

REVERSIBLE DATA HIDING IN ENCRYPTED IMAGE

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

**Janumpally Prem Kumar Reddy, (REG.NO.
38110212)
K Rupak sai
(REG.NO. 38110470)**



**DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119.

APRIL - 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Janumpally Prem Kumar Reddy, (REG.NO. 38110212)K Rupak sai (REG.NO. 38110470)** who have done the Project work as a team who carried out the project entitled “**REVERSIBLE DATA HIDING IN ENCRYPTED IMAGE**” under my supervision from **November 2019** to **April 2020**.

Internal Guide

Dr. S. Vigneshwari, M.E., Ph.D.

Head of the Department

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I, **Janumpally Prem Kumar Reddy, (REG.NO. 38110212) K Rupak sai (REG.NO. 38110470)** hereby declare that the Project Report entitled **“REVERSIBLE DATA HIDING IN ENCRYPTED IMAGE”** done by us under the guidance of **Dr. S. Vigneshwari, M.E., Ph.D.,** (Internal) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE & TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneshwari M.E., Ph.D., and Dr. L. Lakshmanan M.E., Ph.D., Heads** of the Department of Computer Science & Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. Vigneshwari, M.E., Ph.D., Associate Professor**, Department of Computer Science and Engineering for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science & Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Facing a large number of personal photos and limited resource of mobile devices, cloud plays an important role in photo storing, sharing and searching. Meanwhile, some recent reputation damage and stalk events caused by photo leakage increase people's concern about photo privacy.

Images are becoming one of the key enablers of user connectivity in social media applications. Many of them are directly exploring image content to suggest new friends with similar interests. To handle the explosive volumes of images, one common trend is to leverage the public cloud as their robust service backend.

Despite the convenience, exposing content-rich images to the cloud inevitably raises acute privacy concerns. In this paper, we propose a privacy-preserving architecture for image-centric social discovery services, designed to function over encrypted images.

The proposed system is a new technique by using multi-secret sharing as the underlying encryption, which indeed induces a blow-up issue of the key size. For preserving the efficiency of the key size, we apply a compression by using lightweight cryptographic algorithms. This scheme based on the proposed techniques, and show effectiveness, efficiency, and security by experiments and analysis.

CHAPTER NO	TABLE OF CONTENTS	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	iv
	LIST OF ABBRIVIATIONS	v
1	INTRODUCTION	1
	1.1 RELATEDWORK	3
	1.1.1 Share Independent Secret Key	4
	1.1.2 Share No Secret Key	4
	1.2 MAIN CONTRIBUTIONS	5
	1.2.1 Data Embedding	6
	1.2.2 Summarizing our and Wu et al.'s secret sharing-based scheme	7
	1.3 ORGANIZATION STRUCTURE	8
2	LITERATURE SURVEY	9
3	AIM AND SCOPE OF PROJECT	13
	3.1 EXISTING SYSTEM	13
	3.1.1 Drawbacks of Existing System	14
	3.2 PROPOSED SYSTEM	14
	3.2.1 Drawbacks of Proposed System	16
	3.3 ARCHITECTURE	16
	3.4 TECHNOLOGY USED	17
	3.5 EXISTING ALGORITHM	17
	3.6 PROPOSED ALGORITHM	17
	3.6.1 Advantages of Proposed System	17
	3.7 SYSTEM ANALYSIS	17
	3.7.1 Waterfall Model	18
	3.7.2 RAD Model	20
4	METHODOLOGY	24
	4.1 HARDWARE REQUIREMENT	24
	4.2 SOFTWARE REQUIREMENT	24
	4.3 MODULES	25

5	RESULT AND DISCUSSION	29
6	CONCLUSION AND SUMMARY	33
	REFERENCES	34
	APPENNDICES	
	A. Source Code	36
	A. Screen Shot	40

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	Architecture Of The Algorithm	16
3.2	Waterfall Model	18
3.3	Rapid Application Development Architecture	20
5.1	Cover, Encrypted and Stego image	40
5.2	Graphical Survey	41

LIST OF ABBRIVATIONS

RDH	Reversible Data Hiding
RDHEI	Reversible Data Hiding Encrypted Image
LSB	Least Significant Bit
SIK	Share Independent Secret Keys
SNK	Share No Secret Key
SOK	Shared One Key
OAMSS	Operating Addition homomorphism in multi-Secret Sharing
MCGs	Mirroring CipherText Groups
PRF	Probabilistic Random Forest
LSH	Locality Sensitive Hashing
RAD	Rapid Application Development
RAM	Random Access Memory
HTML	Hyper Text Markup Language
IDE	Integrated Development Environment
CFS	Correlation Based Feature

CHAPTER 1

INTRODUCTION

Reversible data hiding (RDH) is a notion that allows to embed the additional and secret message into cover media, such as military or medical images, and to perform a reversible procedure that extracts the hidden secret message and perfectly reconstructs the original cover content. Numerous reversible data hiding methods have been introduced over the last two decades. Two seminal ideas of RDH are difference expansion (proposed by Tian [1]) and histogram shifting (proposed by Ni et al. [2]). In the difference expansion method [1], the differences between two adjacent pixels are doubled to release a new least significant bit (LSB) plane for carrying the secret message. In the histogram shifting method [2], the zero and peak points are used to embed the secret message by slightly modifying the pixel values. Many RDH studies have elaborated these two concepts to improve payload and image quality [3, 4, 5, 6, 7, 8, 9]. Recently, a new direction of RDH known as RDH over an encrypted image (RDHEI) has been introduced. This novel RDHEI notion was firstly introduced by Zhang in 2011 [10], and captures the following real-life scenario regarding owner privacy known as image privacy [10]. An inferior assistant or a channel administrator is in the middle of a workflow and is authorized to insert some additional data such as the origin information, image notations or authentication data, within the encrypted image, where the original image content is unknown to this party. Indeed, medical images are encrypted for preserving the patient privacy, and a database administrator only embeds a few data into the corresponding encrypted images. For the consistency of a medical image, it must guarantee that the original content can be perfectly reconstructed after decryption-then-extraction of the secret message by the receiver. That is, RDHEI not only ensures the accuracy of the reconstructed cover-image and extracted secret message which are two basic tasks of RDH, but also preserves the privacy of the cover-image. More precisely, the work of Zhang [10] formalizes the model to describe the aforementioned scenario. The image provider P intends to preserve the privacy of the cover-image, but still desires a data hider H to embed a secret message.

Therefore, H embeds the message into the encrypted image which is generated by P from the cover-image. Finally, the receiver R can recover the original cover-image and then extract the secret message correctly. The procedure run by R is known as decryption-then-extraction. However, the receiver also can be divided into two steps (decryption and extraction). We specify these two steps to two kinds of receivers, Rdec and Rext, and Rdec performs decryption, and Rext takes Rdec's decrypted image to extract the secret message. The amount of digital images has increased rapidly on the Internet. Image security becomes increasingly important for many applications, e.g., confidential transmission, video surveillance, military and medical applications. For example, the necessity of fast and secure diagnosis is vital in the medical world.

In this paper, we develop an effective and reliable framework for RDH in the encrypted domain. In fact, the proposed method belongs to the third category. Its main contribution is the combination of the modular addition and two-dimensional (2D) histogram modification. Its advantages are mainly manifested in four aspects. First of all, room for data hiding does not need to be vacated before encryption, which is more reasonable compared with the methods in [25–28]. Secondly, completely separable and completely reversible can be achieved, which is more reliable than the methods in [18–21]. Thirdly, the modular arithmetic addition operation, which has additive homomorphism, is utilized for image encryption. It does not cause data expansion, unlike the public-key cryptosystems in [29, 31–33]. Finally, since data embedding in encrypted domain is accomplished by using pairwise coefficient modification, embedded capacity has been greatly improved compared with the methods in [30, 34]. The rest of the paper is organized as follows. In Section 2, we describe the proposed scheme, which includes image encryption, data embedding in encrypted image, data extraction, and original image recovery. Experimental results and analysis are presented in Section 3. Finally, in Section 4, conclusions and future work are drawn.

1.1 RELATED WORK

A comprehensive survey on RDH is presented by Shi et al. [11] to deeply analyze and highlight the advances of RDH for the recent progress. It studies aspects of RDH, including RDH into image spatial domain [1, 2], RDH into image compressed domain (e.g., JPEG) [12, 13, 14], RDH suitable for image semi-fragile authentication [15,16,17], etc. In particular, it also investigates RDHEI, and categorizes the existing RDHEI schemes into two classes: vacating room before encryption and vacating room after encryption by embedding strategies. For key setting, Shi et al. [11] also mentioned the other notion, so-called RDHEI based on public key encryption. However, inspired by the factor of key setting, the present studies identify the following two notions of RDHEI.

Numerous insightful works have proposed this type of RDHEI schemes. _ Share no secret key (SNK). In contrast to SIK, R does not need to share any secret key. This can be easily achieved through public key encryption where R has a public/secret key pair, and P (H, resp.) can use the public key to do image encryption (data embedding, resp.). The first solution, proposed by Chen et al. [9], is to use Paillier homomorphic encryption [10] to encrypt each pixel and rely on specific techniques to complete data embedding. With the use of the homomorphic encryption, the follow-up works of Zhang et al. [11], Li and Li [12], and Shiu et al. [13] respectively implement some reversible data hiding techniques under the public key encryption associated with the homomorphic property. Tsai et al. [3] present a predictor, may be denoted local difference (LD) predictor, that computes difference between pixels intensities in a local area of the image and most central one in the area to bring out prediction-errors.

They embed secret data via histogram modification of the prediction errors. Sachnev et al. [4] further present cross-dot predictor that divides image into two “cross” and “dot” sets. Dot set may be predicted using cross one and vice versa. Cross-dot predictor is also represented as chess-board (CB) predictor in [5]. In general, the content owner expects to send only an encrypted image to the manager without extra information. In addition to VRAE and RRBE, another type of method has recently been proposed by using homomorphic encryption. With the additive homomorphic property of Paillier cryptosystem, Chen et al. [15] firstly

proposed a homomorphic encryption based RDH approach. Shiu et al. [17] improved Chen et al.'s method by adopting the concept of difference expansion into homomorphic encryption.

Moreover, RDH in the homomorphic encrypted domain has also been investigated in [18]. However, the used public-key cryptosystems lead to data expansion after image encryption. In [16], the additive homomorphic property of modulo operation is utilized to realize the RDH in the encrypted domain.

1.1.1 Share independent secret keys (SIK)

R shares independent keys, key_P and key_H , with P and H respectively. Notably, these keys (key_P ; key_H) are secret and used to run image encryption and embedding algorithms. Numerous insightful works [10, 18, 19, 20, 21, 22] have proposed this type of RDHEI schemes.

