

Secure Data Communication Using Cryptography  
and Steganography



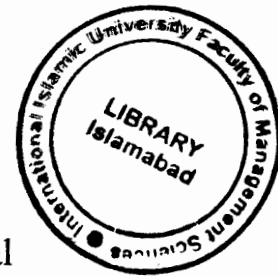
Acc. No. (PMS) T-1407

by

Nighat Mir, Reg#153-CS/MS/2003  
Beenish Aziz, Reg# 155-CS/MS/2003

*Supervised by*

Dr. Syed Afaq Husain  
Dr. Malik Sikandar Hayat Khiyal



Department of Computer Science  
Faculty of Applied Sciences  
International Islamic University, Islamabad.  
(2006)

T01407

**DATA ENTERED**

2/24/88  
MTD

MS

005-8  
NIS

- 1- Computers - Access control
- 2- Data encryption (computer science)
- 3- Cryptography
- 4- Steganography.

**Department of Computer Sciences  
International Islamic University Islamabad**

**Final Approval**

Dated 23-08-2016

It is certified that we have read this thesis report submitted by Night Mir and Beenish Aziz , Registration Number 153-CS/MS/03 and 155-CS/MS/03, and it our judgment that this thesis is of sufficient standard warrant to acceptance by International Islamic University , Islamabad for MS in Computer Science.

**COMMITTEE**

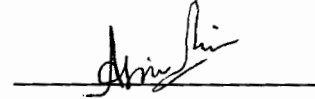
**External Examiner**

Dr. Ijaz Mansoor Qureshi  
Head (Department of Computer Engineering)  
Muhammad Ali Jinnah University  
Islamabad, Pakistan

  
\_\_\_\_\_

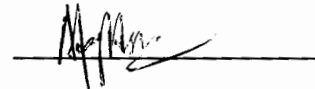
**Internal Examiner**

Mr. Asim Munir  
Assistant Professor,  
International Islamic University  
Islamabad, Pakistan

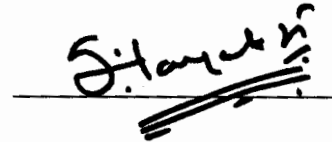
  
\_\_\_\_\_

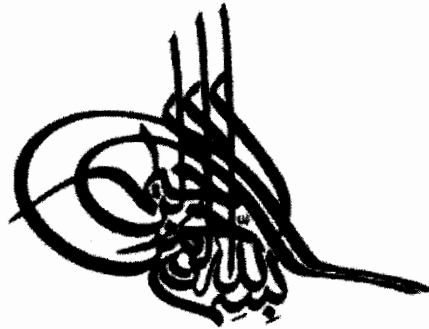
**Supervisors**

Dr. Syed Afaq Hussain  
Associate Professor  
Head (Department of Computer Engineer and Telecommunication)  
International Islamic University  
Islamabad, Pakistan

  
\_\_\_\_\_

Dr. Malik Sikandar Hayat Khoyal  
Associate Professor  
Head (Department of Computer Science)  
International Islamic University  
Islamabad, Pakistan

  
\_\_\_\_\_



*In The Name of*

***ALLAH ALMIGHTY***

*The Most Merciful The Most Beneficent*

**“Lo! In the creation of the heavens and the earth and the alternation of the night and the day, there are surely signs for men of understanding.” (Al-Imran:190-191)**

A dissertation submitted to the Department of Computer Science, Faculty of Applied Sciences, International Islamic University, Islamabad, Pakistan, as a partial fulfilment of the requirements for the award of the degree of

## **MS in Computer Science**

## **DEDICATION**

We dedicate this project to *our Parents* whose prayers and unending efforts enabled us to complete this task, especially to *our Teachers* who helped us a lot in our Project and to the *Loved Ones* who motivated, supported and encouraged us in our entire life”

**Nighat Mir 153-MS/CS/2003**  
**Beenish Aziz 155-MS/CS/2003**

## **DECLARATION**

We hereby declare that this project report, neither as a whole nor as a part thereof has been copied out from any source. It is further declared that we have developed this project and accompanied report entirely on the basis of our personal efforts made under the sincere guidance of our teachers. No portion of the work presented in this report has been submitted in support of any application for any other degree or qualification of this or any other university or institute of learning. If any part of this report is proved to be copied out or found to be reported, we shall stand by the consequences

**Nighat Mir    153-MS/CS/2003**  
**Beenish Aziz    155-MS/CS/2003**

## ACKNOWLEDGEMENT

All the praise of Almighty Allah, The Lord of Creation, the known and unknown universe, Who descended His last Prophet Hazrat Mohammad (SAW), the Benefactor of Humanity; to guide the mankind and enlighten the ones who believe Almighty Allah, Who bestowed us good health, courage and knowledge to carry out and complete our work.

It is unimaginable that an academic effort of this magnitude could successfully come to fruition without the help of others. Expressing gratitude to those whom it is due is a highly regarded Islamic custom based upon the statement of Prophet Muhammad (S.A.W):

***“Whoever does not thank people does not thank Allah.”***

We express our Highest Gratitude to our Kind Supervisors Dr. Syed Afaq Hussain and Dr. M. Sikandar Hayat Khiyal who kept our morale high by his suggestions and appreciation. Their motivation leads us to this success without their sincere and cooperative nature and precious guidance; we could never have been able to complete this task.

We would like to pay special thanks to our class fellows. We will always remember their corporation with us. They helped us in our project whenever we needed.

We would also like to acknowledge the support of our Family members. We would like to admit that we owe all our achievements to our truly, sincere and most loving parents, brothers and sisters, who mean the most to us, and whose prayers are a source of determination for us.

**Nighat Mir 153-MS/CS/2003**  
**Beenish Aziz 155-MS/CS/2003**



## **PROJECT IN BRIEF**

**Project Title:** Secure Data Communication Using Cryptography

**Under Taken By:** Nighat Mir  
Reg. No. 153/CS/MS/2003  
Beenish Aziz  
Reg. No. 155/CS/MS/2003

**Supervised By:** Dr.Syed Afaq Hussain  
Head of Department of Telecommunication  
International Islamic University, Islamabad.  
Dr.Malik Sikandar Khiyal  
Head of Department of CS&IT  
International Islamic University, Islamabad.

**Starting Date:** September, 2004.

**End Date:** February, 2006.

**Tools used:** Matlab 7.

**Hardware Platform:** Intel based

## **ABSTRACT**

To communicate data in a reliable and secure way has been the area for quite sometime and various solutions have been proposed. Two techniques namely Cryptography and Steganography are generally used for this purpose. The image steganographic techniques are evaluated on the basis of their information hiding capacity, perceptibility and robustness against attacks. The proposed system has added flexibility to the system with an additional layer of security by using the splitter, which makes the system impregnable and the additional embedding of pseudo data helps in recognizing attacks and in recovery of data after attacks. The system can detect common image processing attacks like resizing, rotation, cropping and compression.

---

## Table of Contents

1	Introduction .....	1
1.1	Two Main Frameworks of Cryptography .....	3
1.2	Comparison of Symmetric and Asymmetric Algorithm.....	5
2	Literature Review .....	6
2.1	Study of Cryptographic Algorithm.....	6
2.2	Comparison between RSA and DES .....	11
2.3	Comparison OF AES and XOR Algorithm .....	12
2.4	Why Steganography .....	13
2.5	Comparison of Cryptography & Steganography .....	15
2.6	Comparison of Different Steganographic Types .....	16
2.7	Comparison of DCT and DWT .....	18
2.8	Combination of Setganography and Cryptography Method.....	18
2.9	Problem Definition .....	19
2.10	Proposed Solution.....	20
2.11	Goals and Objectives .....	21
3	Transformations and Algorithm Specification .....	24
3.1	Transformations.....	24
3.1.1	Discrete Cosine Transformation (DCT) .....	24
3.1.2	Discrete Wavelet Transformation (DWT).....	26
3.1.3	Fourier Transformation.....	28
3.1.4	Similarities between Fourier and Wavelet Transforms .....	29
3.1.4	Dissimilarities between Fourier and Wavelet Transforms .....	30
3.2	Algorithm.....	31
3.2.1	Advanced Encryption Standard (AES).....	31
3.2.2	Algorithm Specification .....	32
4	Proposed Model.....	37
4.1	Proposed System.....	37
4.1.2	Architecture Design.....	37
4.1.3	ESIS Block Diagram .....	38
	.....	38
4.2	ESIS Working.....	39
4.3	Embedding Algorithm .....	40
4.4	Detection Algorithm .....	40
4.5	Discrete Wavelet Transformation ( DWT).....	46
4.6	Discrete Fourier Transformation (DFT) .....	49
5	Experimental Results.....	52
5.1	Robustness of an Image with DCT Applied.....	53
5.2	Robustness of an Image with DWT Applied.....	53
5.3	JPEG Compression Attack .....	57
6	Implementation.....	58
6.1	AES (Advanced Encryption Standard).....	63
6.2	Splitter .....	65
6.3	Embedding Algorithm .....	66
6.4	Detection Algorithm .....	72
	References .....	84

# **Chapter-1**

# **INTRODUCTION**

**1****Introduction**

Information is the life blood of modern business environment. Information and Communications Technologies play an ever increasing role in delivering timely information when and where required. Data communications is basically how computers communicate with other computers across a network, using the internet TCP/IP protocols and key data communications applications. Data communication is an exchange of data between two parties and it focuses on transmission in a reliable and efficient manner. To achieve reliability secure data communication over the network has been an area of research for quite sometime and various solutions have been proposed. Four Basic Services of secure data communication are [1]:

- **Confidentiality (Secrecy):** The intruder cannot read the encrypted message from cipher text. The confidentiality can be achieved by digital signatures also.
- **Authentication/ Digital Signature:** It should be possible for the receiver of a message to ascertain its origin; an intruder should not be allowed to masquerade as someone else.
- **Integrity:** It should be possible for the receiver of a message to verify that message has not been modified in transit, an intruder cannot substitute a false message for a legitimate one.
- **Non repudiation:** A sender should not be able to falsely deny later that he sent a message.

Authentication systems typically involve parties who play three roles in the authentication process. A "presenter" presents credentials issued by a third-party "issuer" to a "verifier" who wishes to determine the veracity of those credentials. In some cases, one party may play two roles. For example, the verifier and issuer roles are often combined. The issuer usually uses a separate system to perform an initial authentication of prospective credential holders prior to issuing credentials, to ensure that they are

issued only to legitimate parties. Here are examples of authentication/ digital signature systems.

- **SHA-1** (Secure Hash Algorithm version 1) was developed by NIST in 1993. It is based on algorithms developed by Ronald Rivest (MD4 (message digest) and MD5).
- **DSA** (The Digital Signature Algorithm) was proposed by the NIST in 1991 and became a standard in 1993. It is the first digital signature to be accepted by the U.S. government.
- **ECDSA** (Elliptic Curve DSA) is an elliptic curve analogy of DSA.
- **Kerberos** was developed at MIT in the 1980s as a user identification, authentication and authorization system.
- **OPS** (Open Profiling Standard) is backed by a few prominent Internet companies such as VeriSign and Netscape as well as privacy activists such as the Electronic Frontier Foundation

Access control governs a user's ability to make a connection to a particular network, computer or application, or to a specific kind of data traffic. Access from external systems is generally implemented using network firewalls. A firewall is a group of systems or a mechanism that enforces a security policy to protect an internal (trusted) network from an external (entrusted) one. The firewall determines which inside services can be accessed from the outside, which outsiders are permitted access to the permitted inside services, and which outside services can be accessed by insiders. For a firewall to be effective, all traffic to and from the Internet must pass through the firewall, where it can be inspected. The firewall must permit only authorized traffic to pass, and the firewall itself must be immune to penetration. A firewall system cannot offer any protection once an attacker has gotten through or around it.

**Threats:** - A computer network can be attacked in a number of ways with different degrees of damage. These attacks can take several forms:

- **Denial of service.** The attacker disrupts the smooth flow of information by crashing or overloading a critical device such as a server, router or firewall. This is an attack on the availability of information.
- **Theft of information.** The attacker acquires information that is proprietary to the organization. This can be done by eavesdropping, sniffing, by masquerading as an authorized entity, or by a brute-force attack such as the use of a computer program that guesses passwords. This is an attack on the ownership of information and intellectual property.
- **Corruption of data.** The attacker either destroys or corrupts data stored on disk or corrupts data as it is transmitted across the network. This is an attack on the integrity of information.

Even if both access control and authentication security systems are completely effective but any enterprise can still be at risk when data communications travel over a third-party network such as the Internet. So cryptography is used to protect against eavesdropping. It is the study of methods of sending messages in disguised form so that only the intended recipients can remove the disguise and read the message.

Cryptography is used for communicating secrets over non secure channels where sender maintains the secrecy by transferring data (plaintext) into an unintelligible form (cipher text) through a process of encryption and the receiver recovers the original plaintext data by using the process of decryption. Cryptography controls both the encryption and decryption processes.

Digital cryptography is a set of protocols, algorithms, and techniques providing security for digital information by encoding and decoding the information using a specific key.

## 1.1 Two Main Frameworks of Cryptography

Encryption techniques use complicated algorithms to transform digital information (messages, images, or signals) from plaintext to cipher text. Every time the encryption key is changed, the cipher text will be different, although the algorithm stays the same.

The relationship between the encryption and decryption keys classifies the encryption methods in one of two distinct categories: symmetric and asymmetric encryption. ***Symmetric encryption*** is a traditional way of encrypting (also called Private Key Encryption), where the encryption and decryption keys are the same. This method is faster and easier to implement than asymmetric encryption, since the sender and the receiver use the same key to transmit and receive information. Also, the key sizes are smaller in symmetric encryption compared to asymmetric algorithms. However, the private exchanging of the key between the sender and receiver is challenging. Both parties have to agree and trust on a communication medium. Examples of some common Private Key Encryption algorithms are:

- IDEA (International Data Encryption Algorithm)
- FEAL (Fast Data Encipherment Algorithm)
- DES (Data Encryption Standard)
- Triple DES
- AES
- RC5
- RC6
- LOKI

Symmetric encryption is also divided into two groups: block and stream ciphers. Block ciphers work on blocks of data and are commonly used to encrypt the documents. ***Asymmetric encryption*** is a method where the encryption and decryption keys are different. These systems are also called Public Key Encryption Systems, since the encryption key does not have to be a secret. The sender can publish the encryption key and anyone can encrypt messages going to the specific user. However, only the receiver can decrypt the message, since the decryption key cannot be generated with the knowledge of the encryption key. This method is slower and requires more computational power than symmetric encryption. Examples of some common Public Key Encryption algorithms are [2]:

- RSA (Rivest-Shamir-Adelman)



- Diffie-Hellman

### ***1.2 Comparison of Symmetric and Asymmetric Algorithm***

Symmetric encryption enables mutual authentication at no extra cost or overhead. Asymmetric encryption requires the user to have the public verification key of the server. In a mutual authentication protocol, both the server and user verify the identity of the other.

***Security and Efficiency:*** - Both symmetric and asymmetric protocols provide equal security. Both protocols provide a high level of security that is as strong as the algorithm and keys used. Symmetric key algorithms are far more efficient than asymmetric. However, since symmetric protocols involve computing only a single key signature, the differences in efficiency are less significant. Secure authentication requires secure hardware, and it is in this context that we find the main advantage of symmetric over asymmetric protocols. Since symmetric key algorithms are much simpler than asymmetric, the hardware required is significantly less expensive [2].

**Chapter-2**  
**LITERATURE REVIEW**

## 2 Literature Review

### 2.1 Study of Cryptographic Algorithm

“Gigabits per Second Implementation of the IDEA Cryptographic Algorithm” by Antti Hamalainen, Matti Tommiska and Jorna Skytta, 760-769, EPL 2002, describes the algorithm as **IDEA (International Data Encryption Algorithm)** that it is one of the strongest secret-key block ciphers. The algorithm consists of three arithmetic operations: XOR, Addition modulo  $2^{16}$  and Multiplication modulo  $2^{16} + 1$ . The algorithm processes data in 16-bit sub blocks and can be fully pipelined. IDEA encrypts 64-bit plaintext blocks into 64-bit cipher text blocks using a 128-bit input key  $K$ . The algorithm consists of eight identical rounds followed by an output transformation. Each round uses six 16-bit sub keys to transform a 64-bit input into an output of four 16-bit blocks, which are then input to the next round. IDEA is considered highly secure, and it has resisted all forms of attack. The security of IDEA appears bounded only by the weaknesses arising from the relatively small (compared to its key length) block length of 64 bits [3].

#### *Advantages of IDEA*

The following are the advantages

- Minimal key setup time.
- Many of the round operations can be combined into the same cycle to help minimize a pipelined implementation.
- All the operations are extremely fast in hardware.
- The entire algorithm is fast as far as encryption goes.
- The same encryption hardware can be mostly reused for the decryption.

### ***Disadvantages of IDEA***

The following are the disadvantages

- Fixed block size smaller than a cache block
- Requires many registers to store intermediates between a rounds stages.
- The IDEA algorithm is patented and must be licensed for commercial use. [3].

“Data Encryption Standard”, a technical report from *wikipedia* on encyclopedia describes ***The Fast Data Encipherment Algorithm (FEAL)*** as an alternative to DES. The original cipher (called FEAL-4) was a four-round cryptosystem with a 64-bit block size and a 64-bit key size and it was designed to give high performance in software. A number of attacks have been developed against it, which has led to revised versions with more rounds and larger keys. In the wake of these numerous attacks, FEAL and its derivatives should be considered insecure.

“Data Encryption Standard”, a technical report from *wikipedia* on encyclopedia describes the ***The DES (Data Encryption Standard)*** algorithm as the most widely used encryption algorithm in the world. For many years, and among many people, "secret code making" and DES have been synonymous. DES operates on 64-bit blocks of data, using a 56-bit key. DES has two steps to process, first it process the key and secondly it processes a 64-bit data block

DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small; DES keys have been broken in less than 24 hours. There are also some analytical results which demonstrate theoretical weaknesses in the cipher, although they are infeasible to mount in practice. The algorithm is believed to be practically secure in the form of Triple DES although there are theoretical attacks. In recent years, the cipher has been superseded by the Advances Encryption Standard (AES) [4].

### ***Security of DES:***

Unfortunately, with advances in the field of cryptanalysis and the huge increase in available computing power, DES is no longer considered to be very secure. There are algorithms that can be used to reduce the number of keys that need to be checked, but even using a straightforward brute-force attack and just trying every single possible key there are computers that can crack DES in a matter of minutes.

If a time limit of 2 hours to crack a DES encrypted file is set, then we have to check all possible keys ( $2^{56}$ ) in two hours, which is roughly 5 trillion keys per second. Whilst this may seem like a huge number, consider that a \$10 Application-Specific Integrated Circuits (ASICs) chip can test 200 million keys per second, and many of these can be paralleled together [2]. It is suggested that a \$10 million investment in ASICs would allow a computer to be built that would be capable of breaking a DES encrypted message in 6 minutes [2].

It is the conclusion of this author that DES can no longer be considered a sufficiently secure algorithm. If a DES-encrypted message can be broken in minutes by supercomputers today, then the rapidly increasing power of computers means that it will be a trivial matter to break DES encryption in the future (when a message encrypted today may still need to be secure).

