International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 6, August - 2012

A Conceal Extremity of Blowfish Scheme with a Ameliorate Feistel Structure

Hemraj Shobharam Lamkuche
Research Scholar, School of IT & Science,
Dr. G. R. Damodaran College of Science,
Coimbatore, Tamil Nadu, India.

Prof. T. Santha
Head of the Department, School of IT & Science,
Dr. G. R. Damodaran College of Science,
Coimbatore, Tamil Nadu, India.

Abstract: - Blowfish [1] is a new symmetric key block cipher suite. In cryptography, encryption is the process of transmitting information by using the encryption algorithm to make the information illegible. The unreadable information is referred as cipher text. Blowfish is unpatented algorithm which can be used by anyone freely; technique wise it is a new secret key block cipher with a powerful feistel network, iterating a simple encryption function for a static 16 rounds to produce a cipher of input information. Encryption has exemplary used by government and militaries to relieve privy communication. So without our awareness, the intruders can attempt to obtain and gain control over the system to access secure confidential information by using most common attacks [2] like DoS Attack, Backdoor, Trojan horse, Packet Sniffing, etc. So there must not be any compromise in securing our resources. In this paper we have precisely transfigured the feistel network to unleash more power of blowfish algorithm, the experimental results shows the immense speed and power of our modified algorithm which works even faster than the original one.

Keywords: Blowfish Algorithm, Feistel Network, Cryptography, Information Security, Attacks.

1. Introduction

Private Information has not been protected on the internet in general. The evolution of Internet has resulted in large quantities of information being exchanged by business or private individuals. The nature of this information is typically both public and private, and much is transmitted over the hyper text transfer protocol (HTTP) in an insecure manner. According to long-running survey by the Internet system consortium [3], 900 million active hosts are connected to the internet. This survey measures active domains on the internet; in other words, the number of DNS servers connected to the internet. To indicate just how quickly the internet is growing, the number of internet-connected hosts doubled in the five years from January 2007 to January 2012. When we use the Internet, we're not always just clicking around and passively taking in information, such as reading news articles or blog posts; a great deal of our time online involves sending others our own information. Ordering something over the Internet, whether it's a book, a CD or anything else from an online vendor, or signing up for an online account, requires entering in a good deal of sensitive personal information. A typical transaction might include not only our names, e-mail addresses and physical address and phone number, but also passwords and personal identification numbers (PINs).

Information security is provided on computers and over the Internet by a variety of methods. A simple but straightforward security method is to only keep sensitive information on removable storage media like portable flash memory drives or external hard drives. But the most popular forms of security all rely on encryption, the process of encoding information in such a way that only the person (or computer) with

the key can decode it. The encryption algorithms work on chunks of specific sized data along with a key resulting in blocks of cipher text. The National Institute of Standards and Technology (NIST) is a federal agency that approved some commonly used block cipher algorithms are DES, AES, 3-DES, DEA, IDEA, RC2, RC5, CAST, Blowfish and Skipjack.

Encryption [4] is the coding or scrambling of information so that it can only be decoded and read by someone who has the correct decoding key. A symmetric key encryption scheme has 5 elements shown in fig 1.

A. *Plaintext:* This is the original message or data that is fed into the algorithm as input.

B. *Encryption algorithm:* The encryption algorithm performs various substitutions and transformations on the plaintext.

C. *Secret key:* The secret key is also input to the algorithm. The exact substitutions and transformations performed by the algorithm depend on the key.

D. *Cipher text:* This is the scrambled message produced as output. It depends on the plaintext and the secret key.

E. *Decryption algorithm:* This is essentially the encryption algorithm run in reverse. It takes the cipher text and the same secret key and produces the original plaintext.

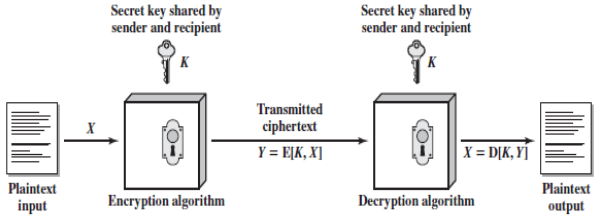

Fig 1: Model of symmetric encryption

2. Existing Methodology

Bruce Schneier [5] specifically designed the Blowfish algorithm as an unlicensed, uncopyrighted alternative to DES and first presented in 1993. Since that time, Blowfish has been extensively tested by the cryptographic community and found to be reasonably secure. Blowfish has been extensively analyzed and no significant weaknesses have been found. It is considered to be a strong algorithm and has been implemented in over 130 commercial applications. The Blowfish algorithm is a symmetric block cipher that can be used as a drop-in replacement for DES or IDEA. It takes a variable-length key, from 32 bits to 448 bits, making it ideal for both domestic and exportable use. Blowfish is classified as public domain; as such it has been analyzed extensively and gone through years of peer review. At no point since its initial release in 1993 has the Blowfish code ever been cracked. This is significant when you consider that the source code to the algorithm is freely available. Blowfish supports key lengths of 32 to 448 bits, making it one of the strongest encryption algorithms on the market.

