Time Complexity Reduction for the Application of Stream Cipher System Based XOR Free Operation

Radhakrishna Dodmane, Ganesh Aithal, Surendra Shetty

Abstract*: *Providing security to user information are the most important aspects of today's internet applications. Due to the very fast increase in the internet applications, the value of data exchange also became important. Hence, the better security and fast processing of the system is at high demand in the communication-related communities. Cryptography is one such area of providing data security. The two major criteria that specify the significance of a cryptographic algorithm are the security provided against the various attacks and the time to perform the operations.

The proposed system is a solution to the cryptographic system to reduce the time complexity by retaining the security of the algorithm as it is. The solution proposed is to reduce the processing time required for the XOR function. This solution is by generating; XOR values for the vector space mentioned by the tuple, later replace the same wherever required. These values of XOR depend on the tuple, which represents the vector. Algorithm and time complexity for the generation of the vector space of XOR values for the proposed space evaluated. The result of the work compared based on two Standard symmetric-key cipher systems, Advanced Encryption Standard (AES) and Blowfish. The results are also tested and analyzed for different data size for the time complexity.

Keywords: Advanced Encryption Standard (AES), Blowfish, Vector Space and XOR free.

I. INTRODUCTION

In modern era, the users of the internet are greatly increasing in an exponential manner. Almost all the users are using the public network for their data transmissions, which made even the confidential information is openly accessible by everyone. Thus increases the vulnerability of the message transmission [1]. So it's a great demand on the techniques that could provide the desirable protection to the confidential data which are transmitted around the public network. Various techniques are proposed to achieve the information security under two major areas such as; the symmetric-key cryptography and asymmetric-key cryptography. . This paper considers standard symmetric key cipher system for the test process.

In binary oriented symmetric key cipher system, almost all standard Stream ciphers use the XOR as default repeated operations. Due to the repeated XOR operations, the time spent on loading and storing of the data of these operations to the cache is larger [2], [3]. Therefore, the increase in loading and storing of the data to the cache from the other cache levels or to the main memory again causes the increase of delay in storing and loading [2]. In case of, the requested data by the CPU, for the operation, if not found in the cache leads to a cache miss. This in turn increases the executions delay. Therefore, there is a need of addressing these issues, which helps in reducing the time complexity. This achieved by reducing the time needed to execute the XOR operation. Hence, the work proposing a new solution of optimization based on time complexity. Even though the hardware XOR operations are very fast [4], but due to its above said memory bound reasons; desired to have a system, which will address these issues through software. Therefore, the work focused on the solutions to address the time complexity issue of the operation of XOR function. As a result, the work proposes a new software solution for XOR operations with the intention towards the time complexity reduction. The study centred especially, towards the security solutions of standard symmetric-key cryptographic area.

In most of the standard cipher system, XOR operations usually performed between the plaintext / cipher text and the key. Therefore, the total cost in terms of time complexity of the XOR operation depends on the number of XOR operations, which depends directly on size of the input and the key length [4]. As input size of the plaintext / cipher text increases the XOR process of the cryptosystem also increases proportionally. This increased number of XOR computations leads to the increased computation complexity of XOR [3], [4] and [5].

Thus, the proposed system is to provide software solutions for generating an XOR for the corresponding value of plain text / cipher text and key. This constructed based on an algorithm, which constructs the vector space of the values for the input tuple. The system uses this vector space for XOR free operations by replacing the appropriate value for the same.

Manuscript published on 30 September 2019
* Correspondence Author

Radhakrishna Dodmane*, CSE department, NMAM Institute of Technology Nitte, Karkala, India. Email: rkdodmane@gmail.com

Ganesh Aithal, Vice-principal, SVMIT Bantakal, Udupi, India. Email: ganeshaithal@gmail.com

Surendra Shetty, MCA department, NMAM Institute of Technology Nitte, Karkala, India. Email: hsshetty4u@nitte.edu.in

This method is tested and compared with two standard symmetric-key cryptographic techniques; 1. Advanced Encryption Standard (AES) and 2. Blowfish.

The next section focuses on a survey of few symmetric-key cryptographic techniques with respect to the time complexity of the encryption/decryption operations. These surveys carried out as a preamble to propose a new method. Section III describes the proposed methodology and the details of the time complexity of the new approach. Section IV presents the comparative analysis of the various standard cipher systems, which extensively uses XOR operations. At the end, the last section concludes.

