Functional Encryption Algorithm for Communication in Cloud Computing based on Attribute Based Encryption (ABE)

Rajnish Choubey
Research Scholar, Ph. D. (CSE), AISECT University, Bhopal, MP, India

Dr. Santosh K. Gandhi
Joint Controller, MP Professional Examination Board, Bhopal, MP, India

Abstract- Functional encryption ensures that user can access encrypted data stored over cloud in which the first step is to encrypt the data files {P1, P2,……..,Pn) by the data owner using Cipher text policy ABE that result in {C1, C2,……., Cn}. Then upload the encrypted files on a cloud based storage server or on a public cloud. Once the files are uploaded on the cloud the data owner list the prospective users and send all of them an email which contains two links. The first link is to send request to the Server a key as trapdoor to retrieve the encrypted data from the storage server. On the server side the third party which is a trusted party use this email to identify that this is an authorized user and then fetch the encrypted data file. Using the email id as an attribute it decrypt the file and get a function f(Ci) of the encrypted file.

Keywords: Encryption, Security, Key generation

I. INTRODUCTION

Functional encryption is a new broad vision of encryption that takes into account users data security concerns. The concept of functional encryption is where a decryption key enables a user to learn a specific function of the encrypted data and no other information. In general a functional encryption system contains a trusted authority that holds a master secret key known only to this authority. As input the authority is given the description of some function f then it uses its master secret key to generate a derived secret key sk[f] for the function f. Now anyone holding sk[f] can compute f(x) from an encryption of any x. In symbols, if E (pk, x) is an encryption of x then decryption accomplishes the following: given E(pk, x) and sk[f], decryption outputs f(x).[4]
In the above process it is to be noted that the secret key holder gets f(x), not the value x that was encrypted.
Types of Functional Encryption: Traditional encryption is a specific example of functional encryption where identity function is the only function used and which presents fully decrypted version of the cipher text. Boneh, Sahai and Waters [4] describe various types of functional encryption techniques:
Identity Based Encryption. A more advanced public-key concept called Identity Based Encryption or IBE for short, is an encryption system where any string can serve as a public key: a user's email address, a date, an IP address, a location, and even the numbers 1; 2, and 3 are all public keys. IBE public keys are often called identities and denoted by id. To obtain the secret key for a particular identity the user communicates with an authority who holds a master key. The authority verifies that the user is authorized to receive the requested secret key and if so it generates the secret key using its master key. IBE was first proposed by Adi Shamir [5,6] in 1984 and the first implementations of IBE were proposed by Boneh and Franklin [5,6] and Cocks [7] in 2001.
Attribute-based encryption. Another encryption concept called Attribute-based encryption or ABE for short, lets the encryptor specify more abstractly who can decrypt a specific ciphertext. ABE was proposed by Sahai and Waters [9] and later refined by Goyal, Pandey, Sahai and Waters into two different formulations of ABE: Key Policy ABE and Ciphertext-Policy ABE.
In a Ciphertext-Policy ABE system the encryptor specifies a policy α on recipient attributes that determines who can decrypt the ciphertext.

II. OBJECTIVE AND SCOPE

The author's aim is to develop a new technique to secure the data shared by the user on the cloud in such a way that it can be accessed only by those users who are intended for it and fulfill certain requirements of attributes specified by the data owner.

This technique also tries to ensure that the Cloud Service Provider (CSP) can only extract a function of the data and not the data itself.
The scope of this technique can be used in various cloud based data centers where user and CSP maintain their data to ensure privacy and security of data.

III. RELATED WORK

The first work in this field was done by Boneh, Sahai and Waters [4,9] formalized the notion of functional encryption.
Boneh, Sahai and Waters [4,9] formalized the notion of functional encryption towards this end, building on and generalizing a number of previous constructs including (anonymous) identity-based encryption (IBE) [5, 6, 7, 8], attribute-based encryption (ABE). Informally, a functional encryption scheme for a circuit family C associates secret keys SKC with every circuit C, and ciphertexts CT with every input x. The owner of the secret key SKC and the ciphertext CT should be able to obtain C(x), but learn nothing else about the input message x itself.[1] Moreover, security should hold against collusions amongst "key holders", namely, a collusion of users that hold secret keys SK_{C1} ; …. ; SK_{Cq} and an encryption of x should learn nothing else about x apart from $C_1(x)$;……;$C_q(x)$.[3]
Koo, Hur and Yoon[1] proposes an efficient data retrieval scheme using attribute-based encryption. The proposed scheme is best suited for cloud storage systems with massive amount of data. It provides rich expressiveness as regards access control and fast searches with simple comparisons of searching entities.

