The Affect of Genetic Algorithms on Blowfish Symmetric Algorithm

Yousef Bani Awwad[†] and Mohammad Shkoukani[††]
[†] Computer Science, Applied Science Private University, Amman, Jordan
[††]Faculty of Information Technology, Applied Science Private University, Amman, Jordan

Summary
Blowfish (BF) Algorithm is one of the most public and common security symmetric algorithms, it is a Feistel cipher with a sixteen rounds and it uses a large key. A genetic algorithm (GA) is a search algorithm for solving optimization problems due to its robustness. In this paper, an encryption and decryption algorithm has been designed for blowfish algorithm using Genetic Algorithm to make the overall cryptography process much highly secure. The proposed algorithm based on combination of genetic algorithm and blowfish algorithm to increase degree of security and protection of blowfish algorithm. The proposed algorithm applied one point crossover and Flip Bit mutation operator for simplification. The authors have modified the original algorithm in three stages on plain text, Function h and key. Experimental results have been conducted via a simulator that has been designed and developed using C# programming language. The results show the efficiency improvement of the proposed algorithm by increasing security over the original plain text according to the avalanche effect. The results show that when one bit in the key has been changed the average avalanche effect in the original blowfish was 38.63, whereas in the proposed algorithm was 45.21. Also when one bit has been changed in the plain text the average avalanche effect in the original blowfish was 37.17, whereas in the proposed algorithm was 46.44. So the results indicate that the average avalanche effects are better in the proposed algorithm than the original one. However, according to the average execution time, the original Blowfish is 6.6 milliseconds and the execution time of the proposed algorithm is 55.1 milliseconds.

Key words:
Cryptography, Blowfish, Genetic Algorithms, Encryption, Decryption.

1. Introduction

Most of information is kept electronically, so the security of information has become an essential issue. Nowadays Cryptography is a strong tool used to memorize the information in computer systems. Cryptography actually means secret writing, it was available only to generals, but now it almost used by everyone. Such as: when an online payment transactions done, phone call made, secure website is used [1, 2]. Cryptography allow us to hide information that is transferred and stored. In cryptography, the original message is encoded in some non-readable format which is called encryption. On the other hand the person who knows how to decode the message can get the original information which is called decryption [1, 4]. Fig. 1 shows the concept of cryptography.

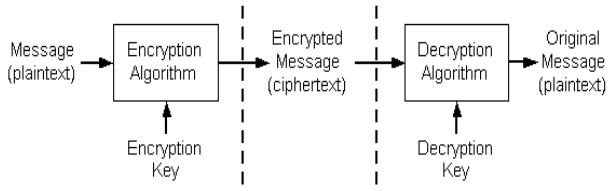

Fig. 1 Concept of Cryptography

There are two main techniques for cryptography which are the symmetric cryptography where the sender and receiver use the same key. The other technique is the asymmetric cryptography where the sender and receiver use two keys, the public key which is used for encryption and the private key which is used for decryption [10, 14].

Blowfish is a variable-length key, 64-bit block cipher. It is a Festal network consisting of 16 rounds. The input is a 64-bit data element. Then the input is divided into two 32-bit halves: LE and RE. Decryption is exactly the same as encryption, except that P1, P2 ... P18 are used in the reverse order as shown in fig. 2.

The Blowfish Function F is the core function that is applied on the left half as follows, Divide left part into four quarters each one include 8 bits: a, b, c, and d and then the operations of XOR and addition mod 2^{32} are performed [3, 9], as shown in fig. 3.

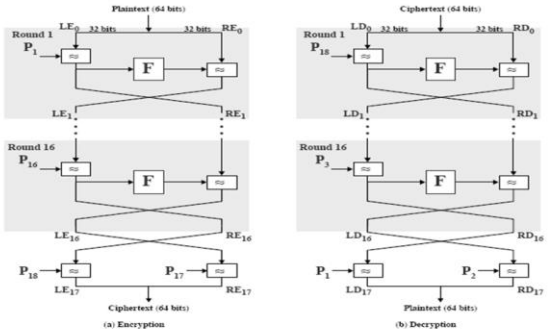

Fig. 2 Blowfish Encryption/Decryption [10]

Manuscript received March 5, 2017
Manuscript revised March 20, 2017

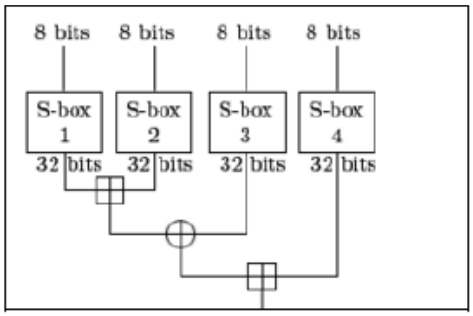

Fig. 3 Blow fish Function F.

