International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

Improving Lightweight Data Sharing Scheme for Compatibility with Advanced Encryption Standard

A M Chandrashekhar
Assistant Professor,
Dept. of Computer Science and Engineering,
Sri Jayachamarajendra College of Engineering,
Mysuru, India

Mrudula N M, Niveditha N Somayaji,
Prithvi Vasanth Pratahkal
Final year students,
Dept. of Computer Science and Engineering,
Sri Jayachamarajendra College of Engineering,
Mysuru, India

***Abstract*—Cloud computing is the availability of computer system resources on demand, in particular data storage and computing power, without direct active user management. But the main disadvantage with cloud-based storage is the security of the stored documents. Because cloud service providers have access to cloud content, they would benefit greatly from using user data for various purposes such as advertising. That would invade the user's privacy. Mobile devices are now able to store and retrieve data from anywhere with the popularity of cloud computing. This would have a major security issue as there are insufficient resources for mobile devices to run complex encryption algorithms. Lightweight data sharing scheme(LDSS) has therefore been introduced and the algorithm uses CP-ABE technology to enable encryption based on attributes. LDSS-CP-ABE algorithm based on Attribute-Based Encryption(ABE) method to provide effective ciphertext access control. A normal ABE operation takes 27 times longer to run on a mobile platform and this overhead was reduced by LDSS-CP-ABE.**

But this is not compatible with the popular AES standard for encryption. We suggest a method for integrating AES and LDSS-CP-ABE algorithms in this paper.We likewise propose another component for the working of LDSS without requiring to utilize encryption and decoding servers independently. This would essentially diminish the expense of actualizing the calculation and would have no impact on the execution.

KeyWords—Mobile computing, cloud computing, data encryption, attribute based encryption, lighweight data sharing scheme, advanced encryption standard, cipher text policy attribute based encryption, data security.

I. INTRODUCTION

Different mobile cloud applications are widely used nowadays. People (data owners) can upload their photos, videos, documents and other files into the cloud in these applications and share this data with others (data users). Cloud service providers (CSP) also provide data management features for data proprietors. Due to the sensitive nature of personal data files, data owners are allowed to choose whether to make their data files public or shared only with specific users of data. Clearly, for many data owners, data privacy of sensitive personal data is a major concern. The state-of-the-art mechanisms provided by the CSP for managing privilege/access control are either not sufficient or not very convenient. They don't meet all data owners' requirements. When people upload their data files to the cloud, they end up leaving the data at a location outside of their control and the CSP may, for commercial and perhaps even other reasons, eavesdrop on user data.

Another concern would be that if people only want to share the data with certain users, which is very cumbersome, they will have to send a password to each data user.[3] The data owner can segregate data users into various groups and send the password to the groups they want to share the data to optimize the privilege management. As a solution to this problem, sensitive personal data should be encrypted before uploading to the cloud in order to secure the data against the CSP in order to solve the above problems. This problem was addressed in [1] and a framework for the same was provided.

A modified version of the algorithm has been developed in the current paper that would make LDSS more efficient with the Advanced Encryption Standard algorithm.[4] We have also developed a customized version of the data sharing scheme framework that would limit the number of servers necessary for implementation.

The key characteristics of this customized LDSS model are :

1) LDSS-CP-ABE is primarily focused on Attribute Based Encryption that provides efficient cipher text access control.

2) We remove the use of proxy servers that were used in the LDSS framework for encryption and decryption mechanisms, eliminating the overhead of sending decryption key to the proxy servers as well as lowering implementation expenses.

3) The modified version of LDSS-CP-ABE is compatible with Advanced Encryption Standard.

II. PRELIMINARY TECHNIQUES

A. Attribute-Based Encryption

ABE is a relatively recent approach that reconsiders the concept of public-key cryptography.[7] There are two major types of attribute-based encryption: one is the Ciphertext-Policy Attribute Based Encryption (CP-ABE), where the access control policy is embedded in ciphertext

; the other is the Key-Policy Attribute Based Encryption (KP-ABE), where the access control policy is embedded in the key attributes of the user.CP-ABE is more appropriate as it resembles roles-based control of access.

B. Bilinear Pairing

Let G1; G2 be two additive cyclic groups of prime order q, and GT another cyclic group of order q. A pairing is a map: e : G1 G2 ! GT , which satisfies the following properties :

8a; b 2 Fq ; 8P 2 G1; Q 2 G2

e P a; Qb = e (P; Q)^{ab}

C. Advanced Encryption Standard (AES)

The Advanced Encryption Standard, or most frequently known as AES, is a symmetric block cipher chosen by the U.S. government to protect confidential information. It is implemented worldwide for software and hardware encryption of sensitive data.

