Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Efficient Method for Secure Data Storage and Updation using Additive Homomorphic Encryption on the Cloud

[1]Shubhangi Arora, [2]Monika, [3]Vinod Kumar
[1]M.Tech Student, [2, 3] Assistant Professor
[1, 2, 3] Department of Computer Science & Applications, Kurukshetra University, Kurukshetra, Haryana, India

Abstract- Many organizations are showing their interest in storing data on public clouds. In the last few years, the growth of the data is remarkable. So, the security issues are also associated with storage of data over cloud. This is a major demoralizing factor for probable adopters of the cloud. Hence, cryptographic methods are used that will offer confidentiality and Homomorphic encryption is one of the cryptographic methods that is used in the cloud applications. The objective is to compare the new framework with other previous approaches by using this approach an organization can store its confidential data on the cloud with high security. The new framework is easy to use and does not depend upon security measures taken by the end users. It handles all the cryptographic operations within the trusted infrastructure of the organization then sends the encrypted data to the cloud. The public clouds have not the ability to decrypt the encrypted data that is stored in the cloud. It handles file uploads in an effective manner to reduce the time for uploading the large encrypted files. The objective is to protect and manage the data from the untrusted parties. The proposed paper addresses the confidentiality and produces encrypted storage with the public clouds. This proposed approach uses EcElGamal Homomorphic encryption for storing the user data and uses an updation method for modified file to reduce the time while the transmission of large encrypted files.

Keywords-Cloud Computing, Cryptography, Cloud Storage Security, Fully Homomorphic Encryption, Delta Encoding

I. INTRODUCTION

Cloud computing provides software applications, deployment platforms and computing resources to be made available on-demand pay-as-you-use model. This has drawn a lot of attention towards the cloud computing in recent years. Today, many organizations prefer the cloud for their day to day operations [1]. The data confidentiality and privacy are the main concerns of the clouds. Encryption is used to secure the data in the public cloud. Homomorphic encryption uses RSA, Pailler, ElGamal and EcElGamal algorithm to provide the security of data. This framework should handle file uploads in an efficient manner to reduce bandwidth consumption.
In recent survey, there have been reports of many security breaches of cloud services such as Drop box [2, 3], Last.fm [4, 5], and iCloud [6, 7]. A survey [8] suggests 72% of the IT professionals blame employees for most data breaches, whereas the rest blame the hackers. It also reveals that 32% data was lost while 18% data was stolen by employees. This increases the focus on insider attack on public clouds.
In a recent study [1], 79 % employees agreed for SaaS and 45 % said they use IaaS for their organization. With time many more organizations are going to adopt cloud based solutions because of the benefits of the cloud computing technology over the local traditional IT infrastructure. This will result a large amount of data is moving to the cloud. But as the rate of cloud adoption is growing, the security risks associated with the cloud applications is also growing. In the past, organizations had ignored to store sensitive information in the cloud due to the security risks. But as large amount of data is being stored, generated, processes and consumed in the digital form, the operational cost of a traditional IT infrastructure is becoming very high. Hence companies that deal with sensitive information such as health care records are also considering cloud as an option. Hence the needs of secure cloud storage options are growing.

II. RELATED WORKS

2.1 Homomorphic Cryptosystems

Homomorphic encryption is a method of encryption that allows some special kind of computations to be transferred on cipher text and obtain an encrypted outcome and then decrypted outcome gives the functions performed on the plaintext. For example, one could add the two numbers which is encrypted and another could decrypt that result, without knowing the value of the individual numbers. Today, most systems operate with help of a trusted party.
Homomorphic encryption derived from the theory of privacy homomorphism [2], introduced by the Rivest et al. In their paper, they have explained about performing operations on the encrypted data. The RSA [3] cryptosystem popularized by them and thus provided by the property of partially homomorphic encryption that allows multiplication of encrypted data which when decrypted will give the product of the plaintext. Many methods that include homomorphism properties have been proposed. Another cryptosystem was developed by Taher ElGamal i.e. the ElGamal Cryptosystem [4] also had some homomorphism properties (multiplicative). Paillier cryptosystem [9] was introduced in 1999 but it had additive

homomorphic properties. Homomorphic encryption has strong potential for use in scenarios ranging from multi-party communication to secure computation in cloud systems. In recent years development of fully homomorphic encryption methods [6–8] have attracted a lot of focus into the field of cryptography.