Genetic algorithm is one of the most powerful Computer algorithms which generate solutions to optimization problems. GA is proposed by Holland and based upon Darwinian evolution theory. Fig. 4 shows the genetic algorithm processes such as selection where the parents' chromosomes are selected from a population, crossover which is performed to produce a new offspring, and mutation which is performed to mutate new offspring [5, 6, 7].

The authors have modified the blowfish fiestel network by applying the genetic algorithm on the Key F(pi), Function (h) and the plaintext in order to strengthen this blowfish security algorithm.

The affect of using genetic algorithms will be implemented and tested to realize the enhancement that occurs in the security level of blowfish encryption algorithm.

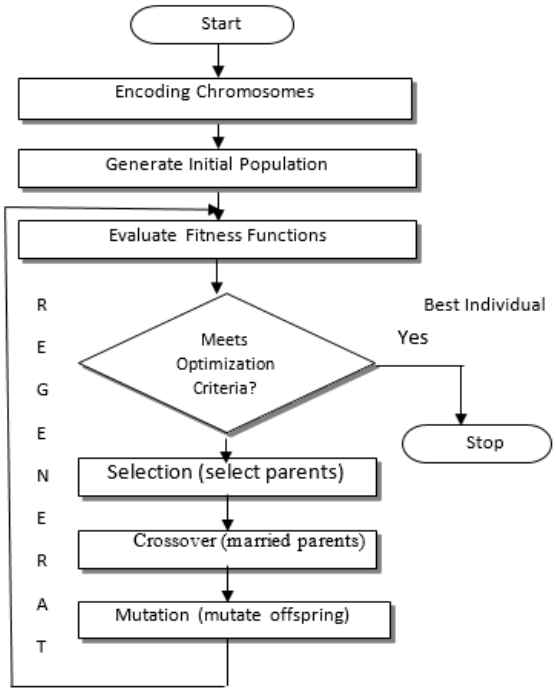

Fig. 4 Genetic Algorithm Processes

2. Related Works

Pradeep Misha and Monika Agrawal improved Blowfish algorithm by increasing its security and decrease both its encryption time and decryption time. They produce a random number between zero and 65535. After that they use a flag with initial value equal to zero. Then Transform the random number into sixteen bits format. All fields with a zero value should modify its flag from zero to one. Finally the F function will execute only if the flag is zero. The main advantage of their modified algorithm is decreasing the execution time rather than the original blowfish algorithm [10].

Saravana and Shanmugam improved the complexity and security of blowfish algorithm by proposing a modified fiestel network with a G function for the blowfish algorithm [11].

Christina and Irudayaraj: they only change the S-boxes in the F-Function and they keep the blowfish algorithm structure without any change. The modified blowfish F-function has two S-boxes instead of four S-boxes as in the original one. The modified algorithm was more secure but the speed of encryption and decryption were slow [12].

3. Proposed Algorithms

The proposed algorithm is based on combination of genetic algorithms and blowfish algorithm in order to increase degree of security and protection of blowfish algorithm, Blowfish is now considered to be insecure for many applications. So it becomes very important to augment this algorithm by adding new levels of security to make it applicable and can be depending on in any common communication channel. Adding additional key and replacing the old XOR by a new operation as proposed by this paper to give more robustness to Blowfish algorithm and make it stronger against any kind of intruding.

Fig. 5 shows that the input of the proposed model is the plaintext which consists of 64 bits, which will be stored in an 8*8 table. After that the table is divided into two 4*8 tables (LT) and (RT), then a crossover rows between (LT) and (RT) tables will be performed and the result will be (LT1) and (RT1) tables. After that XOR process between (LT1) and (RT1) tables will be applied. The output is a 4*8 table; which is the left table part that will be called (A). Again a crossover columns between (LT) and (RT) tables will be performed and the result will be (LT2) and (RT2) tables. After that XOR process between (LT2) and (RT2) tables will be applied. The output is a 4*8 table; which is the right table part that will be called (B).

An XOR operation between left table part (A) and the key (P1) will be applied; the output of this process will be called (X). An XOR operation between the result of (h)

function and right part (B) will be applied and the output will be called (Y). A swapping between (X) and (Y) will be performed.
According to the feistel network and proposed algorithm this will be repeated 16 times, and the final round will be: An XOR operation between (Y) and key (P17) is performed and the output will be called (S1), then an XOR operation between (X) and key (P18) is applied and the output will be called (S2).
Finally, (S1) and (S2) will be merged.

