Introduction to Cryptography
SC402

Project Report

Image Encryption using Paillier Homomorphic Encryption

Dhairya Somaiya (201901047),
Dharmik Patel (201901300),
Mohil Desai (201901301),
Fenil Kamdar (201901418)

supervised by
Manish Gupta

Abstract

With the increase in communication using images, it is necessary to store the images in server in an encrypted manner to prevent theft of user data in case of attacks. To solve this, we have implemented an image encryption and decryption system using Paillier Homomorphic Encryption. We have also analyzed the results obtained from test images.

Contents

1 Introduction

Due to the huge rise of social media platforms in recent years, messages containing text, audio, images and video are exchanged at an enormous rate between people everyday. To serve the purpose of privacy of the user's data, secure and efficient algorithms are needed to encrypt and decrypt the messages of people. Many people have come up with different cryptography algorithms which help in encrypting and decrypting different forms of media. In this report, we are going to discuss the encryption and decryption of images. To encrypt and decrypt images, various types of cryptography like DNA cryptography, symmetric cryptography and elliptic curve cryptography are used. These cryptosystems convert the original image to a form which cannot be decoded easily to ensure that no one can interpret the contents of the image. We will discuss how Paillier cryptosystem alongwith homomorphic encryption can be used to encrypt and decrypt images.

2 Problem Statement

With the rising popularity of image sharing and image editing services, huge number of images are uploaded on such sites everyday. The security of these sites and their servers cannot be trusted as the implementation details are hidden and attackers can access the servers containing the images of the user leading to leak of private data. Thus the objective is to implement a cryptosystem which will encrypt images before storing them on the server and will decrypt the image from server when the user requests the image.

3 Literature Review

Singh, Laiphrakpam Dolendro, and Khumanthem Manglem Singh. "Image encryption using elliptic curve cryptography." Procedia Computer Science 54 (2015): 472-481.[1]

Singh, Laiphrakpam Dolendro, and Khumanthem Manglem Singh have implemented an image encryption algorithm using elliptic curve cryptography. As performing operations of cryptography on every pixel of the image can be very computationally expensive and time consuming, they first create groups of the pixels of the image into a single integer. Then these integers are sent to the ECC system as a plain text message. The ECC system generates a cipher text for each group

integer. The cipher texts are padded through a mechanism and a cipher image is obtained which is sent to the receiver. To obtain the group of pixels back during decryption, the values of the cipher image are scaled down to the 0-255 range by the decryption system after which the original image is obtained by the receiver. Their results show that the frequency of pixels in the cipher image is evenly distributed even if the plain images have varying frequency distribution of pixels. Thus the probability of each pixel in the cipher image getting mapped to a pixel in the original image will be almost equal.

M. Ashtiyani, P. M. Birgani and H. M. Hosseini, "Chaos-Based Medical Image Encryption Using Symmetric Cryptography," 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008, pp. 1-5, doi: 10.1109/ICTTA.2008.4530291.[2] M. Ashtiyani, P. M. Birgani and H. M. Hosseini have implemented an image encryption system using chaos functions and symmetric cryptography. The encryption system first scrambles the pixels of the image by using chaotic mapping. The authors have used Cat Map Chaotic Mapping for this purpose. The pixels of the image are scrambled for n rounds in the first stage. Then the image is sent to the next stage where diffusion in the pixels takes place. This diffusion is performed with the help of the S-AES algorithm. Their results show that the histogram of frequency distribution of pixels in the original image, scrambled image and scrambled plus encrypted image differ highly which suggests that the image has been encrypted efficiently.

Manish Kumar, Akhlad Iqbal, Pranjal Kumar, A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie–Hellman cryptography, Signal Processing, Volume 125, 2016, Pages 187-202, ISSN 0165-1684[3]

Manish Kumar, Akhlad Iqbal, Pranjal Kumar have implemented an image encryption and decryption algorithm which uses DNA cryptography and elliptic curve Diffie-Hellman cryptography. For the encryption of the original image, the image is divided into three layers of RGB (Red, Green and Blue) and DNA sequence matrix is generated. Thereafter, DNA addition and scrambling is performed on the image pixels to introduce randomness between the pixel values. The layers of the image are then interleaved. In the next stage, the obtained image is passed to the ECDHE system and the resultant image is again interleaved in the final stage to obtain the final encrypted image.

