International Journal of Computing Science and Information Technology, **2013, Vol.3 Iss. 1,19-25**
ISSN: 2278-9669, Janaury 2015 (http://ijcsit.org)

Multi Attribute Based Security and Key Distribution for Secure Storage in Clouds

R.Vaishali [1], M.Menaka[2]

PG Scholar[1] , Assistant Professor CSE Department[2] ,Kingston engineering college, Vellore - 632 509 ,Tamilnadu ,India

rvaishu11@gmail.com, menakagce@gmail.com

Abstract

A new decentralized access control scheme is used for secure data storage in clouds that supports anonymous authentication. According to this scheme a user can create a file and store it securely in the cloud. Decryption will only work if the attributes associated with the decryption key match the policy used to encrypt the message. The cloud verifies the authenticity of the users without knowing the user's identity before storing data. This scheme also has the added feature of access control in which only valid users are able to decrypt the stored information. This scheme is resilient to replay attacks and supports creation, modification and reading data stored in the cloud. The proposed scheme is resilient to replay attacks. In this scheme using Secure Hash algorithm for authentication purpose, SHA is the one of several cryptographic hash functions, most often used to verify that a file has been unaltered. The Paillier cryptosystem is a probabilistic asymmetric algorithm for public key cryptography. Paillier algorithm is used for creation of access policy, file accessing and file restoring process.

Keywords: Access control, Authentication, Secure hash algorithm, paillier algorithm, replay attack.

1. Introduction

The mainstay of this is to propose a new decentralized access control scheme for secure data storage in clouds that supports anonymous authentication. The proposed scheme is resilient to replay attacks. A writer whose attributes and keys have been revoked cannot write back stale information. Distributed access control of data stored in cloud so that only authorized users with valid attributes can access them. Authentication of users is also verified who store and modify their data on the cloud. The identity of the user is protected from the cloud during authentication. The architecture is decentralized, meaning that there can be several KDCs for key management. 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. Revoked users cannot access data after they have been revoked. The proposed scheme is resilient to replay attacks. A writer whose attributes and keys have been revoked cannot write back stale information. The protocol supports multiple read and writes on the data stored in the cloud. The costs are comparable to the existing centralized approaches, and the expensive operations are mostly done by the cloud. Proposing 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. The cloud verifies the authenticity of the user without knowing the user's identity before storing data. The 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.

1.1 Organization

The paper is organized as follows: Related work is presented in Section 2. The Proposed work are detailed in Section 3.The mathematical background are detailed in Section 4.We present our privacy preserving access control scheme in Section 5. comparison with other work is presented in Section 6.We conclude in Section 7.

2. Related Work

ABE was proposed by Sahai and Waters [26]. 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.* [27]), 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 Cipher text-policy, CP-ABE ([28], [29]), 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 [30] proposed a multi-authority ABE, in which there are several KDC authorities (coordinated by a trusted authority) which distribute attributes and secret keys to users. Multi-authority ABE protocol was studied in [31], [32], which required no trusted authority which requires every user to have attributes from at all the KDCs. Recently, Lewko and Waters [35] 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. [33] 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 key distribution center 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 Attribute Based Signatures were introduced by Maji *et al.* [23]. This was also a centralized approach. A recent scheme by the same authors [24] 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.

3. Proposed Work

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. The identity of the user is protected from the cloud during authentication.
3. The architecture is decentralized, meaning that there can be several KDCs for key management.
4. 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.
5. Revoked users cannot access data after they have been revoked.
6. The proposed scheme is resilient to replay attacks. A writer whose attributes and keys have been revoked cannot write back stale information.
7. This protocol supports multiple read and writes on the data stored in the cloud.
8. The costs are comparable to the existing centralized approaches, and the expensive operations are mostly done by the cloud.