1.1.2 Share no secret key (SNK).

In contrast to SIK, R does not need to share any secret key. This can be easily achieved through public key encryption where R has a public/secret key pair, and P (H, resp.) can use the public key to do image encryption (data embedding, resp). The first solution, proposed by Chen et al.[23], is to use Paillier homomorphic encryption [24] to encrypt each pixel and rely on specific techniques to complete data embedding.

With the use of the homomorphic encryption, the follow-up works of Zhang et al. [25], Li and Li [26], and Shiu et al. [27] respectively implement some reversible data hiding techniques under the public key encryption associated with the homomorphic property. To summarize the flexibility of key setting, it is clear that only the designated party who has the secret key can be P or H in SIK. However, the advantage of SNK is that anyone can be P or H, since the keys of encryption or embedding are exactly the public key. In addition, as known, those homomorphic encryption-based SNK-type RDHEI schemes are practically inefficient since the underlying encryption schemes usually rely on complicated algebra structures and spend high computational cost. It suffices to give the following question, and we will aim for addressing it in the remainder of this paper. In fact, Wu et al. [28] had proposed a shared one key (SOK) scheme based on

secret sharing. However, their method spends much space cost, since it encrypts a pixel into n shares, where n is the security parameter of secret sharing, and the total cost of an encrypted pixel will blow up to $8n$ bits.

1.2 MAIN CONTRIBUTIONS

Let us briefly summarize our results. Our starting point is to formalize the new notion of key setting and care about. The key setting offers the framework among P , H , and R . For example, if a RDHEI scheme is under public key encryption, anyone can be P and H . If under a symmetric encryption between P and R (H and R , resp.), only specific party who holds the shared secret key can be P (H , resp.). The key use is referred to as party flexibility. In the following, we formalize the key setting for more details. The efficiency to achieve better efficiency, we must avoid using public key encryption. However, if we do not use any public key encryption scheme, it is impossible to protect image privacy (the original purposes of RDHEI) without the shared key between P and R . Thus, for preserving privacy, the ideal class is that receiver shares “only one” secret key with the image provider (SOK, for short). In particular, there is no shared key between H and R , which precisely implies that the embedding procedure does not take any shared key as input. The proposed SOK schemes are inspired from some existing SNK schemes (i.e., [25, 27]). We found that these SNK schemes work with Paillier encryption (or other addition homomorphic encryption) to preserve image privacy, and the property of homomorphic evaluation is used to embed the message. For achieving our above-mentioned requirements, we replace the parts of Paillier encryption with secret sharing that also enjoys homomorphic evaluation in some ways². We show an abstraction of those SNK schemes, and then under the abstraction, introduce our method. The high level idea of our method is composed of the following two steps.

Image recovery If the encryption key is available on the recipient side, the encryption key can be used to decrypt the original image. The sender sends the encryption key to the recipient. This key is used as the start value of the feedback shift register. This initial value is used to generate a pseudorandom number that is added to the pixel value of the image and performs the modular operation with 256. The generated value will be the new pixel value. The same procedure is

performed for all pixels in the cover image. Therefore, the image is decrypted. Pseudo-random numbers are generated by subtracting a random number from a starting value. The result is placed at the end of the starting value by moving the starting value to the left. D. Data extraction Data extraction is the reverse process of data integration. Initially, the key is included in the encrypted image where the data is displayed. Group LSB bits into 5 bits. Then check 5bits based on this control symbol with control symbols 1B, 1C, 1D and 1E. Finally, the original data is acquired. Encryption

Secret sharing acts as a symmetric encryption to encrypt the cover-image, so our method use one shared key between P and R. However, ours does not construct shares for each pixel like Wu et al.'s method. For preserving the total size, we pack t pixels and t random factors together to generate only t shares, and put the shares back as encrypted pixels and set random factors as the key. It suffices to avoid the size blow-up, and also keeps correctness of decryption by using t random factors and t shares. The technique of our method is inspired by the multi-secret sharing, but we slightly modify it for security and framework of SOK.

1.2.1 Data Embedding

Data embedding is a new steganographic method for combining digital information sets. This paper describes the data embedding method and gives examples of its application using software written in the C-programming language. Sandford and Handel produced a computer program that implements data embedding in an application for digital imagery. To provide security for the embedded data, one can remove the key from the combined data and manage it separately. The image key can be encrypted and stored in the combined data or transmitted separately as a ciphertext much smaller in size than the embedded data. The data embedding method applies to host data compressed with transform, or 'lossy' compression algorithms, as for example ones based on discrete cosine transform and wavelet functions. Analysis of the host noise generates a key required for embedding and extracting the auxiliary data from the combined data. The key is stored easily in the combined data. Images without the key cannot be processed to extract the embedded information.

We divide the secret message into several units. Then, for embedding a unit, we generate another t shares without needing any key (as known as the above-mentioned random factors), and then use homomorphic evaluation and embedding procedure to embed message into the t encrypted pixels. The proposed method strictly relies on the properties of secret sharing. Summarizing the main techniques, secret sharing serves as the underlying primitive offering security, multiple secret preserves size complexity, and inherently additive homomorphism realizes the data embedding. We provide the formal description of the technique, and present a clear notion, so-called operating addition homomorphism in multi-secret sharing (OAMSS). Additionally, we also provide another technique to compress the size of a key used in OAMSS.

For generalization, if SNK schemes satisfy some properties, they can be converted to SOK. Hence, our method can be generalized as a converter. As a concrete instantiation, given Shiu et al.'s SNK scheme based on difference expansion, we show the SOK-type RDHEI by slight modification. The scheme overview is described as follows. P will pre-process the cover-image (including double the differences between two 2Here homomorphic evaluation is to perform homomorphic operation over encrypted data and further obtain the encrypted result. H will obtain a new polynomial which carries a secret message in the released LSB plane, and then use addition homomorphism to generate the encrypted image with embedded message. Finally, by decryption R is able to obtain the stego-image (a.k.a image with embedding message), and then recover the cover-image and secret message. We remark that in this scheme, the embedding of the original method of [1] is decoupled into two steps: (1) double the differences as the pre-processing run by P and (2) embed a secret message run by H . Except for the instantiation from [27], we also provide another SOK scheme from [25].

1.2.2 Summarizing our and Wu et al.'s secret sharing-based scheme

Wu et al. [28] introduced a new model, referred to as a distributed model, where, instead of one single H , many specific parties work as data hiders $H_1; \dots; H_n$.

Indeed, they are duplicated to work with the identical task (i.e., embedding the identical secret message). However, summarizing the security definition of the model, the adversary is only able to corrupt a number (such as less than threshold k) of data hiders. Accordingly, Wu et al. [28] relied secret sharing as image encryption, and also accordingly proposed the first secret sharing-based RDHEI scheme in the distributed model. Their technique of image encryption is to use k -out-of- n secret sharing to produce n shares of a pixel, and thus less than k cannot recover the original pixel value by unconditional security of secret sharing. However, in this paper, we still work on the original model of RDHEI with shared one key setting. In particular, our technique uses multi-secretsharing.

A multi-secret sharing scheme is a natural generalization of the secret sharing scheme. We show the close similarity between the Chinese remainder theorem and the uniqueness theorem of interpolating polynomial. Based on the similarity, we propose a threshold multi-secret sharing scheme that is a generalization of the Shamir threshold scheme. We also show that the proposed scheme with the proposed deletion procedure can realize all possible monotone access structures.

1.3 ORGANIZATION STRUCTURE

The rest of this paper is organized as follows. In Section II, we will introduce some preliminaries, including secret sharing and polynomial interpolation, and a clean abstraction of the existing SNK-type RDHEI schemes. Then, in Section III, our techniques are formally presented. A demonstration (associated with its experiments and discussions) will presented in Section IV. Finally, our conclusions of this paper are given in Section VI. In Appendix, we provide some missing details. We will review Paillier additive homomorphic encryption which is used to construct SNK-type schemes. Then, we will show how to handle side information in our scheme. Given the SNK-type scheme of Zhang et al. [25], we demonstrate an alternative SOK-type scheme. Finally, we state our converter and show how to convert SNK with specific properties to SOK.

CHAPTER 2

LITERATURE SURVEY

New Framework for Reversible Data Hiding in Encrypted Domain Fangjun Huang, Yun Qing Shi, Jiwu Huang November 2016. In this method, we present a new framework which allows the numerous RDH schemes developed before for non-encrypted images be conducted in the encrypted domain directly. The separable reversible data-hiding scheme in encrypted image, which can offer relatively high payload and error-free data extraction. In the past more than one decade, hundreds of reversible data hiding (RDH) algorithms have been reported. Via exploring the correlation between the neighboring pixels (or coefficients), extra information can be embedded into the host image reversibly. However, these RDH algorithms cannot be accomplished in encrypted domain directly since the correlation between the neighboring pixels will disappear after encryption. In order to accomplish RDH in encrypted domain, specific RDH schemes have been designed according to the encryption algorithm utilized. In this paper, we propose a new simple yet effective framework for RDH in encrypted domain. In the proposed framework, the pixels in a plain image are firstly divided into sub-blocks with the size of $m \times n$. Then with an encryption key, a key stream (a stream of random or pseudorandom bits/bytes that are combined with a plaintext message to produce the encrypted message) is generated, and the pixels in the same sub-block are encrypted with the same key stream byte. After the stream encryption, the encrypted $m \times n$ sub-blocks are randomly permuted with a permutation key. Since the correlation between the neighboring pixels in each sub-block can be well preserved in the encrypted domain, most of those previously proposed RDH schemes can be applied to the encrypted image directly. One of the main merits of the proposed framework is that the RDH scheme.

A New Histogram Modification Based Reversible Data Hiding Algorithm Considering the Human Visual System. Seung-Won Jung, Le Thanh Ha, and Sung-JeaKo FEBRUARY 2011. This system proposes an improved histogram modification based reversible data hiding technique. In the proposed algorithm, unlike the conventional reversible techniques, a data embedding level is adaptively

adjusted for each pixel with a consideration of the human visual system (HVS) characteristics. To this end, an edge and the just noticeable difference (JND) values are estimated for every pixel, and the estimated values are used to determine the embedding level. This pixel level adjustment can effectively reduce the distortion caused by data embedding. The experimental results and performance comparison with other reversible data hiding algorithms are presented to demonstrate the validity of the proposed algorithm.

An Improved Reversible Data Hiding in Encrypted Images Using Side Match
Wien Hong, Tung-Shou Chen, and Han-Yan Wu APRIL 2012. This system proposes an improved version of Zhang's reversible data hiding method in encrypted images. The original work partitions an encrypted image into blocks, and each block carries one bit by flipping three LSBs of a set of pre-defined pixels. The data extraction and image recovery can be achieved by examining the block smoothness. Zhang's work did not fully exploit the pixels in calculating the smoothness of each block and did not consider the pixel correlations in the border of neighboring blocks. These two issues could reduce the correctness of data extraction. This letter adopts a better scheme for measuring the smoothness of blocks, and uses the side-match scheme to further decrease the error rate of extracted-bits.