O'Keeffe, Michael. "The paillier cryptosystem." Mathematics Department April 18

(2008): 1-16.[4] The above mentioned paper gives us an idea on how the Paillier Cryptosystem works, briefs about some of its interesting properties and how this cryptosystem can be used by working on these properties. It discusses how the Carmichael's Theorem is useful in encryption and decryption of the Paillier system. Finally it discusses the multiplying property, how to change cipher text without the plain text, and property of powers. It gives us the corollary from which we find out how the three properties are related. Finally it takes a real life example of electronic voting and how the system is implemented using these properties.

4 Proposed Algorithm

We use the Paillier Homomorphic Encryption System to perform different operations on the images. Paillier cryptosystem requires a large prime number for the generation of secure keys. We first generate the public key for the cryptosystem by multiplying two large prime numbers of a specified bit length and generating a random number. The private key is generated through a function which takes two prime numbers and the product of those two primes as a parameter. To encrypt the image, each pixel of the image is passed as an integer value to the encryption algorithm. During the decryption process, the pixels of the cipher image are decoded through modular operations.

4.1 Generating Public Keys

To generate the public key, we take two integers a and b of equal bit length and obtain the product n of both the integers. After obtaining the product, we find an integer g in $Z^*_{n^2}$ such that its order is multiple of n. As the integers a and b are of equal length, we can take $g = n + 1$. The tuple (n,g) forms our public key. The flowchart which represents the process to generate public keys is depicted in Fig.[2]

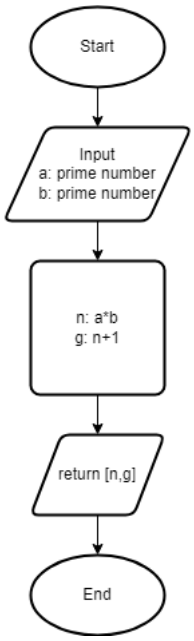

Figure 1: Flowchart to generate public key

4.2 Generating Private Keys

To generate the private key, we pass prime numbers a and b and their product n as parameters to the private key function. We first calculate the $\lambda = lcm(a-1, b-1)$. Then we find the multiplicative inverse μ of λ in the field of Z_n. The tuple (λ, μ) forms our private key. The flowchart in Fig.[**??**]shows the process to generate a private key.

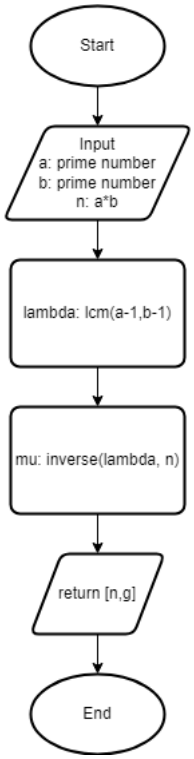

Figure 2: Flowchart to generate private key

4.3 Encrypting the image

We input the public key and the image to be encrypted as parameters to the encryptImage function. The image is converted to an array of pixels. We iterate through each pixel in the array and encrypt the pixel using an encryption system. The encryption system uses Paillier's cryptosystem to encrypt the plaintext values to cipher text. For doing so, The cipher image array is then returned by the encryption function. The flowchart in Fig.[3] shows the process to generate an encrypted image from the original image.