4. Mathematical Background

4.1. Attribute Based Encryption

ABE with multiple authorities as proposed by Lewko and Water proceeds as follows

4.1.1 System Initialization

1. Select a prime q, generator g of G_0, groups G_0 and G_T of order q, a map $e:G_0 x G_0 \rightarrow G_T$ and a hash function H.
2. Each KDC also chooses two random exponents α_i, y_i
3. The secret key of KDC Aj SK[j] is $\{\alpha_i, y_i, iL_j\}$
4. The public key of KDC Aj PK[j] is $\{e(g,g)^{\alpha i}, g^{yi}, iL_j\}$

4.1.2 Key Generation And Distribution by Kdcs

Each KDC has set of attributes I[j,u] and $sk_{i,u} = g^{\alpha i} H(u)^{yi}$

4.1.3 Encryption by Sender

The encryption function is ABE. Encrypt (MSG,χ).Sender decides about the access tree χ

1. choose a random seed s and random vector v
2. calculate $\lambda_x = R_x.v$
3. choose a random vector w_q^h
4. calculate $w_x = R_x.w$,for each Rx choose a $_{xq}$
5. $C_0 =$ MSG$e(g,g)^s$,$C_{1,x} = e(g,g)^{\lambda x}\ e(g,g)^{\alpha}{}_{\pi(x)}{}^{x}$, $C_{2,x} = g^{x,}$,$C_{3,x} = g^{y\pi(x)x}\ g^{w}{}_{x}$,
6. $C=[R, ,C_0, \{ C_{1,x}\ C_{2,x}\ C_{3,x,} \}]$

4.1.4 Decryption by Receiver

The decryption function is ABE. Decrypt(c, $\{sk_{i,u}\}$). Receive takes as input cipher text C, secret keys, group G0and outputs message.

1. calculates the set of attributes $\{(x) : x \in x\ i_u$
2. checks subset x^1 of rows r and (1,0,…0) if not decryption is impossible, if yes checks for constants $c_{x\ q}$
3. decryption proceeds as follows: $dec(x)=c_{1,x} e(h(u)c_{3,x})/\ e(sk_{\pi(x),u}, c_{2,x})$, $msg = c_0/\pi_{x\ x1} dec(x)$

4.2 Attribute Based Signature

4.2.1 System Initialization

Select a prime q and groups G_1,G_2 .Define the mapping e $G_1 * G_1 \rightarrow G_2$ .Let $\mathcal{H} \rightarrow$ hash function. Let $A_0 = h_0^{a0}$ and

(TSig, TVer) mean TSig is the private key with which a message is signed and TVer is the public key used for verification.

1. secret key for the trustee is TSK = $(a_0, TSig)$
2. public key is TPK $=(G_1, G_2, \mathcal{H}, g_1, A_0, h_0, h_1, \ldots h_{tmax}, g_2, TVer)$

4.2.2 User Registration

For a user with identity Uu the KDC draws at random $K_{base} \in G$. Let $K_0 = K_{base}^{1/a0}$. Token is generated as: $\gamma=(u, K_{base}, K_0, \rho)$

4.2.3 KDC Setup

Choose $a,b \in \mathbb{Z}_q^*$ randomly and compute $A_{ij} = h_j^a$ and $B_{ij} = h_j^b$

1. Private Key of KDC is ASK[i] = (a,b)
2. Public Key of KDC is APK[i] = $(A_{ij}, B_{ij} \mid j \quad [t_{max}])$

4.2.4 Attribute Generation

The token verification algorithm verifies the signature contained in γ using the signature verification key TVer in TPK. This algorithm extracts K_{base} from γ and computes $K_x = K_{base}^{1/(a+bx)}$. The key K_x can be checked for consistency using algorithm ABS.KeyCheck(TPK, APK[i], γ,k_x) which checks $e^\wedge(K_x, A_{ij}B_{ij}^{x}) = e^\wedge(K_{base}, h_j)$.

4.2.5 Sign

The algorithm ABS.Sign(TPK, {APK[i] : $i \in$ AT[u]}, γ, {K_x : $x \in J_u$ }, MSG,) has input the public key of the trustee, the secret key of the signer, the message to be signed and the policy claim y.

