A SECURE DATA SHARING IN CLOUD STORAGE WITH INDEPENDENT KEY GENERATION CENTRE AND CERTIFICATE-LESS ENCRYPTION

REVATHI.R#1
#M.E, CSE, ST.JOSEPH COLLEGE OF ENGINEERING, CHENNAI, INDIA

Abstract— A mediated certificateless encryption (mCL-PKE) scheme which solves the certificate revocation problem and the key escrow problem and preserves the data confidentiality in the cloud. Since most of the CL-PKE schemes are based on bilinear pairing and computationally expensive, mCL-PKE scheme does not utilize pairing operation problem. The security mediator supports instantaneous revocation of compromised or malicious users and act as the policy enforcement scheme. This method is highly efficient than the bilinear pairing based scheme. The mCL-PKE scheme with the access control lists is been proposed to overcome the problem of sharing the sensitive information in the cloud storage. The access control list contains the details of the user and this list is generated to the cloud and the data owner for verification purpose. The main problem stated here is the key generation center in the cloud. This will bring vulnerabilities against the secure key generation. A new method is discovered to implement the key generation centre as an independent center which shares key independently. According to the access control, the data is being encrypted by the data owner using the symmetric encryption algorithm and uploads encrypted data items with an intermediate keys to the cloud. The user uses their private key to convert the partially decrypted data to the fully decrypted one. The cloud storage does not perform the decryption operation fully to preserve the data confidentiality as well as the keys information. The extension of the mCL-PKE approach allows the data owner to improve the encryption operation in an efficient way and also to implement high level independent security in the cloud based system. The result of the mCL-PKE schemes is efficient and practical.

Index Terms— Access control list, Bilinear pairing, mCL-PKE, Security mediator.

I. INTRODUCTION

The cloud computing provides a massive computing power and storage capacity which enables users to share sensitive data in the public cloud. Maintaining the data confidentiality is an important functionality in the cloud. An advantage of using a cloud for storage is that the provider is responsible for building and maintaining the storage infrastructure and its associated costs including power, cooling and server maintenance. In Certificate-less Public Key Cryptography (CL-PKC) each user holds a combination of partial private key produced by the KGC and a user-chosen secret key. The key escrow problem can be resolved by using this technique. As the structure of CL-PKC guarantees the validity of the user’s public key without the certificate, it removes the certificate management problem. Since the advent of CL-PKC, many CL-PKE schemes have been proposed based on bilinear pairings. The computational cost required for pairing is still considerably high compared to other operations such as modular exponentiation in finite fields. To improve efficiency a strongly secure CL-PKE without pairing operations is proposed. However, the previous CL-PKE schemes could not solve the key revocation problem. In public key cryptography, we should consider scenarios where some private keys are compromised. There is not an use of securing the corresponding public keys, if the private keys are compromised.

Attribute Based Encryption (ABE) method is been proposed which provide the flexibility for the user to encrypt each data item based upon their access control policy. ABE suffers from the revocation problem because the private keys provided to the existing users has to be updated whenever a user dynamic changes. To address this problem the concept of mediated cryptography to support immediate revocation is proposed. This mCL-PKE scheme, propose a novel approach to assure the confidentiality of data stored in public clouds whereas enforcing the access control requirements.

