Indian Journal of Science and Technology, *Vol 8(15), IPL053, July 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Public Cloud Secure Group Sharing and Accessing in Cloud Computing

R. Megiba Jasmine* **and G. M. Nishibha**

CSE Department, Ponjesly College of Engineering, Nagercoil - 629 003, Tamil Nadu, India;
mejbhajas@gmail.com, gmnishibha144@gmail.com

Abstract

Cloud computing is an emerging computing paradigm in which resources of the computing infrastructure are provided as services over the Internet. Sharing data in a multi-owner manner while preserving data and identity privacy from an untrusted cloud is still a challenging issue, due to the frequent change of the membership. To preserve data privacy, a basic solution is to encrypt data files, and then upload the encrypted data into the cloud. In this paper we are further extending the basic MONA by adding the reliability as well as improving the scalability by increasing the number of group managers dynamically. This paper proposes how user can access data after the time out. The storage overhead and encryption computation cost of our scheme are independent with the number of revoked users.

Keywords: Cloud Computing, Data Sharing, Dynamic Groups, Integrity, Privacy-preserving, Reliability, Scalability

1. Introduction

In cloud computing, the Cloud Service Providers (CSPs), such as Amazon, are able 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 enjoy high-quality services and save significant investments on their local infrastructures. Cloud computing is one of the greatest platform which provides storage of data in very lower cost and available for all time over the internet Cloud computing is Internet-based computing, whereby shared resources, software and information are provided to computers and devices on demand. Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. Cloud Computing means more than simply saving on IT implementation costs. One of the most fundamental services offered by cloud providers is data storage. A company allows its staffs in the same group or department to store and share files in the cloud. By utilizing the cloud, the staffs can be completely released from the troublesome local data storage and maintenance. However, it also poses a significant risk to the confidentiality of those stored files. Cloud offers enormous opportunity for new innovation, and even disruption of entire industries. Cloud computing is the long dreamed vision of computing as a utility, where data owners can remotely store their data in the cloud to enjoy on demand high-quality applications and services from a shared pool of configurable computing resources. Identity privacy is one of the most significant obstacles for the wide deployment of cloud computing. Without the guarantee of identity privacy, users may be unwilling to join in cloud computing systems because their real identities could be easily disclosed to cloud providers and attackers. For example, a misbehaved staff can deceive others in the company by sharing false files without being traceable. Maintaining the integrity of data plays a vital role in the establishment of trust between data subject and service provider. Although envisioned as a promising service platform for the Internet, the new data storage paradigm in "Cloud" brings about many challenging

*Author for correspondence

design issues which have profound influence on the security and performance of the overall system. One of the biggest concerns with cloud data storage is that of data integrity verification at untrusted servers. What is more serious is that for saving money and storage space the service provider might neglect to keep or deliberately delete rarely accessed data files which belong to an ordinary client. CS2 provides security against the cloud provider, clients are still able not only to efficiently access their data through a search interface but also to add and delete files securely. Several security schemes for data sharing on untrusted servers have been proposed secure file system designed to be layered over insecure network and P2P file systems such as NFS, CIFS, Ocean Store, and Yahoo! Briefcase.

2. Existing System

In the literature study we have seen many methods for secure data sharing in cloud computing, however most methods failed to achieve the efficient as well as secure method for data sharing for groups. To provide the best solutions for the problems imposed by existing methods, recently the new method was presented called MONA[1]. This approach presents the design of secure data sharing scheme, Mona, for dynamic groups in an untrusted cloud. In Mona, a user is able to share data with others in the group without revealing identity privacy to the cloud. Additionally, Mona supports efficient user revocation and new user joining. More specially, efficient user revocation can be achieved through a public revocation list without updating the private keys of the remaining users, and new users can directly decrypt files stored in the cloud before their participation. Moreover, the storage overhead and the encryption computation cost are constant. Therefore

Figure 1. Existing system model.

practically in all cases MONA outperforms the existing methods.

