Attribute-Based Encryption for Access of Secured Data in Cloud Storage

Ramiz Shaikh[1] Ankit Dongre[2]
[1]Research Scholar [2]Associate Professor
[1,2]Department of Computer Science & Engineering
[1,2]Jawaharlal Institute of Technology Borawan, M.P, India

Abstract— Cloud security is the most critical task while considering its working environment, i.e. outsourced, distributed and utility based. In such cases making the users data confidential, increases the trust over the system. Also the security procedure does not make the availability affected in any ways. The users of these kind of systems is always retained the services and securities preliminaries with respect to the data itself. As the cloud user can access its data frequently and if here some encryption is used which requires decryption and the repetitive process continues to increase the overheads. It requires some mechanism in which encryption is performed and if the user requires to perform some operations on secure file without decrypting it can be fulfilled. Thus homomorphic encryption lets the user facilitates about the performing operations on encrypted data which reduces the complexity of confidentiality operations. Also to prevent Cloud Servers from being able to learn both the data file contents and user access privilege information used to generate key along with the fastest access of secured data by using Attribute-based encryption (ABE).

Key words: Attribute Based Encryption, Cloud Storage, Data Storage, Homomorephic

I. Introduction

In cloud environment all the data is outsourced for services including storage and security. In such cases the users lost control over its data and the dependency is generated over the cloud provider or service provider. Thus, if the user is equipped with some controls, then the user trust is also increased and the total cloud control over the data is shifted from third party to the user. So for applying the encryption the user holds the key and not the cloud provider to increase the protection over the cloud data. But for providers view such keys are in consistence with their existing business models. These models restrict the cloud provider's capability for information mine or otherwise exploit the users' data. If a cloud provider doesn't have access to the keys, they lose access to the information for his or her own use. Whereas a cloud supplier could comply with keep the information confidential (i.e., they will not show it to anyone else) that promise doesn't stop their own use of the information to enhance search results or deliver ads. Of course, this sort of access to the information has vast worth to some cloud suppliers and that they believe that data access in exchange for providing below-cost cloud services could be an honest trade.

Also, providing onsite cryptography at rest choices would possibly need some suppliers to considerably modify their existing software package systems that may need a considerable capital investment. That second reason is actually very important, too. A lot of cloud providers don't just store client data; they do things with that data. If the client encodes the information, it's a murky blob to the cloud supplier - and a ton of cloud administrations would be unimaginable. Thus the problem is at both the end provider and user. To resolve these issues some third party needs to take the control over it whose primary function is to deal with such security scenarios and monitors the activities of both the user and provider. Hence the key generation is the major component which shows the importance of key and its control for enhancing the security.

A. Homomorphic Encryption (HE)

Homomorphic encryption schemes that allow simple computations on encrypted data have been known for a long time. It has three major components:

- KeyGen: This module will generate the key as per the requirement of the encryption schemes like symmetric or asymmetric algorithms. The generated key should provide effective security against any type of key based attacks. Mainly the key is default generated by the algorithmic component which is known to the provider. Thus some new mechanism needs to be developed for further improvement in security.
- Encrypt: This module provides the wide range of encryption solution for improved security with lesser computational loads on the servers. The practical applicability of homomorphic encryption provides the flexibility in the selection of encryption algorithms means it can be user depended or provider dependent.
- Decrypt: This can be considered as a major functionality of the homomorphic encryption. It differs from the traditional encryption standards where the complete data is decrypted for reading. Here the user encrypts the data by unique key and sends it to the provider. Now the provider or some other user is capable applying the mathematical operations on such homomorphically encrypted ciphertext and then revert the result in the same encrypted format to the user without reducing the confidentiality of the data.

II. Literature Review

In this subsection, we review some closely related works, including noninteractive verifiable computation, pairing delegation and proxy reencryption. Noninteractive Verifiable Computation: Noninteractive verifiable computation [19], [20] enables a computationally weak client to outsource the computation of a function to one or more workers. The workers return the result of the function evaluation as well as a noninteractive proof that the computation of the function was carried out correctly. Since these schemes [19], [20] deal with outsourcing of general computation problems and preserve the privacy of input data, they can be used to outsource decryption in ABE systems. However, the schemes proposed in [19], [20] use Gentry's fully homomorphic encryption system [21] as a building block, and thus the overhead in these schemes is currently too large to be practical. Recently, Parno et al. establish an important connection between verifiable computation and ABE. They show how to construct a verifiable computation scheme with public delegation and public verifiability from any ABE scheme and how to construct a multifunction verifiable computation scheme from the ABE scheme with outsourced decryption presented in [12]. Goldwasser et al. propose a succinct functional encryption scheme for general functions, and show that, by replacing the ABE scheme used in with their succinct functional encryption scheme, one can obtain a delegation scheme with is both publicly verifiable and secret, in the sense that the prover does not learn anything about the input or output of the function being delegated

III. PROPOSED SYSTEM

For fast and effective encryption with lesser ciphertext size, the approach uses, partial homomorphic encryption using RSA cryptosystem. This system is capable of generating fast response with less overhead.

