International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

A Hybrid Approach Based Encryption Decryption Using Blowfish Algorithm

Er. Manoj Kumar Ameria[1], Dr. Amit Sharma[2]

[1]M Tech Scholar, [2]Professor, Computer Science & Engineering,
Vedant College of Engineering & Technology (India)

ABSTRACT

Blowfish is a popular security algorithm that was developed by Bruce Schneier in the advent of the year 1994. The algorithm works on the same line as DES and it consumes block coding with blocks of a size of 64 bit. Blowfish became quite popular after its advent, just because Bruce Schneier himself is one of the most famous cryptology experts and above this the algorithm is non patented, open source freely and freely available for use and modifications.

Blowfish is a 64-bit block cipher with a variable-length key. It defines 2 distinct boxes: S boxes, a P box and four S boxes. Taking into consideration the P box P is a one-dimensional field with 18 32-bit values. The boxes contain variable values; those can be implemented in the code or generated during each initialization. The S boxes S1, S2, S3, and S4 each contain 256 32-bit values.

Keyword*: *algorithm, blowfish, cryptography, encryption, security.

I. INTRODUCTION

With the advancements in the field of communication technology it is possible to exchange messages between the communicating parties in a short span of time. The content of the messages has to be protected from intruders. Cryptography can be used for hiding the data during transmission. In present scenario various data encryption algorithms are available for data security which has always been important in all aspects of life. As we are having number of cryptographic algorithm so sometimes it can create little bit confusion to select best one [1].From the literature review we found that the blowfish algorithm has best performance than the other algorithms. So here data encryption and decryption process occurred using that algorithm. And here also the algorithm is modified by hash algorithm so it provides a great security during data transmission.

II. LITERATURE SURVEY

2.1 Types of Cryptography

Encryption algorithms are classified in two broad categories- Symmetric key Cryptography and Asymmetric Key Cryptography.

2.1.1 Symmetric Key Cryptography

In symmetric Cryptography the key that is used for encryption is similar to the key used in decryption. So the key distribution has to be made prior to the transmission of information. The key plays a very important role this cryptography since the security directly depends on the nature of the key length etc. There are various symmetric key algorithms such as DES &AES.

2.1.2 Asymmetric Key Cryptography

In Asymmetric Cryptography, two different keys are used for encryption and decryption they are public and private. The public key is available to anyone on the network; those who want to encrypt the plaintext should know the Public Key of receiver. Only the authorized person can decrypt the cipher text through his/her own private key. Private Key is kept secret from the others. Symmetric Encryption Algorithm is faster as compared to Asymmetric key algorithms. The memory requirement of Symmetric algorithm is less than asymmetric.

2.1.3 DATA ENCRYPTION STANDARD (DES)

DES (Data Encryption Standard) is the first encryption standard recommended by NIST (National Institute of Standards and Technology). It was developed by an IBM in 1974 and adopted as a national standard in 1997.

DES is a 64-bit block cipher under 56-bit key. The algorithm processes with a permutation of sixteen rounds block cipher and a final permutation. DES application is very popular in the commercial, military, and other domains. DES standard is public & the design criteria used are classified.

2.1.4 ADVANCED ENCRYPTION STANDARD (AES)

AES was invented by scientists named Joan and Vincent Rijmen in the year 2000. AES makes use of the Rijndael block cipher, Rijndael key and block length can be of 128, 192 or 256-bits, if both the key-length and block length are 128-bit. Rijndael performs 9 processing rounds. If the block/key is 192-bit, it will perform 11 processing rounds & for 256-bits, Rijndael performs 13 processing rounds. Each processing round involves the following four steps:

1. Substitute bytes – It uses an S-box to perform a byte by byte substitution of the block.
2. Shift rows – It is simple permutation.
3. Mix column – It is a substitution method where data in each column from the shift row step is multiplied by the algorithm's matrix.
4. Add round key – It is the key for the processing round is XORed with the data.

2.2 Modes of Encryption/Decryption

2.2.1 ECB (Electronic Code Book)

In this mode data is divided into 64-bit blocks and each block is encrypted one at a time. Separate encryptions with different blocks are totally independent of each other. This means that if data is transmitted over a network or phone line, transmission errors will only affect the block containing the error. It also means, however, that the blocks can be rearranged, thus scrambling a file beyond recognition, and this action would go undetected. ECB is the weakest of the various modes because no additional security measures are implemented besides the basic DES algorithm. However, ECB is the fastest and easiest to implement, making it the most common mode of DES seen in commercial applications. This is the mode of operation used by Private Encryptor.

