Volume 5, Issue 8, August 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Novel Anonymous Authentication Scheme Controlled By Decentralized Network for Cloud Data

D. Bullarao*, Venkata Sravani.M, P. Nageswara Rao
Dept of CSE & SITS, JNTUA,
Tirupati, India

Abstract*— We introduce a new novel secure anonymous Decentralized authentication scheme for data storage in clouds , that supports anonymous authentication. In the proposed scheme, the cloud verifies the authenticity of the ser 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, Error correcting and reading data stored in the cloud. We also certify the 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

A Research in cloud computing is receiving a lot of attention from both academic and industrial orlds. 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), infra- structures (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.

Cloud computing, the new term for the long dreamed vision of computing as a utility, enables convenient, on-demand network access to a centralized pool of configurable computing resource that can be rapidly deployed with great efficiency and minimal management overhead.

Another line of works tries to solve these problems by establishing trusted execution environments where the cloud client can verify the integrity of the software and the configuration of the cloud provider's hardware platform. This requires, however, secure software such as secure hypervisors for policy enforcement and attestation mechanisms for integrity verification. The use of trusted computing based remote attestation in the cloud scenario was recently discussed. Trusted Virtual Domains are one approach that combines trusted computing, secure hypervisors, and policy enforcement of information flow within and between domains of virtual machines. However, those approaches require trust in a non-negligible amount of hardware (e.g., CPU, Trusted Platform Module (TPM)) which are under the physical control of the cloud provider. According to the specification of the Trusted Computing Group, the TPM is not designed to protect against hardware attacks, but provides a shielded location to protect keys. However, the TPM cannot perform arbitrary secure computations on data. It can protect cryptographic keys and perform only pre-defined cryptographic operations like encryption, decryption, and signature creation. In particular, if data should be encrypted it must be provided in plaintext to the TPM, and if data should be decrypted it will be given in plaintext as output.

Accountability of clouds is a very challenging task and involves technical issues and law enforcement. Neither clouds nor users should deny any operations performed or requested. It is important to have log of the transactions performed; however, it is an important concern to decide how much information to keep in the log. Accountability has been addressed in TrustCloud [8]. Secure provenance has been studied in [9]. 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.

Efficient search 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. 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.

Access control in clouds is gaining attention because it is important that only authorized users have access to valid service. A huge amount of information is being stored in the cloud, and much of this is sensitive information. Care should be taken to ensure access control of this sensitive information which can often be related to health, important documents (as in Google Docs or Dropbox) or even personal information (as in social networking). There are broadly three types of access control: user-based access control (UBAC), role-based access control (RBAC), and attribute-based access control (ABAC). In UBAC, the access control list contains the list of users who are authorized to access data. This is not feasible in clouds where there are many users. In RBAC (introduced by Ferraiolo and Kuhn [10]), users are classified based on their individual roles. Data can be accessed by users who have matching roles. The roles are defined by the system. For example, only faculty members and senior secretaries might have access to data but not the junior secretaries. ABAC is more extended in scope, in which users are given attributes, and the data has attached access policy. Only users with valid set of attributes, satisfying the access policy, can access the data. For instance, in the above example certain records might be accessible by faculty members with more than 10 years of research experience or by senior secretaries with more than 8 years experience. The pros and cons of RBAC and ABAC are discussed in [11]. There has been some work on ABAC in clouds (for example, [12], [13], [14], [15], [16]). All these work use a cryptographic primitive known as attribute- based encryption (ABE). The eXtensible access control markup language [17] has been proposed for ABAC in clouds [18].

i. System Architecture:

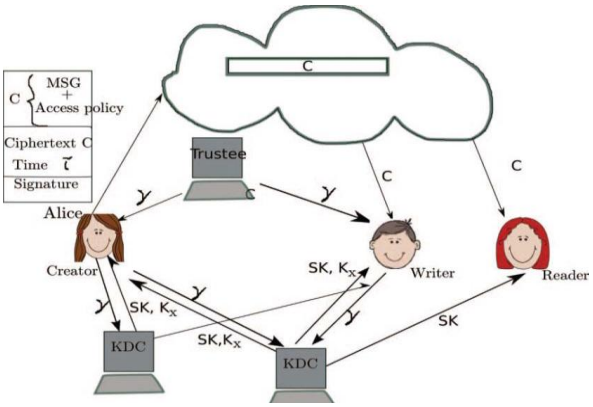

Fig.1 The Secure Proposed model

The architecture of proposed system depicted in 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 cipher text C with signature is c, and is sent to the cloud. The cloud verifies the signature and stores the cipher text 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.

