Journal for Studies in Management and Planning
Available at http://edupediapublications.org/journals/index.php/JSMaP/
Special Issue on Science and Technology
ISSN: 2395-0463
Volume 03 Issue 03
Special Issue

Secure Data Storage with Decentralized Access Control with Anomaly

1 Y.MAHESH 2 B.APARNA

Abstract--- 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.

1 INTRODUCTION

RESEARCH in cloud computing is receiving a lot of attention 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. [2] addressed secure and dependable cloud storage. Cloud servers prone to Byzantine failure, where a storage server can fail in arbitrary ways [2]. 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], [4]. 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. [2] addressed storage security using Reed-Solomon erasure-correcting

codes. Authentication of users using public key cryptographic techniques has been studied in [5]. Many homomorphic encryption techniques have been suggested [6], [7] 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.

2 RELATED WORK

ABE was proposed by Sahai and Waters . 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.), 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, 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 in and , 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. 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. presented a modification of, authenticate users, who want to remain anonymous while accessing the cloud. To ensure anonymous user authentication ABSs were introduced by Maji et al. . 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.

III BACKGROUND

In this section, we present our cloud storage model, adversary model and the assumptions we have made in the paper. presents the notations used throughout the paper. We also describe mathematical background used in our proposed solution.

3.1 Assumptions

We make the following assumptions in our work:

1. The cloud is honest-but-curious, which means that the cloud administrators can be interested in viewing user's content, but cannot modify it. This is a valid assumption that has been made in [12] and [13]. Honest-but-curious model of adversaries do not tamper with data so that they can keep the system functioning normally and remain undetected.

2. Users can have either read or write or both accesses to a file stored in the cloud.

3. All communications between users/clouds are secured by secure shell protocol, SSH.

3.2 Formats of Access Policies

Access policies can be in any of the following formats:

1) Boolean functions of attributes, 2) linear secret sharing scheme (LSSS) matrix, or 3) monotone span programs. Any access structure can be converted into a Boolean function

3.3 Mathematical Background

We will use bilinear pairings on elliptic curves. Let G be a cyclic group of prime order q generated by g. Let GT be a group of order q. We can define the map e : G _ G ! GT . Bilinear pairing on elliptic curves groups is used. We do not discuss the pairing functions which mainly use Weil and Tate pairings and computed using Miller's algorithm. The choice of curve is an important consideration because it determines the complexity of pairing operations.

3.4 Attribute-Based Encryption

ABE with multiple authorities as proposed by Lewko and Waters proceeds as follows [16]:

3.4.1 System Initialization

Select a prime q, generator g of G0, groups G0 and GT of order q, a map e : G0 _ G0 ! GT , and a hash function H : f0; 1g_ ! G0 that maps the identities of users to G0. The hash function used here is SHA-1. Each KDC Aj 2 A has a set of attributes Lj. The attributes disjoint (Li TLj ¼ _ for i 6¼ j).

IV PROPOSED PRIVACY PRESERVING AUTHENTICATED ACCESS CONTROL SCHEME

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. 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 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.

4.1 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 _ ¼ ðu;Kbase;K0; _Þ, where _ is the signature on ukKbase signed with the trustees private key TSig (by (6)). The KDCs are given keys PK½i_; SK½i_ for encryption/decryption and ASK½i_;APK½i_ for signing/verifying. The user on presenting this token obtains attributes and secret keys from one or more KDCs. A key for an attribute x belonging to KDC Ai is calculated as Kx ¼ K1=ðaþbxÞ base , where ða; bÞ 2 ASK½i_. The user also receives secret keys skx;u for encrypting messages. The user then creates an access policy X which is a monotone Boolean function. The message is then encrypted under the access policy as

The user also constructs a claim policy Y to enable the cloud to authenticate the user. The creator does not send the message MSG as is, but uses the time stamp and creates . This is done to prevent replay attacks. If the time stamp is not sent, then the user can write previous stale message back to the cloud with a valid signature, even when its claim policy and attributes have been revoked. The original work by Maji et al. 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. It then signs the message and calculates the message signature as ¼ ABS:SignðPublic key of trustee; Public key of KDCs; token; signing key; message; access claimÞ:

4.2 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:DecryptðC; fski;ugÞ and the message MSG is calculated..

4.3 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.

4.4 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 Iu possessed by the revoked user Uu is noted and all users change their stored data that have attributes i 2 Iu. In [13], 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.

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 ud 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.

[13] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute Based Data Sharing with Attribute Revocation," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS), pp. 261-270, 2010.

[14] G. Wang, Q. Liu, and J. Wu, "Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services," Proc. 17th ACM Conf. Computer and Comm. Security (CCS), pp. 735-737, 2010.

[15] F. Zhao, T. Nishide, and K. Sakurai, "Realizing Fine-Grained and Flexible Access Control to Outsourced Data with Attribute-Based Cryptosystems," Proc. Seventh Int'l Conf. Information Security Practice and Experience (ISPEC), pp. 83-97, 2011.

[16] S. Ruj, A. Nayak, and I. Stojmenovic, "DACC: Distributed Access Control in Clouds," Proc. IEEE 10th Int'l Conf. Trust, Security and Privacy in Computing and Communications (TrustCom), 2011.

[17] http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf, 2013.

[18] http://securesoftwaredev.com/2012/08/20/xacml-in-the-cloud, 2013.

[19] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-Based Access Control in Social Networks with Efficient Revocation," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS), 2011.

[20] R.L. Rivest, A. Shamir, and Y. Tauman, "How to Leak a Secret," Proc. Seventh Int'l Conf. Theory and Application of Cryptology and Information Security (ASIACRYPT), pp. 552-565, 2001.

AUTHOR'S PROFILE:

1.Y.MAHESH

Mahiyuvi44@gmail.com

2.B.APARNA

ASSISTANT PROFESSOR

aparnavsr@gmail.com