International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Articles available online http://www.ijoer.in

Vol.2., Issue.6, 2014

RESEARCH ARTICLE

ISSN: 2321-7758

ANALYSIS ON CRYPTOGRAPHIC ALGORITHMS WITH STEGANOGRAPHY FOR SECURE TRANSFER OF DATA

K.DEVIKA RANI DHIVYA

Assistant Professor, Department of CA & SS, Sri Krishna Arts and Science, Coimbatore, Tamil Nadu, INDIA.

Article Received: 18/12/2014 | Article Revised on: 26/12/2014 | Article Accepted on:30/12/2014

K.DEVIKA RANI DHIVYA

ABSTRACT

Nowadays the transaction of data becomes insecure while transferring on the network they are relatively easy to be hacked by the intruders. To secure the data from intruders a concept called key generation is used to encrypt the data which cannot access by any third parties. Steganography is the practice of hiding data inside something which appears to be nothing out to the natural. It is an art and science of data hiding and unseen communication. The goal here is to secure communications over insecure network from an eavesdropper. There are different file formats where used for information hiding but mostly digital images are popular why because their frequency action on the internet will be high and its main purpose is to maintain secret communication between two or more users. Many cryptography algorithms are used to secure the information. This paper consists of analyzing the cryptographic algorithms with image steganography to improve the security. The analysis consists of RSA algorithm which uses Key concepts for encryption and decryption. Blowfish algorithm, it is a cipher-block algorithm that consist a 32-bit and it contain S-key. The main drawback in RSA is that the efficiency and in blowfish, it must get key to the person out of band specifically not through the unsecured transmission channel. The another algorithm call Tiny Encryption is used to Encrypt the text, it operates on a 64 bit block of data that is then split up into two 32 bit unsigned integers during the encryption process. TEA uses a 128 bit key. This paper shows the effectiveness of the algorithms for hiding data and to improve secure transfer over insecure network.

Key words: Steganography, cryptography, RSA algorithm, Blowfish algorithm and Tiny Encryption Algorithm.

INTRODUCTION

In today's world, Internet communication has become an important part. The information communicated plays a major role in many forms and it is used in many applications. In most of applications, it is most wanted that the communication to be make in secure manner by secret ways. Such secret communication may be of bank transfers, corporate communications, and credit card purchases, on down to a large percentage of everyday email. Nowadays E-mail plays a major role for information transfer thus many incorrectly assume that their communication is safe because it is just a small piece of a huge quantity of information being sent through WWW. Steganography is a attractive subject and it can be used for hidden communication. Encrypted data is difficult to decipher, but it is relatively easy to detect. Encryption only processes a message meaning into other meaning which cannot understand by other. Therefore, it is a technique that hides the existence of a message, if often used to supplement encryption. It's easy to use and works by replacing bits of data in computer files or communication channels. In steganography since message is hidden behind the cover and it is cover which is visible, when in encrypted form, thus providing security to the digital data. Encryption is used to keep the data out of reach of third party, so it is not altered and tampered by the third party. The video stegnography is software alter the originality of the file encrypted form embed the file into the video file. The key is designed in such a way that it prevents the unauthorized persons from stealing the information at any point of time. The system uses the tiny encryption algorithm to encrypt the password. The main goal is to secure the information from the hackers in the insecure network.