The mediated cryptography makes a practical and effective use of security mediator (SEM) which can control security capabilities for every transaction. The user's participation in a transaction will stopped immediately, once the SEM is been notified that a user’s public key should be revoked. A notation of security-mediated certificate-less cryptography is proposed to present a mCL-PKE depends upon the pairing operations. If user directly applies the basic mCL-PKE scheme to the cloud computing environment or several users access the same data, the cost of encryption for the data owner becomes high. In these cases, the data owner should encrypt the data item with the same encryption key for multiple times. To address this problem, the basic mCL-PKE scheme with an extension is been introduced. This extended scheme makes the data owner to apply the data encryption key process only once which in turns provides some added information to the cloud. By making use of this additional information the authorized users can decrypt their content using the private keys. This one is similar to that of the Proxy Re-Encryption (PRE) in which the encryption key is encrypted using the data owner’s public key and continue later to decrypt using different private keys. However, in this extension scheme, the cloud does not perform any transformation it simply acts as the storage model. The security model of the existing schemes are insecure against partial decryption attack, which does not consider any adversary capabilities of the requesting partial decryptions. Hence a secure mediated CL-PKE without pairings is needed.
The concept behind this scheme is, data owner encrypts the data and sends the encrypted content to the cloud. Then the cloud partially decrypts the outer-layer of the encrypted document and it to the requested users. The user, then fully decrypt the inner layer of the document using their secret keys. The main entities of this approach is (1) Data Owner, (2) Cloud and (3) User. The cloud contains the encrypted storage and the Security mediator(SEM). Encrypted storage part contains all the user encrypted data, SEM contains the partial decrypted data and the KGC generates the KGC key for the data owner to encrypt the document. As this KGC is made as an independent one to provide high level security for the user keys. To reduce the amount of the time required for the processes the cloud is sub divided into three parts as mentioned above. Key Generation which in turn reduce the total time of the whole process. Encryption approach are of two types, (1) Symmetric key, (2) Asymmetric key. Both the encryption and decryption process can be done with the same key called Symmetric key but two different set of keys are used in the asymmetric approach. As this symmetric key approach is much faster one than the asymmetric approach for the encrypting and decrypting the document. The key management and key distribution process are easier in symmetric approach whereas it is tedious one in symmetric key approach. Symmetric key approach is used in the proposed method to provide the high level security to the user. In this scheme, no certificate is provided to the users by making the symmetric approach as an efficient and easy one.

To protect the user from the malicious attack it is necessary to have a revocation as a compromised one. SEM supports the immediate revocation from the malicious users. There is no need to update the private key of the user whenever the user is been revoked. The most important thing is that, if more than one users are trying to access the same content then the encryption cost become too high for the owner. The same content has to be encrypted multiple times for multiple user by the data owner. To overcome this problem the extended mCL-PKE scheme makes the data owner to encrypt the file only once and the additional information are provided to the cloud to decrypt the data for the authorized users.

II. RELATED WORK

In 2006, V. Goyal, O. Pandey, A. Sahai, B. Waters [2] introduced a Fine-grained sharing of encrypted data called as Key-Policy Attribute-Based Encryption (KP-ABE). Access structure contains set of attributes and private keys to control the cipher text encryption which shares the audit-log information. To overcome the problem of identity based cryptography system S. Al-Riyami, K. Paterson [4] introduce the CL-PKE scheme. In this scheme no certificates are required to guarantee authenticity of the public keys. An intermediate exists between identity based and traditional PKE approaches. S. Coull, M. Green, and S. Hohenberger [5] proposed the concept of combining the protocol the stateful anonymous credential system with an adaptive Oblivious Transfer protocol to improve the efficiency. This Protocol combines Oblivious and anonymous protocols provides flexibility to the system. Without knowing the identity or item choices it restricts users to access the data item. D. Boneh, X. Ding, G. Tsudik [8] introduced the concept online semi-trusted mediator(SEM) to improvise the user security. SEM is combined with a simple threshold variant of the RSA cryptosystem. To Provide a simplified validation of digital signature and efficient certificate revocation. J. Camenisch, M. Dubovitskaya, G. Neven [9] proposed a protocol which is used to prevent from an anonymous access to a database. This protocol use the Diffie-Hellman model to secure the standard signature scheme without bilinear pairing. This Solves the key revocation problem in an efficient way. It also uses the random oracle based model on the intractability of factoring problem.

III. SYSTEM DESIGN

The proposed scheme is “Two Layer Encryption” and it is extended from the previous scheme of mCL-PKE. Thus, the mCL-PKE scheme does not provide any authorized certification entity for the user.
parameters for the system. Thus running the Set Up is a one-time task in the cloud environment.

B. User Registration

User creates their own public and the private key pair, called SK and PK by using the operations such as SetPrivateKey and SetPublicKey respectively using the mCL-PKE scheme.