2.2.1 RSA Cryptosystem- RSA [10] is a popular public-key cryptosystem both in practice and theory that is used mostly for digital signature. Its strength lies on the intractability of the integer factorization problem. The basic RSA cryptosystem includes the key generation, encryption and decryption procedure is described below. Make the Public Key available to everyone and keep the Private Key a secret.

Algorithm 1 RSA KeyGeneration ()

```
1: Choose two large prime numbers p and q.
2: Compute $n = pq$ and $\phi(n) = (p - 1).(q - 1)$.
3: Choose an integer e and an integer d, such that e, is co-prime to $\phi(n)$ and $1 < e < \phi(n)$.
4: $ed = 1 \bmod m$.
5: Public Key = (e, n)
6: Private Key = d
```

Encryption Procedure

To encrypt, the sender encodes the message into a numerical form M. Encryption is carried as follows:

$EM = M_e \bmod n$.

Decryption Procedure

To decrypt, the receiver uses the following formula first and then decodes the obtained number to get the intended Message,

$M = EM_d \bmod n$.

Homomorphic Properties:

The RSA cryptosystem presents the property of multiplicative homomorphism as shown below. Consider two cipher texts C1 and C2, which are encryptions of plaintexts M1 and M2.Then according to the encryption function of RSA, C1 can be written as follows,

C1 = P1e mod n, where (e, n) is the public key. Similarly C2 can be expressed as follows,

C2 = P2e mod n, where (e, n) is the same public key. Now multiplying the cipher texts as follows,

C1.C2 mod n = P1e mod n.P2e mod n = P1.P2e mod n

This represents a valid encryption for M1*M2. RSA is one of the oldest cryptosystem with homomorphic property but it is not widely used because the security of this system rests upon the difficulty in factoring n into p and q. Hence the numbers p and q must be very large at least 512 bits and n must be at least 1024 bits to ensure the security. So the long768-bit RSA has been broken and hence higher key sizes are recommended for present and future usage [11].

2.2.2 Paillier Cryptosystem- Introduced in 1999, Paillier cryptosystem [9] is a probabilistic asymmetric algorithm for public key cryptography that is based on the composite residuosity classes problem. An integer z is said to be an nth residue mod n^2 if there exists $y \in Z^*_n{}^2$ such that $z \equiv y^2 \bmod n^2$. The main idea, is that it is hard to determine whether an arbitrary element in $Z^*_n{}^2$ is an nth residue mod n^2 without the underlying factorization. This is called the decisional composite residuosity assumption (DCRA).The key generation, encryption, and decryption procedure, along with the homomorphic property of the cryptosystem is presented below.

Algorithm 2 Paillier KeyGeneration ()

```
1: Choose two large prime numbers p and q randomly such that $\gcd(pq, (p - 1)(q - 1)) = 1$.
2: Compute $n = pq$ and $\lambda = \mathrm{lcm}(p - 1, q - 1)$
3: Select random integer g where $g \in Z^*_n{}^2$
4: $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ where $L(u) = u-1/n$
5: Public Key = (n, g)
6: Private Key = $(\lambda, \mu)$
```

Algorithm 3 Paillier Encryption ()

```
1: Let m be a message to be encrypted and $m \in Zn$.
2: Select random r where $r \in Zn$.
3: Compute the cipher text $c = gm.rn \bmod n^2$.
```

Algorithm 4 Paillier Decryption ()

```
1: Compute the message as $m = L(c^\lambda \bmod n^2).\mu \bmod n$.
```

Homomorphic Properties:

Paillier cryptosystem has the additive property for homomorphic encryption. In this cryptosystem the product of two cipher texts will decrypt to the sum of their corresponding plaintexts. If m1 and m2 are the message to be encrypted, E() and D() are the encryption and decryption function respectively and n is from the public Key, then the additive homomorphism property can be expressed as follows.

D (E (m1, public Key)*E (m2, public Key) mod n2) = m1 + m2 mod n.