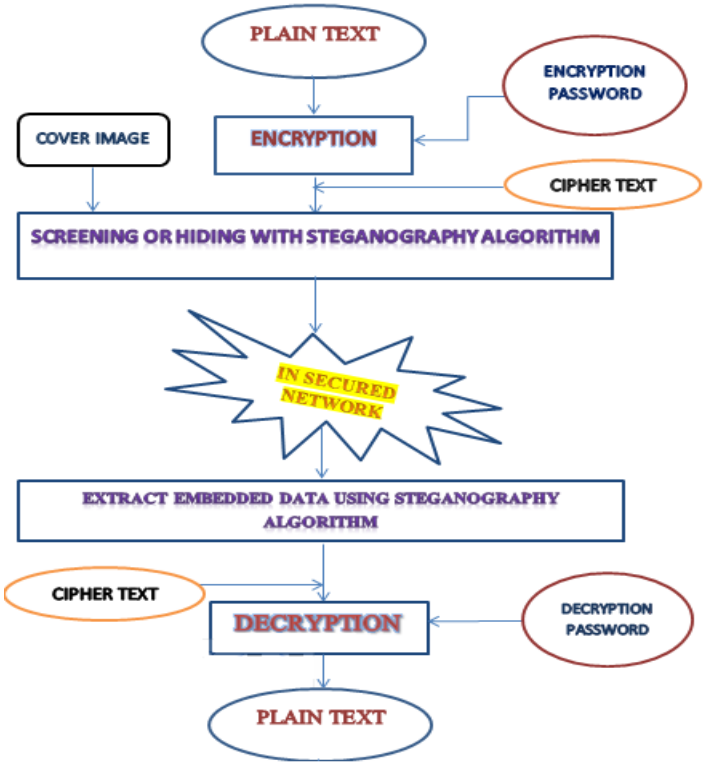

Fig:1 **Processing Flow to Send Secure Data using Steganography Algorithm. [4]**

1. **ANALYSIS ON CRYPTOGRAPHIC ALGORITHMS**

RSA Algorithm:

RSA is a cryptography algorithm that provides security with the help of two keys [1],

- a **public key** and
- a **private ke**

The public key can be known to everyone and is used for encrypting messages [6]. Messages encrypted with the public key can only be decrypted using the private key. The RSA scheme is a block

cipher in which the plaintext and cipher text are integers between 0 and n-1 for some n.A typical size for n is 1024 bits or 309 decimal digits. The RSA scheme has since that time reigned supreme as the most widely accepted and implemented general purpose to public key encryption. The scheme developed by Rivest, Shamir and Adleman makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n.That is, the block size must be less than or equal to $\log_2(n)$, in practice, the block size is k bits, where $2^k < n < 2^{k+1}$ .Encryption and decryption are of the following form, for some plaintext block M and cipher text block C:

$$C = M^e \bmod$$
$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know the value of n. The sender knows the value of e and only the receiver knows the value of d. Thus, this is a public key encryption algorithm with a public key of KU = {d,n}. For this algorithm to be satisfactory for public key encryption, the following requirements to be met:

It is possible to find the values of e, d, n such that $M^{ed} = M \bmod n$ for all M < n.

1. It is relatively easy to calculate M^e and C^d for all values of M < n.
2. It is infeasible to determine d given e and n.

Ready to state the RSA scheme the ingredients are the following:

P, q, two prime numbers (private, chosen)n = pq(public, calculated)

e, with gcd(ϕ(n),e) = 1; 1< e < ϕ(n) (public, chosen)

$d=e^{-1} \bmod \phi(n)$

(private, calculated)

Key generation

1. Choosing two Interger numbers:
   q = 11, **q** = 3.
2. **n** = pq = (11)(3) = 33
   calculate **f (n)** = (p-1) (q-1) = 20
3. Choose **e** = 3 and calculate **MCD(e, p-1)** = MCD(3,20) = 1 (3 and 10 do not have common factors apart from 1)
   calculate MCD(e, q-1) = MCD(3,2) = 1,

and **MCD(e, f(n))** = MCD(e, (p-1)(q-1)) = MCD(3,20) ≡ 1 (mod 20) = 1

Calcuate d so that **ed**=(mod f(n))

for example assume that d = e-1 mod f(n) = 3-1 mod 20

Therefor the public key is (n, e) = (33, 3) and

the private key is (n, d) = (33, 7).

Encryption - The encrypted text C is obtained from the equation 'C = n mod Me', where M is the original message.

Decryption - The M message can be recreated by decrypting the encrypted C text from the equation 'M = n mod Cd'.