1. Compute μ = (MSG y)
2. Choose r_o and r_i then compute Y,W
3. $Y = K_{base}^{r0}, S_i = (k_i^{vi})^{r0}.(g_2 g_1^{\mu})^{ri}$ (Ju)
4. $W = K_0^{r0}$, $P_j = \Pi_{i \quad AT[u]} (A_{ij}B_{ij} \quad ^{\prime(i)})^{M}{}_{ij}{}^{r}{}_{i}$ (
5. Signature:$\sigma = (Y, W, S_1, S_2, \ldots, S_t, P_1, P_2, \ldots, P_t)$

4.2.6 Verify

This uses the algorithm ABS.Verify(TPK, $\sigma = (Y, W, S_1, S_2, \ldots, S_t, P_1, P_2, \ldots, P_t)$, MSG, y)

1. Compute μ = (MSG y)
2. If Y=1 → ABS.Verify = 0→ false
3. Orelse Check $e^\wedge(W, A_0) = e^\wedge (Y, h_0)$

4.3 Paillier Algorithm

4.3.1 Key generation

1. Choose two large prime number p and q randomly and independently of each other such that gcd (pq,(p-1)(q-1))=1
2. Compute n=pq and λ=lcm(p-1,q-1)
3. Select random integer g where g Z^*_{n2}
4. Compute $\mu= (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$
5. public (encryption) key is (n,g)
6. private (decryption) key is (λ,μ)

4.3.2 Encryption

Let m be a message to be encrypted where m belongs to Z^*_n.Select random where r $Z^*_{n..}$Compute cipher text as: $c= g^m .r^n \bmod n^2$

4.3.3 Decryption

Cipher text: c belongs to Z^*_{n2}.Compute message $m =L(c^{\lambda} \bmod n^2)$. μ mod n

4.4 Secure Hash Algorithm

SHA-1 is one of several cryptographic hash functions, most often used to verify that a file has been un-altered.SHA is short for Secure Hash Algorithm.

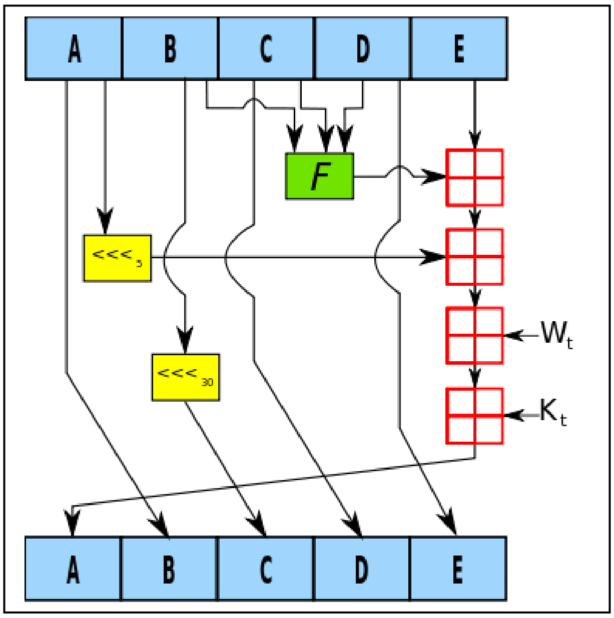

Fig 1 SHA Round Function

File verification using SHA-1 is accomplished by comparing the checksums created after running the algorithm on the two files need to be compared. SHA-1 is the second iteration of this cryptographic hash function, replacing the previous SHA-0. In the SHA-1 iteration A, B, C, D and E are 32bit words of the state. F is a nonlinear function that varies. n denotes a left bit rotation by n places. n varies for each operation. W_t is the expanded message word of round t. K_t is the round constant of round t. F denotes addition modulo 232.

5. Proposed Access Control Scheme

According to this scheme users can create a file and store it securely in the cloud. This scheme consists of use of the two protocols ABE and ABS. There are three users, a creator, a reader and writer. Creator Alice receives a token γ 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 γ. There are multiple KDCs, 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. 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. SKs are secret keys given for decryption, K_x are keys for signing. The message MSG is encrypted under the access policyχ. 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.

