**How to Cite**:
Srivastava, S., Tiwari, A., & Singh, R. (2022). An implementation of modified blowfish using random number generator for quantum computers. *International Journal of Health Sciences*, *6*(S1), 12496–12504. https://doi.org/10.53730/ijhs.v6nS1.8121

# An implementation of modified blowfish using random number generator for quantum computers

**Shipra Srivastava**
Research Scholar (CSE), Dr. K N Modi University, Newai, Rajasthan, India
Corresponding author email: shiprasrivastava2000@gmail.com

**Dr. Anoop Tiwari**
Professor, Department of CSE, Dr. K N Modi University, Newai, Rajasthan, India

**Dr. Ramveer Singh**
Professor, Departmet of IT, Greater Noida Institute of Technology, Greater Noida, UP, India

***Abstract*---**Classical computers are quite safe from attacks by implementing traditional cryptographic techniques and algorithms but in future as quantum computers will come in full fledge, our current cryptographic algorithms will become vulnerable. In this paper authors have presented a modified blowfish scheme named as pseudo random generated blowfish encryption algorithm which can provide security to our data quantum computers also. Analysis of algorithm is done on classical computer as well as on quantum simulators (IBM qx) and result of comparison is also presented. The theoretical security measures are also discussed in this paper.

***Keywords*---**Blowfish Algorithm, Cryptograph, PRNG, Symmetric and Asymmetric algorithms, quantum simulator.

## Introduction

Blowfish is the symmetric key encryption technique which was designed by Bruce Scheneir in 1993 [13,17]. It was an alternative of data encryption standard technique and was very effective against cryptanalysis. The security of currently used cryptographic algorithm is dependent on discrete logarithmic problem, which can be overcome by integer factorization problem. Quantum computers can solve this problem in fraction of seconds. So there is a threat to all currently existing cryptographic algorithms. In practical blowfish encryption standard is almost impossible to break [21]. But due to vulnerability of currently existing algorithms we proposed a modified blowfish algorithm which uses pseudo random number

*Manuscript submitted: 18 March 2022, Manuscript revised: 9 April 2022, Accepted for publication: 27 May 2022*

generator. In our PRNG blowfish approach, key is generated using random number which provides better security in comparison to traditional blowfish [22,23].

The Accenture report claims that by 2028 quantum computers will be able to implement Shor's algorithm. Thus, by 2028 we are able to break many of the classical cryptosystems. Practically there are so many challenges with in availability of quantum computers such as environmental factors associated with existence of qubits and management of quantum systems. As an alternative quantum simulator permits the study of quantum system in programmable fashion[8]. Companies like Atom computing, Google, IBM are providing online simulators to the researchers IBM was the first company which offers universal quantum computing systems via IBM Q network [9,10].The proposed algorithm is designed for quantum computers and result is analyzed in IBM qx simulator.

**Preliminaries**
Data encryption plays an eminent role in cryptographic process

## Cryptography

Cryptography (or cryptology; from Greek κρσπτὸς, kryptos, "hidden, secret"; and γράφω, gráphō, "I write", or -λογία, -logia, respectively) is the practice and study of hiding information. There are two types of cryptosystems: asymmetric key cryptosystem and symmetric key cryptosystem [12].

### Asymmetric key cryptosystem

In asymmetric key cryptosystems two different keys are used for encryption and decryption process. One key is known as public key, which is known to public and other is the private key, which is kept by the owner only. The reason of being called asymmetric is the use of two different keys for encryption and decryption process.RSA, Elgamal etc are the examples of asymmetric key encryption algorithm.

### Symmetric key cryptosystem

In symmetric key cryptosystem same key is used for encryption as well as for decryption process. Some of the most popular examples of modern symmetric key cryptosystems are DES, AES, IDEA, RC5, two fish, Blowfish and many others. All symmetric key cryptosystems rely on sharing of secret keys between the communicating parties.