II. Literature survey

The studies carried out on two symmetric key cipher systems such as Advanced Encryption Standard (AES) and Blowfish. Advanced Encryption Standard (AES), which is a symmetric key block cipher technique. Advanced Encryption Standard proposed in 2001 by National Institute of Standards and Technology [6], [7], [8]. Now, it is being used widely in almost all the areas where information security is the utmost importance. The Advanced Encryption Standard (AES) uses a combination of XOR operations, byte substitution using an S-box, rotations of rows and columns (ShiftRows) and mixing of the columns (MixColumn). Based on the varied key length; 128, 192, and 256 bits the AES processes the data in a bit of various sized blocks such as 128, 192, 512 bits. In case if the platform used is of 64 bit processor, the XOR operation in hardware takes 2, 3 and 8 clocks to process it, respectively for the above bits of plaintext / ciphertext and key combinations. Depending on the varied key length, AES processes in 10, 12 and 14 rounds. Every round of AES includes the combinations of above-said functions. The XOR operations are the major operations where the plain text/cipher, text are processed with key [9]. Therefore, there is a need of the system, which handles this operation efficiently. The hardware implementations are comparatively more secure and efficient to software implementations [9], [10] and [11]. However, as said earlier due to the repeated XOR operations, there is more load and store on the cache, hence cache miss [3], [12] that leads to delay in operation. So the time complexity is purely driven by the system cache as well as total data size of encryption / decryption operations. Further, the time complexity affected by the increase in the number of rounds in the overall process. Thus, a repeated XOR operation increases the time of execution of encryption and decryption process in the overall procedure. Therefore, the total time for the encryption and decryption operations also increases. It in turn affects the performance of the cryptographic procedures.

Blowfish is another symmetric-key block cipher system, designed by Bruce Schneider in 1993 [13]. Blowfish processes the data as a 64-bit block size and a variable key length ranges from 32 bits to 448 bits. It is a Feistel cipher [13] processes the data in 16 round and uses large key-dependent S-boxes and XOR operations. In each round, the messages are divided into two halves such as *Li* and *Ri*. Where both the halves (*Li* and *Ri*) includes XOR operations. Hence, the increase in XOR operation increases the execution delay due to the cache miss as well as number of bits operations as in the earlier case [3]. Hence, XOR operation plays a major / critical role in the performance of symmetric-key cipher systems process [3]. The major and repeated operations in the cipher systems are the XOR operations. Jianqiang Luo et al. [5] addressed the issues of XOR-based performance optimization for AES. The bitwise XOR operations are very fast but its memory bound [11]. XOR-operations are flexible [5] [11] but have a large impact on the CPU behaviour due to the repeated load and store on the cash. Based on the analysis of XOR-operation, Jianqiang Luo et al. [5] proposed new schemes to achieve different performance. This above analysis of XOR operations based on the hardware of deferent schemes proposed [11] leads to high time complexity. This gives the path of the orientation of software for XOR operations. It is also seen that time taken for the repeated load and store on cache has an impact on the system performance [3]. Therefore, the improvement of the time complexity of the XOR operation will be more advantageous for the systems involving large number of XOR operations. It will also beneficial for the end users who use the cryptographic process. From the above inference it has been noted that hardware implementation of XOR will leads an increase in complexity of time also cache miss. The orientation to reduce this complexity in this paper has given towards software. This approach will construct a XOR vector space; the operations of XOR are replaced by XOR free operation based on the constructed vector space. Here it reads appropriate XOR inputs as tuple or vector and replace it by using the values available in this vector space. In case of hardware operations, the XOR operation depends on the number of bit the processor operates. If the numbers of bits of operations are less than the XOR operations of the cipher system, then the time complexity increases. To reduce this, better approach is software modelling of XOR. However, the approach of software for each bit XOR operation will lead to complexity. The approach for the code for the multiple bit XOR operation at once is the second option. In the second case the numbers of bits are operated in parallel, this again depends on the data size of the processor. Further the third method of a XOR free operation or standard method in which, the values of XOR is placed from the vector space constructed from the input vectors. The resultant value is to be substituted for the XOR of plain text / cipher text and key of the cryptographic process from the evaluated cryptographic space. The evaluation result of XOR operation of the plain text / cipher text and key should take minimum time to reduce the time complexity.