New Framework for Reversible Data Hiding in Encrypted Domain
Fangjun Huang, Jiwu Huang, Yun Qing Shi November 2016 . In the past more than one decade, hundreds of reversible data hiding (RDH) algorithms have been reported. Via exploring the correlation between the neighboring pixels (or coefficients), extra information can be embedded into the host image reversibly. However, these RDH algorithms cannot be accomplished in encrypted domain directly since the correlation between the neighboring pixels will disappear after encryption. In order to accomplish RDH in encrypted domain, specific RDH schemes have been designed according to the encryption algorithm utilized. In this paper, we propose a new simple yet effective framework for RDH in encrypted domain.

In the proposed framework, the pixels in a plain image are firstly divided into sub-blocks with the size of $m \times n$. Then with an encryption key, a key stream (a stream of random or pseudorandom bits/bytes that are combined with a plaintext message to produce the encrypted message) is generated, and the pixels in the

same sub-block are encrypted with the same key stream byte. After the stream encryption, the encrypted $m \times n$ sub-blocks are randomly permuted with a permutation key. Since the correlation between the neighboring pixels in each sub-block can be well preserved in the encrypted domain, most of those previously proposed RDH schemes can be applied to the encrypted image directly. One of the main merits of the proposed framework is that the RDH scheme is independent of the image encryption algorithm. That is, the server manager (or channel administrator) does not need to design a new RDH scheme according to the encryption algorithm that has been conducted by the content owner; instead, he/she can accomplish the data hiding by applying the numerous RDH algorithms previously proposed to the encrypted domain directly.

Reversible Data Hiding in Homomorphic Encrypted Domain By Mirroring Ciphertext Group Shijun Xiang and Xinrong Luo November 2017. This paper proposes a novel reversible data hiding scheme for encrypted images by using homomorphic and probabilistic properties of Paillier cryptosystem. In the proposed method, groups of adjacent pixels are randomly selected, and reversibly embedded into the rest of the image to make room for data embedding. In each group, there are a reference pixel and a few host pixels. Least significant bits (LSBs) of the reference pixels are reset before encryption and the encrypted host pixels are replaced with the encrypted reference pixel in the same group to form mirroring cipher text groups (MCGs). In such a way, the modification on MCGs for data embedding will not cause any pixel oversaturation in plaintext domain and the embedded data can be directly extracted from the encrypted domain. In a MCG, the reference cipher text pixel is kept unchanged as a reference while data hider embeds the encrypted

Nowadays, a new challenge consists to embed data in encrypted images. Previous work proposed to embed data in an encrypted image by using an irreversible approach of data hiding. The challenge was to find an encryption method robust to noise. Since the entropy of encrypted image is maximal, the embedding step, considered like noise, is not possible by using standard data hiding algorithms. A new idea is to apply reversible data hiding algorithms on encrypted images by wishing to remove the embedded data before the imaged

encryption. Recent reversible data hiding methods have been proposed with high capacity, but these methods are not applicable on encrypted images. In this paper we propose an analysis of the local standard deviation of the marked encrypted images in order to remove the embedded data during the decryption step.

CHAPTER 3

AIM AND SCOPE OF PROJECT

3.1 EXISTINGSYSTEM

The existing technique uses multi-secret sharing as the underlying encryption, which indeed induces a blow-up issue of the key size. For preserving the efficiency of the key size, we apply a compression by using lightweight cryptographic algorithms. Then, we demonstrate our shared one key SOK scheme based on the proposed techniques, and show effectiveness, efficiency, and security by experiments and analysis.

Secret sharing serves as the underlying primitive offering security, multiple secret preserves size complexity, and inherently additive homomorphism realizes the data embedding. We provide the formal description of the technique, and present a clear notion, so-called operating addition homomorphism in multi-secret sharing.

For prefect accuracy, we require that the reconstructed cover-image and message in the stage of Decryption-then Extraction must be identical to the original cover-image encrypted in Image-Encryption and the message hidden in Message-Embedding.

Our design uses secure one-way functions, like PRF, to protect the sensitive LSH values while preserving the locality-sensitivity clustering. Note that LSH functions do not preserve the one-way property as pseudo-random functions. Thus, it is possible to reveal the input by observing LSH values.

With the fast advancements happening in portable web and distributed storage, protection and security of individual information have increased noteworthy consideration these days. There are no ensures that put away information won't be gotten to by unapproved substances, for example, the cloud supplier itself or vindictive aggressors. Under these particular conditions, touchy pictures, for example, clinical and individual pictures should be scrambled before redistributing for protection safeguarding purposes [1, 2]. As it were, the buyers might want to give the entrusted cloud server just a scrambled form of the information rather

than the first substance. The cloud specialist organization (who stores the information) isn't approved to get to the first substance (i.e., plaintext). Nonetheless, in some application situations, the cloud servers or database administrators need to install some extra messages, for example, confirmation or documentation information, straightforwardly into an encoded information for alter recognition or proprietorship announcement purposes. For instance, patient's data can be inserted into his/her encoded clinical picture to maintain a strategic distance from undesirable presentation of classified data.

3.1.1 Drawbacks of Existing System

- High embedding rate
- High quality of decrypted and recovered images
- The probability of lossless recovery is low
- Better performance than that of VRAE
- Higher data hiding capacity than reversible data hiding techniques
- No reliability in the data
- High Computational Complexity
- Become very complex if the amount of variance
- Not Scalable
- High Storage Requirement

3.2 PROPOSED SYSTEM

Our system is the first to deliver the privacy-preserving image-centric social discovery service in an outsourced architecture of existing social media applications, which allows fast and scalable social discovery for millions of users with a large scale of encrypted images.

Our proposed secure index structure guarantees the privacy protection on image content and user image profiles, while maintaining reasonably good social discovery accuracy, index space efficiency, low service latency, and constant service bandwidth cost.

We extend the system functionality by supporting secure insertion and secure deletion for the updates of user image profiles. We also implement the system compatibility with current image sharing services.

We formalize the security definition and generalize the leakage definition in our system. We prove that our social discovery service and the user profile updates achieve semantic security against adaptive adversaries.

Our design is expected to readily complement existing context-based techniques and serve for more advanced privacy-aware social media applications.

Its fundamental commitment is the mix of the particular expansion and two dimensional (2D) histogram modification. Its points of interest are basically showed in four perspectives. Above all else, space for information stowing away shouldn't be cleared before encryption, which is progressively sensible contrasted and the strategies in [25–28]. Furthermore, totally distinct and totally reversible can be accomplished, which is more solid than the methods]. Tirdly, the secluded number juggling expansion activity, which has added substance homomorphism, is used for picture encryption. It doesn't cause information development, not at all like the open key cryptosystem. At long last, since information implanting in encoded space is practiced by utilizing pairwise coefcient modification, inserted limit has been incredibly improved contrasted and the strategies. Right now, RDH strategy in encoded pictures is delineated. It is made out of three sections, that is, age of the encoded picture, age of the stamped scrambled picture, information extraction, and picture recuperation. To begin with, the substance proprietor scrambles the first picture with encryption key to deliver an encoded picture. At that point, the information hider without knowing the genuine substance of the first picture can install some extra information into the encoded picture. Here, the information hider can be an outsider, for instance, a database supervisor or a cloud supplier, who isn't approved to get to the first substance of the sign (i.e., plaintext). At the less than desirable end, perhaps the substance proprietor himself or an approved outsider can extricate the shrouded information either in encoded or decoded picture.

3.2.1 Drawbacks of Proposed System

- It relies on complicated algebra structures
- Large Embedding distortions
- Recover the cover image only for small data
- May result in incorrect extraction, when block size is small
- High computational cost

3.3 ARCHITECTURE

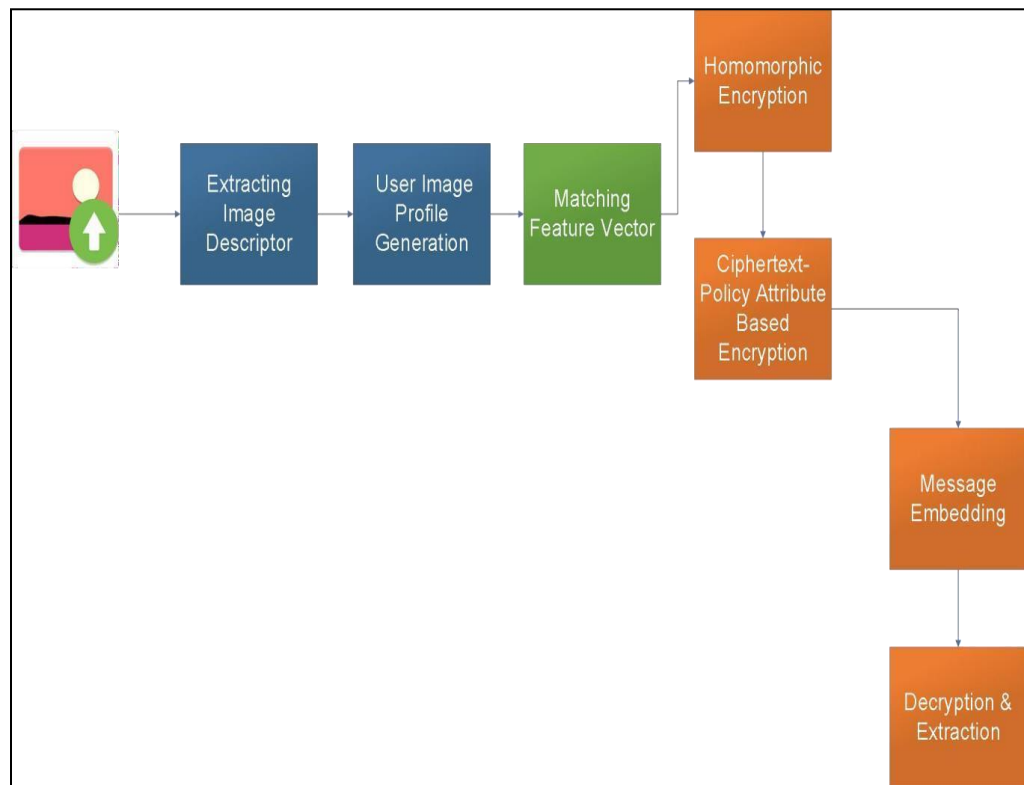


Fig. 3.1: Architecture of the Algorithm

3.4 TECHNOLOGY USED

Backend Technologies

- Python

- Numpy
- Sci-learn
- Eclipse IDE Front end Technologies
- Web Technologies
- Bootstrap

3.5 EXISTING ALGORITHM

↓ Lightweight Cryptographic Algorithms

3.6 PROPOSEDALGORITHM

- ✓ Homomorphic Encryption Algorithm
- ✓ Ciphertext-Policy Attribute Based Encryption Algorithm

3.6.1 Advantages of Proposed Algorithm

- ✓ Higher standard of data security
- ✓ Ensure data privacy
- ✓ Automation and orchestration

3.7 SYATEM ANALYSIS

One of the primary choices we face for our project implementations is “Which development methodology should we use?” There are various software development methodologies are available in the ICT industry. They are

- Agile Software Development
- Crystal Methods
- Dynamic Systems Development Model(DSDM)
- Extreme Programming(XP)
- Feature Driven Development(FDD)
- Joint Application Development(JAD)
- Lean Development(LD)
- Rapid Application Development(RAD)

- Rational Unified Process(RUP)
- Scrum
- Spiral
- Systems Development Life Cycle(SDLC)
- Waterfall (a.k.a.Traditional)

Among all these methodologies, we have decided to compare the two most popular and widely used methodologies:

- Waterfall Model
- RAD Model

3.7.1 Waterfall Model

It is called as traditional approach. Waterfall is a linear method (sequential model) for any software application development. Here application development is segregated into a sequence of predefined phases.

