International Journal of Computer Science
and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 3, Issue 4, July 2014, 161-174

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

POWERFUL KEY POOL FOR SYMMETRIC ENCRYPTION

KARUNA GARG[1], ANKUSH GOYAL[2] & PRERNA MITTAL[3]

[1]M.Tech. Student, Department of CSE, SRCEM, Palwal, Haryana, India

[2]Assistant Professor, Department of CSE, SRCEM, Palwal, Haryana, India

[3]M.Tech. Student, Department of CSE, BSAITM, Faridabad, Haryana, India

ABSTRACT

This introduces a new concept of the generation of an unending pool of keys through pre distribution of multimedia files leaving behind the idea of sending keys every time for encryption and decryption. This can help in avoiding the problem of frequent key exchanges and the after affects associated with it. An already saved file is used to generate any size key and thus can be used for any algorithm. This adds the advantage of one time usage of key and avoids the disadvantage of securing and sending it on the network. It also allows the user to use more than one key for the encryption as it does not have an overhead of sending the keys on the internet. N keys can be used for N rounds of encryption or M keys can be used for M blocks of data for encryption. It gives an extra edge of security on the data with the existing algorithms

KEYWORDS: Data Encryption Standard, Encryption Algorithms, Key Pool, Powerful Key, Pre Distributed Keys, Weak Key

INTRODUCTION

Encryption [3] is a technique of conversion of data into such a code which is only read by the intended receiver and not by any other cryptanalyst [2]. For this purpose sender locks the data with a key and the data is converted into a cipher text. Decryption is the process reverse to the encryption which is used so that the encrypted text /cipher text is converted into a readable text. This process is at the receiver's end.

DES (Data Encryption Standard): Decryption process using two keys K1 and K2 DES is a block cipher. It encrypts the data in a block of 64 bits. It produces 64 bit cipher text. The key length is 56 bits. Initially the key is consisting of 64 bits. The bit position 8, 16, 24, 32,40,48,56, 64 discarded from the key length [16].

Double DES: It is also called 2DES. It's process is the same as DES but repeated same process 2 times using two keys K1 and K2. First it takes plain text, produced the cipher text using K1 and then take up the cipher text as input, produced another cipher text using K2 shown in figure 1. The Decryption Process is shown in figure 2 [10].

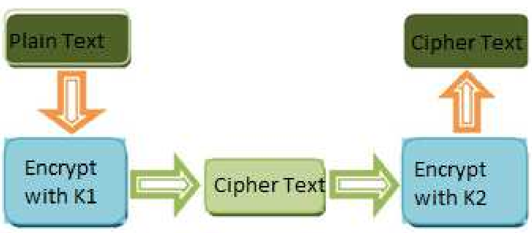

Figure 1: Encryption Process Using Two Keys K1 and K2

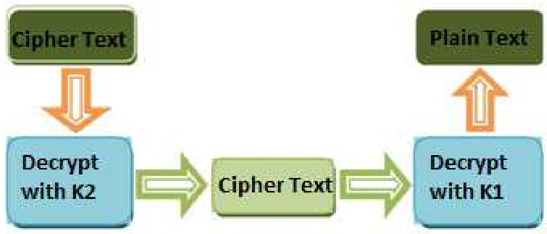

Figure 2: Decryption Process Using Two Keys K1 and K2

Triple DES: Triple DES is DES -three times. It comes in two flavours: One that uses three keys, and other that uses two keys. The Idea of 3-DES is shown in to the figure 4. The plain text block P is first encrypted with a key K1, then encrypted with second key K2, and finally with third key K3, where K1, K2 and K3 are different from each other. To decrypt the cipher text C and obtain the plain text, we need to perform the operation P= DK3 (DK2 (DK1(C))). But in Triple DES with two keys the algorithms works as follows: [1] Encryption the plain text with key K1. Thus, we have EK1 (p). [2] Decrypt the output of step1 above with key K2. Thus, we have DK2 (EK1 (P)). [3] Finally, encrypt the output of step 2 again with key K1.Thus, we have EK1 (DK2 (EK1 (P))). The idea of Triple DES with two keys is stronger than DES.

