IJSEAT International Journal of Science Engineering and Advance Technology

Anonymous Authentication of Data Stored in Clouds with Decentralized Security Mechanism

Chaparala Pushya[1], G Syam Prasad[2]

[1] M.Tech (CSE), Usha Rama College of Engineering and Technology, A.P., India.

[2]HOD of Computer Science and Engineering, Usha Rama College of Engineering and Technology, A.P., India.

***Abstract* — The term that flourished in the IT industry is the Cloud Computing. Inspired and Motivated by the tremendous growth and the huge success of cloud, we propose a new decentralized access control scheme for secure data storage in clouds that supports anonymous authentication. In the proposed scheme, the cloud verifies the authenticity of the series without knowing the user's identity before storing data. Our scheme also has the added feature of access control in which only valid users are able to decrypt the stored information. The scheme prevents replay attacks and supports creation, modification, and reading data stored in the cloud. We also address user revocation. Moreover, our authentication and access control scheme is decentralized and robust, unlike other access control schemes designed for clouds which are centralized. The communication, computation, and storage overheads are comparable to centralized approaches.**

***Keywords* — Access control, authentication, attribute-based signatures, attribute-based encryption, cloud storage**

I. INTRODUCTION

Cloud computing has been gaining a lot of attention and attracting the researches to delve into from both academic and industrial worlds. In cloud computing, users can outsource their computation and storage to servers (also called clouds) using Internet. This frees users from the hassles of maintaining resources on-site. Clouds can provide several types of services like applications (e.g., Google Apps, Microsoft online), infrastructures (e.g., Amazon's EC2, Eucalyptus, Nimbus), and platforms to help developers write applications (e.g., Amazon's S3, Windows Azure).

Much of the data stored in clouds is highly sensitive, for example, medical records and social networks. Security and privacy are, thus, very important issues in cloud computing. In one hand, the user should authenticate itself before initiating any transaction, and on the other hand, it must be ensured that the cloud does not tamper with the data that is outsourced. User privacy is also required so that the cloud or other users do not know the identity of the user. The cloud can hold the user accountable for the data it outsources, and likewise, the cloud is itself accountable for the services it provides. The validity of the user who stores the data is also verified. Apart from the technical solutions to ensure security and privacy, there is also a need for law enforcement.

Recently, Wang et al. [1] addressed secure and dependable cloud storage. Cloud servers prone to Byzantine failure, where a storage server can fail in arbitrary ways [1]. The cloud is also prone to data modification and server colluding attacks. In server colluding attack, the adversary can compromise storage servers, so that it can modify data files as long as they are internally consistent. To provide secure data storage, the data needs to be encrypted. However, the data is often modified and this dynamic property needs to be taken into account while designing efficient secure storage techniques.

Efficient search on encrypted data is also an important concern in clouds. The clouds should not know the query but should be able to return the records that satisfy the query. This is achieved by means of searchable encryption [3], [2]. The keywords are sent to the cloud encrypted, and the cloud returns the result without knowing the actual keyword for the search. The problem here is that the data records should have keywords associated with them to enable the search. The correct records are returned only when searched with the exact keywords.

Security and privacy protection in clouds are being explored by many researchers. Wang et al. [3] addressed storage security using Reed-Solomon erasure-correcting codes. Authentication of users using public key cryptographic techniques has been studied in [4]. Many homomorphic encryption techniques have been suggested to ensure that the cloud is not able to read the data while performing computations on them. Using homomorphic encryption, the cloud receives ciphertext of the data and performs computations on the ciphertext and returns the encoded value of the result. The user is able to decode the result, but the cloud does not know what data it has operated on. In such circumstances, it must be possible for the user to verify that the cloud returns correct results.

Considering the following situation: A law student, Alice, wants to send a series of reports about some malpractices by authorities of University X to all the professors of University X, research chairs of universities in the country, and students belonging to Law department in all universities in the province. She wants to remain anonymous while publishing all evidence of malpractice. She stores the information in the cloud. Access control is important in such case, so that only authorized users can access the data. It is also important to verify that the information comes from a reliable source. The problems of access control, authentication, and privacy protection should be solved simultaneously. We address this problem in its entirety in this paper.

