

A Private Key Encryption Scheme based on Amicable Number with User defined Cipher Block Sequencing Techniques

R. Das^{1*}, S. Dutta²

¹Development Officer, Bankura University, Bankura-722155, W.B., India

²Department of Computer Applications, Dr. B. C. Roy Engineering College, Durgapur-713206, WB, India

*Corresponding Author: ramkrishnadas9@gmail.com., Tel.: +91-9647000821

Available online at: www.ijcsonline.org

Accepted: 19/May/2018, Published: 31/May/2018

Abstract-Same secret key is used for encryption and decryption in Private Key cryptography. Many of the existing private key cryptography systems lacking their security as distribution of the private-key through public communication channel without interpretation is very hard to achieve. So here our focus is on the secret procedure which retrieves secret value used for encryption from the private-key rather than securing the actual private-key value. The secret value is being derived by taking the first or second item from N^{th} pair of Amicable number set. Where N is a user defined positive integer in the range of ($1 \leq N \leq 1024$). The counting of N^{th} pair is being started from a user defined base value towards its forward or backward direction. We impose another level of security by implementing some user defined sequence for positioning encrypted characters block wise in cipher text file. Thus an attempt is made to increase the security in great extent.

Keywords-Amicable number, private key encryption, block cipher, character's positioning sequence.

I. INTRODUCTION

Cryptography is the techniques for secure transmission of information (text, image, audio, video etc.) through public communication channel in between groups or individuals. Private Key cryptography defines as an encryption method where same identical secret key is used for encryption and decryption or two keys derivable from each other. Noticeably, many of the existing private-key encryption systems suffer from lack of security as distribution of the private-key through public communication channel without interpretation is very hard to achieve [8,9,10].

Here our focus is on the secret procedure which derived secret value from the private key rather than only securing the actual private-key value. The secret value is being used both for encryption and decryption. The secret value is generated by computing either first or second amicable number from N^{th} pair of amicable number set. Selection of first or second amicable number is defined by user choice. The value of N is user defined and N is a positive integer in the range of ($1 \leq N \leq 1024$). Starting from a user defined base value the counting of N^{th} pair of amicable number is done towards its forward or backward direction. The selection between forward and backward movements is being determined by the user-input. Moreover, we put some additional security by imposing some user defined

sequence for positioning encrypted characters block wise in cipher text file.

As the base value, forward or backward movement and N^{th} term for computing amicable number is user defined, so changing of these values will generate a new amicable number as a secret value used for encryption. Thus increase the security in a great extent. Beside these, encrypted character positioning in cipher text file corresponding to the character in plain text file is different as user defined cipher block sequencing techniques are implemented here. So the security of the system is increased.

In this paper, Section-II demonstrate the background study. Section-III represents the preliminaries. Overall procedure is discussed in section-IV. Section-V describes the encryption process. Section-VI demonstrates the decryption process. Experimental results are being described in section-VII and Section-VIII draws the conclusions.

II. BACKGROUND STUDY

In 2009, Fibonacci based position substitution (FBPS) is represented by J. K. Mandal and Mangalmay Das where positions of different bits of a plain text character are substituted by using Fibonacci numbers and thus generate the cipher text [1]. In 2013, Udepal Singh and Upasna Garg demonstrate ASCII value based encryption technique

where ASCII values of plain text character and randomize key are being combined using mathematical operation to generate the cipher text [2]. In 2015, Laiphrakpam Dolendro Singh and Khumanthem Manglem Singh introduce an idea of text encryption using elliptic curve cryptography where corresponding ASCII values of plain text are paired up and those values considered as input for elliptic curve cryptography [3]. AES based text encryption with dynamic key selection is demonstrated by Nishtha Mathur and Rajesh Bansode in 2016 where the key length has been increased to 192 bit and number of iterations taken will be 12 as it increase the performance of existing AES scheme [4]. In 2017, Snehal Sherkhane, Amit Waghmare, Suraj Dalvi and Shreya Bamne introduced Hybrid Data Encryption scheme using color code and armstrong number where combination of armstrong numbers and color codes generate secret key used as password in turn to decrypt the file [5]. In 2018, Kiran Kumar R and Bharathi Devi P introduced text encryption algorithm using DNA ASCII table where mapping between DNA sequences to numerical data is done and it converts into binary and then turns into DNA bases [6].

III. PRELIMINARIES

A. Amicable Number

Amicable numbers are a pair of numbers where the sum of all of the proper divisors (excluding itself) of the first number is exactly equals the second number and vice versa. For example 220 and 284 are amicable numbers. The sum of proper divisors of 220 is sum (1, 2, 4, 5, 10, 11, 20, 22, 44, 55, and 110) that is 284. Similarly the sum of proper divisors of 284 is sum (1, 2, 4, 71, and 142) and that is 220. Therefore 220 and 284 are amicable numbers.