AES: It is specified in FIPS 197 [6]. It has three approved key sizes: 128, 192 and 256 bits. AES-128 is assessed at security strength of 128 bits, AES 192 at security strength of 192 bits, and AES-256 at security strength of 256 bits.

Algorithm Used for DES is

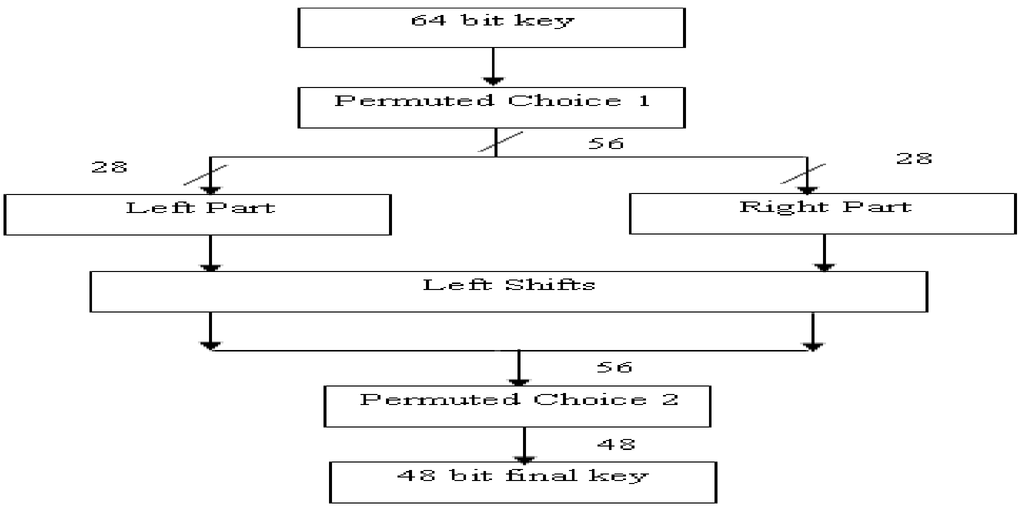

Figure 3: DES Key Generation Algorithm

BACKGROUND WORK

Matrix Based Key Generation [13]

The proposed key generation procedure offers two advantages. First, the procedure is simple to implement and has complexity in determining the sub-keys through crypt analysis. Secondly, the procedure produces a strong avalanche effect making many bits in the output block of a cipher to undergo changes with one bit change in the secret key. As a case study, matrix based key generation procedure has been introduced in Advanced Encryption Standard (AES) by replacing the existing key schedule of AES. The key avalanche and differential key propagation produced in AES have been observed. The paper describes the matrix based key generation procedure and the enhanced key avalanche and differential key propagation produced in AES. It has been shown that, the key avalanche effect and differential key propagation characteristics of AES have improved by replacing the AES key schedule with the Matrix based key generation procedure.

Key Generation Procedure

The key Generation (key scheduling) procedure is based on a matrix initialized using secret key. The values of sub-keys used in various diffusion rounds are taken from selected rows and columns of this matrix. The selection of rows and columns for this purpose is based on the secret key value and other functional logic as explained in the following sub sections.