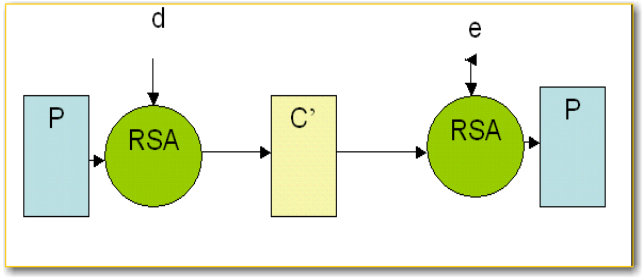

Fig: 2 **Process of RSA**

Blowfish Algorithm

Blowfish is a license-free cipher-block algorithm that propels a 32-bit, variable-length key to 448 bits. The original design was intended to replace the older and less-advanced data encryption standard (DES) by way of public domain access. Its basic functions utilize S-keys, which are key-dependent. Blowfish is a variable-length, a new secret-key block cipher. It is a Fiestal network, iterating a simple encryption function 16 times. Its main features are:

- Block cipher: 64-bit block.
- Variable key length: 32 bits to 488 bits.
- Much faster than IDEA and DES.
- Unpatented and royalty free.
- No license required.
- No license required.

Blowfish uses large subkeys The key is to be calculated before the encryption or decryption of data. Blowfish is a Feistel network algorithm implemented (Feistel Network), which consists of 16 rounds. Input is 64-bit elements, X.For groove Blowfish encryption algorithm with the method described below:

1. Form P-array initials as many as 18 units (P1, P2, P18 .....) Msing each 32-bit value. Array P consists of eighteen 32-bit key subkey:

P1, P2, ..., P18

2. Form S-box by 4 pieces each worth 32-bit with input 256.

Four 32-bit S-boxes each are having 256 entries:

S1, 0, S1, 1, ..., S1, 255
S2, 0, S2, 1, ..., S2, 255
S3, 0, S3, 1, ..., S3, 255
S4, 0, S4, 1, ..., S4, 255

3. Plaintext to be encrypted is assumed as input, Plaintext is taken as 64-bit, and if less than 64-bit then we add bitnya, so that in later operations in accordance with the data.

4. Results decision was divided by 2, the first 32-bit called XL, 32-bit the second is called XR.

5. Next do the operation xor XL = XL and XR Pi = F (XL) xor XR

6. The results of the above operrasi be exchanged XL XR and XR to XL.

7. Do it 16 times, the 16th iteration do another exchange process XL and XR.

8. In the process of doing surgery for 17 XR = XR xor P17 and P18 xor XL = XL.

9. The last process re-XL and XR unite to become 64-bit back.

Generating the Sub keys:

The sub keys are calculated using the Blowfish algorithm. The exact method is as follows:

1. Initialize first the P-array and then the four S-boxes, in order, with a fixed string. This string consists of the hexadecimal digits of pi (less the initial 3).

2.XOR P1 with the first 32 bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (possibly up to P14). Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits.

3. Encrypt the all-zero string with the Blowfish algorithm, using the sub keys described in steps (1) and (2).

4. Replace P1 and P2 with the output of step (3).

5. Encrypt the output of step (3) using the Blowfish algorithm with the modified sub keys.

6. Replace P3 and P4 with the output of step (5).

7. Continue the process, replacing all entries of the P- array, and then all four S-boxes, with the output of the continuously-changing Blowfish algorithm. In total, 521 iterations are required to generate all required sub keys.

These additional requirements should, if possible, be levied on a standard encryption algorithm.

1. It should be simple to code. If possible, the algorithm should be robust against implementation mistakes.
2. It should have a flat key space, allowing any random bit string of the required length to be a possible key. There should be no weak keys.
3. It should facilitate easy key-management for software implementations. In particular, the password that the user enters becomes the key.
4. It should be easily modifiable for different levels of security, both minimum and maximum requirements.

All operations should manipulate data in bite-sized blocks. Where possible, operations should manipulate data in 32-bit blocks.

PROBLEM DESCRIPTION

In Cryptography, the RSA problem summarizes the task of performing an RSA private-key operation given only the public key. The RSA algorithm raises a *message* to an *exponent*, modulo a composite number N whose factors are not known. As such, the task can be neatly described as finding the e^{th} roots of an arbitrary number, modulo N. For large RSA key sizes (in excess of 1024 bits), no efficient method for solving this problem is known; if an efficient method is ever developed, it would threaten the current or eventual security of RSA-based cryptosystems—both for public-key encryption and digital signatures. More specifically, the RSA problem is to efficiently compute P given an RSA public key (N, e) and a ciphertext $C \equiv P^e \pmod N$. The structure of the RSA public key requires that N be a large semiprime (i.e., a product of two large prime numbers), that $2 < e < N$, that e

be comprise to $\phi(N)$, and that $0 \leq C < N$. C is chosen randomly within that range; to specify the problem with complete precision, one must also specify how N and e are generated, which will depend on the precise means of RSA random keypair generation in use. The disadvantages of Blowfish are it must get key to the person out of band specifically not through the unsecured transmission channel. Each pair of users needs a unique, so as number of users increase, key management becomes complicated.

