International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

A Cloud Approach for Secure Multi Data sharing between the Users

Deeti Naga Vijay [1], A. Jyothi[2]

[1]M. Tech student, Department of CSE, Anurag Group of Institutions, Hyderabad, India

[2]Assistant Professor, Department of CSE, Anurag Group of Institutions, Hyderabad, India

Abstract: ***Cloud computing provides an economical and efficient solution for sharing group resource with cloud users. Sharing data in a multi-owner manner while preserving data and identity privacy from an untrusted cloud due to the frequent change of the membership. Cloud intend the secure multiple data sharing between the data users by using the cloud computing. By leveraging group signature and dynamic broadcast encryption techniques. Any user can share the data with other users. The storage overhead and encryption computation cost of our scheme are independent with the number of revoked users the security of our scheme with rigorous proofs, and demonstrate the efficiency of our scheme in experiments.***

Keywords: Cloud computing, data sharing, privacy-preserving, dynamic groups, access control.

1. Introduction

Cloud computing is recognized as an alternative to traditional information technology [1] due to its intrinsic resource-sharing and low-maintenance characteristics. In the cloud computing, the cloud service providers (CSPs), such as Amazon, are capable to deliver various services to cloud users with the help of powerful datacenters. By migrating the local data management systems into cloud servers, users can benefit from high-quality services and save significant investments on their local infrastructures. Single most fundamental services offered by cloud providers is data storage. Allow us to consider a practical data application. Company allows its staffs in the same group or department to store and share files in the cloud. By using the cloud, the staffs can be completely released from the troublesome local data storage and maintenance. Though, it also poses a significant risk to the confidentiality of those stored files. Particularly, the cloud servers managed by cloud providers are not fully trusted by users while the data files stored in the cloud may be sensitive and private, such as business plans. To preserve data privacy, a fundamental solution is to encrypt data files, and after that upload the encrypted data into the cloud [2]. Unluckily, designing an efficient and secure data sharing scheme for groups in the cloud is not an easy task due to the following challenging issues.

First, identity privacy is one of the most significant obstacles for the wide deployment of cloud computing. Exclusive of the guarantee of identity privacy, the users may be unwilling to join in cloud computing systems because their real identities could be easily disclosed to cloud providers and attackers. Alternatively, unconditional identity privacy may incur the abuse of privacy. For ex, a misbehaved staff can deceive others in the company by sharing false files without being traceable. So, traceability, which enables the group manager (example: a company manager) to reveal the real identity of a user, is as well highly desirable.

Second, it is highly recommended that any member in a group should be able to fully enjoy the data storing and sharing services provided by the cloud, which is clear as the multiple-owner manner. Compared with the single-owner manner [3], where only the group manager can store and modify data in the cloud, the multiple-owner manner is more supple in practical applications. More concretely, every user in the group is able to not only read data, except also modify his/ her part of data in the entire data file shared by the company.

Last but not least, groups are normally dynamic in practice, example. New staff participation and current employee revocation in a company. The changes of membership make protected data sharing extremely difficult. On one hand, the nameless system challenges new granted users to learn the content of data files stored before their participation, since it is impossible for new granted users to contact with anonymous data owners, and attain the corresponding decryption keys. Alternatively, an efficient membership revocation mechanism without updating the secret keys of the remaining users is also desired to minimize the complexity of key management. Several security schemes for data sharing on untrusted servers have been proposed [4], [5], [6]. In these approaches, data owners store the encrypted data files in untrusted storage and distribute the corresponding decryption keys only to authorized users. Therefore, unauthorized users as well as storage servers cannot learn the content of the data files because they have no knowledge of the decryption keys. Though, the complexities of user participation and revocation in these schemes are linearly increasing with the number of data owners and the number of revoked users, correspondingly. By setting a group with a single attribute, Lu et al.[7] proposed a secure provenance scheme based on the ciphertext-policy attribute-based encryption technique [8], which let any member in a group to share data with others. Though, the issue of user revocation is not addressed in their scheme. Yu et al. [3] obtainable a scalable and fine-grained data access control scheme in cloud computing based on the key policy attribute-based encryption (KP-ABE) technique [9]. Unfortunately, the singleowner manner hinders the adoption of their scheme into the container, where any user is granted to store and share data. Our contributions. To solve the challenges, we propose Mona, a secure multi-

owner data sharing scheme for dynamic groups in the cloud. The main contributions of this paper contain:

1) We propose a secure multi-owner data allocation scheme. It implies that any user in the group can securely share data with others by the untrusted cloud.
2) Our proposed scheme is able to support dynamic groups efficiently. Specially, new granted users can directly decrypt data files uploaded before their participation without contacting with data owners.
3) User revocation can be easily achieved through a novel revocation list without updating the secret keys of the remaining users. The size and addition overhead of encryption are constant and independent with the number of revoked users.
4) We provide secure and privacy-preserving access control to users, which guarantees several member in a group to anonymously utilize the cloud resource. Furthermore, the real identities of data owners can be revealed by the group manager when disputes occur.
5) We provide rigorous security analysis, and do extensive simulations to demonstrate the efficiency of our scheme in terms of storage and computation overhead.