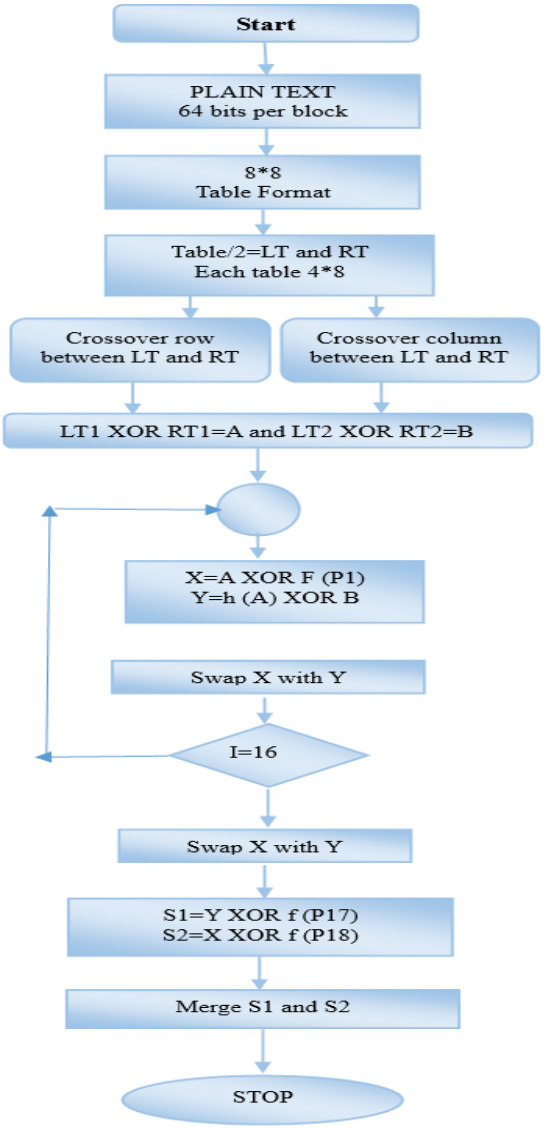

Fig. 5 Proposed algorithm blowfish algorithm with genetic algorithm

Fig. 6 shows the modified algorithm of key (p) which consists of 32 bits that is transformed into a 4*8 table ,which is divided into two 4*8 tables T1 and T2. A crossover by rows is applied between table (T1) and table (T2), the output will be (T11) and (T22). Then a crossover by columns is applied between table (T1) and table (T2), the output will be (T111) and (T222). After that an XOR operation is performed between table (T11) and table (T22), and between table (T111) and table (T222) the output will be called (S1). A mutation by rows is applied between table (T1) and table (T2), the output will be (Tm11) and (Tm22). Then a mutation by columns is applied between table (T1) and table (T2), the output will be (Tm111) and (Tm222). After that an XOR operation is performed between table (Tm11) and table (Tm22), and between table (Tm111) and table (Tm222) the output will be called (S2). Finally an XOR operation will be applied between the two tables (S1) and (S2).

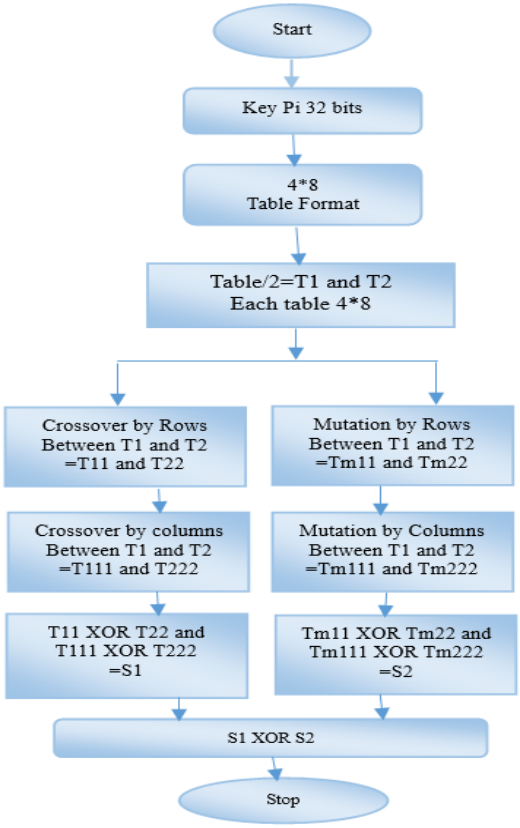

Fig.6 flowchart of modified function F (pi)

Fig. 7 shows the modified (h) function which consists of 32 bits that is transformed into a 4*8 table ,which is divided into four 1*8 tables a, b, c and d. A crossover by rows is applied between table (a) and table (b), the output will be (ab1). Then a crossover by columns is applied between table (a) and table (b), the output will be (ab2). After that an XOR operation is performed between table (ab1) and table (ab2), the output will be called (y). Mutation operation will be applied on table (y) and the output will be called (ym). Again, a crossover by rows is applied between table (c) and table (d), the output will be (cd1). Then a crossover by columns is applied between

table (c) and table (d), the output will be (cd2). After that an XOR operation is performed between table (cd1) and table (cd2), the output will be called (z). Mutation operation will be applied on table (z) and the output will be called (zm). Finally the two tables (ym) and (zm) will be merged.