- **SetPrivateKey**: It takes ID as input and output the user’s (the owner of ID) secret value $SK_{ID}$. Each user runs this algorithm.

- **SetPublicKey**: It takes parameter and secret key value of the user as $SK_{ID}$ as input and returns the user’s public key $PK_{ID}$.

- **SEM-Key Extract**: In KGC, each user registers its own identity and public key. The KGC in turn verifies the user’s private key corresponding to its public key and takes the params, $m_i$ and user identity ID as input. It also generates a SEM-key which is needed during decryption time by the SEM. We assume that the SEM-key is distributed securely to the SEM and for each user KGC runs this algorithm.

Each user generates their own private and public key using the mCL-PKE scheme. The user, send the keys to the KGC in the cloud with the user identity(ID). After receiving the keys from the user, the KGC generates a public key and two partial keys. One partial key is called as SEM key which in turn sent to SEM Storage part. Other partial key is referred to U-Key given to the user. Public key referred as KGC key is used to encrypt the data. SEM key, U-key, SK are used to decrypt the data.

C. Data Encryption and uploading

In the data encryption phase, the data owner needs to download the user’s public key to generate the intermediate keys. The data owner encrypts each data item only once using a random symmetric key $K$. Then the mCL-PKE is implemented to encrypt $K$ using the KGC-Keys of users. Using the extended mCL-PKE scheme, the data owner can encrypt the data using the encryption key at once and some additional information is provided to the cloud so that authorized users, decrypts the encrypted content using their private keys. The data owner uploads the encrypted data along with the intermediate keys and the access control list to the cloud. The cloud storage maintains the encrypted storage separately to store the encrypted content and the intermediate keys are stored to the SEM. The SEM in the cloud maintains the access control list, certified by the data owner.

- **Encrypt**: It takes params, a user’s identity ID, a message $M$ and user’s public key $PK_{user}$ as inputs and returns either a ciphertext CID or a special symbol $\perp$ means an encryption failure. Any independent entity can run this algorithm.

D. User Verification and data retrieval phase

In data retrieval phases, if an user wants some data it forwards a request to the SEM in the cloud, to receive the semi decrypted data. The cloud will verify the user based on the access control list already which is already stored in the cloud and examine whether the encrypted content is present in the cloud storage area.

- **SEM Decrypt**: It takes parameter, a SEM-key, and a ciphertext CID as input, and then returns either a partial decrypted message $C_{id}$ for the user or a special symbol $\perp$. $\perp$ means an decryption failure. SEM in the cloud runs this algorithm using SEM-key.

Once the verification was successful, the SEM partially decrypts the data encrypted using the data owner’s public key as input to the SEM-decryption operation and provides the partially decrypted data along with the intermediate keys. The user load is reduced upon the partial decryption at the SEM. The efficiency of the system can be improved by storing the partially decrypted document in the SEM storage area. Once user is revoked, the data owner automatically updates the access control list present in the SEM thereby avoiding the future requests by the users are not denied. The data owner does not encrypt the data and upload to the cloud whenever a new user is been added to the system. Therefore the revocation problem is completely resolved by this method. Note that existing systems are affected by revoking or adding new users to the system.

IV. PROPOSED ALGORITHM

The public-key cryptographic system contains the presumed difficulty on factoring large integers. Here the proposed RSA algorithm solves factoring problem. RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman.

RSA allows the user to create and publish their public key, along with an auxiliary value, as the product of two large prime numbers. The User must keep the prime factors as a secret one.

Anyone can encrypt a message with use of the public key, but with currently published methods, if the public key is large enough, only the user with knowledge of the prime factors can feasibly decrypt the message.

The SEM architecture is based on a variant of RSA which we call Mediated RSA (mRSA). The main idea is to split each RSA private key into two parts using simple 2-out-of-2 threshold RSA [14; 7]. One part is given to the client and the other one to the SEM. If the SEM and its client cooperate with each other, they employ their respective half-keys in a way that is functionally equivalent to (and indistinguishable from) standard RSA. Also, with the knowledge of a half-key the user cannot be able to derive the entire private key. Therefore, decryption or sign a message can be done neither by client nor the SEM without mutual consent.