2. System Model and Design Goals

2.1 System Model

We consider a cloud computing architecture by combining with an example that a company uses a cloud to enable its staffs in the same group or department to share files. The system model consists of three dissimilar entities: the cloud, a group manager (i.e., the company manager), and a large number of group members (i.e., the staffs) as illustrated in Fig. 1.

Figure 1: System Model

Cloud is operated by CSPs and provides priced abundant storage services. However, the cloud is not fully trusted by users since the CSPs are very likely to be outside of the cloud users' trusted domain. Like [3], [7], we assume that the cloud server is honest but curious. Specifically, the cloud server will not maliciously delete or modify user data due to the protection of data auditing schemes [17], [18], but will try to learn the content of the stored data and the identities of cloud users.

Group manager takes charge of system parameters generation, user revocation, user registration, and enlightening the real identity of a dispute data owner. In the given ex, the group manager is acted by the administrator of the company. So, we assume that the group manager is fully trusted by the new parties. Group members are a set of registered users that will store their private data into the cloud server and share them with others in the group. In our ex, the staffs play the role of group members. Note that, the group membership is animatedly changed, due to the staff resignation and new employee participation in the company.

2.2 Design Goals

In this section, we describe the main design goals of the proposed scheme including access control, data confidentiality, traceability and anonymity, and efficiency as follows: Access control: The requirement of access control is twofold. In First, group members are able to use the cloud resource for data operations. In Second, unauthorized users cannot access the cloud resource at every time, and revoked users will be incapable of using the cloud again once they are revoked. Data confidentiality: in Data confidentiality requires that unauthorized users including the cloud are incapable of learning the content of the stored data. Challenging and an important issue for data confidentiality is to maintain its availability for dynamic groups. Particularly, new users should decrypt the data stored in the cloud before their participation, and revoked users are not capable to decrypt the data moved into the cloud after the revocation. Anonymity and traceability: the Anonymity guarantees that group members can access the cloud without revealing the real identity. Even though anonymity represents an effective protection for user identity, it as well poses a potential inside attack risk to the system. For ex, an inside attacker may store and share a mendacious information to derive substantial benefit. So, to tackle the inside attack, the group manager should have the capability to reveal the real identities of data owners. The efficiency is defined as follows: Any group member can store and share data files with others in the group by the cloud . User revocation can be achieved without involving the remaining users. That is, the remaining users do not need to update their private keys or reencryption operations. And New granted users can learn all the content data files stored before his participation without contacting with the data owner.

3. Literature Survey

Armando Fox, Michael Armbrust, Rean Griffith, Anthony D. Joseph, Randy Katz, Gunho Lee, Andy Konwinski, David Patterson, Ion Stoica, Ariel Rabkin, and Matei Zaharia [1] in Above the Clouds: A View of Cloud Computing paper predict Cloud Computing *will* grow, so developers should obtain it into account. in spite of whether a cloud provider sells services at a low level of abstraction like EC2 or a higher level like AppEngine, we believe that computing, storage and networking must all focus on horizontal scalability of virtualized resources rather than on single node performance. Moreover:

1) *Applications Software* needs to both scale *down* rapidly plus scale up, which is a new requirement. Such software also needs a pay-for-use licensing model to match needs of Cloud Computing.

2) *Infrastructure Software* needs to be aware that it is no longer running on bare metal but on VMs. Furthermore, billing needs to built in from the start.
3) *Hardware Systems* should be designed at the scale of a container (at least a dozen racks), which will be is the minimum purchase size. Cost of operation will competition performance and cost of purchase in significance, rewarding *energy proportionality* such as by putting idle portions of the memory, disk, and the network into low power mode. Processors should work fine with VMs and flash memory should be added to the memory hierarchy, and LAN switches andWAN routers must improve in bandwidth and cost.

Seny Kamara Kristin Lauter[2] in Cryptographic Cloud Storage paper We consider the problem of building a secure cloud storage service on top of a public cloud in- frastructure where the service provider is not completely trusted by the customer. We describe, at a high level, several architectures that combine recent and non standard cryptographic prim- itives in order to achieve our goal. We survey the benefits such architecture would provide to both customers and service providers and give an overview of recent advances in cryptography motivated specifically by cloud storage.

Shucheng Yu, Cong Wang†, Kui Ren†, and Wenjing Lou [3] in Achieving Scalable, Secure , and Fine-grained Data Access Control in Cloud Computing paper Cloud computing is an emerging computing paradigm in which resources of the computing infrastructure are provided as services over the Internet. As shows potential as it is, this paradigm also brings forth many new challenges for data security and access control when users outsource sensitive data for sharing on cloud servers, which are not in the same trusted domain as data owners. To remain sensitive user data confidential against untrusted servers, existing solutions frequently apply cryptographic methods by disclosing data decryption keys only to authorized users. Though, in doing so, these solutions inevitably introduce a heavy computation overhead on the data owner for key distribution and data management when finegrained data access control is desired, and therefore do not scale well. The problem of simultaneously achieving fine-grainedness, data confidentiality and scalability of access control actually still remains unresolved. In this paper addresses this challenging open issue by, alternatively, defining and enforcing access policies based on data attributes, and, on the other hand, let the data owner to delegate most of the computation tasks involved in finegrained data access control to untrusted cloud servers without disclosing the underlying data contents. We complete this goal by exploiting and uniquely combining techniques of attribute-based encryption (ABE), proxy re-encryption, and the lazy re-encryption. Our anticipated scheme also has salient properties of user access privilege confidentiality and user secret key accountability. Wide analysis shows that our proposed schemes is highly efficient and provably secure under existing security models.

