Robust And Auditable Access Control With Multiple Attribute For Public Cloud Storage

DR.CH.V. RAGHAVENDRAN[1] , PAILLA.NAVEEN KUMAR[2]

[1]Professor, [2]MTech Student, Department of CSE From Holy Mary Institute of Technology & Science Bogaram, Telangana State.

Abstract

Data access control is a challenging issue in public cloud storage systems. Ciphertext-Policy Attribute-Based Encryption (CP-ABE) has been adopted as a promising technique to provide flexible, fine-grained and secure data access control for cloud storage with honest-but-curious cloud servers. However, in the existing CP-ABE schemes, the single attribute authority must execute the time-consuming user legitimacy verification and secret key distribution, and hence it results in a single-point performance bottleneck when a CP-ABE scheme is adopted in a large-scale cloud storage system. Users may be stuck in the waiting queue for a long period to obtain their secret keys, thereby resulting in low-efficiency of the system. Although multiauthority access control schemes have been proposed, these schemes still cannot overcome the drawbacks of single-point bottleneck and low efficiency, due to the fact that each of the authorities still independently manages a disjoint attribute set. In this paper, we propose a novel heterogeneous framework to remove the problem of single-point performance bottleneck and provide a more efficient access control scheme with an auditing mechanism. Our framework employs multiple attribute authorities to share the load of user legitimacy verification. Meanwhile, in our scheme, a CA (Central Authority) is introduced to generate secret keys for legitimacy verified users. Unlike other multiauthority access control schemes, each of the authorities in our scheme manages the whole attribute set individually. To enhance security, we also propose an auditing mechanism to detect which AA (Attribute Authority) has incorrectly or maliciously performed the legitimacy verification procedure. Analysis shows that our system not only guarantees the security requirements but also makes great performance improvement on key generation.

Keywords: *Cloud Storage, Access control, Auditing, CPABE.*

1. INTRODUCTION

Cloud storage is a promising and important service paradigm in cloud computing. Benefits of using cloud storage include greater accessibility, higher reliability, rapid deployment and stronger protection, to name just a few [1]. Despite the mentioned benefits, this paradigm also brings forth new challenges on data access control, which is a critical issue to ensure data security. Since cloud storage is operated by cloud service providers, who are usually outside the trusted domain of data owners, the traditional access control methods in the Client/Server model are not suitable in cloud storage environment. The data access control in cloud storage environment has thus become a challenging issue. To address the issue of data access control in cloud storage, there have been quite a few schemes proposed, among which Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is regarded as one of the most promising techniques. [2] A salient feature of CP-ABE is that it grants data owners direct control power based on access policies, to provide flexible, finegrained and secure access control for cloud storage systems. In CP-ABE schemes, the access control is achieved by using cryptography, where an owner's data is encrypted with an access structure over attributes, and a user's secret key is labelled with his/her own attributes.

Only if the attributes associated with the user's secret key satisfy the access structure, can the user decrypt the corresponding ciphertext to obtain the plaintext. So far, the CP-ABE based access control schemes for cloud storage have been developed into two complementary categories, namely, single-authority scenario, and multiauthority scenario. Although existing CP-ABE access control schemes have a lot of attractive features, they are neither robust nor efficient in key generation. Since there is only one authority in charge of all attributes in single-authority schemes, offline/crash of this authority makes all secret key requests unavailable during that period. The similar problem exists in multi-authority schemes, since each of multiple authorities manages a disjoint attribute set. In single-authority schemes, the only authority must verify the legitimacy of users' attributes before generating secret keys for them. As the access control system is associated with data security, and the only credential a user possess is his/her secret key associated with his/her attributes, the process of key issuing must be cautious. However, in the real world, the attributes are diverse. For example, to verify whether a user is able to drive may need an authority to give him/her a test to prove that he/she can drive. [3] Thus he/she can get an attribute key associated with driving ability.

To deal with the verification of various attributes, the user may be required to be present to confirm them. Furthermore, the process to verify/assign attributes to users is usually difficult so that it normally employs administrators to manually handle the verification has mentioned, that the authenticity of registered data must be achieved by out-ofband (mostly manual) means [4]. To make a careful decision, the unavoidable participation of human beings makes the verification time consuming, which causes a single-point bottleneck. Especially, for a large system, there are always large numbers of users requesting secret keys. The inefficiency of the authority's service results in single-point performance bottleneck, which will cause system congestion such that users often cannot obtain their secret keys quickly, and have to wait in the system queue. This will significantly reduce the satisfaction of users experience to enjoy real-time services [5]. On the other hand, if there is only one authority that issues secret keys for some particular attributes, and if the verification enforces users' presence, it will bring about the other type of long service delay for users, since the authority maybe too far away from his/her home/workplace. As a result, single-point performance bottleneck problem affects the efficiency of secret key generation service and immensely degrades the utility of the existing schemes to conduct access control in large cloud storage systems.