For example N(N-1)/2 keys required. Blowfish can't provide authentication and non-repudiation as two people have same key. It also has weakness in decryption process over other algorithms in terms of time consumption and serially in throughput.

Tiny Encryption Algorithm

The Tiny Encryption Algorithm (TEA) block cipher was designed with speed and simplicity in mind. It is a variant of the Feistel Cipher. [3] TEA operates on a 64 bit block of data that is then split up into two 32 bit unsigned integers during the encryption process. TEA uses a 128 bit key, and a magic constant is also utilized which is defined as 2^32/(the golden ratio). This quantity looks like 2654435769 when expressed as an integer. Multiples of this constant are used during each round, and its inclusion in the algorithm was to prevent attacks that try to take advantage of symmetry between rounds.

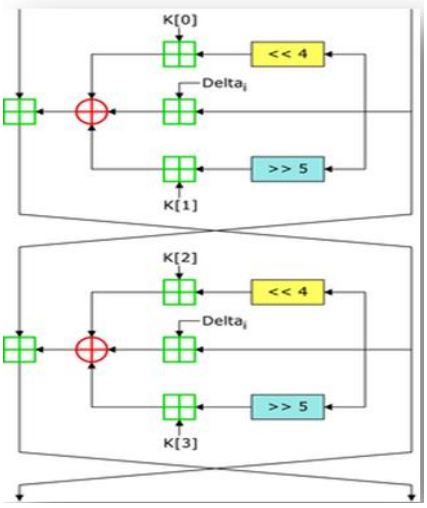

Fig.3. **Two Feistel Rounds of TEA** [14]

The diagram above shows 2 Feistel rounds of TEA. These two Feistal rounds make up one cycle of TEA. The cipher starts with a 64 bit data block that is split up into two 32 bit blocks which we will call L and R. L is the left side of the block (represented by the arrow in the top left of the diagram), and R is the right side (top right of the diagram). These blocks are swapped per round; This swap can be seen where the two lines intersect in the middle of the diagram. TEA has a 128 bit key that is split up into four 32 bit subkeys, which can be seen as K[0-3] in the diagram. Delta is defined as a constant, 2^32/(golden ratio), which is 2654435769 as an integer. Multiples of delta are used in each round (mod 2^32). In the first feistel round R is used as an input to several operations. All addition operations are (mod 2^32).

1. R goes through a left shift of 4 and then is added to K[0]
2. R is added to Delta
3. R goes through a right shift of 5 and then is added to K[1]

An XOR operation is then applied to the result of those three operations and finally, the result of the XOR operation is added to L. This result then becomes R for the next feistel round, because of the swap.

3. RESULT AND DISCUSSION

The present result is gained by comparing of the result obtained from the above steganography algorithms (RSA, Blowfish and TEA algorithms). Comparison with related work, we use JPEG images to test the present algorithm and to be sure that the aim of the data embedding is satisfied. In this work, it shows the efficiency of embedding data is very high when it is compared to the related work. We have been shown the comparison result of JPEG images. The result describes the level of efficiency according to the amount of hidden data.

The above figure gives the compares the amount of hidden data on selected images. This comparison shows that the TEA is efficiently secure and hide large amount of data and exceeds the capabilities of the related algorithms [RSA & Blowfish algorithm].

