Design and Implementation of Proposed B-R Encryption Algorithm

Janan Ateya Mahdi

Received on: 27/6 /2006
Accepted on: 10/6/2009

Abstract

In this paper, a 128-bit block cipher B-R encryption algorithm is proposed which is an evolutionary improvement of 64-bits Blowfish designed to meet the requirements of the Advanced Encryption Standard. It is compact, speedy, simple and easy to understand. It is a Feistel network, iterating a simple encryption function 16 times and the key can be any length up to 128 bytes. The B-R reduced the memory requirement by using two S-boxes each of which of size 259 bytes instead of four S-boxes each of size 1024 bytes in Blowfish algorithm ,so that the total size of S-box of B-R is 518 bytes while the size of S-box of Blowfish is 4096 bytes, without compromising security. Using several techniques will increase the security of B-R algorithm. Block size and key length were increased and RC6 as complex function was used after the F-function to avoid a symmetric to the output of S-boxes A desirable property of an encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext. This is so called Avalanche Effect. An average of Avalanche Effect of proposal B-R is about 65.42857 % if only one bit in the key is changed while for Blowfish the average of Avalanche Effect is about 28.714286. On the other hand the average of Avalanche Effect is about 60.7143 in RC6 Algorithm.

تصميم وتطبيق خوارزمية التشفير B-Rالمقترحة

الخلاصة

في هذا البحث ,اقترحت خوارزمية تشفير B-R وهي تشفير كتلي ذات 128-bits تحسين تطويري إلى 64-bits Blowfish صممت لتحقيق متطلبات معيار التشفير المتقدم(AES) . وهي فوية, سريعة, بسيطة وسهلة الفهم. وهي شبكة Feistel تكرر وظيفة بسيطة 16 مرة ومفتاح يمكن ان يكون اي طول الى 128−bytes. B-R تقلل من متطلبات ا لذاكرة بواسطة استخدام صندوقيين (S-boxes) كل منها ذات حجم 259−bytes بدلا من أربعة صناديق حجم كل منها 1024−bytes في خوارزمية Blowfish لذلك فان الحجم الكلي لصندوق S-box هو518 bytes بينما حجم الصندوق S-box هو 1024bytes في Blowfish وبدون مساومة الأمنية. استخدام عدة تقنيات سوف يزيد من الأمنية لخوارزمية B-R : زيادة حجم الكتلة وطول المفتاح واستخدام RC6 كدالة معقدة بعد دالة F لتجنب التماثل إلى ناتج الصندوق(S-box). الخاصية المطلوبة في خوارزمية التشفير هي أي تغير بسيط في النص الصريح او المفتاح يجب أن يولد تغير ملحوظ في النص المشفر هذا يسمى Avalanche Effect . معدل Avalanche Effect في الخوارزمية المقترحة تقريبا 65.42857% إذا تغير bit واحد في المفتاح بينما معدل Avalanche Effect لخوارزمية Blowfish هو تقريبا 28.714286 . من جهة أخرى فان معدل Avalanche Effect هو 60.7143 في خوارزمية RC6 .

1. Introduction

Symmetric-key block ciphers have long been used as a fundamental cryptographic element for providing information security. Although they are primarily designed for providing data confidentiality, their versatility allows them to serve as a main component in the construction of many cryptographic systems such as pseudorandom number generators, message authentication protocols, stream ciphers, and hash functions. There are many symmetric-key block ciphers which offer different levels of security, flexibility, and efficiency. Among the many symmetric-key block ciphers currently available, some (such as DES, RC5, CAST, Blowfish, FEAL, SAFER, and IDEA) have received the greatest practical interest [1].

Most symmetric-key block ciphers (such as DES, RC5, CAST, and Blowfish) are based on a "Feistel" network construct and a "round function". Different round functions provide different levels of security, efficiency, and flexibility. The strength of a Feistel cipher depends heavily on the degree of diffusion and non-linearity properties provided by the round function. Many ciphers (such as DES and CAST) base their round functions on a construct called a "substitution box" (s-box) as a source of diffusion and non-linearity. Some ciphers (such as RC5) use bit-wise data-dependent rotations and a few other ciphers (such as IDEA) use multiplication in their round functions for diffusion [2].

This paper present some existing modern cipher such as ABC a substitution-permutation network comprising 17 rounds with 3 different kinds of round functions. It is derived from MMB and SAFFER block cipher [3] and Unbalanced Feistel Networks and Block-Cipher Design (UFNs) consisting of a series of rounds in which one part of the block operates on the rest of the block [4]. And also Hybrid SRC encryption algorithm is designed to take advantage of the power which is supported by Serpent and RC6 algorithms overcom their weaknesses, resulting in a much improved security/performance tradeoff over existing ciphers [5].

To meet the requirements of the AES, a block cipher must handle 128-bit input/output blocks. While Blowfish is an exceptionally fast block cipher, extending it to act on 128-bit blocks in the most natural manner would result in using two 64-bit working registers. Blowfish is a variable-length key block cipher. It does not meet all the requirements for a new cryptographic standard. Its large memory requirement makes it infeasible for smart card applications.

