$ are the input characters depending on the window size. The specific pattern was defined using a bit mask N. Finally, the chunking breakpoint was specified as when the hash value satisfied.

$$H \,\&\, ((1 \ll N_{bitmask}) - 1) = 0 . \quad \textbf{(3)}$$

The chunking-based data deduplication rate was evaluated using 740 PDF files with a total memory size of around 1 GB. The fixed-size chunking algorithm was implemented using SHA-512 as fingerprinting to check for repeating data. In addition, the window size in the Rabin Karp chunking algorithm was set as 1025 bytes.

The performance results obtained using the two algorithms are shown in Table 2. As expected, the data deduplication rate under the Rabin-Karp algorithm is better than that under the fixed-size chunking

algorithm. However, since the Rabin-Karp algorithm re-computes the boundary position each time using a rolling hash, it is much slower than the fixed-size chunking algorithm.

Table 2. Rabin-hash chunking vs. fixed-size chunking

	Fixed-size 8K	*Rabin-Karp 8K*
Average block size (byte)	8192.0	11938.7
Total data blocks generated	125775	85535
Deduplication rate (%)	1.96	4.28
Execution speed (KB/ms)	67.77	39.46

4.3 Convergent Encryption Performance

The CE procedure used in the proposed framework to support privacy-preserving data deduplication was implemented as follows:

1. Generate a 256-bit hash key from the plaintext using SHA-256.

2. Use the hash key (CE key) to encrypt the plaintext using AES-256 with the ECB (Electronic Codebook) block cipher mode. (Note that the CBC (Cipher Block Chaining) mode can also be used. However, in this case, the initialization vector must be a constant in order to preserve the deterministic property.)

Figure 8 shows the execution time of the CE process for input data files with various sizes. For the average data chunk size in practical systems (i.e., 8~32 KB) the execution time is around 1.3 ms for a 10 MB file.