Consider c1 and c2 are the cipher texts of the plaintexts m1 and m2. Now following the encryption function of Paillier Cryptosystem,

$c1 = g^{m1}.r^{n1} \bmod n^2$, for some random $r_1 \in Z^*_n{}^2$ and $c2 = g^{m2}.r^{n2} \bmod n^2$, for some random $r_2 \in Z^*_n{}^2$.Now $c1.c2 = g^{m1+m2}.(r_1r_2)\ n \bmod n^2$.This represents a valid encryption for $m_1 + m_2$.

2.2.3 ElGamal Algorithm- ElGamal encryption consists of three components: the key generator, the encryption algorithm, and the decryption algorithm.

```
Algorithm 5 ElGamal KeyGeneration ()
1: Alice generates an efficient description of a cyclic group G of order q with generator g. See below for a discussion on the required properties of this group.
2: Alice chooses an x randomly from {1, ..., q-1}.
3: Alice computes $h=g^x$.
4: Alice publishes h, along with the description of G, q, g as her public key. Alice retains x as her private key, which must be kept secret.
```

```
Algorithm 6 ElGamal Encryption ()
The encryption algorithm works as follows: to encrypt a message to Alice under her public key (G, p, q, g)
1: Bob chooses a random y from (1...q-1), then calculates $c1= g^y$ .
2: Bob calculates the shared secret $s=h^y$ .
3: Bob maps his secret message onto an element m’ of G .
4: Bob calculates $c_{2=}m'.s$.
5: Bob sends the cipher text to Alice $c_1c_2= (g^y ,m'.h^y)= (g^y ,m'. (g^x)^y )$ to Alice.
```

```
Algorithm 7 ElGamal Decryption ()
The decryption algorithm works as follows: to decrypt a cipher text with her private key,
1: Alice calculates the shared secret $s=c_1^x$
2: then computes $m'=c_{2.}s^{-1}$ which she then converts back into the plaintext message, where $s^{-1}$ is the inverse of in the group . (E.g. modular multiplicative inverse if is a subgroup of a multiplicative group of integers modulo *n*).
3: The decryption algorithm produces the intended message, since
$c_{2.}s{-}^{1}=m'.h^y .(g^{xy)\,-1}=m'.g^{xy}.g^{-xy}=m'$
```

2.2.4 EcElGamal cryptosystem with additive property- Assume that here are some elliptic curve C [12] that is described over a finite field F*q* where $q = pn$ is large (and *p* is prime).Suppose that *C*, *q*, and a point $G \in C$ are public and inserted in the system $m \leftrightarrow Pm$. When Alice wants to communicate secretly with Bob, they proceed thus:

```
Algorithm 8 EcElGamal Encryption
1: Bob chooses a random integer number y, and declares the point yG (while y remains secret).
2: Alice chooses her own random integer number x and sends this two points (xG, Pm+x (yG)) to Bob (while x remains secret).[14]
```

```
Algorithm 9 EcElGamal Decryption
1: To decrypt the message, Bob calculates y (xG) from the first part of the pair, then subtracts it from the second part to obtain $Pm+x (yG) -y (xG) = Pm+xyG-xyG =Pm$, and then inserted again to obtain the original message m.
2: Eve, who can only see yG, xG, and Pm + x (yG) must find x from xG or y from yG to make sense of Pm + x(yG), so her problem is reduced to the ECDLP(Elliptic curve discrete logarithm problem), and she is opposed.[14]
```

A technical problem is how to encode characters into points of an elliptic curve (note that $M \in C$).There is a variation of the cryptosystem sometimes called MV-ElGamal (MV stands for Menezes and Vanstone) that avoids this technical problem. In this version a message *M* is
divided into two blocks *m*1 and *m*2 modulo *p*, i.e. F*p*×F*p* is the set of plaintext messages (and
the encoding is very easy).[13]

This is a successful cryptosystem because every operation that Alice and Bob have to carry out (addition and subtraction on the curve) is relatively easy, while the operation that Eve would have to perform to crack the system is extremely difficult (or for real-life villains without the proper resources, perhaps impossible). Now $C1 = ci_{11}, ci12, ci13...ci_{1n}$
and $C2 = ci_{21},ci_{22},ci_{23}...ci_{2n}$, where c_i represents a single block of a cipher text C.As M2 is an updated version of M1,hence it can also be expressed as follows

$M2 = (me_{11} + d1). (me_{12} + d2). (me_{13} + d3)... (me_{1n}+ dn)$,

Where $di = me_{2i} - me_{1i}$, is the difference. Now a single block of cipher text can be expressed as,

c_{1i} = encrypt (mei, public Key).

