Novel Approach for improving Security and Confidentiality in Public Clouds using Certificateless Encryption

Chudaman Devidasrao Sukte
Dept. of Computer Science & I.T,
Dr. B. A. M. University,
Aurangabad, India

Emmanuel M., PhD
Department of IT
P.I.C.T, Pune, India

Ratnadeep R. Deshmukh, PhD
Dept. of Computer Science & I.T,
Dr. B. A. M. University,
Aurangabad, India

ABSTRACT
In order to assure confidentiality of sensitive information stored in public clouds, a commonly embrace approach is to encrypt the information before uploading it to the cloud. Since the cloud does not know the keys used to encrypt the information, the confidentiality of the information from the cloud is assured. However, as many organizations are required to enforce fine-grained access control to the information, the encipher mechanism should also be able to provide fine-grained encryption-based access control**.** Proposed work solves the key escrow problem and revocation problem of the previous systems and performs only a single encryption of each data item and minimize the overhead at the data owner.

Keywords
Cloud Computing, Public key crptography, Certificate less, Data sharing.

1. INTRODUCTION
Cloud Computing is used in many small, medium and large sized companies and as many cloud users need the services of cloud computing, the major concern is the protection and security of their data in the cloud. Protecting data is always of importance and because of the crucial nature of cloud computing and the large amounts of complex data it support, the need is even more important. Henceforth, concerns regarding data privacy and security are showing to be a barrier to the wider uptake of cloud computing services.

Cloud is an emblematic word for the internet. So cloud computing refers to a type of computing which is based on the internet. Cloud computing[1] is concerned with applications and services that run on a distributed network using resources that are virtualized and accessed through common Internet protocols and networking standards. It creates an illusion that resources are limitless and that details of the physical systems on which software runs are kept hidden from the user. Cloud computing can be defined as a computing environment where computing needs by one party can be outsourced to another party and internet is used to access the computing power and resources like a database.

Cloud computing is a model[2] for enabling everywhere, appropriate, on-demand network access to the computing resources that can be quickly provided and released with minimum effort or service provider interaction. Cloud model consists of three service models that is Software as a Service, Platform as a Service and Infrastructure as a Service.

Security[5,6] within cloud computing is an particularly intrusive issue because the devices used to provide services do not belong to the user's premises. The users do not have control or knowledge, what could happen to their data. This is a great concern in cases when users have important and personal information stored in a cloud computing service. cloud computing service facilitator must ensure that the user' information is safe because users will not compromise their privacy. This, however, is becoming increasingly challenging because as security developments are made, there always seems to be someone to figure out a way to breach the security and take undue advantage of user information.

Now a day, use of cloud has become very common. Due to benefits of public cloud storage, organizations have been adopting public cloud facility such as Microsoft Skydrive and Dropbox to manage their information. However, for the adoption of cloud storage services, the public cloud storage model should solve the critical issue of data confidentiality. That is, shared sensitive information must be strongly secured from unauthorized accesses. In order to ensure confidentiality of sensitive information stored in public clouds, a commonly adopted approach is to encrypt the data before uploading it to the cloud. Since cloud does not know the keys used to encrypt[7] the data confidentiality of data from cloud is assured. So, it is necessary to build the secure encryption system for sharing of data[3,4,8,12] among public cloud. Existing system in the literature making use of symmetric key based mechanism, it has the problem of high costs for key management. Traditional public key cryptosystem[13] requires a trusted certificate authority to issue digital certificates that bind users to their public keys, and hence the overall certificate management is very expensive and complex.

2. METHODOLOGY
Certificate less[10,11] public key cryptography(CL-PKC), a model for the use of public key cryptography which avoids the inherent escrow of identity-based[9] cryptography and yet which does not require certificates to guarantee the authenticity of public keys. The lack of certificates and the presence of an adversary who has access to a master key necessitate the careful development of a new security model.