2.2.2 CBC (Cipher Block Chaining)

In this mode of operation, each block of ECB encrypted cipher text is XORed with the next plaintext block to be encrypted, thus making all the blocks dependent on all the previous blocks. This means that in order to find the plaintext of a particular block, you need to know the cipher text, the key, and the cipher text for the previous block. The first block to be encrypted has no previous cipher text, so the plaintext is XORed with a 64-bit number called the Initialization Vector, or IV for short. So if data is transmitted over a network or phone line and there is a transmission error (adding or deleting bits), the error will be carried forward to all subsequent blocks since each block is dependent upon the last. If the bits are just modified in transit (as is the more common case) the error will only affect all of the bits in the changed block, and the corresponding bits in the following block. The error doesn't propagate any further. This mode of operation is more secure than ECB because the extra XOR step adds one more layer to the encryption process.

2.2.3 CFB (Cipher Feedback)

In this mode, blocks of plaintext those are less than 64 bits long can be encrypted. Normally, special processing has to be used to handle files whose size is not a perfect multiple of 8 bytes, but this mode removes that necessity (Private Encryptor handles this case by adding several dummy bytes to the end of a file before encrypting it). The plaintext itself is not actually passed through the DES algorithm, but merely XORed with an output block from it, in the following manner: A 64-bit block called the Shift Register is used as the input plaintext to DES. This is initially set to some arbitrary value, and encrypted with the DES algorithm. The cipher text is then passed through an extra component called the M-box, which simply selects the left-most M bits of the cipher text, where M is the number of bits in the block we wish to encrypt. This value is XORed with the real plaintext, and the output of that is the final cipher text. Finally, the cipher text is fed back into the Shift Register, and used as the plaintext seed for the next block to be encrypted. As with CBC mode, an error in one block affects all subsequent blocks during data transmission. This mode of operation is similar to CBC and is very secure, but it is slower than ECB due to the added complexity.

2.2.4 OFB (Output Feedback)

This is similar to CFB mode, except that the cipher text output of DES is fed back into the Shift Register, rather than the actual final cipher text. The Shift Register is set to an arbitrary initial value, and passed through the DES algorithm. The output from DES is passed through the M-box and then fed back into the Shift Register to prepare for the next block. This value is then XORed with the real plaintext (which may be less than 64 bits in length, like CFB mode), and the result is the final cipher text. Note that unlike CFB and CBC, a transmission error in one block will not affect subsequent blocks because once the recipient has the initial Shift Register value; it will continue to generate new Shift Register plaintext inputs without any further data input. However, this mode of operation is less secure than CFB mode because only the real cipher text and DES cipher text output is needed to find the plaintext of the most recent block. Knowledge of the key is not required.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

III. BLOWFISH ALGORITHM

Blowfish is a symmetric encryption algorithm, meaning that it uses the same secret key to both Encrypt and decrypt messages. Blowfish is also a block cipher [5], meaning that it divide message up into fixed length blocks during encryption and decryption. The block length for Blowfish is 64 bits; messages that aren't a multiple of eight bytes in size must be padded.