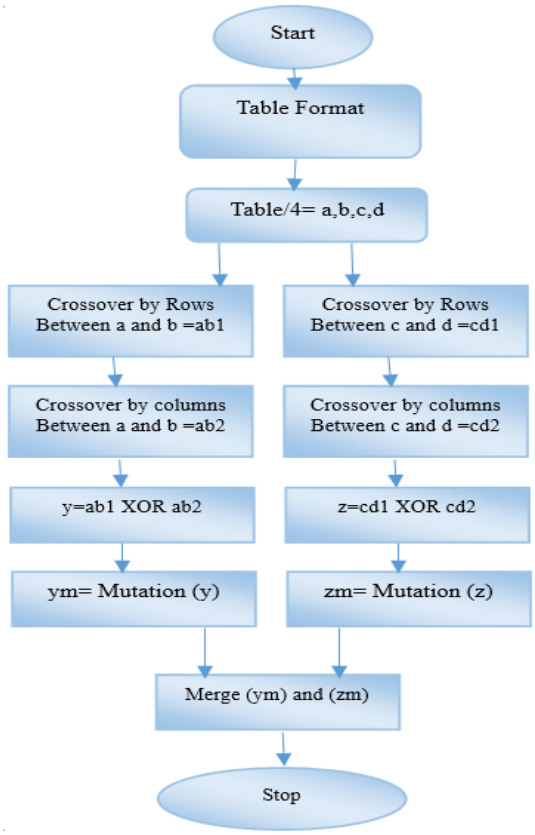

Fig. 7 flowchart of modified function h (A)

4. Simulation

A simulator was developed for testing the proposed Algorithms and the original blowfish algorithm, the developed system used: Microsoft Visual C# 2010 Express Edition and Microsoft .NET Framework 4. The simulation was conducted on HP laptop: core i5, 2.50 GHz, 6 GB RAM, 320 GB HDD, and has Windows 7 professional 64-bit operating system.

The main function of the developed system is to encrypt and decrypt files using proposed algorithm which is Genetic Algorithm with Blowfish and the original Blowfish algorithm with 64-bits block size. Fig. 8 shows the main screen of the developed system, which consists of the main components: key, plaintext, encryption, decryption by original Blowfish algorithm and genetic algorithm with Blowfish.

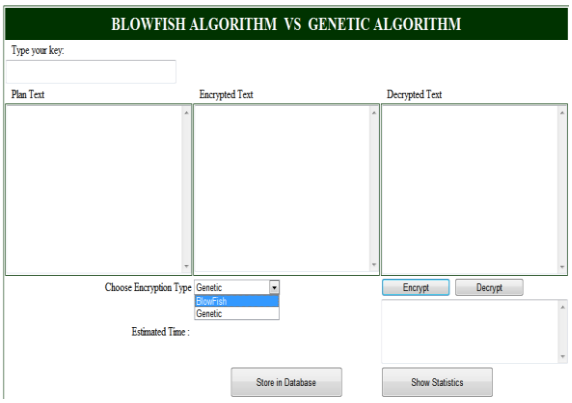

Fig. 8 Main screen of the developed system

Fig. 9 explains the encryption and decryption process using original Blowfish algorithm which includes the plaintext with its encrypted and decrypted text.

Fig. 10 shows the encryption and decryption process using the proposed algorithms (Blowfish algorithm with genetic algorithm) which includes the plaintext with its encrypted and decrypted text.

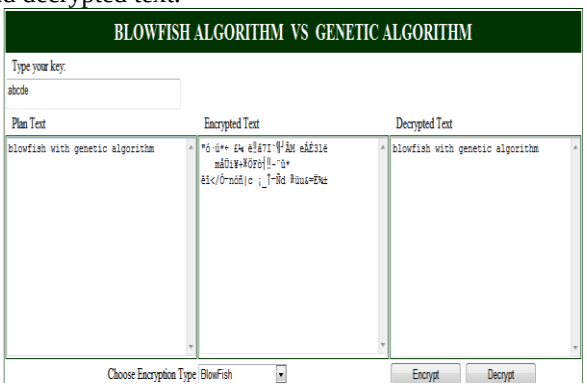

Fig. 9 Encryption/ Decryption process by the original blowfish algorithm

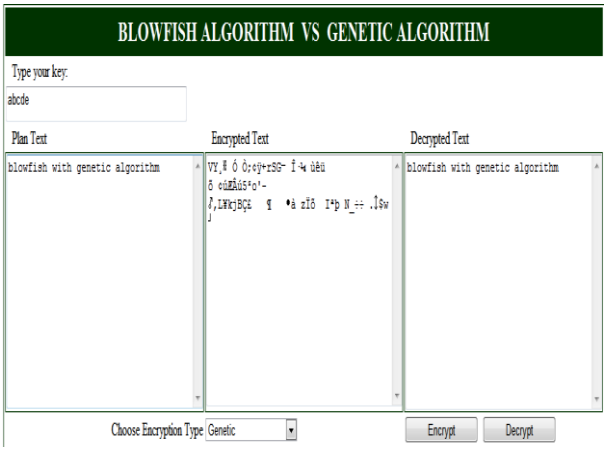

Fig.10 Encryption/ Decryption process by the proposed algorithms (blowfish algorithm with genetic algorithm)