Yanli Ren, Wang and Zhang [2] proposes a new Parallel key-insulated encryption (PKIE) scheme with m helper keys, where $m \in Z, m > 2$. If one of the helper keys is exposed, only 1/m temporary secret keys would be exposed and 1/m ciphertexts could be decrypted, so the new PKIE scheme can greatly decrease loss due to key-exposure. The scheme is provably secure without random oracles based on a bilinear group of composite order. Most important, the scheme is practical and much more efficient than the extended ones from the previous PKIE schemes.

Mitchell et.al.[10] present another approach for secure cloud computing in which they developed a core language for cloud computing that include primitive types, mutable states, functional features and preservation of general recursion form of secrecy.
In this language augmented information flow type system is used to prevent control flow leakage This approach allows programmers write and test programs using conventional methods and means and reduce the effort and knowledge to write efficient code..

The authors present a Haskell-based implementation and prove that cloud implementations based on secret sharing, homomorphic encryption, or other alternatives satisfying our general definition meet precise security requirements. The core of domain-specific language (DSL) presented by Mitchell et. al. is implemented as a Haskell library, an embedded domain specific language (EDSL). Our implementation includes Shamir secret sharing and fully homomorphic encryption; both use SSL network communication between clients and any number of servers.

Chun-I Fan and Shi-Yuan Huang [12] present new technique of Controllable privacy preserving search based on symmetric predicate encryption in cloud storage. The authors mention that current privacy preserving search schemes over encrypted cloud storage services do not provide effective revocation for search privileges. Aiming at symmetric predicate encryptions and cloud storage requirements, Their work propose controllable privacy preserving search in cloud storage. This scheme is based on Private-key hidden vector encryption with key confidentiality, by C. Blundo, V. Iovino, G. Persiano, whose efficiency is much better than other predicate based encryption scemes. The controllable privacy preserving search scheme has two new features. One is revocable delegated search which makes it possible for the secret key owner to control the lifetime of the delegation. The other is undecryptable delegated search. Due to this feature, a delegated person cannot decrypt the returned matched ciphertexts even though he has the delegated privilege of search.
A Controllable Privacy Preserving Search Scheme Based on Symmetric Predicate Encryption is a 6-tuple of probabilistic polynomial-time algorithms (Secret Key Generation, Encryption, Token Generation, Test, Decryption, Revocation of Search Privilege) such that:

- SecretKeyGeneration. It takes the public parameter I as input and outputs a master secret key SK.
- Encryption. It takes SK, an attribute vector $x \in \{0, 1\}n$, and a message M as input. It outputs a ciphertext CT.

- TokenGeneration. It takes SK and a vector y ∈ {0, 1, ∗}n as input. It outputs a predicate token PToken and a time restrictive token TRToken.
- Test. It takes a ciphertext CT, a predicate token PToken, a query time Tq, and a verification information T TRToken q as input. It outputs either a matched ciphertext CT or the distinguished symbol ⊥.
- Decryption. It takes a ciphertext CT and a predicate token PToken as input. It outputs either a message M or the distinguished symbol ⊥.
- Revocation. (i.e., revocation of search privilege) It takes TRToken as input and outputs 1/0.