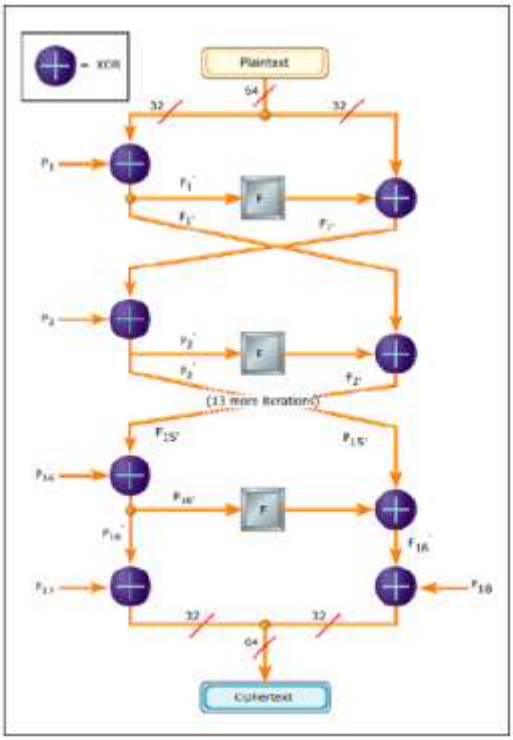

Figure 1: Blowfish algorithm

Blowfish consists of two parts: key-expansion and data encryption. During the key expansion stage, the inputted key is converted into several sub key arrays total 4168 bytes. There is the P array, which is eighteen 32-bit boxes, and the S-boxes, which are four 32-bit arrays with 256 entries each. After the string initialization, the first 32 bits of the key are XORed with P1 (the first 32-bit box in the P-array).The second 32 bits of the key are XORed with P2, and so on, until all 448 or fewer key bits have been XORed cycle through the key bits by returning to the beginning of the key, until the entire P-array has been XORed with the key. Encrypt the all Zero string using the Blowfish algorithm, using the modified P-array above to get a 64 bit block. Replace P1 with the first 32 bit of output (from the 64 bit block). Use the 64 bit output as input back into the Blowfish cipher to get a new 64 bit block. Replace the next values in the P-array with the block. Repeat for all the values in the P-array and all the S-boxes in the order.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

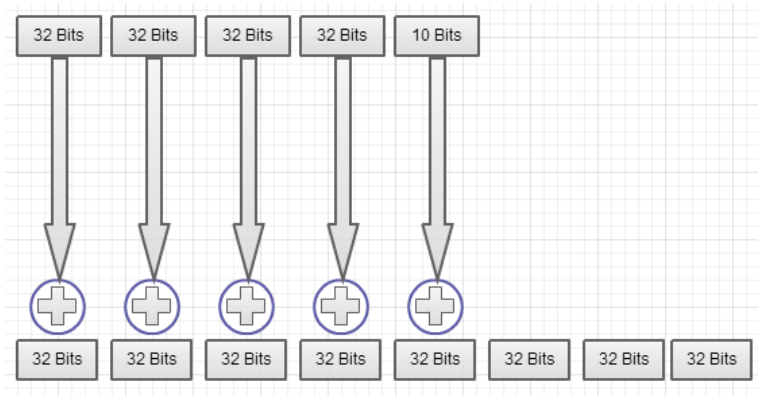

Figure 2:XORing bits once the key has been traversed through once

Encrypt the all zero string using the Blowfish algorithm [1], using the modified P-array above, to get a 64 bit block. Replace P1 with the first 32 bits of output, and P2 with the second 32 bits of output (from the 64 bit block). Use the 64 bit output as input back into the Blowfish cipher, to get a new 64 bit block. Replace the next values in the P-array with the block. Repeat for all the values in the P-array and all the S boxes in order.

IV. PROPOSED SYSTEM

This system basically uses the Blowfish encryption algorithm [1] to encrypt the data file. This algorithm is a 64-bit block cipher with a variable length key. This algorithm has been used because it requires less memory. It uses only simple operations, therefore it is easy to implement. It is a 64 bit block cipher and itis fast algorithm to encrypt the data. It requires 32 bit microprocessor at a rate of one byte for every 26 clock cycles.

It is variable length key block cipher up to 448 bits. Blowfish contains 16 rounds.

Each round consists of XOR operation and a function. Each round consists of key expansion and data encryption. Key expansion generally used for generating initial contents of one array and data encryption uses a 16 round Feistal network [3].

Plain text and key are the inputs of this algorithm. 64 bit Plain text is taken and divides into two 32bits data and at each round the given key is expanded & stored in 18 p-array and gives 32bit key as input and XOR ed with previous round data.

Functionality is to divide a 32-bit input into four bytes and uses those as indices into an S-array. The lookup results are then added and XORed together to produce the output. At16th round there is no function .The out-put of this algorithm should be 64bit cipher text.

4.1 Algorithm Steps

It is having a function to iterate 16 times of network. Each round consists of key-dependent permutation and a key and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookup tables for each round.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

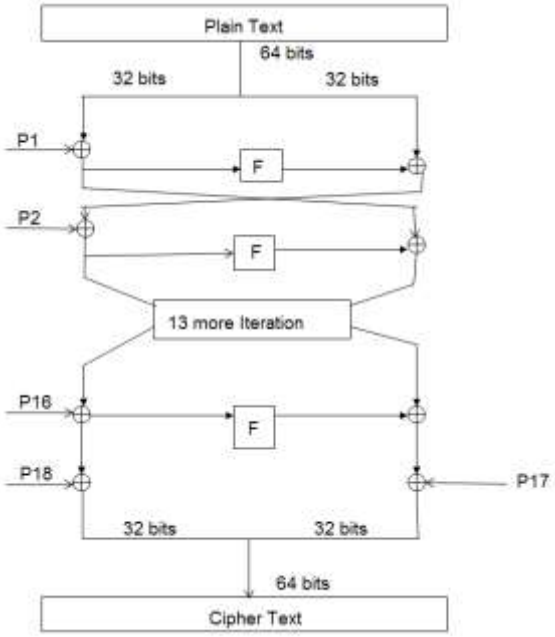

Figure 3: The Fiestel structure of blowfish algorithm[9]