“An Introduction to modern Cryptography”, Oli Cooper, 22nd February 2001 describes LOKI as a cipher designed as a result of the detailed analysis of existing block ciphers, particularly of the DES. Overall design is based on design principles developed from a detailed analysis of the DES which includes permutation P and its role in growth of dependence of cipher bits on input bits and permutation PC2 and key rotation schedule and its role in growth of dependence of cipher bits on key bits. Aim was for structure to be as simple as possible without compromising security

“An Introduction to modern Cryptography”, Oli Cooper, 22nd February 2001 describes RSA as a public-key cryptosystem meaning that different keys are used to encode and decode message. Since the encoding key is of no help in decoding, it can be made public at no risk to security. The security of RSA is based on the difference in the computational

complexity of primly testing. Encoding is fast because it relies on primly testing to construct the keys, while the hardness of decryption, follows from that of factorizing.

### ***Security of RSA***

“An Introduction to modern Cryptography”, Oli Cooper, 22nd February 2001 describes RSA as for encrypting or decrypting, signing or verifying, is essentially a modular exponentiation, which can be performed by a series of modular multiplications. In practical applications, it is common to choose a small public exponent for the public key. There are some restrictions on the prime factors of the modulus when the public exponent is fixed. This makes encryption faster than decryption and verification faster than signing. "Fast multiplication" techniques, such as FFT-based methods, require asymptotically fewer steps, though in practice they are not as common due to their great software complexity and the fact that they may actually be slower for typical key sizes [5].

“*RC6 and the AES*” by M. J. B. Robshaw, 16d Stowe Rd, London, W12 8BN, UK January 9, 2001 describes **RC5** as a fast block cipher designed by Ronald Rivest for RSA Data Security in 1994. It is a parameterized algorithm with a variable block size, a variable key size, and a variable number of rounds. Allowable choices for the block size are 32 bits (for experimentation and evaluation purposes only), 64 bits (for use a drop-in replacement for DES), and 128 bits. The number of rounds can range from 0 to 255, while the key can range from 0 bits to 2040 bits in size. Such built-in variability provides flexibility at all levels of security and efficiency.

There are three routines in RC5: key expansion, encryption, and decryption. In the key-expansion routine, the user-provided secret key is expanded to fill a key table whose size depends on the number of rounds. The key table is then used in both encryption and decryption. The encryption routine consists of three primitive operations: integer addition, bitwise XOR, and variable rotation. The exceptional simplicity of RC5 makes it easy to implement and analyze. Indeed, like the RSA system, the encryption steps of RC5 can be written on the "back of an envelope".

The heavy use of data-dependent rotations and the mixture of different operations provide the security of RC5. In particular, the use of data-dependent rotations helps defeat differential and linear cryptanalysis .

**RC6** is a block cipher based on RC5 and designed by Rivest, Sidney, and Yin for RSA Security. Like RC5, RC6 is a parameterized algorithm where the block size, the key size, and the number of rounds are variable; again, the upper limit on the key size is 2040 bits. The main goal for the inventors has been to meet the requirements of the AES.

There are two main new features in RC6 compared with RC5: the inclusion of integer multiplication and the use of four  $b/4$ -bit working registers instead of two  $b/2$ -bit registers as in RC5 ( $b$  is the block size). Integer multiplication is used to increase the diffusion achieved per round so that fewer rounds are needed and the speed of the cipher can be increased. The reason for using four working registers instead of two is technical rather than theoretical. Namely, the default block size of the AES is 128 bits; while RC5 deals with 64-bit operations when using this block size, 32-bit operations are preferable given the intended architecture of the AES [6].

*“Provably Authenticated Group Diffie-Hellman Key Exchange”* by Emmanuel Bresson, Olivier Chevassul, David Pointcheval, Jean Jacques Quisquater, CCS 01, USA, 2001 describes **Group Diffie-Hellman schemes** as Authenticated Key Exchange are designed to provide a pool of players communicating over an open network with a shared secret key which may later be used to achieve some cryptographic goals like multicast message confidentiality or multicast data integrity. Secure virtual conferencing involving up to a hundred participants is an example of such a multicast scenario. In this scenario the group membership is static and known in advance: at startup the participants would like to engage in a conversation at the end of which they have established a session key. For this scenario group Diffie-Hellman schemes are attractive alternatives to methods that establish a session key between every pair of players in the multicast group or rely on a centralized key distribution center [7].

"AES Algorithm (FIPS 197) Advanced Encryption Standard" by ADI-AMD, vocal technologies 2003 describes *The Advanced Encryption Standard (AES Algorithm)* as a computer security standard that became effective on May 26, 2002 by NIST to replace DES. The cryptography scheme is a symmetric block cipher that encrypts and decrypts 128-bit blocks of data. Lengths of 128, 192, and 256 bits are standard key lengths used by AES Algorithm.

The algorithm consists of four stages that make up a round which is iterated 10 times for a 128-bit length key, 12 times for a 192-bit key, and 14 times for a 256-bit key. The first stage "SubBytes" transformation is a non-linear byte substitution for each byte of the block. The second stage "ShiftRows" transformation cyclically shifts (permutes) the bytes within the block. The third stage "MixColumns" transformation groups 4-bytes together forming 4-term polynomials and multiplies the polynomials with a fixed polynomial mod  $(x^4+1)$ . The fourth stage "AddRoundKey" transformation adds the round key with the block of data [8].

## 2.2 Comparison between RSA and DES

"AES Algorithm (FIPS 197) Advanced Encryption Standard" by ADI-AMD, vocal technologies 2003 describes the comparison that on a 90 MHz Pentium, RSA Data Security's cryptographic toolkit BSAFE 3.0 has a throughput for private-key operations of 21.6 K bits per second with a 512-bit modulus and 7.4 K bits per second with a 1024-bit modulus. The fastest RSA hardware has a throughput greater than 300 K bits per second with a 512-bit modulus, implying that it performs over 500 RSA private-key operations per second. In pure algorithmic terms, RSA is a strong. It has the ability to support much longer key lengths than DES etc

By comparison, DES is much faster than RSA. In software, DES is generally at least 100 times as fast as RSA. In hardware, DES is between 1,000 and 10,000 times as fast, depending on the implementation. Implementations of RSA will probably narrow the gap a bit in coming years, as there are growing commercial markets, but DES will get faster as well.



There are a few possible interpretations of "breaking RSA." The most damaging would be for an attacker to discover the private key corresponding to a given public key; this would enable the attacker both to read all messages encrypted with the public key and to forge signatures. The obvious way to do this attack is to factor the public modulus,  $n$ , into its two prime factors,  $p$  and  $q$ . From  $p$ ,  $q$ , and  $e$ , the public exponent, the attacker can easily get  $d$ , the private exponent. The hard part is factoring  $n$ ; the security of RSA depends on factoring being difficult. In fact, the task of recovering the private key is equivalent to the task of factoring the modulus: use  $d$  to factor  $n$ , as well as use the factorization of  $n$  to find  $d$ . It should be noted that hardware improvements alone will not weaken RSA, as long as appropriate key lengths are used; in fact, hardware improvements should increase the security of RSA [8].

### **2.3 Comparison OF AES and XOR Algorithm**

*"A Light-Weight Encrypting For Real Time Video Transmission"*, Salah Aly, February 5, 2003 describes the comparisons that nowadays in the digital world, multimedia security is becoming more important with the continuous increase in the use of digital communications on the Internet. In addition, special and reliable security in storage and transmission of digital images and videos is needed in many digital applications, such as video conferencing and medical imaging systems, etc. So in the paper presented by Salah Aly in 2003 AES algorithm is used for securing real time video transmission with little processing overhead over the Internet. They adapt AES and XOR algorithms to be used with JPEG, H261, CellB, and MPEG video encoders and decoders. The performance of AES encryption frames is to display the received frames on time. The encryption delays overhead using AES is less than the overhead using XOR algorithm. In addition, AES can achieve satisfactory encryption results with little overhead. So they conclude that using AES is a feasible solution to secure real time video transmissions. So partial or full AES encryption algorithm can be used to secure real time video applications including Video On-Demand (VOD), pay-TV, and video conferencing. This usage depends also on how much security is needed [9].

### ***Performance comparison of AES Candidates***

*In the paper "Performance comparison of AES Candidates" by Bruce Schneier in February 1, 1999, the performances of the leading AES candidates are compared on a variety of common platforms: 32-bit CPUs, 64-bit CPUs, cheap 8-bit smart-card CPUs, and dedicated hardware. First some general observations are made on the performance issues for each of the platforms, and then compared the various AES candidates, and finally look at the specific issues for each of the candidates. Nine algorithms CAST-256, Crypton, DFC, E2, Frog, HPC, Mars, RC6, and Serpent are completely independent of key length. Two algorithms Loki97 and Twofish- encrypt and decrypt independent of key length, but take different times to set up different size keys. Twofish takes longer to set up a longer key, while Loki97 actually takes less time to set up a longer key than it does to set up a shorter key. Four algorithms DEAL, Magenta, Rijndael, and SAFER - encrypt and decrypt at different speeds, depending on key length. This paper concentrated on key setup and encryption for 128-bit keys. As encryption speed is more important than key setup speed. The DEAL and Magenta encrypt 33% slower using 256-bit keys. Rijndael encrypts 20% slower for 192-bit keys and 40% slower for 256-bit keys. SAFER has the largest performance degradation for large keys; it is 50% slower for 192-bit keys and 100% slower for 256-bit keys. The conclusion that is extracted in this paper is that it is very difficult to compare cipher designs for efficiency, and even more difficult to design ciphers that is efficient across all platforms and all uses. It's far easier to design a cipher to be efficient on one platform, and then let the other platforms come out as they may [10].*

## **2.4 Why Steganography**

The use of images is becoming more persistent in modern culture. Visual communication through an image helps to make a subject clear and attractive. Digital images, audio and video files are composed of collections of bits that are translated by their related software into images, sounds or videos. These files contain unused areas or data that is insignificant and can be overwritten. By using steganographic techniques, we can hide our text in an image. **Steganography** literally means covered writing. Its goal is to hide the fact that communication is taking place. This is often achieved by using a (rather large)

cover file and embedding the (rather short) secret message into this file. The result is an innocuous looking file (the stego file) that contains the secret message. There are three different aspects in information-hiding systems contend with each other: capacity, security and robustness. Capacity refers to the amount of information that can be hidden in the cover medium, security to an eavesdropper's inability to detect hidden information and robustness to the amount of modification the stego medium can withstand before an adversary can destroy the hidden information.

### **Steganography Principles**

- The difference of the cover image and stego-image should be visually unnoticeable. The embedding itself should draw no extra attention to the stego-image so that no hackers would try to extract the hidden message.
- The message hiding method should be reliable. It is impossible for someone to extract the hidden message if does not have a special extracting method and a proper secret key.
- The maximum length of the secret message that can be hidden should be as long as possible.
- An IS scheme should be able to live up to some lossy image compression and image processing done to the stego image. The message recovery rate should be high after these image manipulations.

### **How to Design IS**

#### **LSB Image Steganography**

- Message is hidden in the least significant bits(LSB) of the pixels of the cover image.
- The image quality of the stego image of LSB scheme is very close to the original
- It cannot survive under some image processing, such as the JPEG lossy compression [11].

#### **Wavelet Transformation bases IS**

- This Scheme embeds the secrets message by modifying the wavelet coefficients of the original cover image.
- The modification is based on the patterns of the wavelet coefficients
- The experimental results show that the difference between the original image and the embedding image is visually unnoticeable
- The Scheme can still work well under JPEG compression [12].

## **2.5 Comparison of Cryptography & Steganography**

Steganography and encryption are both used to ensure data confidentiality. However the main difference between them is that with encryption anybody can see that both parties are communicating in secret. Steganography hides the existence of a secret message and in the best case nobody can see that both parties are communicating in secret.

So it is true that cryptography is about protecting the content of messages (their meaning) whereas steganography is about concealing their very existence. But it is a fact that Steganography has its place in security and it can not replace cryptography but supplement it. Hiding a message with steganography methods reduces the chance of a message being detected. However, if that message is also encrypted, if discovered, it must also be cracked.

It is often thought that communications may be secured by encrypting the traffic, but this has rarely been adequate in practice. Some writers concentrated on methods for hiding messages rather than for enciphering them; and although modern cryptographic techniques are still preferred hiding over ciphering because it arouses less suspicion. A message in cipher text may arouse suspicion while an invisible message will not. In contrast to cryptography, where the "enemy" is allowed to detect, intercept and modify messages without being able to violate certain security premises guaranteed by a cryptosystem, the goal of steganography is to hide messages inside other "harmless" messages in a way that does not allow any "enemy" to even detect that there is a second secret message present. Steganography, on the other hand, strives for high security and capacity, which often entails that the hidden information is fragile.

### **Advantages of Steganography**

- The existence of hidden message is imperceptible, there is no effort to decode/ extract the message.
- Specially useful in passing secret messages over the internet
- All forms of digital data (Still image, Audio, Vide, Text document, multimedia documents) can be used information hiding.

## 2.6 Comparison of Different Steganographic Types

In the paper “Disguising Text Cryptography using Image Cryptography” by Potdar Vidyasagar Image Steganography, Audio steganography and Video Steganography are discussed. There are a variety of methods for image steganography. These are

1. Replacing Least Significant Bit
2. Replacing Moderate Significant Bit
3. Transformation Domain Techniques
  - a. Discrete cosine Transformation [13].
  - b. Discrete Wavelet Transformation

**Replacing Least Significant Bit:** - A simple scheme proposed by Chang (2002), is to place the embedding data at the least significant bit (LSB) of each pixel in the cover image. Altering LSB doesn't change the quality of image to human perception but this scheme is sensitive a variety of image processing attacks like compression, cropping etc.

**Replacing Moderate Significant Bit:** Chan and Chang (2001) showed how to use the moderate significant bits of each pixel in the cover image to embed the secret message. This method improves sensitivity to modification, but it degrades the quality of stego-image.

**Transformation Domain Techniques:** Other familiar data hiding techniques use the transformation domain of digital media to hide information. Functions such as the discrete cosine transform (DCT) and the discrete wavelet transform (DWT) are widely applied. These methods hide the messages in the significant areas of the cover image which makes them robust against compression, cropping and other image processing attacks. Although image steganographic techniques can be used to disguise the existence of cryptography, these techniques themselves are vulnerable to detection, hence the existence of cryptographic communication can be re-veiled.

## Audio Steganography

Embedding external data in an audio signal can be done in the time-domain as well as in the spectral-domain. Palet.al (2002) compares these different audio steganographic methods based on the method's embedding capacity and robustness.

1. **Low Bit Encoding:** This method has high embedding capacity (41,000 bps), but it is least robust. This method replaces the less significant bits (LSB) of an audio signal. It exploits the absolute threshold of hearing but is susceptible to attacks.
2. **Spread Spectrum:** This method is highly robust but has a negligible embedding capacity (4 bps). This method encodes data in a "wider than required" frequency band to reduce perceptual distortions and improve upon robustness but lacks sufficient embedding capacity.
3. **Perceptual Masking:** This technique has the highest embedding capacity (450,000 bps) but again it's least robust. This method stores data in perceptually insignificant regions of the spectrum. Other techniques like echo hiding, phase coding, and cepstral hiding have medium embedding capacity and robustness. Even audio steganographic techniques are susceptible to detection and cannot disguise cryptographic communication [14].

## Video Steganography

Video steganography is not much different from image steganography. Actually we can say video steganography is a derivative of image steganography, this is because video is made up of a series of images that are transmitted. So whatever techniques (and attacks) that can be applied to images very well apply for videos.

1. **Discrete Cosine Transformation:** Many digital video conferencing systems are mainly based on two-dimensional discrete cosine transformation [14]. Although DCT video encoding is efficient it introduces undesirable blocking artifact at low bit rates .

- 2. Wavelet Coding:** Other approaches like wavelet [15], introduce ringing or rippling artifacts, which become more troublesome in the surrounding area of the image edges.

### 2.7 Comparison of DCT and DWT

In the paper "Comparison of Wavelet and Cosine Basis for Representation of Arbitrarily Shaped Image Segments" by Surendra Ranganath in 1998 describes an algorithm by which the given image segment is successively approximated using 2-D shape-independent basis functions defined on a rectangle circumscribing the image segment. Discrete cosine and discrete wavelet basis were used in the system respectively, and performance comparison between them was made. Simulation results showed that MP-DWT exhibits high energy compactness, and provides better approximation of the given image segment than MP-DCT for non-homogeneous segments. The disadvantage of MP-DWT is that unusual blocks appear in the extrapolation outside the segment due to the selection of basis with small overlap with the image segment. To overcome this CSMask is defined to select only those basis functions having larger overlap with the image segment, thereby greatly improving the extrapolation result. It is also found in this papers that with CSMask, there is almost no change in the approximation results. The improvement brought about by the use of CSMask has significant beneficial implication for coding the segment [16].

### 2.8 Combination of Setganography and Cryptography Method

In the paper "Security Through Obscurity", Dr.K.Duraiswamy proposed a scheme to enhance the data security is the combination of cryptography and steganography. This Scheme has following steps

1. Apply encryption algorithm. This algorithm encrypts the text with a strong block cipher mechanism by applying the cipher block-chaining mode. It also provides the password protection.
2. The cipher text file is embedded into the stego medium. The stego medium may be any image file compressed with lossless compression. The message is hidden

into the LSBs of the image file. The proposed algorithm handles the carrier file in a much careful way, since a very small change in the stego file, which is noticeable, will reveal the fact that it contains some data.

This scheme provides an extra layer of protection. To get the original message, the junk file has to be decrypted with the encryption algorithm with the correct password. There is also an extra layer of protection. If the algorithm is found out, intruder must supply the correct password to get the correct message. Even if one character is altered in the password the original message can't be obtained [17].

Steganography and encryption are both used to ensure data confidentiality. However the main difference between them is that with encryption anybody can see that both parties are communicating in secret. Steganography hides the existence of a secret message and in the best case nobody can see that both parties are communicating in secret.

## 2.9 Problem Definition

Cryptography is a secrete way for communicating secret messages over non secure channels where sender maintains the secrecy by transferring data into unintelligible form through a process of encryption. This encrypted message can be intercepted by a cryptanalyst who forced the sender to reveal the keys that generated the cipher text so that the original message could be known to him. This Scenario highlights few key issues namely: how to maintain the privacy of a sender and how to provide more confidentiality. Another technique for secure communication is steganography whose purpose is to hide both information and the existence of information in an image. Its purpose is not only to communicate in a hidden manner but also to keep aggressive elements away from being aware of this communication..

- Technique for secure communication is steganography whose purpose is to hide both information and the existence of information in an image
- Its purpose is not only to communicate in a hidden manner but also to keep aggressive elements away from being aware of this communication
- In the paper "Security Through Obscurity" by Dr.K.Duraiswamy proposed scheme is the combination of cryptography and steganography to enhance the security



- Even by the combination of Cryptography and Steganography the message can be hacked
- So there is an open end for the research for adding more security layers or can make it robust.