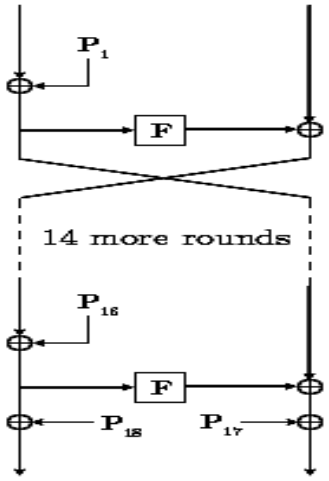

Fig 2: Feistel Structure of Blowfish Algorithm

Using the Feistel [6] network shown in fig 2 gives the cipher two very desirable properties: decryption using the same f function (even if it is non-invertible) and the ability to iterate the function multiple times. These multiple iterations are called rounds. The more rounds, the more secure the algorithm is. The recommended number of rounds depends on the specific algorithm; for Blowfish, it is 16. The algorithm consists of two parts: a key-expansion part and a data- encryption part. Key expansion converts a key of at most 448 bits into several sub-key arrays totaling 4168 bytes. Data encryption occurs via a 16-round through feistel network. Each round consists of a 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 lookups per round. The algorithm keeps two sub-key arrays [7]: the 18-entry P-array and four 256-entry S-boxes. The S-boxes accept 8-bit input and produce 32-bit output. One entry of the P-array is used every round, and after the final round, each half of the data block is XORed with one of the two remaining unused P-entries. The fig 3 shows Blowfish's F-function. The function splits the 32-bit input into four eight-bit quarters, and uses the quarters as input to the S-boxes. The outputs are added modulo 2^{32} and XORed to produce the final 32-bit output.

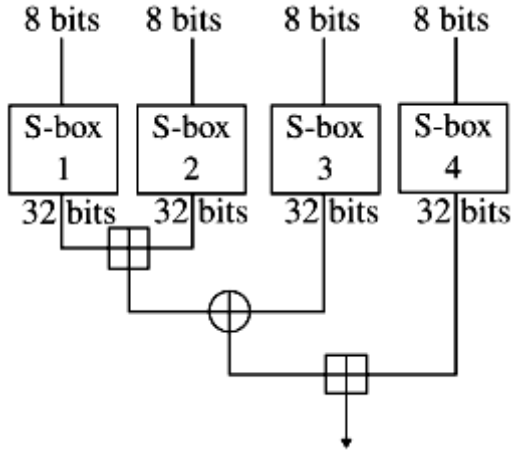

Fig 3: The Feistel F function of Blowfish Algorithm

3. Assert Approach

In our stimulus approach to re-transfigure the Blowfish algorithm which is 64-bit block cipher. Here in our proposed approach we have successfully reduced the round by 2 without affecting the security. In this experimental work the algorithm perform 14 iterations, or rounds, each of which consists of a permutation dependent on the encryption key and a substitution dependent on the encryption key and the data. Each operation is a logical EXCLUSIVE OR (XOR) [1] which returns true if either, but not both, of its operands are true and addition on 32-bit words. A typical implementation of the Blowfish algorithm can encrypt or decrypt a 64-bit block of data in approximately 12 clock cycles, while a 128-bit message requires 24 clock cycles and so on in linear fashion.

Blowfish uses a large number of subkeys which are precomputed before any data encryption or decryption. The P-array consists of 18 four-byte subkeys:

P1, P2, P3, P4, P5, ..., P18

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

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

Here we have enhanced the performance of blowfish algorithm by transforming the F function by adopting the concept of multithreading. We have also transfigured the feistel structure by increasing the performance and decreasing the speed of execution, which ultimately increase security too.

3.1 Original F Function

The most interesting portion of Blowfish is its non-invertible F function as shown in fig 3. This function uses modular arithmetic to generate indexes into the S-boxes. Modular arithmetic is usually used to create non-invertible F functions. Original F Function is defines as follows:

$$F(x) = ((S1 + S2 \bmod 2^{32}) \text{ XOR } S3) + S4 \bmod 2^{32}$$

3.2 Adduce H Function

This research proposed a new improvement in the blowfish algorithm. Here we have used the same original F function for the feistel network, but in addition, we have transfigured our new H function into the feistel network which will give more complex substitution of input data to produce complex output data. The H function operates in very high speed due to an inverter which is a logic gate [8] which implements logical negation.