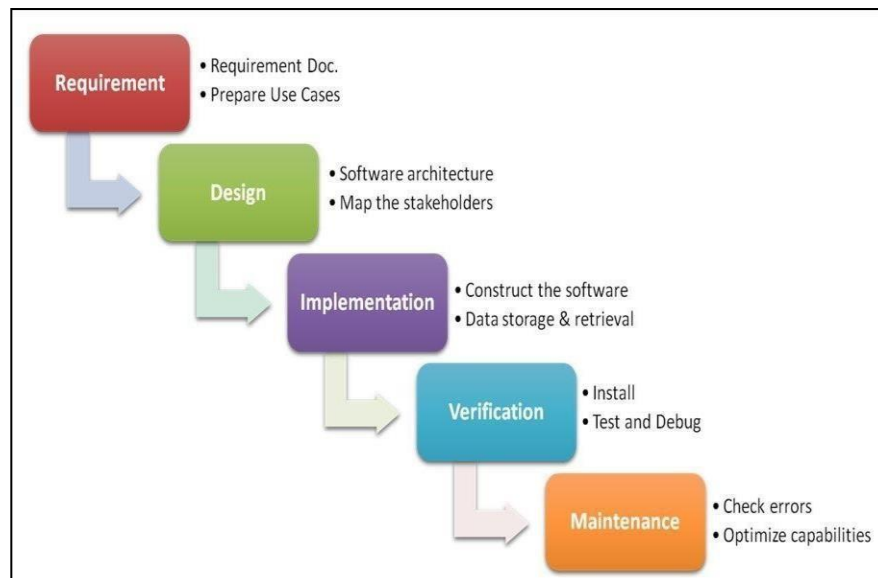


Fig 3.2: waterfall Model

1. Requirement gathering and documentation

In this stage, you should gather comprehensive data approximately what this challenge requires. You may gather this data in a diffusion of ways, from interviews to questionnaires to interactive brainstorming. By means of the stop of this section, the task requirements have to be clean, and also you have to have a necessities file that has been allotted on your team.

2. System design

Using the well-known requirements, your team designs the solutions. During this phase, no development will be happening. But the project team starts specification such as programming language or hardware requirements.

3. Implementation

During this phase software development coding will be happening. Web application programmers take data from the previous stage and create a functional product. Web application programmers write source code in small pieces, which are integrated at the end of this phase or the beginning of the next.

4. Testing

Once all coding is done, testing of the product can begin. Testers methodically find and report any problems. If serious issues arise, your project may need to return to phase one for reevaluation.

5. Delivery/deployment

In this phase, the solution is complete, and your project team submits the deliverables to be deployed or released.

6. Maintenance

The final solution has been implemented to the client and is being used. As a trouble arises, the project team might also want to create patches and updates might also to deal with them. Again, huge troubles may also necessitate a return to segment one.

RAD Model

Agile is an iterative, team-based approach to development. This method highlights the speedy delivery of an software application in comprehensive functional components.

Rapid Application Development (RAD) is a form of agile software development methodology that prioritizes rapid prototype releases and iterations. Unlike the Waterfall method, RAD emphasizes the use of software and user feedback over strict planning and requirements recording.

Phase 1: Requirements planning

This phase is equivalent to a project scoping meeting. Although the planning phase is condensed compared to other project management methodologies, this is a critical step for the ultimate success of the project.

During this stage, web application programmers, clients (software users), and team members communicate to determine the goals and expectations for the project as well as current and potential issues that would need to be addressed during the build. RAD model distributes the analysis, design, build and test phases into a series of short, iterative development cycles.

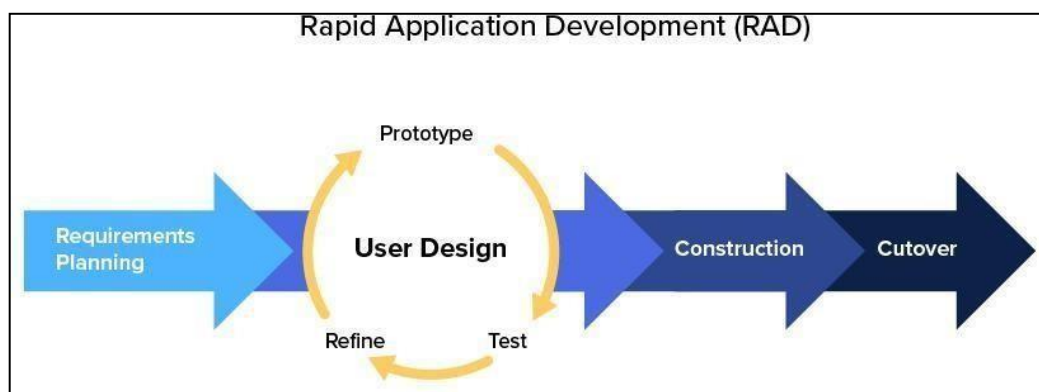


Fig. 3.3: Rapid Application Development Architecture

RAD Model (Taken from Google.com)

References (<https://www.lucidchart.com/blog/rapid-application-development-methodology>)

A basic breakdown of this stage involves:

Researching the current problem Defining the requirements for the project

Finalizing the requirements with each stakeholder's approval

It is important that everyone has the opportunity to evaluate the goals and expectations for the project and weigh in. By getting approval from each key stakeholder and web application programmer, teams can avoid miscommunications and costly change orders down the road.

Phase 2: User design

Once the project is scoped out, it's time to jump right into development, building out the user design through various prototype iterations.

This is the meat and potatoes of the RAD methodology and what sets it apart from other project management strategies. During this phase, clients work hand in hand with web application programmers to ensure their needs are being met at every step in the design process. It's almost like customizable software development where the users can test each prototype of the product, at each stage, to ensure it meets their expectations.

All the bugs and kinks are worked out in an iterative process. The web application programmer designs a prototype, the client (user) tests it, and then they come together to communicate on what worked and what didn't.

This method gives web application programmers the opportunity to tweak the model as they go until they reach a satisfactory design.

Both the software web application programmers and the clients learn from the experience to make sure there is no potential for something to slip through the cracks.

Phase 3: Rapid construction

Phase 3 takes the prototypes and beta systems from the design phase and converts them into the working model.

Because the majority of the problems and changes were addressed during the thorough iterative design phase, web application programmers can construct the final working model more quickly than they could by following a traditional project management approach. The phase breaks down into several smaller steps:

- Preparation for rapid construction
- Program and application development Coding
- Unit, integration, and system testing

The software development team of programmers, coders, testers, and web application programmers work together during this stage to make sure everything is working smoothly and that the end result satisfies the client's expectations and objectives.

This third phase is important because the client still gets to give input throughout the process. They can suggest alterations, changes, or even new ideas that can solve problems as they arise.

Phase 4: Cutover

This is the implementation phase where the finished product goes to launch. It includes data conversion, testing, and changeover to the new system, as well as user training.

All final changes are made while the coders and clients continue to look for bugs in the system.

Benefits of RAD methodology

RAD is one of the most successful software development programs available today, with numerous benefits for both software development teams as well as their clients.

Here are just a few advantages:

RAD lets you break the project down into smaller, more manageable tasks.

The task-oriented structure allows project managers to optimize their team's efficiency by assigning tasks according to members' specialties and experience.

Clients get a working product delivered in a shorter time frame.

Regular communication and constant feedback between team members and stakeholders increases the efficiency of the design and build process.

CHAPTER 4

METHODOLOGY

4.4 HARDWARE REQUIREMENT

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented.

- PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.
- RAM : 512 MB DDRAM
- MONITOR : 15"
- COLOR HARD DISK : 40 GB

4.5 SOFTWARE REQUIREMENT

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team and tracking the team's progress throughout the development activity.

Front End : Web Technologies, Bootstrap Client Side - Html

Backend : Python, numpy , Eclipse IDE

Operating System : Windows 07

4.6 MODULES

The modules present in this project are far more the best ways to get the required output. 3 modules are present as follows:

- ★ Preprocessing
- ★ Feature Selection
- ★ Data encoding and decoding

Module 1: Preprocessing

A Fourier transform converts a time-domain signal to the frequency domain. In other words, a spectrum is the frequency domain representation of the input audio's time-domain signal.

Noise Removal: At the time of image acquisition or during transmission, noises are produced. Several factors are liable for the formation of noise while acquisition or transmission. Subject to the type of noise the image can affect, it degrades the image quantification to different range. In general, the noises in the image can be classified as i) impulse noise (salt & pepper noise) - alternative black and white marks appear on the image hence it name as salt & pepper. This may be due to the sharp and abrupt deviations in image signal. ii) Gaussian noise (Amplifier noise) - each noisy pixel in the image is the summation of the correct pixel value and arbitrary Gaussian distributed value. The clamor produced due to this type, is independent of the pixel intensity.iii) Speckle noise (Multiplicative noise) this type of noise was formed by multiplying some random value of uniform noise with the normal pixel. This noise weakens the quality of active RADAR. In conservative RADAR system, this type of noise is detected, only when the resumed signal from the object having less than or equal to a distinct processing unit. Poisson noise or photon noise is sensed only when the number of photons detected by the sensor are not satisfactory to Mathematical operations: By considering the image arithmetic operations, application of one of the standard mathematical or logical operations to two or more images is the option. The operators are applied in a

step-by-step manner. That is, the value of the output pixel depends only on the value of the input pixel. Therefore, the size of the image must be the same. The major advantage of using mathematical operations is that, it is very fast and simple to implement. Alike mathematical operations, logical operations are frequently used to combine two or more binary images. In the circumstance of digital images, the logical operator is usually applied in a bit-wise manner

Module 2: Feature Selection

High dimensional data consists on features that can be irrelevant, misleading, or redundant which increase search space size resulting in difficulty to process data further thus not contributing to the learning process. Feature subset selection is the process of selecting best features among all the features that are useful to discriminate classes. Feature selection algorithm (FSA) is a computational model that is provoked by a certain definition of relevance.

