INTERNET SECURITY ENHANCEMENT THROUGH MASSAGE / IMAGE ENCRYPTION: DES IN COMPARISON WITH RSA METHODS

Gambo D.[1], Ibrahim B.S.[2], Mahmud M.[3] and Lawal J.[4]

[1,2,3,] Department of Electrical Electronics Engineering Technology,
Nuhu Bamalli Polytechnic, Zaria
danasabegambo@gmail.com

[4]Deparment of statistics,
Nuhu Bamalli Polytechnic, Zaria

Abstract
This paper investigates two types of encryptions algorithms in order to analyse massage encryption procedure using DES (Data Encryption Standard) as well as that of RSA (Rivest-Shamir-Adleman). It is intended to explain how DES works and also compare it with RSA. This is done by investigating the background of DES, looking and the implementation of the DES through MATLAB that includes image Encryption and decryption, and an introduction to how the RSA algorithm functions. Then with the information obtained the paper evaluated how to encrypt massage to be send via internet with maximum protection. Comparison was made by summarizing the advantages and disadvantages between the two encryption methods.

Key words: *Encryption, DES, RSA, Cryptography and MATLAB*

1. Introduction

To provide the security to the data transmitted over different networks using different services, many encryption methods are used. This can be achieved using several methods, such as the DES and RSA cryptography based algorithms. These different algorithms can function in completely different manners. To gain some understanding of how communicated data can be secured we can compare the DES (Data Encryption Standard) and RSA (stands for the creators of the technique, Rivest, Shamir and Adelman) algorithms. Before understanding how these algorithm works, there is the need of understanding the definitions of Cryptography and Encryption given by various scholars:

1.1 Cryptography:

According to (Parr & Pelzl, 2010), "Cryptography is the science of secret writing with the goal of hiding the meaning of a message, which can only be known by the intended recipient". This is to say that cryptography is the means of achieving requirements of communication which are Authentication, Privacy, Integrity, and No n-repudiation.

1.2 Encryption: This is the process of transforming an intelligible message

(called the plaintext) into a representation (called the cipher text) that cannot be understood by unauthorized parties. (Eskicioglu & Delp, 2006)

1.3 Three main types of Ciphering:

Symmetric Ciphers (Secret Key), Asymmetric (Public-key) Ciphers (Public key and Private Key), and Cryptographic Protocols/Hash Functions (The application of algorithms).

1.4 Email and Privacy

Almost every time we go online. For everything from banking and shopping to checking email, we like our Internet transactions to be well-protected, and encryption helps make that possible, such that, only the intended recipient will be able to decipher the message while anybody else sees but gibberish.

2.0 Data Encryption Standard (DES)

DES is one of the most widely accepted, publicly available cryptographic systems and simply define as the systematic block cipher that uses a 56bit-key, 64bit-input block, and a 64-bit output block. It was developed by IBM in the 1970s but was later adopted by the National Institute of Standards and Technology (NIST), as Federal Information Processing Standard 46 (FIPS PUB 46). The Data Encryption Standard (DES) is a block Cipher which is designed to encrypt and decrypt blocks of data consisting of a 64 bit input anda 64-bit output (Mandal, et al., 2012). As in *Figure 1* below;

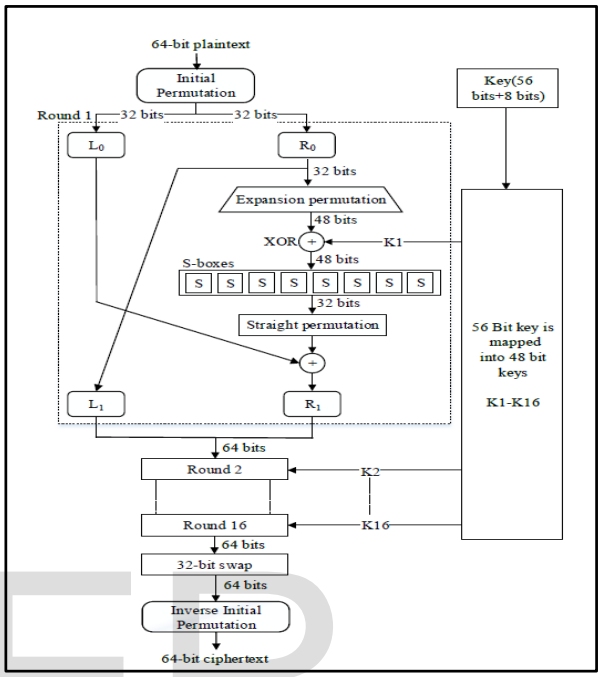

Figure 1- General Depiction of DES (Mandal, et al., 2012)