Furthermore, in multi-authority schemes, the same problem also exists due to the fact that multiple authorities separately maintain disjoint attribute subsets and issue secret keys associated with users' attributes within their own administration domain. Each authority performs the verification and secret key generation as a whole in the secret key distribution process, just like what the single authority does in singleauthority schemes. Therefore, the single-point performance bottleneck still exists in such multi-authority schemes. A straightforward idea to remove the single-point bottleneck is to allow multiple authorities to jointly manage the universal attribute set, in such a way that each of them is able to distribute secret keys to users independently. By adopting multiple authorities to share the load, the influence of the single-point bottleneck can be reduced to a certain extent. However, this solution will bring forth threats on security issues. Since there are multiple functionally identical authorities performing the same procedure, it is hard to find the responsible authority if mistakes have been made or malicious behaviors have been implemented in the process of secret key generation and distribution. For example, an authority may falsely distribute secret keys beyond user's legitimate attribute set. Such weak point on security makes this straightforward idea hard to meet the security requirement of access control for public cloud storage.

Our recent work, TMACS, is a threshold multi-authority CP-ABE access control scheme for public cloud storage, where multiple authorities jointly manage a uniform attribute set. Actually it addresses the single-point bottleneck of performance and security, but introduces some additional overhead. Therefore, in this paper, we present a feasible solution which not only promotes efficiency and robustness, but also guarantees that the new solution is as secure as the original single-authority schemes. The similar problem has been considered and partly tackled in other related areas, such as public key infrastructure (PKI) for e-commerce. To reduce the certificate authority (CA)'s load, one or more registration authorities (RAs) are introduced to perform some of administration tasks on behalf of CA. Each RA is able to verify a user's legitimacy and determine whether the user is entitled to have a valid certificate. After the verification, it validates the credentials and forwards the certificate request to CA. Then, CA will generate a certificate for the user. Since the most heavy work of verification is performed by a selected RA, the load of CA can be largely reduced. However, the security of the scheme with single- CA/multi-RAs partly depends on the trustiness of multiple RAs. In order to achieve traceability, CA should store some information to confirm which RA has been responsible for verifying the legitimacy of a specific user [6].

In this paper, inspired by the heterogeneous architecture with single CA and multiple RAs, we propose a robust and auditable access control scheme (named RAAC) for public cloud storage to promote the performance while keeping the flexibility and fine granularity features of the existing CPABE schemes. In our scheme, we seperate the procedure of user legitimacy verification from the secret key generation, and assign these two sub-procedures to two different kinds of authorities. There are multiple authorities (named attribute authorities, AAs), each of which is in charge of the whole attribute set and can conduct user legitimacy verification independently. Meanwhile, there is only one global trusted authority (referred as Central Authority, CA) in charge of secret key generation and distribution. Before performing a secret key generation and distribution process, one of the AAs is selected to verify the legitimacy of the user's attributes and then it generates an intermediate key to send to CA. CA generates the secret key for the user on the basis of the received intermediate key, with no need of any more verification. In this way, multiple AAs can work in parallel to share the load of the time consuming legitimacy verification and standby for each other so as to remove the single-point bottleneck on performance.

Meanwhile, the selected AA doesn't take the responsibility of generating final secret keys to users. Instead, it generates intermediate keys that associate with users' attributes and implicitly associate with its own identity, and sends them to CA [7]. With the help of intermediate keys, CA is able to not only generate secret keys for legitimacy verified users more efficiently but also trace an AA's mistake or malicious behavior to enhance the security. The main contributions of this work can be summarized as follows.

- To address the single-point performance bottleneck of key distribution existed in the existing schemes, we propose a robust and efficient heterogeneous framework with single CA(Central Authority) and multiple AAs (Attribute Authorities) for public cloud storage. The heavy load of user legitimacy verification is shared by multiple AAs, each of which manages the universal attribute set and is able to independently complete the user legitimacy verification, while CA is only responsible for computational tasks. To the best of our knowledge, this is the first work that proposes the heterogeneous access control framework to address the low efficiency and single-point performance bottleneck for cloud storage [8].
- We reconstruct the CP-ABE scheme to fit our proposed framework and propose a robust and high-efficient access control scheme, meanwhile the scheme still preserves the fine granularity, flexibility and security features of CPABE.
- Our scheme includes an auditing mechanism that helps the system trace an AA's misbehavior on user's legitimacy verification.

