IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 5, Ver. I (Sept - Oct. 2016), PP 64-71
www.iosrjournals.org

Key Policy Attribute Based Encryption in Cloud Storage

Prof. Dipa Dharmadhikari [1], Sonali Deshpande[2]
[1](CSE, MIT/ BAMU, India)
[2](CSE, DIEMS/ BAMU, India)

Abstract: *Cloud Computing is the rapid growing technology and enables highly scalable services to be easily consumed over the Internet on an as-needed basis. It is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices. Cloud computing has become a highly demanded service or utility due to the advantages of high computing power, cheap cost of services, high performance, scalability, accessibility as well as availability. Some cloud vendors are experiencing growth rates of 50% per year but being still in a stage of infancy, it has pitfalls that need to be addressed to make cloud computing services more reliable and user friendly. Data sharing is an important functionality in cloud storage. The proposed system provides how to maintain integrity, confidentiality and availability to share data with others. Here described new Attribute Based Encryption cryptosystems which generates encryption, decryption time along with file size. Proposed work explains encryption with Attribute Based encryption then file is spitted into chunks. Hash key is generated for every data chunk which is unique. Secure hash algorithm is used for generation of hash key. For decryption file is searched on cloud, with private key then it is decrypted. Attribute Based encryption along with secure hash algorithm is applied. This proposed work describes analysis of encryption, decryption time calculated with Identity Based encryption and Attribute Based encryption according to file size. It creates hash key for every chunk which is unique. Proposed algorithm gives confidentiality to work as it is a public key encryption algorithm of cryptography. It is also asymmetric i.e. key set is used for data encryption and decryption. Data is stored on cloud means it is available to everyone through Cloud space. Key generated for encryption and decryption is runtime. In this work analysis is there. Encryption and decryption time required by Identity Based Encryption and Attribute Based Encryption is calculated for comparison purpose. Both algorithms user can apply on different file type such as pdf, ppt, doc, mp3, jpg and file size.*

Keywords: *Identity Based Encryption, Identity-Based Proxy Re-Encryption, key Policy Attribute Based Encryption. key Aggregate Cryptosystem.*

I. Introduction

Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models [1].

National Institute of Standards and Technology [2] has defined Cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. Networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." NIST categorizes Cloud computing into a Service Model and a Deployment Model. The Service Model consists of Infrastructure as a Service (IAAS), Platform as a Service (PAAS), and Software as a Service (SAAS). This "stack" of functionality begins with Infrastructure as a Service where consumers utilize hardware only. Moving up the stack is Platform as a Service. This layer offers the consumer an application environment where programming libraries and software can be used for development. At the top of the stack is Software as a Service.

Identity Based Encryption [3] is a public key encryption uses public key as an identity-string.it uses public key generator, holds master secret key to each user with identity, cost of storing cipher-text is more in identity based encryption. Data privacy is central issue in cloud storage. Data sharing is an important functionality in Cloud storage. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data.

The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of Cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly. However, finding an efficient and secure way to share partial data in Cloud storage is not trivial.

Key Policy Attribute Based Encryption along with secure hashing algorithm is implemented in proposed work. Firstly file is uploaded from client side then it is stored on server. on server that respective file is encrypted along with proposed algorithm. After encryption that file is stored on the cloud. For decryption that particular file is searched and then decrypted from client side. Here time requires for encryption and decryption is less as compare to Identity based Encryption. Attribute based encryption fetches feature of cryptography ie confidentiality, secure hash algorithm fetches feature of integrity and availability is extracted because the data is stored after encryption on cloud [4]. .

Therefore the best solution for the above problem is Bob encrypts files with distinct public keys but only sends Alice a single decryption key. Since decryption key can be sent by secure channel like Email and kept secret small key size is always desirable. Such a system requires costly storage to store decryption keys. So the main aim of this paper is to minimize the cost and also minimizing the communication requirements like aggregate signature [5].

II. Literature Survey

2.5.1 Identity Based Encryption

Guoel tried to build IBE with key aggregation. One of their schemes assumes random oracles but another does not. In their schemes, key aggregation is constrained in the sense that all keys to be aggregated must come from different "identity divisions". While there are an exponential number of identities and thus secret keys, only a polynomial number of them can be aggregated. Most importantly, their key-aggregation comes at the expense of $O(n)$ sizes for both cipher texts and the public parameter, where n is the number of secret keys which can be aggregated into a constant size one. This greatly increases the costs of storing and transmitting cipher texts, which is impractical in many situations such as shared cloud storage [6].