## 2.10 Proposed Solution

Encrypto Split Image steganography (ESIS) is to produce an efficient and stable platform for communication. The main constraint taken into consideration was the error free secure communication.

- We proposed a scheme of combination of cryptography and steganography with one more security layer which is less susceptible to cryptanalyst and steganalyst
- Encrypto Split Image steganography (ESIS) is to produce an efficient and stable platform for communication.
- The main constraint taken into consideration was the error free secure communication.
- The goal of project is to provide a secure communication by introducing the third security level and detecting the almost all possible image processing attacks

First the sender sends the plain text message to the Encryption Phase which converts it into an encrypted secret message by using Advanced Encryption Standard (AES) and then the Splitter Algorithm splits the message into two halves on the basis of bits.

Two cover images are taken in the Stegosystem Encoder and then by using one of the transformations (DCT, DWT or DFT) these images are transformed. these different/same images are embedded to generate the stego-images, which are named as stego-image1 and stego-image2 and then are transferred on the network.

At the receiving end the inverse procedure is applied to get the original message that first we estimate both stego-images that if there is any attack or not and if find any attack we could recover it. After this the stegosystem Decoder decodes the images into encrypted

splitted data. This splitted data is combined back and then passed to Decryption Phase which performs AES Decryption and the receiver gets the original text message .

### ***Advantages of our System***

We proposed one more security level by developing a splitting algorithm and detection algorithm. This solution has following advantages first, encrypted text can be easily identified because the data is scrambled but encrypted (or scrambled) image would not look anomalous (if the image is not very distinct).

Second, the existence of splitting encrypted text into two portions and embedding them into two different images won't be detected because meaningful text cannot be recovered from the one stego image.

Third, no one would look for textual information hidden (not embedded) in an image. rather everyone might sense steganography and try to steganalyse it, as image is normally used in the context of steganography.

Fourth, the detection algorithm can detect the image processing attacks like resizing, rotation, cropping and compression. This also serves for security purpose. The proposed solution inherits the strong features of cryptography and steganography while omitting their weaknesses.

## **2.11 GOALS AND OBJECTIVES**

- How far the data is resistant to possible attacks like image transformation?
- To introduce the third level of security in our proposed Model
- Images when they are distributed, there may be an unintentional or intentional transformation caused due to file operations like compressions, changing formats etc... Also the watermarked image may be edited due to the user needs like it may be cropped, rotated (geometrical attacks).
- Existing techniques for watermarking are sensitive to several transformations of the image thus lacking robustness.

- Imperceptibility: After embedding data, there should not be any perceptible difference found in the quality of image that is being water marked. The watermark should not be easily noticed by simple visual inspection. There should not be noticeable change between the original and watermarked image.
- Robustness: Images are often attacked by strangers knowingly or unknowingly. Such a technique should be chosen that is robust to attacks caused by image processing software. Attacks to deal with in this thesis are
- Data loss caused by rotation of the illustration image, order of all the pixels is going to change.
- Losing pixel information due to editing of image like removing certain area, over writing certain areas with different information

**Chapter 3**  
**TRANSFORMATIONS**  
**AND**  
**ALGORITHM SPECIFICATION**

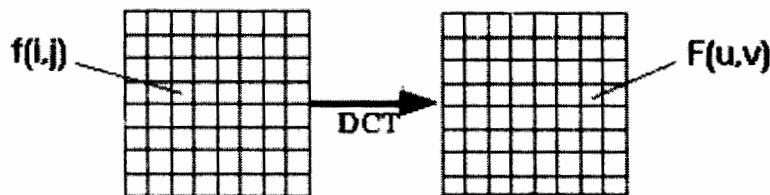
### 3 Transformations and Algorithm Specification

#### 3.1 Transformations

The transformations used in this project are Discrete Cosine transformation (DCT), Discrete Wavelet Transformation (DWT) and discrete Fourier transformation (DFT). The algorithm being used is AES.

##### 3.1.1 Discrete Cosine Transformation (DCT)

The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality). The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain.



##### *DCT Encoding*

The general equation for a 1D ( $N$  data items) DCT is defined by the following equation:

$$F(u) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \Lambda(i) \cdot \cos \left[ \frac{\pi \cdot u}{2 \cdot N} (2i + 1) \right] f(i)$$

and the corresponding *inverse* 1D DCT transform is simple  $F^{-1}(u)$ , i.e.:

where

$$\Lambda(i) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } i = 0 \\ 1 & \text{otherwise} \end{cases}$$

The general equation for a 2D ( $N$  by  $M$  image) DCT is defined by the following equation:

$$F(u, v) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \left(\frac{2}{M}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \Lambda(i) \cdot \Lambda(j) \cdot \cos \left[ \frac{\pi \cdot u}{2 \cdot N} (2i + 1) \right] \cos \left[ \frac{\pi \cdot v}{2 \cdot M} (2j + 1) \right] \cdot f(i, j)$$

and the corresponding *inverse* 2D DCT transform is simple  $F^{-1}(u, v)$ , i.e.:

where

$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } \xi = 0 \\ 1 & \text{otherwise} \end{cases}$$

### **Basic Operation of DCT**

The basic operation of the DCT is as follows:

- The input image is  $N$  by  $M$ ;
- $F(i, j)$  is the intensity of the pixel in row  $i$  and column  $j$ ;
- $F(u, v)$  is the DCT coefficient in row  $k_1$  and column  $k_2$  of the DCT matrix.
- For most images, much of the signal energy lies at low frequencies; these appear in the upper left corner of the DCT.
- Compression is achieved since the lower right values represent higher frequencies, and are often small - small enough to be neglected with little visible distortion.
- The DCT input is an 8 by 8 array of integers. This array contains each pixel's gray scale level; 8 bit pixels have levels from 0 to 255. The output array of DCT coefficients contains integers; these can range from -1024 to 1023.

### 3.1.2 Discrete Wavelet Transformation (DWT)

The fundamental idea behind wavelets is to analyze according to scale. Indeed, some researchers in the wavelet field feel that, by using wavelets, one is adopting a whole new mindset or perspective in processing data. Wavelets are functions that satisfy certain mathematical requirements and are used in representing data or other functions.

Dilations and translations of the "Mother function," or "analyzing wavelet"  $\Phi(x)$  define an orthogonal basis, our wavelet basis:

$$\Phi_{(s,l)}(x) = 2^{-\frac{s}{2}} \Phi(2^{-s}x - l) \quad (3)$$

The variables  $s$  and  $l$  are integers that scale and dilate the mother function  $\Phi(x)$  to generate wavelets, such as a Daubechies wavelet family. The scale index  $s$  indicates the wavelet's width, and the location index  $l$  gives its position. Notice that the mother functions are rescaled, or "dilated" by powers of two, and translated by integers. What makes wavelet bases especially interesting is the self-similarity caused by the scales and dilations. Once we know about the mother functions, we know everything about the basis.

To span our data domain at different resolutions, the analyzing wavelet is used in a scaling equation:

$$W(x) = \sum_{k=0}^{N-2} (-1)^k c_{k+1} \Phi(2x+k) \quad (4)$$

where  $W(x)$  is the scaling function for the mother function  $\Phi(x)$ , and  $c_k$  are the *wavelet coefficients*. The wavelet coefficients must satisfy linear and quadratic constraints of the form

$$\sum_{k=0}^{N-1} c_k = 2 \quad , \quad \sum_{k=0}^{N-1} c_k c_{k+2l} = 2 \delta_{l,0}$$

where  $\delta$  is the delta function and  $l$  is the location index.

One of the most useful features of wavelets is the ease with which a scientist can choose the defining coefficients for a given wavelet system to be adapted for a given problem. In Daubechies' original paper [6], she developed specific families of wavelet systems that were very good for representing polynomial behavior. The Haar wavelet is even simpler, and it is often used for educational purposes.

It is helpful to think of the coefficients  $\{c_0, \dots, c_n\}$  as a filter. The filter or coefficients are placed in a transformation matrix, which is applied to a raw data vector. The coefficients are ordered using two dominant patterns, one that works as a smoothing filter (like a moving average), and one pattern that works to bring out the data's "detail" information. These two orderings of the coefficients are called a *quadrature mirror filter pair* in signal processing parlance. A more detailed description of the transformation matrix can be found elsewhere.

The wavelet transform is actually a subset of a far more versatile transform, the wavelet packet transform.

Wavelet packets are particular linear combinations of wavelets. They form bases which retain many of the orthogonality, smoothness, and localization properties of their parent wavelets. The coefficients in the linear combinations are computed by a recursive algorithm making each newly computed wavelet packet coefficient sequence the root of its own analysis tree.

### **Wavelet Applications**

1. Computer and human Vision
2. FBI Fingerprint Compression
3. Denoising Noisy Data
4. Detecting Self-Similarity in a Time Series
5. Musical Tones



### 3.1.3 Fourier Transformation

Fourier's representation of functions as a superposition of sines and cosines has become ubiquitous for both the analytic and numerical solution of differential equations and for the analysis and treatment of communication signals. Fourier and wavelet analysis have some very strong links.

The Fourier transform's utility lies in its ability to analyze a signal in the time domain for its frequency content. The transform works by first translating a function in the time domain into a function in the frequency domain. The signal can then be analyzed for its frequency content because the Fourier coefficients of the transformed function represent the contribution of each sine and cosine function at each frequency. An inverse Fourier transform does just what you'd expect, transform data from the frequency domain into the time domain.

#### **Discrete Fourier Transforms**

The discrete Fourier transform (DFT) estimates the Fourier transform of a function from a finite number of its sampled points. The sampled points are supposed to be typical of what the signal looks like at all other times.

The DFT has symmetry properties almost exactly the same as the continuous Fourier transform. In addition, the formula for the inverse discrete Fourier transform is easily calculated using the one for the discrete Fourier transform because the two formulas are almost identical.

A discrete transform is a transform whose input and output values are discrete samples, making it convenient for computer manipulation. There are two principal reasons for using this form of the transform:

The input and output of the DFT are both discrete, which makes it convenient for computer manipulations.

There is a fast algorithm for computing the DFT known as the fast Fourier transform (FFT).

The DFT is usually defined for a discrete function  $f(m, n)$  that is nonzero only over the finite region  $0 \leq m \leq M - 1$  and  $0 \leq n \leq N - 1$ . The two-dimensional  $M$ -by- $N$  DFT and inverse  $M$ -by- $N$  DFT relationships are given by

$$F(p, q) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) e^{-j(2\pi/M)pm} e^{-j(2\pi/N)qn} \quad \begin{array}{l} p = 0, 1, \dots, M-1 \\ q = 0, 1, \dots, N-1 \end{array}$$

$$f(m, n) = \frac{1}{MN} \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F(p, q) e^{j(2\pi/M)pm} e^{j(2\pi/N)qn} \quad \begin{array}{l} m = 0, 1, \dots, M-1 \\ n = 0, 1, \dots, N-1 \end{array}$$

The values  $F(p, q)$  are the DFT coefficients of  $f(m, n)$ . The zero-frequency coefficient,  $F(0, 0)$ , is often called the "DC component." DC is an electrical engineering term that stands for direct current. (Note that matrix indices in MATLAB always start at 1 rather than 0; therefore, the matrix elements  $f(1,1)$  and  $F(1,1)$  correspond to the mathematical quantities  $f(0, 0)$  and  $F(0, 0)$ , respectively.)

### 3.1.4 Similarities between Fourier and Wavelet Transforms

The fast Fourier transform (FFT) and the discrete wavelet transform (DWT) are both linear operations that generate a data structure that contains  $\log_2 n$  segments of various lengths, usually filling and transforming it into a different data vector of length  $2^n$ .

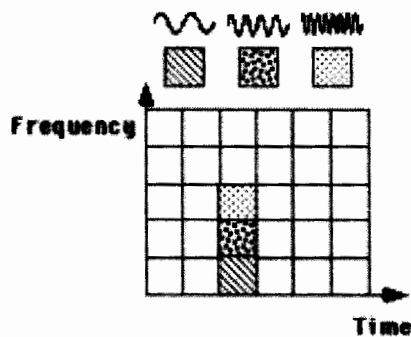
The mathematical properties of the matrices involved in the transforms are similar as well. The inverse transform matrix for both the FFT and the DWT is the transpose of the original. As a result, both transforms can be viewed as a rotation in function space to a different domain. For the FFT, this new domain contains basis functions that are sines and cosines. For the wavelet transform, this new domain contains more complicated basis functions called wavelets, mother wavelets, or analyzing wavelets.

Both transforms have another similarity. The basis functions are localized in frequency, making mathematical tools such as power spectra (how much power is contained in a frequency interval) and scalegrams (to be defined later) useful at picking out frequencies and calculating power distributions.

### 3.1.4 Dissimilarities between Fourier and Wavelet Transforms

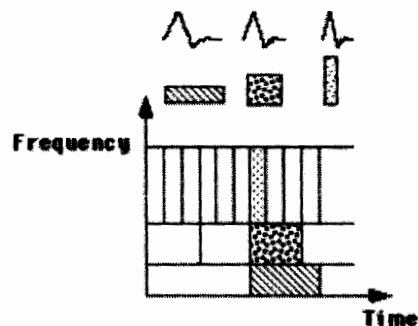
The most interesting dissimilarity between these two kinds of transforms is that individual wavelet functions are *localized in space*. Fourier sine and cosine functions are not. This localization feature, along with wavelets' localization of frequency, makes many functions and operators using wavelets "sparse" when transformed into the wavelet domain. This sparseness, in turn, results in a number of useful applications such as data compression, detecting features in images, and removing noise from time series.

One way to see the time-frequency resolution differences between the Fourier transform and the wavelet transform is to look at the basis function coverage of the time-frequency plane. Figure 1 shows a windowed Fourier transform, where the window is simply a square wave. The square wave window truncates the sine or cosine function to fit a window of a particular width. Because a single window is used for all frequencies in the WFT, the resolution of the analysis is the same at all locations in the time-frequency plane.



**Fig. 1. Fourier basis functions, time-frequency tiles, and coverage of the time-frequency plane.**

An advantage of wavelet transforms is that the windows *vary*. In order to isolate signal discontinuities, one would like to have some very short basis functions. At the same time, in order to obtain detailed frequency analysis, one would like to have some very long basis functions. A way to achieve this is to have short high-frequency basis functions and long low-frequency ones. This happy medium is exactly what you get with wavelet transforms. Figure 2 shows the coverage in the time-frequency plane with one wavelet function, the Daubechies wavelet.



**Fig. 2. Daubechies wavelet basis functions, time-frequency tiles, and coverage of the time-frequency plane.**

One thing to remember is that wavelet transforms do not have a single set of basis functions like the Fourier transform, which utilizes just the sine and cosine functions. Instead, wavelet transforms have an infinite set of possible basis functions. Thus wavelet analysis provides immediate access to information that can be obscured by other time-frequency methods such as Fourier analysis.

## 3.2 Algorithm

In our implementation the security consideration we are using the symmetric key cryptography algorithm, Advanced Encryption Standard (AES).

### 3.2.1 Advanced Encryption Standard (AES)

The AES is the Advanced Encryption Standard. The AES was issued as FIPS PUB 197 by NIST, successor to DES. In January 1997 the AES initiative was announced and in September 1997 public was invited to propose suitable block ciphers as candidates for the AES. The AES algorithm was selected in October 2001 and the

standard was published in November 2002. NIST's intent was to have a cipher that will remain secure well into the next century.

For our project we have implemented this algorithm, and we are specifying its specification here.

### **3.2.2 Algorithm Specification**

AES supports key sizes of 128 bits, 192 bits, and 256 bits, in contrast to the 56-bit keys offered by DES. The AES algorithm resulted from a multi-year evaluation process led by NIST with submissions and review by an international community of cryptography experts. The Rijndael algorithm, invented by Joan Daemen and Vincent Rijmen, was selected as the standard.

Over time, many implementations are expected to upgrade to AES, both because it offers a 128-bit key size, and because it is a federal standard.

An overview of phases of the AES algorithm is given in the figure 2.1. The figure shows different phases of the AES algorithm. It starts with an initial round followed by a number of standard rounds and ends with the final round. Four different operations are necessary to compute these rounds and a key schedule.

### **3.2.3 Encryption Rounds**

The AES algorithm is such that each bit is dependent on all bits from two rounds, e.g. full diffusion is provided. The number of rounds that must be run is dependent on the key length.

In the description of the AES cipher the intermediate cipher result will be called the State. Matrix notations can be used to represent the state.

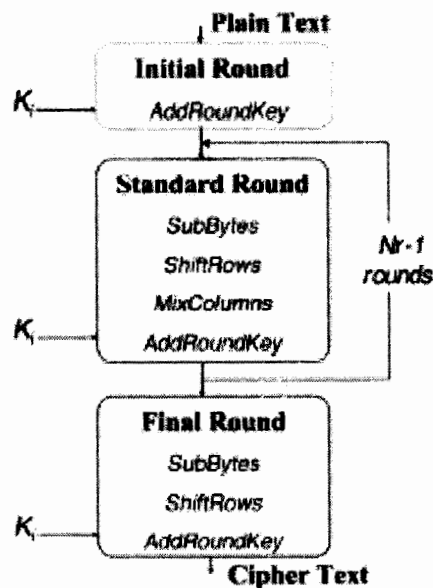


Fig 2.1 - Encryption Rounds

### AddRoundKey() Transformation

The Add Round Key operation is a simple EXOR operation between the State and the Round Key. The Round Key is derived from the Cipher key by means of the key schedule. The State and Round Key are of the same size and to obtain the next State an EXOR operation is done per element:

$$s'(i, j) = s(i, j) \oplus w(i, j).$$

Where  $s$  is the current State,  $s'$  is the next State and  $w$  is the round key.

### SubBytes() Transformation

In the SubBytes step, each byte in the array is updated using an 8-bit S-box. This operation provides the non-linearity in the cipher. The S-box used is derived from the inverse function over  $GF(2^8)$ , having good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function

with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), and also any opposite fixed points.

### ShiftRows() Transformation

In Shift Rows, the rows of State are cyclically shifted with different offsets. In the table of figure 3.2, Row 1 is shifted over  $c_1$  bytes, row 2 over  $c_2$  bytes, and row 3 over  $c_3$  bytes. The values of  $c_1$ ,  $c_2$ , and  $c_3$  depend on the block length  $Nb$ .

Nb	c1	c2	c3
4	1	2	3
6	1	2	3
8	1	3	4

**Fig 2.2 - ShiftRows Table**

### MixColumns() Transformation

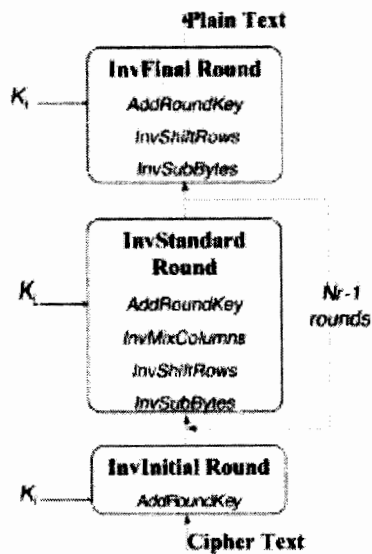
The MixColumn transformation is an operation on different columns. To calculate the MixColumn transformation the columns of the current state are considered as polynomials over  $GF(2^8)$ , e.g. the coefficients of the polynomial are elements of  $F(2^8)$ . Each column (each polynomial) is multiplied by the polynomial

$$a(x) \bmod (x^4+1)$$

$$a(x) = 03x^3 + 01x^2 + 01x + 02.$$

### Decryption Rounds

Each operation that is used for encryption must be inverted to make it possible to decrypt a message. In figure 3.3, the order of these operations is shown:



**Fig 3 - Decryption Rounds**

### Inverse of the AddRoundKey() Transformation

AddRoundKey() is its own inverse, since it only involves an application of the XOR operation.