Improved Cryptosystem Security using User Generated Keys: Security is further increased by giving some controls to the user for generating the keys through its characteristics. This key is complex than some other methods. By using this key with RSA cryptosystem, control on data modifications and reading rights is provided to user without its information and if the providers tries to leak the data it is of no use without this user generated keys. This attributes and characteristics do not match with some other users and hence decoding the data is not possible.

It is clear that one of the most important goals of the researches about the homomorphic encryption schemes is to make them closer to practical applications. In this section, we will discuss possible ways to achieve it. The work evaluates a distributed security scheme for effective encryption of dynamic data of users

A. *Proposed Algorithm*

1) User registers to Cloud Service Provider
2) Send user details (UID, Uname) to Third Party Auditor
3) TPA Authenticates user
4) User Logs in
5) Fetch values of Current Users Attributes (username, ID, timestamp)
6) Generate key by Apply MD5 to Users Attributes (username, ID, timestamp)
7) Convert the key to biginteger and make it final public key
8) Apply Homomorphic (RSA) encryption to Data of message M using key
9) Send Data of message M to CSP
10) Transferring Information (UserID, UserName, FromIP, FileName, FileSize, Failed login attempts) in a Log file stored at TPA
11) Download original message by passing the Private Key

IV. IMPLEMENTATION &RESULT

With help of OpenShift Red hat public cloud developed application CP-ABE.Following step are used for create application.

1) Register user details openshift public cloud and verify through mail.
2) Create application in with following tools
– JBoss Developer Studio
– Mysql 5.0
– PHP MY Admin4.0

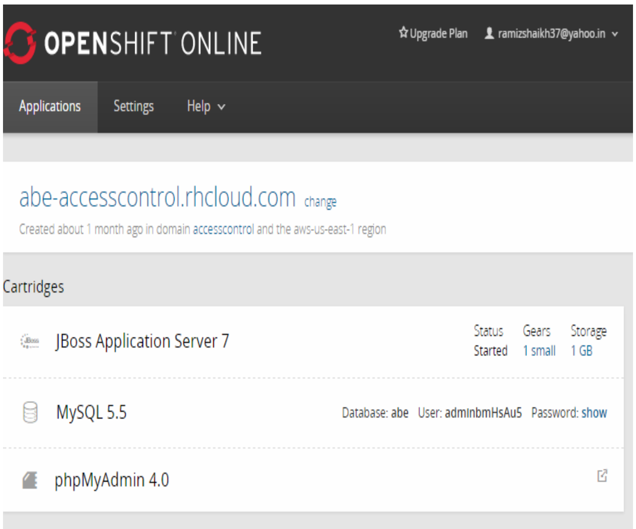

Fig. 1: Application Details on OpenShift

Figure 1 show application details of cloud which is created by us. Functionality performs using language JAVA/JSP/Servlet.

3) Application map with Eclipse IDE.

http://abe-accesscontrol.rhcloud.com/ url is used for performing all the task which are related this application.

Results are measured and analyzed on above factors and compared with traditional and other available systems. Several tables and graphs are shown by which effective comparison can be made on different views which strongly confirms the applicability of suggested approach.

S. No	UserID	UserName
1	ramiz@gmail.com	Ramiz
2	ramizsheikh@gmail.com	Ramiz Sheikh

Table 4.1: User Credential on public cloud

Login form shows the signup and if already signed, it provides the facility to enter id and password to sign in. The user has to provide his email id and password to login the account. The identity and password is been verified first then the user will be able to access his account.

Table 4.2 show time complexities of different file uploading and encryption using CP-ABE.

File Name	File Size (KB)	Encryption Time (in millisecond) (CP-ABE)
Input1.txt	10	15.0
Input2.txt	300	84.0
Input3.txt	2000	274.0

Table 4.2: Time Complexities of CP-ABE for different Files

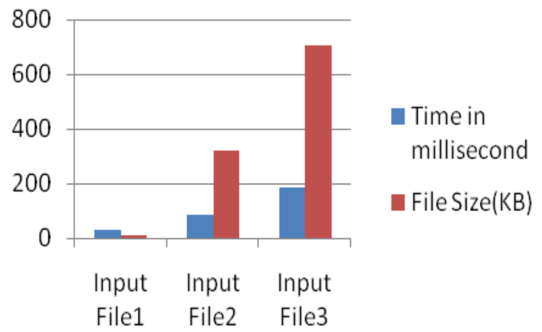

Fig. 3: Graph of CP-ABE for different Files

V. CONCLUSION

Futuristic results of the technique may show the improvement in providing the security with feasible operations on cipher using partially homomorphic cryptosystems and is most suitable for outsourced cloud environment. This improved encryption is faster and less computational overhead is involved. It provides the high end reliability towards the new orientation of the system.