5. Experiment

A comparison between the original Blowfish algorithm and the proposed algorithm (Genetic algorithm with Blowfish algorithm) based on two characteristics: the Avalanche Effect and Execution Time will be done by using the developed system.

5.1 Avalanche Effect

Avalanche effect is an important characteristic for encryption algorithm. This property can be seen when changing one bit in the plaintext and then watching the change to the bits in the cipher text. If an input is changed slightly (for example, flipping a single bit) the output changes significantly (e.g., more than half the output bits flip). One purpose for the avalanche effect is that by changing only one bit in the plaintext there is large change in the cipher text which increases security [8].

Avalanche Effect (%) = (Number of Changed Bits in Cipher text) / (Total number of bits in cipher text) * 100

So when the Avalanche Effect is increased the security in the algorithm is increased.
Table 1 shows the avalanche effect of original Blowfish algorithm on the plaintext when only one bit has been changed in the key by using 192 bits block size.

Table 1: Avalanche Effect of original Blowfish of 192-bits block size on the key

Block no.	***Cipher text 192 _bit in binary***	***avalanche***
1	15fc3edc000000006a8644b003b5ab29099d 9bb0786b5813 5cfcde4000000000e61e63a2b80530b0b6d3 4039c3881a96	39.6%
2	5d196eac000000006130779cb513b3f5eb9d f3eb3d55ccc5 65d8ee9f000000008ed4177cd0e99ce439d7 a4dbec2d5c69	37.5 %
3	25e7183a0000000008fc4055ac9c5e4c4073 b9f2024eb10f 605679f500000000fb2e8c6bf0a05126d10cf e1b5cb6dc3b	36.8%
4	47c521c500000000876f8e7f2ba7c8736f91a a7825be73ea 1811b2c9000000002fb6e32fe96eb24a288e 808eaca074e1	40.6 %
5	2624bd1300000000e3cc6ac9df5735a797cb 67e5d292f927 582a7c8600000000fe0ddf74d9f4dedb4808f 7a37dc5a5b7	38.8 %
6	559b6e7000000000401b6dee2f8aa4cb2244 b8dc899c36d5 3664677a000000004147e1739a03fb8a9f46f cc81f705a30	38.5 %
7	3b16a42900000000bf48fe004e27f9fd30085 fee8240ef60 26648dbc0000000057c4c521b629755fac90 421cc93fb1e0	39.6 %
8	43db5622000000003a60f0632ba63e9d3483 6138244c36e1 02dc51f5000000009b7cbc7f126d515ac02b 50e89d5c8cf3	38.1 %
9	12d94d1700000000c271f1fb198cd33c3072 36b0bff41336 2f57ee3e000000009491a1d737d790af3fd2d 547239b2c95	37.7 %
10	1d166daa00000000f7809c7930edb09cf171 1ef5a299e6aa 2344d58f00000000a61385aa9e28bf4626b3 d8fe91fb9a3c	39.1 %
Key1	block cipher algorithm	
Key2	alock cipher algorithm	

Average avalanche effect=386.3/10=38.63%
Table 2 shows the avalanche effect of the proposed algorithm (Genetic algorithm with Blowfish algorithm) uses the same plaintext, when only one bit is changed for the same key by using a block size of 192 bits.

Table 2: Avalanche Effect of the proposed algorithm of 192-bits block size on the key