This construction of vector space can be done recursively, which would have finite numbers '2^r' where '*r*' is a positive integer. This method is discussed in two separate subsections under the methodology. The former one in methodology, describes how to construct the vector space. This construction is based on recursive method. This again reduces the time complexity of the XOR process. The latter focuses on the Time Complexity of the said algorithm.

III. METHODOLOGY

In case of a hardware driven XOR operation of cryptography, it is bit by bit operation. The bit by bit hardware operation leads to the requirement of a large number of XOR gates [14]. It has been decided to use software orientation for the construction of the vector space of XOR based on the two dimensional vector of plaintext / ciphertext and key. The same is utilized for the cryptographic process. There are two questions in this, they are: 1. from which source the data of XOR is to be taken for the process of cryptography? And 2. How to construct the vector space?

A. Vector Space Construction:

It has been decided to replace the XOR operation of cipher systems. For this a vector space of the input vector are to be constructed well in advance. This vector space is constructed based on the ranges of the plain text and key or cipher text and key whichever is maximum. It ranges from '*1*' to '*2^r*', where '*r*' is an integer, which represents the number of bits of the plain text / cipher text / key. The substitution process is performed from the row (plain text / cipher text) and column (Key) of the vector space constructed. The row and column vector is selected as corresponding to the integers of plaintext or cipher text and key, depending on whether it is encryption or decryption operations respectively. The second question is how to construct this vector space? The construction of the vector space is based on the recursive method [15] as explained below.

Algorithm for the construction of XOR Vector Space of the order '2^r':

Let $r \geq 0$ be an integer to represent the number of digits of the vector, such that the size of the vector space be $2^r \times 2^r$ and '*i*' and '*j*' indicates the row and column values of the vector.

Step 1: Read input $r \geq 0$. //a value to decide the vector space

Step 2: Set the XOR vector space '*R*', of order $2^r \times 2^r$. This value of vector '*R*' has two components, '*i*' and '*j*', in this case it is considered as row and column of the vectors. //define the size of the vector with null values.

Step 3: Initialize the elements of the one part of the vector as row and column value as '1'. //initialize the vector.

Such that,

- The value of $R[i][j]=k$, with $i=1$ are $0 \leq k \leq 2^r - 1$, for respective values of '*j*'.

Step 4: Repeat the following for every '*i*' from 2 to 2^r (where '*i*' is incremented in steps of 2^r, such that $r \geq 1$) and $0 \leq k \leq 2^r - 1$. // construct the full vector space recursively.

- Copy $i \times i$ elements diagonally and cross diagonally (shown in the Fig. 1. as cross arrows) for all integer values of row $log_2 i$ for all $1 \leq i \leq 2^r$.

Step 5: END // close after constructing the required vector space.

Example based on the XOR Vector Space algorithm specified above:

Step 1: Let $r=3$ be the input to construct the vector space. That is the vector space of the order $2^3 \times 2^3$.

Step 2: Set the XOR vector space of the order $2^3 \times 2^3$. Such that,

$$R_{2^3 \times 2^3} = R_{8\times 8} = R[8][8]$$

Step 3: Initialize the elements of the one part of the vector as row, as follows:

- $R[i][j]=k$, where $i=1$ and ('*j*' is from *1* to 2^3) and $0 \leq k \leq 2^r - 1$. That is:

$$R[1][1to8] = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

Step 4: Repeat the following for every '*i*' from 2 to 2^r (where '*i*' is incremented in steps of 2^r, such that $r \geq 1$) ands.

- Copy $i \times i$ elements diagonally and cross diagonally (shown in the Fig. 1. as cross arrows) for all integer values of row $log_2 i$ for all $1 \leq i \leq 2^r$.

