Can Data Be Secured In Cloud Storage And Computational Environment With Homomorphic And Matrix Encryption?

Apoorv Purohit

Submitted as part of the requirements for the degree
of MSc in Cloud Computing
at the School of Computing,
National College of Ireland
Dublin, Ireland.

May 2016

Supervisor Dr Anu Sahni

0.1 Abstract

Cloud Computing is a forthcoming technology that has lured many industries due to its reduced costs, flexibility, scalability, performance, re provisioning of resources, increased profits and many other luxuries. As it shifts the location of data and application software to data center and is not managed by the owner of the data , hence the data is at a vulnerable position for the data owner. Security of data is one of the obstacles for companies to deploy their data on cloud. In this paper we study and compare the two techniques for encryption of data in cloud called as Homomorphic encryption and Matrix encryption. We will also go through the benefits and downsides of both the encryption and eventually compare both the encryption techniques and come up with a unique solution that is expected to be secure and fast.

Declaration

I confirm that the work contained in this MSc project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed .... Date ....
Apoorv Purohit

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past few years Cloud Computing has influenced a lot of people from various fields. Due to its vast application area,it has appealed to almost every area where vigorous computations and large chunks of data are in evidence. Ease of accessibility and dynamic scaling are some of the virtues that have taken the usability of cloud to a wider spectrum. The massive scalable resources that are easily accessible to a cloud makes data storage and data computation much more economical and faster than a client setting up their own infrastructure[5]when clients share their data to a third party, confidentiality of data can be compromised. In this literature review we are going to discuss how privacy and security can be assured with matrix and homomorphic encryption in cloud storage and data heavy computation tasks that are outsourced to cloud service providers. The purpose of this literature review is to analyze matrix and homomorphic encryption and find out how successful these two techniques are when it comes to the security of clients' data in cloud storage and computation. We will also discuss the challenges that are faced when these encryption techniques are implemented. There are many encryption techniques and some have resulted in very secure algorithms but they are limited to quick retrieval of encrypted data in small scale. As the dependence on cloud increases so does the amount of data and we will have to find appropriate cloud storage structures and encryption techniques to handle this [6]. This has resulted in development of different techniques that try to balance the time constraint as well as security constraint. But the fundamental way to encrypt data is that the client/user/data owner encrypts the data before outsourcing it to the cloud[8]. The deployment model that we will be discussing throughout the literature review will be public clouds. To make sure that unsolicited access doesn't happen, the decryption keys are assigned to intended receivers. To add further security the data is also stored in different format than it was before encryption. This makes sure that even if the Cloud Service Provider has the access to the data(ciphertexts), the data itself is not compromised[8]. The proposed algorithm in this paper addresses the issues of computation time constraint, efficiency and practicality. We have taken image data sets and applied the proposed algorithm. We will be using matrix encryption technique on image datasets. We will be using various constraints such as time and ip address of the sender to encrypt the image matrix. Further more to add to the security layer to ensure that the access is only by the intended user we will be using One Time Password service so that access is granted for a specified time, and only to specific email thus reducing the time duration of possible attack.

The paper is divided in four sections. The first section of the paper will examine homomorphic encryption technique in cloud storage environment. The second section will focus on Matrix encryption in computations in cloud. The third section will be a comparison of Homomorphic and Matrix encryption. The fourth section will be the conclusion from our research. Now we proceed to our first section of homomorphic encryption.

Chapter 2

Background

2.1 Homomorphic Encryption in Cloud Storage Environment

[6] describe cloud storage as being derived from cloud computing concept. Cloud storage has increased the convenience level of a client for accessing his/her own data, but it has also given rise to security concerns due to the involvement of a cloud service provider. The outsourcing of the storage model brings along with it various security issues like the involvement of a super user (administrator) who has access to a user's data, hence loss of privacy [7].The need arose for a way to conceal a user's data and also to get the data stored in a cloud without much excessive resource usage. This need gave birth to homomorphic encryption for cloud storage. According to [18] homomorphism is a structure-preserving map between two algebraic structures. The same concept is used in the algorithm as well. Homomorphic encryption enables the placing of data on a cloud server first where it is encrypted without any intervention from a superuser instead of encrypting either at the client site or cloud service provider's site [6].[10] also agree with the high security standard of homomorphic encryption. [10] draw our attention towards the usage of lighter encryption techniques as they focus on image encryption and retrieval that needs to be faster. Although they agree to the fact that homomorphic encryption is robust in security, in the context of image processing they emphasize that faster techniques can compromise security due to the limited encryption involved. So there needs to be a balance between computation time and security robustness to achieve high level of security and fast encryption and decryption time. [6] describe the original

homomorphic encryption algorithm as follows:

Encryption:

Firstly the plaintext is grouped.