Block no.	***Cipher text 192 _bit in binary***	***avalanche***
1	234be14300000000a02f25ed5f91e3b40af61 c6cdbe83797 3e72ce3b0000000093a61b4bc238df8cd5b0 6543676009a8	45.3 %
2	540c1f3000000000c00443be7ecd347c8452 e772dca8b964 244d63be00000000a44120a2eaa31f4b574d 1b5473e7b6ee	46.2%
3	3667b9c4000000002c04d34fdd76671f14b4 52a585642d51 1829241400000000e7525be39437f92ac72b f898f696f6f9	43.8 %
4	51a0d01700000000a6b7bbe6e8b5e355d826 6265e7d6a34f 36d9551d0000000042f0a8eae074ae1b8b48 a8d93cd03dc0	43.9 %
5	54b58bbb0000000007b3d4aeee93af05e52b 4c4908bcfd57 35aadeba000000008d47867066023897dbb 98bf40e7c4cec	47.6 %
6	0a93396e0000000042bfb0842c3197a9d1b6 8a39ad7e3093 2ff95274000000004ea1a38c46ebb2ced9a25 d3087a01138	44.9%
7	5d92248900000000be930bf646b97b186ba2 994ae55b1c9b 3f86ae7b000000003f3be4b8f6e40ddd21974 fad9ae76dfe	43.8 %
8	068f6a03000000008bef0484f2839ce4bf9a4 06ab3d12414 79361696000000000e4fd5023940f11416028 ce36f58fa953	45.3 %
9	744cff78000000003647d3bcbb25543442b1 84bdfed169b7 511ed25a00000000bb2d8791e52c5b32c32e 37e5d71ef488	46.6%
10	689091b700000000fc7d1db91ace6d7a8638 562849bbdc3b 7ef6252b00000000a1c6aa61038ff6e29cd9c 83fe10aebf0	44.7%
Key1	block cipher algorithm	
Key2	alock cipher algorithm	

Average avalanche effect=452.1/10 = 45.21

Table 3 shows the avalanche effect of original Blowfish algorithm on the same key when only one bit is changed in the plaintext by using a block size of 384 bits.

Table 3 Avalanche Effect of original Blowfish of 384-bitsblock size: Change one bit in plaintext.

Block no.	***Cipher text 384 _bite in binary***	***avalanche***
1	1433dd0100000000174dca71c5c885094058d2e098e6bc9d2e863fc42db8403665da177015a6ae13d10462498723e470	36.4 %
	2ef6fe2100000000c242f359b8d89ce9addcc2da6bdfd6d99a2828439de59a23a996168eed36ea0efa61ed2f57ad5ddc	
2	54f094de00000000910a65539b78f0d5bcfe69e946dfc95db6f35e8085100081497e0fd9aa67889c25f5a30a4b058aa2	34.9 %
	34219cf600000000be67df1bc000668b0a5d1f5cbe6230948df9b1c6257a52b4f2a77ea64951d3470293738af94be51f	
3	5f85b1bd0000000015ae6d73f34fa0cb16efa456fe5385fc94ee43a194d698dcd9e3a68aacdbc6af8e01c7723d4b2915	39.1 %
	099674190000000003abbbb8915ecc2de39a3638c8e7ffa5e2a7e166fe27464c045b4adbb9a7d0b82e64f09dbc9c21f5a	
4	1fd6420c000000008d1810ac721944b28dcefb1f2c490136e723365e818988d3ec7dbe62d3b3811016fd98f2983c0e78	40.6 %
	344e7e980000000006c8da0f9ddf19800c9bdceeed10f0a41441132882b2083003d9df498a72cdaa53ebf6283c7d76204	
5	638080990000000004b8eacb9c69b63f07a56b51cde26cb5f4cf7544dd35d903c34f664e37764c4f3227b047bc03e016f	36.4 %
	13fef4f7000000005d6c1a5e916902182191692ad48d61ad6b9d5fc4ab7228c8eaea44498e9aba6031cc3419460275c9	
6	00b02ffb00000000dba2ebc33a60d9cb6899c061ddfc9fe58298b58e4a9d5776c14edda05d15b6caa6b1d7a13b995b81	38.6%
	1cd6803500000000050061df96ecdab0419a297f05b6355cd1661095ed111a4e0017502227b0c26c24482986b952bf124	
7	3297978000000000 7488e55df838b203afe0f622fd10255d8e40aeba34e579e9f572e5ea6e247af641795992f246f374	34.1 %
	703dc3b300000000c5d2df9f63ef5276b412bbcfc167edc6c82e6ef7070066248ed2ae83d015670a4dc8971f685ef5aa	
8	528a8b700000000073fcafc4d1d3e79168cf23b112b47ee51fbfd6250e2cadad1cffa8ea4a0982c8772d5cc1d730c773	37.3 %
	5b8d824e00000000ea1b3cacdde8786bff6a17d53e4f445291a3f6f63d0dff7d3a318a04e1a4ca395509cbc4819c5caa	
Key	300000000000000000000000000000000	

Average avalanche effect=297.4/8=37.17 %

Table 4 shows the avalanche effect of the proposed algorithm (Genetic algorithm with Blowfish algorithm) on the same key when only one bit is changed in the plaintext by using a block size of 384 bits.

Table 4 Avalanche Effect of Proposed Algorithm of 384-bitsblock size: Change one bit in plaintext.