A symmetric-key block cipher, called B-R will be presented, with a block size of 128 bits and a variable key size, ranging from 8 to 128 bytes. The B-R cipher uses a variety of operations to provide a combination of high security, high speed, and implementation flexibility. The main theme behind the design of B-R is to get the best security/performance tradeoff by

utilizing the strongest techniques available today for designing block ciphers.

The design of B-R began with a consideration of Blowfish and RC6 as a potential candidate for an AES submission. Modifications were then made to meet the AES requirements, to increase security, and to improve performance. The outer loop, however, is based around the same found in Blowfish. B-R will be intentionally designed to be extremely simple, to invite analysis shedding light on the security provided by extensive use of key dependent S-box like Blowfish.

2. Blowfish algorithm

Blowfish is a block cipher that encrypts data in 8-byte blocks. The algorithm consists of two parts: a key-expansion part and a data-encryption part. Key expansion converts a variable-length key of at most 64 bytes (512 bits) into several subkey arrays totaling 4168 bytes. These keys must be precomputed before any data encryption or decryption [6].

The P-array consists of 18 32-bit subkeys and also four 32-bit S-boxes with 256 entries each.

The following algorithm shows the Blowfish algorithm structure of 16-rounds. The input is a 64-bit data element, X, which is divided into two 32-bit halves: XL, XR

For I= 1 to 16:
XL = XL XOR Pi
XR = F (XL) XOR XR
swap XL and XR

After the sixteenth round, swap XL and XR again to undo the last swap. Then,
XR = XR XOR P17 and
XL = XL XOR P18.
Finally, recombine XL and XR to get the ciphertext.

2.1 Function F

Figure (1) shows the Function where XL is divided into four eight-bit quarters: a, b, c, and d. Then,

$$\mathbf{F(XL) = ((S1,a + S2,b \bmod 2^{32}) XOR S3,c) + S4,d \bmod 2^{32}.}$$

Where: S1, S2, S3, S4: array of 256 32-bit
a: first 8 bits of XL b: second 8 bits of XL
c: third 8 bits of XL d: fourth 8 bits of XL
Decryption is exactly the same as encryption, except that P1, P2,……., P18 are used in the reverse order [7].

2.2 Subkeys Generation

The subkeys are calculated using the Blowfish algorithm. The flow chart of key generation is shown in figure (2).

. RC6 Algorithm:

RC6 is a fully parameterized family of encryption algorithms [8]. A version of RC6 is more accurately specified as RC6-w/r/b where the word size is w bits, encryption consists of a nonnegative number of rounds r, and b denotes the length of the encryption key in bytes. Since the AES submission is targeted at w = 32 and r = 20.When any other value of w or r is intended in the text, the parameter values will be specified as RC6-w/r. Of particular relevance to the AES effort will be the versions of RC6 with 16-, 24-, and 32-byte keys. For all variants, RC6-w/r/b operates on units of four

w-bit words using the following six basic operations.

- a + b integer addition modulo 2^w
- a - b integer subtraction modulo 2^w
- a ⊕ b bitwise exclusive-or of w-bit words
- a × b integer multiplication modulo 2^w
- a<<<b rotate the w-bit word a to the left by the amount

Given by the least significant lgw bits of b

a>>>b rotate the w-bit word to the right by the amount given by the least significant lgw bits of b.

RC6 works with four 32-bit registers A;B;C;D which contain the initial input plaintext as well as the output ciphertext at the end of the encryption. The first byte of plaintext or ciphertext is placed in the least-significant byte of A; the last byte of plaintext or ciphertext is placed into the most-significant byte of D. Use the (A; B; C; D) = (B; C; D; A) to mean the parallel assignment of values on the right to registers on the left.

4. The Proposed System

B-R block cipher will be introduced. B-R is an improvement of Blowfish designed to meet the requirements of the Advanced Encryption Standard. Like Blowfish, B-R makes essential use of Feistal network. New features of B-R include the use of new F-function and new design of S-box

The philosophy of Blowfish is to exploit operations that are efficiently implemented on modern processors. B-R continues this trendand takes advantage of the Feistal network now efficiently implemented and easy to understand. The new design of F-function in B-R algorithm causes much faster diffusion than Blowfish. This also allows B-R to run with fewer rounds at increased security and with increased throughput.

4.1 B-R Algorithm

The algorithm consists of two parts: a key-expansion part and a data-encryption part. Key expansion converts a variable-length key of at most 128 bytes into several subkey arrays totaling 646 bytes.

B-R has 16 rounds. Each round consists of a key-dependent permutation, and a key- and data-dependent substitution.

Subkeys:

B-R uses a large number of subkeys. These keys must be precomputed before any data encryption or decryption. The P-array consists of 16 64-bit subkeys:
P1, P2,..., P16.
There are also two S-boxes with 32 entries each:
S1,0, S1,1,..., S1,258;
S2,0, S2,1,..,, S2,258;

4.2 Encryption and Decryption

Blowfish algorithm has large lookup table so that doubling it makes performance not free in software and hardware. B-R will be doubled Blowfish to 128-bits but using another F-function to overcome Blowfish restrictions.

Feistal network which makes up the body of the B-R is designed to be as simple as possible, while still retaining the desirable cryptographic properties of the structure.