Mahesh Kallahalla, Erik Riedel† Ram Swaminathan, Qian Wang‡ Kevin Fu§ [4] in Plutus: Scalable secure file sharing on untrusted storage paper Plutus is a cryptographic storage system that enables secure file sharing without placing much trust on the file servers. In particular, it makes novel use of cryptographic primitives to protect and share files. Plutus features extremely scalable key management while allowing individual users to retain direct control over who gets access to their files. We clarify the mechanisms in Plutus to reduce the number of cryptographic keys exchanged between users by using file groups, file read and write access, handle user revocation competently, and allow an untrusted server to authorize file writes. We have built a model of Plutus on OpenAFS. Dimensions of this prototype show that Plutus achieves strong security with overhead comparable to systems that encrypt all network traffic.

4. The Proposed Scheme: Mona

4.1 Overview

To achieve secure data sharing for dynamic groups in the cloud, and expect to combine the group signature and dynamic broadcast encryption techniques. Particularly, the group signature scheme enables users to anonymously use the cloud resources, the dynamic broadcast encryption technique allows data owners to securely share their data files with others including new joining users.

Unfortunately, each user has to compute revocation parameters to protect the confidentiality from the revoked users in the dynamic broadcast encryption scheme, which results in the both the computation overhead of the encryption and the size of the ciphertext increase with the number of revoked users. So, the heavy overhead and large ciphertext size may hinder the adoption of the broadcast encryption scheme to capacity-limited users. To tackle this challenging issue, let the group manager compute the revocation parameters and make the result public available by migrating them into the cloud. Such a design can considerably reduce the computation overhead of users to encrypt files and the ciphertext size. Particularly, the computation overhead of users for encryption operations and the ciphertext size is constant and independent of the revocation users.

4.2 Scheme Description

Here it describes the details of Mona including system initialization, user registration, user revocation, file generation, file deletion, file access and traceability.

4.2.1 System Initialization

The group manager takes charge of the system initialization as follows:generate a bilinear map group system $S = (q, G_1, G_2, e(\cdot,\cdot))$.

TABLE 1
Revocation List

ID_{group}	A_1	x_1	t_1	P_1			
	A_2	x_2	t_2	P_2			
	.	.	.	.			
	A_r	x_r	t_r	P_r	Z_r	t_{RL}	$sig(RL)$

Selecting two random elements $H, H_0 \in G_1$along with two random numbers $\xi_1, \xi_2 \in Z_q^*$, and computing $U = \xi_1^{-1} H \ V = \xi_2^{-1} H \in G_1$ such that

$\xi_1 \cdot U = \xi_2 \cdot V = H$. In addition, the group manager computes $H_1 = \xi_1 H_0$ and $H_2 = \xi_2 H_0 \in G_1$. Randomly choosing two elements $P, G \in G_1$ and a number $\gamma \in Z_q^*$, and computing $W = \gamma \cdot P, Y = \gamma \cdot G$ and $Z = e(G, P)$, in that order. Publishing the system parameters including $(S, P, H, H_0, H_1, H_2, U, V, W, Y, Z, f, f_1, Enc())$, where f is a one-way hash function $\{0,1\}^* \rightarrow Z_q^*$; f_1 is hash function: $\{0,1\}^* \rightarrow G_1$; and $Enc_k()$ is a secure symmetric encryption algorithm with secret key k.
In the end, the parameter $(\gamma, \xi_1, \xi_2, G)$ will be kept secret as the master key of the group manager.

4.2.2 User Registration

For the registration of user i with identity IDi, the group manager at random selects a number $x_i \in Z_q^*$ and computes Ai;Bi as the following equation:

$$\begin{cases} A_i = \dfrac{1}{\gamma + x_i} \cdot P \in G_1 \\ B_i = \dfrac{x_i}{\gamma + x_i} \cdot G \in G_1. \end{cases} \quad (1)$$

Then, the group manager adds (A_i, x_i, ID_i) into the group user list, this will be used in the traceability phase. Once the registration, user i obtains a private key (x_i, A_i, B_i), which will be used for group signature generation and file decryption.