A CL-PKC enabled systems[14] still makes use of TTP which is name as the key generating centre (KGC). By way of contrast to the PKG in ID-PKC, this KGC does not have access to entities' private keys. Instead, the KGC supplies an entity A with a partial private key D_A which the KGC computes from an identifier ID_A for the entity and a master key. Note that we will often equate A with its identifier ID_A. The process of supplying partial private keys should take place confidentially and authentically: the KGC must ensure that the partial private keys are delivered securely to the correct entities. Identifiers can

be arbitrary strings. The entity A then combines its partial private key D_A with some secret information to generate its actual private key S_A. In this way, A's private key is not available to the KGC. The entity A also combines its secret information with the KGC's public parameters to compute its public keyP_A. Note that A need not be in possession of S_A before generating P_A: all that is needed to generate both is the same secret information. The system is not identity-based, because the public key is no longer computable from an identity (or identifier) alone. Entity A's public key might be made available to other entities by transmitting it along with messages (for example, in a signing application) or by placing it in a public directory (this would be more appropriate for an encryption setting). But no further security is applied to the protection of A's public key. In particular, there is no certificate for A's key. To encrypt a message to A or verify a signature from A, entity B makes use of P_A and ID_A.

The cloud is trusted to perform the security mediation service and key generation correctly, but it is not trusted for the confidentiality of the content and key escrowing. Attribute Based Encryption (ABE) has been proposed that allows one to have most of the key generation and management functionality deployed in the untrusted cloud as our mCL-PKE scheme does not have the problem of key escrowing and thus the KGC is unable to learn the full private keys of users.

The proposed system architecture of mediated certificate less encryption system is shown in Figure 1. The architecture consists of three entities: data owner, cloud, and users. The data owner possesses sensitive content that it wants to share with authorized users by storing it in the public cloud and requesting the cloud to partially decrypt the encrypted content when users request the data. The cloud consists of three main services: an encrypted content storage; a key generation center (KGC), which generates public/private key pairs for each user; and a security mediation server(SEM), which acts as a security mediator for each data request and partially decrypts encrypted data for authorized users.

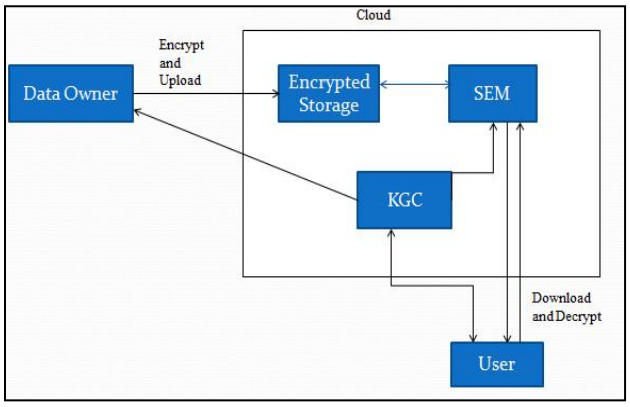

Figure 1. System Architecture

2.1 Proposed Algorithm

The algorithm for basic mediated certificateless public key encryption scheme is given as follows. It is made up of seven tuple described below.

Step 1: Set-Up:

KGC takes as input a security parameter k to generate two

prime numbers p and q. It then performs the following steps:

1. A generator g of Z_p^* with order q is picked.
2. Select x $\in Z_p^*$ uniformly at random and using that value of x calculate y $= g^x$.
3. From the following choose cryptographic hash functions $H_1 : \{0,1\}^* \times Z_p^* \rightarrow Z_q^*$, $H_2 : \{0,1\}^* \times Z_p^* \times Z_p^* \rightarrow Z_q^*$, $H_3 : \{0,1\}^* \rightarrow Z_q^*$, $H_4 : Z_p^* \rightarrow \{0,1\}^{n+k_0}$, $H_5 : Z_p^* \rightarrow \{0,1\}^{n+k_0}$, and $H_6 : Z_p^* \rightarrow \{0,1\}^{n+k_0} \times Z_p^* \times \{0,1\}^{n+k_0} \rightarrow Z_q^*$, where n represents the bit length of a plaintext and k_0 depicts a random bit string.

The system parameters called params are (p, q, n, k_0, g, y, H_1, H_2, H_3, H_4, H_5, and H_6). The master key of KGC is x. The plaintext space is $M = \{0, 1\}^n$ and the ciphertext space is $C = Z_p^* \times \{0,1\}^{n+k_0} \times Z_q^*$

Input: Security parameter K.

Output: System Parameter and Secret Master Key.

Step 2: Set-Private-Key

The entity A sets its private key by selecting $z_A \in Z_q^*$uniformly at

random.

Input: System Parameter and ID.

Output: User Private Key.

Step 3: Set-Public-Key

The entity A computes its public key by $U_A = g^{z_A}$.