Correlation-Based Feature Selection searches feature subset according to the degree of redundancy among the features. The evaluation process aims to find subsets of features that are individually highly correlated with the class but have low inter- correlation. Relevance of group of features grows with the correlation between features and class, and decreases with growing inter-correlation. CFS is used to determine the best feature subset and is usually combined with search strategies such as forward selection, backward elimination, bi-directional search, best first search and genetic search.

Feature construction is one of the key steps in the data analysis process, largely conditioning the success of any subsequent statistics or machine learning endeavor. In particular, one should beware of not losing information at the feature construction stage. It may be a good idea to add the raw features to the preprocessed data or at least to compare the performances obtained with either representation.

We are decomposing the problem of feature extraction in two steps: feature construction, briefly reviewed in the previous section, and feature selection, to which we are now directing our attention.

Module 3: Data Encoding & Decoding

Data hiding process embeds data into digital media for the purpose of security. Digital image is one of the best media to store data. It provides large capacity for hiding secret information which results into stego-image imperceptible to human vision, a novel steganographic approach based on data hiding method such as pixel- value differencing. This method provides both high embedding capacity and outstanding imperceptibility for the stego-image. In this paper, different image processing techniques are described for data hiding related to pixel value differencing. Pixel Value Differencing based techniques is carried out to produce modified data hiding method. Hamming is an error correcting method which is useful to hide some information where lost bit are detected and corrected.

If the pixel difference is located in the edge area, then it can tolerate large distortions which means good embedding site. In order to achieve the goal, predictive value for generating the differencing, also called prediction error (PE), is proposed in this paper. Our proposed scheme contains two procedures: embedding procedures and extracting procedures.

Cryptography and Steganography

Encryption is a method of storing and transferring certain types of data, so that only intended users can read and process it. The cryptographic system can be broadly categorized as a symmetric key system using a single key for both sender and receiver, a public key system using both keys, a public key known to all, and a private key recipient of the message [1] () Is used. Common terms used in encryption are: 1) Plain text 2) Encrypted text 3) Encryption 4) Decryption 5) Key. For example, if secret data to be transmitted is encrypted, channel providers who have no knowledge of the encryption key tend to compress the encrypted data due to limited channel resources. Encrypted binary images can be compressed without loss by detecting the low density parity check code syndrome [1], but encrypted gray image lossless compression method compatible with sequential decomposition and turbo puncturing codes The flow is developed in [2]. Using the lossy compression method proposed in [3], the encrypted gray image can be effectively compressed by ignoring excessive coarse information from the

coefficients generated from the orthogonal transformation. When compressing data, the receiver can reconstruct the main content of the original image by retrieving the count value. Conversion calculations in encrypted domains have also been studied. Based on the similarity characteristics of the basic cryptosystem, a discrete Fourier transform can be implemented in the encrypted domain [4]. In [5], a composite signal representation method that combines multiple signal samples and treats them as a single sample is used to reduce the computational complexity and size of the encrypted data. There is also a certain amount of work on hidden data in encrypted domains. Buyer Cellular Protocol Watermark [6], a digital media product provider, encrypts and integrates original data with an encrypted fingerprint public key switch provided by an encrypted domain buyer. After decrypting with a private key, the buyer can get a watermarked product. This protocol allows the seller to find the version watermark of the buyer, although the buyer does not know the original version. Okamoto-Uchiyama has been proposed to improve the speed of greetings using encryption methods [7]. Encryption of public keys by homomorphic complex signal overload calculation The introduction of a mechanism to express large Communication bandwidth is greatly reduced.

CHAPTER 5

RESULT AND DISCUSSION

We recap a fact of stream-cipher-based SIK schemes with respect to security [31]. This work points out the weakness of stream-cipher-based SIK schemes, i.e., [10, 19, 21, 22]. However, such the attack does not appear in (multi-) secret sharing and Paillier-based schemes.

The testing images and encrypted results shown in Figure 5.3 are of the proposed scheme. There is no information leakage on the encrypted images. With the security advantage, we further discuss the effectiveness in the aspects of payload and image quality. We focus on the results of proposed scheme, presented in Section IV, at first, and postpone the comparisons with

the relevant schemes to the next paragraph. For payload and image quality, experiments are shown in Figure 4 by taking a few typical cover-images. According to the results shown in Figure 4, we conclude that the proposed SOK-type method inherits the properties from the use of difference expansion (DE), and thus it leads better PSNR and payload in smooth images (like Lena). In addition, we compare our scheme with some typical and state-of-the-art works [10, 19, 21, 22, 25, 27, 28, 32, 33] on Lena and Baboon which are representatives of smooth and complex images. The comparison results are provided in Figures 5 and 6 respectively. However, difference expansion like schemes (including [27, 28] and the proposed) enjoy almost identical PSNR and payload, and the extremely negligible difference comes from the use of underlying encryption. Finally, in Appendix C, we will show an alternative solution by modifying from the scheme of Zhang et al. [25]. Its payload and PSNR are almost identical to the results of [25] as well. Finally, we use 10 random images from the USC-SIPI image database and perform the experiments for the maximum payload. The results are shown in Table I. The schemes of [10, 19, 21] are based on LSB flipping, and thus payload is bounded. However, to increase payload is achieved by shrinking the size of blocks, but this cannot recover the cover images. As a result, only less payload can guarantee lossless image recovery in these schemes [10, 19, 21]. The scheme of [22] use pre-processing to restrict the number of peaks in the encrypted image, but

preserve nice image quality. That of [23] applies the traditional RDH method, and also preserve nice payload and image quality. However, that of [27] relies on pre-processing for different expansion, except for some cost of the location map, payload is close to 0.5bpp. That of [25] also can achieve 0.5bpp ideally (the results are influenced by the setting of threshold). Finally, the scheme of [32] takes the reserving room technique, where P has to perform self embedding on bit stream of target areas. However, this induces a few embedding issues, so the result is roughly 0.4bpp. The implementation results are shown in Table III. Obviously, stream-cipher is the most faster than others since this relies on exclusive OR operations, Paillier-based is from RSA-like arithmetic with complicated and mathematical structures, and secret sharing is from arithmetic over small modulo and polynomial properties. However, with packing supported by multi-secret sharing, our method is more efficient than [28], and packs up multiple pixels per time. As a conclusion, since stream-cipher-based schemes face the weakness on security, the proposed scheme The implementation results are shown in Table III. Obviously, stream-cipher is the most faster than others since this relies on exclusive OR operations, Paillier-based is from RSA-like arithmetic with complicated and mathematical structures, and secret sharing is from arithmetic over small modulo and polynomial properties. However, with packing supported by multi-secret sharing, our method is more efficient than [28], and packs up multiple pixels per time.

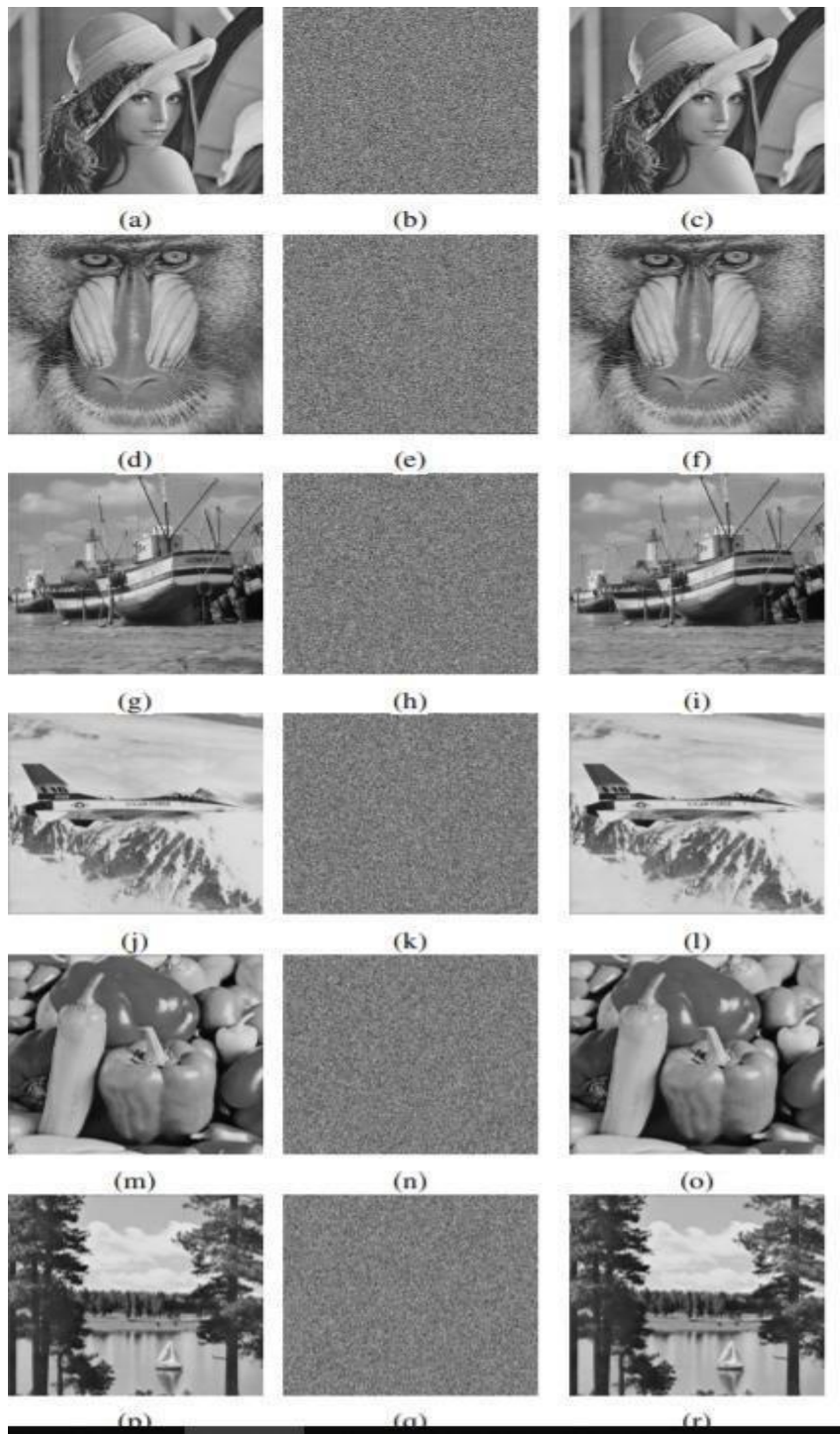


Fig 5.1Left: Cover-image. Middle: Encrypted, Right: Stego (after decryption)

For the discussion of efficiency, we aim for two aspects, theoretical and practical results, to analyze the schemes. 1) Theoretical results: We show the theoretical analysis in Table II with asymptotic notions, where the number of pixels is denoted by N . Let p and p_0 be the size of an encrypted unit and encryption run-time of a unit in Paillier encryption. The Paillier-based schemes [11, 27] perform encryption for each pixel, and thus the space cost and encryption time are pN and p_0N .