4.2.3 User Revocation

User revocation is performed by the group manager via a public available revocation list (RL), based on which group members can encrypt their data files and ensure the confidentiality against the revoked users. As illustrated in Table 1, the revocation list is characterized by a series of time stamps (t1 < t2 <… tr). Let IDgroup denote the group identity. The tuple (Ai; xi; ti) represents that user i with the partial private key (Ai; xi) is revoked at time ti. P1, P2 … Pr and Zr are calculated by the group manager with the private secret γ as follows:

$$\begin{cases} P_1 = \dfrac{1}{\gamma + x_1} \cdot P \in G_1 \\ P_2 = \dfrac{1}{(\gamma + x_1)(\gamma + x_2)} \cdot P \in G_1 \\ \cdots \\ P_r = \dfrac{1}{(\gamma + x_1)(\gamma + x_2) \cdots (\gamma + x_r)} \cdot P \in G_1 \\ Z_r = Z^{\frac{1}{(\gamma + x_1)(\gamma + x_2) \cdots (\gamma + x_r)}} \in G_2. \end{cases} \quad (2)$$

Motivated by the verifiable reply mechanism in [19], to guarantee that users obtain the latest version of the revocation list, we let the group manger update the revocation list each day even no user has being revoked in the day. Other words, the others can verify the freshness of the revocation list from the contained current date tRL. Also, the revocation list is bounded by a signature sig(RL) to declare its validity. And the signature is generated by the group manager with the BLS signature algorithm [20], i.e $sig(RL) = \gamma f_1(RL)$. finally, the group manager migrates
the revocation list keen on the cloud for public usage.

4.2.4 File Generation

To store and share a data file in the cloud, and a group member performs the following operations:

1) Getting the revocation list from the cloud. And in this step, the member sends the group identity IDgroup as a request to the cloud. After that, the cloud responds the revocation list RL to the member.
2) Verifying the validity of the received revocation list. First, checking whether the marked the date is fresh. In Second, verifying the contained signature sig(RL) by the equation $e(\bar{W}, f_1(RL)) = e(\bar{P}, sig(RL))$. If the revocation list is invalid, and the data owner stops this scheme.
3) Encrypting the data file M. This encryption process can be divided into two cases according to the revocation list.

a. **Case 1.** There is no revoked user in the revocation list:
i. Selecting a unique data file identity IDdata;
ii. Choosing a random number $k \in Z_q^*$;
iii. Computing the parameters C1, C2, K, C as the following equation:

$$\begin{cases} C_1 = k \cdot Y \in G_1 \\ C_2 = k \cdot P \in G_1 \\ K = Z^k \in G_2 \\ C = Enc_K(M). \end{cases} \quad (3)$$

b. **Case 2**. There are r revoked users in the revocation list.
i. Selecting a unique data file identity IDdata;
ii. Choosing a random $k \in Z_q^*$;
iii. Computing the parameters C1, C2, K, C as the following equation:

$$\begin{cases} C_1 = k \cdot Y \in G_1 \\ C_2 = k \cdot P_r \in G_1 \\ K = Z_r^k \in G_2 \\ C = Enc_K(M). \end{cases} \quad (4)$$

In (4), Zr and Pr are directly obtained from the revocation list.
4. Selecting a random number τ and computing $f(\tau)$. The hash value will be used for data file deletion operation. In addition, the data owner adds (ID_{data}, τ) into his local storage.
5. Constructing the uploaded data file as shown in Table 2, where t data denotes the current time on the

TABLE 2
Message Format for Uploading Data

Group ID	Data ID	ciphertext	hash	Time	Signature
ID_{group}	ID_{data}	C_1, C_2, C	$f(\tau)$	t_{data}	σ

member, and σ is a group signature on $(ID_{data}, C_1, C_2, C, f(\tau), t_{data})$ computed by the data owner through Algorithm 1 with the private key (A, x).
6. Uploading the data shown in Table 2 into the cloud server and adding the IDdata into the local shared data list maintained by the manager. Going on receiving the data, the cloud first invokes Algorithm 2 to check its validity. And if the algorithm returns true, the group signature is valid; or else, the cloud abandons the data. Also, if several users have been revoked by the group manager, cloud also performs revocation verification by using Algorithm 3. lastly, the data

file will be stored in the cloud after successful group signature and revocation verifications.

4.2.5 File Deletion

File stored in the cloud can be deleted by either the group manager or the data owner (i.e., the member who uploaded the file into the server). To delete a file IDdata, the group manager computes a signature $\gamma f_1(ID_{data})$ and sends the signature along with IDdata to the cloud. Cloud will delete the file if the equation $e(\gamma f_1(ID_{data}), P) = e(W, f_1(ID_{data}))$ holds.

Algorithm (1). Signature Generation

Input: Private key (A, x), system parameter (P, U, V, H, W) and data M.

Output: Generate a valid group signature on M.

begin

Select random numbers $\alpha, \beta, r_\alpha, r_\beta, r_x, r_{\delta_1}, r_{\delta_2} \in Z_q^*$

Set $\delta_1 = x\alpha$ and $\delta_2 = x\beta$

Computes the following values

$$\begin{cases} T_1 = \alpha \cdot U \\ T_2 = \beta \cdot V \\ T_3 = A_i + (\alpha + \beta) \cdot H \\ R_1 = r_\alpha \cdot U \\ R_2 = r_\beta \cdot V \\ R_3 = e(T_3, P)^{r_x} e(H, W)^{-r_\alpha - r_\beta} e(H, P)^{-r_{\delta_1} - r_{\delta_2}} \\ R_4 = r_x \cdot T_1 - r_{\delta_1} \cdot U \\ R_5 = r_x \cdot T_2 - r_{\delta_2} \cdot V \end{cases}.$$