### InvSubBytes() Transformation

InvSubBytes() is the inverse of the byte substitution transformation, in which the inverse Sbox is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation followed by taking the multiplicative inverse in GF(28).

### InvShiftRows() Transformation

InvShiftRows() is the inverse of the ShiftRows() transformation. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row,  $r = 0$ , is not shifted. The bottom three rows are cyclically shifted by  $Nb$



), ( $Nb - r$  shift bytes, where the shift value  $shift(r, Nb)$  depends on the row number.

Specifically, the `InvShiftRows()` transformation proceeds as follows:

for  $r < 4$  and  $c < Nb$

#### **InvMixColumns() Transformation**

`InvMixColumns()` is the inverse of the `MixColumns()` transformation. `InvMixColumns()` operates on the State column-by-column, treating each column as a fourterm polynomial as described in Sec. 4.3. The columns are considered as polynomials over GF(28) and multiplied modulo  $x^4 + 1$  with a fixed polynomial  $a^{-1}(x)$ , given by

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}.$$

# **Chapter-4**

## **PROPOSED MODEL**

## 4 Proposed Model

System design and architecture basically represents an abstraction of the implementation of the system. It is used to conceive as well as document the design of the software system. It is a comprehensive, composite artefact encompassing all subsystems, packages, and the relationships between them. The software design serves as a communication medium between the architect and other project team members regarding architecturally significant decisions. Therefore it should be kept consistent.

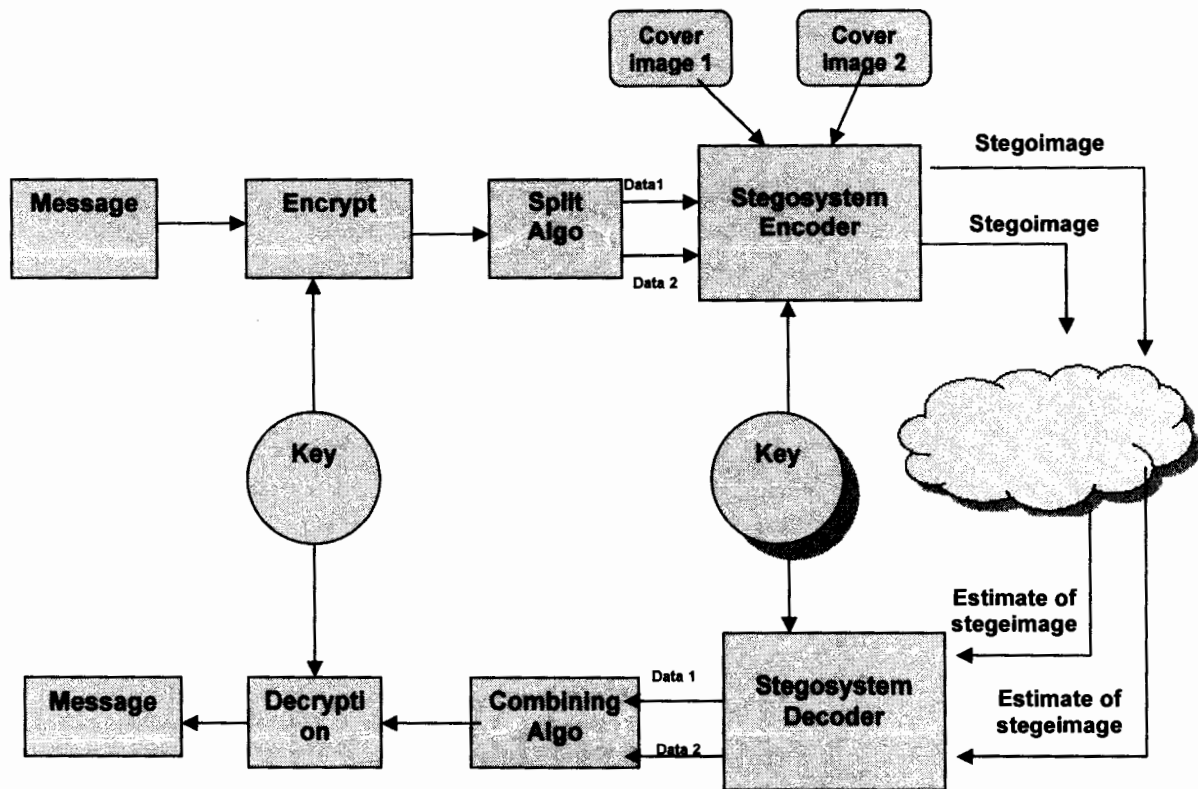
### *4.1 Proposed System*

The Software system provides a comprehensive overview of the system, using different views to depict different aspects of the system. It reflects early decisions and working assumptions on implementing the Vision, as well as decisions concerning the physical and logical architecture and non-functional requirements of the system.

#### **4.1.2 Architecture Design**

The software system diagram or block diagram shown below summarize the important physical groupings of our code that make up the Encrypt Split Image Steganography. (ESIS).

## 4.1.3 ESIS Block Diagram



## 4.2 ESIS WORKING

Encrypt Split Image Steganography comprises of many individual processes. The fundamental concept is the embedding of Splited encrypted text in the cover images. To successfully decode the message and image, attack detection algorithm and combination Algorithm is employed. On sender side the following steps are performed.

1. Message is encrypted first, and the encryption key is generated for Encryption Algorithm AES. For example the encrypted secrete message in binary form .
2. Then encrypted message is taken as input of Splitting Algorithm for splitting the message in two parts
3. In this step transformations are applied on the selected cover images.
4. These transformed cover images and the encrypted parts of messages are taken as input of Embedding Algorithm. The output of this step is the stego-image.

On receiving end the following steps are performed:

1. In this step transformations are applied on the Stego images
2. These transformed images are taken as input of Detection Process. During this process the image processing attacks are detected. If the attacks are detected then those attacks are removed.
3. The encrypted text parts are taken from the stego images and combine them to form an original encrypted message
4. The full encrypted message is then decrypted to find out the original message.

### 4.3 Embedding Algorithm

- Input one array of image and one array of encrypted text for embedding secret message in the image.
- Apply the Discrete cosine transform (DCT) or Discrete wavelet transforms (DWT) or Fast Fourier Transform (FFT) on the image.
- Find the Central point/middle point of the image.
- Select the pixels for embedding.
- Embed the text in the selected pixels around the central point.
- Also embed the specific some values at the corners of an image which makes a pattern
- Replace the least significant bit of each selected pixel.
- Apply the respected inverse transform.

### 4.4 Detection Algorithm

- Input stego- image
- Apply the DCT or DWT or FFT on the stego-image
- Locate and extract the pattern from that specific location.
- Match the extracted pattern with the embedded one.
- Detect the Attacks(Rotation, Resizing, Cropping, Noise)
  - If the extracted and embedded patterns are same then there is no attack
  - If the extracted pattern is the transpose of embedded pattern then there will be the attack of rotation(which may be 90,180,270,360)
  - If the values of the pattern are double, triple or quarter of the original ones then it is Resizing attack
  - If one of the value of the pattern is not extracted then it is the attack of cropping.
- After Detection of attacks, the attacks are removed/recovered
- Then stego image is passed on to extracting Algorithm for extracting of original data.

#### 4.4.1 Example of ESIS Encoding

If we have 40 bits of information which we want to send to other party, bits are placed in some  $m \times n$  matrix ( $m$  row,  $n$  Columns). Actual Data is: H: '00010010', E: '10100010', L: '00110010', L: '00110010', O: '11110010'.

Data Transformed in matrix (5, 8)

	1	2	3	4	5	6	7	8
1	0	0	0	1	0	0	1	0
2	1	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	0
4	0	0	1	1	0	0	1	0
5	1	1	1	1	0	0	1	0

Table1

To obtain Stego-image we have to embed these bits in image. Before embedding one of the transformation (DCT, DWT, FFT) is applied on the selected cover image to obtain the coefficients. If we apply DCT on image, the DCT coefficients of image ( $256 \times 256$ ) before embedding are

	127	128	129	130	131	132	133	134	135
128	-14.496	27.417	-20.457	-5.0873	13.781	-25.8	-6.5634	45.138	-19.105
129	12.961	-10.819	7.4375	-27.753	-9.23	32.674	8.2685	-23.776	19.687
130	11.68	-2.9207	-30.177	16.578	10.914	2.7128	-1.5548	-19.9	-13.679
131	9.6351	2.2956	9.4144	-4.2182	21.704	1.2841	-36.899	1.511	24.007
132	-25.175	2.5304	24.907	14.675	-20.814	2.2309	16.374	-16.144	13.564
133	0.71348	-4.6529	0.34697	3.1824	-16.033	-21.051	17.472	34.434	-30.488
134	-3.5395	4.7112	-0.19071	-15.452	-5.7235	1.1547	6.0022	10.157	-0.8998
135	5.4475	15.636	-18.32	-16.753	11.577	19.782	0.26199	-33.834	-20.734

Figure 3.2: The Values of DCT coefficients before embedding (selected coefficients will be modified)

In embedding Procedure the middle of the image is found. Message is embedded by modifying values of the DCT coefficients. LSB technique is used in our embedding

Algorithm. The Modified Values of DCT coefficients of the cover image are

		128	129	130	131	132	133	134	135
128	196	26.417	-21.457	-5.0673	13.781	-25.8	-7.6634	45.138	-19.105
129	161	-10.819	6.4375	-26.753	-9.23	32.674	8.2685	-22.776	19.687
130	68	-3.9207	-31.177	17.578	11.914	2.7128	-1.5648	-18.9	-13.679
131	151	2.2956	8.4144	-4.2182	21.704	0.28407	-37.899	1.511	24.007
132	75	3.5304	25.907	15.675	-20.814	2.2309	16.374	-16.144	13.564
133	148	-4.6529	0.34697	3.1824	-16.033	-21.051	17.472	34.434	-30.488
134	195	4.7112	-0.19071	-15.452	-5.7235	1.1547	6.0022	10.157	-0.8998
135	175	15.636	-18.32	-16.753	11.577	19.782	0.26199	-33.834	-20.734

**Figure3.3: The Modified DCT Coefficients (Selected Coefficients are modified)**

In next step the inverse of the transform is applied on the image and send this stego image to the receiving party. On receiving end same transformation is applied again and the middle of the image is also found. Then the required information is retrieved and we Found that the Bit Error Rate (BER) is zero. Table2 shows

	1	2	3	4	5	6	7	8
1	0	0	0	1	0	0	1	0
2	1	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	0
4	0	0	1	1	0	0	1	0
5	1	1	1	1	0	0	1	0

**Table2**

Similarly if we apply Discrete Wavelet Transform (DWT) or Fast Fourier Transform (FFT) we found that bit error rate in these cases are also zero, means embedded information is retrieved fully.

### DWT

In DWT the embedding could be possible from level1 to level 4. Each level has four components that are approximated, Horizontal, Vertical and Diagonal. If the information is embedded by modifying the horizontal coefficients of the wavelet level 4 then the values of coefficients before embedding are



	8	9	10	11	12	13	14	15
5 4	606.06	293.19	50.375	277.31	4.3125	-0.75	-3.6875	-7.1875
6 5	40.625	-595.31	-224.25	16.813	6.375	2.625	-1.8125	1.9375
7 5	37.313	914.31	35.437	9	12.75	6.75	4.875	-1.0625
8 5	-10.75	-61.063	-95.563	-6.5	-2.0625	-103	-3	-5
9 0	-470.69	-183.13	45.375	-166.81	13.125	3.8125	38.313	49.5
10 2	-60.375	-63.375	-378.13	293.5	92.813	172.75	-91.5	-101.56
11 3	-80.625	211.88	102.69	-50.063	31.063	117.25	654.31	259.38
12 5	-42.125	24.75	116.19	4.0625	221.75	263.38	271.31	214.81
13 5	-176.94	-53.438	-50.25	-79.063	-136.88	-18.5	37.688	55.875

**Figure 4: DWT Horizontal Coefficients Values at Level 4 before embedding**

The Values of Horizontal Components after embedding are

	7	8	9	10	11	12	13	14	15
6	181.5	40.625	-595.31	-224.25	16.813	6.375	2.625	-1.8125	1.9375
7	-19.375	37.313	914.31	35.437	9	12.75	6.75	4.875	-1.0625
8	-6.25	-11.75	-61.063	-95.563	-6.5	-3.0625	-103	-3	-5
9	-110	-470.69	-183.13	45.375	-167.81	12.125	2.8125	39.313	49.5
10	-292	-61.375	-63.375	-378.13	293.5	92.813	172.75	-90.5	-101.56
11	-626.63	-81.625	210.88	103.69	-50.063	30.063	116.25	655.31	259.38
12	62.375	-42.125	25.75	117.19	5.0625	220.75	262.38	271.31	214.81
13	42.75	-176.94	-53.438	-50.25	-79.063	-136.88	-18.5	37.688	55.875

**Figure 4.1: DWT Horizontal Coefficients Values after embedding**

**DFT**

In Fast Fourier Transform (FFT) the information must be essential to embed at the center of the image because the strongest components of FFT are in the center and we observe that if the components above or below the center of the FFT are modified then the quality of image changed dramatically.

So the coefficients of FFT before embedding are given below

	128	129	130	131	132	133	134	135
128	273.59	-1555.3	168.02	-846.7	2050	-1186.5	1826.5	124.18
129	-740.46	330	-740.46	-145.24	459.22	-915.38	2287.8	-1468.1
130	168.02	-1555.3	273.59	-795.58	-867.71	742.86	-1248.2	443.61
131	-885.37	158.12	-392.7	-751.05	-761.75	-15.436	-931.79	918.84
132	-1357.8	1631.3	-1180.5	-547.48	-262.16	-1212.9	465.83	-732.72
133	-731.11	1348.4	-706.67	422.98	-1646.1	249.93	239.59	542.73

**Figure 5: FFT Coefficients before embedding**

And the values of coefficients after modification are

	127	128	129	130	131	132	133	134	135
127	1.05 + ...	-392.7 - ...	158.12 + ...	-885.37 - ...	655.13 + ...	1328.8 + ...	395.38 + ...	733.49 + ...	-411.16 - ...
128	5.58 - ...	272.59	-1555.3	168.02	-846.7	2049	-1187.6	1827.5	124.18
129	5.24 - ...	-740.46	330	-740.46	-145.24	458.22	-915.38	2287.8	-1468.1
130	6.7 - ...	168.02	-1555.3	273.59	-794.58	-867.71	742.86	-1248.2	443.61
131	5.13 - ...	-885.37	158.12	-892.7	-760.05	-761.75	-15.436	-930.79	918.84
132	7.4 - ...	-1356.8	1631.3	-1180.5	545.48	-263.16	-1213.9	465.83	-732.72
133	11.34 - ...	-731.11 - ...	1348.4 + ...	-706.67 - ...	422.98 + ...	-1646.1 + ...	249.93 - ...	239.59 + ...	542.73 + ...

Figure 5.1: FFT Coefficients after embedding

**Resistance of data to image processing attacks**

One of our goals is to check the robustness of image when watermarks are used in an image. Another key issue is that how far the data is resistant to the possible image processing attacks like image resizing, rotation, cropping, noise and compression.

**DCT**

If the attack on the stego-Image is resizing of an image by 2 times then the values of DCT coefficients after this attack are

	128	129	130	131	132	133	134	135	
128	4.84	45.211	-36.675	-8.6845	23.495	-43.929	-12.861	78.654	-32.401
129	182	-18.493	10.989	-45.612	-15.716	55.562	14.042	-38.629	33.345
130	964	-6.6929	-63.155	29.932	20.261	4.6073	-2.6542	-32.014	-23.141
131	447	3.9138	14.328	-7.1733	36.862	0.48182	-64.2	2.5551	40.558
132	918	6.0112	44.055	26.622	-35.304	3.7791	27.7	-27.278	22.886
133	148	-7.912	0.58926	5.3978	-27.16	-35.614	29.519	58.101	-51.375
134	128	8.0005	-0.32346	-26.174	-9.6825	1.9508	10.128	17.115	-1.5142
135	503	26.518	-31.03	-28.341	19.559	33.377	0.44147	-56.937	-34.846

Figure 6: DCT Coefficients after Resizing (View Selected Portion)

After resizing attack the information bit matrix (5\*8) shown in Table3 is found which indicates that information is not fully recovered.

	1	2	3	4	5	6	7	8
1	1	1	1	1	0	1	0	1
2	1	0	0	0	1	0	1	1
3	1	0	1	0	0	1	1	0
4	1	0	0	0	0	1	0	0
5	0	0	0	0	1	1	0	1

Table 3

$$\begin{aligned} \text{The bit error rate (BER)} &= \text{no. of corrupted bits} / \text{Total no. of bits sent} \\ &= 25/40 \\ &= 0.625 \end{aligned}$$

If the attack on the stego image is rotation of an image by 90 degree then the values of DCT coefficients after this attack are

		128	129	130	131	132	133	134	135
128	999	-26.417	10.819	3.9207	-2.2956	-3.5304	4.6529	-4.7112	-15.636
128	4.85	-21.457	6.4375	-31.177	8.4144	25.907	0.34697	-0.19071	-18.32
130	157	5.0873	26.753	-17.578	4.2182	-15.675	-3.1824	15.452	16.753
131	184	13.781	-9.23	11.914	21.704	-20.814	-16.033	-5.7235	11.577
132	822	25.8	-32.674	-2.7128	-0.28407	-2.2309	21.051	-1.1547	-19.782
133	051	-7.5634	8.2685	-1.5648	-37.899	16.374	17.472	6.0022	0.26199
134	174	-45.138	22.776	18.9	-1.511	16.144	-34.434	-10.157	33.834
135	265	-19.105	19.687	-13.679	24.007	13.564	-30.468	-0.8998	-20.734

**Figure 6.1: DCT Coefficients after Rotation (View Selected Portion)**

So after rotation we found this information bit matrix (5\*8) shown in table 4 which is not fully recovered information.

	1	2	3	4	5	6	7	8
1	1	0	1	1	0	0	1	0
2	0	0	0	0	1	0	1	1
3	1	0	0	0	0	0	1	0
4	1	0	1	1	1	1	0	1
5	1	1	1	1	1	1	0	0

**Table 4**

$$\begin{aligned} \text{The bit error rate (BER)} &= \text{no. of corrupted bits} / \text{Total no. of bits sent} \\ &= 17/40 \\ &= 0.425 \end{aligned}$$

If the attack on the stego image is cropping of an image the information bit matrix (5\*8) is found, which is not fully recovered information. This matrix is shown in table 5.

