Homomorphic Encryption

Of

Image Data

by

Rohan Goyal 2017A3PS0269P

Under the Guidance of

Dr. Ashutosh Bhatia

OBJECTIVE:

To achieve a homomorphic encryption scheme suitable for use on various image formats and other visual data and study the limitations and advantages of the scheme.

MOTIVATION:

Information stored on servers can be openly viewed by the service providers. We seek achieve secrecy from the server space providers without losing the ease of cloud computing. Processing done on large images can be transferred to a remote device without the risk of exposing the information to a third party.

INTRODUCTION:

Good Image processing software need high computing power. Due to the shift on cloud computing such services are easily available on cloud based platforms, but it fails to ensure that the content of the image is hidden from the cloud server. Image processing majorly being arithmetic operations on a matrix can be hidden from server by the use of homomorphic encryption which ensures the privacy of the user from cloud server. The encryption scheme proposed ensures complete loss of original data characteristics and probabilities thereby reducing the chances of a statistical attack. We start by introducing our encryption algorithm and analyzing various aspects associated with it. We then discuss some use case liabilities. Finally we show histogram analysis of two test cases.

TABLE OF CONTENT

METHODOLOGY:

The basic methodology adopted here follows the following workflow:

First the image is rasterized i.e. represented in pixel data.

Next the pixel values are now run through the algorithm (1) to produce the encrypted image data and stored in a matrix representation.

The matrix data produced in the previous step is our encrypted data. Any arithmetic computations that were to be performed on the original data can now be performed on this data.

After computations, data points can be run through the algorithm (2) to obtain rasterized image data. This data can be now used to render an image.

ENCRYPTION SCHEME:

q	private key
i	input
w	random integer

E(q,i,) = wq+i ……………………………………………………………………(1)

DECRYPTION SCHEME:

c	Encrypted data

D(q,c,) = (c×a^{-g}) mod q ……………………………………………………………………(2)

PROOF FOR HOMOMORPHIC PROPERTY

Let c1 and c2 are the two cipher texts for the messages i1 and i2. w1 and w2 are the random positive integer. q is a large prime.

Homomorphic Multiplicative property

(c1 ×c2)

= (w1q+i1) × (w2q+i2) = (w1w2q2+w1qi2+i1w2q+i1×i2)

= (w1w2q+w1i2 +i1w2)q+i1 ×i2

= kq + i1×i2 where k is an integer.

Homomorphic Additive property

(c1+c2)

= (w1q+i1)+ (w2q+i2) = (w1+w2)q+i1+i2

= kq + i1+i2 where k is an integer.

This clearly shows that computations performed on the encrypted data are reflected in the input data as

D(c1 ×c2) = i1×i2 and

D(c1+c2) = i1+i2.

ALGORITHM PARAMETERS AND CONSTRAINS WITH RESPECT TO IMAGE DATA

Images are defined on 8 bit data which implies that the values of pixels can range from 0-255. Pixels in and image are usually defined by a hex code #rrggbb where rr gives value of red input, gg for green input and, bb for blue input. After rasterization we obtain a 3xwxh or 4xwxh matrix where w and h represent the width and height of the image.

- **q** (Private Key)

  Mathematically, q is an integer such that q>255.

  Taking various computations done on image data, the greatest operation preformed on a pixel is multiplying the pixel value by 255. Thus we say that value of q should be at least $(255)^2$ as the largest value held by a pixel is 255.

  We leave some room for carrying out subtraction operations hence.

  q is an integer >= 67000.

  but is advisable to use a greater number to increase the security of the scheme ie for increasing keyspace

- **w** (random integer)

  w is a random integer generated using a **prg** individually for each pixel value. w is different for every value point. This factor is responsible for introduction of randomness in the encrypted data.

  Mathematically w has no constraints but is advisable to use a w less that 10^8 for ease of processing

OVERFLOW OF PIXEL VALUES IN THE DECRYPTED DATA

Due to various operations performed on the encrypted data, the decrypted data has two types of possible overflow scenarios. This is because the pixel value range from 0-255 but out algorithm permits the decrypted value to range from 0-(q-1).

We discuss here the methodology for dealing with such overflows.

- 0-255 for these pixels there is no overflow, we retain their values.
- 255-65025(*k*) As discussed earlier, the greatest operation performed on a pixel in a practical scenario is multiplication by 255. The pixels in this range can be **equated to 255** as in normal image processing norms, anything which is greater that 255 is treated as 255.
- 65026(*k*+1) – (q-1) these value occur due to subtraction on the encrypted data. By observing the algorithm we see that negative values as decrypted as the the negative value plus q. Hence we **equate these values to 0.**

Note:- It is advisable to leave atleast 150 padding for subtraction operations but this may vary for particular computational cases. Hence the limit (*k*) taken here should can be adjusted accordingly. *k* and q must be selected such that

$$k >= H$$

$$k - q < L$$

Here, H is the theoretical upper limit and L is the theoretical lower limit that can be obtained by a pixel value after the computations in question.

TYPES OF RASTERIZATION AND EFFECTS ON THE PRACTICAL USAGE (A RISK TO SECRACY OF THE PRIVATE KEY)

After rasterization we obtain a 3xwxh or 4xwxh matrix where w and h represent the width and height of the image. In case of the 3x matrix, the first value gives red, 2^{nd} green and 3^{rd} blue inputs. In this case all the pixels can be processed through the algorithm.