Our Contributions

The main contributions of this paper are the following:

1. Distributed access control of data stored in cloud so that only authorized users with valid attributes can access them.
2. Authentication of users who store and modify their data on the cloud.
3. The identity of the user is protected from the cloud during authentication.
4. The architecture is decentralized, meaning that there can be several KDCs for key management.
5. The access control and authentication are both collusion resistant, meaning that no two users can collude and access data or authenticate themselves, if they are individually not authorized.
6. Revoked users cannot access data after they have been revoked.
7. The proposed scheme is resilient to replay attacks. A writer whose attributes and keys have been revoked cannot write back stale information.
8. The protocol supports multiple read and write on the data stored in the cloud.
9. The costs are comparable to the existing centralized approaches, and the expensive operations are mostly done by the cloud.

II. Problem Statement

In this section, we propose our privacy preserving authenticated access control scheme. According to our scheme a user can create a file and store it securely in the cloud. This scheme consists of use of the two protocols ABE and ABS, as discussed in Sections 3.4 and 3.5, respectively. We will first discuss our scheme in details and then provide a concrete example to demonstrate how it works. We refer to the Fig. 1. There are three users, a creator, a reader, and writer. Creator Alice receives a token _ from the trustee, who is assumed to be honest. A trustee can be someone like the federal government who manages social insurance numbers etc. On presenting her id (like health/social insurance number), the trustee gives her a token _. There are multiple KDCs (here 2), which can be scattered. For example, these can be servers in different parts of the world. A creator on presenting the token to one or more KDCs receives keys for encryption/decryption and signing. In the Fig. 1, SKs are secret keys given for decryption, Kx are keys for signing. The message MSG is encrypted under the access policy X. The access policy decides who can access the data stored in the cloud. The creator decides on a claim policy Y, to prove her authenticity and signs the message under this claim. The ciphertext C with signature is c, and is sent to the cloud. The cloud verifies the signature and stores the ciphertext C. When a reader wants to read, the cloud sends C. If the user has attributes matching with access policy, it can decrypt and get back original message. Write proceeds in the same way as file creation. By designating the verification process to the cloud, it relieves the individual users from time consuming verifications. When a reader wants to read some data stored in the cloud, it tries to decrypt it using the secret keys it receives from the KDCs. If it has enough attributes matching with the access policy, then it decrypts the information stored in the cloud.

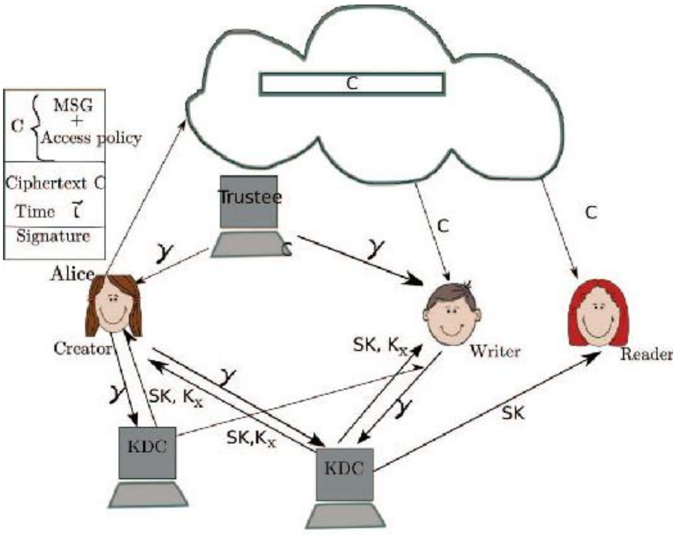

Figure 1: Our secure cloud storage model

Data Storage in Clouds

A user *Uu* first registers itself with one or more trustees. For simplicity we assume there is one trustee. The trustee gives it a token

$$\gamma = (u, K_{base}, K_0, \rho),$$

where p is the signature on ukKbase signed with the trustees private key TSig (by (6)). The KDCs are given keys PK[i], SK[i] for encryption/ decryption and ASK[i] ; APK[i] for signing / verifying. The user on presenting this token obtains attributes and secret keys from one or more KDCs.

The original work by Maji et al. [24] suffers from replay attacks. In their scheme, a writer can send its message and correct signature even when it no longer has access rights. In our scheme a writer whose rights have been revoked cannot create a new signature with new time stamp and, thus, cannot write back stale information.

Reading from the Cloud
When a user requests data from the cloud, the cloud sends the ciphertext C using SSH protocol. Decryption proceeds using algorithm ABE:Decrypt(C, $\{sk_{i,u}\}$) and the message MSG is calculated.

