Volume 4, Issue 4, April 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Improving Cloud Security by Enhanced Hasbe Using Hybrid Encryption Scheme

B.Poornima
PG Scholer,
Department of CSE,
Chettinad college of Engineering and Technology, India

Dr.T.Rajendran
Professor & Head of Dept,
Department of CSE & IT,
Chettinad college of Engineering and Technology, India

Abstract—Cloud computing is an emerging computing paradigm in which resources of the computing infrastructure are provided as services over the Internet. As promising as it is, this paradigm also brings forth many new challenges for data security and access control when users outsource sensitive data for sharing on cloud servers, which are not within the same trusted domain as data owners. To keep sensitive user data confidential against untrusted servers, existing solutions usually apply cryptographic methods by disclosing data decryption keys only to authorized users. However, in doing so, these solutions inevitably introduce a heavy computation overhead on the data owner for key distribution and data management when fine- grained data access control is desired, and thus do not scale well. The problem of simultaneously achieving fine-grainedness, scalability, and data confidentiality of access control actually still remains unresolved. This paper addresses this challenging open issue by, on one hand, defining and enforcing access policies based on data attributes, and, on the other hand, allowing the data owner to delegate most of the computation tasks involved in fine- grained data access control to untrusted cloud servers without disclosing the underlying data contents. We achieve this goal by exploiting and uniquely combining techniques of message Digest encryption (MD5), proxy re-encryption, and lazy re-encryption. Our proposed scheme also has salient properties of user access privilege confidentiality and user secret key accountability. Extensive analysis shows that our proposed scheme is highly efficient and provably secures under existing security models.

Keywords—Access control, message Digest encryption (MD5), proxy re-encryption, lazy re-encryption, ciphertext policy.

I. INTRODUCTION

Distributed computing is a guaranteeing figuring standard which as of late has drawn far reaching consideration from both the scholarly world and industry. By consolidating a set of existing and new strategies from exploration zones, for example, Service-Oriented Architectures (SOA) and virtualization, distributed computing is viewed all things considered a figuring ideal model in which assets in the processing foundation are given as administrations over the Web. Alongside this new ideal model, different plans of action are created, which can be depicted by wording of "X as an administration (Xaas)" [1] where X could be programming, fittings, information stockpiling, and so forth. Effective illustrations are Amazon's Ec2 and S3 [2], Google Application Motor [3], and Microsoft Purplish blue [4] which give clients versatile assets in the pay-as-you- use design at moderately low costs. For instance, Amazon's S3 information stockpiling administration simply charges $0.12 to $0.15 for every gigabyte- month. As contrasted with building their own bases, clients can spare their speculations fundamentally by relocating organizations into the cloud. With the expanding improvement of distributed computing advances, it is not tricky to envision that within a brief period of time more organizations will be moved into the cloud.

As guaranteeing as it may be, distributed computing is likewise confronting numerous tests that, if not decently determined, may block its quick development. Information security, as it exists in numerous different requisitions, is around these tests that might raise extraordinary concerns from clients when they store delicate data on cloud servers. For instance, in health awareness provision situations utilization and divulgence of protected health information (PHI) ought to meet the prerequisites of Health Insurance Portability and Accountability Act (HIPAA) [5], and keeping client information classified against the stockpiling servers is an alternative, as well as a prerequisite.

II. MODELS AND ASSUMPTIONS

A. System Models

Like [9], we accept that the framework is made out of the accompanying gatherings: the Information Manager, numerous Information Purchasers, numerous Cloud Servers, and an Outsider Examiner if important. To get to information documents imparted by the information holder, Information Shoppers, or clients for curtness, download information records of their enthusiasm from Cloud Servers and afterward decode. Not the information holder or clients will be constantly on the web. They come online simply on the need support. The Outsider Examiner is additionally an

online gathering which is utilized for inspecting each document access occasion. What's more, we additionally accept that the information holder can store information records as well as run his code on Cloud Servers to deal with his information documents. This presumption matches with the bound together cosmology of distributed computing which is as of late proposed by Youseff et al. [10].