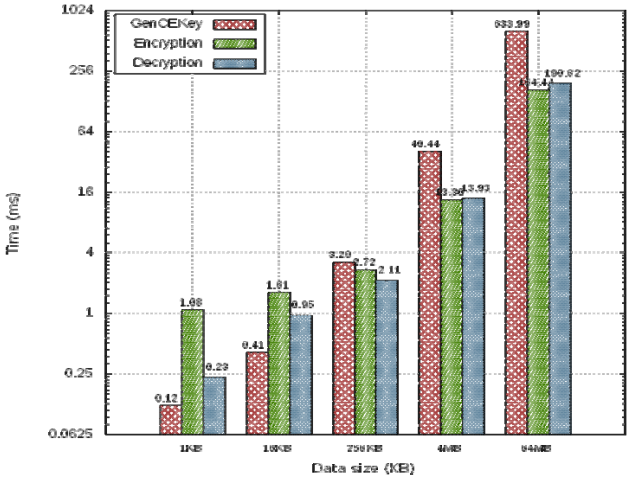


Figure 8. Convergent encryption execution time

4.4 AugR-CP-ABE Encryption Performance

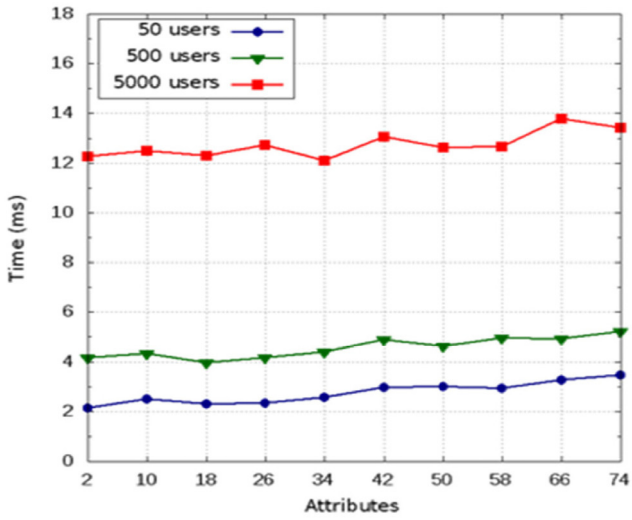
After the CE process, the CE keys were archived to a package. To protect the package from internal or external adversaries, it was additionally encrypted by AugR-CP-ABE. With the features of CP-ABE, any authorized data user who does not have the appropriate secret key cannot decrypt the encrypted package. The performance of the encryption and subsequent decryption procedures was evaluated as a function of the number of attributes and number of users in the system. The access policy function included converting the Boolean formula, constructing the access tree, and parsing the tree if the attributes satisfied the policy. Figure 9(a) shows the execution time for parsing the access policy for different numbers of attributes and users. In addition, Figure 9(b) shows the execution time for the AugR-CP-ABE encryption process. Observing the two figures, it is seen that the parsing process accounts for almost all of the execution time. Furthermore, for 5000 users and more than 70 attributes, the time to parse the access policy is around 3.5 s. The parsing time can be reduced by fully implementing the LSSS matrix. Alternatively, the two processes (parsing and encryption) can truly be separated since, in the majority of cases, there is no need to re-parse the access policy, which may be already determined. Furthermore, in some cases, the data owner may adopt an identical access policy for most files.

Figure 9(c) shows the execution time of the AugR-CP-ABE decryption process for different numbers of attributes and system users. Irrespective of the number of attributes or users, the execution time is less than 500 ms, and is therefore acceptable for practical systems. Furthermore, Figure 10 confirms the space efficiency of the AugR-CP-ABE scheme. Notably, the overhead of AugR-CP-ABE is only $O(\sqrt{N})$.

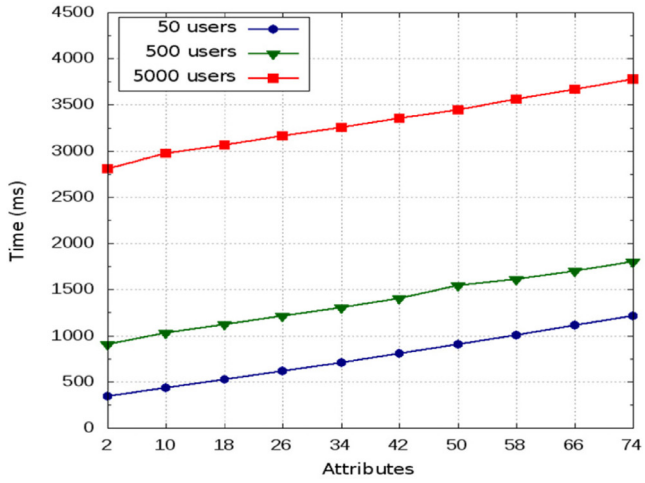
Table 3 shows the effect of the total number of system users on the space efficiency of the ciphertext, public parameter, master secret key and secret key. The results clearly show that why the trend of ciphertext and public parameter are much faster than the others.

5 Conclusions

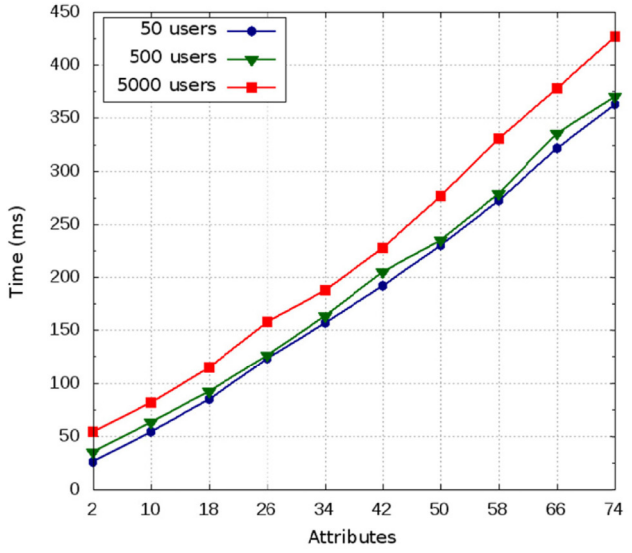
This paper has proposed a novel scheme combining AugR-CP-ABE and a secure data deduplication technique for supporting both privacy-preserving data deduplication and fine-grained data sharing in encrypted cloud databases. In the proposed framework, data owners can predefine a very expressive access policy by themselves for each file they wish to upload and share. Having done so, only authorized users possessing a secret key which satisfies the access policy can successfully decrypt the data. Furthermore, due to the broadcasting property of CP-ABE, data owners need only specify the group by an attributes collection once in the encryption stage. Collectively, these features render the framework favorable for scenarios in which employees from different departments or companies cooperate on confidential projects. The experimental results confirm the feasibility of the proposed scheme for practical cloud environments.