Such that;

- At first the next value of i that is $\mathrm{i}=2$ and $1 \leq j \leq 2^3$, find $log_2 i = log_2 2$. That is when $\mathrm{i}=2$,

$$R[2][1to8] = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 0 & 3 & 2 & 5 & 4 & 7 & 6 \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

- Next value of i that is $\mathrm{i}=4$ and $1 \leq j \leq 2^3$, find $log_2 i = log_2 4$. That is when $\mathrm{i}=4$,

$$R[3to4][1to8]=\begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 0 & 3 & 2 & 5 & 4 & 7 & 6 \\ 2 & 3 & 0 & 1 & 6 & 7 & 4 & 5 \\ 3 & 2 & 1 & 0 & 7 & 6 & 5 & 4 \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

- ○ Next value of i that is $\mathrm{i}=8$ and $1\le j\le 2^3$, find $log_2 i = log_2 8$. That is when $\mathrm{i}=8$,

$$R[5to8][1to8]=\begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 0 & 3 & 2 & 5 & 4 & 7 & 6 \\ 2 & 3 & 0 & 1 & 6 & 7 & 4 & 5 \\ 3 & 2 & 1 & 0 & 7 & 6 & 5 & 4 \\ 4 & 5 & 6 & 7 & 0 & 1 & 2 & 3 \\ 5 & 4 & 7 & 6 & 1 & 0 & 3 & 2 \\ 6 & 7 & 4 & 5 & 2 & 3 & 0 & 1 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix}$$

Step 5: END

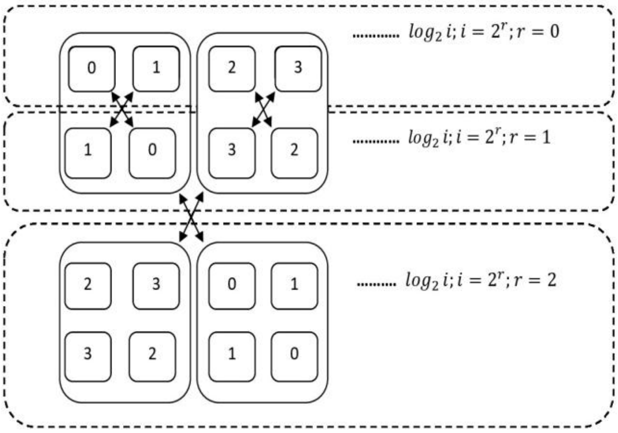

0
1
2
3
1
0
3
2
2
3
0
1
3
2
1
0
.... $log_2 i; i = 2^r; r = 0$
.... $log_2 i; i = 2^r; r = 1$
.... $log_2 i; i = 2^r; r = 2$

Fig. 1 .Pictorial representation of the generation of vector space of XOR.

B. Time Complexity Analysis of the Algorithm of construction of XOR Vector Space:

The time complexity of generating the XOR vector space is presented based on masters theorem [16] as follows:

Let $T(m)$ be the time complexity of the algorithm to construct the XOR vector space where '*m*' is integer decides the size of the vector space:

- Input '*r*' which determines the value of vector space $m=2^r\times 2^r$, this is processed only once during computation and hence takes time of one unit time. Therefore the time complexity of reading the input is $O(1)$.
- The first part of the vector space is to initialize the row as the column values that are 0, 1, 2, 3....k, where $0\le k\le 2^r$. Therefore the time complexity is constant. This is denoted as $O(C)$, where C is constant.
- The remaining elements of the vector space are constructed as follows:
  - ○ The part of the vector row $\left(i=2^r\right)$ value is incremented with $log_2 i$ as integer that is the value equals to $2^1,\ 2^2,\ 2^3,\ 2^4,...,\ 2^r$. That is incremented in 2^r where $r>0$. Therefore, according to the master theorem [16] if the increment of the for-loop is 2^r, than the time complexity is $log_2 r$.

Therefore the time complexity of the algorithm denoted as

$$T(m)=(time\_for\_reading\_input$$
$$+time\_for\_initializing\_the\_first\_row$$
$$+time\_for\_copying\_the\_remaing\_elements)$$

That is, $T(m)=O(1)+O(C)+O(log_2 r)$

Therefore, the time complexity is $O(log_2 r)$.

The XOR vector space is constructed well in advance, that is, the vector space could be created while reading any input file for the cipher system. Hence, all the XOR operations encounters in the cipher system processing can be replaced as a value from the vector space could be termed as XOR free operation. Since it's just a value borrowing from the stored space, leads to less time consumption as well as less power consumption [3]. Thus provides a space to replace XOR operations using XOR free method, which is shown in Fig. 2 using general cipher system.