B. Security Models

In this work, we simply consider Genuine yet Inquisitive Cloud Servers as [6] does. That is to say, Cloud Servers will take after our proposed convention when all is said in done, however attempt to discover however much mystery data as could be expected dependent upon their inputs. All the more particularly, we expect Cloud Servers will be more intrigued by record substance and client access benefit data than other mystery data. Cloud Servers may intrigue with a little number of malignant clients with the end goal of gathering record substance when it will be exceedingly valuable. Correspondence channel between the information owner/users and Cloud Servers are thought to be secured under existing security conventions, for example, SSL. Clients might attempt to get to records either inside or outside the scope of their access benefits.

C. Design Goals

Our main design goal is to help the data owner achieve fine-grained access control on files stored by Cloud Servers. Specifically, we want to enable the data owner to enforce a unique access structure on each user, which precisely designates the set of files that the user is allowed to access. We also want to prevent Cloud Servers from being able to learn both the data file contents and user access privilege information. In addition, the proposed scheme should be able to achieve security goals like user accountability and support basic operations such as user grant/revocation as a general one-to-many communication system would require. All these design goals should be achieved efficiently in the sense that the system is scalable.

III. TECHNIQUE PRELIMINARIES

A. Key Policy Message DigestEncryption (KP-MD5)

KP-Md5 [7] is an open key cryptography primitive for one-to-numerous interchanges. In KP-Md5, information are connected with characteristics for each of which an open key segment is characterized. The encryptor partners the set of credits to the message by scrambling it with the comparing open key parts. Every client is allocated a right to gain entrance structure which is typically characterized as a right to gain entrance tree over information characteristics, i.e., inside hubs of the right to gain entrance tree are edge entryways and leaf hubs are connected with properties. Client mystery key is characterized to reflect the right to gain entrance structure so that the client can unscramble a ciphertext if and just if the information traits fulfill his right to gain entrance structure.

B. Proxy Re-Encryption (PRE)

Proxy Re-Encryption (PRE) is a cryptographic primitive in which a semi-trusted proxy is able to convert a ciphertext encrypted under Alice's public key into another ciphertext that can be opened by Bob's private key without seeing the underlying plaintext. More formally, a PRE scheme allows the proxy, given the proxy re-encryption key $rk_{a\leftrightarrow b}$, to translate ciphertexts under public key pk_a into ciphertexts under public key pk_b and vise versa. Please refer to [8] for more details on proxy re-encryption schemes.

C. Lazy Re-Encryption

Lazy re-encryption operates by using correlations in data updates to decide when to rekey. Since data re-encryption accounts for the larger part of the cost of key replacement, re-encryption is only performed if the data change significantly after a user departs or if the data is highly sensitive and requires immediate re-encryption to prevent the user from accessing it. The cost of rekeying is minimized, but the problem remains of having to re-encrypt the data after a user's departure. Moreover, if a sensitive file does not change frequently, lazy re-encryption can allow a malicious user time to copy off information from the file into another file and leave the system without ever being detected. This is sometimes referred to as lazy update and lazy re-encryption.

IV. OUR PROPOSED SCHEME

A. Main Idea

With a specific end goal to accomplish secure, adaptable and fine-grained access control on outsourced information in the cloud, we use and interestingly consolidate the accompanying three propelled cryptographic methods: KP-Md5, Pre and Lazy re-encryption. All the more particularly, we cohort every information record with a set of characteristics, and dole out every client an expressive access structure which is characterized over these properties. To uphold this sort of access control, we use KP-Md5 to escort information encryption keys of information documents. Such a development empowers us to promptly delight in fine-grainedness of access control. Nonetheless, this development, if sent alone, might present substantial calculation overhead and awkward online trouble towards the information holder, as he is responsible for all the operations of data/user administration. As we will examine in segment V-B, the calculation intricacy on Cloud Servers will be either corresponding to the number of framework characteristics, or direct to the measure of the client access structure/tree, which is autonomous to the amount of clients in the framework. Versatility is

consequently accomplished. Responsibility of client mystery key can likewise be attained by utilizing an improved plan of KP-Md5.

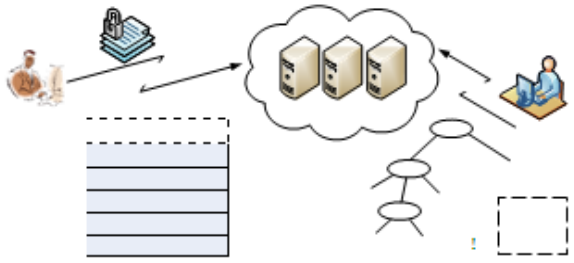

Fig. 1: An examplary case in the healthcare scenario

B. Scheme Description

For clarity we will present our proposed scheme in two levels: *System Level* and *Algorithm Level*. At system level, we describe the implementation of high-level operations, i.e., *System Setup*, *New File Creation*, *New User Grant*, and *User Revocation*, *File Access*, *File Deletion*, and the interaction between involved parties.