Further the encrypted difference for a block i can be expressed as,

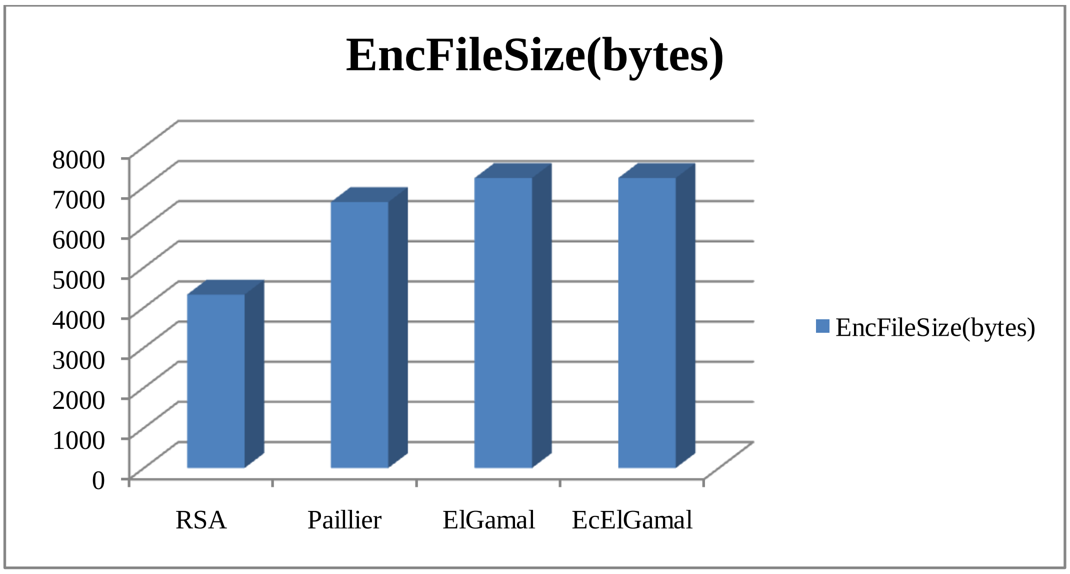
en_di = encrypt (di, public Key).

As the EcElGamal Cryptosystem is used for encryption which presents the additive homomorphism property, Hence decrypt ((c_{1i} * en_di) mod n2, private Key) = $m_{1i} + di = m_{2i}$.

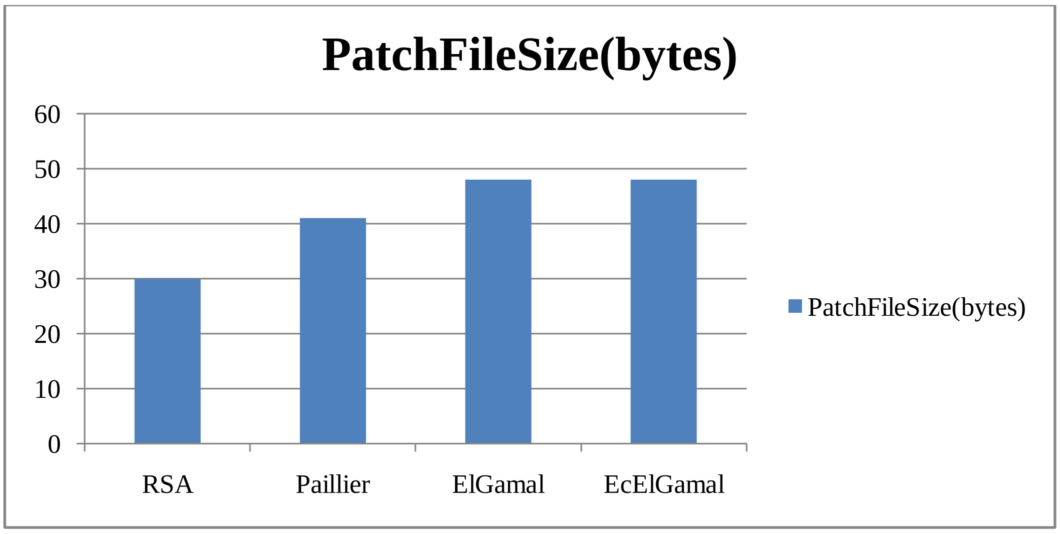
Similarly when this is enforced over all the blocks, one can update the message M1 to M2, by using the cipher text C1 and the encrypted difference. It is important to note that in this proposed framework the cloud providers do not have the private Key and hence cannot decrypt the cipher texts. Yet they can update the content in their encrypted form itself. All interactions with the untrusted cloud involve encrypted contents only.

