Protecting Outsourced Data in Semi-Trustworthy Cloud: A Hierarchical System

YUZHAO WU, Institute for Interdisciplinary Information Sciences, Tsinghua University
YONGQIANG LYU, QIAN FANG and HAO YIN, Research Institute of Information Technology & TNList, Tsinghua University
GENG ZHENG, Department of Computer Science and Technology, School of Information Science and Technology, Tsinghua University
YUANCHUN SHI, State Key Laboratory of Intelligent Technology and Systems, Department of Computer Science, Tsinghua University

***Abstract*—Data outsourcing in cloud is emerging as a successful paradigm that benefits organizations and enterprises with high-performance, low-cost, scalable data storage and sharing services. However, this paradigm also brings forth new challenges for data confidentiality because the outsourced are not under the physic control of the data owners. The existing schemes to achieve the security and usability goal usually apply encryption to the data before outsourcing them to the storage service providers (SSP), and disclose the decryption keys only to authorized user. They cannot ensure the security of data while operating data in cloud where the third-party servicers are usually semi-trustworthy, and need lots of time to deal with the data. We construct a privacy data management system appending hierarchical access control called HAC-DMS, which can not only assure security but also save plenty of time when updating data in cloud.**

Keywords—Data outsourcing, Semi-trustworthy storage, attribute based encryption, hierarchical access control

I. INTRODUCTION

The existing network services allow users to create and share data freely, which brings the service providers valuable, massive, and ever growing volumes of data. Efficient storage, management, and distribution of these data will be a huge management and financial burden for small and emerging companies, leaving them rely more and more on a third party storage service providers to store, manage and distribute their business data, namely "data sourcing", to achieve high-performance, low-cost, scalable data sharing service. Some successful companies nowadays like Apple, Dropbox and other companies successfully completed its business model based on this service mode. This have proved that this service mode is of great value and attraction.

Under this mode of service, the data owner outsources its data to storage service provider (SSP). SSP is responsible for data storage, management and distribution. In general, SSP is a commercial company. Data stored in it is not under the physical control of the Owner. Study usually regards the SSP "honest but curious" of nature. Namely, SSP honestly executes user's setting on data management and distribution, but may attempts to acquire the data content for profit by selling the data content.

Most enterprises and organizations view their data as a very valuable asset, which needs sufficient security measures to protect it from unauthorized access. As more and more corporations and personal data are outsourced to the SSP, data confidentiality becomes the main concern. It will hinder the development of data outsourcing service mode if there are no good solutions.

Due to the diversity of data sources, business requirements and the user types, data must be selectively access based on different strategies to prevent users from obtaining data out of his authorization, which results data breaches. At the same time, data content cannot be gotten by the SSP for its "honest but curious" nature. Researchers have proposed the use of encryption to protect data outsourced to the SSP. These schemes require owner to encrypt the data before outsourcing data to the SSP. After being shared to users, decryption is implemented on the trusted client with the proper key. Existing studies based on encryption fall into the following two categories:

- Single key scheme.

This scheme was mainly used to deal with "database as a service" [7] paradigm, with which Owner can outsource their database to the SSP. The Owner encrypts his data with one single key and builds indexing information to support all SQL clauses without decrypting the data. In order to implement access controls, Owner needs to filter query results of the users to prevent user from accessing to unauthorized data.

- Multiple keys scheme

Data with same access policies are encrypted by the same key. Authorized users gain access to the data by obtaining the corresponding key.

The former scheme is unable to adapt to today's unstructured data storage service. The involvement of Owner in user's query becomes a performance bottleneck of this scheme, which is unable to make full use of the SSP's advantages in the data distribution. What is more, the whole data's confidentiality will be compromised if any of the users was compromised. Those problems have led the researchers to focus on the second scheme. In the second scheme, researchers integrate encryption with access control. The key issue for this scheme is the large number of keys due to the diversity of the access polices. How to achieve the desired security goal without introducing a high complexity on key management and data encryption is a

2332-5666/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCSW.2017.19

great challenge. Researchers have proposed several scheme which can be divided into following categories by the key management:

- Pre-distribution key scheme: Determining keys for every user based on access polices, then deliver all or some keys to user in advance during user registration phase.