Our proposed H function is defines as follows:

$$H(x) = NOT\ (F(x)\ XOR\ Xr)$$

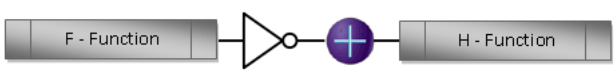

Fig 4: The Feistel H Function of Blowfish Algorithm

3.3 Adduce Encryption Algorithm

When Blowfish is invoked the first step operation is to compute the P array and S-Boxes known as the key schedule. Upon completion of the key schedule the first iteration of Blowfish algorithm begins. The initial input uses zeros as the 64 bit block. The resulting cipher text is 64 bits and is used to overwrite the first P array and the second P array. The cipher text is used as input for the next iteration resulting in a new P3 and P4 subkeys. This cycle will continue until all P arrays have been successfully filled.

A Feistel network [1] for 14 rounds as shown in fig 5 can be described by the following algorithm:

```
Divide x into two 32-bit halves: Xl, Xr

For i = 1 to 14:

        Xl = Xl XOR P(i)
        Xr = Xr XOR F(Xl)
        Xl = Xl XOR H(Xr)

Swap Xl and Xr
Swap Xl and Xr (Undo the last swap.)

        Xr = Xr XOR P15
        Xl = Xl XOR P16

        Xl = Xl XOR P17
        Xr = Xr XOR p18

Recombine Xl and Xr

Function F (see Fig 3):
Divide Xl into four eight-bit quarters:
a, b, c, and d.
```

$F(x) = ((S1 + S2 \bmod 2^{32}) \text{ XOR } S3) + S4 \bmod 2^{32}$

```
Function H (see Fig 4):
It takes the input data from the F
function and by using the inverter
logic circuit.
```

$F(x) = ((S1 + S2 \bmod 2^{32}) \text{ XOR } S3) + S4 \bmod 2^{32}$

$H(x) = NOT\ (F(x)\ XOR\ Xr)$

3.4 Contingent Decryption Algorithm

Decryption is exactly the same as encryption; it is a reverse engineering process. Here the P-arrays are used in reverse order. The implementation of blowfish that require the fastest speeds should unroll the loop and ensure that all the subkeys are stored in cache. A decryption algorithm can be described as follows:

```
        Xl = Xl XOR P17
        Xr = Xr XOR p18

        Xr = Xr XOR P15
        Xl = Xl XOR P16
Swap Xl and Xr

        Xr = Xr XOR H(Xl)
        Xl = Xl XOR F(Xr)
        Xr = Xr XOR P(i)

Recombine Xl and Xr
```