III. EXPERIMENTS AND RESULTS

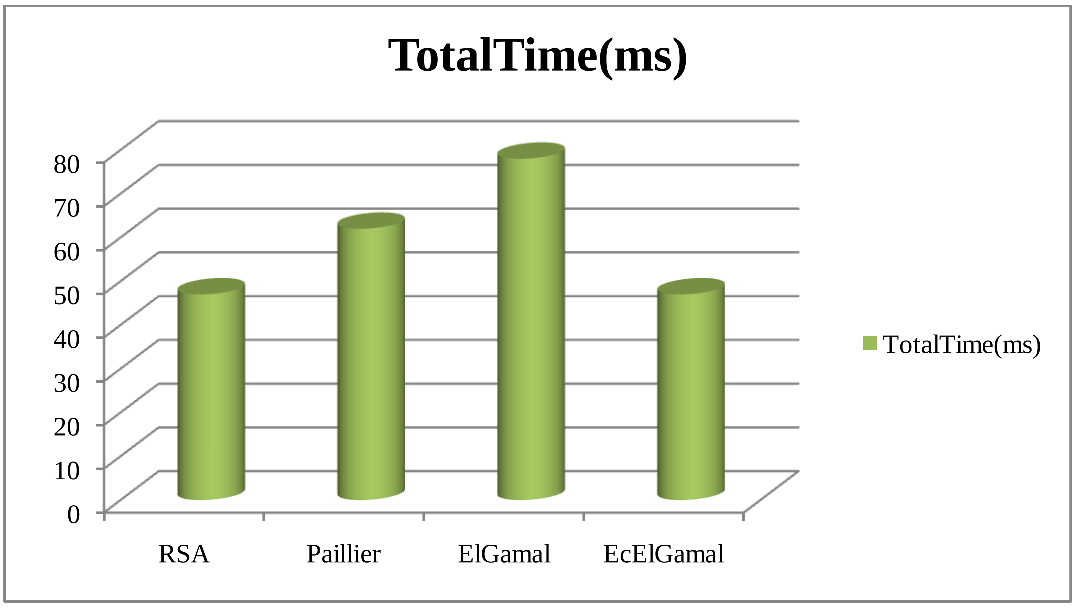
Graphs for 1KB file-



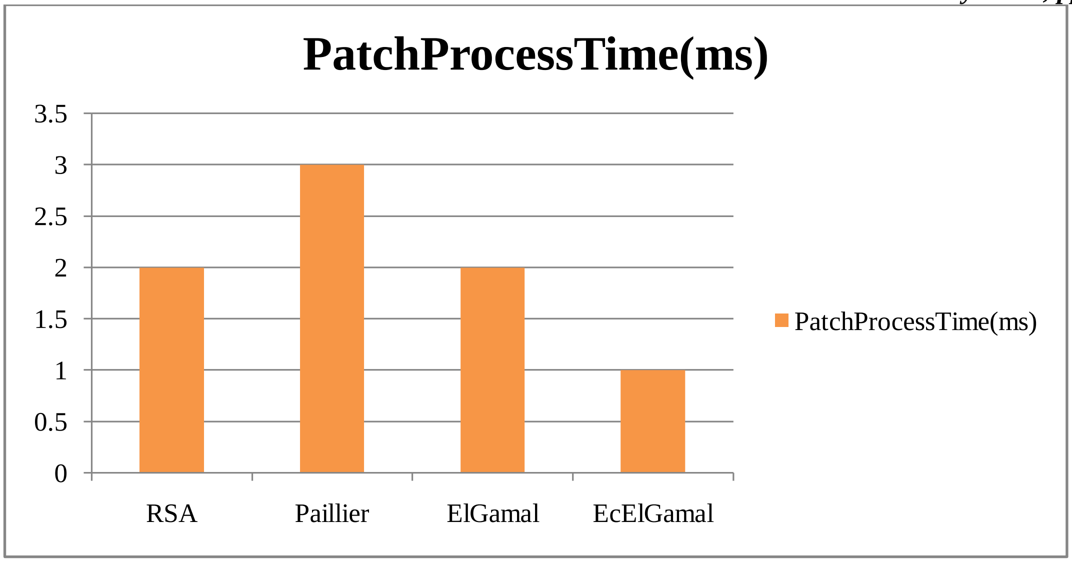
Graph1:EncFileSize(Original Encrypted file)