- Dynamic key-derivation scheme: Users are not given keys directly, instead, keys are dynamically derived from a combination of public information (generated by Owner) and the users' key or attribution credentials.

While the research field is still rapidly developing, it is a good time to analyze the proposed schemes, discuss the existing problems and challenges. In this paper, we review the encryption based schemes to solve data protection problems when data are outsourced to the "honest but curious" SSP. We focus on the schemes' approaches in realizing fine-grained management by enforcing fine-grained encryption. We evaluate these programs from several aspects including encryption policy management, key management, user management, data update. The limitations of these schemes are discussed and possible future trends are also given.

- Our Contributions

In this paper, our contributions lie in:

Firstly, we summarize the general system models of data outsourcing in cloud and analyze several issues on it

Firstly, we suggest a hierarchical access control system to solve the problem faced in data sharing in semi-trustworthy environment.

Secondly, we construct a multi-layer key policy, which can not only increase the security of privacy data but also save large amount of time when updating users' information.

Thirdly, we improve the construction of an access structure based on the ACV-BGKM, which can cut down the time for updating access policies while not sacrificing availability and security.

II. EXISTING MODELS AND ISSUES

In this section we will give a general system model of solving the problem of data outsourcing. Some issues encountered when enforcing encryption and access control on those data are discussed.

A. System models

The systems that use data outsourcing mainly consist of the following parties: the data Owner, data storage service provider (SSP) and the data consumer (user). Some systems also have a trusted third-party auditor (TPA) to audit every single operation on data storage and access to ensure the integrity and coherence of the data. The TPA is beyond the scope of our discussion. [18] Explained more about it. The roles of these parties in the system model are as follows:

Data storage service provider (SSP): Providing structure and unstructured data storage services and highly data consistent views. In addition to the features mentioned above, the SSP also provides high performance, high reliability, and high scalable and low cost data storage services. As we mentioned previously, SSP has "honest but curious" nature, which means that the SSP should not has access to the data content. At the same time, SSP may be subjected to internal attacks and external attacks, which makes the data confidentiality relied on SSP's security under threat.

In addition, many schemes [3; 5; 16; 17; 19; 20] assume SSP have abundant computing power, which allows Owner to delegate part of the computing tasks to the SSP to reduce the Owner's overhead of computing costs without sacrificing any security. This assumption coincides with Today's public cloud based data outsourcing service.

Data Owner: the data Owner is the enterprises or organizations who use the data outsourcing service. It defines the access polices for its data and upload them to SSP. In addition to upload data to SSP, Owner will run its programs on the SSP to manage its data. Maintaining a local data backup may require a lot of cost. Hence data Owner often does not have a local backup of data stored on the SSP.

Data consumer (user): Users obtain data and service by accessing data from SSP. He will try to assess data either within or outside the scope of their authorization. Some malicious users may collude with other users or SSP for the purpose of acquiring data content exceeding their authorization when it is highly beneficial. In the modern network services, the users may be frequently join and leave the system, the users' authorization may also be updated.

The relationship of these parties are shown in the Figure 1.

Figure 1. The relationship of the three parties

Apart from the assumptions above, for simplicity, extra assumptions are made to let the researchers focus more on data protection for data outsourcing. These assumptions are: 1) the communications between the parties are assumed to be secured under the existing security protocols such as SSL and TLS; 2) User's identify authentication is ignored because it is another field of study.

B. Issues

Under the system model discussed above, schemes to protecting data outsourced to the SSP should both ensure that the data on the SSP is encrypted and enforces access control on usage of the data. When the users try to access an item, the cypher text must be decrypted on trusted side. Proper access controls are implemented to ensure that the users can only access its authorized data. Encryption,

however, will bring negative impacts on system's usability, such as encryption cost and key management overhead. The usability of the system should be also achieved when guaranteeing the security. The following issues should be taken into consideration:

- Encryption and decryption should not bring high costs. The Owner and the users need to process vast amounts of data, so there must have secure and efficient encryption and decryption algorithms to ensure the system's availability. The system must be able to efficiently manage and distribute keys. In order to achieve fine-grained access control, the data may require fine-grained encryption, which will cause a lot of keys. In order to ensure the users' accessibility to the data, the Owner and users may need to manage and store many keys. An efficient key management and distribution scheme is a core requirement.
- Data access policies may face frequent changes, the user authorization may also change. The system needs to support flexible, low-cost access policy changes, user revocation to ensure the effectiveness of the access control policy.
- Data must be rekeyed when the access polices or user group for the data are changed to prevent the revoked users to access data. Note that the Owner may not have a local backup for the data, the rekey operation which needs to download and upload data will be unacceptable for that there is usually a large amount of data to process.
- The system should be scalable, especially when the quantity of data and users is very large.