Writing to the Cloud
To write to an already existing file, the user must send its message with the claim policy as done during file creation. The cloud verifies the claim policy, and only if the user is authentic, is allowed to write on the file.

User Revocation
We have just discussed how to prevent replay attacks. We will now discuss how to handle user revocation. It should be ensured that users must not have the ability to access data, even if they possess matching set of attributes. For this reason, the owners should change the stored data and send updated information to other users.

The set of attributes I_u possessed by the revoked user U_u is noted and all users change their stored data that have attributes I € Iu. In [5], revocation involved changing the public and secret keys of the minimal set of attributes which are required to decrypt the data. We do not consider this approach because here different data are encrypted by the same set of attributes, so such a minimal set of attributes is different for different users. Therefore, this does not apply to our model. Once the attributes Iu are identified, all data that possess the attributes are collected. For each such data record, the following steps are then carried out:

REAL LIFE EXAMPLE

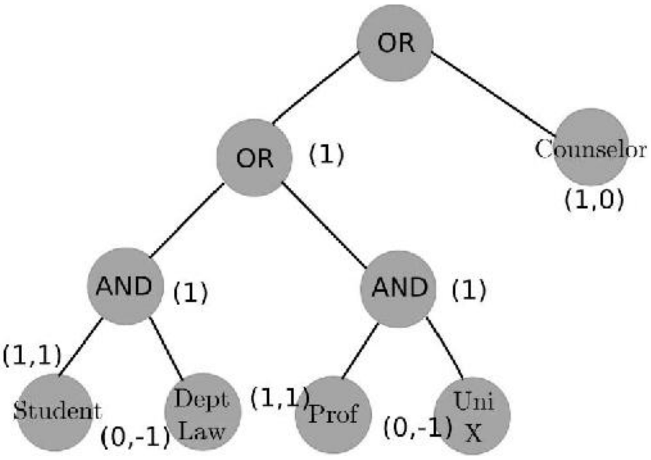

Figure 2: Example of Claim policy

We now revisit the problem we stated in the introduction. We will use a relaxed setting. Suppose Alice is a law student and wants to send a series of reports about malpractices by authorities of University X to all the professors of University X, Research chairs of universities X; Y ;Z and students belonging to Law department in university X. She wants to remain anonymous, while publishing all evidence. All information is stored in the cloud. It is important that users should not be able to know her identity, but must trust that the information is from a valid source. For this reason she also sends a claim message which states that she “Is a law student” or “Is a student counselor” or “Professor at university X.” The tree corresponding to the claim policy is shown in Fig. 2.

The leaves of the tree consists of attributes and the intermediary nodes consists of Boolean operators. In this example the attributes are “Student,” “Prof,” “Dept Law,” “Uni X,” “Counselor.” The above claim policy can be written as a Boolean function of attributes as

Later when a valid user, say Bob wants to modify any of these reports he also attaches a set of claims which the cloud verifies. For example, Bob is a research chair and might send a claim “Research chair” or “Department head” which is then verified by the cloud. It then sends the encrypted data to the Bob. Since Bob is a valid user and has matching attributes, he can decrypt and get back the information. If Bob wants to read the contents without modifying them, then there is no need to attach a claim. He will be able to decrypt only if he is a Professor in University X or a Research chair in one of the universities X; Y ;Z or a student belonging to Department of Law in university X.

Here it is to be noted that the attributes can belong to several KDCs. For example, the Professors belonging to university X have credentials given by the university X, and the Ph.D. degree from a University P, the student counselor might be a psychologist authorized by the Canadian Psychological Association and assigned an employee number by a university, the research chairs can be jointly appointed by the universities X, Y , Z and the government. The students can have credentials from the university and also a department.

Reading from the Cloud and Modifying Data

Suppose Bob wants to access the records stored by Alice. Bob then decrypts the message MSG using his secret keys using function ABE:$_{Decrypt}$. Writing proceeds like file creation. It is to be noted that the time is added to the data so that even if Bob's credentials are revoked, he cannot write stale data in the cloud.

III. Related Work

ABE was proposed by Sahai and Waters [6]. In ABE, a user has a set of attributes in addition to its unique ID. There are two classes of ABEs. In key-policy ABE or KP-ABE (Goyal et al. [7]), the sender has an access policy to encrypt data. A writer whose attributes and keys have been revoked cannot write back stale information. The receiver receives attributes and secret keys from the attribute authority and is able to decrypt information if it has matching attributes. In Ciphertext-policy, CP-ABE ([8], [9]), the receiver has the access policy in the form of a tree, with attributes as leaves and monotonic access structure with AND, OR and other threshold gates.