	1	2	3	4	5	6	7	8
1	0	0	1	1	0	1	0	1
2	1	0	0	0	1	0	1	1
3	1	0	1	0	0	1	1	0
4	1	0	0	0	0	1	0	0
5	0	0	0	0	1	1	0	1

Table 5

The bit error rate (BER) = no. of corrupted bits/ Total no. of bits sent

$$= 23/40$$

$$= 0.575$$

If the attack on the stego image is compression of an image the information bit matrix (5\*8) is found, which is not fully recovered information. This matrix is shown in table 6.

	1	2	3	4	5	6	7	8
1	0	0	0	1	0	0	0	0
2	1	1	0	0	1	0	0	1
3	1	0	1	1	0	0	0	1
4	1	0	0	1	1	0	0	1
5	1	1	0	0	1	1	1	0

Table 6

The bit error rate (BER) = no. of corrupted bits/ Total no. of bits sent

$$= 18/40$$

$$= 0.45$$

#### 4.5 Discrete Wavelet Transformation ( DWT)

If the attack on the stego-Image is resizing of an image by 2 times then the values of DWT coefficients after this attack are

	7	8	9	10	11	12	13	14	15
5	2.25	-2.25	-10	-11	-8.5	-18	2.5	773	846.5
6	-7.5	-18.75	-13	-3	-9.75	-15.25	209.25	29.25	-10
7	2.25	-6.25	-3.5	-6.5	-8.25	-24	156.5	-4.5	6.25
8	5	16	12.25	1.75	2.5	186.25	-1.75	-173.75	-528.5
9	1.5	-11.75	821	710	698.75	293.75	-377	50.5	163
10	-19.25	639	6.25	14.75	13	4.25	324.5	-348.25	267.5
11	807.25	44.5	22.5	4.5	2.5	-60.5	123.75	279.5	-17.25
12	-14	-33	-9.5	-8.5	-6	-6	110.25	117	-45.25
13	6	21	-4.25	-7.5	0.25	-14	-1	-8	54.25

**Figure 7: DWT Coefficients after resizing (View Selected Portion)**

So after resizing we found this information bit matrix (5\*8) shown in table 7 which is not fully recovered information

	1	2	3	4	5	6	7	8
1	0	1	0	1	0	1	0	0
2	1	1	0	0	1	0	1	0
3	0	1	1	0	1	0	0	0
4	1	0	0	1	1	0	0	1
5	0	0	0	0	1	0	1	0

**Table 7**

$$\begin{aligned}
 \text{The bit error rate (BER)} &= \text{no. of corrupted bits} / \text{Total no. of bits sent} \\
 &= 19/40 \\
 &= 0.475
 \end{aligned}$$

If the attack on the stego image is rotation of an image by 90 degree then the values of DWT coefficients after this attack are

	8	9	10	11	12	13	14	15
4	-103	2.8125	172.75	116.25	262.38	-18.5	-51.812	86.688
5	-3.0625	12.125	92.813	30.063	220.75	-136.88	-60.688	-20.062
6	-6.5	-167.81	293.5	-50.063	5.0625	-79.063	-55.75	62.188
7	-95.563	45.375	-378.13	103.69	117.19	-50.25	-16.875	26.75
8	-61.062	-183.13	-63.375	210.88	25.75	-53.438	15.25	20.125
9	-11.75	-470.69	-61.375	-81.625	-42.125	-176.94	-21.5	-22
10	-6.25	-110	-292	-626.63	62.375	42.75	159.94	-262.31
11	-95.313	37.875	9.1875	3.0625	-84.25	-130.56	-83.625	-61.813
12	33.313	-3.75	11.938	-4	-0.8125	0.125	10.062	-47.875

**Figure 7.1: DWT Coefficients after rotation (View Selected Portion)**

So after rotation we found this information bit matrix (5\*8) shown in table 8 which is not fully recovered information

	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	0	0
2	1	0	0	1	0	0	1	1
3	1	1	1	0	1	0	0	1
4	0	1	1	1	0	0	0	0
5	0	1	0	1	0	0	0	1

**Table 8**

The bit error rate (BER) = no. of corrupted bits/ Total no. of bits sent

$$= 21/40$$

$$= 0.525$$

If the attack on the stego image is cropping of an image the information bit matrix (5\*8) is found, which is not fully recovered information. This matrix is shown in table 9

	1	2	3	4	5	6	7	8
1	0	1	0	0	1	1	1	1
2	1	1	0	1	1	0	0	1
3	1	0	0	0	1	0	0	0
4	0	1	1	0	0	0	1	1
5	0	1	1	0	1	1	0	0

**Table 9**

$$\begin{aligned} \text{The bit error rate (BER)} &= \text{no. of corrupted bits/ Total no. of bits sent} \\ &= 24/40 \\ &= 0.6 \end{aligned}$$

If the attack on the stego image is compression of an image the information bit matrix (5\*8) is found, which is not fully recovered information. This matrix is shown in table 10

	1	2	3	4	5	6	7	8
1	0	0	0	1	0	0	1	0
2	1	0	0	0	0	0	1	0
3	0	0	1	1	0	0	1	0
4	0	0	1	1	1	0	1	0
5	0	0	0	0	1	1	0	0

**Table 10**

$$\begin{aligned} \text{The bit error rate (BER)} &= \text{no. of corrupted bits/ Total no. of bits sent} \\ &= 9/40 \\ &= 0.225 \end{aligned}$$

## 4.6 Discrete Fourier Transformation (DFT)

If the attack on the stego-Image is resizing of an image by 2 times then the values of DFT coefficients after this attack are

	127	128	129	130	131	132	133	134	135
125	566.7 - ...	1476.5 + ...	-2027.2 - ...	2307.5 + ...	794.92 + ...	-1783 - ...	3248.7 + ...	-2747 - ...	1833.2 + ...
126	75.2 + ...	-685.26 + ...	1472.7 - ...	-33.699 + ...	2707.7 - ...	-315.69 - ...	851.84 - ...	236.31 - ...	-1479.4 + ...
127	64.1 + ...	-1469.9 + ...	1141.7 - ...	-10.93 + ...	249.14 - ...	997.57 - ...	905.68 - ...	931.16 - ...	-1639.5 + ...
128	957.6 + ...	6.7724	-19.202	6.7075e-012	1802.4	4090.5	-443.91	1802.6	-901.2 - ...
129	301.2 + ...	-9.1424	0	9.0309	1855.1	3664.5	-301.08	2677.9	-1723.2 + ...
130	324.7 + ...	-6.8212e...	18.968	-6.6082	1866.8	1795.3	436.43	-604.06	-420.03 - ...
131	19.14 - ...	10.796	-1114	1416.8	-1489.2	3785.9	-1177.5	2946.9	-1958.6 - ...
132	374.7 - ...	32.882	-1419.4	652.43	-1105.2	1730.1	-697.33	-1068.1	-360.99 + ...
133	75.63 + ...	-2224.1 + ...	1930.1 - ...	-1388.6 + ...	1455.5 - ...	613.81 + ...	-2146.2 - ...	967.18 - ...	899.66 - ...

**Figure8: FFT Coefficients after resizing**

So after resizing we found this information bit matrix (5\*8) shown in table 11 which is not fully recovered information.

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>
<b>1</b>	1	0	1	0	1	0	1	1
<b>2</b>	0	0	1	1	0	1	1	0
<b>3</b>	0	0	0	0	1	0	0	0
<b>4</b>	0	1	0	0	1	1	0	1
<b>5</b>	0	0	1	0	0	1	1	0

**Table 11**

The bit error rate (BER) = no. of corrupted bits/ Total no. of bits sent  
 = 24/40  
 = 0.6

If the attack on the stego-Image is rotation of an image by 2 times then the values of DFT coefficients after this attack are

	127	128	129	130	131	132	133	134
126	52.7 - ...	-2196.6 - ...	-595.96 - ...	799.2 - ...	617.64 - ...	196.05 - ...	1623.2 - ...	1713 + ...
127	3.46 - ...	801.43 - ...	99.063 - ...	748.17 - ...	788.52 + ...	575.63 + ...	-460.33 - ...	302.86 - ...
128	.1 - ...	-167.97	740.24	-272.61	375.05	1152.6	724.58	339.66
129	3.12 - ...	1555.3	-330	1555.3	-158.12	-1631.3	-1348.4	-481.65
130	1.05 + ...	-272.51	740.24	-167.97	685.1	1356.4	702.74	218.97
131	1.52 - ...	748.17	99.063	801.43	-660.46	-722.47	311.46	-712.04
132	.64 + ...	799.2	-595.96	-2196.6	-1362.7	-1521.1	-322.81	-816.9

**Figure 8.1: DFT Coefficients values after rotation**

So after rotation we found this information bit matrix (5\*8) shown in table 8 which is not fully recovered information

	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>
<b>1</b>	1	0	0	0	0	1	0	0
<b>2</b>	0	0	0	0	1	0	1	1
<b>3</b>	1	0	1	1	0	1	1	0
<b>4</b>	1	0	0	0	0	0	0	0
<b>5</b>	0	1	0	0	1	0	0	0

**Table 8**



$$\begin{aligned}\text{The bit error rate (BER)} &= \text{no. of corrupted bits} / \text{Total no. of bits sent} \\ &= 19/40 \\ &= 0.475\end{aligned}$$

**Chapter-5**  
**EXPERIMENTAL**  
**RESULTS**

## 5 Experimental Results

We have tested our system based on the test images “Cameraman”, “Wbarb” and “Peppers” of size 256\*256. If the length of the secret message is 2000 bits and only one bit is changed per pixel then the BER after attacks are shown in table 9 and 10:

	<b>DCT</b>	<b>DWT</b>	<b>FFT</b>
<b>No Attack</b>	0.000	0.000	0.000
<b>Rotation ( 90 intervals)</b>	0.425	0.525	0.475
<b>Resizing</b>	0.625	0.475	0.6
<b>Cropping</b>	0.575	0.6	0.45
<b>JPEGCompression</b>	0.6	0.5	0.55
<b>S/N Ratio</b>	40.2669db	50.8540db	6.2991db

**Table 9: Average Bit Error Rate**

We also check the bit error rate of plain images including White and black against image processing attacks. The results are shown in table 10.

	<b>Rotation</b>	<b>Resizing</b>	<b>Cropping</b>	<b>No Attack</b>
<b>White image</b>	0.15	0.000	0.000	0.00
<b>Balck Image</b>	0.15	0.000	0.000	0.00

**Table 10: Average Bit Error Rate of Plain Images against Attack**

We calculate the capacity of our system considering the 1-bit/pixel, 2-bit/pixel, and 3-bit/pixel embedding schemes. The results are shown in table 11.

<b>Size of image</b>	<b>1-bit/pixel embedding</b>	<b>2-bit/pixel embedding</b>	<b>3-bit /pixel embedding</b>
<b>256*256</b>	32768 bits	65536bits	98304 bits
<b>512*512</b>	131072 bits	262144 bits	393216 bits

**Table 11: Data Embedding Capacity of System**

## 5.1 Robustness of an Image with DCT Applied

In the DCT we embed the data of length of the 128 bit(64 bytes) and of the 2040 bits (255 bytes) by using 2-bit/pixel, 3-bit/pixel and 4-bit/pixel embedding scheme. The experimental results are shown in figures 9 to 12.



Figure 9: original Image



Figure 10: Stego-image (one bit per pixel)

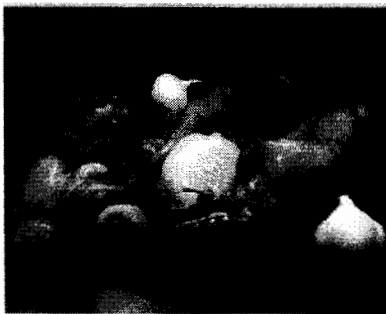


Figure 11: 3-bits changed per pixel



Figure 12: 4 bits changed per pixel

## 5.2 Robustness of an Image with DWT Applied

In DWT we embed the data at different levels e.g. at level 1, 2, 3 and 4. We noticed the quality of an image changed if the embedding is done by changing the 2,3 and 4 bits per pixel respectively. The length of the secret message considered is 1Kb (1024 bits) and the experimental results are given in figures 13 to 16.



Figure 13: Embedding At Level 1 (one bit/ pixel)



Figure 14: Embedding At level 1(4 bits/ pixel)



Figure 15: Embedding At Level 2 (4 bits/Pixel)



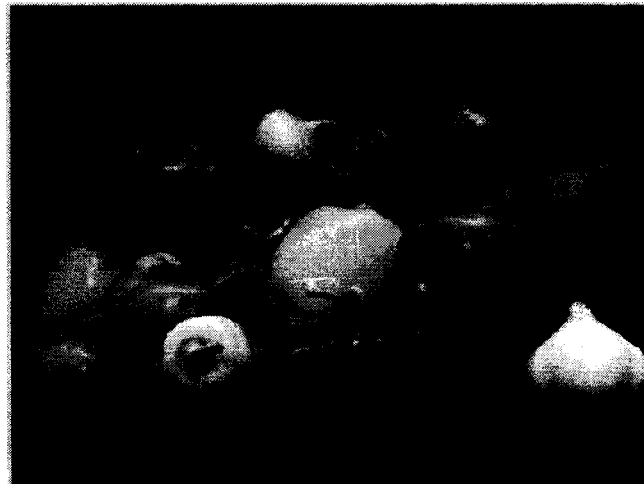
Figure 16: Embedding At Level 4(5 bits/ pixel)

### ***Robustness of Image Against Resizing Attack***

The size of the stego image have been multiple of the original size after the attack of resizing. For example if we resize the image 2- times and it's original size is  $256 * 256$  then the resulting sizing will be  $512 * 512$ . We analyze the quality of images and the recovery of data after introducing the attack of resizing. The quality of stego images are shown in the figures 17 to 19



**Figure 17: Resized Stego-Image ( DCT is applied)**



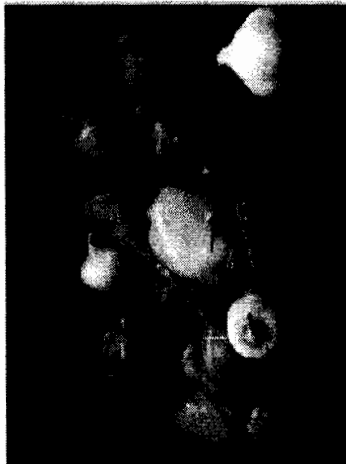
**Figure 18: Resized Stego-Image (DWT applied)**



**Figure 19: Resized Stego-Image (DFT applied)**

***Robustness of Image against Rotation Attack***

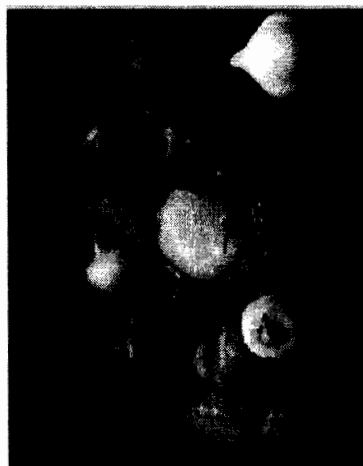
We analyze the quality of images and the recovery of data after introducing the attack of rotation. After attack of rotation the data cannot be recovered fully. The quality of atego images are shown in figures 20 to 22



**Figure 20: Rotated at 90 degree (DCT applied)**



**Figure 21: Rotated at 90 degree (DWT)**



**Figure 22: Rotated at 90 degree (DFT applied)**

### 5.3 JPEG Compression Attack

The signal to noise ratio introduced after a JPEG compression attack on the stago image is given in the figure 23 and 24 graphs for the DCT and DFT.