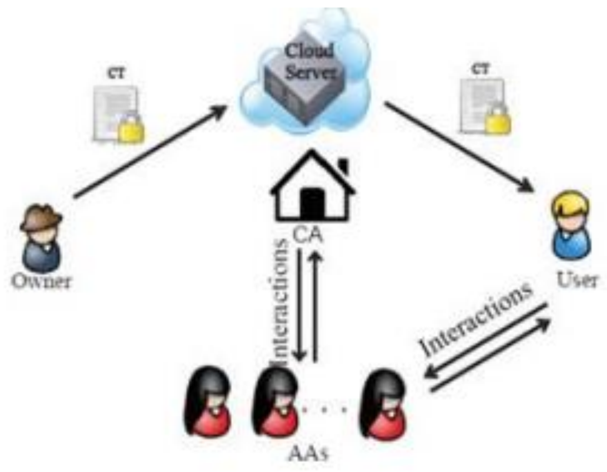

Fig.1. System Model.

2. SYSTEM MODEL AND SECURITY ASSUMPTIONS

We give the definitions of the system model, the security assumptions and requirements of our public cloud storage access control.

A. System Model

The system model of our design is shown in Fig. 1, which involves five entities: a central authority (CA), multiple attribute authorities (AAs), many data owners (Owners), many data consumers (Users), and a cloud service provider with multiple cloud servers(here, we mention it as cloud server.) [9].

- The central authority (CA) is the administrator of the entire system. It is responsible for the system construction by setting up the system parameters and generating public key for each attribute of the universal attribute set. In the system initialization phase, it assigns each user a unique Uid and each attribute authority a unique Aid. For a key request from a user, CA is responsible for generating secret keys for the user on the basis of the received intermediate key associated with the user's legitimate attributes verified by an AA. As an administrator of the entire system, CA has the capacity to trace which AA has incorrectly or maliciously verified a user and has granted illegitimate attribute sets.
- The attribute authorities (AAs) are responsible for performing user legitimacy verification and generating intermediate keys for legitimacy verified users. Unlike most of the existing multi-authority schemes where each AA manages a disjoint attribute set respectively, our proposed scheme involves multiple authorities to share the responsibility of user legitimacy verification and each AA can perform this process for any user independently [10]. When an AA is selected, it will verify the users' legitimate attributes by manual labor or authentication protocols, and generate an intermediate key

associated with the attributes that it has legitimacyverified. Intermediate key is a new concept to assist CA to generate keys.

- The data owner (Owner) defines the access policy about who can get access to each file, and encrypts the file under the defined policy. First of all, each owner encrypts his/her data with a symmetric encryption algorithm. Then, the owner formulates access policy over an attribute set and encrypts the symmetric key under the policy according to public keys obtained from CA. After that, the owner sends the whole encrypted data and the encrypted symmetric key (denoted as ciphertext CT) to the cloud server to be stored in the cloud.
- The data consumer (User) is assigned a global user identity Uid by CA. The user possesses a set of attributes and is equipped with a secret key associated with his/her attribute set. The user can freely get any interested encrypted data from the cloud server. However, the user can decrypt the encrypted data if and only if his/her attribute set satisfies the access policy embedded in the encrypted data.
- The cloud server provides a public platform for owners to store and share their encrypted data. The cloud server doesn't conduct data access control for owners. The encrypted data stored in the cloud server can be downloaded freely by any user.

B. Security Assumptions and Requirements

In our proposed scheme, the security assumptions of the five roles are given as follows. The cloud server is always online and managed by the cloud provider. Usually, the cloud server and its provider are assumed to be "honest-butcurious", which means that they will correctly execute the tasks assigned to them for profits, but they would try to find out as much secret information as possible based on data owners' inputs and uploaded files. CA is the administrator of the entire system, which is always online and can be assumed to be fully trusted. It will not collude with any entity to acquire data contents. AAs are responsible for conducting legitimacy verification of users and judging whether the users have the claimed attributes. We assume that AA can be compromised and cannot be fully trusted. Furthermore, since the user legitimacy verification is conducted by manual labor, mis-operation caused by carelessness may also happen. Thus, we need an auditing mechanism to trace an AA's misbehavior. Although a user can freely get any encrypted data from the cloud server, he/she cannot decrypt it unless the user has attributes satisfying the access policy embedded inside the data. Therefore, some users may be dishonest and curious, and may collude with each other to gain unauthorized access or try to collude with (or even compromise) any AA to obtain the access permission beyond their privileges. Owners have access control over their uploaded data, which are protected by specific access policies they defined.