Access control is also gaining importance in online social networking where users (members) store their personal information, pictures, videos and share them with selected groups of users or communities they belong to. Access control in online social networking has been studied in [19]. Such data are being stored in clouds. It is very important that only the authorized users are given access to those information. A similar situation arises when data is stored in clouds, for example, in Dropbox, and shared with certain groups of people.

A new protocol known as attribute-based signature (ABS) has been applied. ABS was proposed by Maji. In ABS, users have a claim predicate associated with a message. The claim predicate helps to identify the user as an authorized one, without revealing its identity. Other users or the cloud can verify the user and the validity of the message stored. ABS can be combined with ABE to achieve authenticated access control without disclosing the identity of the user to the cloud.

Existing work [12], [13], [14], 15], [16], [18], on access control in cloud are centralized in nature. Except [18], all other schemes use ABE. The scheme uses a symmetric key approach and does not support authentica- tion. The

schemes [12], [13], [16] do not support authentica- tion as well. Earlier work by Zhao et al. [15] provides privacy preserving authenticated access control in cloud. However, the authors take a centralized approach where a single key distribution center (KDC) distributes secret keys and attributes to all users. Unfortunately, a single KDC is not only a single point of failure but difficult to maintain because of the large number of users that are supported in a cloud environment.

II. LITERATURE SURVEY

ABE was proposed by Sahai and Waters [17]. 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. [18]), 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 ([19], [20]), 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 [21] 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 [22], 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.

Access control in clouds is gaining consideration on the grounds that it is imperative that just authorized clients have access to services. Just when the clients have matching set of attributes, would they be able to decrypt the data saved in the cloud. [5][6] Studied the access control in health care. Access control is likewise gaining imperativeness in online social networking where users store their personal data, pictures, films and shares them with selected group of users they belong. Access control in online social networking has been studied in [7].

The work done by [8] gives privacy preserving authenticated access control in cloud. Nonetheless, the researchers take a centralized methodology where a single key distribution center (KDC) disperses secret keys and attributes to all clients. Multi-authority ABE principle was concentrated on in [10], which obliged no trusted power which requires each client to have characteristics from at all the KDCs.

In spite of the fact that Yang et al. [11] proposed a decentralized approach, their strategy does not confirm clients, who need to remain anonymous while accessing the cloud. Ruj et al. [12] proposed a distributed access control module in clouds. On the other hand, the approach did not provide client verification. The other weakness was that a client can make and store an record and different clients can just read the record. write access was not allowed to clients other than the originator.

Time-based file assured deletion, which is initially presented in [13], implies that records could be safely erased and remain forever difficult to reach after a predefined time. The primary thought is that a record is encrypted with an information key by the possessor of the record, and this information key is further encrypted with a control key by a separate key Manager.

III. PROPOSED 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, 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.

A. 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 = (u,Kbase,K0), where 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. A key for an attribute x

belonging to KDC Ai is calculated as Kx=K1=ðaþbxÞbase , where (a, b) € ASK[i]. 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

C=ABE.Encrypt(MSG,X)

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 H(C)||k. 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 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

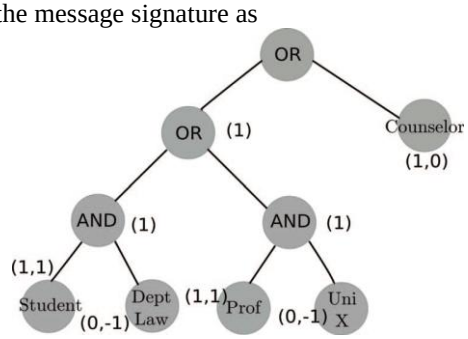

Fig: Example of claim policy

B. 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.

C. 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.

D. Distributed Key Policy Attribute Based Encryption

KP-ABE is a public key cryptography primitive for one-to-many correspondences. In KP-ABE, information is associated with attributes for each of which a public key part is characterized. The encryptor associates the set of attributes to the message by scrambling it with the comparing public key parts. Every client is assigned an access structure which is normally characterized as an access tree over information attributes, i.e., inside hubs of the access tree are limit doors and leaf hubs are connected with attributes. Client secret key is characterized to reflect the access structure so the client has the ability to decode a cipher-text if and just if the information attributes fulfill his access structure. The proposed scheme consists of four algorithms which is defined as follows

Setup:

This algorithm takes as input security parameters and attribute universe of cardinality N. It then defines a bilinear group of prime number. It returns a public key and the master key which is kept secret by the authority party.

*Encryption***:**

It takes a message, public key and set of attributes. It outputs a cipher text.