But for modern formats such as .png, we obtain 4x matrix. Here the 4^{th} value gives the opacity of the pixel on a scale of 0-255. Statistically the value of this pixel is **255 for majority of the images.** Hence it is advised **not to encrypt this information** with the rest of the data as encrypted data in this column of the matrix would expose the data to a statistical attack through which q can be obtained.

This information can be left unencrypted. This may be a problem if we seek to share a transparent image create a particular shape with its boundaries, hence it is advised to encrypt this column with a different value of q.

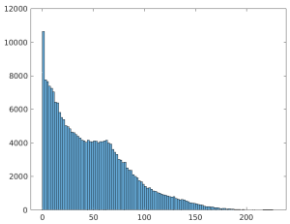
STATISTICAL ANALYSIS ON ENCRYPTED DATA

The way in which the pixels are distributed in an image can be found by performing histogram analysis. The statistical properties of the encrypted image are given by this security analysis. A perfect encrypted image must have uniformly distributed histogram. Therefore by performing this analysis we can obtain a graph with streaks distributed randomly. By looking at this graph we can come into a conclusion that the applied algorithm encrypts the image completely or is there any part of the image which is not properly encrypted. It also shows the histogram of encrypted image where the pixel distribution is uniform. It is fairly uniform which makes it difficult to extract the pixels statistical nature of the plain image.

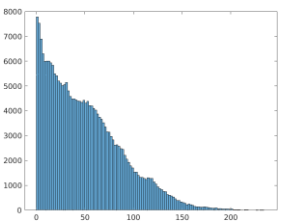
Now we present two test images, corresponding encrypted images and histograms for all three channels for both encrypted and decrypted images

……

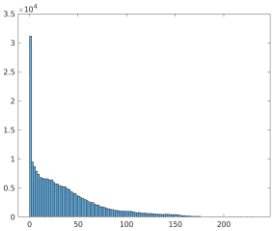
Image 1



blue channel

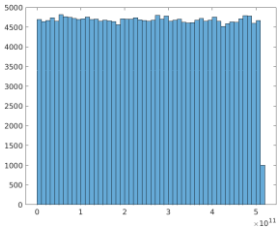


green channel

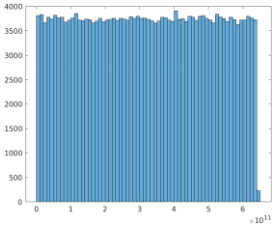


red channel

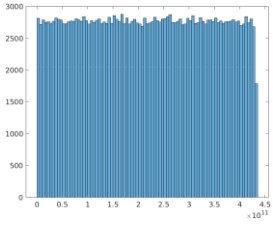
Image 1 encrypted



blue channel

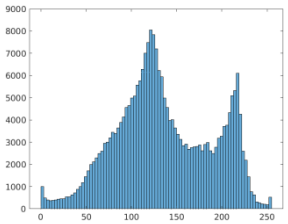


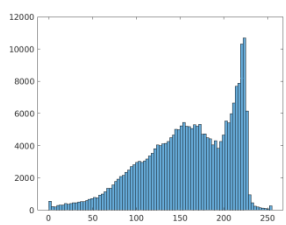
green channel



red channel

……..





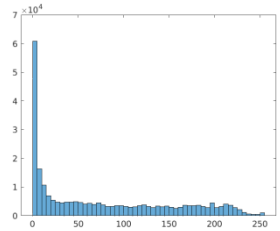
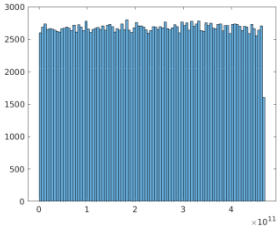
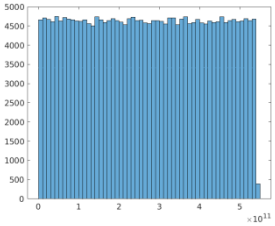


Image 2 blue channel green channel red channel





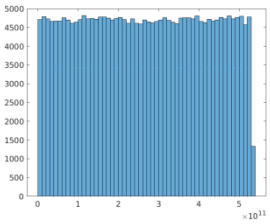


Image 2 encrypted blue channel green channel red channel

Hence by observing these histograms we observe perfect randomness, even distribution and no correlation.

CONCLUSION

Fully Homomorphic Encryption scheme is proposed. The Fully Homomorphic Encryption scheme supports both addition and multiplication. The input images and the corresponding encrypted images are shown. Histogram analysis is performed. The results are tabulated. The analysis that is performed helps us to verify the efficiency of the proposed method in image security.

REFERENCES:

1. A.M. Vengadapurvaja*, G. Nisha, R. Aarthy, N. Sasikaladevi, An Efficient Homomorphic Medical Image Encryption Algorithm For Cloud Storage Security
2. Craig Gentry, a fully homomorphic encryption scheme
3. IsmetOzturk and IbraSogukpinar . Analysis and Comparison of Image Encryption Algorithm .International Journal of Information Technology; 1(2): 64-67.