Framework for Securing Data in Cloud Storage Services

Mai Dahshan and Sherif Elkassass
Department of Computer Science and Engineering, American University in Cairo, Cairo, Egypt

Keywords: Cloud Storage, Cloud Security, Data Confidentiality, Fine Grained Access Control, Trusted Third Party.

Abstract: Nowadays, users rely on cloud storage as it offers cheap and unlimited data storage that is available for use by multiple devices (e.g. smart phones, notebooks, etc.). Although these cloud storage services offer attractive features, many customers are not adopting them, since data stored in these services is under the control of service providers and this makes it more susceptible to security risks. Therefore, in this paper, we addressed the problem of ensuring data confidentiality against cloud and against accesses beyond authorized rights by designing a secure cloud storage system framework that simultaneously achieves data confidentiality and fine-grained access control on encrypted data. This framework is built on a trusted third party (TTP) service that can be employed either locally on users' machine or premises, or remotely on top of cloud storage services for ensuring data confidentiality. Furthermore, this service combines multi-authority ciphertext policy attribute-based encryption (MA-CP-ABE) and attribute-based Signature (ABS) for achieving many-read-many-write fine-grained data access control on storage services. Last but not least, we validate the effectiveness of our design by carrying out a security analysis.

1 INTRODUCTION

Cloud storage is a newly developed concept in the field of cloud computation. It allows users to outsource their data that has been managed internally within the organization or by individual users. The outsourcing of this data eliminates the concerns associated with the installation of the complex underlying hardware, saves increasing high cost in data management and alleviates the responsibilities of its maintenance. Although cloud storage providers(CSPs) often state that they offer safe environment for stored data, there have been cases discovered where users' data has been modified or lost due to some security breach or some human error. A study (CircleID Reporter, 2009) surveyed more than 500 CTO and IT managers in 17 countries, showed that despite the potential benefits of cloud storage, organizations and individuals do not trust the existing cloud storage service providers because the fear of the security threats associated with them. When individual users and organizations outsource their data to multi-tenant environment as the cloud, they expect to have the same level of data security as they would have in their own premises (Sosinsky, 2010); However, this not the case in cloud. Therefore, users cannot trust cloud for their data confidentially.

This idea of securing data in cloud storage services has attracted many researchers to work in this field with the aim of constructing a trusted control model of cloud storage. Most of the research done in this field has focused on providing efficient data access control mechanisms between data owners and data users and cloud storage. The data owners encrypt the data and enforce access control policies on it locally before uploading it to the cloud. After that, they provide decryption keys to users they want to share with and leave to cloud the task of managing the access control without have access to any keys. However, this model of access control is not feasible in cloud-based file sharing service where there is no direct interaction between the data owners and the data users(Yang et al.,2013).

2 PROBLEM STATEMENT

A recent security flaw in the Dropbox authentication mechanism (Newton, 2011) begins the debate about whether cloud storage services are sufficiently secure to storesensitive data or not.(Hu et al. ,2010) evaluated four cloud storage systems: Mozy, Carbonite, Dropbox, and CrashPlan. After the

Dahshan M. and Elkassas S..
Framework for Securing Data in Cloud Storage Services.
DOI: 10.5220/0005043802670274
In *Proceedings of the 11th International Conference on Security and Cryptography* (SECRYPT-2014), pages 267-274
ISBN: 978-989-758-045-1

evaluation, it was found out that none of these systems can provide any guarantees for data integrity, availability, or even confidentiality. Moreover, Amazon S3 encrypts user's data by an encryption key and an S3 master key and both keys are stored at Amazon's servers. Therefore, Amazon is able to arbitrarily decrypt data leading to data's confidentiality or integrity leakage.

Motivated by these limitations, in this paper we pose the following research question:

How to construct cryptographic scheme that can enforce data confidentiality and distributed data access control efficiently in dynamic environments?

In other words. the main objective of the work is to design cloud-based data sharing framework, in which outsourced data, access control policy and identity attributes of a user are considered as confidential information.