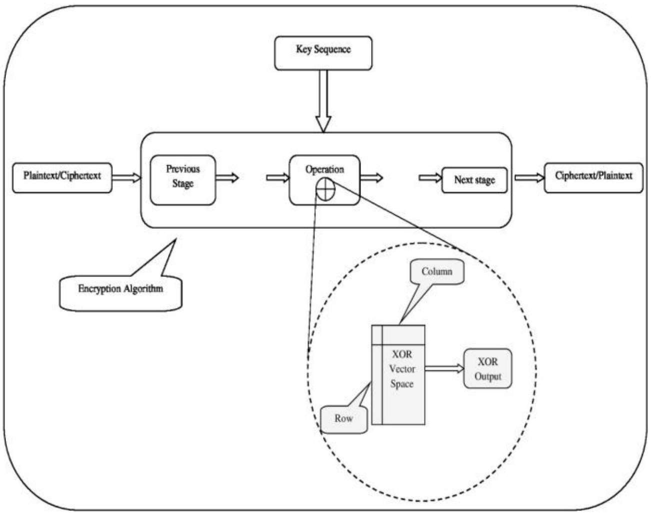


Fig. 2 .General cipher system with XOR free operation.

Retrieval Number: C6176098319/2019©BEIESP
DOI:10.35940/ijrte.C6176.098319
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

IV. RESULTS AND DISCUSSIONS

Even though it has been discussed earlier that the time complexity of evaluating the XOR operation will be reduced by theoretical means, the following section shows how practically it will be evaluated?

To evaluate the time complexity, an experimental setup is formulated. This set up contains an Intel(R) CORE(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz with 4 GB installed RAM. The results of the experiments carried out on the various stream cipher systems are analyzed and discussed with respect to time complexity. The time complexity comparison and analysis are made by conducting the experiments on the standard stream cipher systems v/s proposed method. Due to the overhead of the system and other parameters, consistency of the result may vary. Hence, we have repeated the experiments for fifty times for every varied input size. Then by taking the average of the results we assumed the result as stable.

The size of the vector space of XOR is $2^r \times 2^r$, that is constructed recursively. For every value from (*0 to r*), the vector space is enhanced as per the explanation given in the construction of the XOR vector space algorithm mentioned in section III.

After generation of vector space, the performance of this has to be tested based on implementing this in a standard cryptographic algorithm. This test is done for two such standard algorithms: 1. Advanced Encryption Standard (AES) and 2. Blowfish. Testing of these two standard algorithms is completed based on the standard code taken from www.cis.syr.edu and Bruce Schneider [17], [18].

The results are recorded and analyzed separately for both encryptions and decryptions processes of the algorithm.

To begin with, the test is conducted for the Advanced Encryption Standard (AES) [17]. At first the Advanced Encryption Standard is processed using existing XOR operations. Later all the XOR operations of the Advanced Encryption Standard are carried out as per the Fig. 2 using XOR vector space.

As a part of the encryption time analysis, various input of different file sizes are tested. These tests are repeated for fifty times on each file for the consistent results and finally the average encryption times of these repeated results are presented. The same test is repeated using the XOR free operation of the proposed system. This is repeated fifty times and the average is recorded as the reading which is shown in table I.

The results shows, as the data size increases, the proposed XOR free approach would consume moderately less time. Hence could say proposed approach for encryption works comparatively better for Advanced Encryption Standards. The graphical representations of the difference in the time consumption of the Advanced Encryption Standards encryption are shown in the Fig. 3. The dark grey line in Fig. 4, represents the difference in the time consumed for encryption with respect to proposed approach v/s standard XOR operations of Advanced Encryption Standards.