Algorithmic approach

Similar to RSA, each client $U_i$ has a unique public key and private key. The public key $PK_i$ includes $n_i$ and $e_i$, where the former is a product of two large distinct primes $(p_i, q_i)$ and $e_i$ is an integer relatively prime to $\phi(n_i) = (p_i - 1)(q_i - 1)$.

Logically, there is also a corresponding RSA private key $SK_i = (n_i, d_i)$ where $d_i e_i \equiv 1 \pmod{\phi(n_i)}$. However, as mentioned above, no one party has possession of $d_i$. Instead, $d_i$ is effectively split into two parts: $d_i^u$ and $d_i^{\text{sem}}$ which are
secretly held by the client $U_i$ and a SEM, respectively. The relationship among them is:
\[
  d_i = d_i^{sem} + d_i^u \mod \varphi(n_i) \quad (1)
\]

Unlike plain RSA, an individual client $U_i$ cannot generate its own RSA keys. Instead, a trusted party (most likely, a CA) initializes and distributes the RSA keys to clients. Once a client’s request is received and approved, a CA executes the RSA key generation algorithm described below.

RSA Key Setup: CA generates a distinct set: \{p, q, e, d, $d_i^{sem}$, $d_i$\} for $U_i$. The first four values are generated as in standard RSA. The fifth value, $d_i^{sem}$, is a random integer in the interval [1, n] where $n = p \cdot q$. The last value is set as:
\[
  d_i^u = d_i - d_i^{sem} \mod \varphi(n_i) \quad (2)
\]

**Improved Advanced Techniques**

The KGC in the cloud provides the KGC-keys of users to the data owner. Then the data owner symmetrically encrypts each data item only once with the same access control policy using a random session key $K$. The KGC-Keys of the users are obtained by data owner to encrypt $K$ again. The encrypted content is uploaded to the cloud along with the access control list. In the cloud storage, the encrypted content is stored in encrypted storage area and the access control list certified by the data owner is maintained in the SEM storage area. The data owner uses AES key to encrypt the data.

The plain text is encrypted in the form of blocks. Each blocks has a binary value less than some number $n$. That is, the value of block size must be less than or equal to $\log_2(n)$; in practice, the block size is 2k bits, where $2k < n <= 2k + 1$. The process of Encryption and Decryption can be carried out in the following form, for some plain text block P and cipher text block Q:
\[
  Q = P \mod n \quad (3)
\]
\[
  P = Qd \mod n = (Pe)^d \mod n = Ped \mod n \quad (4)
\]

The value of $n$ must be known by both the sender and receiver. The sender should know the value of $e$, and similarly the receiver should know the value of $d$. Thus, the public key encryption algorithm with a public key of $KU = \{e, n\}$ and private key of $KR = \{d, n\}$. This algorithm must be satisfactory for the public key encryption scheme, and the following requirements should be met:

- The values of $e$, $d$, $n$ should be found such that $\text{Med} = M \mod n$ for all $M < n$.
- Thus the calculation of $Me$ and $Cd$ for all values of $M < n$ is performed easily.
- Determining the value of $d$ given $e$ and $n$ is infeasible.

The value of AES $M$ is uploaded into the SEM and encrypted data is stored in the cloud storage.

**KeyExpansions**—The key-expansion routine creates round keys word by word, where word in an array of four bytes. The routine creates $4x(N_r+1)$ words. For $N_r=4$ words, $N_r=10$; this routine creates 44 words. AES needs a distinct 128-bit key for each round plus one more.

**InitialRound**

**AddRoundKey**—The AddRoundKey is performed at the beginning and end of the cipher in order to provide initial and final randomness to the algorithm. Without this, the first or last portion of the ciphertext could be easily deduced, and therefore would be irrelevant to the security of the cipher.

**SubBytes**—The ByteSub transformation is a non-linear byte substitution that operates independently on each byte of the state using a substitution table(S-box). This process finds the non-linearity in the cipher.