1. Two random prime numbers are chosen , P and Q

2. N=P x Q is computed and a random number R is generated

3. The plain text M is grouped in some sections of length L where the size of L should be less than prime number P that we have chosen above. After this step the plaintext is divided into subtexts as M= m1 m2 m3…

4. Now the encryption algorithm Ci=(mi +R X P) mod N and this results in ciphertext as C= c1 c2 c3… , as in step 3 we had divided the text into subtexts hence ciphertext is also as per the values of 'mi' where mi = m1 m2 m3

5. The ciphertext is then sent to the receiver

Decryption:

1. As the ciphertext is received C = c1 c2 c3..

2. P key is used and the algorithm mi= ci mod P is computed

3. After the above computation is carried out we can have our plaintext.

Homomorphic encryption enables the data to be encrypted in such a manner that even if the encrypted data is present on the cloud server, multiple operations can be performed on it. This makes homomorphic encryption more interesting as it gives the client the liberty to perform any changes wanted without the intervention of the cloud service provider. As per [18] in a homomorphic public key encryption scheme if an Evaluate function is applied over a tuple of cipher texts as

Evaluate(pk, C, c1c2c3..ct)

then the result is an encrypted result of circuit C performed on cipher texts. This works as, the key is used on ciphertext for performing the operation on plaintext and then it is encrypted back using the same public key with the output in encrypted format but with the desired operation performed on the data.This shows that homomorphic encryption gives very good security and also the freedom to manipulate data remaining in an encrypted state. In real life scenarios clients' data are updated regularly by them and hence the functionality of manipulation of data in an encrypted state in homomorphic encryption gives them the required flexibility. However, as the functions or methods applied over data are increased so will be the computation time. This point is raised by [10] and for computation and faster response time simple techniques should be used. A balance has to be realized if both security and response time are requirements for a client. Although variations of homomorphic encryption provide more flexibility to a user, the overheads of excessive usage of computational resources and the delay in response time cannot be ignored. As per [22], in fully homomorphic encryption scheme with the increase in arithmetic operations, especially multiplications, the magnitude of noise level is increased. Noise is the extra terms that are generated as the result of arithmetic operations that are not required or are irrelevant. Hence noise not only increases the computation period but also it increases the latency. There are different versions of homomorphic encryption as stated by[20]

1. Additive homomorphic encryption : The additive homomorphic encryption is additive when

Enc(x + y) = Enc(x) X Enc(y)

where 'x' and 'y' are raw data which are encrypted by adding them and than an encrypt function is applied to them such as that it is equal to the product of encrypt function applied on them seperately. An application of such technique is in electronic voting system where the vote itself is encrypted but due to the property of homomorphic encryption the addition of number of votes is done and the sum is decrypted as per[20]

2. Multiplicative homomorphic encryption: The multiplicative homomorphic encryption is multiplicative when

Enc(x X y) = Enc(x) X Enc(y)

where 'x' and 'y' are raw data, multiplied and encrypted with a function that gives the same value as they were encrypted and then multiplied. The computation load increases as the operations on data increase, which is pretty evident in various schemes of homomorphic encryption. This section raises questions about the existence of encryption techniques that do not hamper the speed of computation and are also secure. The next section describes a technique called Matrix encryption that may help address these issues.

2.2 Matrix Encryption in Computations in Cloud

With increasing involvement of cloud in business and scientific applications a need has emerged now for handling vigorous computations that consume a lot of native CPU cycles [5]. This will ease the resource constrained clients as they can offload their intensive computational tasks to cloud as clouds have access to massive computational resources [5].When computation tasks are offloaded to a cloud, clients risk their data security. To increase the level of security, data encryption is needed. To understand Matrix encryption let us first understand the basic concept of matrix. Matrix is basically a concept of mathematics where certain numerical values are arranged in a packet.

For example :

$$A = \begin{pmatrix} 1 & 3 \\ 4 & 5 \end{pmatrix}$$

Here 'A' is a matrix that has three values distributed in two columns and two rows. There can be changes in the structure like multiple rows with multiple columns, which makes it useful for storing data.

There can be various arithmetic operations performed between two matrices, which makes it flexible for clients to perform any changes in data when present in cloud server. The concept of inverse of a matrix makes it useful for encrypting the data present in a matrix. Hence all the three properties are perfect for our purpose of storing, encrypting and manipulating of data.