B. Cipher Block Sequencing Techniques

Table 1 demonstrate different cipher block sequencing techniques.

Table 1. Different Cipher Block Sequencing Techniques

Choice Value	Choice Term	Definition of Sequencing Technique
00	EEFR (Exchange Even From Right)	Even numbered block's characters are exchanged starting from right end. Odd numbered block's characters are in same order in final encrypted file.
01	EEFL (Exchange Even From Left)	Even numbered block's characters are exchanged starting from left end. Odd numbered block's characters are in same order in final encrypted file.
10	EOFR (Exchange Odd From Right)	Odd numbered block's characters are exchanged starting from right end. Even numbered block's characters are in same order in final encrypted file.
11	EOFL (Exchange Odd From Left)	Odd numbered block's characters are exchanged starting from left end. Even numbered block's characters are in same order in final encrypted file.

Let the fixed block size is 32-bits. So each block having 4 numbers of characters. Figure-1 represents different cipher block sequencing techniques.

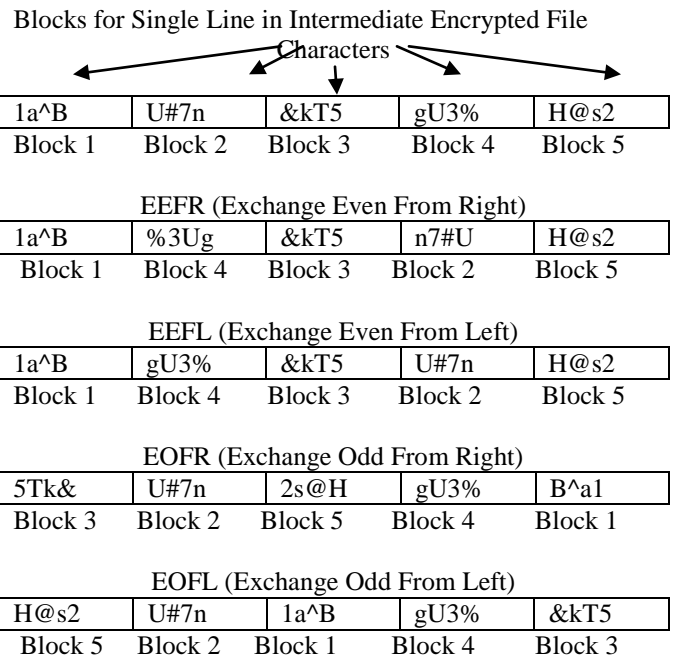


Figure 1: Representation of different cipher block sequencing techniques.

IV. OVERALL PROCEDURE

Step 1: Read single character from plain text file and convert it into 8 bit binary representation.

Step 2: take the user inputs for base value, Nth term for amicable number, selection of forward or backward movement and cipher block sequencing technique and formation of private key is done.

Step 3: Generate the secret value by using the values of different blocks of private key and the secret procedure.

Step 4: Convert the secret value in binary representation and perform bitwise XOR between the bits of plain text character and the secret value. Thus generate an encrypted character. All the characters of plain text file will be encrypted in this manner and intermediate encrypted file will be generated.

Step 5: Rearrange all encrypted characters in cipher text file as per cipher block sequencing technique chosen by user and It generates final encrypted file.

Step 6: Encrypted file and private key will be shared to receiver and receiver follows reverse procedure for decryption.

V. ENCRYPTION PROCESS

A. Plain Text Formation

Read single character at a time from plain text file till the entire file is visited and convert each character into 8-bit binary representation and store it into the array called PLAINTEXT[].

Example: - Let 'a' is a character read from plain text file. We convert it into 8-bit binary representation and store into array called PLAINTEXT [].Figure-2 represents the entire procedure.

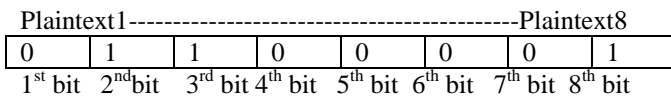


Figure 2: Formation of Plain Text

B. Private key Generation

The size of the private-key is 40 bits having 5 blocks. 1st 1-bit block represents the choice between forward and backward movements from the user-defined base value. 2nd 10-bit block holds Nth numbered pair of amicable number. 3rd 1-bit block holds the value for selecting the either first or second number from Nth amicable number pair. 4th 26-bit block holds the user-defined base value. 5th 2-bit block define the different cipher block sequencing techniques used for storing the encrypted characters in cipher text file. Figure-3 represents block diagram of the 40-bit private-key.