In traditional database environment, access control is over the most important problems. Views mentioned in [2] [3] which is based on SQL is a most commonly used solution. Also in semi-trustworthy environment, using data mining and data obfuscation are both efficient solution for privacy protection. But these two methods both have limitations. When data are required to be encrypted, these methods cannot be used anymore.

Bertino and Ferrari [1], as well as Miklau and Suciu [9] tried key pre-distribution scheme in untrustworthy storage. This scheme does not have good scalability thus cannot support fine-grained access control. Di Vimercati [3] gave data a fine-grained encryption based on access control list (ACL), in which user can only save one key to derive all the authorities he need, but it will become more expensive while users increasing. Goyal [6] first constructed a key-policy attribute-based encryption (KP-ABE), implemented the idea of identity-based encryption purposed by Shamir [15] to access control in the cloud. Wang et al [17] makes a hierarchical management of key policy based on ciphertext-policy attribute-based encryption (CP-ABE), but the hierarchical management just appear in store users and attributes' information. Nabeel constructed and improved a group key management scheme on broadcast called access control vector - broadcast group key management (ACV-BGKM), which used access tree to make the scheme have more advantages [10-13]. They resolved the policies to both owner and the cloud to make it a two-layer key policy, but a single entity's power became weak and it would be easily suffered the collusion attack.

III. OUR SYSTEM FREAMWORK

Previous work always need to download the data from the cloud to local for decrypting when updating access policies. That's because the cloud is semi-trustworthy, if we want to do a re-encrypt or rebuild an access structure for data, we should do them in local in prevent of the cloud to get the privacy data. This will take expensive cost to the local data owner. They also assumption a single owner as trusted authority and indiscriminate users, which cannot work in many scenarios in semi-trustworthy environment.

We try to put the encrypt work in the cloud while not sacrificing the security and build a hierarchical access control privacy data management system (**HAC-DMS**). The system is based on these assumptions on the cloud storage authority:

- Storage authority is honest but curious. It will honest execute the operation we call it to do, but may get the content of data we put on it.
- Storage authority would not conspire with deleted users. In this assumption we can put more operation in the cloud to save update time.

Under these assumptions, the cloud storage authority is semi-trustworthy. The work cloud did for the data owner is trustworthy but the data owner should prevent the cloud from getting the privacy data.

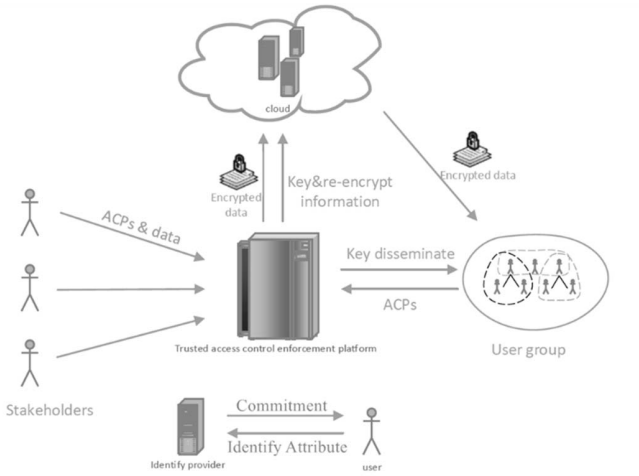

Figure 2. Framework of HAC-DMS

The framework of HAC-DMS is in Figure 2. It includes these modules:

- **Stakeholders**: Who produce and owe the data, which include producers and owners. They will create the data and keep their keys for the initial encryption on the data individually. They don't directly transmit data with the cloud but through a third-party trusted management platform (**TMP**).