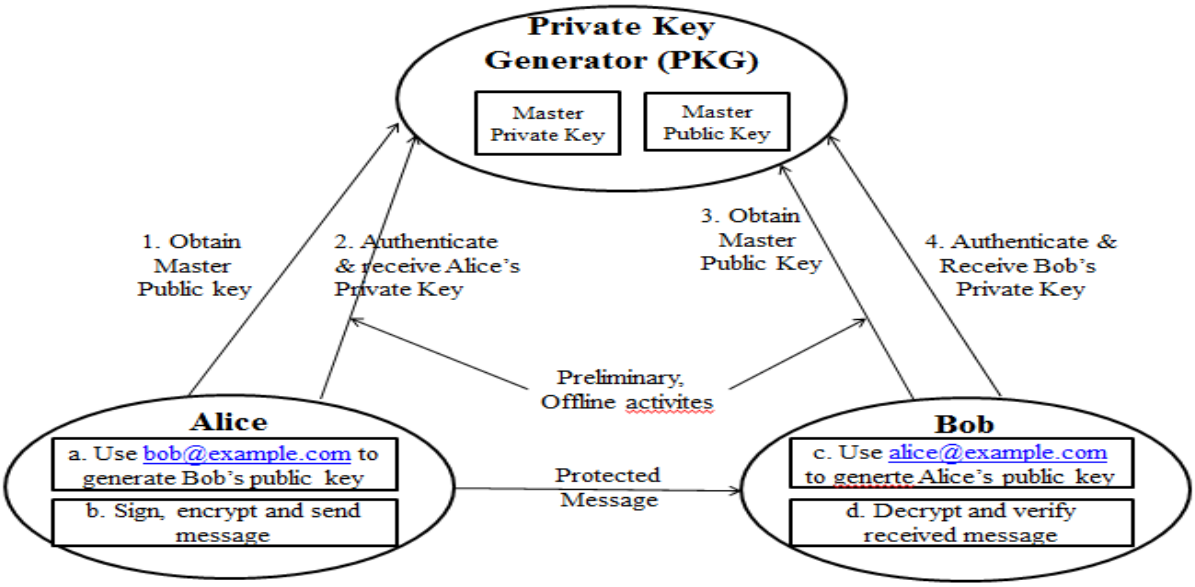

Figure 2.1: Identity Based Encryption

2.5.2 Identity-Based Proxy Re-Encryption

In a proxy re-encryption scheme [7], a proxy can convert an encryption computed under Alice's public-key into an encryption intended for Bob. Such a scheme can be used by Alice to temporarily forward encrypted messages to Bob without giving him her secret key. The fundamental property of proxy re-encryption schemes is that the proxy is not fully trusted, *i.e.*, it does not know the secret keys of Alice or Bob and does not learn the plaintext during the conversion. The proxy and Bob, however, are not allowed to collude, thus it is usually assumed that at least one of the two is honest or that their collusion is preventable or detectable via other means A number of proxy re-encryption protocols have been proposed in the context of public-key encryption[9].

2.5.3 Key Aggregate Cryptosystem

Cloud storage is gaining popularity recently. In enterprise settings, we see the rise in demand for data outsourcing, which assists in the strategic management of corporate data. It is also used as a core technology behind many online services for personal applications. Nowadays, it is easy to apply for free accounts for email,

photo album, and file sharing and/or remote access, with storage size more than 25GB (or a few dollars for more than 1TB). Together with the current wireless technology, users can access almost all of their files and emails by a mobile phone in any corner of the world. Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unexpected privilege escalation will expose all data. In a shared tenancy Cloud Computing environment, things become even worse.

Data from different clients can be hosted on separate virtual machines but reside on a single physical machine. Data in target virtual machines could be stolen by instantiating another virtual machines co-resident with the target one. Regarding availability of files, there are a series of cryptographic schemes which go as far as allowing a third party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owner's anonymity. Likewise, Cloud users probably will not hold the strong belief that the Cloud server is doing a good job in terms of confidentiality.

A cryptographic solution, with proven security relied on number-theoretic assumptions is more desirable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the technical staff. These users are motivated to encrypt their data with their own keys before uploading them to the server. Data sharing is an important functionality in Cloud storage. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of Cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly. However, finding an efficient and secure way to share partial data in Cloud storage is not trivial. Below there is a Drop box.