1) System Level Operations: System level operations in our proposed scheme are designed as follows.

System Setup In this operation, the data owner chooses a security parameter κ and calls the algorithm level interface $\mathrm{ASetup}(\kappa)$, which outputs the system public parameter PK and the system master key MK. The data owner then signs each component of PK and sends PK along with these signatures to Cloud Servers.

New File Creation Before uploading a file to Cloud Servers, the data owner processes the data file as follows.

- Select a unique ID for this data file;
- randomly select a symmetric data encryption key $\mathrm{DEK} \xleftarrow{} \mathrm{K}$, where K is the key space, and encrypt the data file using DEK;
- define a set of attribute I for the data file and en- crypt DEK with I using KP-MD5, i.e., $(\tilde{\mathrm{E}}, \{\mathrm{E_i}\}_{\mathrm{i}\in\mathrm{I}}) \leftarrow \mathrm{AEncrypt}(\mathrm{I},\mathrm{DEK},\mathrm{PK})$.

New User Grant When a new user wants to join the system, the data owner assigns an access structure and the corresponding secret key to this user as follows.

- Assign the new user a unique identity w and an access structure P;
- Generate a secret key SK for w, i.e., $\mathrm{SK} \leftarrow \mathrm{AKeyGen}(\mathrm{P}, \mathrm{MK})$;
- Encrypt the tuple $(\mathrm{P}, \mathrm{SK}, \mathrm{PK}, \delta_{\mathrm{O},(\mathrm{P},\mathrm{SK},\mathrm{PK})})$ with user w's public key, denoting the ciphertext by C;
- Send the tuple $(\mathrm{T}, \mathrm{C}, \delta_{\mathrm{O},(\mathrm{T},\mathrm{C})})$ to Cloud Servers,where T denotes the tuple $(\mathrm{w}, \{\mathrm{j}, \mathrm{sk_j}\}_{\mathrm{j}\in\mathrm{LP}} \backslash \mathrm{Att_D})$.

On receiving the tuple $(\mathrm{T}, \mathrm{C}, \delta_{\mathrm{O},(\mathrm{T},\mathrm{C})})$, Cloud Servers pro- cesses as follows.

- Verify $\delta_{\mathrm{O},(\mathrm{T},\mathrm{C})}$ and proceed if correct;
- Store T in the system user list UL;
- Forward C to the user.

On receiving C, the user first decrypts it with his private key. Then he verifies the signature $\delta_{\mathrm{O},(\mathrm{P},\mathrm{SK},\mathrm{PK})}$. If correct, he accepts (P, SK, PK) as his access structure, secret key, and the system public key.

User Revocation We start with the intuition of the user revocation operation as follows. Whenever there is a user to be revoked, the data owner first determines a minimal set of attributes without which the leaving user's access structure will never be satisfied. Next, he updates these attributes by redefining their corresponding system master key components in MK. Public key components of all these updated attributes in PK are redefined accordingly. Then, he updates user secret keys accordingly for all the users except for the one to be revoked. Finally, DEKs of affected data files is re-encrypted with the latest version of PK.

File Access This is likewise the second phase of client repudiation. In this operation, Cloud Servers react client ask for on information record get to, and redesign client mystery keys and re-scramble asked for information documents if important. If correct, they redesign this current client's mystery key parts to the most recent rendition and re-scramble the DEK s of asked for information records utilizing the most recent form of PK.

File Deletion This operation must be performed at the appeal of the information manager. To erase a document, the information holder sends the record's extraordinary I D alongside his signature on this I D to Cloud Servers. On the off chance that check of the manager's mark returns correct, Cloud Servers erase the information document.

V. ANALYSIS OF OUR PROPOSED SCHEME

A. Security Analysis

We first analyze security properties of our proposed scheme, starting with the following immediately available properties.

1) Fine-grainedness of Access Control: In our proposed plan, the information manager can characterize and uphold expressive and adaptable access structure for every client. Particularly, the right to gain entrance structure of every client is characterized as a rationale equation over information record qualities, and can speak to any fancied information

document set.