Figure (3) illustrates the architecture of the B-R algorithm with 16-rounds. The input is a 128-bit data element, P, and the output is the ciphertext C.

Thus the cipher may be formally described by the following algorithm:

The Proposed B-R Algorithm

Input: Plaintext 128-bits P.
Output: Ciphertext 128-bits C.
Begin:
Split plaintext into two 64-bit halves: XL, XR
For I= 1 to 16 Do
XL=XL ⊕ P[I]
XR = F (XL) ⊕ XR
Combined XL and XR into X.
Split X into four 32-bits A, B, C and D respectively.
t= (B × (2B +1)) <<<5
u= (D × (2D +1)) <<<5
A= (A⊕t) <<< u) + P [I] // first 32 bits of P[I]
C= (C⊕u) <<< t) + P [I] // second 32 bits of P[I]
Temp=A
A=B
B=C
C=D
D=Temp
Combined A, B, C, D into X
Split X into XL and XR
Swap XL and XR
Next I
Recombine XL and XR into X.
End.

Decryption is exactly the same as encryption, except that P1, P2,……., P16 are used in the reverse order. The encryption and decryption algorithms have the same complexity.

4.3 The Function F

In Figure (4),the F-function of the proposed algorithm has 64-bit input XLi, 64-bit input subkey P_i and 64-bit output Z .In the proposed B-R algorithm XL_i divides into eight 8-bit a, b, c, d, e, f, g and h while The previous Blowfish uses each quarter as index to one of the S-boxes, so that it uses four S-boxes while in the proposed algorithm each four quarters uses one common S-box. And consequently the number of S-boxes is reduced from four to two.

The entries are overlapped in each of the two S-boxes: entry 0 would consist of byte 0 through 7, entry 1 through 1 to 8, etc. This simplification would reduce the memory requirements for the four S-boxes from 8192-byte (2^8×8 ×4) to two S-boxes of 259 bytes. Additional steps will be required to eliminate the symmetries when different bytes of the input are equal, or when the 64-bits input to function F is a bytewise permutation of another 64-bits input. A more complex combining function in the proposed system is used to eliminate the symmetries.

Function F of proposed algorithm

Input: XL {the length of XL is 64-bits}
Step1: Divide XL into eight 8-bits: a, b, c,d,e,f,g and h.
Step2: Store eight 8-bits in index [8], where index [1] =a, index [2] =b,..etc
Step3: For i: =1 up to 4 Do

X1[i]=0
For j:=0 up to 7 Do
X1[i]=X1[i] OR (S1[index[i]+j])
ROTATE_RIGHT (X1[i], 8)
Next j : Next i
For i: =1 up to 4 Do
X2[i]=0
For j=0 up to 7 Do
X2[i]=X2[i] OR (S2[index[4+i]+j])
ROTATE_RIGHT (X2[i], 8)
Next j : Next i

$Z1=(((X1[1] + X1[2]) \text{ MOD } 2^{32}) \text{ ^ } X1[3]) + X1[4]) \text{ MOD } 2^{32}$

$Z2=(((X2[1] + X2[2]) \text{ MOD } 2^{32}) \text{ ^ } X2[3]) + X2[4]) \text{ MOD } 2^{32}$

Combined Z1, Z2 into Z.
Output: Z {the length of Z is 64 bits)

Example

If XL (64 bits) contains 0x11083aeb47809123 in hexadecimal then it divided into 8 8bits:
a=0x11, b= 0x08, c= 0x3a, d= 0xeb, e= 0x47, f= 0x80, g= 0x91 and h= 0x23

$Y1=((S\text{-}Box1[11] + S\text{-}Box1[08] \text{ MOD } 2^{32}) \text{ XOR } S\text{-}Box1[3a]) + S\text{-}Box1[eb] \text{ MOD } 2^{32}$.

$Y2=((S\text{-}Box2[47] + S\text{-}Box2[80] \text{ MOD } 2^{32}) \text{ XOR } S\text{-}Box2[91]) + S\text{-}Box2[23] \text{ MOD } 2^{32}$.

Where S-Box1, S-Box2 are a array [0...259] of unsigned long (3 bits) and Y1, Y2 are 32bit registe Y=combined Y1 and Y2 where Y is 64 bits register which is the output of the F-function

The non-reversible function is designed for strength, speed, and simplicity. The function that combines the two S-box outputs is as fast as possible. Function F is the primary source of the algorithm security, so that a more complicated reversible function is required.

4.4 One Round of RC6

In the B-R algorithm one round of RC6 is applied as a reversible mixing function after F function which is key dependent permutation. It is used to overcome a Feistel structure weakness that each round transformation always keeps one half of the block constant so that the RC6 algorithm would further confuse the entry values into the Feistel network and ensure a complete avalanche effect after the first two rounds.

The RC6 algorithm is required to provide the necessary diffusion and confusion to the input block, such that additive differences will be destroyed as the key is changed. This could provide a protection against linear and differential cryptanalysis. Figure 4 illustrates the algorithm of one round of
RC6 which has 128-bit input and 128-bit output [8].