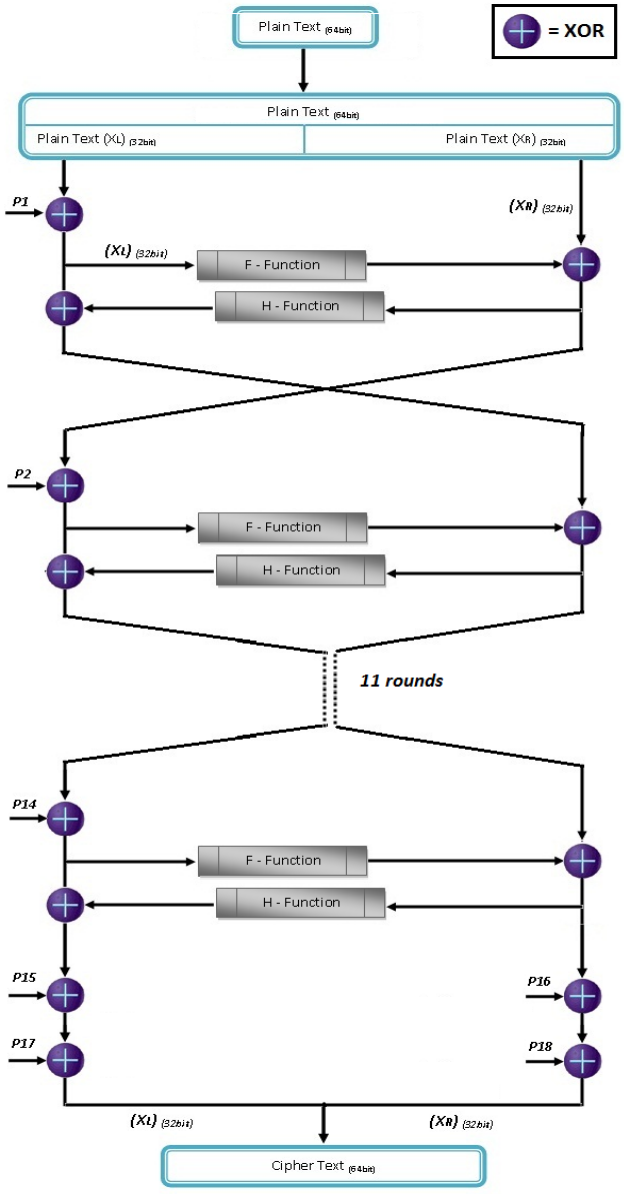

Fig 5: An Improved Feistel structure Blowfish Algorithm

4. Simulation and Results

In this paper we have successfully enhanced the power of blowfish algorithm by reducing the execution time using the new transfigured feistel network, it addition to it we have also introduced a new H function for more complex substitution of input data through encryption and decryption algorithm. For simulation we have used the Microsoft Visual Basic 6.0 for 32-Bit Windows Development IDE and we developed a file encryption program which takes any source file (Image, Audio, Document, Text, etc) to encrypt by using a static secret key and produces an encrypted version of source file as shown in table 1,2 and 3.

Table 1: A Table of Comparison of Execution Time Elapsed for Image File (102 KB)

Time Vs Algorithm	Start Time(ms)	End Time(ms)	Total Time(ms)
Original Blowfish	*47807.98400*	*47811.73400*	*3.7500*
Modified Blowfish	*47724.64000*	*47727.96200*	*3.3220*

Table 2: A Table of Comparison of Execution Time Elapsed for Microsoft Document File (97.5 KB)

Time Vs Algorithm	Start Time(ms)	End Time(ms)	Total Time(ms)
Original Blowfish	*48379.09200*	*48382.73100*	*3.6390*
Modified Blowfish	*48324.92700*	*48328.33200*	*3.4050*

Table 3: A Table of Comparison of Execution Time Elapsed for Portable Document File (207 KB)

Time Vs Algorithm	Start Time(ms)	End Time(ms)	Total Time(ms)
Original Blowfish	*48990.36000*	*48997.34900*	*6.9890*
Modified Blowfish	*49065.83400*	*49072.55100*	*6.7170*

We have precisely simulated and tested our file encryption application software with original blowfish algorithm and our new proposed blowfish algorithm. The test perform on ASUS X53S Laptop with Intel® Inside™ Core™ i5 2430M CPU with computing speed of 2.40 GHz. Installed RAM: 6.00 GB, System Type: 32-Bit, Operating System: Microsoft Windows 7 Ultimate Edition Copyright © 2009 Microsoft Corporation.

5. Conclusion & Future work

Cryptography is an evolving area and what is safe today may not be safe tomorrow. This paper will satisfy our foremost aim of providing a system which is *"Infeasible to get breached"*. It also provides a high degree of data security when transmitting the data over any insecure medium. Intruders will not have any idea about our proposed new H function in the newly transfigured feistel structure for blowfish algorithm, so breaching the system is highly impossible for a time of period. Our work has reduced the execution time of blowfish algorithm and also at the same time the level of security is increased due to H function. The algorithm is more efficient in energy consumption to reduce the consumption of battery power device.

6. References

[1]. B. Schneier, Description of a new variable-length key, 64-bit block cipher (blowfish), Fast software encryption, Cambridge security workshop proceedings(December 1993), Springer-Verlag, 1994, pp. 191-204.

[2]. Charles. P. Pfleeger, "Security in computing", fourth edition, prentice hall, October 2006, ISBN-13: 978-0-13-239077-4.

[3]. Internet System Consortium http://ftp.isc.org/www/survey/report/current/

[4]. W. Stallings, "Cryptography and Network Security", fourth edition, prentice hall, 2005.

[5]. B. Schneier, "Applied Cryptography: protocols, algorithms and source code in C", second edition, john wiley & sons, 1996.

[6]. B. Schneier, The Blowfish Encryption Algorithm Retrieved October 25, 2008, http://www.schneier.com/blowfish.html

[7]. "Blowfish Encryption Algorithm.pdf" from pocket Brief, http://www.pocketbrief.net/related/BlowfishEncryption.pdf

[8]. "Inverter Logic Gate" http://en.wikipedia.org/wiki/Logic_gate