AES consists of three ciphers of blocks: AES-128, AES-192 and AES-256.[5] The Rijndael cipher, of which AES is a part, was designed to accept additional block sizes and key lengths, but AES has not adopted those functions. To encrypt and decrypt, symmetric ciphers use the same key, so both the sender and the receiver need to know and use the same secret key.

The algorithm for AES encryption defines a number of transformations to be carried out on data stored in an array.[6] The cipher's first step is to put the data into an array ; after that the cipher transformations are repeated over a number of rounds of encryption.

III. SECURITY ASSUMPTIONS

LDSS is designed in accordance with the same assumptions proposed in [1] that the CSP is honest but curious, meaning that the CSP will execute the user-requested operations faithfully, but will look at what users have stored in the cloud. The CSP will accurately store user data, undertake initial access control, and update data as requested by users. However, to get the data in plain text, CSP can perform malicious actions such as collusion with users.[8]

In LDSS, the server for proxy encryption and the server for proxy decryption are introduced to help users encrypt and decrypt data in order to minimize user-side overhead. Essentially, proxy servers are also cloud machines. So we think they're just as honest but curious as the CSP.

IV. RELATED WORKS

In this section, we focus on the works of various algorithms and schemes that are closely related to our research. Access control is an important data protection mechanism to ensure that only legitimate users can acquire data. There has been substantial cloud-based research on data access control issues, focusing mostly on ciphertext access control.[9] The cloud is generally regarded as honest and curious. Before sending to the cloud, sensitive data must be encrypted.

In [1] the authors have shown an efficient framework for using access control in mobile cloud. The algorithms used for setup, key generation, encryption and decryption are all lightweight.This means that the algorithms can be implemented with limited amount of computing resources such as memory and processing power. [10] This implies that the algorithms could run on a mobile platform and would not use much of the devices resources. Advanced Encryption Standard (AES) has been in the cryptanalysis's limelight since it was released. Reseach on AES became more popular after it was declared as the Type - I Suite - B Encryption Algorithm in the year 2003.[11]

Since this algorithm has all the complexity that is required for a cryptographic algorithm, it would be the most preferred for many of the users. AES has multiple variations of its own. It has a fixed block size of 128 bits, but the key sizes are 128 bits, 192 bits and 256 bits as depicted in [13].

V. OUR PROPOSED MECHANISM

We modified the LDSS framework to remove the need for additional servers for encryption / decryption methods and also reduce the decryption key transfer overhead. We have also modified the LDSS-CP-ABE algorithm for better compatibility with Advanced Encryption Standard(AES).

In this section, we describe the modified version of LDSS system design. First, we give the overview of the changed LDSS framework and then we present the revised LDSS-CP-ABE algorithm.

A. Modified LDSS Framework Overview

We propose a modified version of LDSS, a framework of lightweight data-sharing schemes in the mobile cloud (see Fig. 1). It has 4 components.

1) Data Owner: The data owner would upload the data to the cloud and share it with friends. The data owner would have full command over all the policies of access control.
2) Data User: The data user accesses the cloud data. The data user can either be the data owner or a third party who has access to that data that is to be obtained.
3) Trusted Authority: It is the responsibility of trusted authority to generate and distribute attribute keys. It is also responsible for providing master keys to all the data owners.
4) Cloud Service Provider: Cloud Service Provider stores Data Owners' data. It performs the operations demanded by Data Owner dutifully, although it may look at the data stored in the cloud by Data Owner.

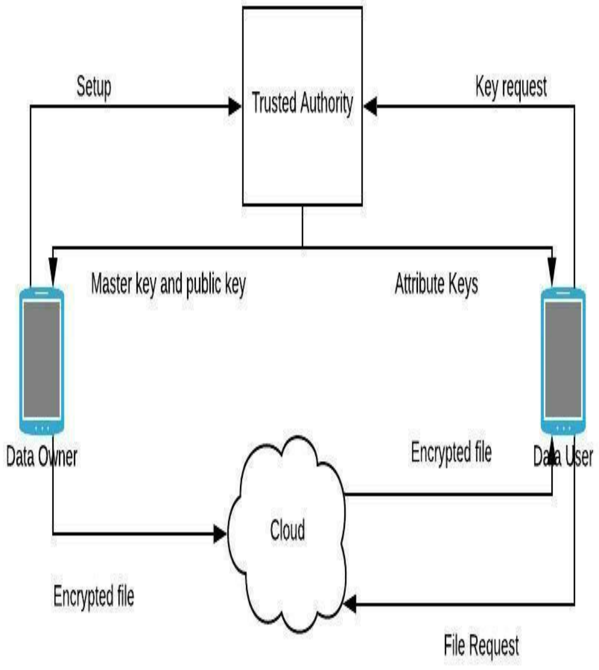

Fig. 1. LDSS modified framework

The data owner first asks the trusted authority to recognize it and generate master key and public key for itself, as shown in the figure 1. The trusted authority generates and sends to the data owner both the master key and the public key.[12] The data owner will use this master key for further communication with the trusted authority.