All the approaches take a centralized approach and allow only one KDC, which is a single point of failure. Chase proposed a multiauthority ABE, in which there are several KDC authorities (coordinated by a trusted authority) which distribute attributes and secret keys to users. Multiauthority ABE protocol was studied [10], which required no trusted authority which requires every user to have attributes from at all the KDCs. Recently, Lewko and Waters proposed a fully decentralized ABE where users could have zero or more attributes from each authority and did not require a trusted server. In all these cases, decryption at user's end is computation intensive. So, this technique might be inefficient when users access using their mobile devices. To get over this problem, Green et al. [11] proposed to outsource the decryption task to a proxy server, so that the user can compute with minimum resources (for example, hand held devices). However, the presence of one proxy and one KDC makes it less robust than decentralized approaches. Both these approaches had no way to authenticate users, anonymously. Yang et al. [34] presented a modification of [33], authenticate users, who want to remain anonymous while accessing the cloud.

To ensure anonymous user authentication ABSs were introduced by Maji et al. [12]. This was also a centralized approach. A recent scheme by Maji et al. takes a decentralized approach and provides authentication without disclosing the identity of the users. However, as mentioned earlier in the previous section it is prone to replay attack.

IV. Conclusion

We have presented a decentralized access control technique with anonymous authentication, which provides user revocation and prevents replay attacks. The cloud does not know the identity of the user who stores information, but only verifies the user's credentials. Key distribution is done in a decentralized way. One limitation is that the cloud knows the access policy for each record stored in the cloud. In future, we would like to hide the attributes and access policy of a user.

References

[1] S. Ruj, M. Stojmenovic, and A. Nayak, "Privacy Preserving Access
Control with Authentication for Securing Data in Clouds," Proc. IEEE/ACM Int'l Symp. Cluster, Cloud and Grid Computing, pp. 556- 563, 2012.

[2] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward Secure and Dependable Storage Services in Cloud Computing," IEEE Trans. Services Computing, vol. 5, no. 2, pp. 220-232, Apr.- June 2012.

[3] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy Keyword Search Over Encrypted Data in Cloud Computing," Proc. IEEE INFOCOM, pp. 441-445, 2010.

[4] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," Proc. 14th Int'l Conf. Financial Cryptography and Data Security, pp. 136- 149, 2010.

[5] H. Li, Y. Dai, L. Tian, and H. Yang, "Identity-Based Authentication for Cloud Computing," Proc.

First Int'l Conf. Cloud Computing (CloudCom), pp. 157-166, 2009.

[6] C. Gentry, "A Fully Homomorphic Encryption Scheme," PhD dissertation, Stanford Univ., http://www.crypto.stanford.edu/ craig, 2009.

[7] A.-R. Sadeghi, T. Schneider, and M. Winandy, "Token-Based Cloud Computing," Proc. Third Int'l Conf. Trust and Trustworthy Computing (TRUST), pp. 417-429, 2010.

[8] R.K.L. Ko, P. Jagadpramana, M. Mowbray, S. Pearson, M. Kirchberg, Q. Liang, and B.S. Lee, "Trustcloud: A Framework for Accountability and Trust in Cloud Computing," HP Technical Report HPL-2011-38, http://www.hpl.hp.com/techreports/ 2011/HPL-2011-38.html, 2013.

[9] R. Lu, X. Lin, X. Liang, and X. Shen, "Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing," Proc. Fifth ACM Symp. Information, Computer and Comm. Security (ASIACCS), pp. 282-292, 2010.

[10] D.F. Ferraiolo and D.R. Kuhn, "Role-Based Access Controls," Proc. 15th Nat'l Computer Security Conf., 1992.

[11] D.R. Kuhn, E.J. Coyne, and T.R. Weil, "Adding Attributes to Role-Based Access Control," IEEE Computer, vol. 43, no. 6, pp. 79-81, June 2010.

[12] M. Li, S. Yu, K. Ren, and W. Lou, "Securing Personal Health Records in Cloud Computing: Patient-Centric and Fine-Grained Data Access Control in Multi-Owner Settings," Proc. Sixth Int'l ICST Conf. Security and Privacy in Comm. Networks (SecureComm), pp. 89-106, 2010.