3 RELATED WORK

3.1 Data Confidentiality

Current cloud storage services try to secure user's data by encrypting them either on server side or client side. In server side encryption (e.g. Amazon S3) the data owner relies on the service for securing its data; however, this solution isn't feasible for two reasons. First, the user will send his plaintext to service which exposes it to internal attacks where the attacker can exploit vulnerabilities of servers to achieve user's data. Second, there is no guarantee that the service will encrypt the data before uploading it to the cloud (Chacos ,2012).

On the other hand, in client side encryption (e.g. Wuala and TrueCrypt), the service encrypts user's data locally before it is uploaded to the cloud. Although, these solutions appear ideal methods for securing users' data, they are not so. This is because the keys involved in the process of encryption are managed by software manner. Therefore, if the TCB of client is corrupted, the attacker will intercept keys. In addition, the encryption software's are complicated for end users to use and may end up with incorrect configuration (Deniability et al.,2010).Moreover, client side encryption by cloud storage services may expose user's data to key disclosure, manipulated file content and the most dangerous threat is the secret agent working at the provider. This agent may be able to manipulate the client software by injecting a malware in the customer's system (Borgmann et al., 2012).

3.2 Fine Grained Access Control

One of the most challenging issues in current cloud-based file sharing service is the enforcement of access control policies and the support of policies updates. The current deployment model of cloud storage services cannot be fully trusted by data owners; as a result, traditional server-based access control methods are no longer applicable to cloud storage systems.

To prevent the un-trusted servers from accessing sensitive data in a traditional server-based system, traditional methods usually encrypt files by using the symmetric encryption approach with content keys and then use every user's public key to encrypt the content keys and only users holding valid keys can access the data. These methods require complicated key management schemes and the data owners have to stay online all the time to deliver the keys to new user in the system. Moreover, these methods incur high storage overhead on the server for storing multiple encrypted copies of the same data for users with different keys (Goh et al., 2003).

Another prevalent methodology for enforcing access control policy, which is employed by most of the current CSP, is to provide the remote cloud server the power of key management and distribution under the assumption that the server is trusted or semi-trusted. However, the server cannot be trusted by the data owners in cloud storage systems and thus these methods cannot be applied to access control for cloud storage systems (Sahai et al., 2005).

Attribute-based encryption (ABE) (di Vimercati et al., 2007) is regarded as one of the most suitable technologies for realizing a fine-grained attribute-based access control mechanism. Since its introduction, two complementary schemes have been proposed, which are: key-policy ABE (KPABE) (Sahai et al., 2005) and ciphertext- policy ABE (CP-ABE) (Bethencourt et al., 2007). It is more convenient to use CP-ABE in the cloud environment, because the encryptor holds the ultimate authority about the encryption policy unlike KP-ABE scheme where the encryptor does not have entire control over the encryption policy because the encryption policy is described in the keys. Moreover, the access policy checking is implicitly conducted inside the cryptography. That is, there is no one to explicitly evaluate the policies and make decisions on whether allows the user to access the data (Waters et al., 2011).

Most of the ABE approaches take a centralized approach and allow only one single authority (Tang

et al.,2012)(Zhiquan et al.,2012) for issuing users' keys. However, this method may suffer from failure or corruption because the authority can decrypt all the encrypted data. Moreover, the authority may become the performance bottleneck in the large scale cloud storage systems. To address this issue, multi-authority attribute-based access control schemes were proposed, where multiple parties could play the role of an authority. Although, multi-authority ABE tries to solve the problem of single authority CP-ABE, it needs to tie together different components of a user's secret key from multiple authorities. (Chase M.,2007) (Muller et al.,2009) suggest using a central authority to provide a final secret key to integrate the secret keys from different attribute authorities. However, the central authority would be able to decrypt all the ciphertext because it holds the master key of the system. Thus, the central authority would be a vulnerable point for security attacks and a performance bottleneck for large scale systems. To overcome this problem, (Chase and Chow, 2009)(Jung et al.,2013) propose a multi-authority attribute-based access control schemes without a central authority. They presented secure multi-authority CP-ABE scheme that remove the central authority by using a distributed PRF (pseudo-random function). However, they have to define a pre-determined number of authorities at initialization, can tolerate collusion attacks for up to N-2 authorities' compromise, and degrade the performance of the system due to interaction among the authorities during the system setup. (Lewko A. and Waters B., 2011) proposed secure scheme secure against any collusion attacks and it can process the access policy expressed in any Boolean formula over attributes. However, this method is constructed in composite order bilinear groups that incur heavy computation cost. (Liu,2011) presents a fully secure multi-authority CP-ABE scheme in the standard model with multiple CAs and AAs. Each CA or AA operates independently from the others. Before requesting the attribute-related keys from the AAs, the user must ensure that he has obtained the identity-related keys from all the CAs leading to performance degradation. (Yang et al.,2013)(Yang and Jia, 2013), eliminates the collusion problem associated with previous work while maintaining high performance.