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. 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 as follows: here x1=y1, x2=y2 and xr=yr.

$$\begin{cases} P_1 = \dfrac{1}{\gamma + x_1}.P \in G_1 \\ P_2 = \dfrac{1}{(\gamma + x_1)(\gamma + x_2)}.P \in G_1 \\ P_r = \dfrac{1}{(\gamma + x_1)(\gamma + x_2)\ldots(\gamma + x_r)}.P \in G_1 \\ Z_r = \dfrac{1}{Z(\gamma + x_1)(\gamma + x_2)\ldots(\gamma + x_r)} \in G_2 \end{cases}$$

Motivated by the verifiable reply mechanism in [13], 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. In other words, the others can verify the freshness of the revocation list from the contained current date tRL. In addition, the revocation list is bounded by a signature sig (RL) to declare its validity. The signature is generated by the group manager with the BLS signature algorithm[14]. Finally, the group manager migrates the revocation list into the cloud for public usage.

2.1 Disadvantage

However as per reliability and scalability concern this method needs to be workout further as if the group manager stop working due to large number of requests coming from different groups of owners, then entire security system of MONA failed down. In revocation list the time given for each user is fixed after time expire user cannot access the data until group manager update the revocation list and give it to the cloud.

3. Proposed System

To achieve the reliable and scalable in MONA, in this paper we are presenting the new framework for MONA.

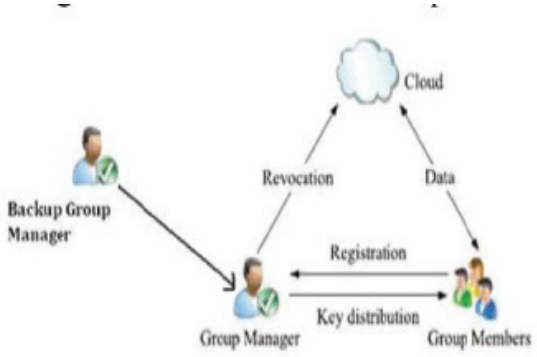

Figure 2. Proposed system model.

In this method we are further presenting how we are managing the risks like failure of group manager by increasing the number of backup group manager, hanging of group manager in case number of requests more by sharing the workload in multiple group managers. This method claims required efficiency, scalability and most importantly reliability.

3.1 Advantages

To overcome the disadvantage of existing system MONA, in the proposed MONA is if the group manager stop working due to large number of requests coming from different groups of owners, then backup group manager will remains available. Here user gets extra time for accessing data after the time out by sending request to the cloud.

3.2 Scheme Description

This section describes system, initialization, user registration, user revocation, file generation, file deletion and file access.

3.3 System Initialization

The group manager takes charge of system initialization as follows: Generating a bilinear map group system S= (q, G1, G2, e (.,.)). The system parameters including (S, P, H, H0, H1, H2, U, V, W, Y, Z, f, f1, Enc()), where f is a one-way hash function: {0,1}* —> Z*q ; f1 is hash function: {0,1}* —> G1; and Enck() is a secure symmetric encryption algorithm with secret key k.

3.4 User Registration

For the registration of user i with identity IDi, the group manager randomly selects a number xi belong to Z*q and computes Ai, Bi as the following equation:

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

Then, the group manager adds (Ai, xi, IDi) into the group user list, which will be used in the traceability phase. After the registration, user i obtains a private key (xi, Ai, Bi), which will be used for group signature generation and file decryption.

3.5 Revocation List

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. The list is characterized by time stamp t1, t2,…tr. In the proposed system once the user time stamp over does not wait for the group manager to update the time stamp or revocation list here once the time over the user immediately send request for extra time for access the data to the cloud. Then the cloud will send that request to the group manager once the see it and give permission then the cloud will time to access the data but if the group manager did not give permission then the cloud will not give permission for access of the data.

3.6 File Generation

To store and share a data file in the cloud, a group member performs the following operations: Getting the revocation list from the cloud. In this step, the member sends the group identity IDgroup as a request to the cloud. Then, the cloud responds the revocation list RL to the member. Verifying the validity of the received revocation list. First, checking whether the marked date is fresh. Second, verifying the contained signature sig (RL) by the equation e(W, f1 (RL)) = e(P, sig(RL)). If the revocation list is invalid, the data owner stops this scheme. Encrypting the data file M. Selecting a random number T and computing fT. The hash value will be used for data file deletion

Table 1. Revocation list

IDgroup	D1	y1	t1	P1			
	D2	y2	t2	P2			
	.	.	.	.			
	Dr	yr	tr	Pr	Wr	tRL	sig(RL)

operation. In addition, the data owner adds (IDdata, T) into his local storage. Constructing the uploaded data file as shown in Table 2, where t_{data} denotes the current time on the member, and a group signature on (ID_{data}, C1, C2, C, f(T); t_{data}) computed by the data owner through private key (A, x).