The algorithm of one round
Input: A, B C, and D of 32 bits
Begin:

$t = (B \times (2B + 1)) \lll 5$
$u = (D \times (2D + 1)) \lll 5$
$A = ((A \oplus t) \lll u) +$ P[i] // **first 32 bits of P[I]**
$C = ((C \oplus u) \lll t) +$ P[i]// **second 32 bits of**

Temp=A
A=B
B=C
C=D

D=Temp

End.

Output: A, B C, and D of 32 bits

B-R increased the complexity of algorithm by using combinations of basic operations addition, Ex-oring and multiplication. The usage of multiplication in RC6-function has to do with our ability to analyze them: Analyzing a multiplication of two data words turns out to be a very hard task [9].

4.5 Key generation

The B-R algorithm uses variable key size from 8 byte up to 128 bytes. The subkeys are calculated using the B-R algorithm like Blowfish, so that the complexity of key generation is dependent on the B-R algorithm

The number of rounds is 16 and this number affects the size of the P-array and therefore the subkey-generation process; 16 iterations permits key lengths up to 5168 bits (646 *8).

There areTwo basic ways to ensure that the key is long enough to ensure a particular security level. One is to carefully design the algorithm so that the entire entropy of the key is preserved, so there is no better way to crypanalyze the algorithm other than brute force. The other is to design the algorithm with so many key bits that attack that reducing the effective key length by several bits are irrelevant. Since Blowfish is designed for large microprocessors with large amounts of memory, then one of the aims of the proposed system is to reduce the memory requirement [7]

The range of values, which a key will take, becomes large, where a large key space is necessary to prevent exhaustive search for a key (Solving the problem of finding the correct value for a key by testing possible values until the correct One is found).

The proposed system still uses the same key generation process of the Blowfish because it is designed to preserve the entire entropy of the key and to distribute that entropy uniformly throughout the subkey. It is also designed to distribute the set of allowed subkeys randomly throughout the domain of possible subkey.

The P-array and S-boxes must be precomputed before any data encryption or decryption. The strength of various FN systems (specially the resistance of FN against Differential Cryptanalysis and Linear Cryptanalysis) is tied directly to the architecture of the S–boxes of FN [7].

S-boxes are either fixed for all keys or key dependent. Hence those ciphers with key-dependent S-box are, in general, more secure than fixed S-boxes. There are two different philosophies regarding key dependent S-boxes. In some ciphers, the S-box is constructed specifically to ensure that no two entries are identical e.g. KHUFU and WAKE while others simply create the S-box randomly and hope for the best e.g. REDOC II and Blowfish. Most key-dependent S-boxes are created by some process completely orthogonal to the underlying cipher. Blowfish uses repeated iterations of itself. The results are S-boxes that are effectively randomn, but the cost is an enormous performance penalty in key-setup time [7].

While fixed S-boxes must be designed to be resistante to

differential and linear cryptanalysis, key-dependent S-boxes are much more resistante to these attacks. The B-R uses key dependent S-boxes like Blowfish with reduce the key-setup time by reducing the size of S-boxes by using the two S-boxes of size 259 bytes instead of four S-boxes of size 1024 bytes boxes.

Linear and differential cryptanalysis works only if the analyst knows the composition of the S-boxes. If the S-boxes are key-dependent and chosen by a cryptographically strong method, then linear and differential cryptanalysis are much more difficult. Since the structure of the S-boxes is completely hidden from the cryptanalyst, these attacks have a more difficult time exploiting than structure. Additionally, these S-boxes are easier to implement and can be created on demand, reducing the need for large data structures stored with the algorithm [7].

5 Security of B-R algorithm

In this section a B-R algorithm and each function that it performs will be evaluated. Some principles that are useful will be discussed to show that this B-R algorithm will be adapting the AES requirement.

The B-R algorithm adapts Blowfish and Serpent algorithm to meet the AES requirement and overcomes the weaknesses of these algorithms by the following:

1. Increased the security of the original Blowfish algorithm by using block size of 128-bits instead of 64 bits
2. The time-consuming subkey-generation process of Blowfish algorithm adds considerable complexity for a brute-force attack. Furthermore, a total of 48 iterations of the encryption algorithm are required to test a single key, so that the B-R algorithm uses the same key generation of Blowfish algorithm.
3. The size of S-box of B-R is 518 bytes while the size of S-box of Blowfish is 4096 bytes. It can be possible to implement the algorithm on smart cards. The requirement for small processors the most difficult RAM and ROM limitations are severe for this platform. Also, efficiency is more important on these small machines.
4. Each function of the proposed algorithm is dependent on its key, so this will prevent fixed output and increase the non-linearity of the algorithm.

5.1 An Attacker of the B-R Algorithm

A. Differential and linear cryptanalystsis

Differential and linear cryptanalystsis are worked against block cipher algorithms that use constant S-boxes [10]. These attacks are heavily dependent on the structure of S-boxes. The B-R algorithm is patient to these types of attacks because the key-dependent S-box is used and this could provide protection against linear and differential cryptanalysis. Since the structure of the S-boxes is completely hidden from the cryptanalyst, these attacks have a more difficult time exploiting that structure. Key-dependent S-boxes are easier to implement and less

susceptible to arguments of "hidden" properties. Additionally, these S-boxes can be created on demand, reducing the need for large data structures stored with the algorithm.