4 MODELS AND ASSUMPTIONS

4.1 System Model

Based on our preliminary work (Dahshan and Elkassass, 2014) and similar to (Yang and Jia, 2013) , the system model consists of six entities(as in Figure 1).

The cloud storage provider (CSP) is a semi-trusted entity. It is responsible for providing data storage service (i.e. Backend Storage Servers) and verification of users' data ciphertexts before it is stored in the cloud.

Trusted third party (TTP) service: is an independent entity that is trusted by all other system components, and has capabilities to perform extensive tasks (i.e. encryption, decryption and signature). It maintains a key management service that creates, manages, and destroys user's data files encryption and decryption keys (DEK).

Data owner encrypts his data with the help of the TTP service (which could be local or re-mote) by defining the access policies over attributes from multiple attribute authorities.

Each user has a global identity in the system. A user can be either a reader or a writer and a reader who may be entitled a set of attributes. we differentiate writer from reader not at the individual user level, but at the attribute level.

4.2 Security Model

- We consider the cloud service providers (CSPs) are honest but curious. CSP will try to learn information but will honestly follow any protocol provided by the Data Owner (DO).
- The certificate authority (CA) is fully trusted in the system. The CA is used to certify the attribute authorities and the users that want to join the system and provide global secret/public keys to both attribute authorities and the users respectively.
- We assume that legitimate users behave honestly, by which we mean that they never share their decryption key with the revoked users.
- All communications between users/clouds are secured by SSL/TLS protocol in order to secure the data in transit.

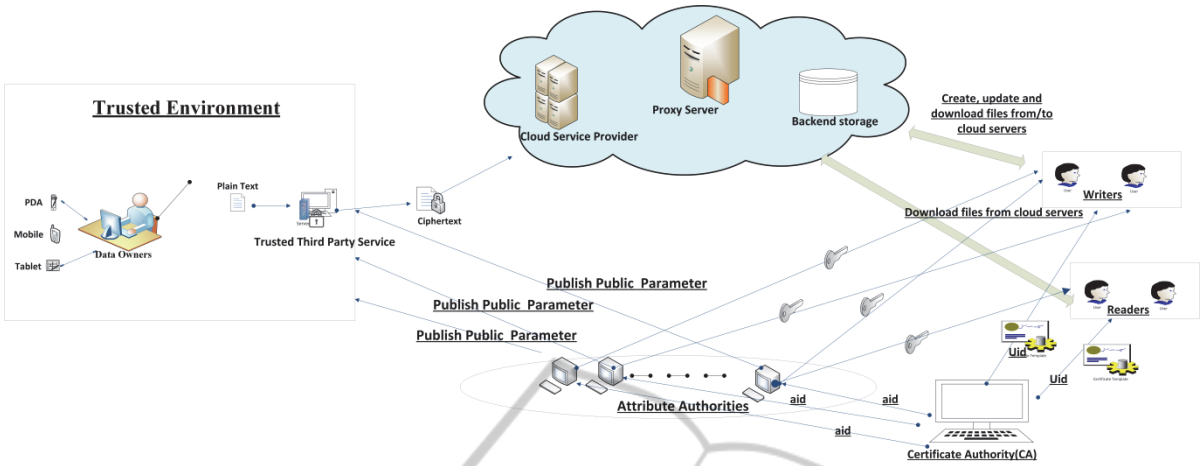

Figure 1: Secure Cloud Storage Service Design.