Table- I: Average difference in time between encryption/decryption of Standard approach v/s proposed (XOR free) method for Advanced Encryption Standards

File Size	Encryption Time analysis		Decryption Time analysis	
	Average Time in seconds elapsed using the proposed method	Average Time in seconds elapsed using the existing method	Average time in seconds elapsed using the proposed method	Average Time in seconds elapsed using the existing method
10MB	287.4682	295.5780	316.2375	332.4416
20 MB	305.3946	319.0005	441.8143	459.8744
30 KB	449.0715	468.9474	642.6782	668.7801
40 MB	633.7815	659.6088	699.5568	733.8118
50 MB	756.0279	791.8737	878.1025	921.1603
60 MB	879.071	923.651	1027.899	1082.471
70 MB	1038.219	1096.578	1123.453	1195.829
80 MB	1123.453	1195.829	1247.058	1333.45
90 MB	1254.437	1344.579	1432.172	1532.4
100 MB	1383.172	1494.579	1612.09	1724.919

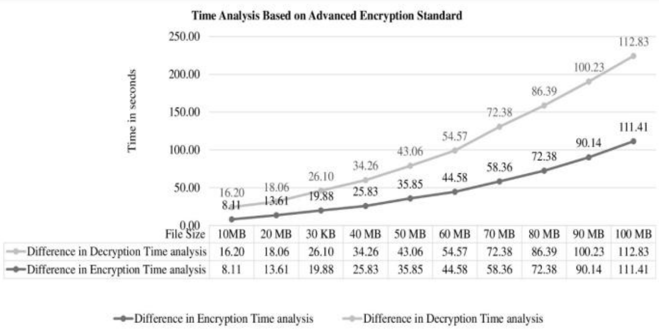


Fig. 3 .Average difference in the time between encryption/decryption of standard approach v/s proposed (XOR free) method for Advanced Encryption Standards with reference to the table I.

The experiments also carried out on decryption process using Advanced Encryption Standards. The results are repeated as like in encryption and the average of fifty test results considered for each file and is repeated for different input size. Table I also contains the difference in the decryption time between both the proposed XOR free operation v/s standard XOR approach using Advanced Encryption Standards. The graphical representations of these differences are plotted using light grey line on the charts shown in Fig. 3.

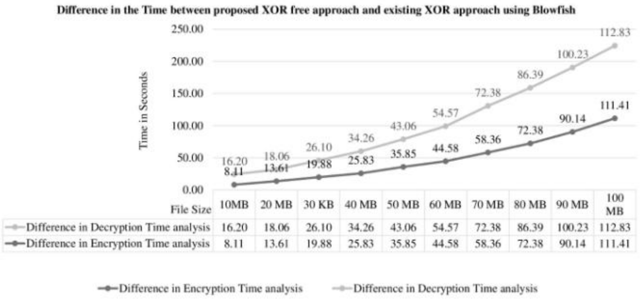
Secondly, the proposed XOR free system is also tested against the Blowfish algorithm [18] for the same input file size as like Advanced Encryption Standards. The test was conducted on the same computer system used for the Advanced Encryption Standards analysis.

Table II shows the time complexity analysis of the encryption and decryption using the proposed XOR free approach v/s existing method on Blowfish algorithm. The test is repeated and recorded fifty times for every different input file size. The computed average times from the fifty repeated observations for each input file size are recorded and are used for the analysis. From the study it's obvious that as the size of the input file increases, the proposed XOR free approach reduce the total time exponentially compared to the existing XOR approach.

Table- II: Average difference in time between the proposed XOR free approach v/s standard approaches for Encryption/Decryption Blowfish

	Encryption Time analysis		Decryption Time analysis	
File Size	AVG Time in seconds elapsed using the existing method	AVG Time in seconds elapsed using the proposed method	AVG Time in seconds elapsed using the existing method	AVG Time in seconds elapsed using the proposed method
10MB	2.724338	2.489142	0.002506	0.000501
20 MB	5.375882	4.848416	3.653145	3.143843
30 KB	8.200386	7.304710	5.396083	4.796802
40 MB	10.60201	9.698261	7.537973	6.549919
50 MB	13.50485	12.13322	9.053618	7.908793
60 MB	16.55609	14.57863	11.50914	9.921053
70 MB	20.71033	17.90042	14.36431	12.41386
80 MB	24.36797	21.27302	15.79634	13.09119
90 MB	27.85867	23.08000	17.74162	13.96394
100 MB	31.61525	26.10582	22.71033	17.90042

Fig. 4 shows the graphical representation of the analysis of the time elapsed using blowfish stream cipher system. The results shown in Fig. 4 are the analysis of the difference in the total time elapsed for the encryption process of Blowfish stream cipher using the proposed XOR free operation v/s existing XOR approach. The analysis of the proposed solution on Blowfish confirms the increased performance benefit.



