A novel role symmetric encryption algorithm for authorized deduplication in cloud

Yuanyuan Zhang
Faculty of Software, Fujian Normal University
Fuzhou, China
zyy837603010@163.com

Jinbo Xiong∗
Fujian Engineering Research Center of Public Service Big Data
Mining and Application
Fuzhou, China
jinbo810@163.com

Jun Ren
Faculty of Software, Fujian Normal University
Fuzhou, China
renjun0324@163.com

Lili Wang
Fujian Engineering Research Center of Public Service Big Data
Mining and Application
Fuzhou, China
wanglili@163.com

Mingwei Lin
Fujian Engineering Research Center of Public Service Big Data
Mining and Application
Fuzhou, China
linmwcs@163.com

ABSTRACT
The explosive growth of multimedia data promotes us to enter the era of big data. To improve the storage efficiency and reduce the management expenditure of these massive data, deduplication is a promising technology to meet these requirements. However, it arises serious privacy concerns and poses new security challenges, such as privacy leakage and unauthorized access. To tackle these problems, in this paper, we propose a novel role symmetric encryption (RSE) algorithm and construct an authorized deduplication scheme (RSEDup) based on the role symmetric encryption to achieve the authorized deduplication in cloud. The RSEDup scheme is the first solution to prevent privacy leakage and achieve the authorized deduplication effectively. Performance evaluation shows the efficiency of the proposed scheme.

KEYWORDS
Authorized deduplication, Unauthorized access, Privacy leakage, Role symmetric encryption

∗Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MOBIMEDIA 2017, July 13-14, Chongqing, People’s Republic of China
Copyright © 2017 EAI 978-1-63190-156-0

1 INTRODUCTION
The explosive growth of global multimedia data promotes the technology of cloud computing gradually increasing, more and more enterprises and individuals utilize cloud platform for data storage and management to achieve the resources sharing. And the gradually evolution of multimedia data processing pattern brings a challenge for these cloud platforms, where a traditional single type develops into heterogeneous multimedia big data [1]. Statistics shows that the global data volume has been reached 8.61 ZB by the end of 2015, and expects to 2020, that will unexpectedly reach 44 ZB [2]. Furthermore, the proportion of multimedia data in heterogeneous networks and wireless sensor networks [3], such as, image and video data has exceeded 90%. Therefore, how to improve the storage efficiency and reduce the management expenditure in heterogeneous networks and wireless sensor networks is a critical challenge for the cloud service providers [4].

Data compression is the direct technology to save storage space and bandwidth overheads [5]. However, different individuals may use the different data compression technologies, the same file will generate multiple replicas in the cloud server. To solve the above issue, data deduplication is proposed as a promising technology to...
meet the requirement, which is aimed at duplicating the replicas for the data sets. For a plaintext, the cloud server adopts random sampling or extracting hash value to check with source data, if exists, the plaintext will not be uploaded [6] [7]. In terms of deduplication granularity, Harnik et al. [8] categorized deduplication strategies into two types, the file-level deduplication [9] and the block-level deduplication [10] [11], based on the data units handled. From the perspective of deduplication architecture, data deduplication can be divided into two basic approaches, the target-based approach (or server-side deduplication), and the source-based approach (or client-side deduplication) [12]. However, deduplication may arise serious privacy concerns and pose new security challenges [13], such as privacy leakage [14] and unauthorized access [15].

Typically, the cloud service platforms are honest-but-curious, that is, they may leak privacy because of curiosity or collusion with the adversary. In order to prevent privacy leakage, the widely solution is to encrypt the privacy information [16]. Meanwhile, for achieving data deduplication in cloud, the convergence encryption (CE) [17] and message-locked encryption (MLE) [9] has been considered, where the encryption key is a hash of the file, therefore, the identical file generates the same key and same ciphertext. However, the CE algorithm easily suffers from the content guessing attack, where the adversary may guess the content based on the intercepted ciphertext [18] [19].

However, the above schemes could not satisfy the access control [20], that is, different individuals own different privileges, only authorized owners can assess the corresponding files. Li et al. [15] proposed a scheme in a hybrid cloud architecture for secure authorized deduplication. However, the solution of authorized deduplication is not suitable for the widely used hierarchical environment.

Moreover, most of the cloud applications are implemented under hierarchical environment, e.g., company managements, school managements, administrative agencies. Gonzalez-Manzano et al. [21] proposed an ase-PoW scheme in hierarchical environment, which used a lightweight access control procedure to resist against content guessing attack, where the encryption key is linked to the owners’ attributes. However, the proposed scheme does not consider key management, and does not describe the revocation and update policies of key. Furthermore, the process of key calculation is symmetric recursive encryption with the high computational overhead.