5 OUR PROPOSED SCHEME

5.1 Main Idea

In order to achieve secure fine-grained access control on outsourced data, we utilize the following cryptographic techniques: MA-CP-ABE, ABS(Cao et al., 2011). The proposed service transfers the trust from the cloud to the TTP service. It also provides security for users' data with minimal overhead on users. This TTP service has encryption/decryption service that can be employed either locally or on top of the cloud storage. Since users' data does not have the same level of importance, offering a flat encryption schemes without looking at the importance of data by applying the same encryption algorithm for all types of data expose the client machine to huge computation overhead(Patel et al.,2012).Therefore, the service offers different encryption algorithms according to data's severity. The TTP service does not store users' data; it only stores the encryption and decryption keys. These keys are critical components; therefore, they are stored separately using a Hardware Security Module or other secure elements(Shin et al.,2012). For achieving data confidentiality against unauthorized users, the TTP service collaborates with a number of attribute authorities to achieve fine grained access control. More specifically, the data owner associates each data file with a set of attributes, and assign each user (either a reader or a writer) an access structure which is defined over these attributes. To enforce this type of access control, we make use of MA-CP-ABE. By doing so, we prohibit the cloud and unauthorized users from getting access to owner's plaintext or credentials, unlike most of the currently available cloud storage services that either do not provide file sharing services or give the cloud provider full power over access control . Moreover, we provide read or write or both accesses to a file stored in the cloud by utilizing both MA-CP-ABE and ABS. Last but not least, we shift most of the heavy computations from the owner/user to the cloud. We believe that our proposed scheme combine different algorithms to form a larger and more generic solution that supports the needs of a cloud-based collaboration environment.

5.2 Scheme Description

We shall present the system level of following operations: File Creation, User Grant, and File Access.

5.2.1 New File Creation

The file creation process passes with two phases: Encrypt Phase and Sign Phase.

Encrypt Phase:

a) Data owner selects the file along with sensitivity level to be uploaded, defines a set of attributes I_u for read access policy ($\bar{\mathbf{A}}$) and a set of attributes J_u for write claim predicate ($\ddot{\mathbf{Y}}$).
b) It sends the file with its sensitivity level along with $\bar{\mathbf{A}}$ and $\ddot{\mathbf{Y}}$ to TTP service.
c) TTP service asks the different authorities for the related public/secret keys for $\bar{\mathbf{A}}$ and $\ddot{\mathbf{Y}}$ based on their attributes.
d) Each AA run **SKeyGen** algorithm and return related secret keys and public keys for both $\bar{\mathbf{A}}$ and $\ddot{\mathbf{Y}}$

e) TTP service generates a symmetric key according to the sensitivity level.
f) The TTP service encrypts data file (F) with symmetric key (DEK) and encrypts DEK with the different authorities' public keys $\{Pk_{aid}\}_{aid\in I_A}$, global public parameter(GPP) producing ciphertext CT.

CT ⟶ CP-ABE.Encrypt (GPP, $\{Pk_{aid}\}_{aid\in I_A}$, DEK , Ā)

Sign Phase

After the encryption, the TTP signs the CT both for reader/writer differentiation and for providing integrity verification to all parties that want to access the file.

a) TTP service first hashes the CT which is generated in the Encrypt Phase to produce (H(c)). A timestamp is attached with hash code to prevent replay attacks (H(c) ||t).
b) The hash is then signed by the secret key of claim predicate (**Ÿ**) to produce the signature **δ**

δ ⟶ CP-ABS.Sign(GPP, h(CT)|| t, Ÿ,$\{Pk_{aid}\}_{aid\in I_A}$},($GPK_{uid}$,GSK'_{uid}),$SK_{uid,aid}$)

c) After the Encrypt Phase and Sign Phase, TTP service will send the ciphertext CT, the attribute based encrypted decryption key$\{DEK\}_{\{Pk_{aid}\}aid\in I_A}$, the signature **δ,** period of validity t and claim predicate **Ÿ** {CT , $\{DEK\}_{\{Pk_{aid}\}aid\in I_A}$, **δ** , t , **Ÿ** } to owner.
d) The owner will upload {CT,$\{DEK\}_{\{Pk_{aid}\}aid\in I_A}$, **δ** , t , **Ÿ** } to the cloud storage provider (CSP).
e) The cloud storage provider (CSP) first checks the validity of t with current time, and obtain all verification keys that corresponds to attributes depicted in the claim predicate **Ÿ** from the AAs, then verify the **δ** by the boolean value result

R0→Verify (GPP, h(CT)|| t, δ, Ÿ$\{Pk_{aid}\}_{aid\in I_A}$})

If the signature is a valid signature, the CSP will accept the upload request and save the time, Ÿ and verification keys with the encrypted file CT.