In addition, they achieve SNK-type. However, secret sharing and multi-secret sharing-based schemes are SOK type. For a encryption unit, we denote threshold of secret sharing by n and run-time of share generation by s . The mathematical backgrounds of Paillier encryption and secret sharing are different, but, according to some real-life practical implementations, we say $p \ll n$ and $p_0 \ll s$ (Later we will provide some results regarding implementations). The secret sharing-based [28] tackles each pixel to perform sharing, and

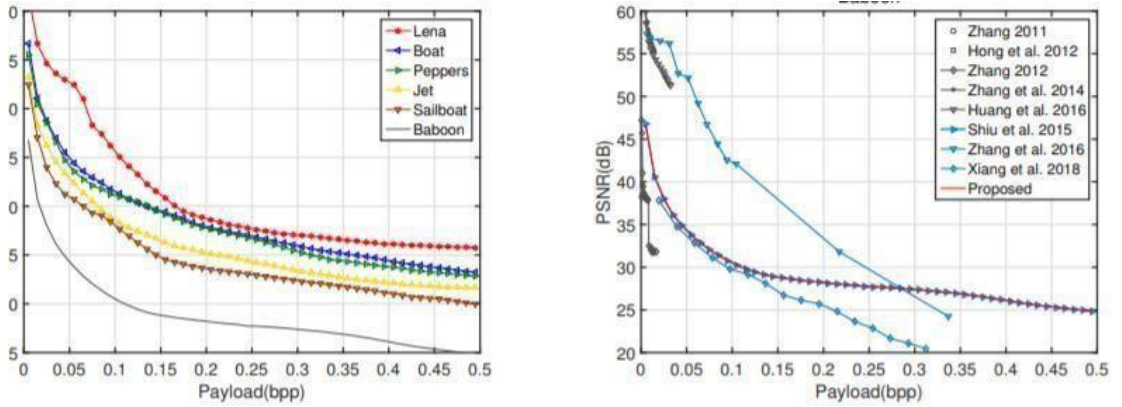


Fig - 5.2 Graphical survey

CHAPTER 6

CONCLUSION AND SUMMARY

To achieve scalable services over millions of encrypted images, we design a secure and efficient index structure, which enables practical and accurate social discovery from the cloud, without revealing any image profile or image content. For completeness, we further enrich our service with secure updates, facilitating user's image update.

Our framework not only protects the outsourced photos so that no unauthorized users can access them, but also enables users to encode their image search so that the search can also be outsourced to an untrusted cloud server obviously without leakage on the query contents or results. Our analysis shows the security of the framework, and the implementation shows a small storage overhead and communication overhead for both mobile clients and cloud servers. In this paper, we present a new class of reversible data hiding in encrypted images, referred to as shared-one-key (SOK). In this class, only the image provider has a shared secret key with the receiver, and in particular, anyone who knows the embedding procedure can hide. For flexibility, SOK is much weaker than SNK. However, the existing SNK schemes rely on additive homomorphic encryption. We use secret sharing as the underlying ingredient to construct our SOK scheme to achieve better efficiency and preserve the total size. Then, we convert a SNK scheme with some properties to a SOK version. To demonstrate the effectiveness, we provide a full description of the SOK scheme from the SNK schemes. Finally, we intend to conduct a subsequent study, so propose

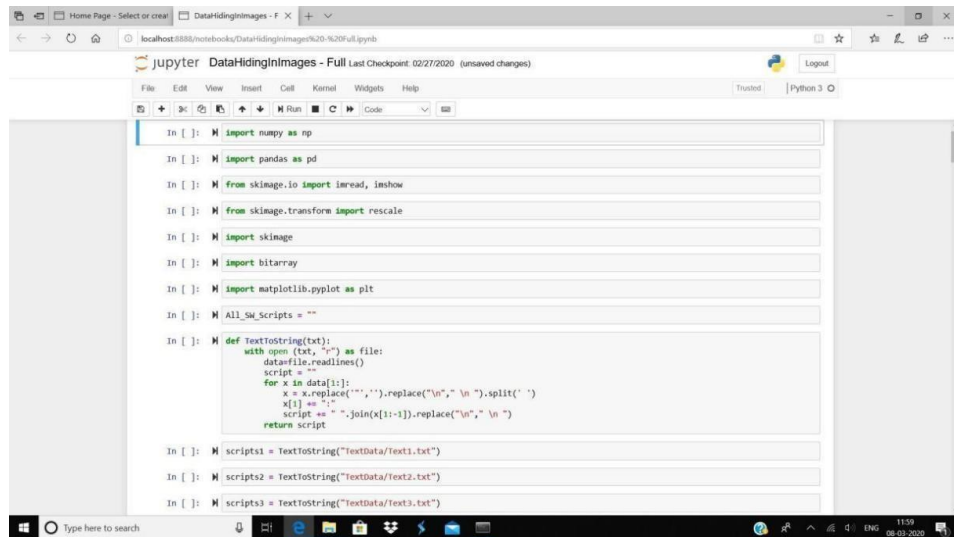
REFERENCES

- [1] J. Tian, "Reversible data embedding using a difference expansion," IEEE transactions on circuits and systems for video technology, vol. 13, no. 8, pp. 890–896, 2003.
- [2] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," IEEE Transactions on circuits and systems for video technology, vol. 16, no. 3, pp. 354–362, 2006.
- [3] W. Hong, "Adaptive reversible data hiding method based on error energy control and histogram shifting," Optics Communications, vol. 285, no. 2, pp. 101–108, 2012.
- [4] W. Hong and T.-S. Chen, "A local variance-controlled reversible data hiding method using prediction and histogram-shifting," Journal of Systems and Software, vol. 83, no. 12, pp. 2653–2663, 2010.
- [5] S.-W. Jung, S.-J. Ko et al., "A new histogram modification based reversible data hiding algorithm considering the human visual system," IEEE Signal Processing Letters, vol. 18, no. 2, pp. 95–98, 2011.
- [6] Y.-Y. Tsai, D.-S. Tsai, and C.-L. Liu, "Reversible data hiding scheme based on neighboring pixel differences,"
- [7] Tsai, Y.-C. Hu, and H.-L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," Signal Processing, vol. 89, no. 6, pp. 1129–1143, 2009.
- [8] D. M. Thodi and J. J. Rodríguez, "Expansion embedding techniques for reversible watermarking," IEEE transactions on image processing, vol. 16, no. 3, pp. 721–730, 2007.
- [9] W.-L. Tai, C.-M. Yeh, and C.-C. Chang, "Reversible data hiding based on histogram modification of pixel differences," IEEE transactions on circuits and systems for video technology, vol. 19, no. 6, pp. 906–910, 2009.
- [10] X. Zhang, "Reversible data hiding in encrypted image," IEEE signal processing letters, vol. 18, no. 4, pp. 255–258, 2011.
- [11] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: advances in the past two decades," IEEE Access, vol. 4, pp. 3210–3237, 2016.
- [12] J. Fridrich, M. Goljan, and R. Du, "Invertible authentication watermark for jpeg images," in Information Technology: Coding and Computing, 2001. Proceedings. International Conference on. IEEE, 2001, pp. 223–227.
- [13] H. Sakai, M. Kuribayashi, and M. Morii, "Adaptive reversible data hiding for jpeg images," in Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on. IEEE, 2008, pp. 1–6.
- [14] Nikolaidis, "Reversible data hiding in jpeg images utilising zero quantised coefficients," IET Image Processing, vol. 9, no. 7, pp. 560–568, 2015.
- [15] . Ni, Y. Q. Shi, N. Ansari, W. Su, Q. Sun, and X. Lin, "Robust lossless image data hiding," in Multimedia and Expo, 2004. ICME'04. 2004 IEEE

- International Conference on, vol. 3. IEEE, 2004, pp.2199–2202.
- [16] “Robust lossless image data hiding designed for semi-fragile image authentication,” IEEE Transactions on circuits and systems for video technology, vol. 18, no. 4, pp. 497–509,2008.
 - [17] D. Zou, Y. Q. Shi, Z. Ni, and W. Su, “A semi-fragile lossless digital watermarking scheme based on integer wavelet transform,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 10, pp. 1294– 1300,2006.
 - [18] M. S. A. Karim and K. Wong, “Universal data embedding in encrypted domain,” Signal Processing, vol. 94, pp. 174–182,2014.
 - [19] W. Hong, T.-S. Chen, and H.-Y. Wu, “An improved reversible data hiding in encrypted images using side match,” IEEE Signal Processing Letters, vol. 19, no. 4, pp. 199–202,2012.
 - [20] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, “Reversible data hiding in encrypted images by reserving room before encryption,” IEEE Transactions on information forensics and security, vol. 8, no. 3, pp. 553–562,2013.
 - [21] X. Zhang, “Separable reversible data hiding in encrypted image,” IEEE transactions on information forensics and security, vol. 7, no. 2, pp. 826–832, 2012.