To tackle the above issues, we put forward a novel role symmetric encryption (RSE) algorithm and construct RSEDup scheme based on the role symmetric encryption to achieve secure deduplication in cloud. Up to now, it is the first solution to prevent privacy leakage and achieve the authorized deduplication effectively.

The contributions of the proposed scheme are shown as follows:

- We propose a novel role symmetric encryption algorithm in cloud, and define a tree structure represented as a hierarchical company to manage users’ role keys. For the hierarchical company, the company assigns the master key to each department, and the other keys of groups are assigned and calculated by the superior nodes. Different role keys represent different privileges, the encryption key is related to the role key according to access control policies, therefore, only users with specific role permissions can access the corresponding file.

- We propose a RSEDup scheme based on the RSE algorithm. We define the access control policies based on role, one key is related to one role. When the user’s permission is revoked, the corresponding role key should be deleted and updated. For the hierarchical company, we just delete the role key node of the given user, and reassign the master key to the department that includes the deleted node, the group keys are assigned and calculated by the superior node, and has no effect on the other departments, which can achieve flexible access control.

- We adopt the role symmetric encryption to prevent privacy leakage, and utilize the role-based access control mechanism to achieve authorized deduplication. Moreover, we consider the revocation and update policies of key, and implement our RSEDup scheme in hierarchical environments to achieve secure deduplication. Meanwhile, the performance evaluation makes it clear that the proposed scheme is efficient.

The rest of this paper is organized as follows: Section 2 introduces the system model. Section 3 describes the details of the RSE algorithm and the RSEDup scheme. Section 4 gives the performance analysis and evaluation. Finally, we draw the conclusions in Section 5.

2 SYSTEM MODEL

In this section, we describe the system model of the proposed RSEDup scheme.
The RSEDup scheme consists of three entities: a user, a cloud server (CS) and a role certificate service (RCS), as is shown in Figure 1.

![System model](image)

**Figure 1: System model. A user sends key request to RCS, and obtain the role key, then performs cryptography operations on files. The user also can request CS to upload/download files, CS returns the result to the user.**

*User.* Users who belong to different roles own the corresponding role keys in hierarchical environments. According to the role keys and access control policies, the user obtains the corresponding file keys to upload or download files from CS.

*Cloud Server.* CS is responsible for secure storage and authorized deduplication. CS stores the uploaded files in the storage servers, and implements authorized deduplication when a user requests to upload the file.

*Role Certificate Service.* RCS is responsible for role key management, including key generation, assignment, revocation and update.

3 CONSTRUCTION

In this section, we first describe the RSE algorithm, then give an elaborate description of our proposed RSEDup scheme.

**Role Symmetric Encryption Algorithm**

In the proposed RSE algorithm, we define a hierarchical company as a tree structure to manage users’ role keys, which is composed of different Departments \((D_i)\). Each department consists of different Groups \((G_i)\). Every department or group owns a unique identifier. The different files \((f_i)\) are managed by members belonging to a specific group which is the only master group of the given file \(f_i\), and the other members of groups also have part of permissions of it. A user, who works in different \(G_i\), owns given roles and manages specific \(f_i\). Meanwhile, the corresponding role keys are assigned to each user according to the roles.

In order to describe the process of key management and file management, we assume that a company is composed of two departments, as is shown in Figure 2. We define the departments as \(D_1\) and \(D_2\). \(D_1\) consists of two groups, \(G_1\) and \(G_2\), and the \(G_1\) is composed of two sub-groups \(G_3\) and \(G_4\), the \(G_2\) is composed of only one sub-group \(G_5\). While, \(D_3\) consists of one group \(G_6\), which is composed of only one sub-group \(G_7\). We assume that \(f_1\) is owned by \(G_3\), and the all members of \(G_3\) manage \(f_1\), the \(G_3\) is the master group of \(f_1\). While, \(f_2\) is owned by \(G_5\), and managed by members who work in \(G_3\) and have permission to access \(f_2\) in \(G_4\), the \(G_5\) is the master group of \(f_2\). And \(f_3\) is owned by \(G_7\), and managed by members who work in \(G_7\) and have permission to access \(f_3\) in \(G_5\), the \(G_7\) is the master group of \(f_3\).

Key management consists of key generation, assignment, revocation and update. For key generation and assignment, the company assigns the master key \(MK_1\) and \(MK_2\) to \(D_1\) and \(D_2\) respectively, and then, \(D_1\) calculates the hash of \(MK_1\) and concatenates with the identifier of \(G_1\), and assigns the hash of the above result \((KG_1)\) to \(G_1\). Similarly, the other group keys are assigned and calculated by the superior nodes. For key revocation and update, when the users’ permission is revoked, the node is deleted and the company just reassigns the master key to the department that consists of the deleted node, the other keys are updated by the superior nodes.