5.2.2 New User Grant

a) The data owner defines the role of user and determines if he is a reader or writer and sends this information to AAs.
b) The user sends his certificate to AAs to get his designated keys.
c) Each AA validates the signature to check if the user is a legal user or not.
d) If the user is a legal user, then each AA will assign him an attribute set S that is related to his identity/role in its administration domain. Otherwise, it aborts.
e) Each AA runs the **SKeyGen** algorithm to generate all secret key components for the user. If the user is a reader, he will only receive secret key components to de-crypt the ciphertext. If he is a writer, he will receive secret key components to decrypt the data. In addition to, secret key components to sign the data.
f) After the user receives the key, he is able to either read or write to data files stored at a CSP.

5.2.3 File Access

Whenever a user wants to read the file, he processes as follows:

a) The reader requests the file from the CSP.
b) The cloud sends the file **{CT,**$\{DEK\}_{\{Pk_{aid}\}aid\in I_A}$**, δ , t , Ÿ }** to the reader
c) The user sends **{CT,**$\{DE\}_{\{Pk_{aid}\}aid\in I_A}$**, δ , t , Ÿ }** to TTP
d) TTP request corresponding public keys from AA to verify signature (**δ)**

R1→**Verify(GPP, h(CT)|| t, δ, Ā , $\{Pk_{aid}\}_{aid\in I_A}$})**

g) If the signature is valid, the TTP uses user's secret keys ($\{SK_{uid,aid_k}\}_{aid_k\in I_A}$) to decrypt attribute based encrypted decryption key$\{DEK\}_{MA-CP-ABE}$ and get symmetric decryption keyDEK. Otherwise, abort.
h) The TTP decrypts encrypted file CT using symmetric decryption key DEK to obtain plaintext F.
i) TTP send plaintext F to reader.

Whenever a user wants to update a file, he processes as follows:

a) Download the file as a reader (same steps as stated above) to get plaintext.
b) The user encrypts the plaintext F as in the encryption phase producing CT1
c) Then the user sign the CT1 as in sign phase producing new signature **δ**1 with a new timestamp
d) Upload the updated encrypted file CT1 ,the attribute based encrypted decryption key$\{DEK\}_{MA-CP-ABE}$, the new signature **δ**1 with a new timestamp **t1**and claim predicate T_{sign} to the CSP.

e) The CSP will first check the validity of t1, then verify the **δ**1 by the **Ÿ** and verification keys to check if the user is able to update the file according to his secret keys or not.
f) If the user is valid user, the updated file will be stored on the cloud otherwise the CSP will reject the update request.

6 ANALYSIS OF PROPOSED SCHEME

6.1 Security Analysis

6.1.1 Data Initialization and Key Generation

In multi-authority CP-ABE, user keys come from different authorities. Therefore, user's secret keys must be tied together for the same user without exposing it to any collusion attacks. Based (Yang and Jia, 2013), these issues are resolved by using CA to tie secret keys,CA is not involved in any creation of secret keys or management of attributes. CA is responsible only for issuing global keys and global unique identities to legal users and authorities along with global master key GPMK. Since each user has a unique global identifier uid andsecret keys issued by different AAs for the same uid, users can be tied keys for decryption without the need for a central authority as in (Chase M.,2007). Therefore, a colluding user cannot combine his secret keys from a certain set of authorities with another user who has enough keys from the other authorities to decrypt the ciphertext, because each key has his uid. In addition, each user key contains a random number t for randomizing the key. Due to this random number t and the AA global identifier aid, each component associated with the attribute in the secret key is distinguishable from each other. Therefore, users and authorities cannot collude by combine their keys to get access to user's data, even if some AAs may issue the same attributes. Furthermore, the CA do not havefull control over encrypted data, because its GPMK is a share of the key not the whole key as in (Chase M.,2007).