## Traditional Blowfish Algorithm

Blowfish's Algorithm is symmetric key encryption technique which is based on fiestel network. In fiestel cipher, both encryption and decryption process have very similar operations. Since blowfish was an alternative of data encryption standard hence it provides better security with no effective cryptanalysis [15].Blowfish is a block cipher which uses block size of 64 bits, key size of variable length from 32 bit to 448 bits and iterates 16 times. There are two components in the process of Blowfish Algorithm: data encryption and key expansion [6]. The key expansion process starts with the initialization of four S boxes each having 512 entries of 32 bit each and P-array which includes the use of several sub-keys. There are total eighteen sub-keys of size 32 bit each in the P array: blowfish algorithm encrypts data iterating a 16-rounds of operation and each round is classified into data-dependent substitution and key-dependent permutation [16].

Obtained data of size 32-bit is then transferred to section 2 for the F function which in turn converts it into segment of 32-bit(L) which is XOR'ed with other segment of 32 bit (R) . The L and R segments are then interchanged for subsequent Blowfish Algorithm iterations. Figure 1 shows the structure of traditional blowfish algorithm.

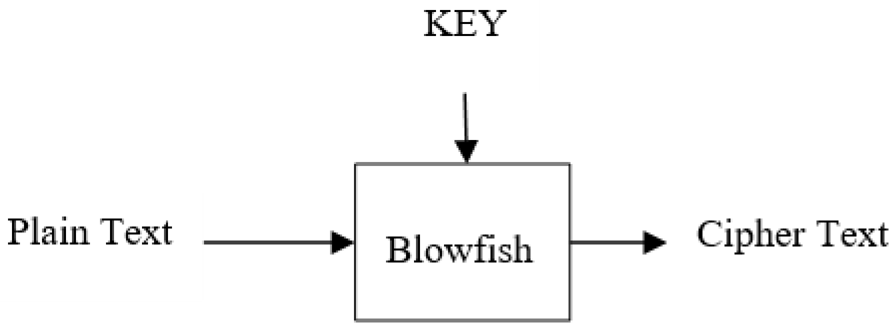

Figure 1: Traditional Blowfish algorithm

**Blowfish Encryption Algorithm**

Figure 2 illustrates the entire process of encryption. The process starts with generation of 18 sub keys {P[0], P[1],......P[17]} The same sub keys are needed for both encryption as well as for decryption process. All eighteen sub keys are stored in the P-array and each array is of size 32 bit. The entries in the P-array are initialized with the value of pi. After that each sub key is changed with respect to input keys [15]. The obtained P-array stores 18 sub keys which is used for entire encryption and decryption process. Four S-boxes are used and initialized with 256 entries per S-boxes. Then Encryption process starts which consists of two parts : rounds and post-processing. Encryption consists of 16 rounds taking plain text from previous round and corresponding sub keys. The output got after 16 rounds is processed and we get our final cipher text as output [16].