Nomenclature

M[i][j]–Element of matrix M with row i and column j

K(i) – i

th character of secret key, K

Ks1(r), Ks2(r) – sub-keys used in rth round

P – Plaintext, C–Cipher text,

K – change in secret key value

Matrix Initialization

A matrix M with 16 rows and 256 columns is defined. Each column of every row is filled with a number between 0 and 255 (both the numbers included) in an order depending on the characters of secret key. The first column in the ith row of the matrix is filled with ASCII code of ith character of the secret key, K (that is, M[i][1] = Integer value of K[i]). The subsequent columns of the ith row of the matrix are filled with numbers that have increments of 1 from the previous column value till the number is 255. Subsequent columns are filled with numbers starting from 0 and ending with ASCII code of the ith character of secret key minus 1. The distribution of characters in the columns of all the sixteen rows of the matrix thus becomes key dependent. Without knowing the secret key the element in a column of any row of the matrix M cannot be determined by an adversary. Plate 1: shows the matrix initialization pseudo code.

```
For i 0 to 15 // rows
For j  0 to 255 // columns
M[i][j] = (int)K[i] + j
If M[i][j] > 255 { M[i][j] = M[i][j] – 256 }
EndFor // columns
EndFor // rows
```

Plate 1. Matrix initialization pseudo code

Sub Key Generation

Sub-keys used in round operations are generated by key scheduling procedure. In this procedure two sub-key matrices Ks1 andKs2 (of size 16 * 16) are derived from the base matrix M. These pairs of key can be used in substitution and diffusion operations performed in a typical block cipher. It is desirable that the key scheduling be a complex procedure so that an adversary must find it extremely difficult to derive the sub-keys during crypt analysis. Another desirable feature of key schedule is that a small change in the secret key should get well diffused in to the sub-keys. This means that one bit change in secret key should cause many bits to change in sub-keys. These two desirable features are considered while designing the key scheduling procedure. The procedure is explained in steps as follows:

- Secret key, K, is transposed (T) to get K1. It is a byte-level transposing operation performed in this process whereby the LS byte takes the place of MS byte position and the MS byte takes the LS byte position after the transpose operation. For example, if, bytes in array, K, is {K0,K1,K2,K3,K4,,….K14,K15} then after performing the transpose operation, K1 = K Transposed, the contents of K1 will become {K15, K14,….K5,K4,K3,K2,K1,K0}.
  - K1 is XORed with K to get K2. This operation can cause up to 2 bits to change in K2 when 1 bit is changed in secret key K.
  - Left half of K2 and right half of K2 is XORed to get K3.
  - Transposed left half of K2 and transposed right half of K2 are XORed to get K4.
  - K3 and K4 are concatenated to get K5. The operation of 1 bit change in secret key K can cause up to 4 bits to change in K5.
  - Sum of integer values of bytes in K5 is calculated to get L.
  - Kse1 is calculated such that Kse1 = L % 23. When secret key has 1 bit change, Kse1 can have up to 4 counts change.
  - Kse2 is calculated such that Kse2 = L % 15.
  - When secret key has 1 bit change, Kse2 can have up to 4 counts change. (Kse1 + Kse2) can have up to 8 counts change with one bit change in secret key.

Steps 1 through 8 in the key scheduling procedure are shown in figure 1.

Two matrices Ks1 and Ks2 of size 16 x 16 are derived from the base matrix, M, such that

Ks1[row][column]=M[row][Kse1+Kse2+column]

Ks2[row][column]=M[row][Ks1[row][column]]

Columns of Ks1 matrix are chosen from the base matrix M depending upon Kse1 and

Kse 2 values. Here, an element of Ks1 can have up to 8 counts change with one bit change in secret key. Columns of Ks2 matrix are chosen from the base matrix depending upon element values of columns of Ks1 matrix. An element of Ks2 can have up to 8 counts change with one bit change in secret key.

- Ks1[row][column]=M[row][Ks2[row][column]] columns of Ks1 matrix are chosen from the base matrix M depending upon element values in columns of Ks2 matrix. The regeneration of sub-key matrix, Ks1, is carried out in order to achieve further indirection for adding complexity.
- Rotate vertically down ith column of matrix Ks1 number of times equal to ((int(K[i]) % 12) + Kse1).
- Rotate vertically down ith column of matrix Ks2 number of times equal to ((int(K[i]) % 10) + Kse1). The vertical rotations shuffle the elements of sub-key matrices thereby providing more changes in the sub-key values while one bit change is applied on the original secret key, K.This procedure facilitates many bits to change in the sub-keys due to one bit change in the secret key. This is a desirable feature of any key scheduling procedure that can produce high diffusion and hence enhances the security of the cipher. The sub-keys, Ks1 and Ks2, for round operations (round 1: through round 10:), generated from a given secret key, K are shown in plate. 2 and plate. 3. The ten values shown under the heading key schedule represents the value of sub-keys (in hex format) to be used in ten rounds.