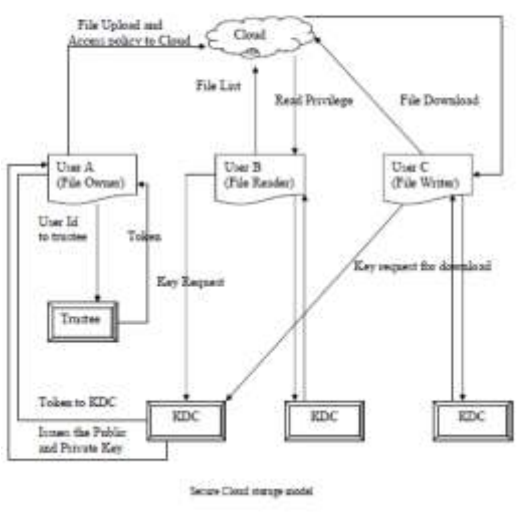

Figure 2 Secure Cloud Storage Model

5.1 Creation of KDC

Different number of KDC's are created to register the user details.KDC name, KDC id and KDC password are given as input to create each KDC. Inputs will be saved in the database and the new users are registered first in the KDC by providing details such as username, user id and type are given as inputs.The user will enrol the personal details to KDC. KDC will verify the user details and it will save it in the database.

5.2 KDC authentication

After KDC gives a user id to the user, the new user will enroll their personal details to the database by giving inputs such as user name, user id, password, university, type, etc. The key distribution center will verify the user details and if the user details are valid, their details will be stored in the database. The Key distribution center mainly verifies the user type and university name with its database to authenticate the users. Each key distribution center has a set of attributes $L_{j.}$

5.3 Trustee and user accessibility

Users receive 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 to the trustee, trustee gives a token. There are multiple KDCs, which can be scattered. Users on presenting the token to KDC receive keys for encryption/decryption and signing. SK are secret keys given for decryption, K_x are keys for signing.

User can login with their credentials and request the token from trustee for the file upload using the user id. After the user id is received by the trustee, trustee will create token using user id, key and user signature (SHA).Then the trustee will issue a token to the particular user and then trustee will be able to view the logs.

5.4 Data storage in clouds

User on receiving the token from the trustee presents the token to the KDC. Then the token is verified by the KDC if the user credentials are valid, KDC will provide the public and Private Key to the user. After users receive their keys, they can encrypt the files with the public keys and set their Access policies (privileges).The message is encrypted under the access policies. The access policies decide who can access the data stored in the cloud. 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. A user U_u first registers itself with one or more trustees. The trustee gives it a token $\gamma = (u, K_{base}, K_0, \rho)$ where ρ is the signature on $u \| K_{base}$ signed with the trustees private key TSig.

5.5. File accessing

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. When a user requests data from the cloud, the cloud sends the cipher text C. Decryption proceeds using algorithm 'ABE.Decrypt' and the message is obtained. Write proceeds in the same way as file creation. 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.

Table 1: Comparison of proposed scheme with existing access control scheme

Schemes	**Fi-ne-grained access control**	**Central-ized/Decentralized**	**Write/ read ac-cess**	**Type of access control**	**Privacy pre-serving authen-tication**	**User revocat-ion**
Secure and efficient access	Yes	Centralized	1-W-M-R	Symmet-ric key cryptog-raphy	No authentication	No
Fine grained access control	Yes	Centralized	1-W-M-R	ABE	No authentication	No
Attribute based data sharing	Yes	Centralized	1-W-M-R	ABE	No authentication	No
Outsourc-ing the decryption	Yes	Centralized	1-W-M-R	ABE	No authentication	No
proposed scheme	Yes	Decentralized	M-W-M-R	ABE	Authentication	Yes