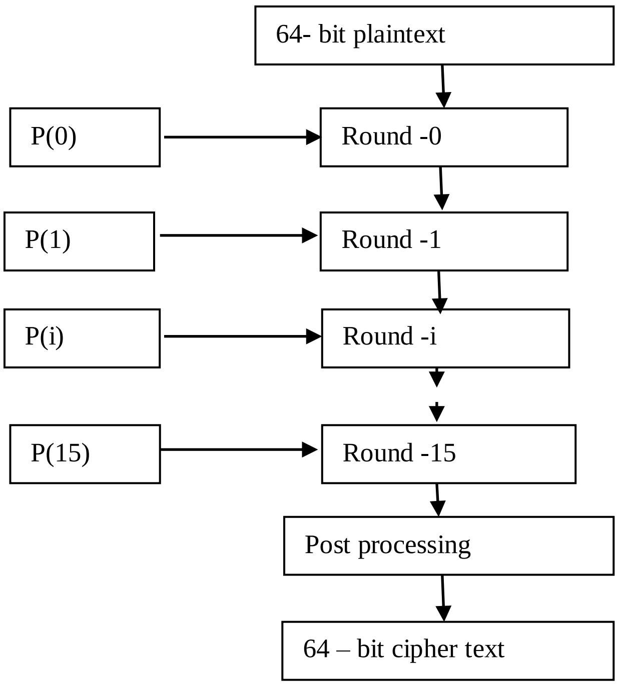

Figure 2: Blowfish encryption process

**Pseudo random number generator**

Applications in cryptography use random numbers for generation of encryption keys, it creates initial parameter values and introduce random numbers into protocol. Generally random numbers are generated using Pseudorandom number generator (PRNG) [1, 2].
PRNG uses an algorithm, which use mathematical formula for producing sequences of random numbers, the numbers are deterministic in nature, that is sequence of given numbers can be reproduce later, if we know the starting point in sequence. PRNG is periodic in nature, it means sequence will at last repeat itself [3]. There are various application where many random numbers are required such as simulation.
We categorized computer generated random number into two types on the basis of how they are generated.

1. True Random Number
2. Pseudo Random Number

A computer system could use a seed value and procedure or algorithm to a generate a number that can be based on fact predictable [3]. In python we can generate random number by using following command:-

1. *Import Random*
2. *randomlist = random.sample(range(0-511),14)*
3. *Print (randomlist)*

```
12500
```

This code generates 14 random numbers in the range from 0 to 511.

## Our approach

In our approach, we will encourage the user to use data blowfish with more efficiency and security. We emphasis on secrecy of key as we know that key always play very important role in cryptograph [7]y. In traditional blowfish, the key size is of 32 bit and data block size is of 64 bit. Blowfish follows the block cipher mode encryption and decryption. In this paper, we will show how secret keys are generated using pseudo random number generator which will work as a secret key in Data Blowfish for each block of message.

**In Traditional Blowfish:**

PT = {m1, m2, ..................., mn} (64- bit block of message)
K = {K} (32- bit Key)

**Key Generation:**

• Eighteen sub keys {P[0], P[1], P[2],....,P[17]} are needed for both encryption as well as decryption process and the same sub keys are used for both the processes.

• These eighteen sub keys are stored in a P-array and each array is having each P-array 32-bit entry.

• The eighteen –array are initialized with the first 18 digits of pi. Now each of sub keys are changed w.r.t the input key as:

P[0] = "243f6a88"
P[1] = "85a308d3"
.
.
.
P[17] = "8979fb1b"

• Now each of the sub key is changed with respect to the input key as:
P[0] = P[0] XOR 1st 32-bits of input key
P[1] = P[1] XOR 2nd 32-bits of input key
P[i] = P[i] XOR (i+1)th 32-bits of input key
(roll over to 1st 32-bits depending on the key length)
.
.
.
P[17] = P[17] XOR 18th 32-bits of input key
(roll over to 1st 32-bits depending on key length)

**For Encryption-Decryption:**

Ci = EK {mi}
Cipher Text C = {C1, C2, ………………, Cn}
And mi = DK {Ci}
Plain Text M = {m1, m2, ………………, mn}

As per traditional Blowfish, Encryption process follows the feistel structure (16-rounds) and make a new key for each round by the permutation on bits of key k . Same key k applies on each block of message m using shifting property for encryption and decryption.

**PRNG Blowfish Approach:**

M = {m1, m2, m3, ……………., mn} (64- bit block of message)
K = {K} (32- bit Key)

**Key Generation:**

Figure 3 shows the process of new generated key by Pseudo random number generator [5,6].