Uploading the data shown in Table 2 into the cloud server and adding the ID_{data} into the local shared data list maintained by the manager. On receiving the data, the cloud first check its validity. If the algorithm returns true, the group signature is valid; otherwise, the cloud abandons the data. In addition, if several users have been revoked by the group manager, the cloud also performs revocation verification. Finally, the data file will be stored in the cloud after successful group signature and revocation verifications.

3.7 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 and sends the signature along with IDdata to the cloud.

4. Related Work

E. Goh, H. Shacham, N. Modadugu, and D. Boneh[4] the use of SiRiUS is compelling in situations where users have no control over the file server (such as Yahoo! Briefcase or the P2P file storage provided by Farsite). They believe that SiRiUS is the most that can be done to secure an existing network file system without changing the file server or file system protocol. Key management and revocation is simple with minimal out-of-band communication. File system freshness guarantees are supported by SiRiUS using hash tree constructions. SiRiUS contains a novel method of performing file random access in a cryptographic file system without the use of a block server. Extensions to SiRiUS include large scale group sharing using the NNL key revocation construction. B. Wang, B. Li, and H. Li[5] in this paper, we propose Knox, a privacy-preserving auditing scheme for shared data with large groups in the cloud. They utilize group signatures to compute verification information on shared data, so that the TPA is able to audit the correctness of shared data, but cannot reveal the identity o f the signer on each block. With the group manager's private key, the original user can efficiently add new users to the group and disclose the identities of signers on all blocks. The efficiency of Knox is not affected by the number of users in the group. 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[2] the data centers hardware and software is what we will call a cloud. When a cloud is made available in a pay-as-you-go manner to the general public, they call it a public cloud; the service being sold is utility computing. They use the term private cloud to refer to internal data centers of a business or other organization, not made available to the general public, when they are large enough to benefit from the advantages of cloud computing that we discuss here. Thus, cloud computing is the sum of SaaS and utility computing, but does not include small or medium-sized data centers, even if these rely on virtualization for management. People can be users or providers of SaaS, or users or providers of utility computing. They focus on SaaS providers (cloud users) cloud providers, which have received less attention than SaaS users. S. Kamara and K. Lauter[3] in this paper consider the problem of building a secure cloud storage service on top of a public cloud infrastructure where the service provider is not completely trusted by the customer. They describe, at a high level, several architectures that combine recent and non-standard cryptographic primitives in order to achieve our goal. 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. A. Fiat and M. Naor[6] they introduce new theoretical measures for the qualitative and quantitative assessment of encryption schemes designed for broadcast transmissions. The goal is to allow a central broadcast site to broadcast secure transmissions to an arbitrary set of recipients while minimizing key management related transmissions. They present several schemes that allow centers to broadcast a secret to any subset of privileged users out of a universe of size so that coalitions of users not in the privileged set cannot learn the secret. V. Goyal, O. Pandey, A. Sahai, and B. Waters[7] they develop a new cryptosystem for One-grained sharing of encrypted data that call Key-Policy Attribute-Based Encryption (KP-ABE). In cryptosystem, cipher texts are

Table 2. Message format

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

labeled with sets of attributes and private keys are associated with access structures that control which cipher texts a user is able to decrypt. They demonstrate the applicability of our construction to sharing of audit-log information and broadcast encryption. Our construction supports delegation of private keys which subsumes Hierarchical Identity-Based Encryption (HIBE). The data owner uses a random key to encrypt a file, where the random key is further encrypted with a set of attributes using KP-ABE. Then, the group manager assigns an access structure and the corresponding secret key to authorized users, such that a user can only decrypt a cipher text if and only if the data file attributes satisfy the access structure. To achieve user revocation, the manager delegates' tasks of data file re-encryption and user secret key update to cloud servers. However, the single owner manner may hinder the implementation of applications with the scenario, where any member in a group should be allowed to store and share data files with others.