Now the question arises, why data has to be outsourced to a cloud service provider. This is because of two factors namely computation time and storage. When computation time increases due to numerous calculations, latency increases. To handle that, either the hardware needs to be upgraded or it has to be handled by resources (outsourcing it to a cloud service provider) that perform the computation and send it back to the source in lesser time and lower cost than performing on native machine. As per [12] the computations can be done by a broker and then sent to the storage cloud. The proposal of a broker in between the client and cloud where the data is stored is a good one by [17]. [11] have also proposed a computation scheme where a random matrix of the same order can be added to the original matrix to conceal the original data. Another scheme proposed by [11] is multiplicative splitting, it is just like additive splitting but here the matrix is multiplied with a random matrix of the same order. However, [5] have considered a model where there is no broker and the Matrix is inversed and computed on an outsourced cloud server. [12] place emphasis on security by letting a broker do the encryption for the client whereas [5] focus on the client handling some of the encryption part while the rest calculation part is done by the outsourced cloud server. [5] have also examined algorithms that affect the processing speed of calculations. The main difference however lies in the involvement of a third party by [12].

Although involving a broker as per [17] for performing matrix computations makes the job easier it doesn't necessarily reduce the time of computations in situations where the broker has to follow the protocols of commodity cloud. If the cloud broker cannot follow or adapt to commodity cloud then the actual computation time may increase. In the context of security,[11] have given a very good concept of randomization. The usage of a random matrix that is not generated by any circuit or any function makes the matrix (data) more secure and indistinguishable.

However, [5] have actually addressed the same question that was raised in the first section of this paper. [5] have given a very good balance between security and time of computation. With the help of the permutation technique [5] have been able to reduce the time complexity from O (2.373 to the power of n) to O (2 to the power of n). But this is only one part of the story, as this is for the verification of computed data from the cloud. There are different algorithms for key generation, matrix inversion computation encryption, solving matrix inversion computation, decrypting matrix inversion computation [5]. However these procedures tend to overlook the fact that clients have constrained resources. The key generation and matrix inversion computation are done by the client themselves. Although in context of security this turns out to be a robust feature, we can't ignore the fact that clients have limited computational resources. The fact that [5] have been able to reduce the time complexity can also not be ignored. [5] have been able to satisfy security and latency issues but only in certain conditions. [11] have been able to address the security issue in a very good way by introducing a random matrix in their technique, but a serious weakness is the involvement of a cloud broker, which can be risky if the cloud broker is untrustworthy. [12] also consider encrypting the cloud storage after computation is done. [12] have devised a good technique of splitting the matrices and storing them in two clouds, which may also serve the purpose of redundancy. This technique is followed between three clouds in total and a client. The encryption key for matrices is held by both the clouds via a broker [12]. The fact that the decryption key is only held by the first cloud makes it more secure. As far as the security of matrices is concerned, this scheme is very secure due to the involvement of two more clouds and other varieties of encryption techniques. The distribution of keys makes it more complex which leads us to its downsides. For instance, if any of the matrix manipulation has to be reverted back, then due to the presence of decryption keys only in the first cloud this makes it an example of single point failure. Single point of failure also comes into play when data is handled in two clouds through a cloud broker.

And lastly, due to the numerous computations that are handled by different clouds the communication protocol needs to be robust and hence there is more dependency on it as well. But the fact that it has a randomized matrix,the involvement of multiple clouds via a cloud broker, and multiple encryption keys makes it one of the best encryption techniques if some situations are ignored. These features of Matrix encryption assure more privacy as multiple keys at multiple clouds positioned in a server makes it very difficult for security to be breached. But the privacy factor is more assured in the [12] technique. The technique by [12] assures that the keys are not in the wrong cloud, which in turn assures the keys' security. In this case the single point failure has become a boon for privacy assurance, but it is limited only if every other factor that plays a role in security is working properly. [13] proposed a new way of encrypting the image matrix by dividing the images into different segments. The technique proposed by [13] is based on Color information present in images. The Color information present in images is present in different channels. The most familiar ones are RGB(Red, Green, Blue), HSV(Hue, Saturation, Value) and CMYK(Cyan, Magenta, Yellow, Key) as stated by[13]. Each pixel is represented by a tuple of values RG, GB, BR. These values are changed or transformed to scramble the image.

One more way of Matrix encryption as proposed by [2] is where the image matrix is encrypted using self invertible matrices. Although the calculation of inverse of a matrix involves multiple high computation steps which increases the computation load and also the time constraint but [2] have proposed a new algorithm that computes self invertible matrix. The various predefined functions used by [2] generate key matrices that are applied to the function which are then applied to different sections of the image matrix in 8X8 form. Although [13] and [2] have similar approach but when the time constraint and complexity is compared both are very different. The technique proposed by [2] is faster and less computation tasks are performed whereas [13] divides the image in segments using tuples which increases the required memory. Although as with all the varied techniques that we have analysed until now there are a few downsides to [2] too. As the functions are defined under six cases, hence the generation of key matrix is limited to these six ways and if compromised can reveal the key matrices or the image matrix.