F {K and Rn} = [K_new (X)]_P)
Where Rn is generated by Pseudo Random Number Generator (PRNG) and [1 ≤ Rn ≤ (256 = 4294967295)].
[K_new (X)]_P) = [K_new (X)]_1) to [K_new (X)]_n)

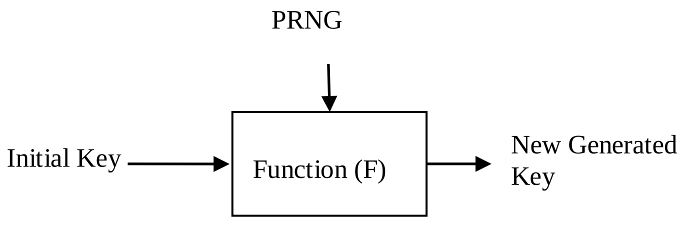

Figure 3: Process of new generated key of PRNG Blowfish

**Function F:**

**Step 1-** Input the value of initial key K of 32-bit.
**Step 2-** Input the generated random number Rn , generated by PRNG*. (*PRNG Property- 256 no., random number generator)
**Step 3-** Convert Rn into 32- bit binary number.
**Step 4-** Now, we have
Key K = {Kb1, Kb2, Kb3, ……………………., Kb32}
And Rn ={Rb1, Rb2, Rb3, ………………………, Rb32}
Where Kbr is the bit of Key and
Rbr is the bit of Random number.

Here r =1, 2, 3..............32.

**Step 5-** Apply above conditions on K and Rn.
IF Rbr and Kbr is the same then new Kbr is invert of Kbr (convert 1 to 0 or 0 to 1) of corresponding Kbr.
ANDIF Rbr and Kbr is not same, then Retain the same (1 to 1 or 0 to 0) of corresponding Kbr.

**Step 6**- [K_new (X)]_P) = Result of step 5.

Using this function F every time we get the result [K_new (X)]_P) for each block of message. For each block of M we generate a new no. nj and implement function F. Finally get a new key for every block of message [4].

**For Encryption/Decryption:**

In encryption phase, PRNG Blowfish takes a message block mn and a new generated key [K_new (X)],P), implement encryption process as per traditional Blowfish [19,20].

Decryption Process is the inverse step of encryption process. In decryption, we also use the same key which is used in encryption.

Ci = E_([K_new (X)]_P)) {mi} an
mi = D_([K_new (X)]_P)) {Ci}, where 1≤ i ≤ n.
Cipher Text C = {C1, C2, ................., Cn} and
Plain Text M = {m1, m2, ................., mn}.

**Comparison of encryption-decryption time in classical computer and quantum computers**

Table 1 shows the comparative analysis of encryption/decryption time and total execution time (in seconds) of PRNG blowfish algorithm on classical computer and on quantum simulators (IBM qx).We can compare that on the same input encryption/ decryption time is less in quantum computers but total execution time is comparatively more in quantum computers in comparison to classical computers [14].

Table 1
Comparison of encryption-decryption time (in Sec.) and total execution time (in Sec.) of PRNG Blowfish

| **Parameters** | **Classical computer** | **Quantum computer** | **Comparison** |
|---|---|---|---|
| Encryption time | 0.00009903206955641508 | 0.00001250810455530882 | Encryption time is less in quantum computers |
| Decryption Time | 0.00006284704431891441e | 0.0001244029845111072 | Decryption time is less in quantum computers |
| Total execution time | 0.00023013807367533445 | 0.0006635080208070576 | Total execution time is less in classical computers |

## Conclusion

We have examined PRNG Blowfish symmetric key algorithm, It is successful in secure key generation for quantum computers. There are two cases:
**Case I:** If we take single key in place of n keys then n Knew(X) is K. It is successfully implemented in quantum simulators and is need of the future.
**Case II:** If Knew(X1) = Knew (X2) = Knew (Xn) then proposed approach will work like blowfish.
Our section 4 completely advocates the plausibility of ORDES. It is successful and needful for current communication scenario.