The third party mechanism deals with continuous monitoring of user record. This monitoring along with improved throughput and efficiency is achieved. Out of these methods an enhanced secure scenarios is generated through our proposed TPA-HE. At the initial level of our research, we get the following benefits.

– Improved security solution with less operational overheads and retains reliability on novel encryptions
– Unauthorized access is blocked using improved key generation through user characteristics.
– Continuous monitoring gives the user behavior measurements and analyzes the affection of such novel cryptosystem on other services.

REFERENCES

[1] Shucheng Yu , “Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing”, in Proceedings of IEEE Infocomm, ISSN: 978-1-4244-5837-0/10, 2010.
[2] Srijith “Towards Secure Cloud Bursting,Brokerage and Aggregation” 2010 Eighth IEEE European conference on web services
[3] Cong Wang, Student Member, IEEE, Sherman S.-M. Chow, Qian Wang, Student Member, IEEE,KuiRen, Member, IEEE, and Wenjing Lou, Member, IEEE “Privacy-Preserving Public Auditing for Secure Cloud Storage, IEEE-2012
[4] Cong Wang1, Qian Wang1, Kui Ren1, and Wenjing Lou2,” Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing”, IEEE INFOCOM 2010, San Diego, CA, March 2010
[5] Ms. Vaishnavi Moorthy1, Dr. S. Sivasubramaniam2,” Implementing Remote Data Integrity Checking Protocol for Secured Storage Services with Data Dynamics and Public Verifiability In Cloud Computing, IOSR Journal of EngineeringMar. 2012, Vol. 2(3) pp: 496-500
[6] C. Hota, S. Sanka, M. Rajarajan, S. Nair, "Capability-based Cryptographic Data Access Control in Cloud Computing", in International Journal of Advanced Networking and Applications, Volume 01, Issue 01, 2011.
[7] Rosario Gennaro and Daniel Wichs, Fully Homomorphic Message Authenticators IBM Research, T.J. Watson, May 23, 2012
[8] K. Kajendran, J. Jeyaseelan, J. Joshi, “An Approach for secures Data storage using Cloud Computing” In International Journal of Computer Trends and Technology- May to June Issue 2011
[9] W. Luo, G. Bai, “Ensuring the Data Integrity In Cloud computing” In Proceedings of IEEE CCIS, 2011.
[10] S. Sanka, C. Hota, and M. Rajarajan, “Secure data access in cloud computing,” in 2010 IEEE 4th International [13] http://en.wikipedia.org/wiki
[11] H. Shacham and B. Waters, “Compact proofs of retrievability,” in Proc. of Asiacrypt 2008, vol. 5350, Dec 2008, pp. 90–107.
[12] Dianli GUO and Fengtong WEN, “A More Secure Dynamic ID Based Remote User Authentication Scheme for Multi-server Environment”, in Journal of Computational Information Systems, ISSN; 1553–9105, Vol. 9:No. 2, 2013, 407-413
[13] Kristin Lauter, Michael Naehrig andVinod Vaikuntanathan, “Can Homomorphic Encryption be Practical”, in ACM, 2008.
[14] Craig Gentry, “Computing Arbitrary Functions of Encrypted Data”, in ACM by IBM T.J. Watson Research Center, 2008.
[15] Bharath K. Samanthula, Gerry Howser, Yousef Elmehdwi, and Sanjay Madria, “An Efficient and Secure Data Sharing Framework using Homomorphic Encryption in the Cloud”, in Cloud 1st conference by ACM, ISSN: 978-1-4503, DOI: 1596-8/12/08, 2012.
[16] Robert Griffin and Subhash Sankuratripati, “Key Management Interoperability Protocol Profile Version 1.1”, in OASIS Standards Organizations at http://docs.oasis-open.org/kmip/profiles/v1.1/os/kmip-profiles-v1.1-os.doc, 2013.
[17] Web Article, “Amazon Web Services: Overview of Security Processes” by Amazon Services at http://aws.amazon.com/security,June 2013.
[18] K. Raen, C. Wang, Q. Wang, “Security Challenges for the Public Cloud”, Published by IEEE Computer Society, Jan/Feb 2012
[19] J.-P. Aumasson, L. Henzen, W. Meier, and R. Phan, “SHA-3 proposal BLAKE,” December 2010.
[20] GALS System Design: Side Channel Attack Secure Cryptographic Accelerators
[21] AES encryption and decryption http://www.iis.ee.ethz.ch/~kgf/acacia/c3.html
[22] Kamara, S., Lauter, K.: “Cryptographic cloud storage”. In: Proceedings of the 14th international conference on financial cryptography and data security, FC'10, pp. 136-149. Springer-Verlag, Berlin, Heidelberg (2010)