**ShiftRows**—The ShiftRows operates on individual rows of the state. It provides diffusion throughout the AES algorithm. This operation will not change the values of byte in the row, but will change their order.

**MixColumns**—In the MixColumns transformation operates on the State column-by-column, treating each column as a four-term polynomial. Both techniques of ShiftRows, MixColumns provides diffusion in the cipher.

V. DISCUSSION

The main objective of the project is to create a independent key generator in the cloud storage. A high level security can be achieved that safeguards data thereby avoiding cipher text attack in datasets. This technique improves the efficiency of encryption for the data owner and shares data securely on the cloud storage. By using the AES algorithm we prevent the cipher-text attacks and share the data from sender to the receiver using proper requirement methods. Thereby it also supports the immediate revocation and assures the confidentiality of data stored in the cloud.

VI. RESULTS

A. Deployment of cloud with Storage Content and SEM

The mCL-PKE scheme guarantees the data confidentiality stored in cloud whereas enforcing access control requirements. Fig.2 includes the deployment of cloud service environment for the transmission of data. The cloud consist of two main services: an encrypted content storage and a security mediation server (SEM). However the sharing of data in cloud is not secure we need to implement some security mechanisms for secure data transfer. SEM which acts as a security mediator for each data request and partially decrypts encrypted data for authorized users. This new extended scheme reduces the overhead of using a pairing-free approach. Further, the costs of computation for the decryption process at the user level are reduced by enabling semi-trusted security mediator which will do decryption partially before the decrypt the encrypted data.

B. Environmental setup with independent KGC

The initial setup is done by the Key generation centre in Fig. 3. It will run the set up operation using mCL-PKE scheme and generates master key and system parameters. KGC contains the keys of user's and generate keys during key process. This setup operation is a one-time task. The cloud is trusted to perform the security mediation service and key generation correctly, but it is untrusted for the data confidentiality and escrow problem. This approach allows user to have secure the key generation and management functionality deployed in the cloud. It overcomes the key
escrow problem and hence the KGC is unable to learn the full private keys of users.

C. Identity Token Issuance

User generates their own private and public key pair, called SK and PK, using the SetPrivateKey and SetPublicKey operations in Fig. 4. The user's identity (ID) and their public keys are sent to the KGC, to produce two partial private keys and a public key. The two partial keys are named as a SEM-key and U-key. The SEM storage part in the cloud stores the key called SEM-key. The U-key, is given to the user. The KGC-key, referred to as public key, consists of the KGC generated public key as well as the user generated public key. For the encrypting the data, KGC-keys are used. To decrypt the encrypted data, SEM key, U-key, and SK are used together. The partial private key and the public key for user i as SEM-key i, U-key i, KGC-key i respectively.

Fig. 2. Cloud Deployment.

Fig. 3. Key Generation Center.

Fig. 4. User key generation.

Fig. 5. Experimental Result.

Fig. 5 shows the performance level of the secure data sharing in proposed system and the existing system. The overall results produces more security in my system than the existing one.

VII. CONCLUSION

Our approach mainly focuses on the revocation problem and key escrow problem using mCL-PKE scheme. This scheme also give supports to the immediate revocation and guarantees the data confidentiality in the cloud storage thereby enforcing the access control policies of the data owner. User generates keys to key generation center and the KGC return two keys one for the user and another for SEM. Data owner gets keys of users and encrypt data with user’s public key and send it to cloud storage. User requests for the data and the cloud verifies users access control list and partially decrypt the encrypted data and send it to the user. User decrypts the original content with his private key. When compared to other methods this method shares secure data with certificate less encryption. The future enhancement includes using an additional efficient encryption and decryption mechanism for encryption and decryption process.
ACKNOWLEDGMENT

The preliminary version of the paper appears in the Proceedings of the National Conference on Advances in Computing Technology (AICT ’15) as an invited paper.

REFERENCES


First Author

First Author

First Author

First Author

First Author

First Author

First Author

First Author

First Author

First Author

First Author