Graph2: PatchFileSize (After modification of the file)

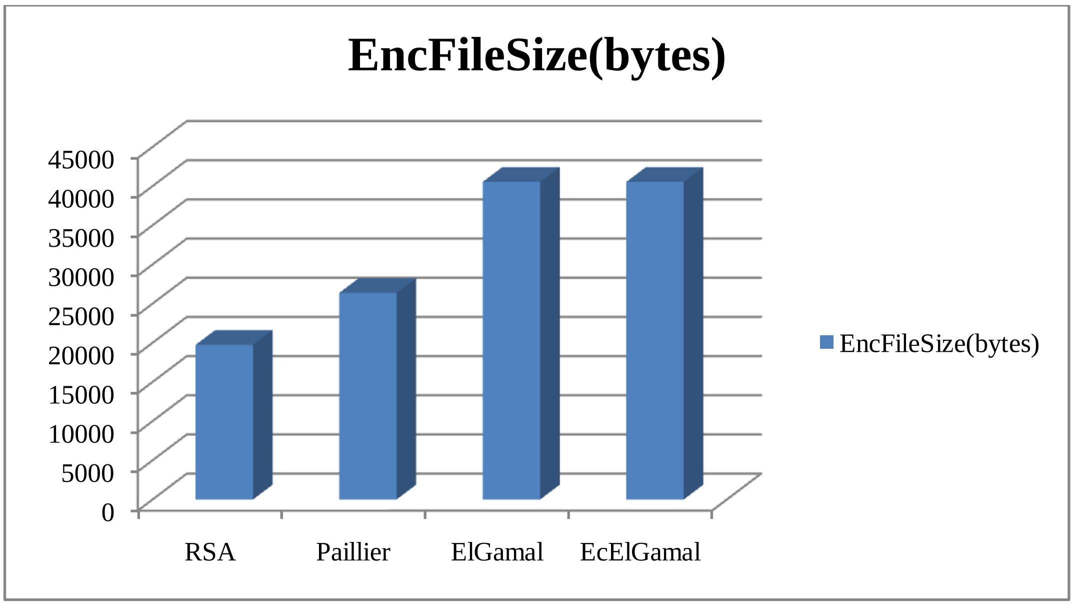


Graph3: Total Time of enc and dec (ms)