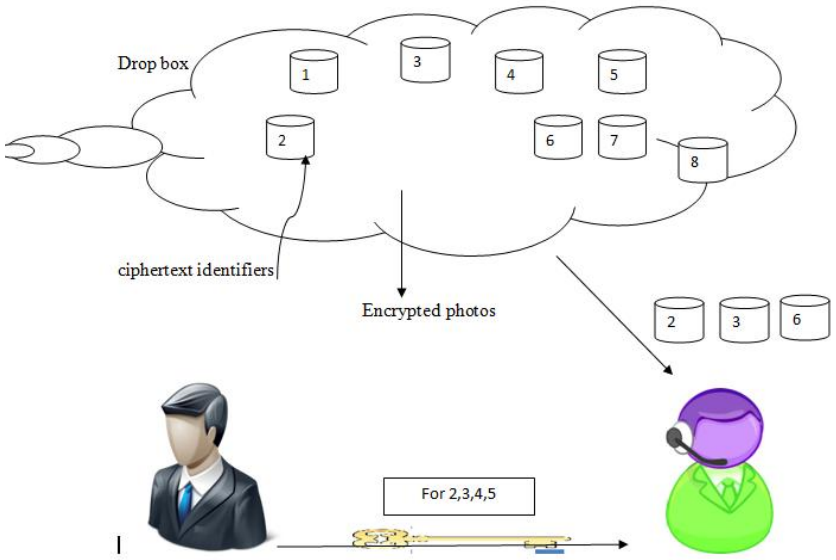

Figure2: key Aggregate cryptosystem

III. System Development

3.1 Proposed system Framework

In key policy attribute based encryption there are five steps global set up, authority set up, keygen, encryption, and decryption. In global set up security level parameter l is generated. In authority set up authority is given. In keygen public key is generated which is used for encryption, after encryption data is spitted in four chunks by using secure hash algorithm it is assigned a hash key for every data chunk which is unique and then for decryption that particular file is searched on cloud then it is downloaded then merged, decrypted using private key.

In traditional cryptography, messages were kept secret, but that approach can't be applied to modern cryptographic systems. Main aim of this paper is to keep data more secure and enhance power of encryption & decryption keys without increasing size of them. The design of basic scheme is inspired from the collusion resistant broadcast encryption scheme 35 proposed by Boneh *et al*. Although their scheme supports constant-size secret keys, every key only has the power for decrypting cipher texts associated to a particular index. From above requirements there is need to devise a new Extract algorithm and the corresponding Decrypt algorithm.

3.3.1 System Model

1. Global Setup (1^l) → params. This algorithm takes as input a security parameter *l* and outputs the system parameters params.

2. Authority Setup (1^l) → (SK_i, PK_i, *Ai*).Each authority Ai generates his secret-public key pair KG *(1^l) (SKi; PKi)* and an access structure *Ai*, for i=1,2; . . .N.

3. Keygen (*Ski,GID,Ai* $_{GID}$) Sk_u^i Each authority Ai takes as input his secret key SKi, a global identifier GID and a set of attributes *Ai* GID, and outputs the secret keys *SKiU*, where $Ai_{GID} = A_{GID} \cap A_i$, A_{GID} and *Ai* denote the attributes corresponding to the GID and monitored by Ai, respectively.

4. Encryption (params, *M, Ac*). This algorithm takes as input the system parameters params, a message M and a set of attributes *AC*, and outputs the cipher text CT, where $A_c=\{ A_c^1, A_c^2 ..... A_c^N \}$ ~$A_c^i \cap A_i$.

5. Decryption: This algorithm takes as input a GID the secret keys cipher text CT and outputs the message M, where I_c is the index set of the authorities A_i such that $A^i_c \neq \{ \}$

3.3.2 Steps of algorithm

Step I: Initialization ()

```
1: Procedure Initialization ( )
2: Initialize cipher index classes with its file size
3: Generate public key by using list of cipher index class, generate random index
4: for each i=0 where i<bytes1.length
5: String j= cipher index class name + random (i);
6: String str= Integer.toBinaryString (j);
7: increment i;
8: end for
9: end Procedure
```

Step II Key generation

```
1: Procedure Summation Keygen ( )
2: Masking public key & byte format
3: for each i=0 upto i<byptes12.lenght
4: int j=bytes12 [i];
5: String s3=Integer.toBinaryString ( j );
6: String temp= temp + Integer.parseInt (s3);
7: S3=toBinaryString (temp);
8: end for
9: end Procedure
```