Set c =f(M,T1, T2, T3,R1,R2,R3,R4,R5)

Construct the following numbers

$$\begin{cases} s_\alpha = r_\alpha + c\alpha \\ s_\beta = r_\beta + c\beta \\ s_x = r_x + cx \\ s_{\delta_1} = r_{\delta_1} + c\delta_1 \\ s_{\delta_2} = r_{\delta_2} + c\delta_2 \end{cases}.$$

Return $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$

End

Algorithm (2). Signature Verification

Input: System parameter (P, U, V, H, W), M and a signature $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$

Output: True or False.

begin

Compute the following values

$$\begin{cases} \tilde{R}_1 = s_\alpha \cdot U - c \cdot T_1 \\ \tilde{R}_2 = s_\beta \cdot V - c \cdot T_2 \\ \tilde{R}_3 = (\frac{e(T_3, W)}{e(P, P)})^c e(T_3, P)^{s_x} e(H, W)^{-s_\alpha - s_\beta} \\ \qquad e(H, P)^{-s_{\delta_1} - s_{\delta_2}} \\ \tilde{R}_4 = s_x \cdot T_1 - s_{\delta_1} \cdot U \\ \tilde{R}_5 = s_x \cdot T_2 - s_{\delta_2} \cdot V \end{cases}.$$

if $c = f(M, T_1, T_2, T_3, \tilde{R_1}, \tilde{R_2}, \tilde{R_3}, \tilde{R_4}, \tilde{R_5})$

Return True

else

Return False

end

Algorithm (3). Revocation Verification

Input: System parameter (H0, H1, H2), a group signature σ_r and a set of revocation keys A1,……..,Ar

Output: Valid or Invalid.

begin

set $temp = e(T_1, H_1)e(T_2, H_2)$

for $i = 1$ to n

if $e(T_3 - A_i, H_0) = temp$

Return Valid

end if

end for

Return Invalid

end

In addition, Mona also allows data owners to delete their files stored in the cloud. Particularly, the data owner does the following actions:

. Obtaining the tuple (ID_{data}, τ) from his local storage.

Invoking Algorithm 1 to calculate a group signature on (ID_{data}, τ).

Sending (ID_{data}, τ) and the signature as a deletion request to the cloud.

Upon receiving the deletion request, the cloud can calls Algorithms 2 and 3 to check whether the group signature. After an successful group signature verification, the cloud can delete the data file if $f(\tau)$ equals to the hash value contained in the file.

4.2.6 File Access

To learn the content of a common file, a member does the following actions:

1. Getting the data file and the revocation list from the cloud server. From this operation, the user first adopts its private key (A, x) to calculate a signature σ_u on this message $(ID_{group}, ID_{data}, t)$ by using this Algorithm 1, where t can denote the current time, and the IDdata can obtained from the local shared file list maintained by the manager. Then, the user can sends a data request contain $(ID_{group}, ID_{data}, t, \sigma_u)$ to the cloud server. Upon receiving to the request, the cloud server employs Algorithm 2 to check the validity of the signature and performs a revocation verification with Algorithm 3 if necessary according to the revocation list. After a successful authentication, the cloud server can responds to the corresponding data file and the revocation list to the user.

2. Checking the validity of the revocation list. This operation was similar to the step 2 of file generation phase.

3. Verifying the validity of the file and decrypting it. The format of downloaded file coincides with that given in Table 2. That operation can be divided into three cases according to the time stamp tdata and the revocation list. Assume that there are r revoked users in the revocation list.

a. **Case 1** $(t_{data} < t_1)$. This case indicates that there is no revoked user before the data file is uploaded

i. Invoking Algorithm 2 to check the group signature σ. If the algorithm returns false, the user can stops this protocol.

ii. Using his partial private key (A,B) to calculate $K = e(C_1, A)e(C_2, B)$.

iii. Decrypting the ciphertext C with the computed key $\hat{K}$.

Correctness:

$$\hat{K} = e(C_1, A)e(C_2, B)$$
$$= e\left(k \cdot Y, \frac{1}{\gamma + x} \cdot P\right) e\left(k \cdot P, \frac{x}{\gamma + x} \cdot G\right)$$
$$= e(G, P)^{\frac{k\gamma}{\gamma+x}} e(P, G)^{\frac{kx}{\gamma+x}}$$
$$= Z^k = K.$$

b. **Case 2** $(t_i < t_{data} < t_{i+1})$ This case indicates that I revoked users have been revoked before the data file is uploaded

i. Verifying the group signature _ by using Algorithm 2.

ii. Inputting A1, A2,…., Ai to call Algorithm 3. If the algorithm can returns invalid, the user can terminates this operation.

iii. Computing the value

$$A_{i,r} = \frac{1}{(\gamma + x)\prod_{\lambda=1}^{i}(\gamma + x_\lambda)} P$$

by using Algorithm 4 with the input (A, x), $(P_1, x_1), \ldots, (P_i, x_i)$. The correctness of Ai;r is due to the following relation:

$$\frac{1}{x - x_i}\left(P_i - \frac{1}{(\gamma + x)\prod_{\lambda=1}^{i-1}(\gamma + x_\lambda)} P\right)$$
$$= \frac{1}{x - x_i}\left(\frac{(\gamma + x) - (\gamma + x_i)}{(\gamma + x)(\gamma + x_i)(\prod_{\lambda=1}^{i-1}(\gamma + x_\lambda))}\right) P$$
$$= \frac{1}{(\gamma + x)\prod_{\lambda=1}^{i}(\gamma + x_\lambda)} P.$$

iv. Calculating the decryption key $\hat{K} = e(C_1, A_{i,r})e(C_2, B)$.

v. Decrypt the ciphertext C with the key $\hat{K}$.