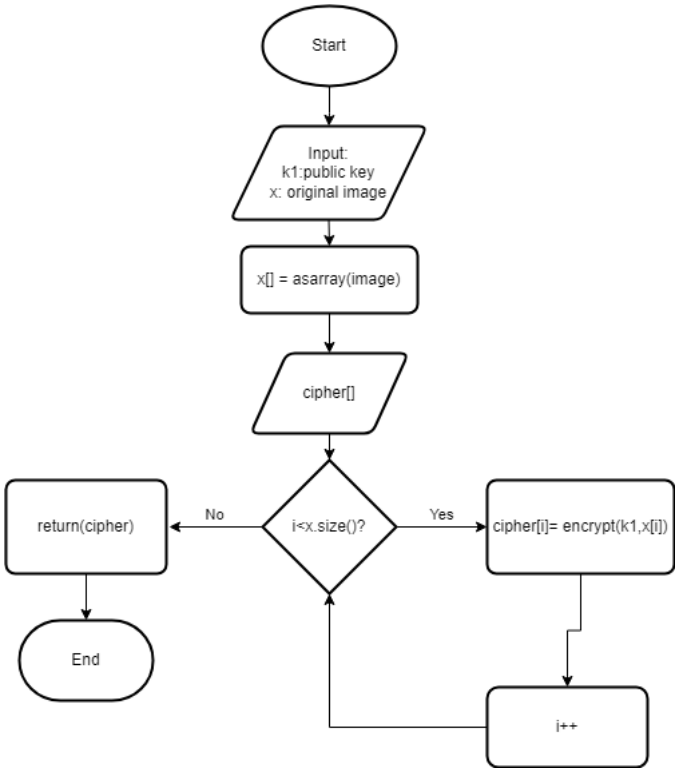

Figure 3: Flowchart to encrypt the image

4.4 Decryption of the encrypted image

To decrypt the image, the public key, private key and the cipher image is passed to a Paillier decryption function. Here each value in the cipher image array is decrypted and a plain image array is formed. After obtaining the decrypted image array, range validation is performed. If any value of the pixel decodes to a value larger than 255, it is set to 255. If any value of the pixel decodes to a value less than 0, it is set to 0. The final image from the array is reshaped using appropriate functions. The flowchart in Fig.[4] shows the process to generate the original image from an encrypted image.

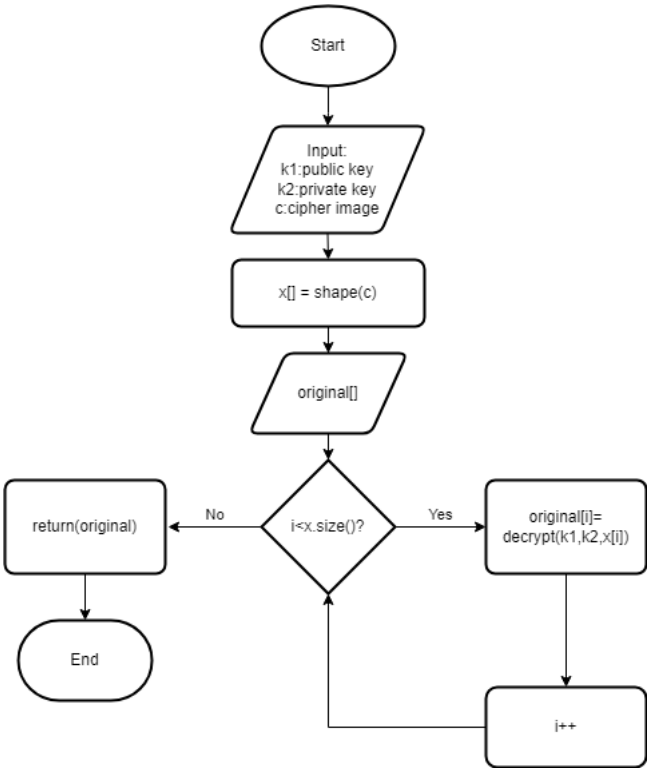

Figure 4: Flowchart to Decrypt the cipher image

5 Paillier Cryptosystem

The Paillier cryptosystem performs encryption and decryption using modular arithmetic operations. These operations can be time consuming if the product of the two prime numbers is very large.

5.1 Paillier Encryption Algorithm

1. Let $g = n + 1$ where g belongs to $Z^*_{n^2}$.
2. Let m represent the integer value of the pixel such that $0 \leq m < n$
3. Let r be a random number such that $r < n$.
4. The cipher text for the pixel in cipher image can be obtained as $c = g^m * r^n \% n^2$

5.2 Paillier Decryption Algorithm

1. Let c be the cipher text for the pixel in cipher image such that c belongs to $Z^*_{n^2}$.
2. Let $x = c^\lambda \% n^2$
3. Let L be a function such that $L(x) = \frac{x-1}{n}$
4. To obtain the original pixel value from cipher image, we calculate $m = (L(x) * \mu) \% n$.