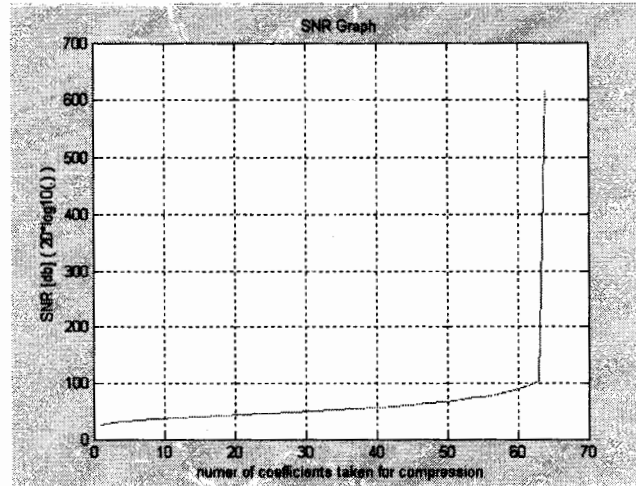


Figure 23: SNR Graph of DCT

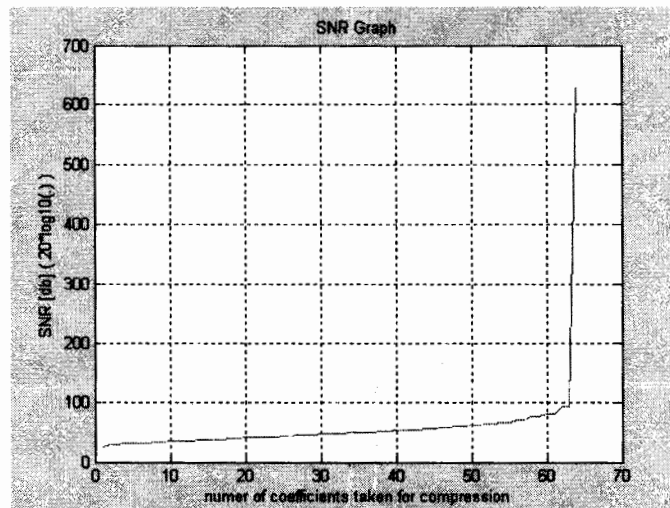


Figure 24: SNR Graph for FFT



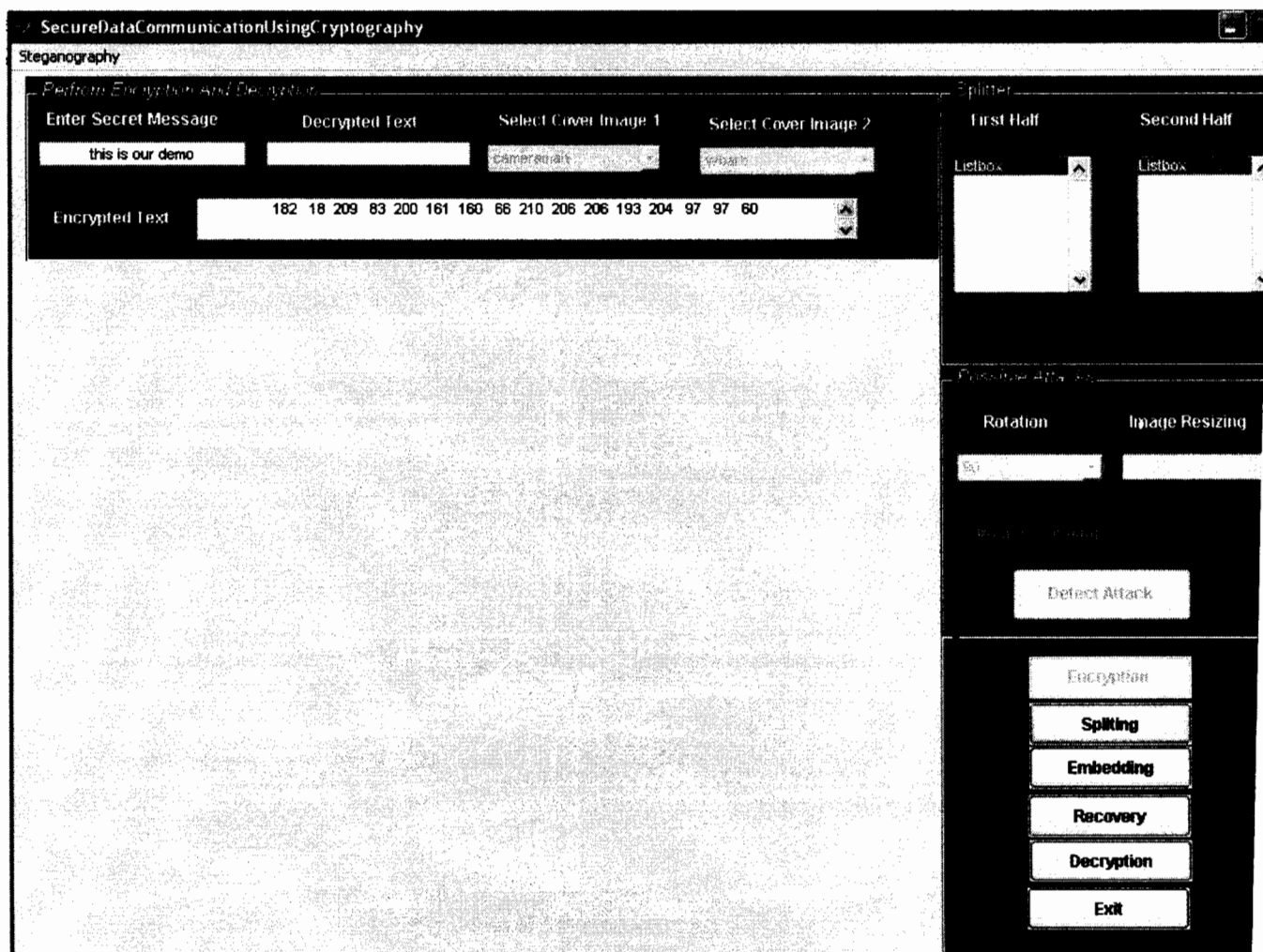
# **Chapter-6**

# **IMPLEMENTATION**

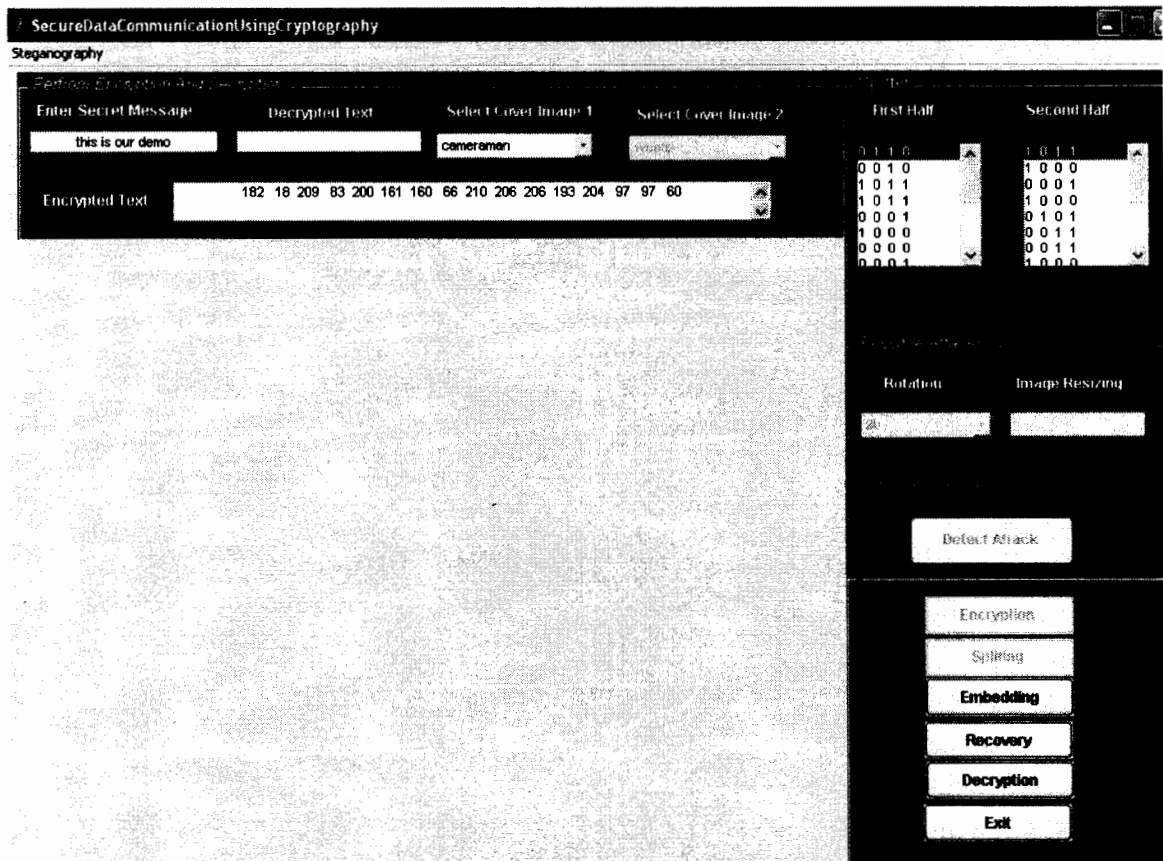
## 6 Implementation

Tool for the implementation for the Proposed System MATLAB is used, the System is divided in to different modules as Encryption, Splitting, Embedding, Attack Detection, Recovery and Decryption of the original message.

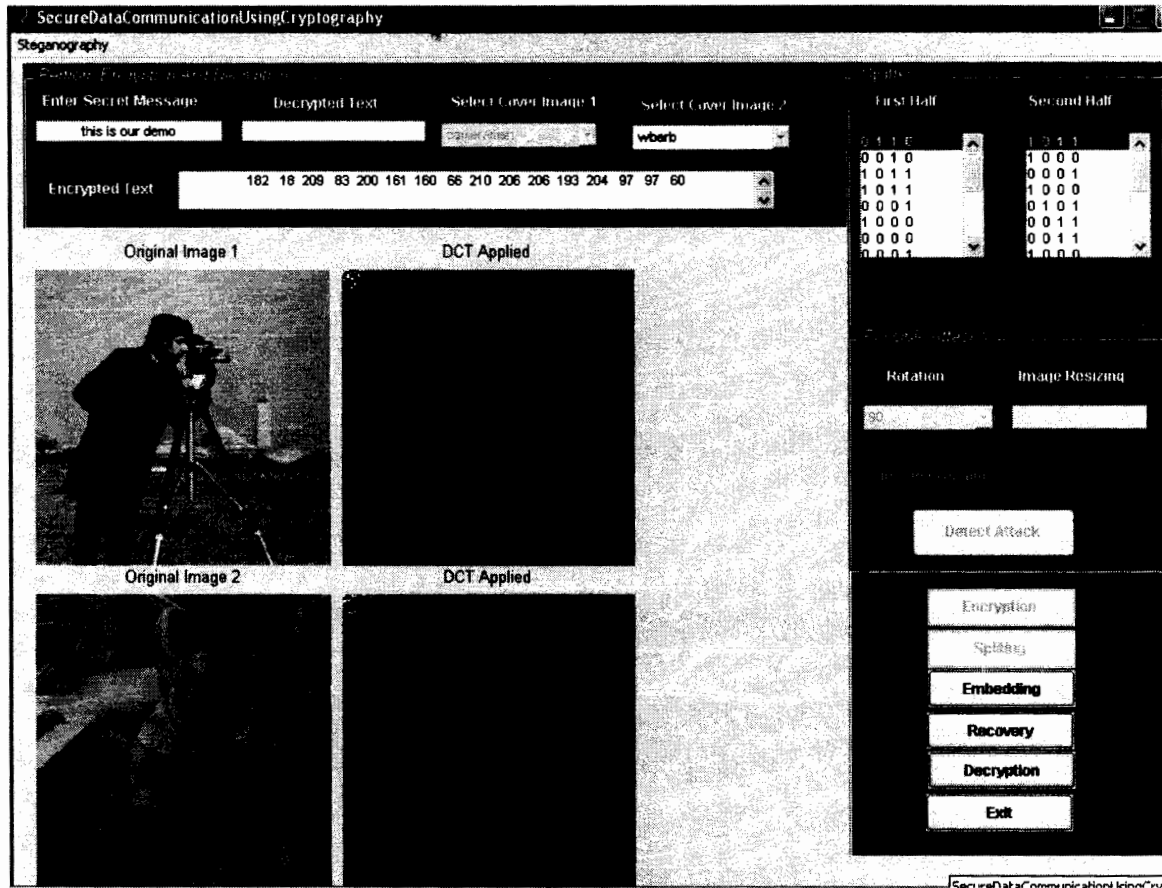
The First screen shot describes the Encryption Module that a Secret Text Message of 16 bytes is taken as an input which is Encrypted by AES(Advanced Encryption Standard), which changed the plain text message into Cipher Text Message.



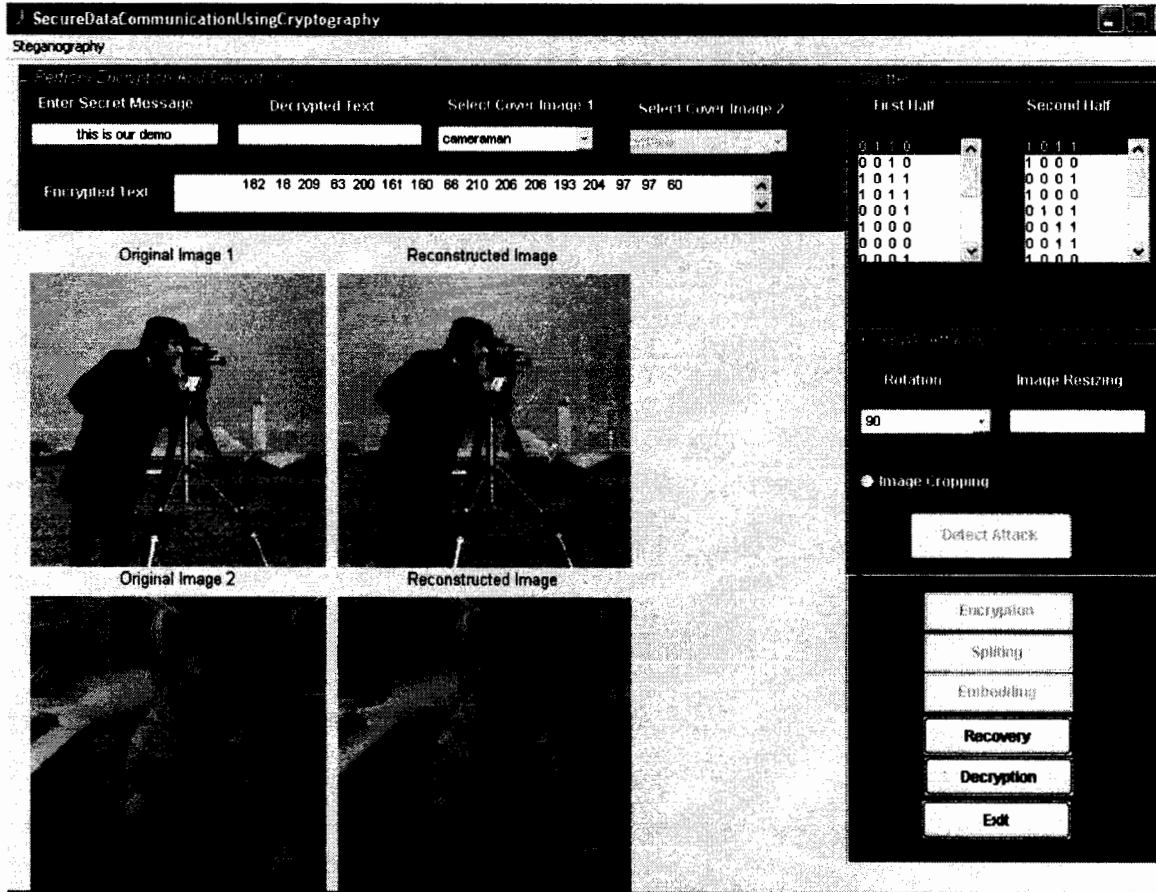
This Screen shot describes the Splitting Module that the Cipher Text Message is divided/splitted on the basis of Bits into two halves named as First half and Second Half shown in two list boxes.



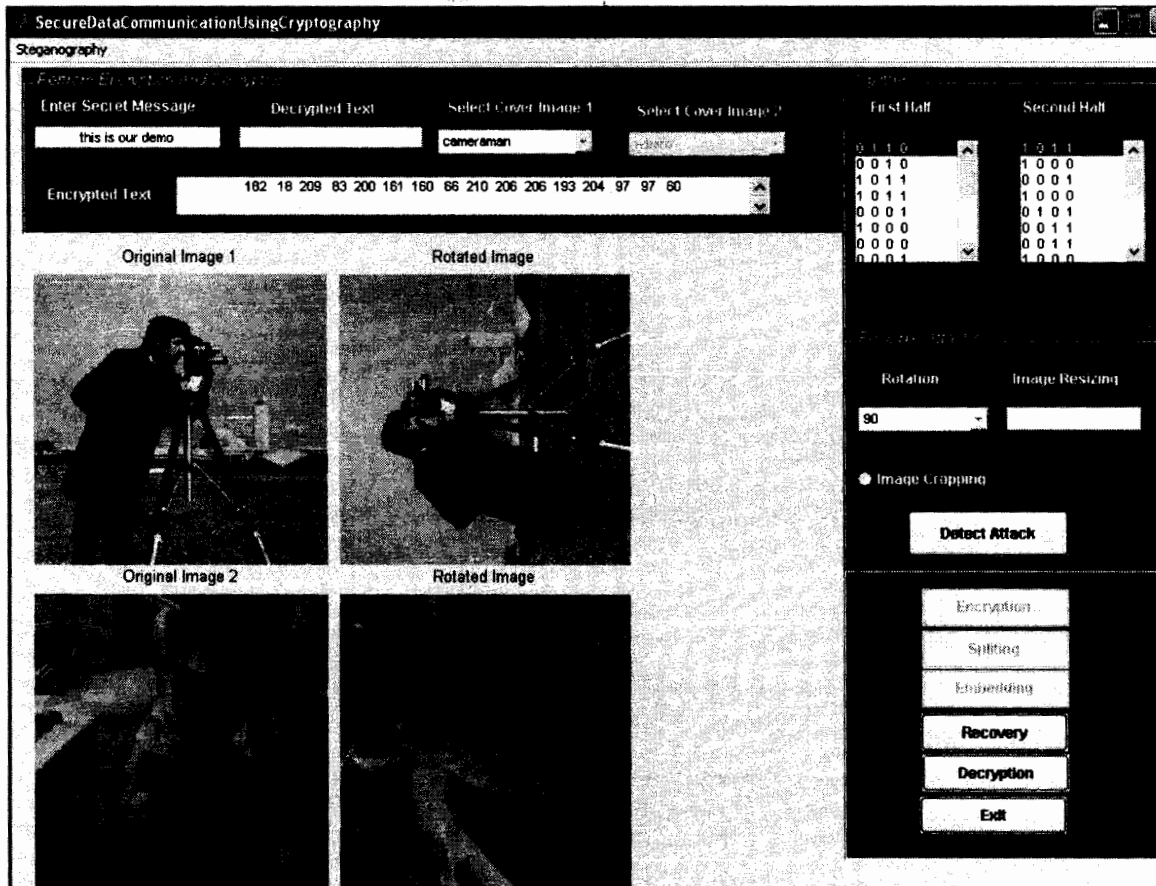
Then two cover images are selected to hide the encrypted , splitted message, for that we chose three Transformation Techniques named DCT,DWT and DFT.



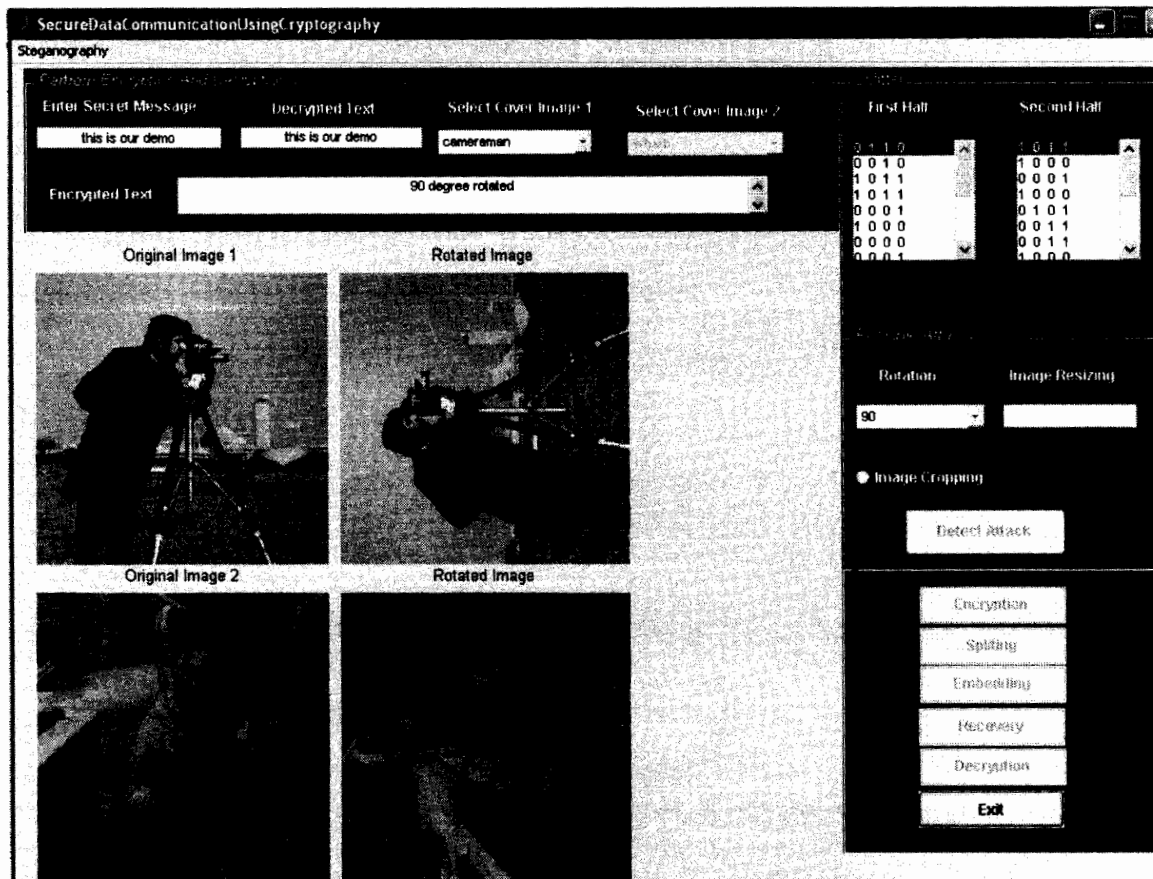
Now in this screen shot the message is embedded into Transformed Image and result is in the form of two Stego Images.