Encryption of a block of the message takes place in 16 stages or rounds. From the input key, sixteen 48 bit keys are generated, one for each round. In each round, eight so-called S-boxes are used. These S-boxes are fixed in the specification of the standard. Using the S-boxes, groups of six bits are mapped to groups of four bits. The contents of these S-boxes have been determined by the U.S. National Security Agency (NSA). The S-boxes appear to be randomly filled, but this is not the case. Recently it has been discovered that these S-boxes, determined

in the 1970s, are resistant against an attack called differential cryptanalysis which was first known in the 1990s.The block of the message is divided into two halves. The right half is expanded from 32 to 48 bits using another fixed table. The result is combined with the sub key for that round using the XOR operation. Using the S-boxes the 48 resulting bits are then transformed again to 32 bits, which are subsequently permutated again using yet another fixed table. This by now thoroughly shuffled right half is now combined with the left half using the XOR operation. In the next round, this combination is used as the new left half.

2.1 DES Implementation In MATLAB

2.1 Introduction to Implementation

The first step with DES encryption is to ensure that the input data is the right size for the encryption, which is 64-bits of information and a 56-bit key. As DES is a form of symmetric encryption it uses Block ciphering, which is one of the two types most symmetric algorithms use (Stream cipher or Block cipher), as can be seen in *Figure2*.

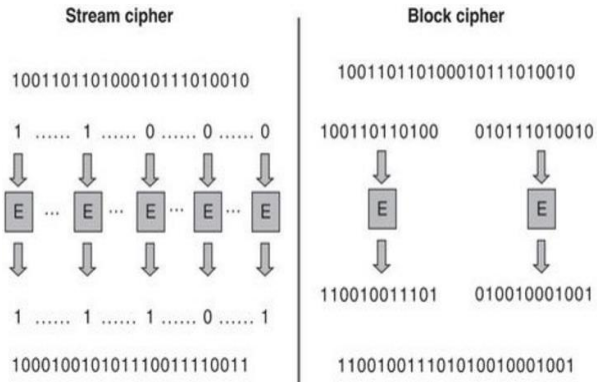

Figure 2 – Symmetric encryption Cipher types (Martin, 2012)

Therefore the first step of implementing DES is to convert the inputs, which for this implementation will be ASII-text or Images, to the correct format of data.

2.2 DES - ASCII Text Implementation

For ASCII text implementation the following steps need to be taken, in Matlab to acquire the correct data format for the DES function:

- Read the text from the Text file using the functions available in Matlab.
- Convert into binary representation, so that the ASCII characters become a multi-bit word.
- If required, binary representation is not the correct bit length per character, use "Bit-Padding" to extend word lengths, as seen in *Figure 3.*

In the case of this implementation the correct word length is 8, so that the total number of bits is divisible by 64. These padded bits will be zeros and will be removed after encrypting/decrypting.

```
elseif ((40 < Length)&&(Length <= 48))              % For sending messages of 64bits
    SendLen=48;
    Repeat=6;
end

ASCMessageIn =zeros(1,(SendLen*8));     % Create array full of 0's
OutMessage =zeros(1,(SendLen));     % Create array full of 0's
```

Figure 3 – Bit Padding for ASCII

After bit-padding, the encryption function then alters the input data's bits so that they are in a completely different arrangement. This is done in several stages (Grabbe, 2006):

i. Initial Permutation, which completely rearranges the order of the data.

ii. Splitting this data of 64-bits into two sets of 32-bits, called the "Left and Right" blocks (R_0 and L_0).

iii. Expanding the Right-block to 48-bits by bit-padding, reusing pre-selected bits from the block.