Correctness

$$e(C_1, A_{i,r})e(C_2, B)$$
$$= e\left(kY, \frac{1}{(\gamma + x)\prod_{\lambda=1}^{i}(\gamma + x_\lambda)} P\right)$$
$$e\left(kP_i, \frac{x}{\gamma + x} \cdot G\right)$$
$$= e(P, G)^{\frac{k\gamma}{(\gamma+x)\prod_{\lambda=1}^{i}(\gamma+x_\lambda)}} e(P, G)^{\frac{kx}{(\gamma+x)\prod_{\lambda=1}^{i}(\gamma+x_\lambda)}}$$
$$= e(P, G)^{\frac{k\gamma+kx}{(\gamma+x)\prod_{\lambda=1}^{i}(\gamma+x_\lambda)}} = Z_i^k = K.$$

c. Case $(t_r < t_{data})$. This case indicates that r revoked users have been revoked before the data file is uploaded

i. Verifying the group signature _ by using Algorithm 2.

ii. Inputting $A_1, A_2, \ldots A_r$ to call Algorithm 3. If the algorithm returns invalid, the user terminates this operation.

iii. Computing the value

$$A_{r,r} = \frac{1}{(\gamma + x)\prod_{\lambda=1}^{r}(\gamma + x_\lambda)} P$$

by using Algorithm 4 with the input (A, x), $(P_1, x_1), \ldots, (P_r, x_r)$.

iv. Calculating the decryption key $\hat{K} = e(C_1, A_{r,r})e(C_2, B)$.

v. Decrypting the ciphertext C with the key $\hat{K}$.

4.2.7 Traceability

When a data dispute occurs, the tracing operation is performed by the group manager to identify the real identity of the data owner. Given a signature $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$, the group manager employs his private key (ξ_1, ξ_2) to compute $A_i = T_3 - (\xi_1 \cdot T_1 + \xi_2 \cdot T_2)$. Given the parameter Ai, the group manager can look up the user list to find the corresponding identity.

Algorithm 4. Parameters Computing

Input: The revoked user parameters $(P_1, x_1), \ldots, (P_r, x_r)$, and private key (A, x).

Output: Ar;r or NULL

begin
  set $temp = A$
  for $\lambda = 1$ to r
    if $x = x_\lambda$
      return NULL
    else
      set $temp = \frac{1}{x - x_\lambda}(P_\lambda - temp)$
  return $temp$
end

5. Security Analysis

In this section, we prove the security of Mona in terms of access control, anonymity, data confidentiality and traceability that are defined above

Theorem 1. Based on the group signature technique, the proposed scheme can achieve efficient access control.

Proof. To access the cloud, a user needs to compute a group signature for his/her authentication. That employed group signature system can be regarded as a variant of the short group signature [12], which inherits the inherent enforceability property, anonymous authentication, and tracking capacity. The look of Theorem 1 can be derived from the following three lemmas: tu

Lemma 1.1. Unrevoked users are able to access the cloud.

Proof. The proof ofLemma1.1 is equivalent to the correctness of Algorithm 2 (group signature verification $\tilde{R}_1 = R_1$ holds since $\tilde{R}_1 = s_\alpha \cdot U - c \cdot T_1 = (r_\alpha + c\alpha)U - c \cdot \alpha \cdot U =$ R1.Analogously,we can directly obtain $\tilde{R}_2 = R_2, \tilde{R}_4 = R_4, \tilde{R}_5 = R_5. \tilde{R}_3 = R_3$ holds due to the following relations:

$$\tilde{R}_3 = \left(\frac{e(T_3,W)}{e(P,P)}\right)^c e(T_3,P)^{s_x} e(H,W)^{-s_\alpha - s_\beta} e(H,P)^{-s_{\delta_1} - s_{\delta_2}}$$
$$= \left(\frac{e(T_3,W)}{e(P,P)}\right)^c e(T_3,P)^{r_x + cx_i} e(H,W)^{-r_\alpha - c\alpha - r_\beta - c\beta}$$
$$e(H,P)^{-r_{\delta_1} - cx_i\alpha - r_{\delta_2} - cx_i\beta}$$
$$= \left(\frac{e(T_3,W)}{e(P,P)}\right)^c e(T_3, x_i P)^c e(-(\alpha+\beta)H, W + x_i P)^c$$
$$e(T_3,P)^{r_x} e(H,W)^{-r_\alpha - r_\beta} e(H,P)^{-r_{\delta_1} - r_{\delta_2}}$$
$$= \left(\frac{e(T_3,W)}{e(P,P)}\right)^c e(T_3, x_i P)^c e(-(\alpha+\beta)H, W + x_i P)^c$$
$$R_3 = \left(\frac{e(T_3,W)}{e(P,P)}\right)^c e(T_3 - (\alpha+\beta)H, W + x_i P)^c e(T_3, W)^{-c} R_3$$
$$= \left(\frac{e(A_i, W + x_i P)}{e(P,P)}\right)^c R_3 = R_3.$$

Lemma 1.2. Revoked users cannot utilize the cloud after their revocation.