After embedding there is a possibility of any attack on the system or Stego Images here one of the possible Attacks (Rotation) is introduced on the selected Stego Images.



Detection Algorithm detected the Rotational Attack and also Recovered it, and after recovery the original Secret Message is achieved by the Decryption and Recombining Algorithm, for Decryption the reverse of AES is used.



## 6.1 AES (Advanced Encryption Standard)

```
function [s_box, inv_s_box, w, poly_mat, inv_poly_mat] = aes_init
clc
[s_box, inv_s_box] = s_box_gen(1);
rcon = rcon_gen(1);
key_hex = {'00' '01' '02' '03' '04' '05' '06' '07' ...
```

```

    '08' '09' '0a' '0b' '0c' '0d' '0e' '0f'};
key = hex2dec(key_hex);
w = key_expansion(key, s_box, rcon, 1);
[poly_mat, inv_poly_mat] = poly_mat_gen(1);
function ciphertext = cipher(plaintext, w, s_box, poly_mat, varargin)
if nargin > 4
    verbose_mode = 1;
else
    verbose_mode = 0;
end

if iscell(plaintext) | prod(size(plaintext)) ~= 1
    error('Plaintext has to be a vector (not a cell array) with 16 elements.')
end
if any(plaintext < 0 | plaintext > 255)
    error('Elements of plaintext vector have to be bytes (0 <= plaintext(i) <= 255).')
end
if iscell(w) | any(size(w) ~= [44, 4])
    error('w has to be an array (not a cell array) with [44 x 4] elements.')
end
if any(w < 0 | w > 255)
    error('Elements of key array w have to be bytes (0 <= w(i,j) <= 255).')
end
state = reshape(plaintext, 4, 4);
if verbose_mode
    disp_hex('Initial state :          ', state)
end
round_key = (w(1:4, :));
if verbose_mode
    disp_hex('Initial round key :          ', round_key)
end
state = add_round_key(state, round_key);
for i_round = 1 : 9
    if verbose_mode
        disp_hex(['State at start of round ', num2str(i_round), ':   '], state)
    end
    state = sub_bytes(state, s_box);
    if verbose_mode
        disp_hex('After sub_bytes :          ', state)
    end
    state = shift_rows(state);
    if verbose_mode
        disp_hex('After shift_rows :          ', state)
    end
    state = mix_columns(state, poly_mat);
    if verbose_mode

```



```

    disp_hex ('After mix_columns :      ', state)
end
round_key = (w((1:4) + 4*i_round, :));
if verbose_mode
    disp_hex ('Round key :              ', round_key)
end

state = add_round_key (state, round_key);

end

if verbose_mode
    disp_hex ('State at start of final round : ', state)
end
state = sub_bytes (state, s_box);
if verbose_mode
    disp_hex ('After sub_bytes :          ', state)
end
state = shift_rows (state);
    if verbose_mode
        disp_hex ('After shift_rows :          ', state)
    end
round_key = (w(41:44, :));
if verbose_mode
    disp_hex ('Round key :              ', round_key)
end
state = add_round_key (state, round_key);
    if verbose_mode
        disp_hex ('Final state :              ', state)
    end
end
ciphertext = reshape (state, 1, 16);

```

## 6.2 Splitter

```

function Splitting_Callback(hObject, eventdata, handles)
global R1; global R2; global ciphertext; global c;
j=1;t=1;k=1;
for i=1:16
    while j<=7 & t<=4
        Ptt1(i,t)=c(i,j);
        Ptt2(i,t)=c(i,j+1);
        j=j+2;
        t=t+1;
    end
    RR1=num2str(Ptt1);

```

```

    RR2=num2str(Ptt2);
    j=1;
    t=1;
    k=k+1;
end
Ptt1;Ptt2;
R1=Ptt1;R2=Ptt2;
set(handles.Split1,'String',RR1);
set(handles.Split2,'String',RR2);
set(handles.Splitting,'Enable','off');
set(handles.PopimageSelection,'Enable','on');
guidata(hObject,handles);

```

### 6.3 Embedding Algorithm

```

function Embedding_Callback(hObject, eventdata, handles)
global I;global I2;global K;global K2;global R1;global Rtt;global Rtt2;
global wt;global trans;global trr;global R2;global arr;global arr2;
sz1=size(I)/2;
sz2=size(I2)/2;
m=1;n=1;
for i=1:2
    for j=1:2
        no=real(I(m,n));
        I(m,n)=no+1;
        Rtt(i,j)=I(m,n);
        n=256;
    end
    m=n;
    n=1;
end
m=1;n=1;
for i=1:2
    for j=1:2
        no=real(I2(m,n));
        I2(m,n)=no+1;

```

```
Rtt2(i,j)=I2(m,n);
n=256;
end
m=n;
n=1;
end
% Rtt
switch trans;
case 'DCT'
k=1;l=1;
for i=128:144
for j=128:131
% md= mod(I(i,j),2)
% fl=floor(mod(I(i,j),2))
if((l<=4)&(k<=16))
Ir(k,l)=floor(I(i,j));
if(R1(k,l)==1)
if(floor(mod(Ir(k,l),2))==0)
I(i,j)=Ir(k,l)+1;
else
I(i,j)=Ir(k,l);
end
else
if(floor(mod(Ir(k,l),2))==0)
I(i,j)=Ir(k,l);
else
I(i,j)= Ir(k,l)-1;
end
end
l=l+1;
```

```
    end
end
    k=k+1;
    l=1;
end
Ir
k=1;l=1;
for i=128:144
    for j=128:131
        % md= mod(I(i,j),2)
        % fl=floor(mod(I(i,j),2))
        if((l<=4)&(k<=16))
            Ir2(k,l)=floor(I2(i,j));
            if(R2(k,l)==1)
                if(mod(Ir2(k,l),2)==0)
                    I2(i,j)=Ir2(k,l)+1;
                else
                    I2(i,j)=Ir2(k,l);
                end
            end
        else
            if(floor(mod(Ir2(k,l),2))==0)
                I2(i,j)=Ir2(k,l);
            else
                I2(i,j)= Ir2(k,l)-1;
            end
        end

    end
    l=l+1;
end
end
    k=k+1;
```

```
l=1;
end
Ir2
K= idct2(I);
axes(handles.axes6);
imshow(K,[0 255]);
set(handles.axes6, ...
    'Visible','off', ...
    'Units','pixels', ...
    'Position',[270 265 240 240]);
title('Reconstructed Image');
K2=idct2(I2);
axes(handles.axes10);
imshow(K2,[0 255]);
set(handles.axes10, ...
    'Visible','off', ...
    'Units','pixels', ...
    'Position',[270 1 240 240]);
title('Reconstructed Image');
case 'DFT'
k=1;l=1;
for i=128:144
for j=128:131
    if((l<=4)&(k<=16))
        ar1=real(I(i,j));
        ar2=abs(floor(ar1));
        I(i,j)=ar2;
        Ir(k,l)=ar2;
        md=mod(ar2,2);
        fl=floor(md);
%       Ir(k,l)=real(I(i,j));
```

```
    if(R1(k,l)==0)
        if(fl==0)
            I(i,j)=I(i,j);
            arr(k,l)=I(i,j);
        else
            I(i,j)=I(i,j)-1;
            arr(k,l)=I(i,j);
        end
    else
        if(fl==0)
            I(i,j)=I(i,j)+1;
            arr(k,l)=I(i,j);
        else
            I(i,j)= I(i,j);
            arr(k,l)=I(i,j);
        end
    end

    end
    l=l+1;
end
end
k=k+1;
l=1;
end
arr
lr
K= ifft2(I);
axes(handles.axes6);
imshow(K,[0 255]);
set(handles.axes6, ...
    'Visible', 'off', ...
```

```
'Units', 'pixels', ...
'Position', [270 265 240 240]);
title('Reconstructed Image');
k=1;l=1;
for i=128:144
    for j=128:131
        if((l<=4)&(k<=16))
            ar1=real(I2(i,j));
            ar2=abs(floor(ar1));
            I2(i,j)=ar2;
            Ir2(k,l)=ar2;
            md=mod(ar2,2);
            fl=floor(md);
%       Ir(k,l)=real(I(i,j));
            if(R2(k,l)==0)
                if(fl==0)
                    I2(i,j)=I2(i,j);
                    arr2(k,l)=I2(i,j);
                else
                    I2(i,j)=I2(i,j)-1;
                    arr2(k,l)=I2(i,j);
                end
            else
                if(fl==0)
                    I2(i,j)=I2(i,j)+1;
                    arr2(k,l)=I2(i,j);
                else
                    I2(i,j)= I2(i,j);
                    arr2(k,l)=I2(i,j);
                end
            end
        end
    end
end
```

```

        end
        l=l+1;
    end
end
    k=k+1;
    l=1;
end
Ir2
arr2
K2=ifft2(I2);
axes(handles.axes10);
imshow(K2,[0 255]);
set(handles.axes10, ...
    'Visible', 'off', ...
    'Units', 'pixels', ...
    'Position', [270 1 240 240]);
end
title('Reconstructed Image');
set(handles.Embedding,'Enable','off');
set(handles.PopimageSelection,'Enable','on');
set(handles.PopimageSelection2,'Enable','off');
set(handles.popRotationSelection, 'Enable','on');
set(handles.EdTxtImgReszz, 'Enable','on');
set(handles.radiobutton2, 'Enable','on');
guidata(hObject,handles);

```

## 6.4 Detection Algorithm

```

function AttDetect_Callback(hObject, eventdata, handles)
global J;global J2;global Rtt; global Rtt2;global trans; global cH4;
global cV4;global cD4; global cA4;global cH2; global cV2;global cD2;
global cA2;global cH3;global cV3;global cD3;global cA3; global cH1;

```



```
global cV1;global cD1;global cA1;global c; global s; global ass;
global ass2;global ass1;global K; global K2;
SZ=size(J)/2; SZ2=size(J2)/2;
switch trans
    case 'DCT'
        RT=dct2(J);
        RT2=dct2(J2);
        m=1;n=1;
        for i=1:2
            for j=1:2
                try
                    ass(i,j)=real(RT(m,n));
                    n=256;
                catch
                    msgbox('Image is Cropped','Attack');
                    K=J;
                    K2=J2
                    set(handles.Recovery,'Enable','on');
                end
            end
        end
        m=n;
        n=1;
    end
    m=1;n=1;
    for i=1:2
        for j=1:2
            ass2(i,j)=real(RT2(m,n));
            n=256;
        end
    end
end
```

```
        m=n;
        n=1;
    end
case 'DFT'
    RT=fft2(J);
    m=1;n=1;
    for i=1:2
        for j=1:2
            try
                ass(i,j)=real(RT(m,n));
                n=256;
            catch
                msgbox('Image is Cropped','Detecting Attack');
                K=J;
                K2=J2;
                set(handles.Recovery,'Enable','on');
            end
        end
    end

    end
    m=n;
    n=1;
end
RT2=fft2(J2);
m=1;n=1;
for i=1:2
    for j=1:2
        ass2(i,j)=real(RT2(m,n));
        n=256;
    end
end
m=n;
n=1;
```

```
end
case 'DWT'
[c,s] = wavedec2(J,4,'db1');
cA4=appcoef2(c,s,'db1',4);
cA3 = appcoef2(c,s,'db1',3);
cA2 = appcoef2(c,s,'db1',2);
cA1 = appcoef2(c,s,'db1',1);
[cH2,cV2,cD2]=detcoef2('all',c,s,2);
[cH3,cV3,cD3]=detcoef2('all',c,s,3);
[cH4,cV4,cD4]=detcoef2('all',c,s,4);
[cH1,cV1,cD1]=detcoef2('all',c,s,1);
m=1;n=1;
for i=1:2
    for j=1:2
        try
            ass1(i,j)=real(cD4(m,n));
            n=16;
        catch
            msgbox('Image is Cropped','Detecting Attack');
            set(handles.Recovery,'Enable','on');
            K=J;
        end
    end
    break;
end
m=n;
n=1;
end
[c,s] = wavedec2(J2,4,'db1');
cA4=appcoef2(c,s,'db1',4);
cA3 = appcoef2(c,s,'db1',3);
```

```

cA2 = appcoef2(c,s,'db1',2);
cA1 = appcoef2(c,s,'db1',1);
[cH2,cV2,cD2]=detcoef2('all',c,s,2);
[cH3,cV3,cD3]=detcoef2('all',c,s,3);
[cH4,cV4,cD4]=detcoef2('all',c,s,4);
[cH1,cV1,cD1]=detcoef2('all',c,s,1);
m=1;n=1;
  for i=1:2
    for j=1:2
      ass2(i,j)=real(cD4(m,n));
      n=16;
    end
    m=n;
    n=1;
  end
end
switch trans
case 'DCT'
  ass=real(ass);
  Rtt=real(Rtt);
  Rtt=floor(Rtt)
  ass=floor(ass)
  if(ass==Rtt)
    set(handles.edit7,'String','360 degree rotated');
elseif(Rtt(2,1)==ass(1,2))
  set(handles.edit7,'String','90 degree rotated');
  set(handles.Recovery,'Enable','on');
  set(handles.AttDetect,'Enable','off');
  K=imrotate(J,270);
elseif(Rtt(1,2)==ass(2,1))
  set(handles.edit7,'String','270 degree rotated');

```

```
K=imrotate(J,90);
set(handles.Recovery,'Enable','on');
set(handles.AttDetect,'Enable','off');
elseif(Rtt(1,1)==ass(1,1))
    set(handles.edit7,'String','180 degree rotated');
    K=imrotate(J,360);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
end
ass2=real(ass2);
Rtt2=real(Rtt2);
Rtt2=floor(Rtt2)
ass2=floor(ass2);
if(ass2==Rtt2)
    set(handles.edit7,'String','360 degree rotated');
elseif(Rtt2(2,1)==ass2(1,2))
    set(handles.edit7,'String','90 degree rotated');
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    K2=imrotate(J2,270);
elseif(Rtt2(1,2)==ass2(2,1))
    set(handles.edit7,'String','270 degree rotated');
    K2=imrotate(J2,90);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt2(2,2)==ass2(2,2))
    set(handles.edit7,'String','180 degree rotated');
    K2=imrotate(J2,360);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
end
```

```

sg=0;
rtt=Rtt(1,1);
Ass=ass(1,1);
if((rtt==((Ass)/2))&(sg==0))
    set(handles.edit7,'String','2 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    aa=SZ/2;
    K=imresize(J,aa);
elseif((rtt==floor((Ass)/4))&(sg==0))
    set(handles.edit7,'String','4 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    aa=SZ/4;
    K=imresize(J,aa);
end
rtt=Rtt2(1,1);
Ass=ass2(1,1);
if((rtt==((Ass)/2))&(sg==0))
    set(handles.edit7,'String','2 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    ab=SZ2/2;
    K2=imresize(J2,ab);

elseif((rtt==floor((Ass)/4))&(sg==0))
    set(handles.edit7,'String','4 Times Resized');

```

```
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    ab=SZ2/4;
    K2=imresize(J2,ab);
end
case 'DFT'
    ass=real(ass);
    Rtt=real(Rtt);
    Rtt=floor(Rtt)
    ass=floor(ass);
    if(ass==Rtt)
        set(handles.edit7,'String','360 degree rotated');
    elseif(Rtt(2,1)==ass(1,2))
        set(handles.edit7,'String','90 degree rotated');
        K=imrotate(J,270);
        set(handles.Recovery,'Enable','on');
        set(handles.AttDetect,'Enable','off');
    elseif(Rtt(1,2)==ass(2,1))
        set(handles.edit7,'String','270 degree rotated');
        K=imrotate(J,90);
        set(handles.Recovery,'Enable','on');
        set(handles.AttDetect,'Enable','off');
    elseif(Rtt(1,1)==ass(1,1))
        set(handles.edit7,'String','180 degree rotated');
        K=imrotate(J,360);
        set(handles.Recovery,'Enable','on');
        set(handles.AttDetect,'Enable','off');
    end
    ass2=real(ass2);
    Rtt2=real(Rtt2);
    Rtt2=floor(Rtt2)
```

```

ass2=floor(ass2)
if(ass2==Rtt2)
    set(handles.edit7,'String','360 degree rotated');
elseif(Rtt2(2,1)==ass2(1,2))
    set(handles.edit7,'String','90 degree rotated');
    K2=imrotate(J2,270);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt2(1,2)==ass2(2,1))
    set(handles.edit7,'String','270 degree rotated');
    K2=imrotate(J2,90);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt2(1,1)==ass2(1,1))
    set(handles.edit7,'String','180 degree rotated');
    K2=imrotate(J2,360);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
end
sg=0;
rtt=Rtt(1,1);
Ass=ass(1,1);
if((rtt==((Ass)/4))&(sg==0))
    set(handles.edit7,'String','2 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    aa=512/4;
    K=imresize(J,aa);

elseif((rtt==floor((Ass)/6))&(sg==0))

```



```

    set(handles.edit7,'String','4 Times Resized');
    aa=1024/6;
    K=imresize(J,aa);
    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
end
rtt=Rtt2(1,1);
Ass=ass2(1,1);
if((rtt==((Ass)/4))&(sg==0))
    set(handles.edit7,'String','2 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    ab=512/4;
    K2=imresize(J2,ab);
elseif((rtt==floor((Ass)/6))&(sg==0))
    set(handles.edit7,'String','4 Times Resized');

    set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
    ab=1024/6;
    K2=imresize(J2,ab);
end
case 'DWT'
    ass1=real(ass1);
    ass1=floor(ass1)
    Rtt=real(Rtt);
    Rtt=floor(Rtt)
    if(ass1==Rtt)
        set(handles.edit7,'String','360 degree rotated');
    elseif(Rtt(2,1)==ass1(1,2)) % for DWT

```

```

set(handles.edit7,'String','180 degree rotated');
K=imrotate(J,360);
set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt(2,2)==ass1(2,1)) %for DWT
set(handles.edit7,'String','90 degree rotated');
K=imrotate(J,180);
set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt(1,1)==ass1(2,1)) % for DWT
set(handles.edit7,'String','270 degree rotated');
K=imrotate(J,90);
set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
end
ass2=real(ass2);
Rtt2=real(Rtt2);
Rtt2=floor(Rtt2)
ass2=floor(ass2)
if(ass1==Rtt)
    set(handles.edit7,'String','360 degree rotated');
elseif(Rtt(2,1)==ass1(1,2)) % for DWT
set(handles.edit7,'String','180 degree rotated');
K2=imrotate(J2,90);
set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');
elseif(Rtt(2,2)==ass1(2,1)) %for DWT
set(handles.edit7,'String','90 degree rotated');
K2=imrotate(J2,180);
set(handles.Recovery,'Enable','on');
    set(handles.AttDetect,'Enable','off');

```

```
elseif(Rtt(1,1)==ass1(2,1)) % for DWT
set(handles.edit7,'String','270 degree rotated');
K2=imrotate(J2,360);
set(handles.Recovery,'Enable','on');
set(handles.AttDetect,'Enable','off');
end
sg=0;
rtt=Rtt(2,2)
Ass=ass1(1,1)
if((Ass==(rtt/3))&(sg==0))
set(handles.edit7,'String','2 Times Resized');
sz=SZ/2;
K=imresize(J,sz);
set(handles.Recovery,'Enable','on');
set(handles.AttDetect,'Enable','off');
elseif((rtt==floor((Ass)/4))&(sg==0))
set(handles.edit7,'String','4 Times Resized');
sz=SZ2/2;
K2=imresize(J2,sz);
set(handles.Recovery,'Enable','on');
set(handles.AttDetect,'Enable','off');
end
end
guidata(hObject,handles);
```

## References

- [1] "Internet Security" by Man Young Rhee, Edition june 2003
- [2] "Applied Cryptography Second Edition: protocols, algorithms, and source code in C," by Schneier, B. John Wiley & Sons, 1996.
- [3] "An Evaluation of Image Based Steganography Methods" by *Kevin Curran, Karen Bailey*, International Journal of Digital Evidence Fall 2003, Volume 2, Issue 2.
- [4] "Gigabits per Second Implementation of the IDEA Cryptographic Algorithm" by Antti Hamalainen, Matti Tommiska and Jorna Skytta, 760-769, EPL 2002
- [5] "Data Encryption Standard", a technical report from wikipedia on encyclopedia
- [6] "AES Algorithm (FIPS 197) Advanced Encryption Standard" by ADI-AMD, vocal technologies 2003
- [7] "An Introduction to modern Cryptography", by Oli Cooper, 22nd February 2001
- [8] "RC6 and the AES", by M. J. B. Robshaw, January 9, 2001
- [9] "Provably Authenticated Group Diffie-HGellman Key Exchange" by Emmanuel Bresson, Olovier Chevassul, David Pointcheval, Jean Jacques Quisquater, CCS 01, USA, 2001
- [10] "A Light-Weight Encrypting For Real Time Video Transmission",by Salah Aly, February 5, 2003
- [11] "Performance Comparison of the AES Submissions",by Bruce Schneier, John Kelsey Doug Whitingz David Wagnerx Chris Hall{Niels Ferguson k Version 2.0} February 1, 1999
- [12] "High Capacity Image Steganographic Model", by Lee, Y. K., Chen, L. H., 2000 in IEE Proceedings Vision, Image and Signal Processing, pp. 288-294.