Moreover, each user is issued a certificate from the CA that it is presented to AA for requesting the secret keys. The AA validates this certificate using the verification keys issued from CA before issuing any keys to users. By doing his validation step, we prevent any user from using a fake uid to request a decryption keys from AAs. In addition, this certificate prevents attribute authorities from colluding with each other, because, users do not present their unique identifiers to every authority for requesting the key. They just submit a pseudonym based on user unique identifier that proves to the attribute authority that he has this uid, without revealing the uid itself.

6.1.2 File Creation

Initially, the TTP service encrypts users data using a symmetric key selected by the user according to sensitivity of data either locally or remotely. The keys are stored on a hardware device which makes them it hard to for attackers to break.All user sensitive data sent to the CSP are encrypted. Therefore, the cloud has no access to plaintext.

6.1.3 File Access

Our design offers two-layer encryption for data before outsourcing it to the cloud. The data is encrypted according to the level of sensitivity. Then, the encryption key is encrypted with MA- CP-ABE secret key. After that, the encrypted outsourced data$\{F\}_{DEK}$, the attribute based encrypted decryption key$\{DEK\}_{MA-CP-ABE}$, encrypted decryption key, signature and claim predicate $T_{sign}$are uploaded to the cloud. In order for the adversary to extract any information about F, he has to decrypt DEK firstly in order to extract any information about F. However, such session key (DEK) is encrypted with the access control policy (τ) it would further require MA-CP-ABE secret key (SK) that can satisfy (τ). Since, SK is only shared with the legitimate users by the data owner, the computational complexity for an attacker would be equal to deciphering CP-ABE without SK. Actually, MA-CP-ABE used in this paper is provably secure under given the decisional q-parallel Bilinear Diffie-Hellman Exponent (q-parallel BDHE) problem is hard. Furthermore, unauthorized users cannot update any file, because any user must be authenticated he to the cloud by providing the secret keys that satisfy its claim predicate predicateT_{sign}. Since these users cannot present these credentials to the cloud, they are not allowed to update the file. Therefore the message integrity with non-repudiation can be provided by our proposed scheme. Moreover, our proposed scheme is resistant to replay attacks, because, whenever an unauthorized user upload an old version encrypted file with an old signature which was signed by a former writer to the cloud storage server, they are not able to replace data with stale information from previous writes. This is because of

the period of validity t (time stamp) associated with each file.

7 CONCLUSIONS

In this paper, we defined a new framework for data security in cloud storage services. Through this framework, we were able to achieve data confidentiality and fine grained access control by delegating key management and enforcement access control to a TTP with minimal overhead on cloud users .Our framework was also able to conquer two of most important outsourced data sharing attacks: replay attacks and collusion attacks. In addition, our scheme was able to shift most of the extensive computation load to the cloud. Our future work is to evaluate this system and implement it in a real application to prove its efficiency.

REFERENCES

Bethencourt J., Sahai A., and Waters B., 2007. Ciphertext-policy attribute based encryption. *In 28th IEEE Symposium on Security and Privacy*.

Borgmann M., Hahn T., Herfert M., Kunz T., Richter M., Viebeg U., and Vowe S., 2012. On the Security of Cloud Storage Services. Fraunhofer Institute for Secure Information Technology SIT. Available from: http://www.sit.fraunhofer.de/en/cloudstudy.html html [Accessed 6 March 2014].

Cao D., Zhao B., Wang X., Su J., and Ji G., 2011. Multi-authority Attribute-Based Signature. *In INCoS '11, Third International Conference on Intelligent Networking and Collaborative Systems.*

Chacos B., How to encrypt your cloud storage for free. PCWorld. Available from: http://www.pcworld.com/article/2010296/how-to-encrypt-your-cloud-storage-for-free.html [Accessed 6 February 2014].