To guarantee secure access control in public cloud storage, we claim that an access control scheme needs to meet the following four basic security requirements:

- Data confidentiality. Data content must be kept confidential to unauthorized users as well as the curious cloud server.
- Collusion-resistance. Malicious users colluding with each other would not be able to combine their attributes to decrypt a ciphertext which each of them cannot decrypt alone.
- AA accountability. An auditing mechanism must be devised to ensure that an AA's misbehavior can be detected to prevent AAs' abusing their power without being detected.
- No ultra vires for any AA. An AA should not have unauthorized power to directly generate secret keys for users. This security requirement is newly introduced based on our proposed hierarchical framework.

3. PROPOSED ACCESS CONTROL

This section first gives an overview of our proposed scheme, and then describes the scheme in detail. Our scheme consists of five phases, namely System Initialization, Encryption, Key Generation, Decryption, and Auditing & Tracing.

A. Overview of Our Scheme

To achieve a robust and efficient access control for public cloud storage, we propose a hierarchical framework with single CA and multiple AAs to remove the problem of single-point performance bottleneck and enhance the system efficiency [11]. In our proposed RAAC scheme, the procedure of key generation is divided into two sub-procedures: 1) the procedure of user legitimacy verification; 2) the procedure of secret key generation and distribution. The user legitimacy verification is assigned to multiple AAs, each of which takes responsibility for the universal attribute set and is able to verify all of the user's attributes independently. After the successful verification, this AA will generate an intermediate key and send it to CA. The procedure of secret key generation and distribution is executed by the CA that generates the secret key associated with user's attribute set without any more verification. The secret key is generated using the intermediate key securely transmitted from an AA and the master secret key. In our one-CA/multiple-AAs construction, CA participates in the key generation and distribution for security reasons: To enhance auditability of corrupted AAs, one AA cannot obtain the system's master secret key in case it can optionally generate secret keys without any supervision. Meanwhile, the introduction of CA for key generation and distribution is acceptable, since for a large-scale system, the most time consuming workload of legitimacy verification is offloaded and shared among the multiple AAs, and the computation workload for key generation is very light. The procedure of key generation and distribution would be more efficient than other existing schemes. To trace an AA's misbehavior in the procedure of user legitimacy verification, we first find the suspected data consumer based on abnormal behavior detection, which is similar to the mechanisms used. For a suspected user, our scheme can trace the responsible AA who has falsely verified this user's attributes and illegitimately assigned secret keys to him/her.

4. PERFORMANCE ANALYSIS

As we have mentioned, in reality, the tedious procedure of user legitimacy verification is much more complicated than secret key generation. In our scheme, the load of legitimacy verification is shared among multiple AAs, while a much lighter computational task is assigned to the single CA [12]. Thus, the efficiency of key distribution is improved. More Specifically, multiple AAs are standby for the legitimacy verification in the system. When there is a key request, an idle AA is selected by a scheduling algorithm to perform the verification and other AAs are standby to serve the subsequent user requests. We give the theoretical performance analysis as the following steps. Firstly, we model our system in queueing theory, and then we analyze the state probabilities to obtain the two important factors, the mean failure probability and the average waiting time for users. Finally, to show the significant performance improvement of our proposed RAAC, we compares it with single-AA system. It's important to note that, the comparison between RAAC and multi-authority systems is similar, since each authority independently manages a disjoint attribute subset. When a user requests secret keys with regard to one certain attribute subset, he/she has to go to the only and exclusive authority that issues secret keys with that attribute subset. The queue condition is just the same as the one in single-authority schemes.

A. Modeling in Queuing Theory

For simplicity, we assume there is a central coordinator which assigns users' key requests to AAs. The coordinator maintains each AA's state with the boolean value of 0/1, where state 0 indicates that the AA is available to conduct verification, and state 1 indicates the AA is occupied and is not available right now. Each time the coordinator assigns a key request to an AA with the state 0. If all AAs are busy, the new users who are requesting the secret keys will wait in a queue to be served. The coordinator can adopt First Come First Service (FCFS) algorithm to serve the arriving users. It's important to note that some other strategies can also be adopted in our architecture, such as a user arriving at a nearest AA according to his/her knowledge and decision. Thus, each AA may separately maintain a queue of its own. However, this model may not achieve load balance as some AAs may be unoccupied while other AAs are always busy in serving users' requests. Therefore, we introduce a central coordinator and adopt a single arrival queue as our strategy. The queueing model of our system can be treated as a Markov process. The central coordinator is deployed at the entrance of the system to monitor each AA's state (occupied/unoccupied) and assign each arriving users to an unoccupied AA. Furthermore, we model our system as follows. On AAs' side, the queueing model can be described as $M/M/C/N/\infty$, where C is the number of AAs, N is the capacity of our system and $N = C + K$ (K is the queue length that indicates the maximum number of the queued users.).