[Permutation: When data is rearranged into a different order, also known as Transposition ciphering]

```
% 3.1 initial permutation
C = IP(P);
switch mode
    case 'ENC' % if encryption, split 64 message to two
        L{1} = HALF_L(C); % left-half 32-bit
        R{1} = HALF_R(C); % right-half 32-bit
    case 'DEC' % if decryption, swapping two halves
        L{1} = HALF_R(C);
        R{1} = HALF_L(C);
end
```

Figure 4 – Splitting of input data from 64-bits to 32-bits

DES requires a 56-bit key to be made;*Figure 5* shows how the code is implemented. This code uses a 56-bit key, but sometimes DES has a 64-bit key. These 64-bit keys are reduced to 56 normally by not using every 8th bit of the key. (This implementation uses a key made from permuting the input data)

```
% 2.1 define permuted choice 1 (PC1)
PC1L = @(key64) key64([57   49  41  33  25  17  9 ...
                       1   58  50  42  34  26  18 ...
                       10  2   59  51  43  35  27 ...
                       19  11  3   60  52  44  36]);
PC1R = @(key64) key64([63   55  47  39  31  23  15 ...
                       7   62  54  46  38  30  22 ...
                       14  6   61  53  45  37  29 ...
                       21  13  5   28  20  12  4]);
```

Figure 5 – Defining the Key for the Encryption

Then sub-keys are derivedfrom the 56-bit key, but they are only 48-bits long, as can be seen in *Figure 6*. Normally DES performs 16 rounds of encryption with a different sub-keys made per round.

```
% 2.2 define permuted choice 2 (PC2)
PC2 = @(key56) key56([14 17 11  24  1   5   3   28 ...
                      15 6   21  10  23  19  12  4 ...
                      26 8   16  7   27  20  13  2 ...
                      41 52  31  37  47  55  30  40 ...
                      51 45  33  48  44  49  39  56 ...
                      34 53  46  42  50  36  29  32]);
```

Figure 6 – Defining Sub-key for Encryption

Then two functions are carried out on the input (block) data:

i. The sub-keys are compared to the Right-block and an XOR (exclusive or) function is used.

ii. Substitution of the data (in this case 48-bits to 32-bits):

iii. Evaluate/ split data into 6-bit blocks.

iv. Create substitution table, as can be seen in *Figure 7*, with 16 columns and 4 rows based on size of substitution blocks (4-bit and 2-bit).

v. Substitute 6-bit block for 4-bit block using method from *Equations 1* and *2*

```
% 1.4 define eight substitution tables
% input: 0  1   2   3   4   5   6   7   8   9   10  11  12  13  14  15
st{1} = [14 4   13  1   2   15  11  8   3   10  6   12  5   9   0   7;...
         0  15  7   4   14  2   13  1   10  6   12  11  9   5   3   8;...
         4  1   14  8   13  6   2   11  15  12  9   7   3   10  5   0;...
         15 12  8   2   4   9   1   7   5   11  3   14  10  0   6   13];
```

Figure 7 – Substitution Table 1

$$input = U = 101110,\ row = first\ \&\ last\ bit = 10,\ column = middle\ 4\ bits = 0111 \qquad \text{Eq. 1}$$

$$\therefore\ 4bit\ representation\ equals\ the\ value\ in\ row(2)\ column(7) = "11" = 1011 \qquad \text{Eq. 2}$$

After substitution there is another permutation of the data. This data, originally from the Right-block, is then compared using XOR with the Left-block. Finally the original Right-block data is combined with this data to make a 64-bit array output, as shown in *Figure 8*. The order of using Left and Right blocks is altered depending on if decryption or encryption is occurring (Wu, 2012).

```
% 3.3 final permutation
switch mode
    case 'ENC'
        C = [L{end},R{end}];
    case 'DEC'
        C = [R{end},L{end}];
end
```

Figure 8 – Final Permutation

Once all 16 rounds have been performed there is a final permutation to reverse the initial permutation at the start of the encryption. After the message has been encrypted and decrypted the final part of the coding is to revert the output of the DES encryption back to ASCII format. To do this simply do the invert all the padding and convert back to a decimal value that Matlab can use to write to an ASCII text file.