Step III: Encryption of file

```
1: Procedure Encryption (b)
2: key initialization
3: Takes whole file as msg
4: FOS=new FileoutputStream (out)
5: byte [] b=new byte [8];
6: int i-cis.read (b);
7: while i! = -1 do
8: fos.write (b, 0, i);
9: i=cis.read (b);
10: end while
11: end Procedure 37
```

Step IV: Decryption of file

```
1: Procedure Decryption (b)
2: Encrypt. nit (cipher.DecryptMode, Secret key);
3: cis=new fileOutputStream (fis, encrypt);
4: fos=new fileOutputStream (dec);
```

5: byte [ ] b=new byte [8];
6: inti=cis.read (b);
7: while i! =-1 do
8: fos.write (b, 0, i);
9: i=cis.read (b);
10: end while
11: end Procedure

IV. Figures And Tables

In key policy Attribute Based Encryption first user upload data from client side then that uploaded file is stored on server side. On server side that respective file is encrypted by using attribute based encryption with public key. After encryption that file get stored on cloud in four chunks. At the time of decryption that particular file is downloaded then merged, decrypted in client side with private key which is generated runtime. Decryption is done in the client side. Fig 4.1 describes whole architecture of attribute based encryption.

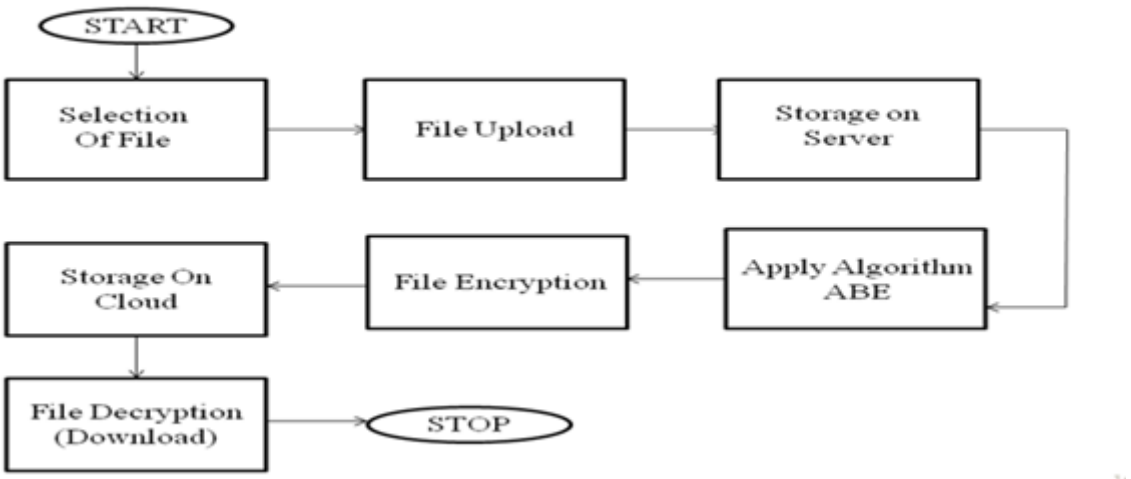

Figure 3: Prposed System Diagram

3.2.1 System Modules

1. **Client Side**: In client side user is going to login, if new one then first registration will be done and then file is uploaded then user will logout. After uploading file is redirected to server side for encryption then for decryption it is downloaded then merged and decrypted from client side again.
2. Client side has following parts

- Login
- Register
- File upload
- Logout

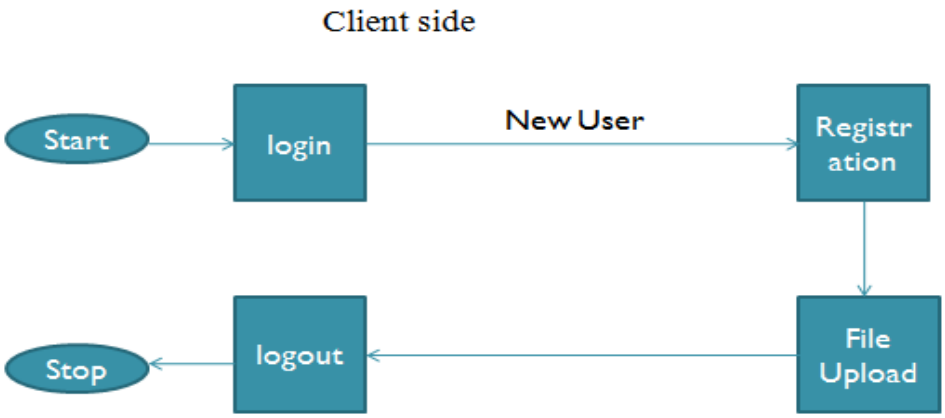

Figure 4: Client Side

Server Side: In server side there are two parts such as master and analysis. In master part user has again two parts file upload on server side and users data. In analysis crystal report is generated as per algorithm, file type, average time is also calculated. file is encrypted with attribute based encryption then after encryption data is stored on cloud.