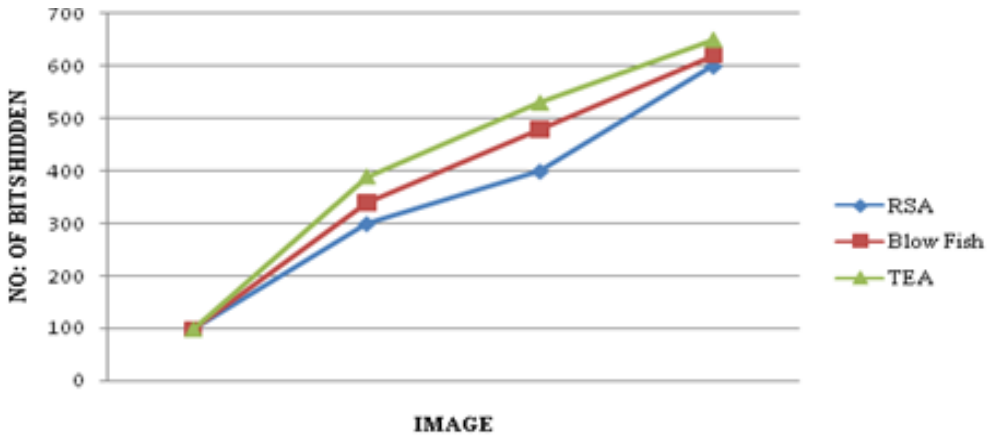

Fig. 4. **The amount of hidden data for bitmap images using the steganography algorithms**

CONCLUSION

In current world the data transfers becomes more important and must. For the purpose of secure message transfer many techniques. Image steganography become more popular, thus the proposed system proves best efficiency. Many types of images where used to prove that hidden information is more secure. The main intentions of the work are to analyze the various steganography algorithms and make a system using it to provide secure transfer of data. This approach provides more security to the hidden information from the eavesdroppers. In future, this process can enhance with some more effective cryptography algorithms with other image formats. This analysis concludes that these algorithms are used for data security in that the TEA provides high data security with effectiveness.

REFERENCE

[1]. Ajtai .M and C. Dwork, A public-key cryptosystem with worst-case/average-case equivalence, *Proc. 29th ACM STOC* (1997), 284-297.

[2]. Benjamin Andrews, Scott Chapman and Steven Dearstyne "Tiny Encryption Algorithm (TEA) Cryptography 4005.705.01" Graduate Team ACD - Final Report.

[3]. Chandramouli, R. And Memon. N , 2001. Analysis of LSB based image steganography techniques. Proc. Of ICIP, Thessaloniki, Greece.

[4]. Devika Rani Dhivya . K, "Comparison of Novel Network Security Algorithms for Securing and Screening Data Using Steganography", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 3, Issue 11, November 2013.

[5]. Devika Rani Dhivya . K and C. Gomathi, "Secure Data Transfer Through Insecure Network By Hiding Data Using Image Steganography", International Journal of Computer Application" ISSN: 2250-1797 , Volume 4, Issue 4, July-August 2014.

[6]. Lisa M. Marvel, Charles G. Boncelet, Charles T. Retter, "Spread Spectrum Image Steganography", in IEEE trans. on Image Processing, Vol.8, No.8, August 1999, pp.1075-1083.

[7]. Nameer N. EL-Emam ,"Hiding a Large Amount of Data with High Security Using Steganography Algorithm", Journal of Computer Science 3 (4): 223-232, 2007 SSN 1549-3636 2007 Science Publications

[8]. Neil F. Johnson, Sushil Jajodia, *"Exploring Steganography: Seeing the Unseen", IEEE Computer, 1998, pp.26-34*

[9]. Stefan Hetzl, *"A Survey of Steganography", January 2002*

[10]. Tu Ran, *"Steganography: The Art of Hiding Data", Spring 2002*

[11]. W. Bender, D. Gruhl, N. Morimoto, A. Lu, *"Techniques for Data Hiding", IBM Systems Journal, vol. 35, nos.3&4, 1996, pp.313-335*.

[12]. J.R.Smith, B.O.Comisky, *"Modulation and Information Hiding in Images", Proceedings of the First Information Hiding Workshop,*

Isaac Newton Institute, Cambridge, U.K., May 1996. Springer-Verlag Lecture Notes in Computer Science Volume 1174, pp.207-226

[13]. Lisa M. Marvel, Charles T. Retter, *"The Use of Side Information in Image Steganography", International Symposium on Information Theory and Its Applications, Honolulu, Hawaii, USA, November 5-8, 2000.*

[14]. https://upload.wikimedia.org/wikipedia/commons/a/a1/TEA_InfoBox_Diagram.png