Key Generation:

It takes as input an access tree, master key and public key. It outputs user secret key.

Decryption:

It takes as input cipher text, user secret key and public key. It first computes a key for each leaf node. Then it aggregates the results using polynomial interpolation technique and returns the message.

E. File Assured Deletion

The policy of a file may be denied under the request by the customer, when terminating the time of the agreement or totally move the files starting with one cloud then onto the next cloud nature's domain. The point when any of the above criteria exists the policy will be repudiated and the key director will totally evacuates the public key of the associated file. So no one can recover the control key of a repudiated file in future. For this reason we can say the file is certainly erased. To recover the file, the user must ask for the key supervisor to produce the public key. For that the user must be verified. The key policy attribute based encryption standard is utilized for file access which is verified by means of an attribute

connected with the file. With file access control the file downloaded from the cloud will be in the arrangement of read just or write underpinned. Every client has connected with approaches for each one file. So the right client will access the right file. For making file access the key policy attribute based encryption.

1. *System Initialization*

Select a prime q, and groups G1 and G2, which are of order q. We define the mapping ˆe : G1 ×G1 → G2. Let g1, g2 be generators of G1 and hj be generators of G2, for j ∈ [tmax], for arbitrary tmax. Let H be a hash function. Let A0 = ha0 0 , where a0 ∈ Z∗ q is chosen at random. (TSig,TV er) mean TSig is the private key with which a message is signed and TV er is the public key used for verification. The secret key for the trustee is TSK = (a0, TSig) and public key is TPK = (G1,G2,H, g1,A0, h0, h1, . . . , htmax, g2, TV er).

2. *User Registration*

For a user with identity Uu the KDC draws at random Kbase ∈ G. Let K0 = K1/a0 base . The following token γ is output γ = (u,Kbase,K0, ρ), where ρ is signature on u||Kbase using the signing key TSig.

3. *KDC Setup*

We emphasize that clouds should take a decentralized approach while distributing secret keys and attributes to users. It is also quite natural for clouds to have many KDCs in different locations in the world. The architecture is decentralized, meaning that there can be several KDCs for key management.

4. *Attribute generation*

The token verification algorithm verifies the signature contained in γ using the signature verification key TV er in TPK. This algorithm extracts Kbase from γ using (a, b) from ASK[i] and computes Kx = K1/(a+bx) base , x ∈ J[i, u]. The key Kx can be checked for consistency using algorithm ABS.KeyCheck(TPK,APK[i], γ,Kx), which checks ˆe(Kx,AijBx ij) = ˆe(Kbase, hj), for all x ∈ J[i, u] and j ∈ [tmax].

5. *Sign*

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.

6. *Verify*

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.

TABLE-1 COMPARATIVE STUDY ON EXISTING VS. PROPOSED SYSTEM

S.NO	TECHNIQUE	EXISTING	PROPOSED
1	Approach	Centralized	Decentralized
2	Key Encryption	Use ABE(Attribute Based Encryption) For secret key	Use KDC(Key Distribution Center) for Key Encryption
3	Authentication	Does Not Provide Authentication	Authenticate the validity of the message without revealing the identity of the user who has stored information in the cloud.
4	Type Of Key	symmetric key approach	Public key Approach
	Attack Model	Resistant to replay attacks	Resistant to collusion attacks

IV. SECURITY OF THE PROTOCOL

Theorem 1. Our access control scheme is secure (no outsider or cloud can decrypt ciphertexts), collusion resistant and allows access only to authorized users.

Proof. We first show that no unauthorized user can access data from the cloud. We will first prove the validity of our scheme. A user can decrypt data if and only if it has a matching set of attributes. This follows from the fact that access structure S (and hence matrix R) is constructed if and only if there exists a set of rows X0 in R, and linear constants.

We next observe that the cloud cannot decode stored data. This is because it does not possess the secret keys ski;u (by (3)). Even if it colludes with other users, it cannot decrypt data which the users cannot themselves decrypt, because of the above reason (same as collusion of users). The KDCs are located in different servers and are not owned by the cloud. For this reason, even if some (but not all) KDCs are compromised, the cloud cannot decode data.

Theorem 2. Our authentication scheme is correct, collusion secure, resistant to replay attacks, and protects privacy of the user.

Proof. We first note that only valid users registered with the trustee(s) receive attributes and keys from the KDCs. A user's token is Kbase;K0 where is signature on ukKbase with TSig belonging to the trustee. An invalid user with a different user-id cannot create the same signature because it does not know TSig.