- **Semi-trustworthy Storage Platform**: The semi-trustworthy cloud storage platform, which is used to store the encrypted data and do the key publishing. The cloud platform will never get the plaintext of the data but only the media ciphertext after several layers' initial encryption. It will receive acquirement from stakeholders and users and transmit data with the TMP. The TMP will directly deal with the data and send them to the user.
- **Trusted Management Platform**: A Trusted Management Platform, which helps stakeholders to do the base layer's encryption and decryption, and transmit data with the cloud and users. It will be guaranteed online all the work hours of the system.
- **Users**: The data consumers, who may have the need to make further data management on the data they get. When they want to access the data, the will send their acquirement to the cloud and receive data from the TMP.

We divide the framework into stakeholder layer, TMP layer, cloud layer and user layer. The data uploaded to the cloud layer will first encrypted by multiple stakeholders. Each stakeholder will do an initial physical encryption on the data, which will ensure the cloud cannot get the original data. The cloud layer provides a logical encryption on the data. When the access structure changes, the cloud layer can execute re-encryption of the data without download them to the stakeholder layer, which can ensure low cost and flexible updating while not breaking the security of the data. For each stakeholder's encryption, we implement stream cipher on it to make sure each stakeholder's encryption can execute individually without affect others' encryption and decryption. Stream cipher provides good usability but lacks of security against attacks on the keystreams. We implement Public Unclonable Function (PUF) on hardware-based stream cipher, proposed by Qiu et al [14], which can output keystreams in a continuous, free-of-configuration, high-speed and secure manner.

In our framework, cloud only provides logical encryption on the data which have already be encrypted. When revoking users or changing the access policies, we just update the key in the cloud. Moreover, we call a third-party TMP here to instead owner to do the initial encryption. Owner firstly send the data to the TMP to do the initial encryption, and then send the encrypted data and access control policies to the cloud. When a user who have the authority want to get the data, it also need to download the encrypted data to the TMP and then send it to user, in which way we can prevent the semi-trustworthy cloud from getting the true data. Although the user is connecting to the cloud here, we will directly send users the data without through the cloud's service.

In a social network platform who stores its data on a semi-trustworthy cloud platform, a user (here is a producer) will upload his individual information and create his own file. Not only the social website but also the user can encrypt their data first and then upload to the cloud. When others authorized users want to access the user's personal homepage, the TMP of the social website first download the encrypted data from the cloud and decrypt, then show it to the user. If this user wants to use applications in the social website using the producer's data, for example, an application to find two users' common friends, the TMP will also help him to give the application the authorization and send data to it.

IV. KEY MANAGEMENT OF HAC-DMS

In this part, we will describe how our system works when stakeholders manage the data and users access the data. The basic idea is that when encrypting we send initial encrypted data to the cloud and do key policy management in TMP; when decrypting users get data from the cloud and decrypt them in TMP.

Figure 3 gives a flow chart for how our system works. Then we will analysis it in detail.

- ENCRYPTION：

When execute the encryption (e.g. the data updating), our system runs as follow：

SETUP Generate access structure A from users' attributes set and authorization information; generate public information PI1, PI2 and public key key1, key2 using randomized algorithm

ENCRYPTION Firstly, generate media ciphertext M from key_1,PI_1， and plaintext m. Then send M, key_2, PI_2 and attributes set R to cloud and generate ciphertext E using randomized algorithm

KEY GENERATION Generate decryption key D_1 from A, PI_1 and key_1, and generate decryption key D_1 from A, PI_2 and key_2.

- DECRYPTION：

When a user wants to decrypt the data storing in the cloud, our system do as follow：

Firstly, cloud decides if user's attribute r is in A. If it is, decrypt E to M using D_2 and send to TMP. Secondly, TMP decides if user's attribute r is in A. If it is, decrypt M to m using D_1 and send to user.

The key policy which the cloud used to encryption the media ciphertext is improved from ACV-BGKM. When a user who have the authority access the data, the cloud first decrypt ciphertext to media ciphertext and send it to TMP. Here if data owners don't want to use the TMP, he can just receive the media ciphertext and decrypt it himself and then send the plaintext to the user.

If the user who get the data want to do further publish of the data, he can do the same as what the owner do when he encrypt the data and called a similar TMP to help him. In

this way users can do hierarchical publish for the data.