File Size	10MB	20 MB	30 KB	40 MB	50 MB	60 MB	70 MB	80 MB	90 MB	100 MB
Difference in Decryption Time analysis	16.20	18.06	26.10	34.26	43.06	54.57	72.38	86.39	100.23	112.83
Difference in Encryption Time analysis	8.11	13.61	19.88	25.83	35.85	44.58	58.36	72.38	90.14	111.41

Fig. 4 .Average difference in time between the proposed XOR free approach v/s standard approaches for Encryption/Decryption Blowfish

Table II also shows analysis of the difference in the time taken for processing the Blowfish decryption based on the proposed XOR free method v/s the existing XOR approach. The result shown in Table II and in Fig. 4 is the difference in the total time computed based on proposed XOR free operations and existing XOR operations for blowfish by repeating the test for fifty times for every different input file size.

The results shows the proposed XOR free approach compared to existing XOR process achieves better time complexity. These reductions in the time for processing are due to the replacement of the existing XOR approach by the XOR free approach. XOR free approach is a just a value borrowing from the XOR vector space which is constructed well in advance. Usually the XOR vector space is constructed and stored while reading the input file for the cipher systems. The time required to construct the vector space is almost negligible.

The analysis made with respect to the time complexity shows the proposed XOR free approach takes less time compared to the standard method. It has also been observed that as the size of the input file increases the time complexity achieved via proposed XOR free approach is moderately higher compared to the standard method of operations.

V. CONCLUSION

Since majority of all cipher system works on binary orientation, will have XOR operations. This work tries to reduce the time taken for the operation of XOR. This is done by referring from the vector space constructed in advance. The expression $O\left(log_2 r\right)$, shows the result of the construction of vector space theoretically. The vector space is constructed as recursive matrix easily according to the proposed method. The theoretical result shows the time complexity of vector space generation using the proposed method has a great impact on the cipher system performance. The same is proved in case of experimental results. Result of the experimental setup shows the considerable improvement in the time complexity. The usage of XOR free approach in the places where XOR operations are involved offers the best outcome. The results are tested against the Advanced Encryption Standards and Blowfish and have achieved the moderately good. Which are shown in the Table I and Table II. And the same thing is shown in the fig. 3 and fig. 4 Further this XOR free method and construction of XOR table can be used in some of the cryptographic operations where large number of bit XOR is used.

REFERENCES

1. David J. Icove, “Collaring the cybercrook: an investigator’s view-In the information age, the cloak is the network and the dagger is the data packet”, IEEE Spectrum, Volume: 34, Issue: 6, pp. 31-36, June 1997.
2. J. Cortadella, J.M. Llaberia, “Evaluation of A+B=K conditions without carry propagation” in IEEE Transactions on Computers, Volume: 41,Issue: 11, pp. 1484 – 1488, Nov 1992.