One thing that we have learnt from the variations of matrix encryption is that it gives the benefit of randomness and also provides the data in a packet form which gives the access to each pixel value which makes it perfect for encrypting each pixel in an image. This section has analyzed matrix encryption and argued that some variations of matrix encryption can't satisfy some features in some cases. The next section will compare both the encryption techniques according to their application areas.

2.3 Comparison of Homomorphic and Matrix Encryption

Homomorphic encryption provides security in all phases of data manipulation without data being in its plaintext form. Matrix encryption works in different ways as data can be encrypted on the client site and then for further computations be sent on a cloud server. As per [3] in fully homomorphic encryption apart from keygen, encrypt, decrypt methods of public key encryption the fully homomorphic encryption provides one more method called Evaluate that performs basic arithmetic operations on cipher text. When it comes to Matrix encryption it also encrypts the data on the client site with a public key, but here the key can be a matrix that can be randomized or derived from a function or circuit.

As we discussed in the previous section that when there is a randomized matrix used for computation on data it is more secure as the generation of random matrix doesn't involve a function or method that can be attacked and information be revealed. But for performing arithmetic operations on matrices especially when there is a lot of data to be computed, it may show latency. Whereas in homomorphic encryption all operations/computations are to be performed on an encrypted data set with the evaluate function, in a way all the operations are done on the data directly but in a shield of encryption, without having any latency. However, various schemes of matrix encryption where multiple clouds are used have been proven much faster and secure but with the limitation of single point failure. Matrix encryption has been found more useful when dealing with multimedia data types because of its good compression and fast retrieval ability. Whereas, homomorphic encryption has still been in the theoretical domain. We have come across various positive and negative points of both the techniques and variations in these schemes that only help them to be useful for a particular scenarios. There has to be a perfect balance between security and computation time when choosing a technique. As per [14] the Arnold transformation can be applied for encrypting the image matrix that consists of three components R,G and B. But in Arnold Transformation the image matrix is converted to grayscale which results in a two dimensional array. The Arnold transformation is position shifting of grayscale image pixels as per [14]. The inverse of Arnold transformation decrypts the image. This encryption technique by [14] using Arnold transformation has a lot less computations than homomorphic encryption and has randomized pixel values which makes the image scrambled. But because the image is stored in the same format, this makes it easy to analyse by a hacker, as the values of pixels have a possibility of being recognized. There have been many variations in homomorphic as well as matrix encryption techniques to suit the needs of the client. But a common ground has not yet been found where both the techniques and even other techniques can be compared. The theoretical community is working on innovating new techniques day by day, but there needs to be a common standard setup for security and privacy. The current trend is going in a haphazard manner, new encryption techniques are evolved from older ones by inculcating some changes in them which is a sign of a new field. The client's data remains secure in homomorphic as well as matrix encryption. In some situations where buffering speed matters, a variation of matrix encryption can be used where the amount of data is less. If security is of prime concern than homomorphic encryption can be used as it has robust security features. When it comes to multimedia data handling we can use matrix encryption with two clouds and a broker cloud. But in general, homomorphic encryption is fast as well as secure and has less failure points. It comes down to individual scenarios, when a particular encryption technique may give you the perfect balance of security and computation time. We also need to generalize some norms for cloud security and privacy. Emergence of cloud showed us new possibilities and to convert those possibilities into reality we need to have a systematic path. One of the many complicated factors to find common ground for cloud security is the different cyber laws for different countries. Generalization will surely help theoretical concepts to be applied in a more fruitful manner. While going through different variations of Homomorphic encryption and Matrix encryption we have found variations of Matrix encryption to be effective in encrypting images in terms of time constraint(encryption and decryption times). Some techniques use segmentation of image matrix and then encrypting each segment whereas some use different cloud services to store encryption keys for image matrix to have no single point of failure.

While going through these different variations we have realized a new technique that shall include the benefits of all the variations. This technique will have:

1. Three different layers of encryption

2. Random computations on Matrix values for scrambling the image

3. Exposure of cipher text(encrypted file) for a fixed time duration to minimize the probability of compromising of data

4. Transmission of private key through different service provider

Chapter 3

Design

The design for the proposed algorithm is chosen to have various fields as shown in the figure below

Figure 3.1: Interface

As can be seen we have chosen to include the fields named

Receiver's mobile number: Here, the mobile number of the intended receiver will be entered, for example : 353892170541

Receiver's email id : Here, the receiver's email id will be provided

Sender's email id : Here, the email id of sender or the user who is encrypting the image will be provided

Choose file : Here, the image file that needs to be encrypted will be uploaded to the server

In the figure below it can be seen that an email has been received by the user as entered by the sender previously.