File management consists of file encryption and deduplication. For file encryption, \(f_1\) is only owned by \(G_3\), the members of \(G_3\) can access \(f_1\), therefore, \(f_1\) is symmetrically encrypted with the hash of group key \(KG_3\). \(f_2\) is owned by \(G_5\), and parts of members in \(G_4\) and \(G_5\) can access \(f_2\), therefore, \(f_2\) is symmetrically encrypted with the hash of group key \(KG_5\) concatenating with the hash of group key \(KG_4\). Similarly, \(f_3\) is symmetrically encrypted with the hash of group key \(KG_7\) concatenating with the hash of group key \(KG_5\). In terms of deduplication, we take \(f_1\) as an example. The user first uploads the encrypted \(f_1\) and the identifier to the server. Then, the server stores the encrypted \(f_1\) in the storage servers. Therefore, when a user uploads the same file \(f_1\), the server will retrieve the storage servers to check whether the file exists. If \(f_1\) exists, the server performs deduplication and returns the results to the user.
In order to obtain the role key $rk_i$, the user requests to RCS with a list of role $\{role_1, role_2, ..., role_n\}$, and then, RCS performs role certification and sends a corresponding list of role key $\{rk_1, rk_2, ..., rk_n\}$ according to the mechanism of key generation and assignment [22], as is shown in Figure 3.

![Figure 3: Role certification. The user request to obtain the role key, then RCS performs role certification and returns the list of role keys.](image)

To facilitate exploration of the above process, we define that a given file $f$ belongs to an ordered node chain, $\langle G_1, G_2, ..., G_i \rangle$, $i \in [1, m]$. Each node owns its unique identifier $G_i$, each role tree owns a master key $MK_\alpha$. Therefore, the role key $rk_i$ is defined as follows:

$$rk_i = H_1(\cdots H_1(H_1(MK_\alpha) || G_1) || G_2) \cdots || G_i) \quad (1)$$

After receiving the role keys, the user calculates the file keys according to the access control policies, we define $fk$ as the file key, which describe as follows:

$$fk = H_x(rk_1)||H_x(rk_2)||...||H_x(rk_i) \quad (2)$$

$i \in [1, n], x = 1or x = 2$.

The role keys are based on the access control policies and the hash operation depends on whether the group is the master group. If $f$ is only owned by one group, the $x$ is equal to 1. Otherwise, we assume that a given file $f$ managed by multiple groups, the role key of master group runs $H_1$ to obtain a piece of file key, and other groups runs $H_2$ to get other parts of file key, and then, concatenating with these keys to generate the file key of $f$.

**Role Symmetric Encryption Authorized Deduplication Scheme**

Our RSEDup scheme includes the following three phases: file uploading and file storage and file retrieving, as is shown in Figure 4.

**File uploading** The user performs RSE algorithm to get the file key $fk$, then the user calculates and uploads the encrypted file. Specifically, the user encrypts symmetrically $f$ with $fk$, we define the ciphertext as $\Theta$, $\Theta = Enc_{fk}f$, and calculates the hash of ciphertext, $h_f = H_3(\Theta)$ as the index, and then, calculates the hash of the user’s id, $eid = H_4(id)$, as is shown Algorithm 1. After that, the user integrates and sends $\{\Theta, h_f, eid\}$ to CS for requesting to store a file.

**File storage** After receiving the information from the user, CS firstly calculates the hash of $\Theta$, $h_f = $
A novel role symmetric encryption algorithm for authorized deduplication in cloud computing

Figure 4: RSEDup scheme. The user performs RSE algorithm to get the file key, then user calculates and uploads the encrypted file. CS stores the uploaded encrypted file and other information. When the user requests to upload a file, CS retrieves the storage servers to check whether the file exists and performs deduplication, then, CS returns the results to the user.

**Algorithm 1** File uploading

**Input:** rolelist, f, id

**Output:** h_f, Θ, eid

1. sending the role list to RCS and obtaining the role key rk;
2. calculating the file key fk according to the access control policies;
3. Θ ← Enc_fk(f);
4. h_f ← H_3(Θ);
5. eid ← H_4(id);

**Algorithm 2** File storage

**Input:** h_f, Θ, eid

**Output:** the array ξ[h_f]

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>h'_f = H_3(Θ);</td>
</tr>
<tr>
<td>2</td>
<td>if (h'_f == h_f) then return ⊥;</td>
</tr>
<tr>
<td>3</td>
<td>end if</td>
</tr>
<tr>
<td>4</td>
<td>ξ[h_f].ENC ← Θ;</td>
</tr>
<tr>
<td>5</td>
<td>ξ[h_f].EID ← EID;</td>
</tr>
</tbody>
</table>

**Algorithm 3** File retrieving

CS requires the user to send the index and encrypted identifier, {h_f, eid}. Secondly, if the index h_f is found in storage server and eid ∈ ξ[h_f].EID is verified, CS performs deduplication. Otherwise, CS requests the users to upload the encrypted file, and stores in the storage server. Thirdly, CS returns the file address to the user, as is shown Algorithm 3.