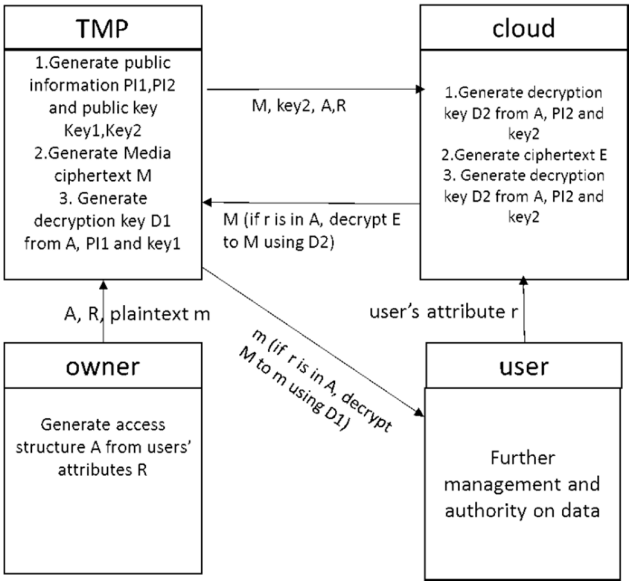

Figure 3. Key management in HAC-DMS

- UPDATE**:**

Update includes updating the content of the data and updating the access policies. For updating the content of the data, in cloud we can just update the part we have changed because the stream cipher we used is linear encryption algorithm. But if owner didn't store the original data (in usual situations), it needs to first download the part it need to change from the cloud, re-encryption after edit it and then upload.

Updating the access policies includes changing access policies and revoking users.

When changing the access policies, owner select a new key_2 and generate PI_2 following the new access policies, and send them to the cloud to re-encryption the data. A valid user can recover key_2 from PI_2 and his attributes to access the data.

When revoking a user, owner revoking the user's attributes and other information, and select a new key_2 and generate PI_2 in the same way as changing the access policies.

We can see both the operations here is very convenient and don't need to download data from the cloud. When updating our policy makes a good performance in both sides.

V. DISCUSSION

In this section, we will analyze the common problems and challenges in achieving data protection and utility. Some possible trends are also given.

- Personal privacy**:** In some situation, the data owner is not the data producer. For example, in the health care scenario, the hospital acts as the Owner collects electronic health record [7] from the patients, and then outsource EHR to the SSP to enjoy the storage service. It is Owner that is responsible to ensure the data confidential and define the access polices, and the producer's personal data protect requirements are not taken into consideration. Nowadays, more and more data leak incidents have aroused producers' concerns about the safety of their own data. Realizing producers' definitive security requirements will make user be willing to put more valuable data to the Owner, which will benefit both the Owner and SSP. Therefore, realizing producers' custom security policy will be a very meaningful research direction.
- Trusted Hardware**:** Recent years some work has prospered to add trusted hardware to the system, and the data is computed in plain text either in the trusted hardware or trustworthy computing environments isolated by certain hardware, so that the security of the data does not rely on the safety of the software system. However, the computing capability of this scheme is often restricted to trusted hardware. The extra trusted hardware increases the cost of the system, at the expense of the flexibility of the system at the same time. In order to strike a balance between the security of data and the efficiency and flexibility of the system, future research may be based on a fusion model, the trusted hardware will embed into the existing software-based system. The hardware is built in to ensure the safety of high sensitive processing. The flexibility for the data sharing and data processing efficiency is also achieved based on the original software-based system.
- Trusted TMP: The significance of the TMP is mainly because the cloud is semi-trustworthy and the owner's storage ability is not enough. If the owner has enough storage and computing ability, it can do the whole work the TMP was to do by himself. In fact, the TMP doesn't need too strong storage and computing ability. That's owe to the stream cipher we used is a kind of linear encryption. The TMP is only treated as a mediator platform. Using the stream cipher, it will not cost too many space and time. Here another considering is that users may access to the data any time, but the owner may be not online. Mobile cloud computing(MCC) is a new solution which could make data owner be online continuously [4], but research on MCC service availability is still in its early stage. [8]

VI. CONCLUSION

Protecting outsourced data in cloud is a complex problem which most mainly solutions for it but previous works all have many disadvantages and limitations. In this paper, we propose a hierarchical access control framework called HAC-DMS which can satisfy the requirement to deal with the complexity of entities who access the data. The multi-layer key policy we used and access structure we improved can increase the security of the data and reduce the cost of maintain the system at the same time. There are many works can be done under our framework, which will be the future works.