Figure 3.2: Interface

Now the receiver needs to follow the link and download the attached file as that is the encrypted file that needs to be uploaded, with receiver's email id.

Once the email id and the file is validated by AWS(Amazon Web Sevices) server, than One Time Password (OTP) is sent to the mobile number provided by the sender from a different server of Message Bird SMS service using their API. This One Time Password needs to be provided by the receiver to download the decrypted file.

The user can only download the file once within next 20 minutes of receiving the One Time Password from the server.

3.1 Specification

After going through both the encryption techniques and careful consideration the research has identified a need for faster encryption technique that is robust in security as well as faster in encryption and decryption.In today's era where multimedia data usage is increasing day by day, it is becoming increasingly difficult to keep that data secure. As in our research thesis we have identified that there are two issues when encryption of data is concerned. One is the the time constraint and other is security. In this research thesis we come up with a matrix encryption algorithm that uses a random key that is generated by the time and date present on the server.

The encryption algorithm proposed by this research thesis will solve two problems. First problem being the time constraint. The time constraint shall be less when compared to other matrix encryption variations, due to less and faster calculations performed on matrices. As the random matrix shall contain time constraint of any region, hence it shall be extremely difficult to be breached or known. The second part of the random matrix will include the ip(Internet Protocol) address of the client computer.

For example, if the ip address is:

172.230.245.11

and the time at the location of user(encrypt-er) is:

26-Apr-2016-8:59

than the random matrix will be a combination of both:

$$\begin{pmatrix} 172 & 230 & 245 & 11 \\ 26 & 04 & 2016 & 0859 \end{pmatrix}$$

This encryption algorithm performs double encryption of image matrices,hence better security than other matrix single encryption techniques. To add a further layer of security we will be encrypting the image matrix after using the above generated matrix with 128 bit AES (Advanced Encryption Standard) encryption method. 128 bit AES encryption method is considered to be one of the most secure encryption methods as it requires

3.4 x 10^{38}

possible combinations to break a key. The encryption algorithm that will be developed is expected to be faster than all the other matrix encryption techniques and homomorphic encryption.

First the image will be processed with a Python package called OpenCV.This will convert the image into a matrix format as shown below. Here a,b,c... will be algebraic numerals.These can be in a binary format or also they can be different numerals, depending on the images' dataset. In our case we are taking the three channels of color Red, Green, Blue (R,G,B) values in the image matrix.

$$\begin{pmatrix} a[R][G][B] & b[R][G][B] \\ d[R][G][B] & e[R][G][B] \end{pmatrix}$$

This package will also be used for analysing changes in image after the encryption is done.

When compared to [2] the proposed technique doesnot have fixed 6 cases as in the case of [2]. When also compared to [14]Pant et al the proposed technique is not just a result of one function applied to the matrix but is a combination of different functions and that to with multiple points of authentication. Also when compared to simple transformation of an image matrix or scrambled image matrix, the proposed technique has more encryption layers.

3.2 Methodology

There are various methodologies that can be applied. The methodology of implementing other variations of the matrix encryption and comparing the results or comparing the findings of proposed technique with the same data sets used by other research papers.

The source code for implementing other variations is not available, hence the best suited way is to compare the findings with other technique whilst using the same data set.

Chapter 4

Implementation

This section discusses in detail about the implementation details about the encryption algorithm and the frameworks that we are going to use. The application software that we are going to develop will be able to encrypt and decrypt the images. We will be using Python programming language platform for developing our application and also the APIs, Packages, Libraries that are provided with it. We will also be discussing the parts of code that show the resultant encrypted matrix after the first layer and the second layer of encryption of AES

The Algorithm for Encryption:

Step 1: Using OpenCV in Python, convert an Image (bmp,jpeg,png) to a matrix of values in three channels [Red][Green][Blue] with respective integer values

Step 2: Fetch time + ip address + date from server and concatenate it in a string

Step 3: Convert the values of ip address + time + date into ASCII and than in binary format and form a matrix

Step 4: Find the modulus with key length; if 1 than multiply the elements with 2 else multiply the elements with 9

Step 5: Convert the resultant matrix in a string with — sign and dollar sign between elements of rows and columns to mask the difference

Step 6: Encrypt the whole string with AES encryption algorithm and send it to the receiver

The Algorithm for Decryption:

Step 1: Decode from Cipher generated by AES

Step 2: Split the string from — and dollar sign

Step 3: Divide the elements by 9 and 2 respectively

Step 4: Check for valid email address, One Time Password from messagebird service.