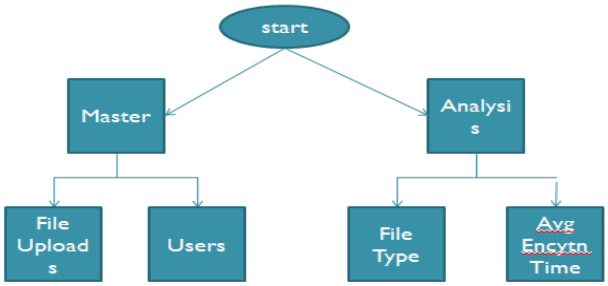

Figure 5: Server Side

Storage Side: In storage side, the data is stored on cloud after encryption in four chunks then for decryption the encrypted file is searched on cloud and then decrypted.

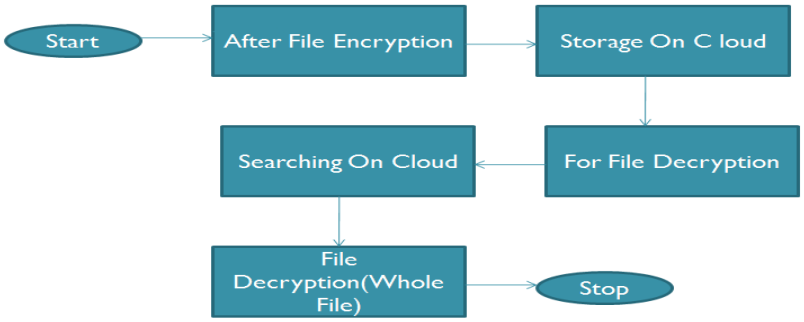

Figure 6: Storage Side

4.1 Comparison between ABE and IBE Algorithm

The identity-based encryption (IBE) is an important primitive of ID-based cryptography. As such it is a type of public-key encryption in which the public key of a user is some unique information about the identity of the user.

Table 2: Factors of IBE and ABE

Factors	ABE	IBE
Algorithm	Asymmetric Algorithm	Asymmetric Algorithm
Encryption	Faster	Slower
Decryption	Faster	Slower
Security	Highly secured	Least Secure

4.2 File type, File size, Encryption and Decryption time

In table 4.2 all values are given file type, file size, encryption time, decryption time algorithms.

Table 1: File type, File size, Encryption and Decryption time

File Type	File Size	ABE		IBE	
		Encr Time	Decr Time	Encr Time	Decr Time
Doc	3Mb	434sec	361sec	640 sec	665 sec
Ppt	3Mb	391 sec	489 sec	680 sec	500 sec
Mp3	3Mb	86sec	76 sec	800 sec	753 sec

4.3 Evaluation of Algorithms

The following graphs shows file type and average encryption and decryption time needed to calculate. Both with IBE, ABE graphs are generated.

4.3.1 Evaluation of IBE

Graphical representation of file type for IBE describes time required for encryption and decryption time by using identity based encryption. In the following graph X-axis represents file type for 3 MB data and Y-axis represents time in seconds.