(a) Execution time for parsing access policy

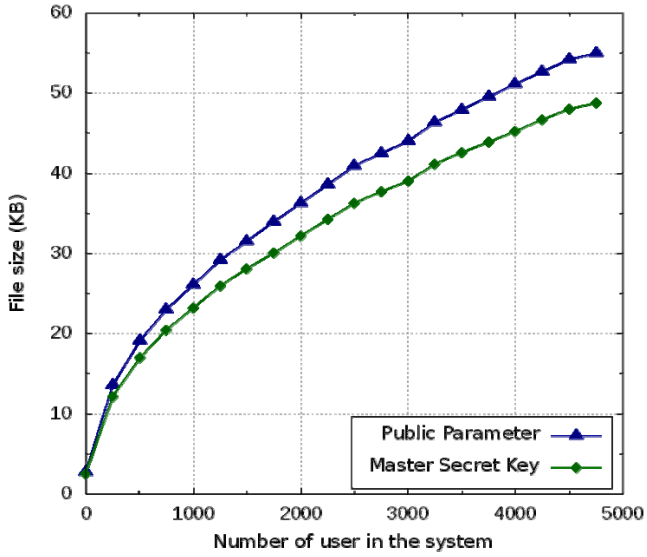


(b) Execution time for AugR-CP-ABE encryption

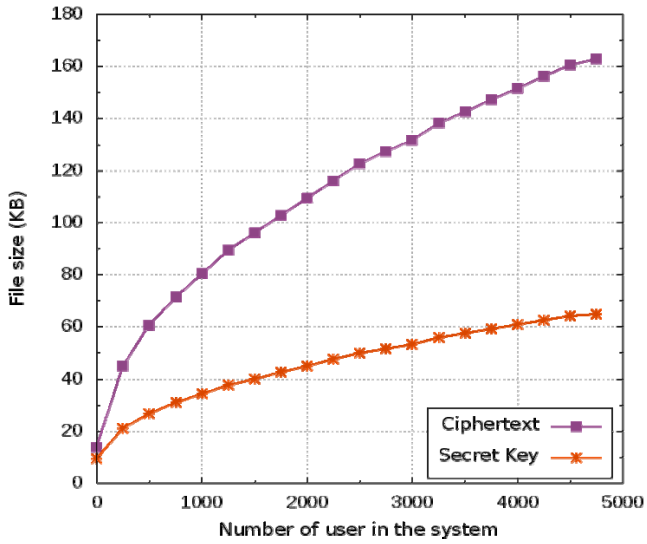


(c) Execution time for AugR-CP-ABE decryption

Figure 9. Performance of AugR-CP-ABE



(a) Storage overheads for public parameter and master secret key



(b) Storage overhead for ciphertext and secret key

Figure 10. Storage overheads for AugR-CP-ABE

Table 3. Space efficiency of AugR-CP-ABE

Parameter	Space Efficiency
Ciphertext	$O(16\sqrt{n})$
Public Parameter	$O(5\sqrt{n})$
Master Secret Key	$O(3\sqrt{n})$
Secret Key	$O(\sqrt{n})$

Acknowledgements

The authors gratefully acknowledge the support of the Ministry of Science and Technology of Taiwan under Grant MOST 106-2221-E-041-003 and MOST 106-3114-E-006-003.

References

[1] A. Harrington, C. Jensen, Cryptographic Access Control in a Distributed File System, *8th ACM Symposium on Access Control Models and Technologies*, Como, Italy, 2003, pp. 158-165.