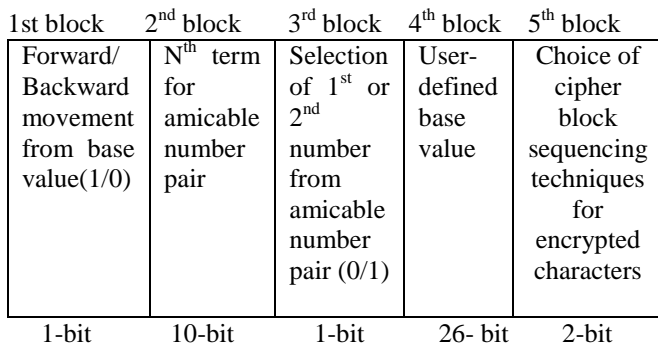


Figure 3: Block Diagram of 40-bit Private-key

Read user inputs and stores their corresponding binary value in their respective blocks in an array called KEY[] with dimension 40.

C. Formation of Secret Value from Private Key

We search the Nth pair of amicable number by making a forward or backward movement from user defined base

value in selective mode. Where N is a user defined positive integer in the range of (1<=N<=1024). Now we select first or second amicable number from that Nth amicable number pair in selective mode which will be considered as secret value used for encryption or decryption.

Step 1. Determine the movement (forward or backward) and the Nth pair of amicable number by using the value of the 1st and the 2nd block of the private-key respectively. Find out Nth amicable number pair by making a forward or backward movement from the user-defined base value.

Step 2. Selection of either first or second amicable number from Nth amicable number pair is determined from 3rd block value of the private key. Now we make 24-bit binary representation of that amicable number and store that value into the array DERIVEDVALUE[] with dimension 24. This example describes private key formation and secret value generation process from the private key. Figure-4 demonstrates bit wise representation of private key for some input value.

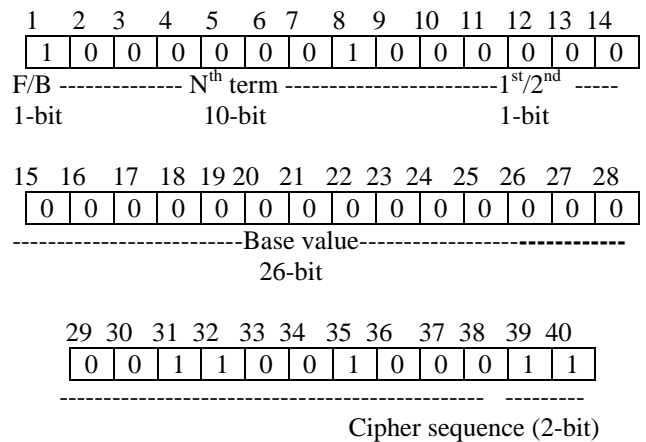


Figure 4: 40-bit Binary Representation of Private-key for a Specific Value.

User defined base value is 200 contained into 4th block of the private key. 1st bit of private key is 1 so forward movement is made (201 .202..) from base value .Next block represents the Nth term which is 8. So we take 8th amicable number pair for consideration with a forward movement from base value. Selection of first or second number from Nth amicable number pair is determined by the value of 3rd block of the private key. Here the value is '0' so we take first number which is 17296. We convert it into 24 bit binary representation and store it into the array called DV[]. Figure-5 represents the 24-bit representation of secret value.

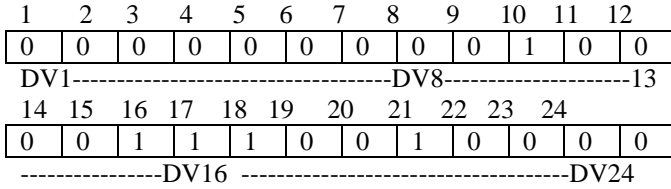


Figure 5: 24-bit Representation of Secret Value derived from Private Key.

D. Bitwise XOR operation and Formation of Cipher Text
 Bitwise XOR operation is being performed between the 8-bit binary representation of plain text character and 24-bit binary representation of secret value cumulatively. 3rd block, 2nd block and the 1st block each of which 8 bits of the secret derived value (DV) are used for performing XOR with 8-bit plain text character value cumulatively. After 3 times, we get binary value of corresponding ASCII code of a final encrypted character. In this way all characters of plain text file are encrypted. Figure-6 demonstrates entire XOR process where IET means intermediate encrypted text.

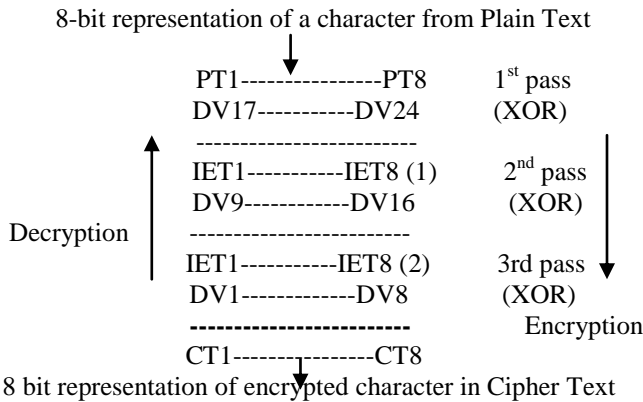


Figure 6: Block wise Cumulative XOR between Plain Text's character and Secret Value.