Input: User Private Key.

Output: User Public Key.

Step 4: SEM-Key-Extract

KGC picks $s_0, s_1 \in Z_q^*$ and calculates $w_0 = g^{s_0}$, $w_1 = g^{s_1}$, $d_0 = s_0 + xH_1(ID_A, w_0)$, $d_1 = s_1 + xH_2(ID_A, w_0, w_1)$. SEM-key for A is assigned as d_0 by KGC. Finally KGC sets (U_A, w_0, w_1, d_1).

Input: User Registration to KGC and System Parameter, Master key and User Identity.

Output: SEM Key to Each ID.

Step 5: Encrypt

It executes the following steps to perform encryption for a plaintext $M \in \{0, 1\}^n$ for the entity A with identity ID_A and public keys (U_A, w_0, w_1, d_1):

1. First it check's whether $g^{d_1} = w_1 . y^{H_2(ID_A, w_0, w_1)}$. If the consequent result is not valid, encryption algorithm must be aborted.
2. Calculate $\sigma \in \{0,1\}^{k_0}$ and compute $r = H_3(M, \sigma, ID_A, U_A)$.
3. Calculate $C_1 = g^r$.
4. Calculate $C_2 = (M \| \sigma) \oplus H_4(U_A^r) \oplus H_5(w_0^r . y^{H_1(ID_A, w_0).r})$.
5. Calculate $C_3 = H_6(U_A, (M \| \sigma) \oplus H_4(U_A^r), C_1, C_2)$.

Final ciphertext is C = (C1, C2, C3).

Input: System Parameter, User's Identity, User's Public Key, M essage M

Output: Cipher Text or Special Symbol $\perp$ i.e. Encryption Failure

Step 6: SEM-Decrypt

Using the ciphertext C from above step, ID_A, A's public keys (U_A, w_0, w_1, d_1), SEM executes the following steps using the SEM-key d_0:

1. Check whether ID_A is a legitimate user whose key has been revoked or not.
2. Calculate $C_1^{d_0}$.
3. Calculate $C_2 \oplus H_5(C_1^{d_0})$.
4. Check whether $C_3 = H_6(U_A, C_2 \oplus H_5(C_1^{d_0}), C_1, C_2)$.

If the outcome is valid, SEM sends C_1 and $C_2' = (M \| \sigma) \oplus H_4(U_A^r)$ to entity A.

Input: System Parameter, SEM-key, Cipher Text.

Output: Partially Decrypted Message for the user or special symbol $\perp$ meaning decryption failure.

Step 7: User-Decrypt:

Given C_1 from SEM, A using his private key z_A do the following steps.

1. Calculate $C_1^{z_A} = U_A^r$
2. Parse M' and σ' from $M' \| \sigma' = H(C_1^{z_A}) \oplus C_2'$
3. Calculate $r' = H(M', \sigma', ID_A, U_A)$ and $g^{r'}$
4. Check whether $g^{r'} = C_1$

Input: System Parameter, Users Private Key, Partial Decrypted Message

Output: Either fully decrypted message M or decryption failure giving special symbol $\perp$.

The improved approach is similar to the basic approach, the difference is while the encryption and decryption. In the improve approach the data owner encrypts the data encryption key once for a data item and provides some additional information to the cloud so that the authorize users can decrypt the content using private key. In this approach data owner calculates the intermediate key for each authorized user and give it to cloud and which is given to user when they make request.

3. EXPERIMENTAL SETUP

We run our experiments on following system configurations: Operating System: Windows 7, RAM size: 4GB, Processor: Intel I3, and Coding Platform: Java/J2EE, IDE: Eclipse Luna, Wamp server, Back End:MySql 5.5.24.

Certificateless cryptography is a variant of identity based cryptography intended to prevent the key escrow problem. Ordinarily, keys are generated by certification authority or a key generation center that is given complete power and is implicitly trusted. To prevent the complete breakdown of the system in the case of a compromised KGC, the key generation process is split between the KGC and the user. The KGC first generates a key pair, where the private key is now the partial private key of the system. The remainder of the key is a random value generated by the user, and is never revealed to anyone, not even the KGC. All cryptographic operation by the user are performed by using a complete private key which involves both the KGC's partial key, and user's random secret value. The proposed algorithm is implemented using three different hash functions. Below are the tables showing the different values obtained when the algorithm is implemented using different hash function. Each of these tables show the time taken by algorithm when implemented with different hash function. below table 1 consists of four entries showing the encryption time for both basic and improved approach and also the decryption time for both the schemes for different file sizes. Here six different file sizes are taken all in KB for getting encryption and decryption results when SHA-1 is used as hashing function.