2.3 Image Implementation

For Image encryption using the DES algorithm code the following steps must be taken:

i. Convert the image into a numerical RGB (numeric) format.
ii. Change to greyscale values, shown in *Figure 9*, which is capable of being altered into 64-bits.
iii. Convert into binary 8-bit values, which each represent an individual pixel of the image.

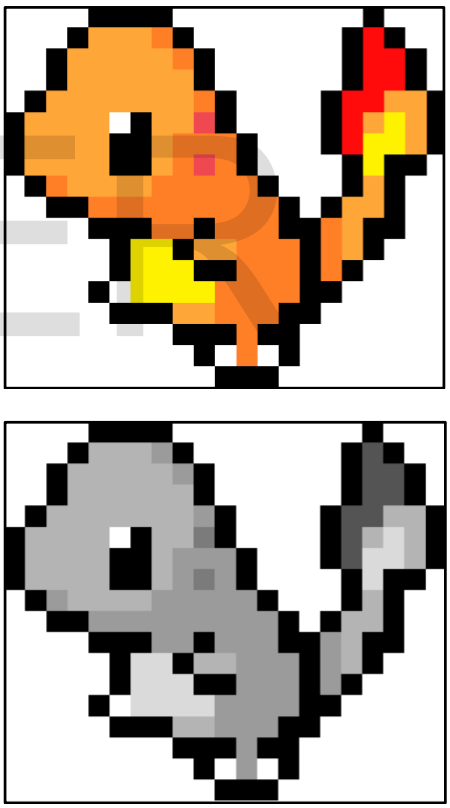

Figure 9 – RGB to Greyscale Conversion

[It is important to note that each pixel of the image requires a lot of steps and conversions. This means that time taken to encrypt the image increases greatly with the pixel ratio of the image]

Like the ASCII encryption, to return the data back to an image simply convert the binary data output from the DES algorithm code into uint8 data and the data into a matrix with data in the same order as it was input. This also allows the ciphered image to be visualized, given in *Figure 10*.

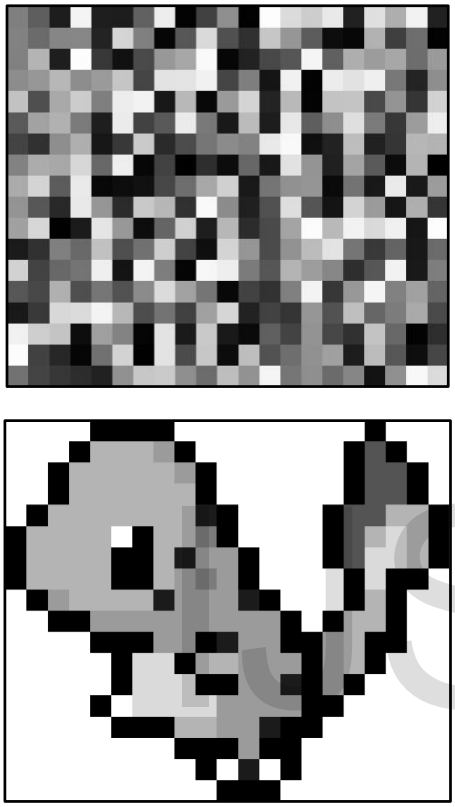

Figure 10 Encrypted Image to Decrypted Image Conversion

Observing the differences between the encrypted and decrypted, *Figures 9* and *10*, show there are several pixel errors. These errors are increase when there are a greater variety of values (colours) in the image data. Therefore, it is believed that the errors are caused by the DES algorithm not decrypting some values back correctly.

3.0 RSA (Rivest, Shamir and Adelman)

RSA is designed by Ron Rivest, Adi Shamir, and Leonard Adleman in 1978. It is one of the best known public key cryptosystems for key exchange or digital signatures or encryption of blocks of data. RSA uses a variable size encryption block and a variable size key (Singh & Supriya, 2013). A message can be encrypted by representing it as a number M, raising M to a specified power e, and then taking the remainder when the result is divided by the publicly specified product, n, of two large secret primer numbers p and q. Decryption is similar; only a different, secret, power d is used, where e * d ≡ 1(mod (p - 1) * (q - 1)). The security of the system rests in part on the difficulty of factoring the published divisor, (Rivest, et al., 1978) as show in *Figure 11*.