If the data owner has to upload data to the cloud, the set of attributes must be sent to the trusted authority which determines who can access the data. Then, for that particular data being uploaded, the trusted authority would store this information.

The data owner then encrypts the key using the encryption algorithm as described below. The key that is being encrypted would be the symmetric key that is being used to encrypt the file(using AES algorithm). The encrypted file is then sent for storage purposes to the cloud service provider.

The data user would send a key request to the trusted authority during the retreat of the file. In return, the trusted authority would send the data user's attribute keys based on the data user's access control on the requested file. The data user would then collect the encrypted file from the cloud service provider and decrypt the symmetric key using the attributes received from the trusted authority to decrypt the encrypted file.

B. Modified LDSS-CP-ABE Encryption / Decryption Algorithm

This section consists of the revised encryption / decryption algorithm for better AES compatibility. The setup and key generation would be the same as the LDSS setup and key generation algorithms as depicted in [1].

The LDSS algorithm consists of 6 sub functions.

Setup(A,VA): Generate the Data Owner's Master Key MK, the Public Key PK based on the Data Owner's attribute A and version attribute VA. KeyGen(Auser, MK): Generate attribute keys SKu

for a data user depending on his attribute set Auser and the master key MK.

KeyConvToB10(Kb16): Convert the symmetric key from base 16 to base 10.

Encryption(Kb10, PK, ACT): Generate the cipher text

CT using the symmetric key of base 10 Kb10, public key PK and access control tree ACT.

KeyConvToB16(Kb10): Convert the symmetric key from base 10 to base 16.

Decryption(CT, ACT, SKuser): Decrypt the cipher text CT using the access control tree ACT and the attribute keys SKuser.

The first two functions, Setup() and KeyGen(), are the same as in the previous LDSS algorithm in [1]. The encryption function is executed by the data owner for encrypting the symmetric key. In this function, the symmetric key that should be encrypted is first converted into an integer of base 10. The AES algorithm would use a 128 bit hexadecimal key and we would not be able to encrypt a hexadecimal key using the previous LDSS algorithm.

Then, Kb10 must be encrypted using the formula given in the algorithm and the cipher text CTk must be calculated. The public key PK would be generated during the setup and would be shared by the trusted authority. Then, the cipher text for various attributes in the access control tree ACT is calculated as per the algorithm below. The final cipher text would be a combination of cipher text of key and access control tree.

(including the left subtree Ta, right subtree Tv, and left subtree has num leaf nodes).
OUTPUT: The ciphertext CT.

1) Convert Kb16 of base 16 to Kb10 of base 10.

2) Choose E 2 Zp as the secret of ACT randomly, and calculate CTk = fg^{bE}; Kb10 e(g; g)^{aE}g.

3) Get value of both the children (namely Ea, Ev) of the root node according to the access control tree.

4) Calculate CTv = fg^{E}v ; gr:Xv^{E}v g.
5) Return CT = fCT k; CT a; CT vg.

The decryption function would be used by the data user to decrypt the symmetric key in base 10 and then convert it to base 16 so that this decrypted key can be further used to decypt the encrypted file. First, x is randomly choosen and SKuser0, SKr0, SKr0 are calculated. Then, the left and the right leaf nodes in ACT would be decrypted. Symmetric key that was encrypted would be decrypted using the formula given in step 4. The actual symmetric key would be obtained by converting the obtained symmetric key that is in base 10 to base 16.

Decryption Function

INPUT: Cipher text CT, the access control tree ACT (including the left subtree ACTa, right subtree ACTv, and left subtree has num leaf nodes), SKuser (attribute keys of user U). OUTPUT: The plaintext of Kb16 of base 16.

1) Randomly choose x, and get SKuser0 = SKr0 = SKr0 = SK^x, SKa^x, SKv^x.

2) For all leaf nodes z of ACTa, calculate q(0)

DecryptLeafNode(CTa, SKuser0, z) = e(g,g) z .

3) For all the leaf node in right subtree, calculate DecryptLeafNode(CTv, SKuser0, VA) = e(g,g) $^{q(0)}{}_{v}$.

4) Let CTk-1 = g^{bE}, CTk-2 = Kb10 * $e(g,g)^{aE}$, Evaluate

$$^{K}b10^{=} \frac{CTk\text{-}2}{e(CTk\text{-}1,SKr)=F\ x =} \quad \frac{CTk\text{-}2}{e(CT_k\text{-}1,SK_r)=e(g;g)^{rE}}$$

5) Convert Kb10 of base 10 to Kb16 of base 16.