Secret Key, K: 4C 69 66 65 27 73 20 62 65 61 75 74 69 66 75 6C

Key Schedule (Ks1)

- 6D 6A 49 40 6D 52 F2 49 40 3D 83 67 6E 34 3D 31
- 68 6F 35 3E 32 6E 6B 4A 41 6E 53 F3 4A 41 3E 84
- 3F 33 6F 6C 4B 42 6F 54 F4 4B 42 3F 85 69 70 36
- 34 70 6D 4C 43 70 55 F5 4C 43 40 86 6A 71 37 40
- 6E 4D 44 71 56 F6 4D 44 41 87 6B 72 38 41 35 71
- 39 42 36 72 6F 4E 45 72 57 F7 4E 45 42 88 6C 73
- 74 3A 43 37 73 70 4F 46 73 58 F8 4F 46 43 89 6D
- 50 47 74 59 F9 50 47 44 8A 6E 75 3B 44 38 74 71
- 39 75 72 51 48 75 5A FA 51 48 45 8B 6F 76 3C 45
- 49 76 5B FB 52 49 46 8C 70 77 3D 46 3A 76 73 52

Plate 2: Secret Key K and Sub-Key ks1 for 10 Rounds

Secret Key, K: 4C 69 66 65 27 73 20 62 65 61 75 74 69 66 75 6C

Key Schedule (Ks2)

- D8 D0 F8 F6 E0 DA F8 E6 A6 E0 DA D8 5C F4 4E D2
- D1 F9 F7 E1 DB F9 E7 A7 E1 DB D9 5D F5 4F D3 D9
- E2 DC FA E8 A8 E2 DC DA 5E F6 50 D4 DA D2 FA 8
- DD FB E9 A9 E3 DD DB 5F F7 51 D5 DB D3 FB F9 E3

- F8 52 D6 DC D4 FC FA E4 DE FC EA AA E4 DE DC 60
- D5 FD FB E5 DF FD EB AB E5 DF DD 61 F9 53 D7 DD
- E6 E0 FE EC AC E6 E0 DE 62 FA 54 D8 DE D6 FE FC
- 55 D9 DF D7 FF FD E7 E1 FF ED AD E7 E1 DF 63 FB
- E2 00 EE AE E8 E2 E0 64 FC 56 DA E0 D8 00 FE E8
- DB E1 D9 01 FF E9 E3 01 EF AF E9 E3 E1 65 FD 57