3. Dan Nicolaescu, Alex Veidenbaum, Alex Nicolau "Reducing Data Cache Energy Consumption via Cached Load/Store Queue", in ISLPED'03, August 25–27, 2003, Seoul, Korea, Copyright 2003 ACM 158113682X/03/0008.
4. Baolei Mao, Wei Hu, AlricAlthoff, Janarbek Matai, Yu Tai, Dejun Mu, Timothy Sherwood, and Ryan Kastner, "Quantitative Analysis of Timing Channel Security in Cryptographic Hardware Design", in IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems, Vol. 37, No. 9, pp. 1719-1732, September 2018.
5. JianqiangLuo, MochanShrestha, LihaoXu, and James S. Plank.: "Efficient Encoding Schedules for XOR-Based Erasure Codes". IEEE Transactions on Computers, Vol. 63, No. 9, pp. 2259 - 2272 September 2014.
6. Joan Daemen, Vincent Rijmen, "The First 10 Years of Advanced Encryption", in IEEE Security & Privacy, Volume: 8, Issue: 6, pp. 72 - 74 Nov.-Dec. 2010.
7. L. Ali, I. Aris, F. S. Hossain, and N. Roy.: "Design of an Ultra-High-Speed AES Processor for Next Generation IT Security," - Computers and Electrical Engineering, Vol. 37, No. 6, pp. 1160–1170, Nov. 2011.
8. J. Daemen, V. Rijmen, "The Design of Rijndael: AES—the Advanced Encryption Standard", Springer, 2002, pp. 31-50.
9. O. Song, and J. Kim, "Compact Design of the Advanced Encryption Standard Algorithm for IEEE 802.15.4 Devices," Journal of Electrical Engineering & Technology, Vol. 6, No. 3, pp. 418–422, 2011.
10. XiaoqiangZhang, Ning WU, XinxingZheng.: "The Design Method of Compact Composite Field AES S-Box Based on AND-XOR Array Structure", 2017 12th IEEE Conference on Industrial Electronics and Applications, pp. 1882-1886, 978-1-5090-6161-7/2017.
11. J. Blomer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, "An XOR-based Erasure-Resilient Coding Scheme", Technical Report TR-95-048, International Computer Science Institute, Technical. University of California, Berkeley, CA, U.S.,August 1995.
12. Dag Arne Osvik, Adi Shamir, EranTromer, "Cache Attacks and Countermeasures: The Case of AES". D. Pointcheval (Ed.): CT-RSA 2006, LNCS 3860, pp.1-20, Springer-Verlag Berlin Heidelberg 2006.
13. Bruce Schneier. "The Blowfish Encryption Algorithm", World Wide Web Journal, Volume 2, Issues 3-4, O'Reilly & Associates, 1997, Retrieved October 25, 2008, http://www.schneier.com/blowfish.html.
14. Magnus Find, Mika Goos, MattiJarvisalo, PetteriKaski,MikkoKoivisto and Janne H. Korhonen, "Separating OR, SUM, and XOR Circuits", Journal of Computer and System Sciences, Volume 82, Issue 5, pp. 793-801, August 2016.
15. Feng Ding and Tongwen Chen, "Gradient-Based Iterative Algorithms for Solving a Class of Matrix Equations", IEEE Transactions on Automatic Control, Vol. 50, No. 8, pp. 1216-1221, August 2005.
16. Canadian Journal of Mathematics, "The Master Theorem of MacMahon", Vol. 20, No. 4, pp. 93-98, 1968.
17. http://www.cis.syr.edu/~wedu/minix/code.html.
18. https://www.schneier.com/academic/blowfish/download.html.

AUTHORS PROFILE

Mr. Radhakrishna Dodmane, Associate Professor,Dept. of CSE NMAM Institute of Technology, Nitte. He has completed his BE in CSE, masters in CSE and pursuing his PhD in the area "Cryptography and Network Security" from the University of VTU karnataka. His areas of interests are security and data communication. He has about 16 years of teaching experience. Published around eight papers in the international journals and conferences. Received the best paper award for one of his paper presented in the international conference organized in association with springer.

Dr. Ganesh Aithal, Professor and Vice-Principal, SMVI Technology and Management. Had completed his BE in Electrical Power, masters in Digital Electronics (M.Tech) and PhD in Electronics and Communication. He has published more than 30 papers in various international journal and conferences. His areas of interests are Cryptography and Network Security, Data Mining and Business Analytics. He has guided two PhD students and currently five students are pursuing under his guidance. He has book chapter under his name. He has about 32 years of teaching and research experience.

Dr. Surendra Shetty, Professor and Head, Dept. Of MCA, had completed his B.Sc. in 2001 and Master of Computer Applications during 2004. Dr. Surendra Shetty had been awarded his doctoral degree for his research work "Audio Data Mining Using Machine Learning Techniques" in 2013 from university of Mangalore. He has published more than 25 research papers in different international journals and conferences. He is currently guiding six research scholars. Dr. Surendra Shetty authored two book chapters in different publications entitled "Machine Learning Approach for Carnatic Music Analysis" and "Applications of Unsupervised Techniques for Clustering of Audio Data". He has received research grant of 20 lakhs from VGST (GoK) for carrying out research on "Automatic Natural Language Processing and Speech Disorder Problems in Kannada Language". He has 15 years of teaching experience. The Research areas of interest are Cryptography, Data mining, Pattern Recognition, Speech Recognition, MIS, Software Engineering and Testing.