APPENDICES

A. SOURCE CODE



```
In [ ]: import numpy as np

In [ ]: import pandas as pd

In [ ]: from skimage.io import imread, imshow

In [ ]: from skimage.transform import rescale

In [ ]: import skimage

In [ ]: import bitarray

In [ ]: import matplotlib.pyplot as plt

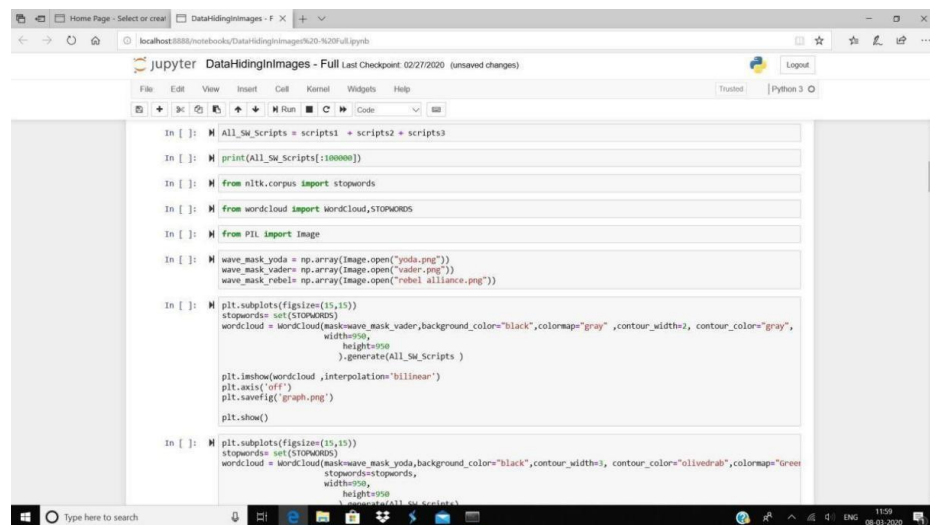
In [ ]: All_Sm_Scripts = ""

In [ ]: def TextToScript(txt):
    with open(txt, "r") as file:
        datafile.readlines()
        script = ""
        for x in datafile.readlines():
            x = x.replace(" ", "").replace("\n", " ").split(' ')
            x[1] += ":"
            script += " ".join(x[1:-1]).replace("\n", " ")
        return script

In [ ]: scripts1 = TextToScript("TextData/Text1.txt")

In [ ]: scripts2 = TextToScript("TextData/Text2.txt")

In [ ]: scripts3 = TextToScript("TextData/Text3.txt")
```



```
In [ ]: All_Sm_Scripts = scripts1 + scripts2 + scripts3

In [ ]: print(len(All_Sm_Scripts))

In [ ]: from nltk.corpus import stopwords

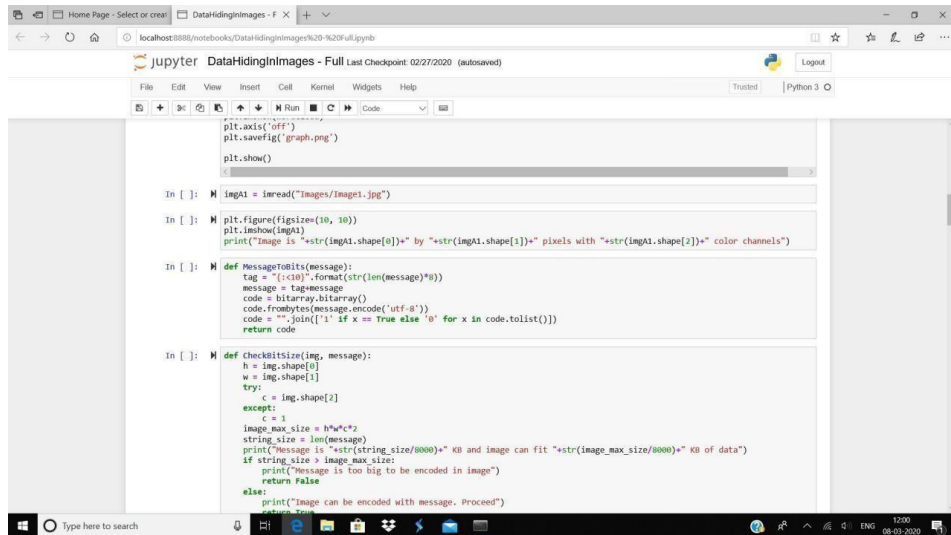
In [ ]: from wordcloud import WordCloud, STOPWORDS

In [ ]: from PIL import Image

In [ ]: wave_mask_yoda = np.array(image.open("yoda.png"))
    wave_mask_vader = np.array(image.open("vader.png"))
    wave_mask_rebel = np.array(image.open("rebel alliance.png"))

In [ ]: plt.subplots(figsize=(15,15))
    stopwords = set(STOPWORDS)
    wordcloud = WordCloud(mask=wave_mask_vader, background_color="black", colormap="gray",
        width=998, height=998,
        ).generate(All_Sm_Scripts )
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.savefig('graph.png')
    plt.show()

In [ ]: plt.subplots(figsize=(15,15))
    stopwords = set(STOPWORDS)
    wordcloud = WordCloud(mask=wave_mask_yoda, background_color="black", contour_width=3, contour_color="olive", colormap="green",
        width=998, height=998,
        ).generate(All_Sm_Scripts )
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.savefig('graph.png')
    plt.show()
```



The screenshot shows a Jupyter Notebook titled "DataHidingInImages - Full Last Checkpoint: 02/27/2020 (autosaved)". The code in the notebook includes:

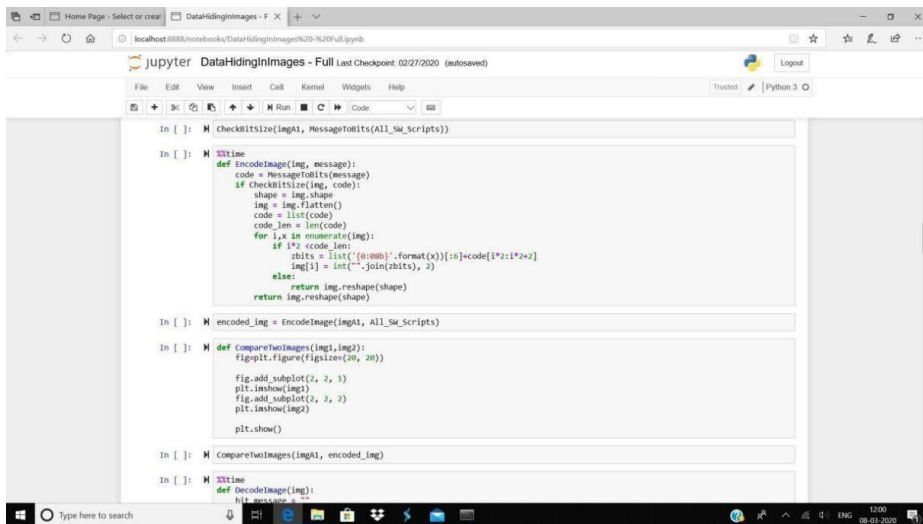
```
plt.axis('off')
plt.savefig('graph.png')
plt.show()

In [ ]: M imgA1 = imread("Images/image1.jpg")

In [ ]: M plt.figure(figsize=(10, 10))
plt.imshow(imgA1)
print("Image is "+str(imgA1.shape[0])+" by "+str(imgA1.shape[1])+" pixels with "+str(imgA1.shape[2])+" color channels")

In [ ]: M def MessageToBits(message):
tag = "<img>".format(str(len(message)*8))
message = tag+message
code = bytearray.bitarray()
code.frombytes(message.encode('utf-8'))
code = "".join('1' if x == True else '0' for x in code.tolist())
return code

In [ ]: M def CheckBitsize(img, message):
h = img.shape[0]
w = img.shape[1]
try:
c = img.shape[2]
except:
c = 3
image_max_size = h*w*c*2
string_size = len(message)
print("Message is "+str(string_size/8000)+" KB and image can fit "+str(image_max_size/8000)+" KB of data")
if string_size > image_max_size:
print("Message is too big to be encoded in image")
return False
else:
print("Image can be encoded with message. Proceed")
```



The screenshot shows the continuation of the Jupyter Notebook with the following code:

```
In [ ]: M checkBitsize(imgA1, MessageToBits(All_SM_Scripts))

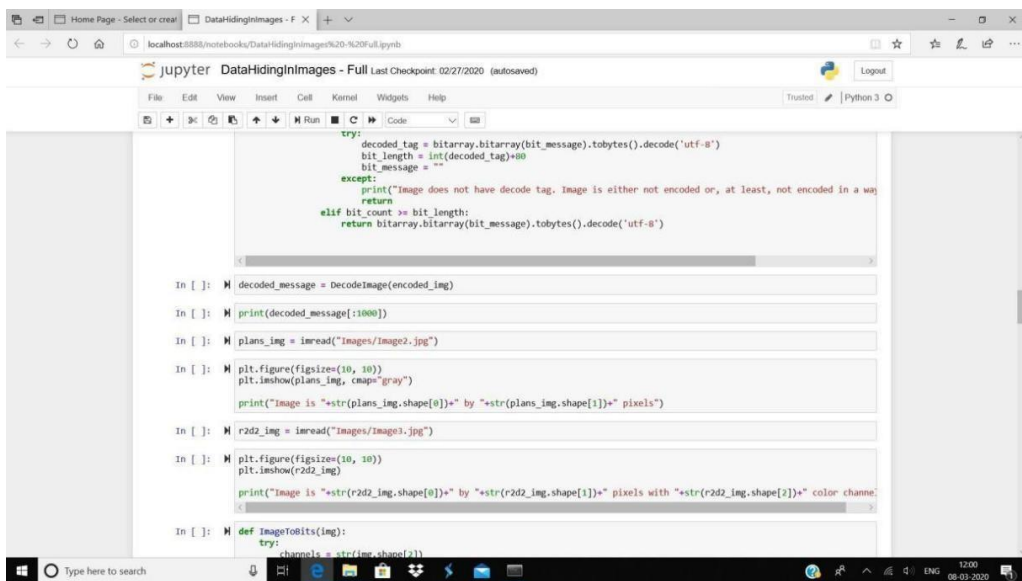
In [ ]: M @time
def EncodeImage(img, message):
code = MessageToBits(message)
if checkBitsize(img, code):
shape = img.shape
img = img.flatten()
code = list(code)
code_len = len(code)
for i,x in enumerate(img):
if i% code_len:
zbits = list('0:800b').format(x)[0:code[i%2:i%2+2]]
img[i] = int("".join(zbits), 2)
else:
return img.reshape(shape)
return img.reshape(shape)

In [ ]: M encoded_img = EncodeImage(imgA1, All_SM_Scripts)

In [ ]: M def CompareTwoImages(img1,img2):
fig=plt.figure(figsize=(20, 20))
fig.add_subplot(2, 2, 1)
plt.imshow(img1)
fig.add_subplot(2, 2, 2)
plt.imshow(img2)
plt.show()

In [ ]: M CompareTwoImages(imgA1, encoded_img)

In [ ]: M @time
def DecodeImage(img):
bits_message = ""
```



A Jupyter Notebook interface titled "DataHidingInImages - Full Last Checkpoint: 02/27/2020 (autosaved)". The code in the notebook includes a function to decode a message from a bit array, followed by several interactive cells. These cells load and display two images: "plans_img" (Image2.jpg) and "r2d2_img" (Image3.jpg). The notebook also contains a function to convert an image to bits.

```
try:
    decoded_tag = bitarray.bitarray(bit_message).tobytes().decode('utf-8')
    bit_length = int(decoded_tag)*80
    bit_message = ""
except:
    print("Image does not have decode tag. Image is either not encoded or, at least, not encoded in a way")
    return
elif bit_count >= bit_length:
    return bitarray.bitarray(bit_message).tobytes().decode('utf-8')
```

```
In [ ]: decoded_message = DecodeImage(encoded_img)
```

```
In [ ]: print(decoded_message[:1000])
```