Plate 3: Secret Key K and Sub-Key ks2 for 10 Rounds

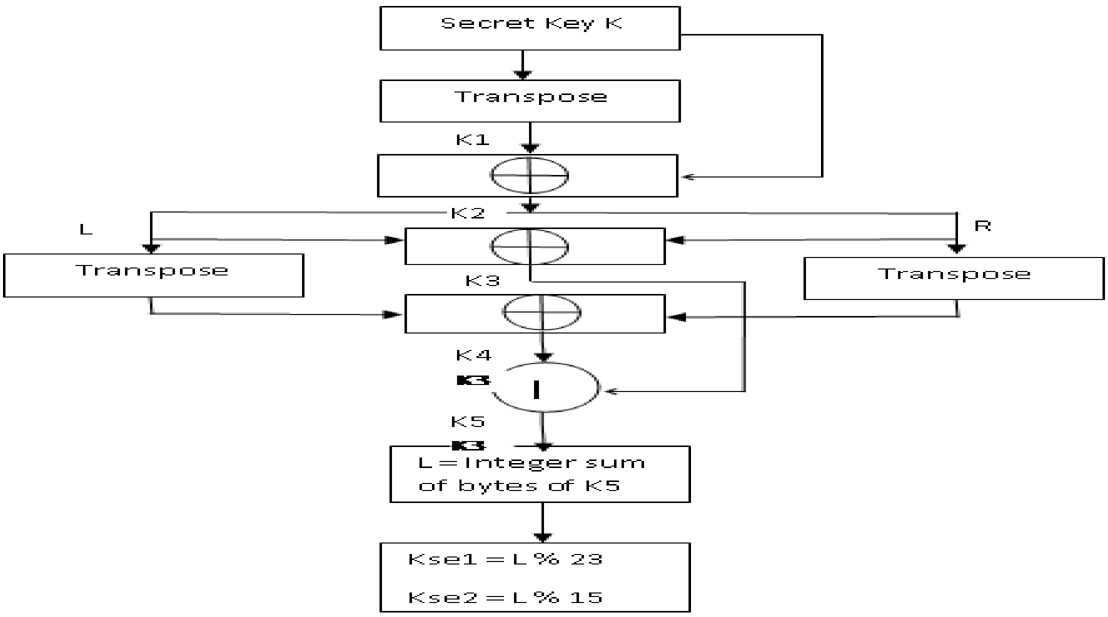

Figure 4

Modified One Time Pad Data Security Scheme

Random Key generation approach: In the proposed work [23] Random key generation techniques have been proposed. Permutation techniques can be used in conjunction with other techniques include substitution, encryption function etc. for effective performance. The goal of this article is to show how the one-time pad encryption technique can be achieved by a combining of these techniques. In this article simulation methodology is to check the encrypted text for alphabets. Here first the ASCII chart is created for small alphabet that shown in Table 1 and then random number key is generated after that plain text is entered which is sent to the recipient as most secure one, then next access the equivalent ASCII number of given plain text from Table 1 hereafter the first ASCII character equivalent number is added with the first random key number i.e. add key to the plain text then implement the equivalent ASCII character of addition repeat the process still end of string, then the encrypted text is written. While sending the encrypted text to the recipient the random number key should be added at the end of encrypted text, key factor is that here at the begin of encrypted message the first number is added that treat as the length of plain text which can be helpful for the recipient to identify the actual string or message.

Algorithm: * Random Key Generation ( );

Step 1 Create Ascii Chart ( ) ;

Step 2 Create random_array();

Step 3 Create Get_Plain_Text( )

Step 4 Write _file for Plain text()

Step 5 Create encrypted Text ()

Step 6 Write file for encrypted text()

Step 7 Send the encrypted file to the recipient

Step 8 Perform decryption process

Step 9 get the Plain text at the destination

Step 10 end

Table 1: Sample ASCII Chart

Alphabet	ASCII	Alphabet	ASCII
A	97	n	110
B	98	o	111
C	99	p	112
D	100	q	113
E	101	r	114
F	102	s	115
G	103	t	116
H	104	u	117
I	105	v	118
J	106	w	119
K	107	x	120
L	108	y	121
M	109	z	122

Data Based Transposition to Enhance Data Avalanche and Differential Data Propagation in Advanced Encryption Standard

In symmetric block ciphers, substitution and transposition operations are performed in multiple rounds to transform plaintext blocks into cipher text blocks. In advanced Encryption Standard (AES) the transposition of data is facilitated by shift row and mix column operations. In Matrix Array Symmetric Key (MASK) Encryption, a block cipher proposed by the author, the data transposition is achieved by data based rotations. The data based transposition procedure offers two advantages. First, it is simple to implement and secondly, the procedure produces a strong data avalanche effect and differential data propagation. In this paper the possibility of improvising the performance of AES using data based transposition in its diffusion rounds is examined. As a case study, the data based transposition procedure has been introduced in AES. The data avalanche and differential data propagation produced in AES have been observed. The paper describes the data based transposition procedure and the enhanced data avalanche and differential data propagation produced in AES. It has been shown that, the data avalanche effect and differential data propagation characteristics of AES have been improved.