V. 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.

REFERENCES

[1] S. Ruj, M. Stojmenovic and A. Nayak, "Privacy Preserving Access Control with Authentication for Securing Data in Clouds", *IEEE/ACM International Symposium on 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 T. Services Computing*, vol. 5, no. 2, pp. 220–232, 2012.

[3] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *IEEE INFOCOM.* , pp.441–445, 2010.

[4] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Financial Cryptography Workshops*, ser. Lecture Notes in Computer Science, vol. 6054. Springer, pp. 136–149, 2010.

[5] H. Li, Y. Dai, L. Tian, and H. Yang, "Identity-based authentication for cloud computing," in *CloudCom*, ser. Lecture Notes in Computer Science, vol. 5931. Springer, pp. 157–166, 2009.

[6] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009, http://www.crypto.stanford.edu/craig.

[7] A.-R. Sadeghi, T. Schneider, and M. Winandy, "Token-based cloud computing," in *TRUST*, ser. Lecture Notes in Computer Science, vol. 6101. Springer, 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. Available at http://www.hpl.hp.com/techreports/2011/HPL-2011-38.html.

[9] R. Lu, X. Lin, X. Liang, and X. Shen, "Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing," in *ACM ASIACCS*, pp. 282–292, 2010.

[10] D. F. Ferraiolo and D. R. Kuhn, "Role-based access controls," in *15th National Computer Security Conference*, 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, 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 multiowner settings," in *SecureComm*, pp. 89–106, 2010.

[13] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *ACM 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," in *ACM 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," in *ISPEC*, ser. Lecture Notes in Computer Science, vol. 6672. Springer, pp.83–97, 2011.

[16] S. Ruj, A. Nayak, and I. Stojmenovic, "DACC: Distributed access control in clouds," in *IEEE TrustCom*, 2011.

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

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

[19] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-based access control in social networks with efficient revocation," in *ACM ASIACCS*, 2011.

[20] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *ASIACRYPT*, ser. Lecture Notes in Computer Science, vol. 2248. Springer, pp. 552–565, 2001.

[21] X. Boyen, "Mesh signatures," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 4515. Springer, pp. 210–227, 2007.

[22] D. Chaum and E. van Heyst, "Group signatures," in *EUROCRYPT*, pp. 257–265, 1991.

[23] H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures: Achieving attribute-privacy and collusion-resistance," *IACR Cryptology ePrint Archive*, 2008.

[24] "Attribute-based signatures," in *CT-RSA*, ser. Lecture Notes in Computer Science, vol. 6558. Springer, pp. 376–392, 2011.

[25] A. Beimel, "Secure Schemes for Secret Sharing and Key Distribution," PhD Thesis. Technion, Haifa, 1996.

[26] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 3494. Springer, pp. 457–473, 2005.

[27] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *ACM Conference on Computer and Communications Security*, pp. 89–98, 2006.

[28] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy.* , pp. 321–334, 2007.

[29] X. Liang, Z. Cao, H. Lin and D. Xing, "Provably Secure and Efficient Bounded Ciphertext Policy Attribute Based Encryption," in *ACM ASIACCS*, pp 343–352, 2009.

[30] M. Chase, "Multi-authority attribute based encryption," in *TCC*, ser. Lecture Notes in Computer Science, vol. 4392. Springer, pp. 515–534, 2007.

[31] H. Lin, Z. Cao, X. Liang and J. Shao, "Secure Threshold Multi-authority Attribute Based Encryption without a Central Authority," in *INDOCRYPT*, ser. Lecture Notes in Computer Science, vol. 5365, Springer, pp. 426–436, 2008.

[32] M. Chase and S. S. M. Chow, "Improving privacy and security in multiauthority attribute-based encryption," in *ACM Conference on Computer and Communications Security*, pp. 121–130, 2009.

[33] Matthew Green, Susan Hohenberger and Brent Waters, "Outsourcing the Decryption of ABE Ciphertexts," in USENIX Security Symposium, 2011.

[34] Kan Yang, Xiaohua Jia and Kui Ren, " DAC-MACS: Effective Data Access Control for Multi-Authority Cloud Storage Systems", IACR Cryptology ePrint Archive, 419, 2012.

[35] A. B. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 6632. Springer, pp. 568–588, 2011.

[36] "http://crypto.stanford.edu/pbc/."

[37] "libfenc: The functional encryption library. http://code.google.com/p/libfenc/."

[38] W. Wang, Z. Li, R. Owens, and B. Bhargava, "Secure and efficient access to outsourced data," in *ACM Cloud Computing Security Workshop (CCSW)*, 2009.

[39] J. Hur and D. Kun Noh, "Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems", IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 7, pp. 1214–1221, 2011.