TABLE 1. COMPARISION TABLE USING SHA-1 HASH FUNCTION

File Size(in KB)	Basic_enc_time (in sec)	Basic_dec_time (in sec)	Improv_enc_time (in sec)	Improv_dec_time (in sec)
1.46	7	1.7	3	1.6
2.27	8.6	2.5	3.9	2.4
4.06	15.3	4.7	7	4.1
5.64	21	8.5	11	7.2
7.24	31	9.4	15	8.5
9.32	37	13.5	17	11.5

From above table 1 it can be observed that the time taken for decryption is half of the encryption. And also the improved scheme takes less time than the basic approach for different file sizes. below table 2 gives the data when SHA-256 is used, instead of SHA-1. The purpose of using SHA-256 is quite clear as it is considered to be secure than SHA-1 and after 2010, the security policies have made it clear to use SHA-256 over SHA-1, because of its unbrokenness till now. After one or two year may be 2017, browser like chrome will not be supporting SHA-1 so all the system implemented with SHA-1 needs to update accordingly in order to persist with the policies.

TABLE 2. COMPARISION TABLE USING SHA-256 HASH FUNCTION

File Size(in KB)	Basic_enc_time (in sec)	Basic_dec_time (in sec)	Improv_enc_time (in sec)	Improv_dec_time (in sec)
1.46	2.3	1.7	2.1	1.6
2.27	3.5	2.4	3.2	2.2
4.06	7.8	4.3	7.2	4.1
5.64	11.5	5.9	10.5	5.1
7.24	21.8	7.4	13.7	6.8
9.32	30.1	9.6	27.8	8.6

From the above table 2. it is quite clear that SHA-256 performs almost similar to SHA-1 despite having more number of iterations and complication, this performs even better usually if proper hardware resources and computing ability is met.

Below Table 3 shows the values using MD5 algorithm. The reason for picking this algorithm for showing the result is that this is by far considered as the fastest algorithm and this will be clearer by the table below as the entries in it depict the same. But when the security is taken into consideration MD5 is taken as the best algorithm. So, despite having the better result we will consider SHA-1 or SHA-256 over MD5.

TABLE 3. COMPARISION TABLE USING MD5 HASH FUNCTION

File Size(in KB)	Basic_enc_time (in sec)	Basic_dec_time (in sec)	Improv_enc_time (in sec)	Improv_dec_time (in sec)
1.46	2.4	1.5	2.2	1.3
2.27	3.4	2.1	3.1	2.0
4.06	7.2	3.6	6.8	3.3
5.64	10.6	5.2	10	4.7
7.24	14.8	6.6	13.8	6.3
9.32	19.7	8.4	18.1	7.8

From all the three tables above, it is quite clear that the improved scheme performs better than the basic scheme. For the hash function, we use SHA1 as the elementary operation. The basic idea of the hash function construction is as follows. Based on the field of the hash function output, we break the input into multiple blocks, and the block number is dynamically adjusted. For each block, we execute SHA1, convert the output into decimal numbers, say $a_1, a_2, \ldots, a_n$. if the output field is Z^*_q, then we compute $(a_1 \| a_2 \| a_3 \| \ldots \| a_n) \bmod q$, where $\|$ denotes the concatenation operation, to get the final result of the hash function.

Figure 2. shows the time required to perform the encryption operation in both the schemes i.e. in basic mCL-PKE scheme and improved mCL-PKE scheme for different message sizes. It can be seen from the graph, the encryption time increases linearly as the message size increases. Also it is clear that the time required for encryption using improved scheme is lesser than the basic approach. In terms of percentage it is decreased by 50%. As the bit length of q increases, the cost increases non-linearly since the encryption algorithm performs exponentiation operations. The time complexity of the encryption algorithm is O(nt) where n is the number of user, as the user increases the time needed also increases. Space complexity of basic scheme is O(ns) where s is the size of the file and the number of user and improved scheme is O(n) since only one time encryption is needed.