Ruixuan Li et. al.[13] Proposes a flexible multi-keyword query scheme, called MKQE. The authors state that cloud computing infrastructure is a promising new technology and greatly accelerates the development of large scale data storage, processing and distribution. However, security and privacy become major concerns when data owners outsource their private data onto public cloud servers that are not within their trusted management domains. To avoid information leakage, sensitive data have to be encrypted before uploading onto the cloud servers, which makes it a big challenge to support efficient keyword based queries and rank the matching results on the encrypted data. Most current works only consider single keyword queries without appropriate ranking schemes. In the current multi-keyword ranked search approach, the keyword dictionary is static and cannot be extended easily when the number of keywords increases. Furthermore, it does not take the user behavior and keyword access frequency into account. For the query matching result which contains a large number of documents, the out-of-order ranking problem may occur. This makes it hard for the data consumer to find the subset that is most likely satisfying its requirements. The proposed flexible multi-keyword query scheme, called MKQE to address the aforementioned drawbacks. MKQE greatly reduces the maintenance overhead during the keyword dictionary expansion. It takes keyword weights and user access history into consideration when generating the query result. Therefore, the documents that have higher access frequencies and that match closer to the users' access history get higher rankings in the matching result set. Our experiments show that MKQE presents superior performance over the current solutions.

In MKQE, the authors assume that the amount of data continues to increase from time to time. Accordingly, the keyword dictionary has to be expanded periodically. the authors propose a new dictionary construction paradigm, introduce a new trapdoor generation algorithm to reduce the query latencies, and take the keyword access frequencies into consideration to generate better matching result sets. In summary, we make the following contributions.

- Introduced partitioned matrices in the system design. The keyword dictionary can be expanded dynamically without touching the contents in the original dictionary. We design the novel storage and encryption algorithm to manage the keyword dictionary. MKQE greatly reduces both the dictionary reconstruction overhead and the file index re-encryption time as new keywords and files are added.
- Design a novel trapdoor generation algorithm. It can effectively reduce the impacts of dummy keywords on the ranking scores. With this new strategy, the out-of-order problem in the matching result set is solved.
- Take the keyword access frequencies into account when the system generates the ranked list of the returning results. Besides, we add the weights of the keywords in the index file. The files which contain more frequently accessed keywords will have higher weights in the query. The files with higher weights will have higher probabilities to appear in the first k locations of the matching result set. Hence, the data consumers have better chances to retrieve the desired files easily.

IV. PROBLEM DESCRIPTION

The present scenario demands an efficient data retrieval scheme using attribute-based encryption which is best suited for cloud storage systems with massive amount of data. It should also provide rich expressiveness as regards access control and fast searches with simple comparisons of searching entities. The proposed scheme should also guaranty data security and user privacy during the data retrieval process.

V. PROPOSED SCHEME

The proposed scheme will comprise of following four steps: Key Setup algorithm, Encryption algorithm, Decryption algorithm and a Key Generation function, The proposed technique of Functional Encryption can be applied in Cloud computing environment at different levels, such as Data Centers, Storage servers, PaaS, SaaS, IaaS etc.

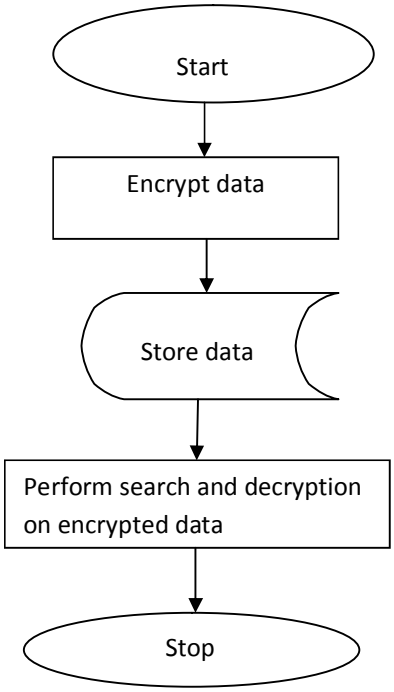
In the proposed scheme the author proposes a new scheme using Attribute based Private Key Policy Encryption also known as KP-ABE. The goal is to use proposed scheme to perform searches on encrypted data stored on cloud based storage by n number of users whose identity is not known in advance. Thus in this scheme the data owner can