Algorithm

Divide x into two 32-bit halves: xL, xR

For i = 1to 32:

xL = XL XOR Pi

xR = F(XL) XOR xR

Swap XL and xR

Swap XL and xR (Undo the last swap.)

xR = xR XOR P17

xL = xL XOR P18

Recombine xL and xR .

And after recombining these, we get the cipher text of 64-bits according to the inputs.

Decryption is exactly the same as encryption only the P1, P2,,,,,,,,,P18 are used in reverse order.

Subkeys:-

Blowfish uses a large number of sub keys[10]. Theses keys must be precomputed before any data encryption or decryption.

The P-array consists of 18 32-bit sub keys [10]:

P1,P2,,,,,,,,P18.

There are four 32-bit S-boxes with 256 entries each [10]:

S1,0, S1,1,,,,, S1,255;

S2,0, S2,1,,,,, S2,255;

S3,0, S3,1,,,,, S3,255;

S4,0, S4,1,,,,, S4,255.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

For developing the logic, key generation is one of the complex schedules of Blowfish encryption because of the fact that it include variable size key. So for the development of the key the use of S BOX (Substitution Box) can be a greater aid for developing of logic of this algorithm [1].

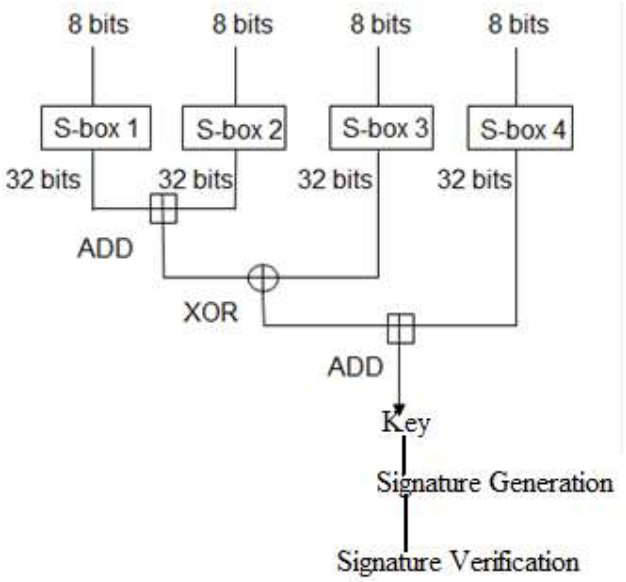

Figure 4: Key generation using fiestel network for blowfish algorithm [1]

Signature Generation

Followings are the signature generation steps:

- Creates a message digest *H(m)* as an integer of the information to be sent between 0 and *n-1*. Compute the signature by using the key k as $s = H(m)^k \bmod n$
- *s* is the signature of the message m. Send *s* with the message *m* to recipient.

Signature Verification

Signature verification steps are as follows:

- By using sender key k, compute integer $v = s^k \bmod n$. *v* be the message digests calculated by sender.
- Independently computes the message digest of the message that has been signed.
- If both message digests are identical, the signature is valid.

Simulation

The simulation uses the provided classes in java environment to simulate the performance of Blowfish and proposed Algorithm. The Cipher class provides the functionality of a cryptographic cipher used for encryption and decryption. It forms the core of the JCE framework.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (02), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

Table 1: Algorithms' Settings

Key Size (bit)	Blowfish Execution Time(ms)		Proposed Algorithm Execution Time(ms)			
	Key Generation Time(ms) (a)	Total Execution time(ms) (a)	Key Generation Time(ms) (A)	Signature Generation Time(ms) (B)	Signature verification time(ms) (C)	Total Execution time(ms) (A+B+C)
32	1049	1049	1042	375	390	1807
64	2055	2055	1869	375	375	2619
128	2093	2093	1875	375	391	2641
256	2178	2178	2029	375	391	2795

The evaluation is meant to evaluate the results by using block ciphers. Hence, the load data (plaintext) is divided into smaller block size as per algorithm settings given in Table 1 above.