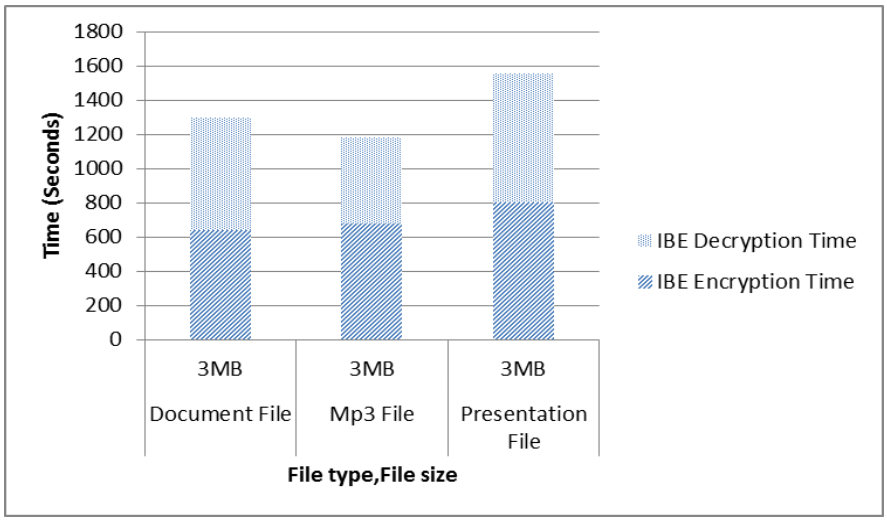

Figure 7: Evaluation of IBE

4.3.2 Graphical representation of file type for ABE

Graphical representation of file type for ABE describes time required for encryption and decryption time by using attribute based encryption. In following graph X-axis represents file type for 3 MB data and Y-axis represents time in seconds.

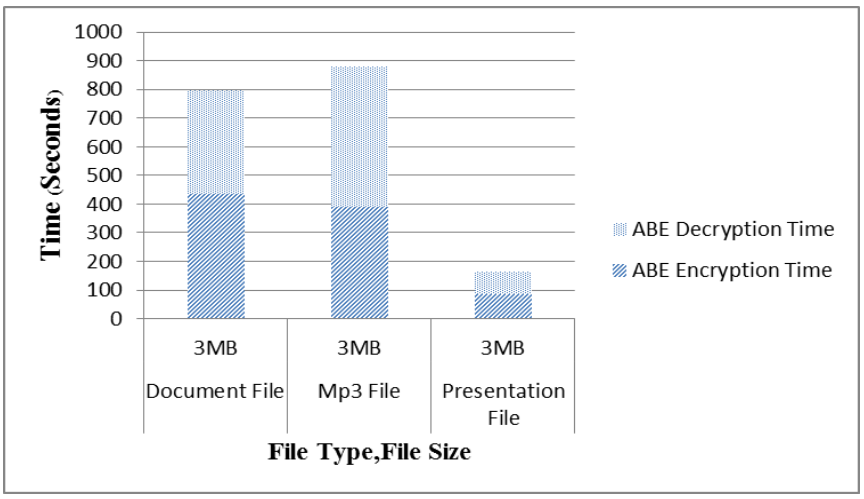

Figure 8: Evaluation of ABE

V. Conclusion

This chapter concludes the work done as part of dissertation and discusses results observed. Moreover it suggests the work done that can be done in near future related to proposed algorithm to enhance the security. In Attribute Based Encryption, data integrity is maintained through Secure Hash Algorithm. It creates hash key for every chunk which is unique. Proposed algorithm gives confidentiality to work as it is a public key encryption algorithm of cryptography. It is also asymmetric i.e. key set is used for data encryption and decryption. Data is stored on cloud means it is available to everyone through Cloud space. Key generated for encryption and decryption is runtime. In this work analysis is there. Encryption and decryption time required by Identity Based Encryption and key Policy Attribute Based Encryption is calculated for comparison purpose.

Both algorithms user can apply on different file type such as pdf, ppt, doc, mp3, jpg and file size. Key policy attribute based encryption reduces encryption time by 66% and decryption time by 33%.

References

Journal Papers:

[1]. Peter Mell, Tim Grance, "*The NIST Definition of Cloud Computing*", Version 15, 10-7-09.
[2]. Robert Bohn ,http://www.nist.gov/itl/cloud/News/September 15, 2016.
[3]. Cheng-Kang Chu, Sherman S. M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H.Deng, "*Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage*", IEEE, 2014.
[4]. Jinguang Han, Willy Susilo, Yi Mu, Jun Yan," Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption" IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23, NO. 11, NOVEMBER 2012 .
[5]. F. Guo, Y. Mu, Z. Chen, and L. Xu, "*Multi-Identity Single-Key Decryption without Random Oracles*," Proc. Information Security and Cryptology (Inscrypt '07), vol. 4990, pp. 384-398, 2007.
[6]. Matthew Green, Giuseppe Ateniese, "*Identity-Based Proxy Re-Encryption*", In the 12th Annual Network and Distributed Syste Security Symposium, pages 29–43, 2005.
[7]. Cong Wang, Qian Wang, Kui Ren and Wenjing Lou "*Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing*" in IEEE INFOCOM 2010.