When a user requests to download the file, CS verifies the identity and performs retrieving, then returns the encrypted file to the user. Finally, the user calculates file...
ALGORITHM 3 File retrieving

input: \( h_f, \text{eid} \)
output: the results.

1: retrieving the existence of the index \( h_f \);
2: if \( h_f \) exists then
3: if \( \text{eid} \in \xi[\{h_f\}.EID] \) then
4: performing deduplication;
5: return sending file address.;
6: end if
7: else
8: return uploading the encrypted file.;
9: end if

key based on RSE algorithm to decrypt the encrypted file.

4 PERFORMANCE ANALYSIS AND EVALUATION

In this section, we give the performance analysis and evaluation of our proposed schemes.

Complexity Analysis

In terms of the complexity of our scheme, we consider the computational cost of three phases. In order to make the results more intuitive, we assume the hash functions have the same computational cost. For the RSEDup scheme, we use the recursive hash and concatenating operations to obtain the file key, which is related to the role key and access control policies, regardless of the files. In the file uploading phase, the user encrypts symmetrically \( f \) to obtain the ciphertext \( \Theta \), and calculates the hash of ciphertext as index, and calculates the hash of the user’s \( id \), therefore, the computational cost is \( O(F) \cdot \text{AES} \cdot \text{hash} \cdot \text{hash} \). In the phase of file storage, CS calculates the hash of \( \Theta \), verifies the results, and establishes the maps of 2-tuples. The main computational cost is \( O(F) \cdot \text{hash} \). In the phase of file retrieving, the computational cost is related to the specific retrieving algorithm and deduplication algorithm.

Performance Measurements

We measure the computational cost of the proposed RSEDup scheme by using OpenSSL library for cryptographic operations. Particularly, we employ AES-256 and SHA-256 for symmetric encryption and hash algorithms, and we use C++ to develop a prototype system on Linux. Our experiment is implemented on a test computer with the following configurations: CPU: Intel Core i5-4590 3.30GHz; RAM: 8GB; Hard disk: WDC WD10EZX-08M2NA0(1 TB / 7200 r/min); and OS: Ubuntu 12.04.4 LTS.

We measure the computational cost of two phases in RSEDup scheme: the phase of file uploading and file storage. All of the computations run 200 times to get the average value. In our experiments, we choose nine files with different sizes as follows: 2MB, 4MB, 8MB, 16MB, 32MB, 64MB, 128MB, 256MB and 512MB. The results are shown in Figure 5 and Figure 6.

Figure 5: The computational cost of file uploading.

Figure 6: The computational cost of file storage.

From the above two figures for the computational cost of the two phases, we know that the trend of these curves is consistent with the above analysis with the increasing of the file size. The main computational cost of the file uploading phase is the running time of AES symmetric encryption and hash algorithm over the whole file. The running time of generating role keys and file keys has not significantly increased, therefore, the computational
A novel role symmetric encryption algorithm for authorized deduplication in cloud 10th, July 2017, Chongqing, China

cost of generating keys can be considered later. In the file storage phase, the cloud server calculates the hash of the encrypted file, verifies the results, and establishes the maps of 2-tuples. The main computational cost is the hash of Θ. From Figure 6, with the increasing of file size, and the computational cost has a significantly growth.

5 CONCLUSION

In the era of big data, one of the critical challenges is to improve the storage efficiency and reduce the management expenditure. Data deduplication is a promising solution, and also has privacy concerns and security challenges. In this paper, we propose a novel role symmetric encryption (RSE) algorithm and construct an authorized deduplication scheme (RSEDup) to achieve the authorized deduplication in hierarchical environments. We adopt the role symmetric encryption to prevent privacy leakage, and utilize the role access control mechanism to achieve authorized deduplication. The performance evaluation makes it clear that the proposed scheme is efficient.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (61402109, 61370078, 61502102, and 61502103); National Natural Science Foundation of Fujian Province (2015J05120, 2016J05149, 2017J01737 and 2017J05099); Fujian Provincial Key Laboratory of Network Security and Cryptology Research Fund (Fujian Normal University) (15008); Distinguished Young Scientific Research and Cryptology Research Fund (Fujian Normal University (15008)); Distinguished Young Scientific Research Talents Plan in Universities of Fujian Province (2015, 2017).

REFERENCES