5. Performance Evaluation

In this section, we first analyze the storage cost of Mona, and then perform experiments to test its computation cost. Storage Without loss of generality, we set q=160 and the elements in G1 and G2 to be 161 and 1,024 bit, respectively. In addition, we assume the size of the data identity is 16 bits, which yield a group capacity of data files. Similarly, the size of user and group identity are also set as 16 bits. Group manager. In Mona, the master private key of the group manager $(G, \gamma, \xi_1, \xi_2) \in G_1 \times \overline{Z}_q^3$. Additionally, the user list and the shared data list should be stored at the group manager. Considering an actual system with 200 users and assuming that each user share 50 files in average, the total storage of the group manager is $(80.125+42.125*200+2*10{,}000)*10^{-3} \approx 28.5$ Kbytes, which is very acceptable. Group members. Essentially, each user in our scheme only needs to store its private key $(Ai, Bi, xi) \in G_1^Z \times Z_q$ which is about 60 bytes. It is worth noting that there is a tradeoff between the storage and the computation overhead. For example, the four pairing operations including (e(H, W), e(H, P), e(P, P), e(Ai, P)) $\in G_2^4$ can be pre-computed once and stored for the group signature generation and verification. Therefore, the total storage of each user is about 572 bytes. The extra storage overhead in the cloud. In Mona, the format of files stored in the cloud is shown in Table 2. Since C3 is the cipher text of the file under the symmetrical encryption, the extra storage overhead to store the file is about 248 bytes, which includes. $(ID_{group}, ID_{data}, C_1, C_2, C_3, f(\tau), t_{data}, \sigma)$

5.1 Simulation

The simulation consists of three components: client side, manager side as well as cloud side. Both client-side and manager-side processes are conducted on a laptop with Core 2 T7250 2.0 GHz, DDR2 800 2G, Ubuntu 10.04 X86. The cloud-side process is implemented on a machine that equipped with Core 2 i3-2350 2.3 GHz, DDR3 1066 2G, Ubuntu 12.04 X64. In the simulation, we choose an elliptic curve with 160-bit group order, which provides a competitive security level with 1,024-bit RSA.

5.2 Client Computation Cost

In Figure 3, we list the comparison on computation cost of clients for data generation operations between Mona and the way that directly using the original dynamic broadcast encryption. It is easily observed that the computation cost in Mona is irrelevant to the number of revoked users. On the contrary, the computation cost increases with the number of revoked users in ODBE. The reason is that the parameters (Pr, Zr) can be obtained from the revocation list without sacrificing the security in Mona, while several time-consuming operations including point multiplications in G1 and exponentiations in G2 have to be performed by clients to compute the parameters in ODBE. From Figure 3(a) and 3(b), we can find out that sharing a 10 Mbyte file and a 100-Mbyte one, cost a client about 0.2 and 1.4 seconds in our scheme, respectively,

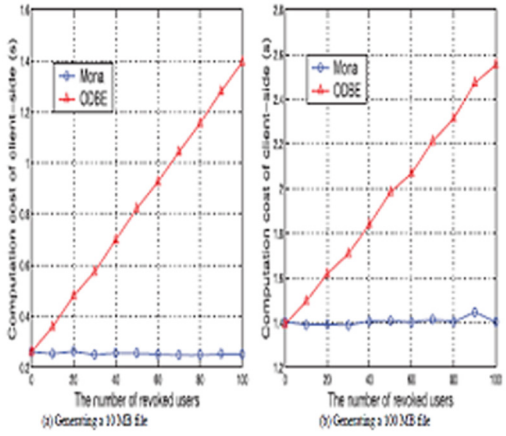

Figure 3. Comparison on computation cost for file generation between Mona and ODBE.

which implies that the symmetrical encryption operation domains the computation cost when the file is large. The computation cost of clients for file access operation with the size of 10 and 100 Mbytes are illustrated in Figure 4. The computation cost in Mona increases with the number of revoked users, Besides the above operations, P1, P2, …, Pr need to be computed by clients in ODBE.

Therefore, Mona is still superior than ODBE in terms of computation cost. Similar to the data generation operation, the total computation cost is mainly determined by the symmetrical decryption operation if the accessed file is large, which can be verified from Figure 4(a) and 4(b). In addition, the file deletion for clients is about 0.075 seconds, because it only costs a group signature on a message (IDdata, T) where T is a 160-bit number in Z*q.