2) User Access Privilege Confidentiality: Our proposed plan simply reveals the leaf hub data of a client access tree to Cloud Servers. As inside hubs of a right to gain entrance tree could be any limit doors and are obscure to Cloud Servers, it is hard for Cloud Servers to recoup the right to gain entrance structure and consequently determine client access benefit data.

3) User Secret Key Accountability: This property might be instantly attained by utilizing the improved development of KP-Md5 [11] which could be utilized to uncover the personalities of key abusers.

Presently we break down information classifiedness of our proposed plan by giving cryptographic security evidence.

B. Performance Analysis

This section numerically evaluates the performance of our proposed scheme in terms of the computation overhead introduced by each operation as well as the ciphertext size.

1) Computation Complexity: We analyze the computation complexity for the following six operations: *system setup*, *new file creation*, *file deletion*, *new user grant*, *user revocation*, and *file access*.

System Setup In this operation, the data owner needs to define underlying bilinear groups and generate PK and MK. As is described in Section III-A, the main computation overhead for the generation of PK and MK is introduced by the N group multiplication operations on G_1.

New File Creation The main computation overhead of this operation is the encryption of the data file using the symmetric DEK as well as the encryption of the DEK using KP- MD5. The complexity of the former depends on the size of the underlying data file and inevitable for any cryptographic method. The computation overhead for the latter consists of $|\mathbf{I}|$ multiplication operations on G_1 and 1 multiplication operation on G_2, where $\mathbf{I}$ denotes the attribute set $\mathbf{I}$ of the data file. All these operations are for the data owner.

File Deletion This operation just involves the data owner and Cloud Servers. The former needs to compute one signature and the latter verifies this signature.

New User Grant This operation is executed interactively by the data owner, Cloud Servers, and the user. The computation overhead for the data owner is mainly composed of the generation of the user secret key and encryption of the user secret key using the user's public key. The former accounts for $|L|$ multiplication operations on G_1, where L denotes the set of leaf nodes of the access tree.

User Revocation This operation is composed of two stages. The second stage can actually be amortized as the file access operation. Here we just count the operation overhead for the first stage. That for the second stage will be included in the file access operation. The first stage occurs between the data owner and Cloud Servers.

File Access This operation occurs between Cloud Servers and the user. In the worst case, the algorithm $AUpdateSK$ would be called $|L|-1$ times, which represents $|L|-1$ multiplication operations on G_1.

VI. CONCLUSION

This paper points at fine-grained information access control in distributed computing. One test in this connection is to attain fine-grainedness, information classifiedness, and versatility simultaneously, which is not given by present work. In this paper we propose a plan to accomplish this objective by abusing KP- Md5 and extraordinarily joining together it with systems of proxy re-encryption and Lazy re-encryption. In addition, our proposed plan can empower the information holder to delegate the majority of computation overhead to compelling cloud servers. Secrecy of client access benefit and client mystery key responsibility might be attained. Formal security evidences demonstrate that our proposed plan is secure under standard cryptographic models.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Kon- winski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. USB-EECS-2009-28, Feb 2009.

[2] Amazon Web Services (AWS), Online at http://aws. amazon.com.

[3] Google App Engine, Online at http://code.google.com/appengine/.

[4] Microsoft Azure, http://www.microsoft.com/azure/.

[5] 104th United States Congress, "Health Insurance Portability and Ac- countability Act of 1996 (HIPPA)," Online at http://aspe.hhs.gov/ admnsimp/pl104191.htm, 1996.

[6] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi and P. Samarati, "Over-encryption: Management of access control evolution on outsourced data," in *Proc. of VLDB'07*, 2007.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Message Digesten- cryption for fine-grained access control of encrypted data," in *Proc. Of CCS'06*, 2006.

[8] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. of EUROCRYPT '98*, 1998.

[9] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *Proc. of ESORICS '09*, 2009.

[10] L. Youseff, M. Butrico, and D. D. Silva, "Toward a unified ontology of cloud computing," in *Proc. of GCE'08*, 2008.

[11] S. Yu, K. Ren, W. Lou, and J. Li, "Defending against key abuse attacks in kp-MD5 enabled broadcast systems," in *Proc. of SECURECOMM'09*,2009.