specify a group of users that can perform various type of search on the stored data based upon their attributes. This ensures the privacy of the data from rest of the world.
The first step is to encrypt the data files {P1, P2,……..,Pn) by the data owner using Cipher text policy ABE that result in {C1, C2,…….., Cn}. Then upload the encrypted files on a cloud based storage server or on a public cloud. Once the files are uploaded on the cloud the data owner list the prospective users and send all of them an email which contains two links. The first link is to send request to the Server a key as trapdoor to retrieve the encrypted data from the storage server. On the server side the third party which is a trusted party use this email to identify that this is an authorized user and then fetch the encrypted data file. Using the email id as an attribute it decrypt the file and get a function f(Ci) of the encrypted file.
This function f(Ci) is sent back to the user. Then to decrypt the file the user has to click on the second link which generate the decryption key after checking that the user is still a valid user and send it back. Now using this key the user can decrypt the file and get the plaintext Pi).
At the same time, other users can also perform their own search with the link provided in their emails. This way we can achieve a secure and flexible search in a multi-user environment.
To support the technique we can calculate signal to noise ratio (SNR) and mean square error (MSE).

ALGORITHM

1. Encrypt the data
2. Upload the encrypted data on cloud based Storage
3. List prospective users
4. Email listed users an email containing links to search and decrypt data on cloud
5. Using link 1 request Cloud Service Provider to search encrypted data
6. Return the result to user
7. Using link 2 generate decryption key and decrypt the data

FLOW CHART



VI. OUTCOME

Cloud computing provides an efficient way for end users to access data anywhere and anytime. However, security and privacy concerns force data owners to encrypt sensitive data before uploading onto the cloud.
Today Cloud Computing has become a popular platform to share, store and distribute data and applications. Many individuals are using applications hosted upon cloud to store and share their data. Some of the big players of this arena are companies like Amazon with S3, Web Service, Google with Google Docs, Google Drive etc, Microsoft with Azure, Hodoop, Big Data, Manjrasoft Aneka etc.
The outcome of the research work would be a set of algorithms and functions to provide higher level of security using functional encryption for individual users as well as cloud service providers.

REFERENCES

[1] Dongyoung Koo, Junbeom Hur and Hyunsoo Yoon, Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage, Elsevier, Computers and Electrical Engineering 39, pages 34–46, 2013.
[2] Yanli Ren, Shuozhong Wang, Xinpeng Zhang, Practical parallel key-insulated encryption with multiple helper keys, Elsevier. Computers and Mathematics with Applications 65. pages 1403–1412, 2013
[3] Sergey Gorbunovy, Vinod Vaikuntanathanz, Hoeteck Weex, Functional Encryption with Bounded Collusions via Multi-Party Computation, September 5, 2012, A preliminary version of this work appeared in the Proceedings of the 32nd Annual International Conference on Cryptology (CRYPTO 2012).
[4] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In TCC, pages 253-273, 2011.
[5] Adi Shamir. Identity-based cryptosystems and signature schemes. In CRYPTO, pages 47-53, 1984.
[6] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In CRYPTO, pages 213-229, 2001.
[7] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In IMA Int. Conf., pages 360-363, 2001.
[8] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In CRYPTO, pages 290{307, 2006.
[9] Dan Boneh, Amit Sahai, Brent Waters, Functional Encryption:A New Vision for Public Key Cryptography, 2012, ACM 0001-0782/08/0X00.
[10] Mitchell et.al. Information-flow control for programming on encrypted data, Under license to IEEE.DOI 10.1109/CSF.2012.30
[11] http://en.wikipedia.org/wiki/Encryption, Wikipedia, online encyclopedia.
[12] Chun-I Fan and Shi-Yuan Huang, “Controllable privacy preserving search based on symmetric predicate encryption in cloud storage”, Future Generation Computer Systems 29 (2013) 1716–1724, ScienceDirect, www.elsevier.com/locate/fgcs.
[13] Ruixuan Li a,□, Zhiyong Xub, Wanshang Kanga, Kin Choong Yowc, Cheng-Zhong Xu, “Efficient multi-keyword ranked query over encrypted data in cloud computing”, Future Generation Computer Systems 30 (2014) 179–190, ScienceDirect, www.elsevier.com/locate/fgcs.