Block no.	***Cipher text 384 _bite in binary***	***avalanche***
1	1433dd0100000000174dca71c5c885094058d2e098e6bc9d2e863fc42db8403665da177015a6ae13d10462498723e470	43.4 %
	2ef6fe2100000000c242f359b8d89ce9addcc2da6bdfd6d99a2828439de59a23a996168eed36ea0efa61ed2f57ad5ddc	
2	54f094de00000000910a65539b78f0d5bcfe69e946dfc95db6f35e8085100081497e0fd9aa67889c25f5a30a4b058aa2	45.9%
	34219cf600000000be67df1bc000668b0a5d1f5cbe6230948df9b1c6257a52b4f2a77ea64951d3470293738af94be51f	
3	5f85b1bd0000000015ae6d73f34fa0cb16efa456fe5385fc94ee43a194d698dcd9e3a68aacdbc6af8e01c7723d4b2915	47.1 %
	099674190000000003abbbb8915ecc2de39a3638c8e7ffa5e2a7e166fe27464c045b4adbb9a7d0b82e64f09dbc9c21f5a	
4	1fd6420c000000008d1810ac721944b28dcefb1f2c490136e723365e818988d3ec7dbe62d3b3811016fd98f2983c0e78	47.7 %
	344e7e980000000006c8da0f9ddf19800c9bdceeed10f0a41441132882b2083003d9df498a72cdaa53ebf6283c7d76204	
5	638080990000000004b8eacb9c69b63f07a56b51cde26cb5f4cf7544dd35d903c34f664e37764c4f3227b047bc03e016f	48.4 %
	13fef4f7000000005d6c1a5e916902182191692ad48d61ad6b9d5fc4ab7228c8eaea44498e9aba6031cc3419460275c9	
6	00b02ffb00000000dba2ebc33a60d9cb6899c061ddfc9fe58298b58e4a9d5776c14edda05d15b6caa6b1d7a13b995b81	45.6%
	1cd6803500000000050061df96ecdab0419a297f05b6355cd1661095ed111a4e0017502227b0c26c24482986b952bf124	
7	3297978000000000 7488e55df838b203afe0f622fd10255d8e40aeba34e579e9f572e5ea6e247af641795992f246f374	47.1 %
	703dc3b300000000c5d2df9f63ef5276b412bbcfc167edc6c82e6ef7070066248ed2ae83d015670a4dc8971f685ef5aa	
8	528a8b700000000073fcafc4d1d3e79168cf23b112b47ee51fbfd6250e2cadad1cffa8ea4a0982c8772d5cc1d730c773	46.3%
	5b8d824e00000000ea1b3cacdde8786bff6a17d53e4f445291a3f6f63d0dff7d3a318a04e1a4ca395509cbc4819c5caa	
Key	300000000000000000000000000000000	

Average avalanche effect=371.5/8=46.44

Table 5 Execution Time (Milliseconds) of Encryption and Decryption of Different data packet size

Plaintext Size (Bytes)	Key Size (Bytes)	Original Blowfish		Blowfish with geenetic		Original Blowfish	Blowfish With genetic
		Encryption Time	Decryption Time	Encryption Time	Decryption Time		
41	8	3.0	2.7	24.9	25.5	5.7	50.4
82	12	5.7	3.2	30.8	26.8	8.9	57.6
48	16	2.9	3.1	32.0	24.7	6.0	56.7
74	20	2.9	1.5	24.4	21.0	4.4	45.4
45	24	3.3	3.1	28.7	25.2	6.4	53.9
72	28	3.5	3.1	26.6	27.0	6.6	53.6
58	40	5.1	4.5	35.7	27.3	9.6	63.0
52	44	3.2	2.8	28.6	26.5	6.0	55.0
49	44	3.2	3.1	26.3	28.5	6.3	54.8
63	56	3.3	2.8	32.9	27.2	6.1	60.1
Average Execution Time						6.6	55.1

5.2 Execution time

The encryption time is the time which is taken by the algorithms to transform the plaintext to the cipher text. Decryption time can be defined as the time required for converting cipher text into plaintext.
The summation of encryption time and decryption time is considered as the execution time which is measured by milliseconds [13].
Execution time=Encryption time + Decryption Time [13].
Table 5 shows the average execution time for the original blowfish algorithm and the proposed algorithm.

6. Results and Discussion

The results show that the average avalanche effect of the proposed algorithm is 45.21% when one bit in the key has changed, whereas the average avalanche effect of the original Blowfish algorithm is 38.63%. Also the average avalanche effect is stronger in the proposed algorithm when one bit has been changed in the plaintext which is equal to 46.44%, but by using the original blowfish algorithm the average avalanche effect was 37.17 %.
It is demonstrated that change one bit in the plaintext and change one bit in the key produce strong avalanche effect in the proposed algorithm. Hence the security of the proposed algorithm is stronger than the original one.

The results also show that the average execution time for the proposed algorithm is 55.1 milliseconds whereas in the original blowfish algorithm is 6.6 milliseconds, which better than the proposed algorithm. The proposed algorithm needs more time to be executed rather than original blowfish algorithm because it use genetic algorithm in the proposed algorithm which increase security.