System Parameters

The experiments are conducted using Intel core i3 processor with 4GB of RAM. The simulation program is compiled using the default settings in jdk 1.7development kit for JAVA. The experiments will be performed couple of times to assure that the results are consistent and are valid to compare the different algorithms.

Experiment Factors

Since the security features of each algorithm as their strength against cryptographic attacks is already known and discussed. The chosen factor here to determine the performance is the algorithm's speed to encrypt/decrypt data blocks of various sizes.

Results and Analysis

This section will show the results which are obtained by running the simulation program using different data loads. The results show the impact of changing data load on each algorithm and the impact of Cipher Mode used.

The results show the superiority of proposed algorithm over blowfish algorithms in terms of the processing time.

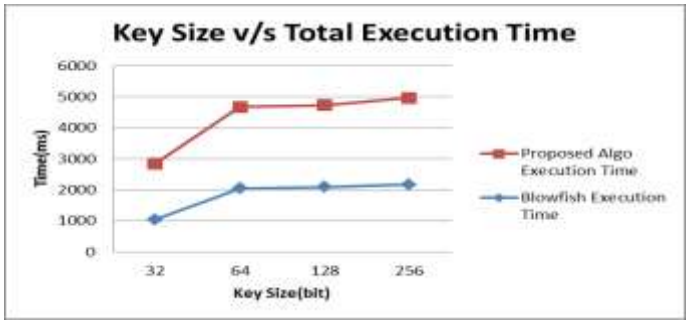

Figure 5: Diagram showing key size v/s Blowfish & Proposed Algorithm's execution time.

V. CONCLUSION

The presented simulation results showed that proposed algorithm has a better performance than blowfish encryption algorithms used. Since Blowfish has not any known security weak points so far, this makes it an excellent candidate to be considered as a standard encryption algorithm.

In future this analysis can be implemented in better simulators to get better results. This analysis can be done in another simulator by taking networking into consideration to show which algorithm performs better in network. The simulators which can be used are: MATLAB, ns2, ns3, OPNET, NetSim etc. These simulators will give better results for cryptographic applications in network.

Blowfish is a 16 pass block encryption algorithm that is never broken.

REFERENCES

[1] Sweta K Parmar, K.C Dave, "Implementation of data Encryption and Decryption Algorithm for Information security", Proceedings of SARC-IRAJ International Conference, 14th July 2013, Delhi, India, ISBN: 978-93-82702-21-4.

[2] Deepak Kumar Dakate, Pawan Dubey, "Performance Comparison of Symmetric Data Encryption Techniques", International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 4, June 2012.

[3] Amogh Mahapatra and Rajballav Dash, thesis on, "Data Encryption & Decryption by using Hill Cipher Technique and Self Repetitive Matrix", National Institute of Technology Rourkela 2007.

[4] E. Thambiraja, G Ramesh, Dr. R. Umarani, "A survey on various most common encryption techniques", International journal of advanced research in computer science and software engineering, volume 2, issue 7, july 2012.

[5] Jawahar Thakur, Nagesh Kumar, "DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis", International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459,Volume 1, Issue 2, December 2011.

[6] S. Pavithra and Mrs. E. Ramadevi, " Performance Evaluation of Symmetric Algorithms", Journal of Global Research in Computer Science, Volume 3, No. 8, August 2012.

[7] Pratap Chandra Mandal, " Evaluation of performance of the Symmetric Key Algorithms: DES, 3DES ,AES and Blowfish", Journal of Global Research in Computer Science, Volume 3, No. 8, August 2012.

[8] AL.Jeeva, Dr.V.Palanisamy, K.Kanagaram, "Comparative Analysis of Performance Efficiency & Security Measures of Some Encryption Algorithms", International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622, Vol. 2, Issue 3, May-Jun 2012, pp.3033-3037.

[9] Manmeet Kaur , Manjit Kaur , Gurmohan Singh, " Comparison of TACIT Encryption Algorithm with Various Encryption Algorithms", International Journal of Electronics and Computer Science Engineering, ISSN- 2277-1956.

[10] B. Schneier, applied cryptography, John Wiley & Sons, New York, 1994.

[11] P. Karthigai Kumar , K. Baskaran, "An ASIC implementation of low power and high throughput blowfish crypto algorithm", Microelectronics Journal 41 (2010) 347–355.

[12] M. Anand Kumar and Dr.S.Karthikeyan, "Investigating the Efficiency of Blowfish and Rejindael (AES) Algorithm", I. J. Computer Network and Information Security, 2012, 2, 22-28, Published Online March 2012 in MECS.