Proof. Lemma 1.2 is equivalent to the accuracy of Algorithm 3 (revocation verification). The accuracy of revocation verification is based on the following relation:

$$e(T_3 - A_i, H_0) = e(A_i + (\alpha+\beta)\cdot H - A_i, H_0)$$
$$= e(\alpha H, H_0) e(\beta H, H_0)$$
$$= e(\alpha U, \xi_1 H_0) e(\beta V, \xi_2 H_0)$$
$$= e(T_1, H_1) e(T_2, H_2).$$

Lemma 1.3. An attacker is unable to access the cloud server based on the statement of the intractability of q-SDH problem in G1.

Proof. The brief security analysis can be shown as follows: Suppose that an attacker A succeeds to forge a valid group signature with a non negligible probability in polynomial time. In addition to that we think f is a random oracle. With the help of Forking Lemma [21], by using the oracle replay method, the attacker A obtains two suitable signatures $(M, \sigma_0, c, \sigma_1)$ and $(M, \sigma_0, c', \sigma_1')$ as follows:

$$\begin{cases} \sigma_0 = (T_1, T_2, T_3, c, R_1, R_2, R_3, R_4, R_5) \\ c = f(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5) \\ c' = f'(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5) \\ \sigma_1 = (s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2}) \\ \sigma_1' = (s_\alpha', s_\beta', s_x', s_{\delta_1}', s_{\delta_2}') \end{cases} \quad (5)$$

$$\begin{cases} s_\alpha = r_\alpha + c\alpha, s_\alpha' = r_\alpha + c'\alpha \\ s_\beta = r_\beta + c\beta, s_\beta' = r_\beta + c'\beta \\ s_x = r_x + cx, s_x' = r_x + c'x \\ s_{\delta_1} = r_{\delta_1} + c\delta_1, s_{\delta_1}' = r_{\delta_1} + c'\delta_1 \\ s_{\delta_2} = r_{\delta_2} + c\delta_2, s_{\delta_2}' = r_{\delta_2} + c'\delta_2. \end{cases} \quad (6)$$

Then, A can compute an SDH tuple $(\hat{x} = \Delta s_x / \Delta c, \hat{A} = T_3 - ((\Delta s_\alpha + \Delta s_\beta)/\hat{\Delta c}) \cdot H)$ such that $A = \frac{1}{\gamma + \hat{x}}$ and $e(\hat{A}, W + \hat{x}P) = e(P,P)$, where $\Delta s_x = s_x - s_x'$, $\Delta c = c - c'$, $\Delta s_\alpha = s_\alpha - s_\alpha'$, an $\Delta s_\beta = s_\beta - s_\beta'$. Obviously, this contradicts with q-SDH assumption.

Theorem 2. The proposed scheme supports traceability and privacy preserving.

Proof. The display of this theorem is double. On one hand, the group manager has the capability to identify the real signer. Given a valid group signature $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$ and the private tuple (ξ_1, ξ_2), the group manger can compute the private key of the signer through the equation $A_i = T_3 - (\xi_1 \cdot T_1 + \xi_2 \cdot T_2)$. The correctness of the equation holds based on the following relation:

$$T_3 - (\xi_1 \cdot T_1 + \xi_2 \cdot T_2) = A_i + (\alpha+\beta)\cdot$$
$$H - (\xi_1\alpha \cdot U + \xi_2\beta \cdot V) = A_i.$$

On the other hand, other entities cannot reveal the signer's identity from a group signature, if not, DL assumption will be in inconsistency. Additional proofs on the accuracy, anonymity, unforgeability and traceability of group signatures can be found in [12].

Theorem 3. The proposed scheme protects data confidentiality under the hardness of theWBDHEproblem andGDHEproblem.

Proof. Theorem 3 can be deduced from the following two lemmas:

Lemma 3.1. The cloud server is unable to learn the content of the stored files.

Proof. To prove this lemma, we take a data file (C_1, C_2, C) as an example to demonstrate the data confidentiality, where $C_1 = k \cdot Y, C_2 = k \cdot P, K = Z^k, C = Enc_K(M)$ and no user has been revoked before the data file is uploaded. Suppose if the cloud server can calculate $K = Z^k$, i.e., "given $C_1 = k \cdot Y, C_2 = k \cdot P, P,$ for unidentified γ, computing $e(C_1, P)^{\frac{1}{\gamma}} = e(G,P)^k = K$."This contradicts with the WBDHE statement. On other hand, given the revocation list, and the cloud server can learns the incomplete private key of a revoked user i, i.e $(A_i, x_i)$for a revoked user. Without the knowledge of the other part private key Bi, it is also not capable to calculate the decryption key through the equation $e(C_1, A_i) e(C_2, B_i) = Z^k$. Thus, the accuracy of Lemma 3.1 can be ensured.