Data Based Transposition

The data based transposition is applied to left and right half data parts of a data block using right and left half data parts respectively. This is achieved by rotating one half data block number of times equal to a decimal digit extracted from the other half data block. To facilitate this, a data block in the diffusion section of a symmetric cipher is first bifurcated into two equal parts, the left half part and the right half part. The procedure involved in this method is discussed in the following sub sections. Refer to Figure 2: that shows the block diagram of data based rotation scheme.

Number of Rotations of Right Half Data

Here the number of rotations to be applied on the right half data block is computed. From the byte sum, LDBS of left half data block, an integer number, RDRI (Right Data Rotation Integer) is obtained such that RDRI = LDBS MOD 6. This number lies in the range 0 to 5. The value of this integer depends on the decimal value of left half data block, LHDB.

Rotate Right Half Data Block

The Right Half Data Block, RHDB, is rotated right number of times equal to the integer value RDRI to get RHDB1. Left half data block LHDB is rotated right number of times equal to RDRI. If number of bytes in RHDB = 8, then pseudo code of this operation is as follows:

```
For i=1: (RDRI + 1)
For j=8:-1:1
RHDB (j+1) = RHDB (j)
End For
RHDB (1) = RHDB (9)
End For
RHDB1=RHDB
```

Byte Sum of Right Half Data Block

The decimal values of all the bytes in the right half data block, LHDB1, are added to get right data byte sum, RDBS. This addition can be performed by a loop as indicated in the pseudo code.

```
RDBS=0
For i = 1 to Number of bytes in RHDB1
RDBS = RDBS + decimal value of RHDB1 (i)
EndFor
```

Number of Rotations of Left Half Data

Here, the number of rotations to be applied on the left half data block is computed. From the byte sum RHDB1 of right half data block, an integer number, LDRI (Left Data Rotation Integer) is obtained such that LDRI = RDBS MOD 6.

This number lies in the range 0 to 5. The value of this integer depends on the decimal value of right half data block, RHDB1.

Rotate Left Half Data Block

The Left Half Data Block, LHDB, is rotated right number of times equal to the integer value RDRI to get RHDB1. Left half data block LHDB is rotated right number of times equal to LDRI. If number of bytes in RHDB = 8, the pseudo code of this operation is as follows:

For I = 1: (LDRI + 1)

For j = 8:-1:1

LHDB (j+1) = LHDB (j)

End For

LHDB (1) = LHDB (9)

End For

LHDB1 = LHDB

Concatenate Left and Right Data Blocks

LHDB1 and RHDB1 are concatenated to get DB1 that provides the output data block generated from the transposition section.