4.1 Why Python for building the application

Python is an object oriented programming language, that has support for automatic memory management, multi-threading, socket communication,and also has better portability than other languages across operating systems.

1. Python uses a delegation model and loaders are hierarchical.

2. Python is portable and can be used across many platforms and Operating Systems

3. Polymorphism, inheritance and encapsulation are the three robust and important features of an object-oriented language.

4. Inheritance enables code reuse

5. Object composition is favoured over inheritance.

6. When using implementation inheritance, we have to make sure that the subclasses depends only on the behaviour of the superclass, not the actual implementation. An abstract base class usually provides an implementation inheritance.

7. Python favours interface inheritance to implementation inheritance because it promotes the deign concept of coding to interface and reduces coupling. The interface inheritance can achieve code reuse through object composition.

8. The design by contract specifies the obligations of a calling-method and called-method to each other using preconditions,post-conditions and class invariants.

9. When using Python, we have the access to libraries and packages like numpy, matplotlib, OpenCV which give us the flexibility to use the defined functions for analysis of Videos and images.

10. We can set the initial capacity of a collection appropriately and program in terms of interfaces as opposed to implementations.

11. Pycrypto, Django library packages and being open source gives scope for development at a fast pace.

12. Serialization is a process of writing an object to a file or a stream. Transient variables cannot be serialized.

13. Python performance can be improved by using buffering, minimising access to the underlying hard disk and operating systems.

The above points and benefits about Python make it the best tool for creating our application. It will be portable and can be used on any platform. Also the flexibility of Python will make the application to accept multiple and different format of images to be encrypted and decrypted and also the ability to analyse.

The proposed algorithm for encryption is different in a lot of ways:

Basic Differences From Proposed Algorithm			
Matrix Variation	Encryption Layers	Cloud server	Single point failure
Arnold Transformation	1	1	Yes
Scrambling	1	1	Yes
Proposed Algo	3	2	No

As can be seen in the above table the differences between the proposed algorithm and other variations of Matrix encryption,namingly Arnold transformation and Simple(Scrambling) that are used by [2] are on different levels. First of all the encryption layers are three in the proposed algorithm whereas in the above two are only one. Specially in Arnold transformation there are six cases supporting one layer of encryption because of equations satisfying reversible matrices.

In the point made in Cloud Server the Arnold Transformation and Simple Matrix encryption are dependent on one cloud server whereas the proposed algorithm has 2 cloud servers on which it is dependent.

In the last point the Arnold transformation would not function if the data on the server is compromised and the same with Simple transformation, whereas with the new proposed algorithm there is no single point failure. Even if the data is attacked there are 2 more layers to prevent it from single point failure.

4.2 Use of OpenCV package

We will be using OpenCV Lab application for our first encryption of image to matrix format. There are a lot of benefits for using OpenCV , as it will not only be able to provide the matrix form of an image but also in the testing phase of histogram tests, randomness tests via accessing the encrypted images. This application will also let us know the gray scale testing of an image before and after the encryption is done. As per the OpenCV website which can be accessed here http://opencv.org/ OpenCV is free and open source software for numerical computation, image analysis, video analysis providing a powerful computing environment for engineering and scientific applications.

4.3 Layers of encryption

First layer :

Figure 4.1: Ip+Time+Date Matrix

The first layer of encryption includes formation of a matrix. The elements in the matrix are the binary equivalent values of the IP address , the time when the encryption is being done and the date on which the encryption is being done. Now this matrix will be used to tamper the values of image matrix. As we discussed in the previous section that the values of image matrix include Red, Green and Blue pixels, these will be multiplied by 2 and 9 as shown in the image below

Figure 4.2: Encryption using ip+date+time

These operations when performed on the image matrix than the pixel values change to some random values as shown in the image below:

Figure 4.3: Encrypted Matrix After First Layer

Second Layer : The second layer of encryption includes conversion of matrix values in to a string of values and then encrypting them using AES(Advanced Encryption Standard) of 128 bits before sending it to the intended receiver's email id. Please see the figure below for the code:

Figure 4.4: AES encryption

The actual image pixel values change as shown in the figure below:

Figure 4.5: AES Encrypted Image

Chapter 5

Evaluation

The research thesis will be evaluating the proposed encryption algorithm on the basis of various testings such as :

1. Histogram analysis before and after encryption

The histogram analysis will let us know whether the intensity of pixels has changed or not. If it has changed than how much is the difference. More the difference from unencrypted image better the encryption. We will compare Histogram Analysis with different formats(jpeg,bmp,png) of same 256X256 resolution images that have been experimented on by [2]. Let us take a look at the images the histograms of our unencrypted image and encrypted image

Figure 5.1: Histogram Lena Raw png

The above image shows the histogram of unencrypted image in PNG format. Now as we encrypt the image with the first layer of encryption we get

Figure 5.2: Histogram Lena Enc png

Now let us compare the encrypted image histogram with that by [2]

Figure 5.3: Chottaray Histo encrypted

As seen above the encrypted image histograms from this paper's proposed algorithm and the one in [2]. Both of them show similar pixel values. But the main difference that were observed were in Black and white BMP format images. But first let us see the histogram results for colored JPG images from our proposed Algorithm.