Lemma 3.2. Even under the collusion with revoked users, this cloud server can also not capable of learning the content of the files stored after their revocation.

Proof. We first define two polynomial functions $f(X) = \Pi_{i=1}^{r}(X + x_i)$ and $g(X) = \Pi_{i=1}^{n-r}(X + x_i')$. Let G0 and P0 denote two elements in group G1. Then, we set G = $f(\gamma)G_0$and $P = f(\gamma)g(\gamma)P_0$.To retain the confidentiality against the revoked users, the data owner calculate the header information C1, C2 and the encryption key K as follows:

$$\begin{cases} C_1 = kY = k\gamma f(\gamma) \cdot G_0 \\ C_2 = kP_r = \dfrac{k}{(\gamma + x_1)(\gamma + x_2) \cdots (\gamma + x_r)} P = kg(\gamma) P_0 \\ K = Z_r^k = Z^{\frac{k}{(\gamma+x_1)(\gamma+x_2)\cdots(\gamma+x_r)}} = Z^{\frac{k}{f(\gamma)}} \\ = e(G, P)^{f(\gamma)} = e(G_0, H_0)^{kf(\gamma)g(\gamma)}. \end{cases} \quad (7)$$

We can observe that it is not possible for revoked users to compute the encryption key K, since "given $k\gamma f(\gamma) \cdot G_0$ and $kg(\gamma)P_0$, computing $e(G_0, H_0)^{kf(\gamma)g(\gamma)}$, is an instance of (t,n)-GDHE problem, which has been demonstrated to be intractable in polynomial time [14]. By the analysis above, we conclude that the proposed scheme get the security goals including access control, data confidentiality as well as traceability and anonymity.

6. Conclusion

In this paper, we design a secure data sharing method, Mona, for the dynamic groups in an untrusted cloud. In Mona, a user is capable to the share data with others in the group with no revealing identity privacy to the cloud. Additionally, Mona can supports efficient user revocation and new user joining. More especially, efficient user revocation can be achieved from side to side a public revocation list without inform the private keys of the remaining users, and new users can straight decrypt files stored in the cloud before their participation. Also, the storage transparency and the encryption computation cost are constant. Wide analyses show that our future scheme satisfies the desired security requirements and guarantees efficiency as well.

References

[1] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing," Comm. ACM, vol. 53, no. 4, pp. 50-58, Apr. 2010.
[2] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," Proc. Int'l Conf. Financial Cryptography and Data Security (FC), pp. 136- 149, Jan. 2010.
[3] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing," Proc. IEEE INFOCOM, pp. 534-542, 2010.
[4] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc. USENIX Conf. File and Storage Technologies, pp. 29-42, 2003.
[5] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, "Sirius: Securing Remote Untrusted Storage," Proc. Network and Distributed Systems Security Symp. (NDSS), pp. 131-145, 2003.
[6] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," Proc. Network and Distributed Systems Security Symp. (NDSS), pp. 29-43, 2005.
[7] R. Lu, X. Lin, X. Liang, and X. Shen, "Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing," Proc. ACM Symp. Information, Computer and Comm.Security, pp. 282-292, 2010.
[8] B. Waters, "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization," Proc. Int'l Conf. Practice and Theory in Public Key Cryptography Conf. Public Key Cryptography, http://eprint.iacr.org/2008/290.pdf, 2008.
[9] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," Proc. ACM Conf. Computer and Comm. Security (CCS), pp. 89-98, 2006.
[10] D. Naor, M. Naor, and J.B. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," Proc. Ann. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO), pp. 41-62, 2001.
[11] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," Proc. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO), pp. 213-229, 2001.
[12] D. Boneh, X. Boyen, and H. Shacham, "Short Group Signature," Proc. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO), pp. 41-55, 2004.
[13] D. Boneh, X. Boyen, and E. Goh, "Hierarchical Identity Based Encryption with Constant Size Ciphertext," Proc. Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT), pp. 440-456, 2005.
[14] C. Delerablee, P. Paillier, and D. Pointcheval, "Fully Collusion Secure Dynamic Broadcast Encryption with Constant-Size Ciphertexts or Decryption Keys," Proc. First Int'l Conf. Pairing-Based Cryptography, pp. 39-59, 2007.

Author Profile

A. Jyothi MCA, M.Tech [Ph.D.] working as assistant professor in Computer Science Engineering from CVSR COLLEGE OF ENGINEERING from ANURAG GROUP OF INSTITUTIONS Venkatapur (V), Ghatkesar (M), Ranga Reddy District, Hyderabad-500088, Telangana State.

Deeti Naga Vijay received the B.Tech degree in Information Technology from JNTU hyderabad 2012 and pursuing M.Tech. degree in Computer science and Engineering from JNTU Hyderabad.