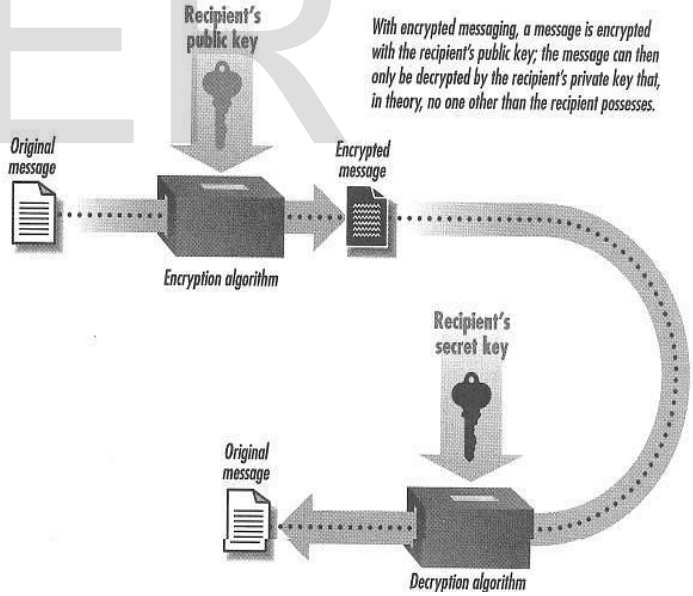

Figure 11 Typical example on how RSA algorithm works **(Marcos, 2016)**

RSA operations can be decomposed in three broad steps; key generation, encryption and decryption as further explained below (Zhou & Tang, 2011)

i. I would like to receive encrypted messages from everyone
ii. I obtain a key and a decoder
iii. I published my key and kept the decoder secret
iv. Everybody can use my key to encrypt their message to me
v. I am the only one who can decrypt the message using my decoder
vi. The key is known as ***public key*** and the decoder is known as ***private key.***

3.1 Numerical Operation of RSA

Public key can be used to encrypt the message and send, the receiver is to use the private key to decode the message, below is the numerical example:

Generate two large random prime numbers, p & q

Find $n = (p.q)$

Find $phi = \varphi(n) = (p-1).(q-1)$

Choose an integer e, $1 < e < phi$, such that $(e \times d) mod phi = 1$

The public key is (n, e) and the private key (d, n)

All the values d, p, q and phi are kept secret.

Where:

n is referred to as the modulus

e is referred to as the public key

d is referred to as the secret exponent

Example

Encryption of message from X to Y, using the following steps:

Obtains Y's public key (n, e)

Represents the plaintext massage as a positive integer M, such that $1 < M < n$,

Computes the cipher text $C = M^e \, mod \, n$

Send the cipher text $C \; to \; Y$

Decryption of message from X by Y, using the following steps:

Uses the private key (n, d) to compute $M = C^d \, mod \, n$

3.2. Daily Life Application of RSA

When buying something from e-bay, they send their public key to your browser, your information get encrypted using e-bay's public key and sent to them. They use their private key (i.e. decoder) to decrypt the encrypted data

3.3 DES and RSA Comparison

The table below shows the brief comparison between RSA and DES algorithms, in terms of Creators, number of rounds, Key length, Block Size, Cipher type, Speed and Security, it's also shows that Asymmetric Algorithms such as RSA is slower than that of Symmetric Algorithms and is least secure algorithm as compared to DES.

Table 1 - Comparison between RSA and DES

Factor	RSA	DES
Created By	Ron Rivest, Adi Shamir, and Leonard Adleman In 1978	IBM in 1975
Round(s)	1	16

Key Length	Depends on number of bits in the modulus n where n=p*q	56 bits
Block Size	Variable	64 bits
Cipher Type	Asymmetric Block Cipher	Symmetric Block Cipher
Speed	Slowest	Slow
Security	Least Secure	Not Secure Enough

According to research done and literature survey it can be found that RSA algorithm is least secure compare with DES and it's block size variability is also another advantage over DES. Furthermore, for RSA to be more secured the length of P & Q has to be large, which increases its processing time and the performance gets degraded in comparison with DES (Kakkar, et al., 2012).