Chase M., 2007. Multi-authority attribute-based encryption. *In TCC' 07, The Fourth Theory of Cryptography Conference.*

Chase M. and Chow S.M., 2009. Improving privacy and security in multi-authority attribute-based encryption. *In CCS '09, 16th ACM conference on Computer and communications security.*

CircleID Reporter, 2009. Survey: Cloud computing 'no hype', but fear of security and control slowing adoption. Available from: http://www.circleid.com/posts/20090226_cloud_computing_hype_security [Accessed 7 January 2014].

Dahshan M. and Elkassass S. 2014. Data Security in Cloud Storage Services. *In CLOUD COMPUTING'14, The Fifth International Conference on Cloud Computing, GRIDs, and Virtualization.*

Deniability P., Gasti P., Ateniese G., and Blanton M., 2010. Deniable cloud storage: sharing files via public-key deniability. *In WPES '10, 9th annual ACM workshop on Privacy in the electronic society.*

Di Vimercati S. D. C., Foresti S., Jajodia S., Paraboschi S., and Samarati P., 2007. A data outsourcing architecture combining cryptography and access control. *In CSAW '07, ACM workshop on Computer security architecture.*

Goh E., Shacham H., Modadugu N., and Boneh D., 2003. SiRiUS: Securing remote untrusted storage. *In NDSS'03 , Tenth Network and Distributed System Security Symposium.*

Hu W., Yang T., and Matthews J. N, 2010. The good, the bad and the ugly of consumer cloud storage. *In ACM SIGOPS'10, Operating Systems Review.*

Jung T., Li X., Wan Z., and Wan M., 2013. Privacy preserving cloud data access with multi- authorities. *InINFOCOM'13, 33rd IEEE International Conference on Computer Communications.*

Lewko A. and Waters B., 2011. Decentralizing attribute-based encryption. *In Proceedings of EUROCRYPT'11, 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques.*

Liu Z., Cao Z., Huang Q., Wong D. S., and Yuen T. H.,2011. Fully secure multi-authority ciphertext-policy attribute-based encryption without random oracles. *In ESORICS'11, The European Symposium on Research in Computer Security.*

Muller S., Katzenbeisser S., and Eckert C.,2009. On multi-authority ciphertext-policy attribute-based encryption. *In Bulletin of the Korean Mathematical Society.*

Newton, D. 2011. Dropbox authentication: insecure by design. Available from: http://dereknewton.com/2011/04/dropbox-authentication-static-host-ids/[Accessed 17 February 2014].

Patel H. R., Patel D., Chaudhari J., Patel S., and Prajapati K., 2012. Tradeoffs between performance and security of cryptographic primitives used in storage as a service for cloud computing. *In CUBE '12, 2012 International Information Technology Conference.*

Sahai A., and Waters B.,2005. Fuzzy Identity-based Encryption. *In EUROCRYPT'05, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques.*

ShinJ., KimY., ParkW., and ParkC., 2012. DFCloud: A TPM-based secure data access control method of cloud storage in mobile devices*In CloudCom'12, IEEE 4th International Conference on Cloud Computing Technology and Science.*

Sosinsky, B., 2010. Cloud Computing Bible. John Wiley & Sons. First Edition.

Tang Y., Lee P. P. C., Lui J. C. S., and Perlman R., 2012. Secure Overlay Cloud Storage with Access Control and Assured Deletion. *In Proc. of TDSC'12, 2012 IEEE Transactions on Dependable and Secure Computing.*

Waters B., 2011. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably

secure realization. *In PKC'11, 4th International Conference on Practice and Theory in Public Key Cryptography.*

Yang, K., Jia, X., Ren, K., and Zhang B., 2013. DAC-MACS: Effective data access control for multi-authority cloud storage systems. In *INFOCOM'13 , 33rd IEEE International Conference on Computer Communications*.

Yang K., and Jia X., 2013. Expressive, Efficient and Revocable Data Access Control for Multi- Authority Cloud Storage. *In TPDS'13, IEEE Transactions on Parallel and Distributed Systems*.

Zhiquan L., Hong C., Zhang M., and Feng D., 2012. A secure and efficient revocation scheme for fine-grained access control in cloud storage. *In CloudCom'12, IEEE 4th International Conference on Cloud Computing Technology and Science.*