B. Dictionary Attacks

As the block size is 128 bits, a dictionary will require 2^{128} different pliantexts to allow the attacker to encrypt or decrypt arbitrary message under an unknown key. This attack applies to any deterministic block cipher with 128-bit blocks regardless of its design [2]. So that the B-R algorithm used 128 bits block size while the Blowfish is 64 bits block size.

C. Key-Collision Attacks

For key size *k*, key collision attacks can be used to forge messages with complexity only $2^{k/2}$. Thus, complexity of forgoing messages under 64-bit keys is only 2^{32}, and under 128-bit keys it is 2^{64}. This attack applies to any deterministic block cipher, and depends only on its key size, regardless of its design [2]. The maximum key length of B-R algorithm is 128 bytes (1024 bits) so that the complexity of messages under the control of 1024 bits is 2^{1024}. On the other hand the complexity of Blowfish under the control of 512 bits is 2^{256}.

5.2 Avalanche Effect

This section is prepared for making a statistical test on the ciphertext that is produced from encrypting the plaintext

1111111111111111
2222222222222222
3333333333333333
4444444444444444
5555555555555555
8888888888888888

And using a key of length 32 bytes where the key is ***"30000000000000000000000000000000"***.

Horst Feistel referred to the avalanche effect as [11]: "*a small change in the key gives rise to a large change in the ciphertext".* Table (1) represents the avalanche effect on the ciphertext when only one bit is changed on the key and performing the B-R algorithm. From the table the average of avalanche effect is computed and equal to % 65.42857

Table (2) represents the avalanche effect on the ciphertext when only one bit is changed on the key and performing the RC6 algorithm. From the table the average of avalanche effect is computed and equal to 60.7143

Table (3) represents the avalanche effect on the ciphertext when only one bit is changed on the key and performing the Blowfish algorithm. From the table the average of avalanche effect is computed and equal to 28.714286

6. Memory requirement

One of the aims of the B-R algorithm is to reduce the memory requirement without compromising security. Hence the number of byte of S-box of B-R is 518 bytes. Table (4), illustrates the memory requirement for the S-box of various types of block cipher algorithms.

7. Time requirement

In this section the time requirements are computed for the B-R algorithm and comparisons between the B-R, Blowfish and RC6 algorithms are taken. Figure (5) show the curve speed comparison of B-R and Serpent and Blowfish algorithms.

8.Test vector

The following example shows the encryption and decryption output for each round of the proposal B-R encryption algorithm using plaintext (represented in hexadecimal) and key (64-bytes) which are shown in table (5):

key="00000000000000000000000000000000000000000000000000000000000000000000000000000000"

plaintext=00000000000000000000000000000000

Conclusion

During the design process, several things can be concluded about cipher design:

- The proposed B-R algorithm is a Feistel network so it is easy to implement and understands.

3. B-R uses the same algorithm for encryption and decryption with the same key schedules, and supports variable key-length up to 128 bytes.
4. B-R has a low memory requirement (the S-boxes just uses 518 bytes), so it may be easily implemented on smart cards or other devices with restricted memory.

6. There is no such thing as a key-dependent S-box, only a complicated multi-stage nonlinear function that is implemented as a key-dependent S-box for efficiency.
7. Mixing operation from different algebraic groups. XOR, addition, and multiplication.

8. Because of the large key and input block size of the proposed algorithm, exhaustive key search and the matching ciphertext attack are infeasible.

9. From the results that were obtained in section 3 after measuring the strength of the proposed algorithm we can conclude that the proposed algorithm increases the security and reduce the memory requirement compared with Blowfish and RC6 algorithms.

References

[1] Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. M. Matyas, L. O'Connor, M. Peyravian, D. Safford and N. Zunic, "Mars a candidate cipher for AES", *First Advanced Encryption Standard (AES) Conference*, Ventura, CA, 1998.

[2] R. Anderson, E. Biham, and L. Knudsen, "Serpent: A Proposal for the Advanced Encryption Standard, " *First Advanced Encryption Standard (AES) Conference*, Ventura, CA, 1998.

[3] Dieter Schmidt "*ABC - A Block Cipher*", Wikipedia the free Encycleopedia, May 27, 2002.

[4] Bruce Schneier and John Kelsey," *Unbalanced Feistel Networks and Block Cipher Design", Counterpane* Systems, 101 East Minnehaha Parkway, Minneapolis, MN 55419, 2005, http://fschneier,kelseyg@counterpane.com

[5] Dr. Salah M. Al-qaarawy and Ashwaq T. hashim""hybrid SRC Encryption Algorithm", Science and Technology: Journal, vol 6 .p 26..

[6] Bruce Shnier "Applied Cryptography Second Edition Protocols. Algorithms, and Source, and Source Code in C", *John Wiley and Sons, Inc.*, 1996.

[7] Fast Software Encryption Cambridge Security Workshop Proceedings (December 1993), Springer-Verlag, pp. 191-204 1994.

[8] R. Rivest, M. Robshaw, R. Sidney, and Y. Yin, "The RC6 TM Block Cipher," *First Advanced Encryption Standard (AES) Conference*, Ventura, CA, 1999.