[2] D. Boneh, M. Franklin, Identity-Based Encryption from the Weil Pairing, *21st Annual International Cryptology Conference on Advances in Cryptology*, Santa Barbara, California, 2001, pp.213-229.

[3] V. Goyal, O. Pandey, A, Sahai, B. Waters, Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data, *13th ACM conference on Computer and Communications Security*, Alexandria, VA, USA, 2006, pp. 89-98.

[4] A. Sahai, B. Waters, Fuzzy Identity-Based Encryption, *24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Aarhus, Denmark, 2005, pp. 457-473.

[5] B. P. Gopularam, S. Dara, N. Niranjan, Experiments in Encrypted and Searchable Network Audit Logs, *International Conference on Emerging Information Technology and Engineering Solutions*, Pune, India, 2015, pp.18-22.

[6] D. Lubicz, T. Sirvent, Attribute-Based Broadcast Encryption Scheme Made Efficient, *1st International Conference on Cryptology in Africa*, Casablanca, Morocco, 2008, pp. 325-342.

[7] J. A. Akinyele, M. W. Pagano, M. D. Green, C. U. Lehmann, Z. N. J. Peterson, A. D. Rubin, Securing Electronic Medical Records Using Attribute-Based Encryption on Mobile Devices, *1st ACM workshop on Security and privacy in smartphones and mobile devices*, Chicago, IL, USA, 2011, pp. 75-86.

[8] Z. Liu, D. S. Wong, Practical Attribute-Based Encryption: Traitor Tracing, Revocation, and Large Universe, *The Computer Journal*, Vol. 59, No. 7, pp. 983-1004, July, 2016.

[9] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-Policy Attribute-Based Encryption, *2007 IEEE Symposium on Security and Privacy*, Berkeley, CA, USA, 2007, pp. 321-334

[10] M. Chase, Multi-Authority Attribute-Based Encryption, *4th Conference on Theory of Cryptography*, Amsterdam, The Netherlands, 2007, pp. 515-534.

[11] Y. Fu, H. Jiang, N. Xiao, L. Tian, F. Liu, L. Xu, Application-Aware Local-Global Source Deduplication for Cloud Backup Services of Personal Storage, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 25, No. 5, pp. 1155-1165, May, 2014.

[12] X. Xu, Q. Tu, Data Deduplication Mechanism for Cloud Storage Systems, *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Xi'an, China, 2015, pp. 286-294.

[13] H. Biggar, Experiencing Data De-Deduplication: Improving Efficiency and Reducing Capacity Requirements, *White Paper: The Enterprise Strategy Group*, February, 2007.

[14] B. Mao, H. Jiang, S. Wu, L. Tian, Leveraging Data Deduplication to Improve the Performance of Primary Storage Systems in the Cloud, *IEEE Transactions on Computers*, Vol. 65, No. 6, pp. 1775-1788, January, 2016.

[15] Y. Fu, H. Jiang, N. Xiao, A Scalable Inline Cluster Deduplication Framework for Big Data Protection, *13th International Middleware Conference*, Montreal, QC, Canada, 2012, pp. 354-373.

[16] S. Goldwasser, S. Micali, Probabilistic Encryption, *Journal of Computer and System Sciences*, Vol. 28, No. 2, pp. 270-299, April, 1984.

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

[18] K. G. Figueroa, S. Pancho-Festin, An Access Control Framework for Semi-trusted Storage Using Attribute-Based Encryption with Short Ciphertext and Mediated Revocation, *International Symposium on Computing and Networking*, Shizuoka, Japan, 2014, pp. 507-513.

[19] Y. Han, D. Jiang, X. Yang, The Revocable Attribute based Encryption Scheme for Social Networks, *International Symposium on Security and Privacy in Social Networks and Big Data (SocialSec)*, Hangzhou, China, 2015, pp. 44-51.

[20] A. Boldyreva, V. Goyal, V. Kumar, Identity-based Encryption with Efficient Revocation, *15th ACM Conference on Computer and Communications Security,* Alexandria, VA, USA, 2008, pp. 417-426.