This secret key encryption algorithm uses a key that is 56 bits, or seven characters long. At the time it was believed that trying out all 72,057,594,037,927,936 possible keys (a seven with 16 zeros) would be impossible because computers could not possibly ever become fast enough. In 1998 the Electronic Frontier Foundation (EFF) built a special-purpose machine that could decrypt a message by trying out all possible keys in less than three days. The machine cost less than $250,000 and searched over 88 billion keys per second (IUS, 2016).

The DES algorithm is repeated 16 times to produce the ciphertext. It has been found that the number of rounds is exponentially proportional to the amount of time required to find a key. So, as the number of rounds increases, the security of the algorithm increases exponentially, while that of RSA is only one way.

4.0 Conclusion and Recommendation

This paper presents a detailed study of the popular Encryption Algorithms such as RSA & DES. The use of internet and network is growing rapidly,so there are more requirements to secure the data transmitted over different networks using different services. To provide the security to the network and data different encryption methods are used. In this paper, a survey on the existing works on the Encryption techniques has been done. To sum up, all the techniques are useful for real-time Encryption. Each technique is unique in its own way, which might be suitable for different applications and has its own pro's and con's. The Security provided by these algorithms can be enhanced further, if more than one algorithm is applied to data.

References

Eskicioglu, A. & Delp, E., 2006. Chapter 1. Protection of Multimedia Content in Distribution Networks. In: B. Furht & D. Kirovski, eds. *Multimedia Encryption and Authentication Techniques and Applications.* s.l.:Auerbach Publications, pp. 1-60.

Grabbe, J., 2006. *The DES Algrithm Illustrated.* [Online] Available at: http://page.math.tuberlin.de/~kant/teaching/hess/krypto-ws2006/des.htm
[Accessed 29th March 2016].

IUS, 2016. *The DES Encryption Algorithm.* [Online] Available at: http://www.iusmentis.com/technology/encryption/des/[Accessed 1 April 2016].

Kakkar, A., Singh, M. & Bansal, P., 2012. Comparison of Various Encryption Algorithms and Techniques for Secured Data Communication in Multinode Network. *International Journal of Engineering and Technology,* 2(1), pp. 87-92.

Kessler, G., 2016. *An Overview of Cryptography.* [Online] Available at: http://www.garykessler.net/library/crypto.html
[Accessed 14 April 2016].

Mandal, A. K., Parakash, C. & Tiwari, A., 2012. Performance Evaluation of Cryptographic Algorithms: DES and AES. *IEEE Students' Conference on Electrical, Electronics and Computer Science,* p. 1 to 5.

Marcos, R., 2016. *Network Applications Lecture Notes,* Sheffield: Sheffield Hallam University.

Martin, K. M., 2012. *Everyday Cryptography: Fundamental Principles and applications.* 1st ed. Oxford: Oxford University Press.

Parr, C. & Pelzl, J., 2010. *Understanding cryptography : a textbook for students and practitioners.* 1st ed. Berlin: Springer.

Rivest, R., Shamir, A. & Adleman, L., 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM,* 21(2), pp. 120-126.

Rodriguez-Clark, D., 2013. *Permutation Cipher.* [Online] Available at: http://crypto.interactive maths.com/permutation-cipher.html#intro[Accessed 29 March 2016].

Singh, G. & Supriya, 2013. A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security. *International Journal of Computer Applications* , 67(19).

Singh, S., Maakar, S. & Kumar, S., 2013. A Performance Analysis of DES and RSA Cryptography. *International Journal of Emerging Trends & Technology in Computer Science,* 2(3), p. 418.

Wu, D. Y., 2012. *DES: Data Encryption Standard.* [Online] Available at: http://www.mathworks.com/matlabcentral/fileexchange/37847-data-encryption-standard--des-[Accessed 20 02 2016].

Zhou, X. & Tang, X., 2011. *Research and Implementation of RSA Algorithm for Encryption and Decryption.* s.l., 6th International Forum on Strategic Technology.