The above functions along with setup(), keyGen() would make up the entire modified LDSS algorithm.

Encryption Function

INPUT: The symmetric key Kb16, public key PK

fG0; g; g^b; $e(g; g)^a$; $fXigi=1^k$; Xvg, access control tree T

VI. CONCLUSION AND FUTURE WORK

Lightweight Data Sharing Scheme is one of the significant advances in data security in mobile cloud. Despite the lack of resources for mobile devices to run complex and more secure algorithms, the framework offers a way to have the same security for uploaded documents. The concern was that this algorithm was not compatible with various symmetric encryption algorithm such as Advanced Encryption Standard(AES).

We modified the LDSS-CP-ABE algorithm in the above paper to enhance AES compatibility. We also revised the LDSS framework in which proxy servers would be eliminated for encryption and decryption methods to eliminate the overhead of sending private keys to servers.

This modification reduces the cost of implementation by 67 percent. The previous version would require 3 servers, one as Trusted Authority, and the other two as proxy servers for encryption and decryption process. After the modification, the entire system would require only one server working as Trusted Authority. In this paper, we have eliminated the two proxy servers for encryption and decryption which would not only reduce the cost of implementation but also reduce the overhead of transfer of keys between the data owner / data user and the proxy servers that are used for encryption and decryption.

We'd be developing a new approach in the future to ensure data integrity. We could also use cipher-text- based retrieval from the cloud to further exploit the potential of mobile cloud.

REFERENCES

[1] Ruixuan Li, Chenglin Shen, Heng He, Xiwu Gu, Zhiyong Xu, and Cheng-Zhong Xu, A Lightweight Secure Data Sharing Scheme for Mobile Cloud Computing, IEEE Transactions on cloud computing, Vol. 6, No. 2, April-June 2018

[2] A. M Chandrashekhar , Puneeth L Sankadal ,A, Prashanth Chillabatte "Network Security situation awareness system" International Journal of Advanced Research in Information and Communication Engineering(IJARICE), Volume 3, Issue 5, May 2015.

[3] Barry Sosinsky, Cloud Computing Bible, Weily Publishing, Edition 3

[4] Seyed Hossein Kamali, Reza Shakerian, Maysam Hedayati, Mohsen Rahmani, A new modified version of Advanced En-cryption Standard based algorithm for image encryption in 2010, International Conference on Electronics and Information Engineering, 02 September 2010

[5] A. M. Chandrashekhar and K. Raghuveer, "Confederation of FCM Clustering, ANN and SVM Techniques of Data mining to Implement Hybrid NIDS Using Corrected KDD Cup Dataset", IEEE International Conference on Communication and Signal Processing (ICCSP),2014, pp 672-676

[6] Nikhil Chaudhari, Mohit Saini, Ashwin Kumar, G. Priya, "A Review on Attribute Based Encryption" in 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN), 26 October 2017

[7] A. M. Chandrashekhar and K. Raghuveer , "Improvising Intrusion detection precision of ANN based NIDS by incorporating various Data Normalization Technique - A Performance Appraisal", International Journal of Research in Engineering &amp; Advanced Technology(IJREAT), Volume 2, Issue 2, Apr-May, 2014

[8] John Bethencourt, Amit Sahai, Brent Waters, "Ciphertext-Policy Attribute-Based Encryption" in 2007 IEEE Symposium on Security and Privacy (SP '07), 04 June 2007

[9] A.M.Chandrashekhar and K. Raghuveer, "Diverse and Conglomerate Modi-operandi for Anomaly Intrusion Detection Systems", International Journal of Computer Application (IJCA) Special Issue on "Network Security and Cryptography (NSC)", 2011

[10] Baodong Qin, Robert H. Deng, Shengli Liu, Siqi Ma, "Attribute-Based Encryption With Efficient Verifiable Outsourced Decryption" in IEEE Transactions on Information Forensics and Security (Volume: 10 , Issue: 7 , July 2015), 04 March 2015

[11] A.M.Chandrashekhar, Rahil kumar Gupta, , Shivaraj H. P, "Role of information security awareness in success of an organization" International Journal of Research(IJR) Volume 2, Issue 6, May 2015

[12] Medien Zeghid, Mohsen Machhout, Lazhar Khriji, Adel Baganne, Rached Tourki, A Modified AES Based Algorithm for Image Encryption in Semantic Scholar, 2007

[13] A.M.Chandrashekhar, Sowmyashree K.K, Sheethal R.S, "Pyramidal aggregation on Communication security" International Journal of Advanced Research in Computer Science and Applications (IJARCSA), Volume 3, Issue 5, May 2015

[14] N. SaravanaKumar, G.V. RajyaLakshmi, B.Balamurugan, "Enhanced Attribute Based Encryption for Cloud Computing", Procedia Computer Science, Volume 46, 2015