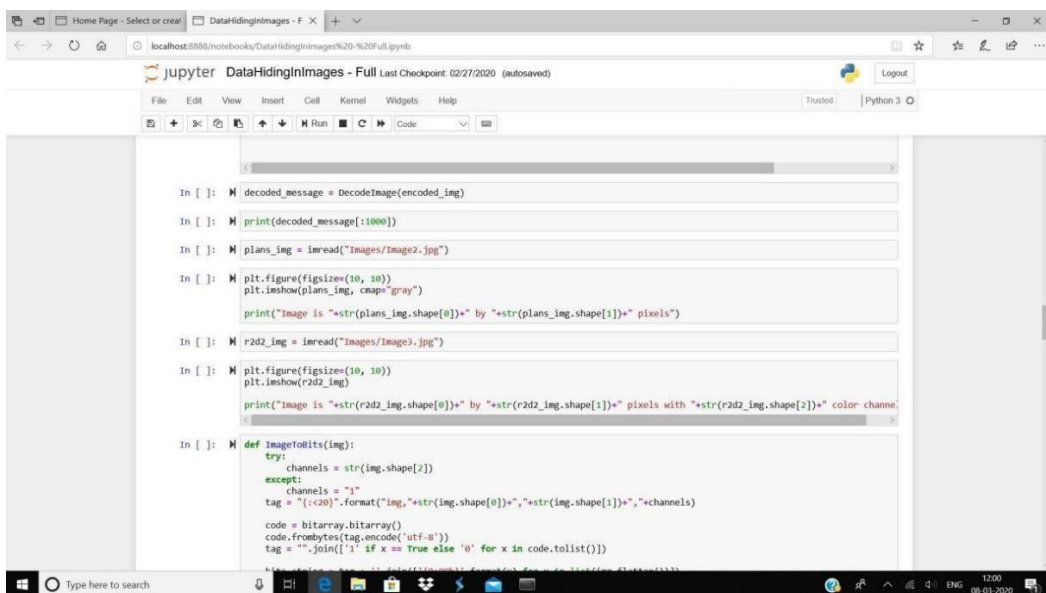
```
In [ ]: plans_img = imread("Images/Image2.jpg")
```

```
In [ ]: plt.figure(figsize=(10, 10))
plt.imshow(plans_img, cmap="gray")
print("Image is "+str(plans_img.shape[0])+" by "+str(plans_img.shape[1])+" pixels")
```

```
In [ ]: r2d2_img = imread("Images/Image3.jpg")
```

```
In [ ]: plt.figure(figsize=(10, 10))
plt.imshow(r2d2_img)
print("Image is "+str(r2d2_img.shape[0])+" by "+str(r2d2_img.shape[1])+" pixels with "+str(r2d2_img.shape[2])+" color channels")
```

```
In [ ]: def ImageToBits(img):
try:
    channels = str(img.shape[2])
```



A Jupyter Notebook interface titled "DataHidingInImages - Full Last Checkpoint: 02/27/2020 (autosaved)". The code in the notebook includes a function to encode a message into a bit array, followed by several interactive cells. These cells load and display two images: "plans_img" (Image2.jpg) and "r2d2_img" (Image3.jpg). The notebook also contains a function to convert an image to bits.

```
In [ ]: decoded_message = DecodeImage(encoded_img)
```

```
In [ ]: print(decoded_message[:1000])
```

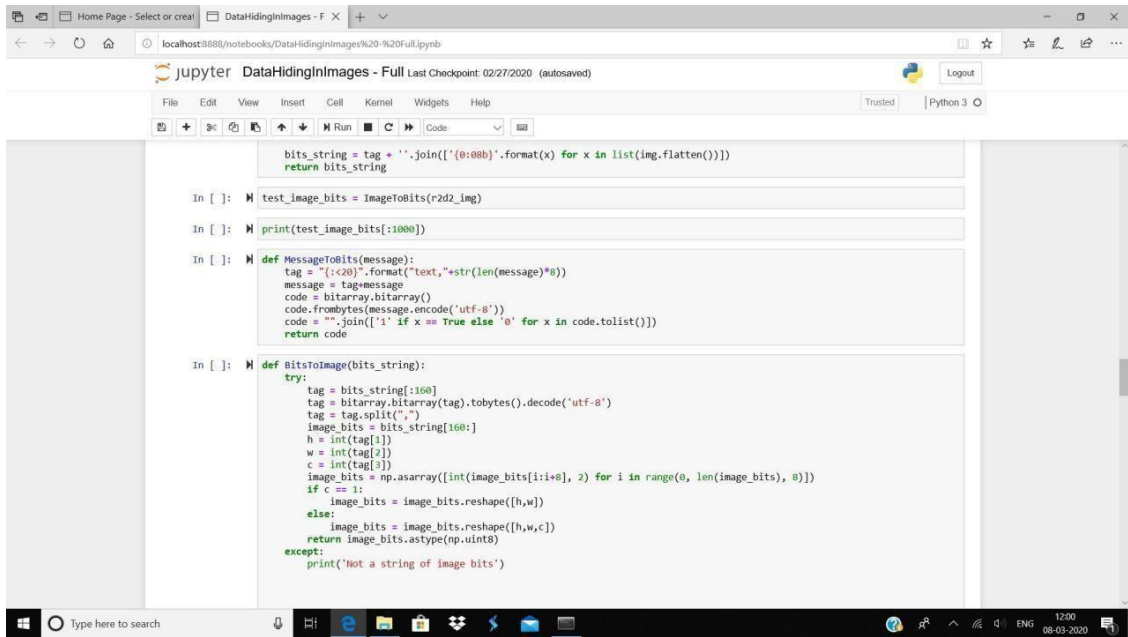
```
In [ ]: plans_img = imread("Images/Image2.jpg")
```

```
In [ ]: plt.figure(figsize=(10, 10))
plt.imshow(plans_img, cmap="gray")
print("Image is "+str(plans_img.shape[0])+" by "+str(plans_img.shape[1])+" pixels")
```

```
In [ ]: r2d2_img = imread("Images/Image3.jpg")
```

```
In [ ]: plt.figure(figsize=(10, 10))
plt.imshow(r2d2_img)
print("Image is "+str(r2d2_img.shape[0])+" by "+str(r2d2_img.shape[1])+" pixels with "+str(r2d2_img.shape[2])+" color channels")
```

```
In [ ]: def ImageToBits(img):
try:
    channels = str(img.shape[2])
except:
    channels = "1"
tag = "{:020}".format("img,"+str(img.shape[0])+", "+str(img.shape[1])+", "+channels)
code = bitarray.bitarray()
code.frombytes(tag.encode('utf-8'))
tag = "".join(['1' if x == True else '0' for x in code.tolist()])
```



The screenshot shows a Jupyter Notebook titled "DataHidingInImages - Full Last Checkpoint 02/27/2020 (autosaved)". The notebook is running on a local host. The code in the notebook defines a function `BitsToImage` that takes a string of bits and returns an image. The function uses a tag to store the dimensions of the image and the message. The code is as follows:

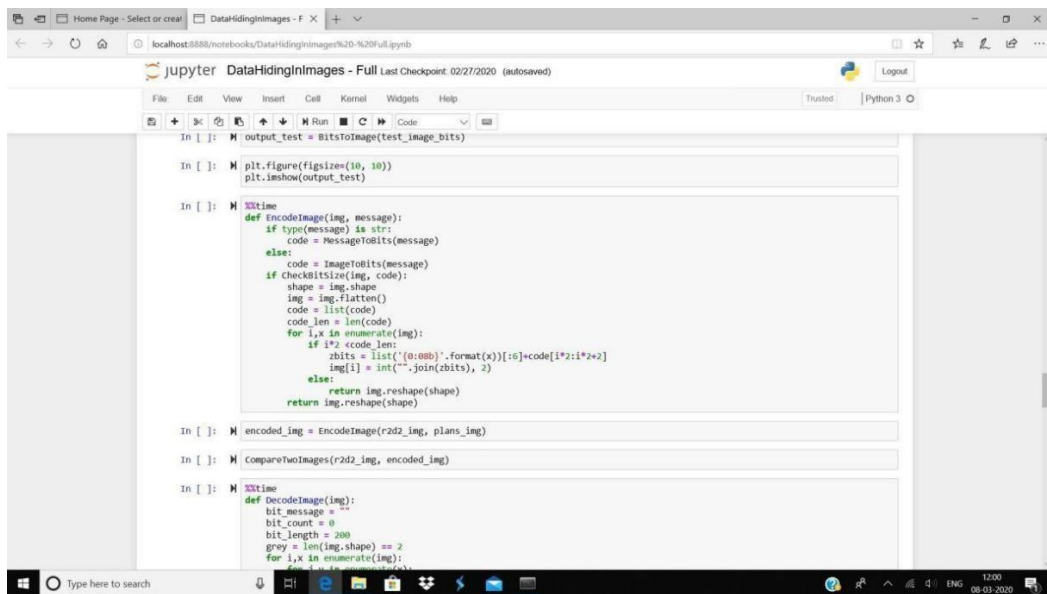
```
bits_string = tag + ''.join(['0:00b'.format(x) for x in list(img.flatten())])
return bits_string

In [ ]: M test_image_bits = ImageToBits(r2d2_img)

In [ ]: M print(test_image_bits[:1000])

In [ ]: M def MessageToBits(message):
    tag = "{:<20}".format("text,"+str(len(message)*8))
    message = tag+message
    code = bytearray.bitarray()
    code.frombytes(message.encode('utf-8'))
    code = ''.join(['1' if x == True else '0' for x in code.tolist()])
    return code

In [ ]: M def BitsToImage(bits_string):
    try:
        tag = bits_string[:160]
        tag = bytearray.bitarray(tag).tobytes().decode('utf-8')
        tag = tag.split(",")
        image_bits = bits_string[160:]
        h = int(tag[1])
        w = int(tag[2])
        c = int(tag[3])
        image_bits = np.asarray([int(image_bits[i:i+8], 2) for i in range(0, len(image_bits), 8)])
        if c == 1:
            image_bits = image_bits.reshape([h,w])
        else:
            image_bits = image_bits.reshape([h,w,c])
        return image_bits.astype(np.uint8)
    except:
        print('Not a string of image bits')
```



The screenshot shows a Jupyter Notebook titled "DataHidingInImages - Full Last Checkpoint 02/27/2020 (autosaved)". The notebook is running on a local host. The code in the notebook defines two functions: `EncodeImage` and `DecodeImage`. The `EncodeImage` function takes an image and a message and returns an encoded image. The `DecodeImage` function takes an encoded image and returns the original image and message. The code is as follows:

```
In [ ]: M plt.figure(figsize=(10, 10))
plt.imshow(output_test)

In [ ]: M %time
def EncodeImage(img, message):
    if type(message) is str:
        code = MessageToBits(message)
    else:
        code = ImageToBits(message)
    if checkBitsize(img, code):
        shape = img.shape
        img = img.flatten()
        code = list(code)
        code_len = len(code)
        for i,x in enumerate(img):
            if i%2 < code_len:
                zbits = list('0:00b'.format(x))[1:6]+code[i%2:i%2+2]
                img[i] = int(''.join(zbits), 2)
            else:
                return img.reshape(shape)
        return img.reshape(shape)

In [ ]: M encoded_img = EncodeImage(r2d2_img, plans_img)

In [ ]: M CompareTwoImages(r2d2_img, encoded_img)

In [ ]: M %time
def DecodeImage(img):
    bit_message = ""
    bit_count = 0
    bit_length = 200
    grey = len(img.shape) == 2
    for i,x in enumerate(img):
        for j,y in enumerate(x):
            if i%2 < bit_count:
```

B. Screen Shots