[9] Thilo Zieschang, " Combinatorial Properties of Basic Encryption Operations", Advances in Cryptology Eurocrypt'97, *International Conference on the Theory And Application of Cryptographic Techniques Konstanz, Germany, May 11-15, 1997 Proceedings, Springer*, 1997.

[10] Ian Harvey," the Effect of Multiple Algorithms in the Advanced Encryption Standard", *An internet Survey*, 4th January 2000.

[11] Shakir M. " A new feedback symmetric block cipher method", *Ph. D , Thesis Univ. of tech, Baghdad*, 1997.

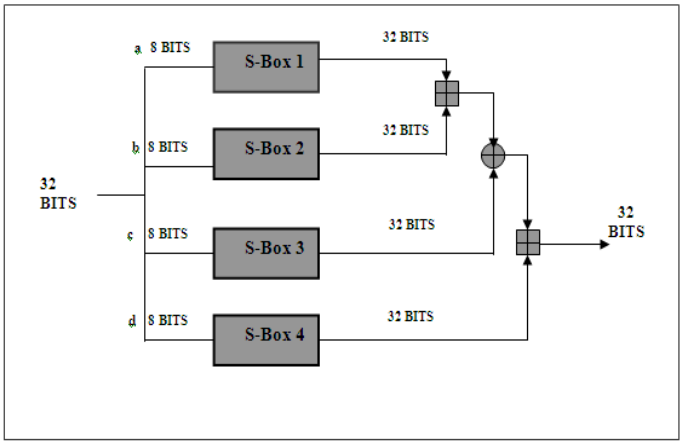

Figure (1) The F-function of Blowfish algorithm

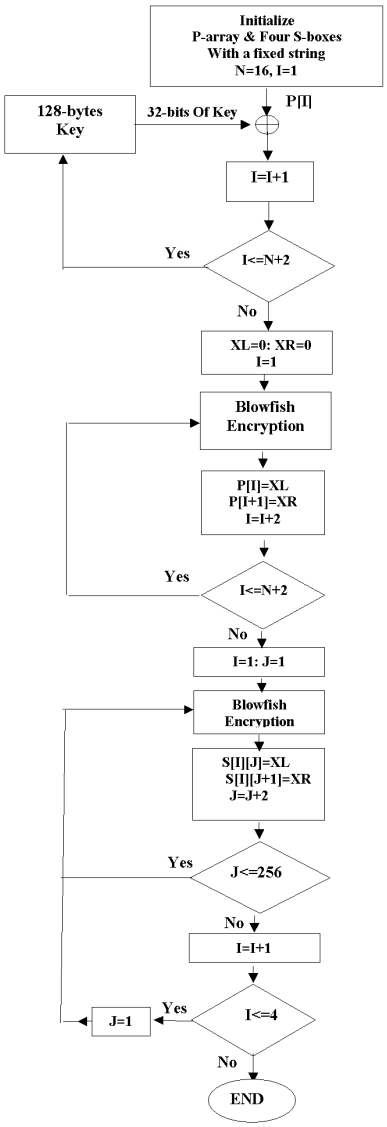

Figure (2) The Flow chart of key generation of Blowfish

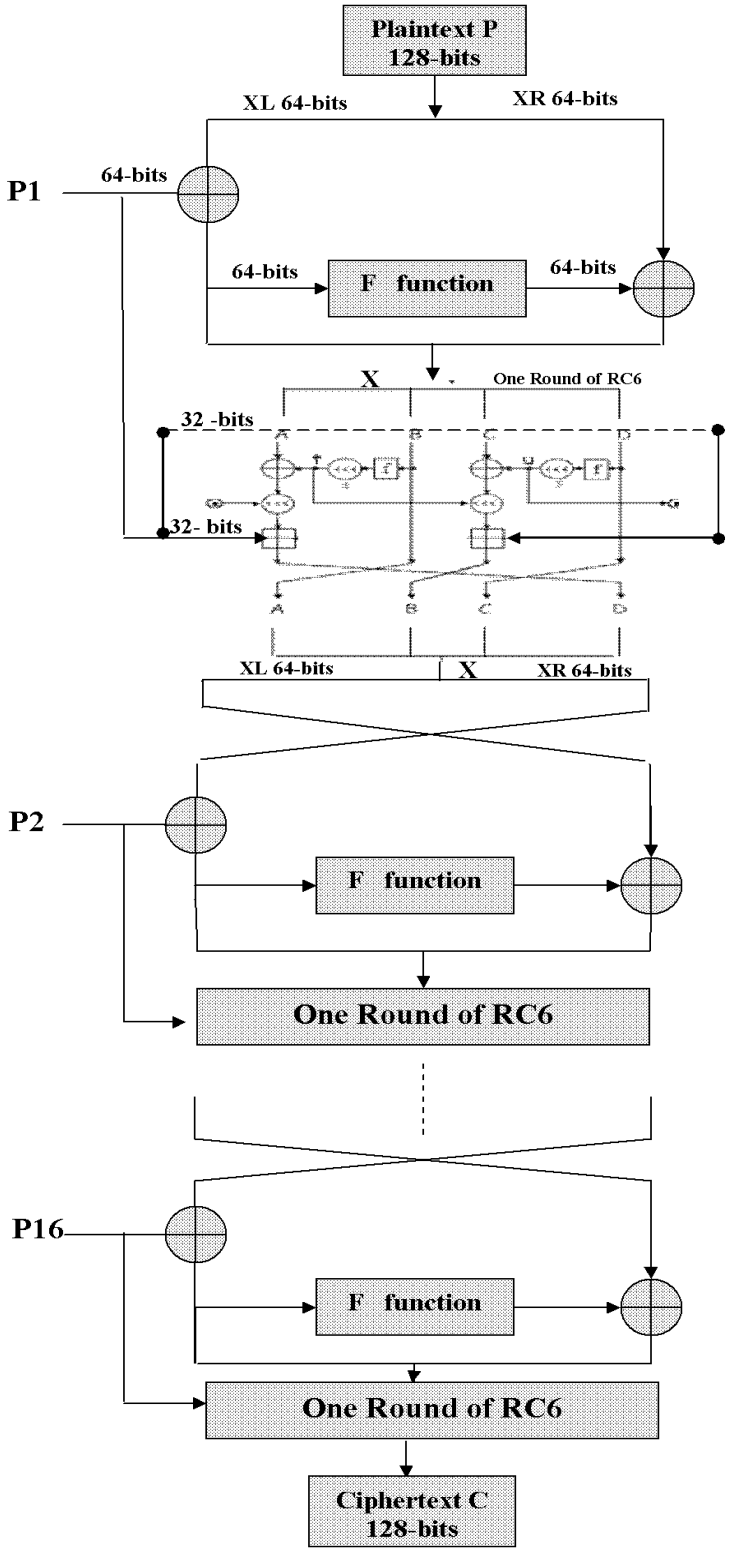

Figure (3) The B-R Algorithm

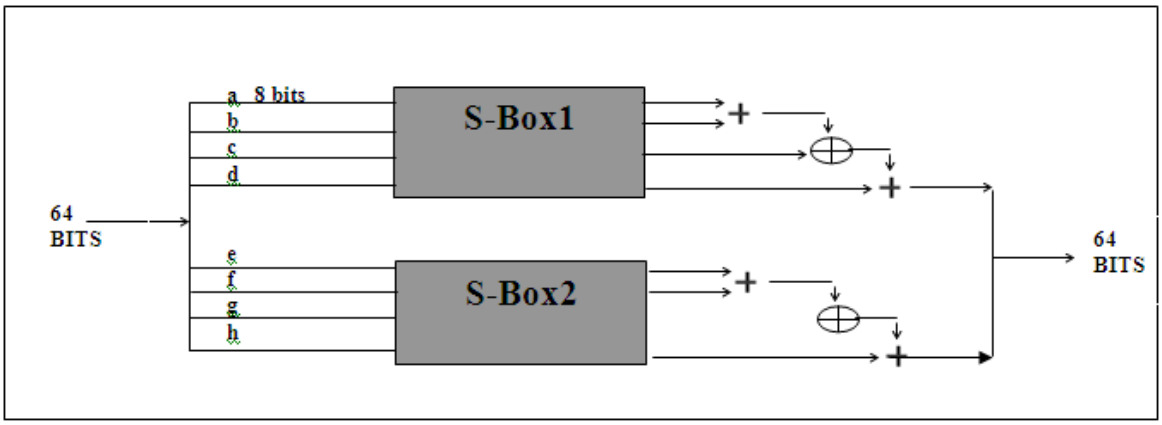

Figure (4) The F-function of Proposed B-R algorithm

Table (1) Avalanche effect of B-R on the ciphertext when Only one bit is changed on the key

	Ciphertext 128-bits in Hexadecimal	Avalanche
1	E27B407E705134BBd90e4fdfc4130cec CC5630CD1864195B2672392ec43b451d	63
2	1EAD97501CC80087935fcfafb9a4b6ce 85346519EF925D70e7a13bb87ae617ed	71
3	A337B77F9475EA60e1e60de59d3a53c8 E6441BB67CC485D529aef73bb0d868fa	68