Table 2: Comparison of computation and size of cipher text while creating a file

Schemes	**Computation by creator**	**Computation by cloud**	**Size of cipher text**
Fine grained data access control	$(m+2)E_0$	0	$m\log\lvert G_0\rvert + \lvert G_T\rvert + m \log m + \lvert MSG\rvert$
Attribute based data sharing	$(m+2)E_0$	0	$m\log\lvert G_0\rvert + \lvert G_1\rvert + \lvert MSG\rvert$
Proposed approach	$(3m + 1)E_0 + 2mE_T + \tau p$ (encrypt)$(2l + 2)E1 +$ $2tE2 + \tau\mathcal{H}$ (sign)	$2m\,\tau p + \tau\mathcal{H} +$ $O(mh)$(decrypt)	$2m\lvert G_0\rvert + m\lvert G_T\rvert + m^2 + \lvert MSG\rvert + (l+t+2)\lvert G_1\rvert$

Table 3: Comparison of computation during read and write by user and cloud

Schemes	**Computation by user while write**	**Computation by user while read**	**Computation by cloud while write**
Fine grained data access control	No write access	$m\,\tau p$	No write access
Attribute based data sharing	No write access	$m\,\tau p$	No write access
Proposed approach	$(3m + 1)E_0 + 2mE_T +$ τp (encrypt)$(2l + 2)E1 +$ $2tE2 + \tau\mathcal{H}$ (sign)	$2m\,\tau p + \tau\mathcal{H} + O(mh)$ (decrypt)	$(l + 2t)\,\tau p +$ $l(E_1 + E_2)$ $+ \tau\mathcal{H}$(verify)

6. Comparison with other Access Control Schemes

On comparing the proposed scheme with other access control schemes and it seems that proposed scheme supports many features that the other schemes did not support. 1-W-M-R means that only one user can write while many users can read. M-W-M-R means that many users can write and read. Most schemes do not support many writes which is supported by this scheme. This scheme is robust and decentralized; most of the others are centralized. This also supports privacy preserving authentication, which is not supported by others. Most of the schemes do not support user revocation, which proposed scheme does. On comparing the computation and communication costs incurred by the users and clouds and it seems that distributed approach has comparable costs to centralized approaches. The most expensive operations involving pairings and is done by the cloud. When comparing the computation load of user during read, the proposed scheme seems to have comparable costs. This scheme also compares well with the other authenticated scheme of fine grained access control.

The creator needs to encrypt the message and sign it. Creator needs to calculate one pairing e(g,g). Encryption takes two exponentiations to calculate each of $C_{1,x}$. So this requires $2mE_T$ time, where m is the number of attributes. User needs to calculate three exponentiation to calculate $C_{2,x}$ and $C_{3,x}$. So time taken for encryption is $(3m + 1)E_0 + 2mE_T + \tau p$ To sign the message, $Y,W,S_i{}'$s and P_js have to be calculated as well as $\mathcal{H}(C)$ So, time taken to sign is $(2l + 2)E1 + 2tE2 + \tau\mathcal{H}$.The cloud needs to verify the signature. Time taken to verify is $(l +2t) \tau p + l(E_1 + E_2) + \tau\mathcal{H}$. To read, a user needs only to decrypt the cipher text. Decryption takes $2m\tau p + \tau_H + O(mh)$. Writing is similar to creating a record. The size of cipher text with signature is $2m|G_0|+m|G_T| +m^2 +|MSG|+ (l+t+2) |G_1|$.

7. Conclusion

Thus the proposed decentralized access control technique supports 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. Moreover, this authentication and access control scheme is decentralized and robust, unlike other access control schemes designed for clouds which are centralized. One limitation is that the cloud knows the access policy for each record stored in the cloud. In future, SQL queries are used for hiding the attributes and access policy of a user. Files stored in cloud can be corrupted. So for this issue using the file recovery technique is used to recover the corrupted file and to hide the access policy and the user attributes.

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, "Trust cloud: A framework for accountability and trust in cloud computing," HP Technical Report HPL-2011-38.

[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," Ph D 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