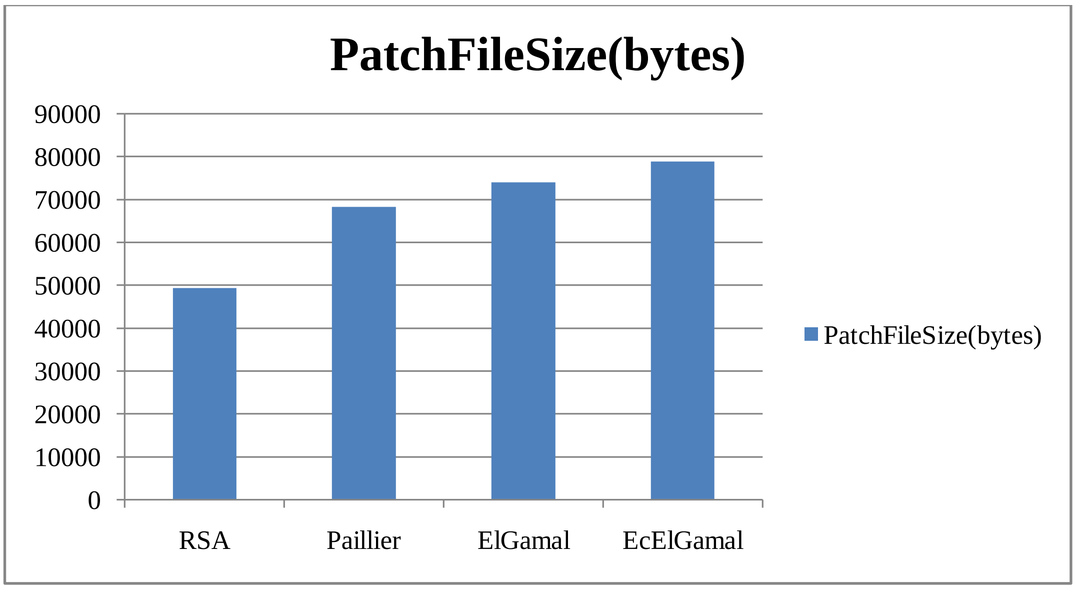


Graph4:PatchProcessTime(After modification of the file)