4	74FF2FAD694A37A06f6b00d93fbd749d B16EB3B57C624F04515f53801d9d52c3	52
5	21F170C66BEA155A86978ecd8b2c28bf CBE8DFAABED6EDE1a9e2437a9b39926d	72
6	B15969DE7F83275Aea598272fcbb84cd B5F39784677954407e3113334a186199	60
7	3828A698A1E1D5DA98c56b2f2c9bc847 54B259F10C3060FC3c449791e88c9388	72
Key1	30000000000000000000000000000000	
Key2	20000000000000000000000000000000	

Table (2) Avalanche effect of RC6 on the ciphertext when Only one bit is changed on the key

	Ciphertext 128-bits in Hexadecimal	Avalanche
1	41065260443119991376178141999232634233774243421416156784827940316119121750681124	56
2	29912427951632039507368659598733559713594244481511174227221338686122421526017054	56
3	32246702225354839007999777362736516111231091862727109577271853142317125982248 9	57
4	18647718723203136746256996728739582922471067685432418808500825435554542334463388	62
5	38399709142965467950982470071182474528415867228111810744256129038973616213854 6	72
6	14936259534237752172564486294405213107735383472853344259021212262782246890405	60
7	18774570732646065449362982169328284508992803692638259496060912558900612515466287	62
Key1	30000000000000000000000000000000	
Key2	20000000000000000000000000000000	