6 Results

Figure 5: Original Lena Image

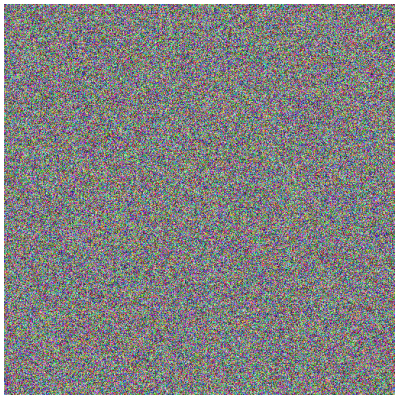

Figure 6: Encrypted Lena Image

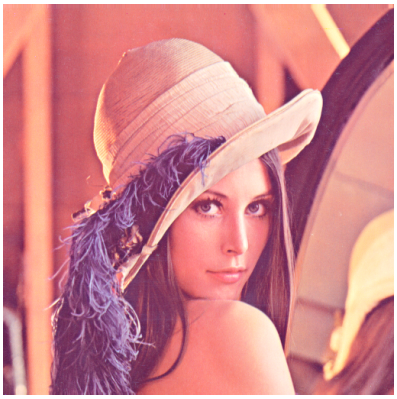

Figure 7: Decrypted Lena Image

Figure 8: Original Airplane Image

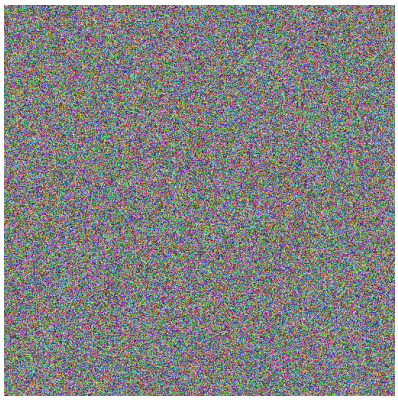

Figure 9: Encrypted Airplane Image

Figure 10: Decrypted Airplane Image

Figure 11: Original Baboon Image

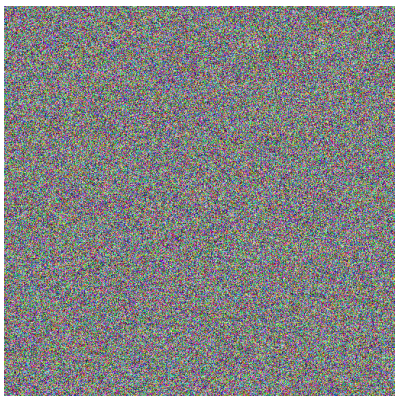

Figure 12: Encrypted Baboon Image

Figure 13: Decrypted Baboon Image

7 Conclusion

We have implemented the Paillier Homomorphic Encryption Scheme to encrypt and decrypt the images. By using modular arithmetic operations and Rabin-Miller Primality tests, we were able to obtain the public and primary keys. The keys and the original image are then passed on to the encryption system. As the prime number can be very large, significant amount of time is spent in performing modular arithmetic operations on each pixel of the image. If a bit length of 128 length is selected, it takes around 15-20 minutes to obtain the decrypted image. Thus there is a tradeoff between security and time complexity while selecting the appropriate bit length for prime numbers. Homomorphic operations are performed on the encrypted image in order to adjust the brightness of the image. We can also observe from the results that the encrypted images secure the whole original image and it is not possible in any way to obtain the original image without knowing the cryptosystem implementation details.

References

[1] Singh, Laiphrakpam Dolendro, and Khumanthem Manglem Singh. "Image encryption using elliptic curve cryptography." Procedia Computer Science 54 (2015): 472-481.

[2] M. Ashtiyani, P. M. Birgani and H. M. Hosseini, "Chaos-Based Medical Image Encryption Using Symmetric Cryptography," 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008, pp. 1-5, doi: 10.1109/ICTTA.2008.4530291.

[3] Manish Kumar, Akhlad Iqbal, Pranjal Kumar, A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie–Hellman cryptography, Signal Processing, Volume 125, 2016, Pages 187-202, ISSN 0165-1684

[4] O'Keeffe, Michael. "The paillier cryptosystem." Mathematics Department April 18 (2008): 1-16.