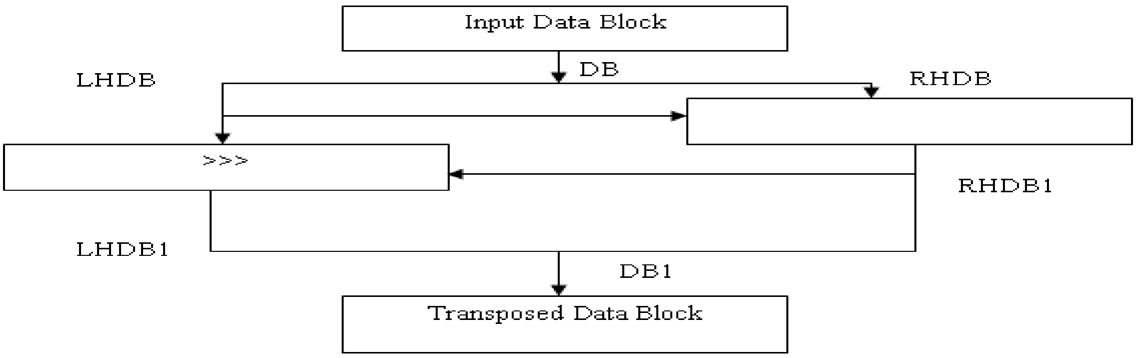

Figure 5: Block Diagram of Data Based Rotation Scheme

It has been shown that the data based transposition procedure, incorporated in Advanced Encryption Standard, as a modification, have produced enhanced data avalanche and differential data propagation in its diffusion rounds. Propagation of differential data through bytes of data block in diffusion rounds should exhibit a random nature in order to facilitate strong resistance against differential attack on a symmetric cipher. A strong data avalanche facilitates better resistance against linear attack on a symmetric cipher. The enhanced data avalanche and differential data propagation characteristics facilitate higher resistance against linear and +differential attacks on the modified cipher. The data based transposition procedure discussed in the paper could be incorporated in any symmetric cipher to enhance resistance of the cipher against linear and differential attacks.

PROPOSED WORK

In this paper we introduce a concept which eliminates the weakness of 56 bit key of DES. DES goes under 16rounds for the complete encryption. For every round sub key is generated from a single key. That means f the key is deduced, decryption of text becomes simpler. But if we give completely different key for every round of DES then cryptanalyst has to apply brute force attack on 16 56 bit sized keys i.e. 16*56 combinations need to be applied. It means 289 combinations need to be tried which makes it strongest of all algorithms. When we choose a position then we can provide different keys for different rounds.

Round 1: 223 88 68 114 200 131 192 199

Round 2: 243 92 129 243 158 180 61 103

Round 3: 94 223 215 59 237 9 255 0

Round 4: 65 31 241 61 29 167 11 158

Round 5: 128 81 127 68 57 241 87 63

Round 6: 88 155 239 15 231 95 120 171

Round 7: 159 172 77 247 134 184 170 48

Round 8: 58 157 43 160 67 90 40 225

Round 9: 46 28 182 149 36 210 245 226

Round 10: 174 62 177 55 222 31 206 147

Round 11: 197 92 227 246 137 190 240 254

Round 12: 117 247 116 105 12 100 30 120

Round 13: 169 187 20 158 129 123 114 67

Round 14: 117 114 79 237 19 105 254 97

Round 15: 252 233 124 85 207 214 38 251

Round 16: 195 92 184 74 200 65 228 122

The only concept we need to remember at the time of decryption,

Round 16 key will be used for round 1 and

Round 15 key will be used for round 2 and so on and the process will be reversed.

In the DES Actual Key Generation Algorithm

Step 1: We take one key.

Step 2: It undergoes Permuted choice -1.It makes it 56 bit key.

Step 3: Then the 56 bit key is divided into 2 halves.

Step 4: Now these left part and right part undergoes left shifts which makes sub keys.

Step 5: There are 16 sub keys generated for every round of algorithm.

Step 6: Now these 56 bit sub keys undergoes PC-2 (Permuted choice-2).

Step 7: And the resultant key generated is of 48 bits.

PKPFSE Algorithm

Step 1: A file is converted into bytes.

Step 2: A position is chosen to generate keys.

Step 3: Now 64*16 /8 =128 bytes are chosen.

Step 4: First 8 bytes are used for Round 1.

Step 5: Next 8 bytes are used for Round 2 and so on.

Step 6: 16 altogether different keys are generated.

Step 7: Now this key undergoes PC-1 and 56 bit key is generated.

Step 8: And then this key undergoes PC-2 and 48 resultant key is generated.