In the proposed work improved scheme was implemented, where the data owner performs only one encryption per data item and creates a set of intermediate keys that allows authorized users to decrypt the data. Figure 3. compare the time to decrypt the data for both the

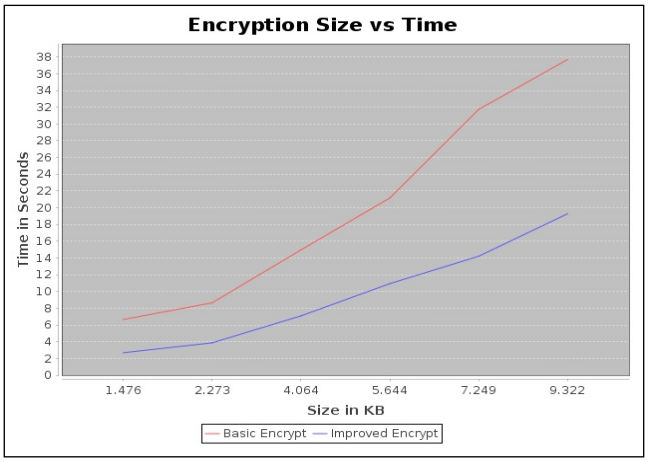

Figure 2. Comparison of Encryption

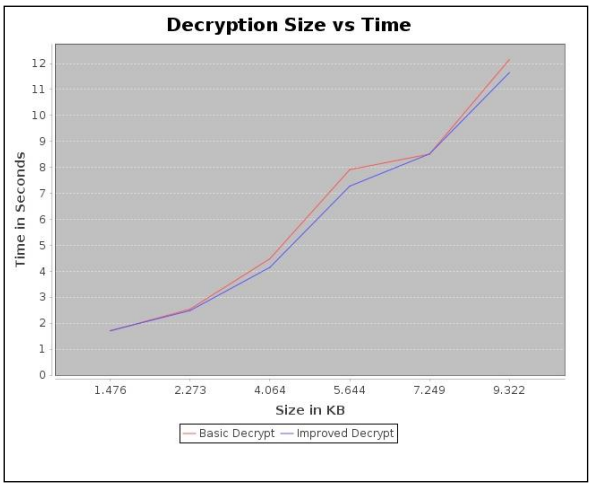

Figure 3. Comparison of Decryption

basic and improved approach. It can be seen from the graph as more users are allowed to access the same data item, the better the improved scheme performs compared to the basic scheme. The improved scheme takes 2% less time as compared to basic scheme.

As the SHA-1 is constructed, in the same way SHA-256 is used with this system. Figure 4. and figure 5. shows the comparison of basic and improved scheme on both the parameter that is encryption and decryption time. SHA-256 is secure, so by using this the main feature of the system remains intact that is improved scheme performs better over the basic one.

MD5 takes on average 35% less time than SHA-1 and SHA-256 for encryption and 40% less time for decryption which is definitely faster
compare to both. But from security point of view SHA-256 is preferred and used.