## References

1. A.M. Eskicioglu, “Protecting Intellectual Property in Digital Multimedia Networks,” IEEE Computer, July 2003, pp. 39-45.
2. B. Jun and P. Kocher. The Intel Random Number Generator. Cryptography Research Inc. white paper, Apr. 1999.
3. C. Petrie and J. Connelly. A Noise-based IC Random Number Generator for Applications in Cryptography. IEEE TCAS II, 46(1):56– 62, Jan. 2000
4. CNET News.com, Users take crack at 56-bit crypto. Available on-line at http://news.com.com/2100-1023-278658.html?legacy=cnet, 1997.
5. D.B. Ojha, Ramveer Singh, Ajay Sharma, Awakash Mishra and Swati garg “An Innovative Approach to Enhance the Security of Data Encryption Scheme” International Journal of Computer Theory and Engineering, Vol. 2,No. 3, June, 2010,1793-8201
6. Elminaam, Diaa Salama Abd, Hatem Mohamed Abdual-Kader, and Mohiy Mohamed Hadhoud. "Evaluating the performance of symmetric encryption algorithms." IJ Network Security 10.3 (2010): 216-222.
7. Gallagher, Patrick. "Digital signature standard (DSS)." Federal Information Processing Standards Publications, volume FIPS (2013): 186-3.
8. Irfan Ahmad, Karunakar Pothuganti, Analysis of different convolution neural network models to diagnose Alzheimer’s disease, Materials Today: Proceedings, 2020, ISSN 2214-7853,https://doi.org/10.1016/j.matpr.2020.09.625.
9. Kaur, Randeep, and Supriya Kinger. "Analysis of security algorithms in cloud computing." International Journal of Application or Innovation in Engineering and Management 3.3 (2014): 171-6.
10. Kunz-Jacques, S., Muller, F.: New Improvements of Davies-Murphy Cryptanalysis.In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 425–442. Springer, Heidelberg (2005)
11. Mahajan, Prerna, and Abhishek Sachdeva. "A Study of Encryption Algorithms AES, DES and RSA for security." Global Journal of Computer Science and Technology (2013).
12. Mandal, Akash Kumar, Chandra Parakash, and Archana Tiwari. "Performance evaluation of cryptographic algorithms: DES and AES."

Electrical, Electronics and Computer Science (SCEECS), 2012 IEEE Students' Conference on. IEEE, 2012.

13. N. V. Ganapathi Raju, A. Radhanand, K. N. Balaji Kumar, G. Pradeep Reddy, and P. Sampath Krishna Reddy, “Machine learning based power saving mechanism for fridge: An experimental study using GISMO III board,” Mater. Today Proc., vol. 33, pp. 4819– 4822, 2020, doi: 10.1016/j.matpr.2020.08.387.
14. Nadeem, Aamer, and M. Younus Javed. "A performance comparison of data encryption algorithms." Information and communication
15. Nie, Tingyuan, and Teng Zhang. "A study of DES and Blowfish encryption algorithm." Tencon 2009-2009 IEEE Region 10 Conference. IEEE, 2009.
16. P. Kohlbrenner and K. Gaj. An embedded true random number generator for fpgas. In FPGA ’04: Proceeding of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays, pages 71–78. ACM Press, 2004.
17. P.C. van Oorschot, A.J. Menezes, and S.A. Vanstone, “Handbook of Applied Cryptography,” CRC Press, Inc., 1997.
18. R. B. P. Dept. The Evaluation of Randomness of RPG100 by Using NIST and DIEHARD Tests. Technical report, FDK Corporation, 2003.
19. Ramveer Singh et. al. /, An Ordeal Random Data Encryption Scheme (Ordes), International Journal of Engineering Science and Technology Vol. 2(11), 2010, 6349-6360.
20. Sebastien Kunz-Jacques, Frederic Muller, New Improvements of Davies-Murphy Cryptanalysis, Advances in Cryptology, proceedings of ASIACRYPT 2005, Lecture Notes in Computer Science 3788, pp. 425–442, Springer, 2005.
21. Stallings, William, and Mohit P. Tahiliani. Cryptography and network security: principles and practice. Vol. 6. London: Pearson, 2014.
22. Tirthani, Neha, and R. Ganesan. "Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography."
23. Whitfield Diffie and Martin E. Hellman, “New Directions in Cryptography” IEEE transactions on Information Theoty, 22, 644-654.