7. Conclusion

In this paper the proposed algorithm and the original blowfish algorithm have been developed using C# language. The main comparison factors which were used are avalanche effect and execution time; the results show that the avalanche effect in the proposed algorithm is better than the original blowfish algorithm in both changes when one bit has been changed in the key or in the plaintext. The avalanche effects in the proposed algorithm when one bit has been changed in the key and when one bit has been changed in the plaintext were 45.21 % and 46.44 % respectively. But the avalanche effects in the original blowfish algorithm when one bit has been changed in the key and when one bit has been changed in the plaintext are 38.63 % and 37.17 % respectively. That indicates that the proposed algorithm is stronger than the original one. According to the execution time factor which was used to compare the proposed algorithm with original blowfish algorithm the results show that the average execution time for the proposed algorithm is 55.1 milliseconds whereas in the original blowfish algorithm is 6.6 milliseconds, which better than the proposed algorithm. The proposed algorithm needs more time to be executed rather than original blowfish algorithm since it uses genetic algorithm processes which increases security.

Acknowledgment

The authors are grateful to the Applied Science Private University, Amman, Jordan, for the full financial support granted to this research.

References

[1] Bulgurcu, Burcu, Hasan Cavusoglu, and Izak Benbasat. "Information security policy compliance: an empirical study of rationality-based beliefs and information security awareness." MIS quarterly Vol. 34 No.3, pp. 523-548, 2010.
[2] Johnston, Allen C., and Merrill Warkentin. "Fear appeals and information security behaviors: an empirical study." MIS quarterly pp. 549-566, 2010
[3] Singh, Simar Preet, and Raman Maini. "Comparison of data encryption algorithms." International Journal of Computer Science and Communication Vol. 2, No. 1 pp.125-127, 2011.
[4] M. Tariq Banday, "Easing PAIN with Digital Signatures", International Journal of Computer Applications Vol. 29– No.2, September 2011
[5] Devaraj, D., and B. Yegnanarayana. "Genetic-algorithm-based optimal power flow for security enhancement." IEE Proceedings-Generation, Transmission and Distribution Vol. 152 No.6, pp. 899-905, 2005.
[6] Wael Raef Alkhayri, Suhail Owis, Mohammad Shkoukani, "A New Selection Operator - CSM in Genetic Algorithms for Solving the TSP", International Journal of Advanced Computer Science and Applications, Vol. 7, No. 10, 2016.
[7] Banković, Zorana, et al. "Improving network security using genetic algorithm approach." Computers & Electrical Engineering Vol. 33, No. 5, pp. 438-451, 2007.
[8] Ganesh Patidar, Nitin Agrawal, SitendraTarmakar ,"A block based Encryption Model to improve Avalanche Effect for data Security", International Journal of Scientific and Research Publications,Volume 3, Issue 1, January 2013 1 ISSN 2250-3153.
[9] Mandal, Pratap Chnadra. "Superiority of Blowfish algorithm." International Journal of Advanced Research in Computer Science and Software Engineering Vol. 2, No. 9, pp. 196-201, 2012.
[10] Monika Agrawal, Pradeep Mishra,"A Modified Approach for Symmetric Key Cryptography Based on Blowfish Algorithm",International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958 , Volume-1, Issue-6, August 2012.
[11] Saravana Kumar, A.Shanmugam, "Modified F – Function for Feistel Network in Blowfish Algorithm", International Journal of Engineering and Innovative Technology (IJEIT) Volume 4, Issue 4, October 2014.
[12] Christina L, Joe Irudayaraj V S, "Optimized Blowfish Encryption Technique",International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization)Vol. 2, Issue 7, July 2014.
[13] Aman Kumar, SudeshJakhar ,Sunil Makkar ,"Comparative Analysis between DES and RSA Algorithm's", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue7,pp. 386-390, July 2012.
[14] Thakur, Jawahar, and Nagesh Kumar. "DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis." International journal of emerging technology and advanced engineering Vol. 1, No. 2, pp.6-12, 2011.

Yosef Bane Awwad received her B.Sc. degree from Irbid Private University, Amman, Jordan, in 2008 in computer science and M.Sc. degree from Applied Science Private University in 2016 in computer science. He is a researcher. His research interests include Electronic Commerce, Software Engineering, and Information Security.

Mohammad Shkoukani received his B.Sc. degree from Applied Science Private University, Amman, Jordan in 2002, and M.Sc. degree from Arab Academy for Banking and Financial Sciences in 2004, both in computer His Ph.D. degree in computer information systems from Arab Academy for Banking and Financial Sciences, Amman, Jordan in 2009. He is an associate professor at Applied Science Private University, Amman, Jordan, His research interests include Information Security, Agent Oriented Software Engineering, System Analysis and Design, and Electronic Commerce Applications.