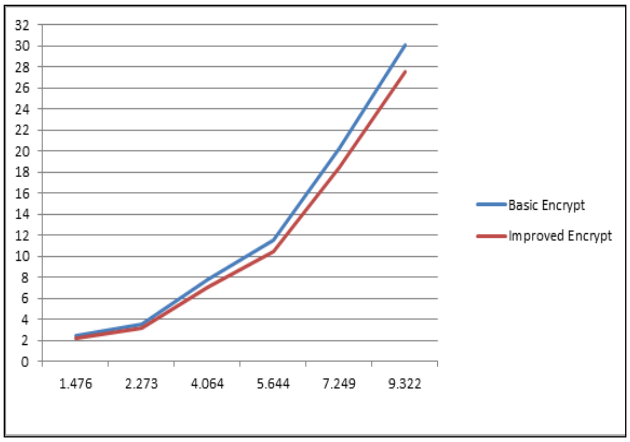

Figure 4. Comparison of Encryption

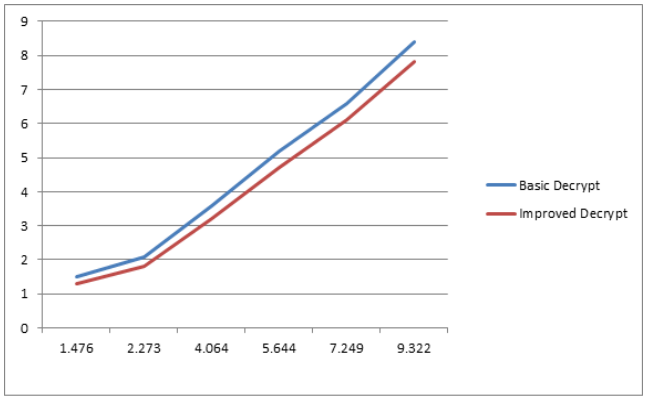

Figure 5. Comparison of Decryption

4. CONCLUSION AND FURTHER WORK

For secure data sharing in public cloud the proposed approach is called Mediated Certificateless encryption. This approach not only increases the level of security but also removes the drawback like key escrow problem. the proposed approach supports immediate revocation and assures the confidentiality of the data stored in an untrusted public cloud while enforcing the access control policies of the data owner. If the basic mCL-PKE scheme is applied directly to cloud computing and if many users are authorized to access the same data, because in real time the number of users will be high so the encryption costs at the data owner can become quite high. In such case, the data owner has to encrypt the same data encryption key multiple times, once for each user, using the user's public keys. this drawback was eliminated in the proposed approach.

5. REFERENCES

[1] Peter Mell, Timothy Grance "The NIST Defination of Cloud Computing," NIST, U.S., 2011

[2] Margaret Rouse "What is public cloud? Definition from WhatIs.com", 2015. [Online]. Available: http://searchcloudcomputing.techtarget.com/definition/public cloud [Accessed:27 - Dec- 2015].

[3] Interroute "What is a Hybrid Cloud?,"2015. [Online].Available:http:// www.interoute.com/cloud article/what hybrid cloud. [Accessed: 27-Dec-2015].

[4] InformationWeek, "Why IT Needs To Push Data Sharing Efforts InformationWeek", 2015. [Online]. Available: http:// www.informationweek.com/services/integration/why it needs to push data sharing effort/225700544. [Accessed: 27- Dec- 2015].

[5] Zhou M, Zhang R, Xie W, Qian W, Zhou A "Security and privacy in cloud computing: a survey". Sixth International conferences on Semantics knowledge and grid (SKG) pp 105-112, 2010.

[6] Danan Thilakanathan, Shiping Chen, Surya Nepa and Rafael A, "Secure Data Sharing in the Cloud", S. Nepal, M. Pathan, Security, Privacy and Trust in Cloud Systems, Springer Verlag Berlin Heildelberg 2014.

[7] Li J, Zhao G, Chen X, Xie D, Rong C, Li W, Tang L, Tang Y, "Fine grained data access control systems with user accountability in cloud computing",IEEE second international conference on cloud computing technology and science(Cloud com), pp 89-96, 2010.

[8] Tu S, Niu S, Li H, Xiao ming Y,Li M,"Fine grained access control and revocation for sharing data on clouds," IEEE 26th international parallel and distributed processing symposium workshops and PhD forum (IPDPSW),pp 2146-2155, 2012.

[9] Dan Boneh and Matt Franklin, "Identity Based Encryption from the Weil Pairing," SIAM Journal on Computing, 32(3): 586-615, 2003.

[10] S. Al Riyami, K. Paterson, "Certificateless public key cryptography," in Proc. ASI ACRYPT, C. S.Liah, Ed Berlin, Germany: Springer, LNCS 2894, pp. 452- 473, 2003.

[11] Sherman S.M. Chow, Colin Boyd, and Juan Manuel G. Nieto, "Security Mediated Certificateless Cryptograph", Public key Cryptology- PKC (LNCS), 3958, pp. 508-524, 2006.

[12] Lei Xu, Xiaoxin Wu and Xinwen Zhang,"CL-PRE: a Certificateless Proxy Re-encryption Scheme for Secure Data Sharing with Public Cloud," ASIACCS'12, May 2-4, 2012.

[13] Garykessler.net, "An Overview of Cryptography", 2015. [Online]. Available: http://www.garykessler.net/library/crypto.html. [Accessed: 27- Dec- 2015].

[14] Seung-Hyun Seo, Mohamed Nabeel. "An Efficient Certificateless Encryption for Secure Data Sharing in Public Clouds". IEEE transactions on knowledge and data engineering, Vol No. 26, Issue: 9, pages: 2107-2119, Sept 2014.

[15] A. Kahate, "cryptography and network security". mc graw hill, pp. 38-198, 2016.