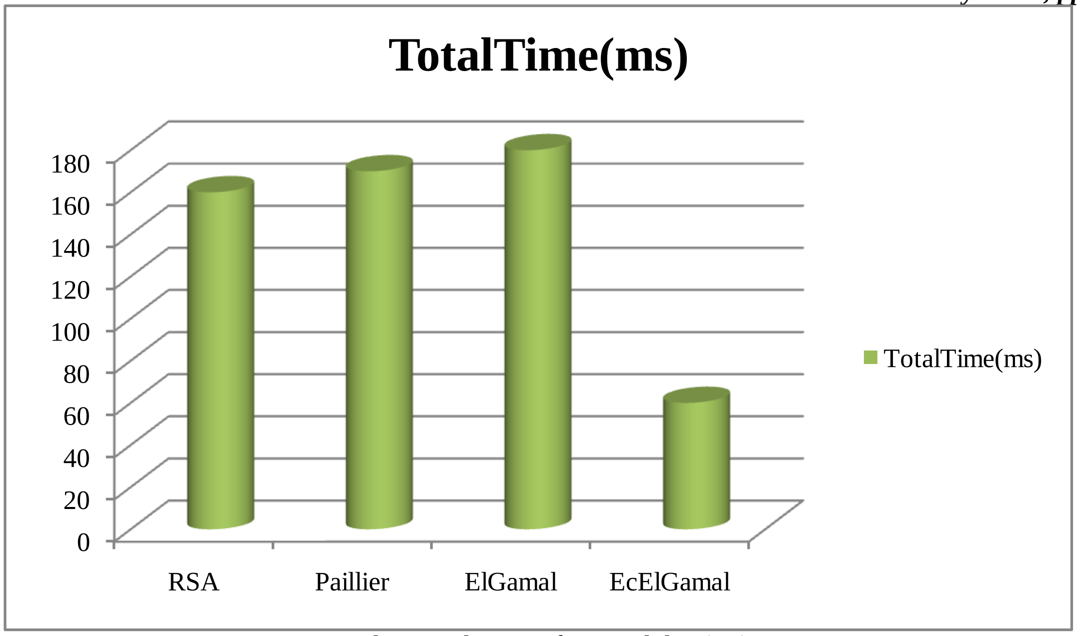
Graphs for 10240 KB file-



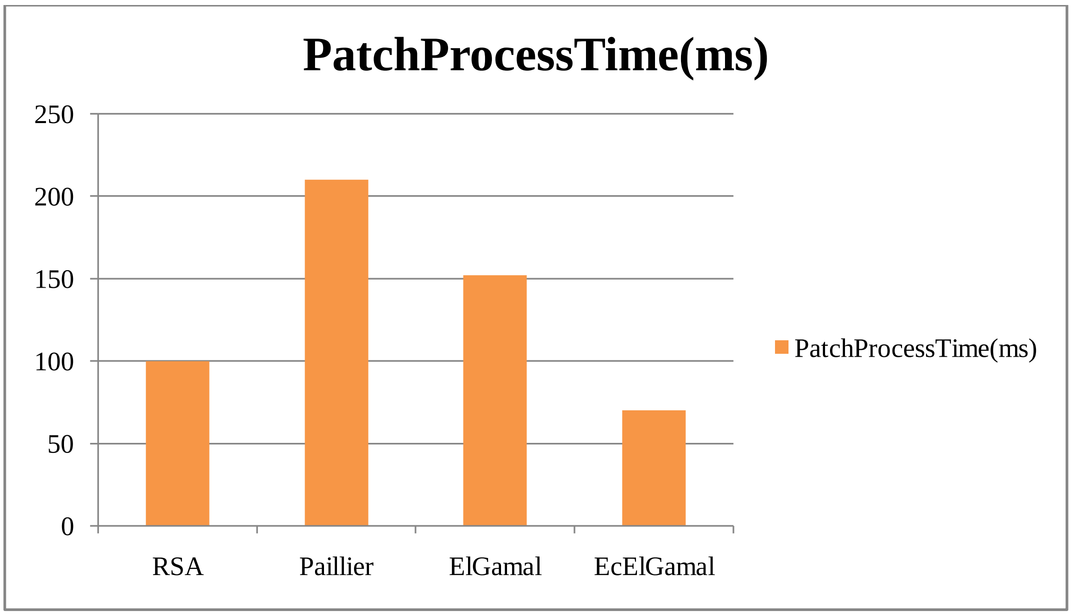
Graph5:EncFileSize(Original Encrypted File)



Graph6: PatchFileSize (After modification of the file)



Graph7: Total Time of enc and dec (ms)



Graph8: PatchProcessTime (After modification of the file)

The above graphs has been proposed that the process time of patch file and total encryption and decryption time for EcElGamal algorithm is less than from all the previous approaches (RSA, Paillier, ElGamal) in case of both for small files (1 KB) and large files (10240 KB) because RSA and ElGamal approach was based on multiplicative homomorphic property but the proposed approach is based on additive homomorphic property. Paillier approach was also based on additive homomorphic property yet EcElGamal approach has best results than Paillier approach so that it is proved from the graphs even if the size of encryption file size and patch file size are large however it will take less encryption and decryption time and patch process time from all the previous approaches. Thus, EcElGamal approach is useful for achieving reduced process time of patch file on the large encrypted files.

IV. CONCLUSION

In summary, EcElGamal model is best for organizations to save and handle their data stored over untrusted public clouds. It also includes homomorphic encryption method with additive homomorphism to update encrypted files, rather the transfer of full encrypted files for every updates was examined. The EcElGamal model has delivered encouraging performance results as compared to other previous solutions in the test environment. Hence the EcElGamal model could be considered for use in certain world scheme.

REFERENCES

[1] Ponemon research study info graphic: Who's minding your cloud? http://www.ca.com/us/collateral/white-papers/na/ponemon-research-study-infographic-whosminding- your's-cloud.aspx, 2013.

[2] Drop box. https://www.dropbox.com/.

[3] Drop box confirms it was hacked, offers users help. http://news.cnet.com/8301-1009_3-57483998-83/dropbox-confirms-it-was-hacked-offers-users-help/.

[4] Last.fm. http://www.last.fm/.

[5] Last.fm password security update. http://www.last.fm/passwordsecurity, 2012.

[6] icloud. https://www.icloud.com/.

[7] Another apple disaster: The icloud gets hacked. http://www.forbes.com/sites/timworstall/2012/08/07/another-apple-disaster-the-icloud-gets-hacked/.

[8] Securing the clouds [info graphic]. http://www.tappin.com/blog/2012/12/cloudsecurity- info graphic/, 2012.

[9] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the 17th international conference on Theory and application of cryptographic techniques, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[10] Peter Mell and Tim Grance. The NIST definition of cloud computing. Technical report, July 2009.

[11] Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thom´e, JoppeW. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman Te Riele, Andrey Timofeev, and Paul Zimmermann. Factorization of a 768-bit RSA modulus. In Proceedings of the 30th annual conference on Advances in cryptology,

[12] https://www.math.hmc.edu/~ursula/teaching/math189/finalpapers/elaine.pdf, accessed on 14.03.2016

[13] https://www.uam.es/personal_pdi/ciencias/fchamizo/asignaturas/cripto1011/ecc.pdf, accessed on 14.03.2016

[14] Shubhangi Arora, Monika and Vinod Kumar(2016).Homomorphic Encryption for secure data storage on the cloud.Ijarcsse,5(6),270-275.