- [13] "JPEG Compression Immune Steganography Using Wavelet Transform", by Jianyun Xu, Andrew H.Sung, Peipei Shi, Qingzhong Liu. International Conference on Information Technology: Coding and Computing (ITCC'04) 2004 IEEE
- [14]"Steganography in Video Conferencing System", by *Westfield, Wolf, G.*, 1998 in Second International Information Hiding Workshop.
- [15] "The Future of Audio Steganography", by *Pal, S.K., Saxena, P. K., Mutto, S.K.*, 2002 in Pacific Rim Workshop on Digital Steganography, Japan
- [16]"A zero tree wavelet coder", by *Martucci, S. A., Sodagar, I., Chiang*, 1997 in IEEE Transactions Circuits and Systems for Video Technology, pp. 109-118.
- [17] "Comparison of Wavelet and Cosine Basis for Representation of Arbitrarily Shaped Image Segments" by *Surendra Ranganath* in 1998
- [18] "Comparison of JPEG and JPEG 2000 in low-power confidential image transmission", by *Mara Rhepp*, winter 2004
- [19] "Security Through Obscurity" , by *Duraiswamy* , IEEE., 1998
- [20] "MEPG Video Encryption Algorithms" , by *Bharat Bhargava Changgui Shi Sheng-Yih Wang*, IN 47907, USA August 13, 2002

# A Secure Model for Data Communication Using Cryptography & Steganography

Syed A. H., M. S. H. Khiyal, Nighat Mir, Beenish Aziz,

Department of Computer Science, International Islamic University, Islamabad.

[drafaq@iiu.edu.pk](mailto:drafaq@iiu.edu.pk), [drsikandar@iiu.edu.pk](mailto:drsikandar@iiu.edu.pk), [beenish\\_480@yahoo.com](mailto:beenish_480@yahoo.com), [nighatahmed@yahoo.com](mailto:nighatahmed@yahoo.com)

## Abstract

*To communicate data in a reliable and secure way has been the area for quite sometime and various solutions have been proposed. Two techniques namely Cryptography and Steganography are generally used for this purpose. The image steganographic techniques are evaluated on the basis of their information hiding capacity, perceptibility and robustness against attacks. The proposed system has added flexibility to the system with an additional layer of security by using the splitter, which makes the system impregnable and the additional embedding of pseudo data helps in recognizing attacks and in recovery of data after attacks. The system can detect common image processing attacks like resizing, rotation, cropping and compression.*

**Key Words:** Steganography, Watermarking, encryption, AES, Digital Fourier Transformation, Wavelet Transformation.

## 1. Introduction

Data communication is the exchange of information between computers/parties across a network using different protocols. Communication applications focus on transmission in an efficient and reliable manner. To communicate data in a reliable and secure way has been the area of research for quite sometime and various solutions have been proposed. Two techniques are generally used for secure data communication, namely, Cryptography and Steganography.

### 1.1 Cryptography

It is the science of keeping secrets and is used for communicating secret data over non-secure channel objectives of cryptography for secure data where sender maintains message secrecy by converting data (Plain-text) into an unintelligible form (Cipher-text) by the process of encryption. The receiver recovers the original plain-text data by the process of decryption. Basic concerns for communication are Confidentiality, Authentication and Integrity [1].

Main frameworks for achieving the goals of cryptography are Symmetric and Asymmetric encryption. Symmetric Encryption is based on private key encryption in which the same key is used for both encryption and decryption where as Asymmetric Encryption involves a public key for decryption and a private key for encryption. Different Cryptographic algorithms have been proposed including the well known AES (Advanced Encryption Standard) which is based on Symmetric encryption [2].

### 1.2 Steganography

It aims at hiding data (text, image, audio, video etc.) in such a way that there is no indication of the hidden message. This is achieved by using a cover file and an embedding file. The term "cover" is used to describe the original data and the information to be hidden in the cover data is called "embedded" data. The "stego" contains both cover and embedded data.

There are three characteristics of information hiding system, capacity, security and robustness [3]. A lot of research has recently focused on using images as a cover for transferring covert messages [5, 6, 7].

### 1.3 Image Steganography principles:

- 1 The difference of the cover image and the stego image should be visually un-noticeable.
- 2 The message hiding method should be reliable.
- 3 Length of the message should be unlimited
- 4 The message recovery rate should be high after these image manipulations [4]

## 2. Previous work

There are a variety of methods suggested for image steganography. In 2000, Lee & Chen described a method to place the embedding data at the least significant bit (LSB) of each pixel of the cover image [5]. Altering LSB doesn't change the quality of image to human perception but this scheme is sensitive to a variety of image processing attacks like compression, cropping etc. Lee & Chen also presented a method to use the most significant bit of each pixel in the cover image to embed

the secret message [5]. This method improves sensitivity to modification, but it degrades the quality of stego-image.

A number of researches have used image transformation techniques before hiding data in the images. Functions such as the Discrete Cosine Transform (DCT), Discrete Fourier Transformation (DFT) and Discrete Wavelet Transform (DWT) are widely applied [6], [7], [8], [9], and [10]. These methods hide the messages in the significant areas of the cover image which makes them robust against compression, cropping and some other image processing attacks.

### 3. Problem Identification

An encrypted message can be intercepted by a cryptanalyst who may force the sender to reveal the keys that generated the cipher text or use other methods to break/guess the key to uncover the message. This method also has the drawback that the sender and receiver's identity can be known by the analyst. In Image Steganography, the existence of hidden information in an image is usually imperceptible and therefore does not cause an alarm.

In the paper "Security Through Obscurity", Dr. K. Duraiswamy proposed a scheme to combine cryptography and steganography to enhance the security while maintaining it impervious. [11]. Even with the combination of Cryptography and Steganography, the message can be hacked [12] or attacked to make the message useless for the receiver [13].

### 4. Proposed Solution (ESIS):

An Encrypto-Split Image Steganography (ESIS) System is suggested here to produce an efficient and stable platform for secret communication. This proposed scheme adds an additional layer of splitting the encrypted message into two parts, each of which is hidden in a different cover image.

First the sender sends the plain text message to the Encryption Phase which converts it into an encrypted secret message by using Advanced Encryption Standard (AES) and then the Splitter Algorithm splits the message into two halves on the basis of bits. Two cover images are taken by the Stego-Encoder for hiding each encrypted message part. The cover image is transformed by using one of the three given transformations (DCT, DWT or DFT). Each of these two cover images are

converted into stego images by hiding respective message part into their coefficients.

At the receiving end, the inverse procedure is applied to get the original message. First we analyzed both stego-images for any attack. In case of an attack, the stego-image is recovered by applying the inverse of attack. Then the stego Decoder extracts the message which is in encrypted form. This data received from both images is combined back and passed to Decryption Phase which performs Decryption to give the original text message.

### Proposed methodology for Attack detection:

For attack detection an image is changed in such a way that some special data is hidden at four corners of any selected image. Basically the coefficient of the image is modified by changing with least significant bits of the coefficients. When the stego image is analyzed for the attack the actual values of stego image are compared with the received stego image. If any attack is detected first it is reversed by recovering the actual data and then embedded data is extracted.

### 4. System Design

The system block diagram shown in Figure 1 summarizes the important physical modules and functionality that make up the Encrypt Split Image Steganography (ESIS) System.

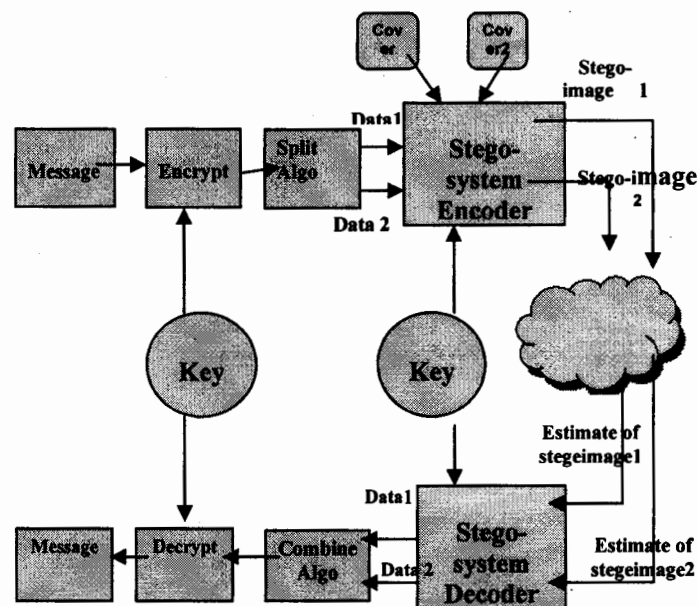


Fig.1 ESIS Block Diagram

### 5. Experimental Results

We have tested our system based on the test images "Cameraman", "Wharf" and "Peppers" of size 256\*256. If the length of the secret message is 2000 bits and only one bit is changed per pixel then the BER after attacks are:

	DCT	DWT	FFT
No Attack	0.000	0.000	0.000
Rotation ( 90 intervals)	0.425	0.525	0.475
Resizing	0.625	0.475	0.6
Cropping	0.575	0.6	0.45
JPEGCompression	0.6	0.5	0.55
S/N Ratio	40.2669db	50.8540db	6.2991db

Table 1 Average Bit Error Rate

	Rotation	Resizing	Croppi ng	No Attack
White image	0.15	0.000	0.000	0.00
Black Image	0.15	0.000	0.000	0.00

Table 2 Bit Error Rate of Plain Images against Attack

Size of image	1-bit/pixel embedding	2-bit/pixel embedding	3-bit /pixel embedding
256*256	32768 bits	65536bits	98304 bits
512*512	131072 bits	262144 bits	393216 bits

Table 3 Data Embedding Capacity of System

#### 4.1 Robustness of an Image when DCT Applied

In the DCT we embed the data of length of the 128 bit(64 bytes) and of the 2040 bits (255 bytes) by using 2-bit/pixel, 3-bit/pixel and 4-bit/pixel embedding scheme. The experimental results are shown in figures 2 to 5.

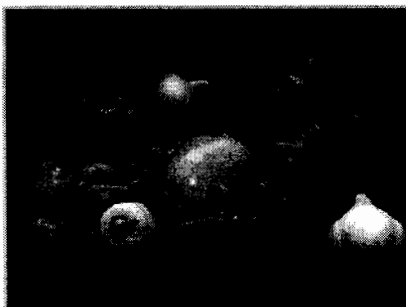


Figure 2: original Image



Figure 3: Stego-image (one bit per pixel)



Figure4:3-bits changed per pixel (64 bytes data embedded)



Figure 5:4 bits changed per pixel (64 bytes data embedded)

#### 4.2 Robustness of an Image with DWT Applied

In DWT we embed the data at different levels e.g. at level 1, 2, 3 and 4. We noticed the quality of an image changed if the embedding is done by changing the 2, 3 and 4 bits per pixel respectively. The length of the secret message considered is 1Kb (1024 bits) and the experimental results are given in figures 6 to 9.





Figure 6: Embedding At Level 1 (one bit per pixel)



Figure 9: Embedding At Level 4 (5 bits per pixel Changed)



Figure 7: Embedding At level 1 (4 bits per pixel changed)



Figure 8: Embedding At Level 2 (4 bits changed)

### 4.3 JPEG Compression Attack

The signal to noise ratio introduced after a JPEG compression attack on the stego image is given in the figure 10 and 11 graphs for the DCT and DFT.

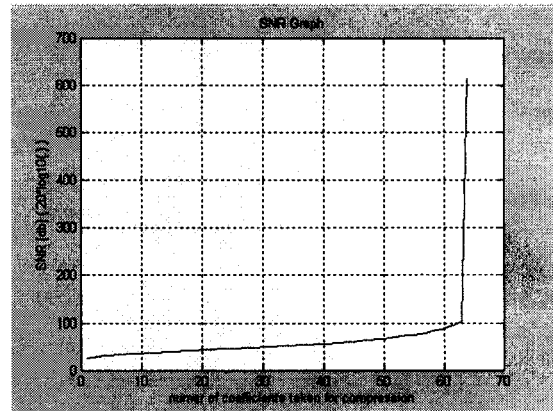


Figure 10: SNR Graph of DCT

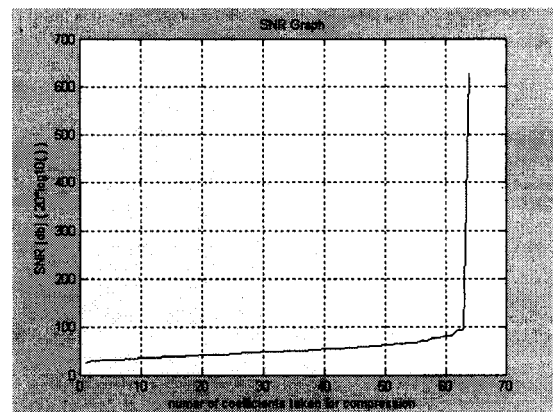


Figure 11: SNR Graph for FFT

## Conclusion

Our proposed model provides one additional security level by developing a splitting algorithm and detection algorithm. This solution has the following advantages.

First, encrypted text can be easily identified because the data is scrambled but encrypted (or scrambled) image would not look anomalous (if the image is not very distinct).

Second, the existence of splitting encrypted text into two portions and embedding them into two different images won't be detected because meaningful text cannot be recovered from any one stego image.

Third, no one would look for textual information hidden (not embedded) in an image. Rather one might sense steganography and try to steg-analyse it, as image is normally used in the context of steganography.

Fourth, the detection algorithm can detect the image processing attacks like resizing, rotation, cropping. This also serves for security purpose. The proposed solution inherits the strong features of cryptography and steganography while omitting their weaknesses.

## Future Goals

- Making it more robust
- Treating on network, introducing transmission noise
- Detecting the attack of Compression.

## References:

- [1] "Internet Security" by Man Young Rhee, Edition June 2003
- [2] "Applied Cryptography Second Edition: protocols, algorithms, and source code in C," by Schneier, B. John Wiley & Sons, 1996.
- [3] "An Evaluation of Image Based Steganography Methods" by Kevin Curran, Karen Bailey, International Journal of Digital Evidence Fall 2003, Volume 2, Issue 2.

- [4] "Gigabits per Second Implementation of the IDEA Cryptographic Algorithm" by Antti Hamalainen, Matti Tommiska and Jorna Skytta, 760-769, EPL 2002
- [5] "Data Encryption Standard", a technical report from wikipedia on encyclopedia
- [6] "AES Algorithm (FIPS 197) Advanced Encryption Standard" by ADI-AMD, vocal technologies 2003
- [7] "An Introduction to modern Cryptography", by Oli Cooper, 22nd February 2001
- [8] "RC6 and the AES", by M. J. B. Robshaw, January 9, 2001
- [9] "Provably Authenticated Group Diffie-Hellman Key Exchange" by Emmanuel Bresson, Olivier Chevassul, David Pointcheval, Jean Jacques Quisquater, CCS 01, USA, 2001
- [10] "A Light-Weight Encrypting For Real Time Video Transmission", by Salah Aly, February 5, 2003
- [11] "Performance Comparison of the AES Submissions", by Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall (Niels Ferguson k Version 2.0) February 1, 1999
- [12] "High Capacity Image Steganographic Model", by Lee, Y. K., Chen, L. H., 2000 in IEE Proceedings Vision, Image and Signal Processing, pp. 288-294.
- [13] "JPEG Compression Immune Steganography Using Wavelet Transform", by Jianyun Xu, Andrew H. Sung, Peipei Shi, Qingzhong Liu. International Conference on Information Technology: Coding and Computing (ITCC'04) 2004 IEEE
- [14] "Steganography in Video Conferencing System", by Westfield, Wolf, G., 1998 in Second International Information Hiding Workshop.
- [15] "The Future of Audio Steganography", by Pal, S.K., Saxena, P. K., Mutto, S.K., 2002 in Pacific Rim Workshop on Digital Steganography, Japan
- [16] "A zero tree wavelet coder", by Martucci, S. A., Sodagar, I., Chiang, 1997 in IEEE Transactions Circuits and Systems for Video Technology, pp. 109-118.

[17] "Comparison of Wavelet and Cosine Basis for Representation of Arbitrarily Shaped Image Segments" by Surendra Ranganath in 1998

[18] "Comparison of JPEG and JPEG 2000 in low-power confidential image transmission", by Mara Rhepp, winter 2004

[19] "Security through Obscurity", by Duraiswamy, IEEE, 1998

[20] "MEPG Video Encryption Algorithms", by Bharat Bhargava Changgui Shi Sheng-Yih Wang, IN 47907, USA August 13, 2002