REFERENCES

[1] E. Bertino and E. Ferrari. 2002. Secure and selective dissemination of XML documents. *ACM Transactions on Information and System Security (TISSEC) 5*, 290-331.

[2] E. Bertino and L. M. Haas. 1988. Views and security in distributed database management systems. In *International Conference on Extending Database Technology* Springer, 155-169.

[3] S. D. C. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi and P. Samarati. 2007. Over-encryption: management of access control evolution on outsourced data. In *Proceedings of the 33rd international conference on Very large data bases* VLDB endowment, 123-134.

[4] H. T. Dinh, C. Lee, D. Niyato and P. Wang. 2013. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing 13*, 1587-1611.

[5] X. Dong, J. Yu, Y. Luo, Y. Chen, G. Xue and M. Li. 2014. Achieving an effective, scalable and privacy-preserving data sharing service in cloud computing. *Computers & security 42*, 151-164.

[6] V. Goyal, O. Pandey, A. Sahai and B. Waters. 2006. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security* Acm, 89-98.

[7] H. Hacigumus, B. Iyer and S. Mehrotra. 2002. Providing database as a service. In *Data Engineering, 2002. Proceedings. 18th International Conference on* IEEE, 29-38.

[8] H.-w. Lv, J.-y. Lin, H.-q. Wang, G.-s. Feng and M. Zhou. 2015. Analyzing the service availability of mobile cloud computing systems by fluid-flow approximation. *Frontiers of Information Technology & Electronic Engineering 16*, 553-567.

[9] G. Miklau and D. Suciu. 2003. Controlling access to published data using cryptography. In *Proceedings of the 29th international conference on Very large data bases-Volume 29* VLDB Endowment, 898-909.

[10] M. Nabeel and E. Bertino. 2014. Privacy preserving delegated access control in public clouds. *IEEE Transactions on Knowledge and Data Engineering 26*, 2268-2280. DOI:https:/doi.org/10.1109/TKDE.2013.68

[11] M. Nabeel, N. Shang and E. Bertino. 2013. Privacy preserving policy-based content sharing in public clouds. *IEEE Transactions on Knowledge and Data Engineering 25*, 2602-2614. DOI:https:/doi.org/10.1109/TKDE.2012.180

[12] M. Nabeel, N. Shang, J. Zage and E. Bertino. 2010. Mask: a system for privacy-preserving policy-based access to published content. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data* ACM, 1239-1242. DOI:https:/doi.org/10.1145/1807167.1807329

[13] M. Nabeel, M. Yoosuf and E. Bertino. 2014. Attribute based group key management. In *Proceedings of the 14th ACM symposium on Access control models and technologies*.

[14] P. Qiu, Y. Lyu, D. Zhai, D. Wang, J. Zhang, X. Wang and G. Qu. 2016. Physical unclonable functions-based linear encryption against code reuse attacks. In *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE* IEEE, 1-6.

[15] A. Shamir. 1984. Identity-based cryptosystems and signature schemes. In *Workshop on the Theory and Application of Cryptographic Techniques* Springer, 47-53.

[16] Z. Wan, J. e. Liu and R. H. Deng. 2012. HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Transactions on Information Forensics and Security 7*, 743-754.

[17] G. Wang, Q. Liu and J. Wu. 2010. Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In *Proceedings of the 17th ACM conference on Computer and communications security* ACM, 735-737.

[18] Q. Wang, C. Wang, J. Li, K. Ren and W. Lou. 2009. Enabling public verifiability and data dynamics for storage security in cloud computing. In *European symposium on research in computer security* Springer, 355-370.

[19] S. Yu, C. Wang, K. Ren and W. Lou. 2010. Achieving secure, scalable, and fine-grained data access control in cloud computing. In *Infocom, 2010 proceedings IEEE* Ieee, 1-9.

[20] S. Yu, C. Wang, K. Ren and W. Lou. 2010. Attribute based data sharing with attribute revocation. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security* ACM, 261-270.