[21] R. Ostrovsky, A. Sahai, B. Waters, Attribute-Based Encryption with Non-Monotonic Access Structures, *14th ACM Conference on Computer and Communications Security*, Alexandria, VA, USA, 2007, pp. 195-203.

[22] N. Attrapadung, H. Imai, Conjunctive Broadcast and Attribute-Based Encryption, *3rd International Conference Palo Alto on Pairing-Based Cryptography*, Palo Alto, CA, USA, 2009, pp. 248-265.

[23] B. Waters, Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, *14th International Conference on Practice and Theory in Public Key Cryptography*, Taormina, Italy, 2011, pp. 53-70.

[24] Y. Rouselakis, B. Waters, Practical Constructions and New Proof Methods for Large Universe Attribute-based Encryption, *20th ACM Conference on Computer and Communications Security*, Berlin, Germany, 2013, pp. 463-474.

[25] Z. Liu, Z. Cao, D. S. Wong, Traceable CP-ABE: How to Trace Decryption Devices Found in the Wild, *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 1, pp. 55-68, January, 2015.

[26] D. T. Meyer, W. J. Bolosky, A study of Practical Deduplication, *9th USENIX Conference on File and Storage Technologies*, San Jose, CA, USA, 2011, pp. 1-13.

[27] C. Wang, Z. G. Qin, J. Peng, J. Wang, A Novel Encryption Scheme for Data Deduplication System*, International Conference on Communications, Circuits and Systems*, Chengdu, China, 2010, pp. 265-269.

[28] Q. Chai, G. Gong, Verifiable Symmetric Searchable

Encryption For Semi-honest-but-curious Cloud Servers, *IEEE International Conference on Communications*, Ottawa, ON, Canada, 2012, pp. 917-922.

[29] J. Zhang, S. Han, J. Wan, B. Zhu, L. Zhou, Y. Ren, W. Zhang, IM-Dedup: An Image Management System Based on Deduplication Applied in DWSNs, *International Journal of Distributed Sensor Networks*, Vol. 9, No.7, pp.1-11, July, 2013.

[30] H. Tang, Y. Cui, C. Guan, J. Wu, J. Weng, K. Ren, Enabling Ciphertext Deduplication for Secure Cloud Storage and Access Control, *11th ACM on Asia Conference on Computer and Communications Security*, Xi'an, China, 2016, pp. 59-70.

[31] H. Cui, R. H. Deng, Y. Li, G. Wu, Attribute-Based Storage Supporting Secure Deduplication of Encrypted Data in Cloud, *IEEE Transactions on Big Data*, Vol. 5, No 3, pp. 330-342, September, 2019.

Biographies

Jung-Shian Li is a full professor in the department of electrical engineering, National Cheng Kung University, Taiwan. He obtained his PhD in 1999 in computer science from the Technical University of Berlin, Germany. His research interests include network protocol design, security, and network management. He is the director of TWISC@NCKU.

I-Hsien Liu is a researcher fellow in the TWISC@NCKU and department of electrical engineering, National Cheng Kung University, Taiwan. He obtained his Ph.D. in 2015 in computer and communication engineering from the National Cheng Kung University. He interests are Cloud security, Wireless Network, and Reliable Transmission in Mobile networks.

Chao-Yuan Lee comes from Taipei, Taiwan. He received the B.S. and M.S. degree in Department of Engineering Sciene and Institute of Computer and Communication Engineering repectively from Cheng-Kung University in 2014 and 2016. His research interests involve cryptography and cloud storage system.

Chu-Fen Li is an Associate Professor in the Department of Finance at the National Formosa University, Taiwan. She received her Ph.D. in information management, finance and banking from the Europa-Universität Viadrina Frankfurt, Germany. Her current research interests include intelligence finance, e-commerce security, financial technology, IoT security management.

Chuan-Gang Liu is an Associate professor in the department of Applied informatics and Multimedia, Chia Nan university of Pharmacy and Science. He graduated from the National Cheng Kung University with MS and Ph.D. degrees in electrical engineering. His research interests include wireless networks, network security, and performance analysis.