Figure 5.4: Histogram Lena colored

Figure 5.5: Histogram Lena Enc

As we can see here that a shape similar to Parabola is revealed in histogram of colored jpeg image which shows that the values can be predicted if general equation of Parabola is used. The standard form is

(x - h)2 = 4p (y - k),

where the focus is

(h, k + p)

and the directrix is y = k - p.

Now we check our results with BMP format images. Let us see the results of BMP images histogram that is encrypted by our proposed algorithm

Figure 5.6: Histogram BMP Enc Lena

Now let us see the histogram of unencrypted image of BMP format

As we can see the values are very different and hence cannot be predicted

Now let us compare our encrypted histogram with with the one with [2].

As we can see that the values of pixels are in a fix. All the pixels have similar values. We should not forget that the histogram that we have achieved is after the first encryption, so there is one more layer of encryption which will be applied to our data set, whereas for [2] that was the only layer of encryption. Now we will be taking a look at the file that is received by the intended receiver which can be seen in figure below. The AES encrypted string is the result of AES encryption applied on encrypted matrix.

2. Timing constraint after the proposed algorithm

Figure 5.7: Histogram BMP Unenc Lena

Figure 5.8: Histogram BMP Enc Lena

The time taken by the proposed algorithm is compared to the time taken by the [2] algorithm as both have been tested on the same image of png format and in grayscale with 256X256 resolution.

As mentioned in [2] the time taken to encrypt the same image is

0.960 seconds

whereas when the proposed algorithm was tested it revealed the timings of

0.79 seconds

which is significant when large data have to be processed. This is also only the computation of one layer of encryption. When another layer of AES technique is added than the time increases. As it has been mentioned [2] the AES technique takes

1171.609 seconds to encrypt the same data set hence it significantly increases the total encryption time.

We have seen that total time constraint is more than [2].

Hence, the proposed technique has a faster encryption time for the first layer of encryption. The proposed technique is also better in terms of security for BMP and grayscale set of images. But before sending the image to the server the proposed technique encrypts the image with AES technique which takes 1171.609 seconds, hence increasing the time taken for both layers of encrytption to execute. This added layer of encryption and One Time Password authentication increase the security parameters making the proposed technique secure.

Figure 5.9: AES encryption applied on Encrypted Matrix

Chapter 6

Conclusions

We have compared the histograms of the same unencrypted and encrypted images with [2] of 256X256 resolution. We have noted that the histograms are not in a fixed line as witnessed in [2] after encryption. The histograms have revealed a new insight, when the colored images of PNG are encrypted using the proposed algorithm then a shape of Parabola is revealed which can be predictable than [2] but when it comes to BMP type and Grayscale(Black and White) images than the proposed algorithm is better in terms of security as the histogram values are unpredictable and random.

Equation of a Parabola:

$(x\text{-}h)2 = 4p(y\text{-}k)$

In this equation using trying and testing the values of Focus and Directrix can be revealed which will reveal set of values that will satisfy the equation, hence the attacker will be able to get a set of possible images. But we should remember that this is the case with JPG and PNG colored images and not with BMP and Black and White images. In BMP and Grayscale images the encryption has achieved the same unpredictable histogram as [2] and that too in a faster time. We should also remember that this is the result of one encryption layer, after this layer we encrypt the images with AES technique which makes this technique more secure.

We have realized that the security of images and time constraint is better than [2] for BMP and Grayscale images.

6.1 Future Work

As per the results of the images tested using our new proposed algorithm we have realized that the new technique can be used for BMP type images and be further applied to image sharing applications like Instagram, Snapchat and Whatsapp for better security and faster processing. We would also like to take our research further and increase the limit from 2560000 pixel density to high density images.

Bibliography

[1]

[2] S. K. Chhotaray, A. Chhotaray, and G. S. Rath. A new method of generating public key matrix and using it for image encryption. In *Signal Processing and Integrated Networks (SPIN), 2015 2nd International Conference on*, pages 453–458, Feb 2015.