Table (3) Avalanche effect of Blowfish on the ciphertext when Only one bit is changed on the key

	Ciphertext 128-bits in Hexadecimal	Avalanche
1	84be747ce0eb025 de43f7cc977cc136	32
2	84be747ce0eb025 de43f7cc977cc136	32
3	6e0d8d148ded8f0 9799158c5d8293	27
4	8c6572a5f769eaf5 515a0dad24a62ebf	37
5	bb90a55f7e3906dc 2947ff6ac58187f0	32
6	8b132f52dd8a32fe 2b23739143733bf5	28
7	3cd8404ded07f19f 56fc563dc0913715	27
8	8ad12cb189fdfa89 336a082d6382faf7	35

9	509f96c9f63d14df 18ae18ccc317ced9	25
10	3ea8bd04e477c226 9633bc3e24a7906e	23
11	3ea8bd04e477c226 9633bc3e24a7906e	23
12	ba3a4fc581d2de93 3815eef52b062bc	28
13	c0de2bd94a78455f 85a132c92ad89588	27
14	34bc6bca8e181d26 f2b1d9a146099f17	26
Key1	30000000000000000000000000000000	
Key2	20000000000000000000000000000000	

Table (4) Memory requirement comparisons of S-box

Algorithm	Length of block	Memory requirement in byte
B-R	128-bits	518
Blowfish	64-bits	4096
Serpent	128-bits	512
Twofish	128-bits	1024
Rijndeal	128-bits	512
Mars	128-bits	2048
Triple-DES	64-bits	256

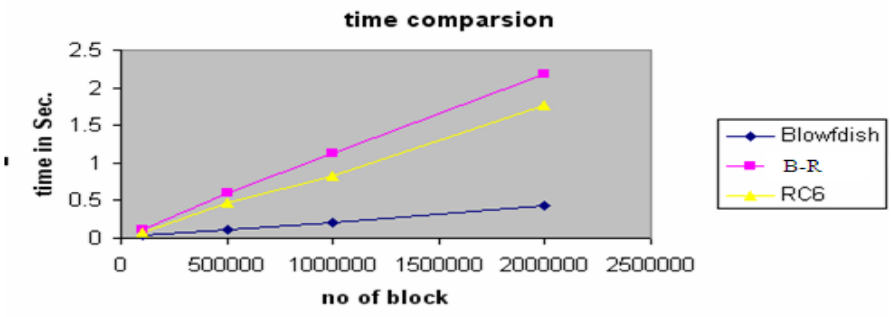

Figure (5) Time comparison of B-R, Blowfish and RC6.

Table (5) Encryption and Decryption step of B-R Proposal Algorithm

Round No.	P Array in Hexadecimal after key generation	Encryption output in Hexadecimal after each round	Decryption output in Hexadecimal after each round
0	E16A03B33F366D92	42bdc6e76eb13fd05d5eb70fa01d6c47	b3635515780e2a7c24eaa41bbca4a10c
1	708075A66CB8DF67	8df883e40915cd32bdc9db36270505a8	ef394df9814fd1b48056dad8d668316a
2	6787B7F6A7BF4705	c7edec9dbdbb4c8979973ca54059524b	adf44f37da389e111e955ad02024c47d
3	AC4245BE8B00E8BB	a76672ac070b245150d9fd250640b7c0	caabd886d775a36a7bf67456853ea9af
4	7CB038CA11D367B7	ec3d12037cb71c49ad7dc8fa72580adc	da51a4f8179a7c2638b3e79f085b5d47
5	B8080C3973207D96	35081e3b244e1489490b7326b5b1202d	998c9d14bad12526633edbfab554ed01
6	A2D611361C1D7FE5	52b1d3a07baff2de5fc470a3422e0aa5	c1232c7cfbaa6a8b79de0ccd3f0cf28e
7	D370335C7F23A4AD	e7b28e85944d0e2cccd918995914a230	fc523bc85575e8c8f0b559568d979b8a
8	13832636C4C4721D	fc523bc85575e8c8f0b559568d979b8a	e7b28e85944d0e2cccd918995914a230
9	5A45C1B58F845C22	c1232c7cfbaa6a8b79de0ccd3f0cf28e	52b1d3a07baff2de5fc470a3422e0aa5
10	F83C8AC22CD503D3	998c9d14bad12526633edbfab554ed01	35081e3b244e1489490b7326b5b1202d
11	274B590CBF35E829	da51a4f8179a7c2638b3e79f085b5d47	ec3d12037cb71c49ad7dc8fa72580adc
12	A60B080879DC0296	caabd886d775a36a7bf67456853ea9af	a76672ac070b245150d9fd250640b7c0
13	8CACFB2B3CACB641	adf44f37da389e111e955ad02024c47d	c7edec9dbdbb4c8979973ca54059524b
14	9E84D5A3644428FE	ef394df9814fd1b48056dad8d668316a	8df883e40915cd32bdc9db36270505a8
15	75E373D55A0D7E7	b3635515780e2a7c24eaa41bbca4a10c	42bdc6e76eb13fd05d5eb70fa01d6c47