Comparison of Old DES and New DES

Table 2

Features	Old DES	New DES
Key size	64 bits	64 bits
No of keys	1	16
Key sending	Yes	No
Key generation	Manual	Automatic
Future scope	Nil	To a large extent
Implementation	Yes	Yes
Speed	Fast	Slow
Key	One time Usage	Unlimited key generation
Key Security	Key need to be encrypted	Key pool generated file need to be sent securely once
Redistribution of key	No	Yes

CONCLUSIONS

This thesis is on a new research topic about making already given algorithms stronger by giving them powerful keys for every round. By applying different key for every round it has made the cryptanalyst to break 1024 bits instead of 64 which takes much longer to attack for in case of say DES. This work not only emphasizes on the security strength because of key size but also takes advantage of key security by not sending it on the network. Other than the advantage of one time usage of keys, the key pool generated can be saved and use back.

REFERENCES

1. William Stallings, "Network Security Essentials (Applications and Standards)", Pearson Education, 2004, pp. 2–80.
2. Vinod Shoukeen," Encryption and Decryption Technique for Message Communication", IJECT vol. 2, Issue 2, June 2011
3. Majdi Al-qdah & Lin Yi Hui," Simple Encryption/Decryption Application", International Journal of Computer Science and Security, Volume (1): Issue (1).
4. http://csrc.nist.gov/groups/ST/toolkit/block_ciphers.html
5. Federal Information Processing Standards Publication 197 November 26, 2001,"Announcing the advanced encryption standard (AES), http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.
6. The Transitioning of Cryptographic Algorithms and Key Sizes,

   http://csrc.nist.gov/groups/ST/key_mgmt/documents/Transitioning_CryptoAlgos_070209.pdf
7. Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid," Recommendation for Key Management – Part 1: General (Revision 3)", NIST Special Publication 800-57, Computer Security Division,Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-8930, July 2012z.
8. Key size, http://en.wikipedia.org/wiki/Key_size
9. Yashpal Mote, Paritosh Nehete, Shekhar Gaikwad," Superior Security Data Encryption Algorithm (NTRU)", An International Journal of Engineering Sciences ISSN: 2229-6913 Issue July 2012, Vol. 6.
10. Ajay Kakkar, M. L. Singh, P.K. Bansal," Comparison of Various Encryption Algorithms and Techniques for Secured Data Communication in Multinode Network", International Journal of Engineering and Technology Volume 2, No. 1, January, 2012
11. Paul A.J, P Mythili and Paulose K Jacob," Matrix based Key Generation to Enhance Key Avalanche in Advanced Encryption Standard", IJCA Proceedings on International Conference on VLSI, Communications and Instrumentation (ICVCI) (2):31–34, 2011.
12. Key Stretching,http://en.wikipedia.org/wiki/Key_stretching
13. http://www.cryptosys.net/3des.html
14. http://www.schneier.com/paper-blowfish-oneyear.html
15. Bruce Schneier, "The Twofish Encryption Algorithm", Dr. Dobb's Journal, December 01, 1998, http://www.drdobbs.com/security/the-twofish-encryption-algorithm/184410744.
16. http://en.wikipedia.org/wiki/Twofish
17. RSA Algorithm, http://pajhome.org.uk/crypt/rsa/contrib/RSA_Project.pdf

18. M.CATHERINE JENIFER, P.JAYACHANDAR,” CRYPTANALYSIS ON RSA ALGORITHM”, International Journal of Communications and Engineering, Volume 03– No. 3, Issue: 01 March 2012.
19. B. Padmavathi, S. Ranjitha Kumari,” A Survey on Performance Analysis of DES, AES and RSA Algorithm along with LSB Substitution Technique”, International Journal of Science and Research (IJSR), Volume 2 Issue 4, April 2013.
20. D. Sravana Kumar, CH. Suneetha,A. Chandrasekhar,” ENCRYPTION OF DATA USING ELLIPTIC CURVE OVER FINITE FIELDS”, International Journal of Distributed and Parallel Systems (IJDPS) vol. 3, No.1, January 2012.
21. Sharad Patil, Manoj Devare & Ajay Kumar,” Modified One Time Pad Data Security Scheme: Random Key Generation Approach”, International Journal of Computer Science and Security (IJCSS), volume (3) : Issue (2)