[3] S. Dara. Cryptography challenges for computational privacyin public clouds. In *Cloud Computing in Emerging Markets (CCEM), 2013 IEEE International Conference on*, pages 1–5, Oct 2013.

[4] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[5] Xinyu Lei, Xiaofeng Liao, Tingwen Huang, Huaqing Li, and Chunqiang Hu. Outsourcing large matrix inversion computation to a public cloud. *IEEE Transactions on Cloud Computing*, 1(1):1, 2013.

[6] Jian Li, Sicong Chen, and Danjie Song. Security structure of cloud storage based on homomorphic encryption scheme. In *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, volume 01, pages 224–227, Oct 2012.

[7] Jian Li, Danjie Song, Sicong Chen, and Xiaofeng Lu. A simple fully homomorphic encryption scheme available in cloud computing. In *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, volume 01, pages 214–217, Oct 2012.

[8] Qin Liu, Guojun Wang, and Jie Wu. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. *Information Sciences*, 258:355 – 370, 2014.

[9] Qin long HUANG, Zhao feng MA, Yi xian YANG, Jing yi FU, and Xin xin NIU. Secure and privacy-preserving {DRM} scheme using homomorphic encryption in cloud computing. *The Journal of China Universities of Posts and Telecommunications*, 20(6):88 – 95, 2013.

[10] Wenjun Lu, A.L. Varna, and Min Wu. Confidentiality-preserving image search: A comparative study between homomorphic encryption and distance-preserving randomization. *Access, IEEE*, 2:125–141, 2014.

[11] M. Nassar, A. Erradi, and Q.M. Malluhi. Practical and secure outsourcing of matrix computations to the cloud. In *Distributed Computing Systems Workshops (ICDCSW), 2013 IEEE 33rd International Conference on*, pages 70–75, July 2013.

[12] Mohamed Nassar, Abdelkarim Erradi, Farida Sabri, and Qutaibah M. Malluhi. Secure outsourcing of matrix operations as a service. *2013 IEEE Sixth International Conference on Cloud Computing*, 0:918–925, 2013.

[13] R. Orduna, A. Jurio, D. Paternain, H. Bustince, P. Melo-Pinto, and E. Barrenechea. Segmentation of color images using a linguistic 2-tuples model. *Information Sciences*, 258:339 – 352, 2014.

[14] Ashish Pant, Arjun Arora, Suneet Kumar, and R.P Arora. Sophisticated image encryption using opencv. *International Journal Of Advanced Research In Computer Science And Software Engineering*, 2, 2012.

[15] Yong PENG, Wei ZHAO, Feng XIE, Zhong hua DAI, Yang GAO, and Dong qing CHEN. Secure cloud storage based on cryptographic techniques. *The Journal of China Universities of Posts and Telecommunications*, 19, Supplement 2(0):182 – 189, 2012.

[16] Mark D. Ryan. Cloud computing security: The scientific challenge, and a survey of solutions. *Journal of Systems and Software*, 86(9):2263 – 2268, 2013.

[17] Sandeep K. Sood. A combined approach to ensure data security in cloud computing. *Journal of Network and Computer Applications*, 35(6):1831 – 1838, 2012.

[18] Li Tao, Ye Xiaojun, and Wang Jianmin. Protecting data confidentiality in cloud systems. In *Proceedings of the Fourth Asia-Pacific Symposium on Internetware*, Internetware '12, pages 18:1–18:12, 2012.

[19] M. Tebaa, S. El Hajji, and A. El Ghazi. Homomorphic encryption method applied to cloud computing. In *Network Security and Systems (JNS2), 2012 National Days of*, pages 86–89, April 2012.

[20] M. Tebaa, S. E. Hajji, and A. E. Ghazi. Homomorphic encryption method applied to cloud computing. In *Network Security and Systems (JNS2), 2012 National Days of*, pages 86–89, April 2012.

[21] Lifei Wei, Haojin Zhu, Zhenfu Cao, Xiaolei Dong, Weiwei Jia, Yunlu Chen, and Athanasios V. Vasilakos. Security and privacy for storage and computation in cloud computing. *Information Sciences*, 258(0):371 – 386, 2014.

[22] Xiaojun Zhang, Chunxiang Xu, Chunhua Jin, Run Xie, and Jining Zhao. Efficient fully homomorphic encryption from {RLWE} with an extension to a threshold encryption scheme. *Future Generation Computer Systems*, 36(0):180 – 186, 2014. Special Section: Intelligent Big Data Processing Special Section: Behavior Data Security Issues in Network Information Propagation Special Section: Energy-efficiency in Large Distributed Computing Architectures Special Section: eScience Infrastructure and Applications.