Here, the first M describes that arrivals of key requests follow a Poisson process in the system, and the second M means the verification service times are exponentially distributed. ∞ means the source of key requests is infinite. When there are N users in the system, other new arrivals of users' requests will be rejected. This property can ensure that a user will not wait in the queue for an irrationally long time. On CA's side, the queueing model can be described as M/M/1. The following assumptions are made to describe our system.

- **Assumption 1:** The instant user request arrival event constitutes a stationary Poisson process with the parameter λ.
- **Assumption 2:** For each AA, the service time of different individual users are independent and identically distributed exponential random variables, in which the mean value is $1/\mu 1$.
- **Assumption 3:** For CA, the service time of individual users are independent and identically distributed exponential random variables, in which the mean value is $1/\mu 2$.

5. CONCLUSION

We proposed a new framework, named RAAC, to eliminate the single-point performance bottleneck of the existing CP-ABE schemes. By effectively reformulating CPABE cryptographic technique into our novel framework, our proposed scheme provides a fine-grained, robust and efficient access control with one-CA/multi-AAs for public cloud storage. Our scheme employs multiple AAs to share the load of the time-consuming legitimacy verification and standby for serving new arrivals of users' requests. We also proposed an auditing method to trace an attribute authority's potential misbehavior. We conducted detailed security and performance analysis to verify that our scheme is secure and efficient. The security analysis shows that our scheme could effectively resist to individual and colluded malicious users, as well as the honest-but-curious cloud servers. Besides, with the proposed auditing & tracing scheme, no AA could deny its misbehaved key distribution. Further performance analysis based on queuing theory showed the superiority of our scheme over the traditional CP-ABE based access control schemes for public cloud storage.

Reference

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology Gaithersburg, 2011.

[2] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," IEEE Transactions on Parallel & Distributed Systems, vol. 27, no. 9, pp. 2546–2559, 2016.

[3] Z. Fu, X. Sun, S. Ji, and G. Xie, "Towards efficient content-aware search over encrypted outsourced data in cloud," in in Proceedings of 2016 IEEE Conference on Computer Communications (INFOCOM 2016). IEEE, 2016, pp. 1–9.

[4] K. Xue and P. Hong, "A dynamic secure group sharing framework in public cloud computing," IEEE Transactions on Cloud Computing, vol. 2, no. 4, pp. 459–470, 2014.

[5] Y. Wu, Z. Wei, and H. Deng, "Attribute-based access to scalable media in cloud-assisted content sharing," IEEE Transactions on Multimedia, vol. 15, no. 4, pp. 778–788, 2013.

[6] J. Hur, "Improving security and efficiency in attributebased data sharing," IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 10, pp. 2271– 2282, 2013.

[7] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 7, pp. 1214–1221, 2011.

[8] J. Hong, K. Xue, W. Li, and Y. Xue, "TAFC: Time and attribute factors combined access control on timesensitive data in public cloud," in Proceedings of 2015 IEEE Global Communications Conference (GLOBECOM 2015). IEEE, 2015, pp. 1–6.

[9] Y. Xue, J. Hong, W. Li, K. Xue, and P. Hong, "LABAC: A location-aware attribute-based access control scheme for cloud storage," in Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM 2016). IEEE, 2016, pp. 1–6.

[10] A. Lewko and B. Waters, "Decentralizing attributebased encryption," in Advances in Cryptology– EUROCRYPT 2011. Springer, 2011, pp. 568–588.

[11] K. Yang, X. Jia, K. Ren, and B. Zhang, "DAC-MACS: Effective data access control for multi-authority cloud storage systems," in Proceedings of 2013 IEEE Conference on Computer Communications (INFOCOM 2013). IEEE, 2013, pp. 2895–2903.

[12] J. Chen and H. Ma, "Efficient decentralized attribute based access control for cloud storage with user revocation," in Proceedings of 2014 IEEE International Conference on Communications (ICC 2014). IEEE, 2014, pp. 3782–3787.