6. Conclusion

In conclusion, cloud computing is very attractive environment for business world in term of providing required services in a very cost effective way. However, assuring and enhancing security and privacy practices will attract more enterprises to world of the cloud computing In Thus to achieve the reliable and scalable in MONA, in this paper we are presenting the new framework for MONA. In this method we are further presenting how we are managing the risks like failure of group manager by increasing the number of backup group manager, hanging of group manager in case number of requests more by sharing the workload in multiple group managers. This method claims required efficiency, scalability and most importantly reliability. Extensive analyses show that our proposed scheme satisfies the desired security requirements and guarantees efficiency as well. Here we also show that how user gets extra time even after the time out this also one of the advantage of proposed schema.

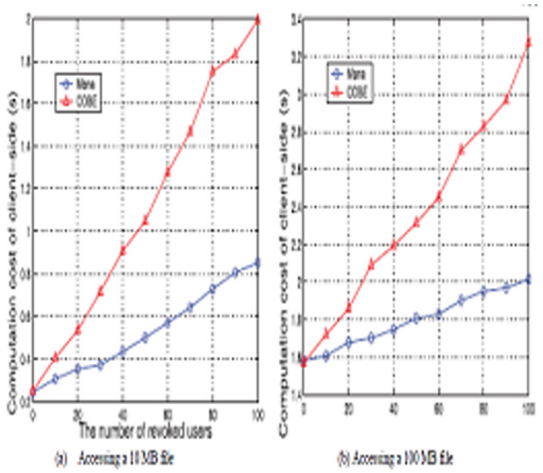

Figure 4. (a), (b) Comparison on computation cost for file access between Mona and ODBE.

7. References

1. Liu X, Zhang Y, Wang B, Yan J. Mona: secure multi-owner data sharing or dynamic groups in the cloud. IEEE Transactions on Parallel and Distributed Systems. 2013 Jun; 24(6):1182–91.
2. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. Comm ACM. 2010 Apr; 53(4):50–8.
3. Kamara S, Lauter K. Cryptographic cloud storage. Proceedings International Conference on Financial Cryptography and Data Security (FC); 2010 Jan. p. 136–49.
4. Goh E, Shacham H, Modadugu N, Boneh D. Sirius: securing remote untrusted storage; Proceedings of the Network and Distributed Systems Security Symposium (NDSS); 2003. p. 131–45.
5. Wang B, Li B, Li H. Knox: privacy-preserving auditing for shared data with large groups in the cloud. Proceedings of the 10th International Conference on Applied Cryptography and Network Security; 2012. p. 507–25.
6. Fiat A, Naor M. Broadcast Encryption. Proceedings of the International Cryptology Conference on Advances in Cryptology (CRYPTO); 1993. p. 480–91.
7. Goyal V, Pandey O, Sahai A, Waters B. Attribute-based encryption for fine-grained access control of encrypted data. Proceedings of the ACM Conference Computer and Communication Security (CCS); 2006. p. 89–98.
8. Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. J Cryptology. 2000; 13(3):361–96.
9. Lu R, Lin X, Liang X, Shen X. Secure provenance: the essential of bread and butter of data forensics in cloud computing. Proceedings of the ACM Symposium Information, Computer and Communication Security; 2010. p. 282–92.

10. Naor D, Naor M, Lotspiech JB. Revocation and tracing schemes for stateless receivers. Proceedings of Annual International Cryptology Conference Advances in Cryptology (CRYPTO); 2001. p. 41–62.
11. Boneh D, Franklin M. Identity-based encryption from the weil pairing. Proceedings of the International Cryptology Conference Advances in Cryptology (CRYPTO); 2001. p. 213–29.
12. Boneh D, Boyen X, Shacham H. Short group signature. Proceedings of the International Cryptology Conference Advances in Cryptology (CRYPTO); 2004. p. 41–55.
13. Sheng B, Li Q. Verifiable privacy-preserving range query in two-tiered sensor networks. Proceedings IEEE INFOCOM; 2008. p. 46–50.
14. Boneh D, Lynn B, Shacham H. Short Signature from the Weil Pairing. Proceedings International Conference on Theory and Application of Cryptology and Information Security: Advances in Cryptology; 2001. p. 514–32.