Example- We read a character 'a' from plain text whose ASCII value is 97. We generate binary representation of plain text and secret value and perform XOR operation between them. Figure-7 represents the XOR procedure.

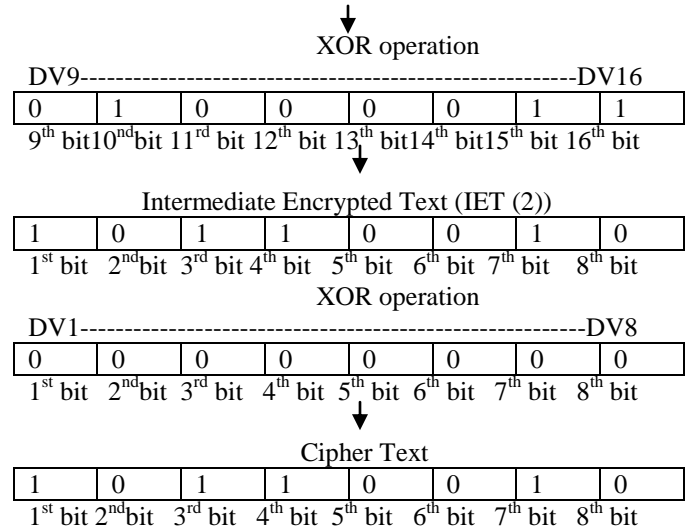
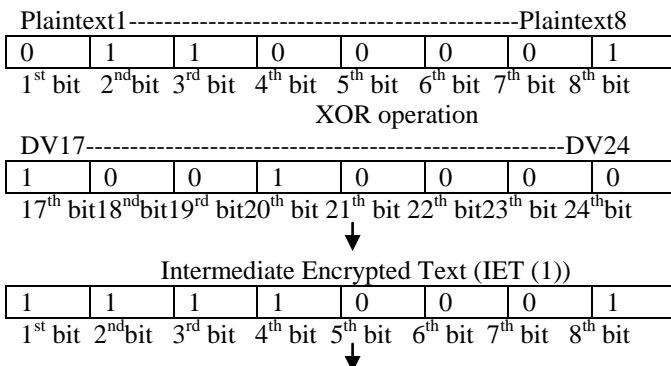


Figure 7: Generation of Cipher Text's Character by Block Wise Cumulative XOR operation between Plain Text's Character and Secret Value derived from Private Key.

ASCII value corresponding to the cipher text is 178 and the symbols corresponding to that ASCII is . All characters are encrypted in that manner and stored into the intermediate encrypted file.

E. Positioning of Encrypted character's blocks in cipher text

The intermediate encrypted file is divided into fixed size blocks and are numbered sequentially 1...n, (where n is a positive integer). The user defined cipher block sequencing techniques value is contained in the 5th block of the private key which is used for positioning encrypted characters in cipher text file. The definition of cipher block sequencing techniques are given in Table-I. Encrypted character's blocks are being positioned as per user chosen sequencing technique among four choices and final encrypted file is generated from intermediate encrypted file. Final encrypted file (cipher text) is sent to the receiver with secret private key file.

VI. DECRYPTION PROCESS

A. Repositioning of Encrypted character's block in Cipher text

The blocks of cipher text file are repositioned by using the same cipher block sequence technique (using step E in encryption section) which is used at encryption time. Now we get actual sequence of encrypted characters in cipher text corresponding to source character sequence in the plain text file.

B. Conversion of Cipher Text character into binary representation

Receiver visits single character at a time from cipher text file till entire file is visited and converts that character into


8 bit binary representation and store it into an array called CIPHERTEXT [].

C. Formation of Secret Value from Private-key

Derive secret value from private key (using step C in encryption section) and store 24-bit binary representation of that secret value into an array called DERIVEDVALUE [].

D. XOR operation and Formation of Decrypted Text

Perform XOR operation between array CIPHERTEXT [] and DERIVEDVALUE [] where 1st block, middle block and last block of array DERIVEDVALUE [] is taken for XOR operation in 1st pass, 2nd pass and in 3rd pass respectively. Intermediate result is stored into an array IET [] and final result is stored into array DECRYPTED []. Corresponding ASCII code is generated from array DECRYPTED [] and from there we get the decrypted character and store it in decrypted text file.

Example- we read an encrypted symbol  from the cipher text whose ASCII value is 178. We convert secret value and encrypted character into binary and perform XOR operation between them. Figure-8 represents the entire procedure.

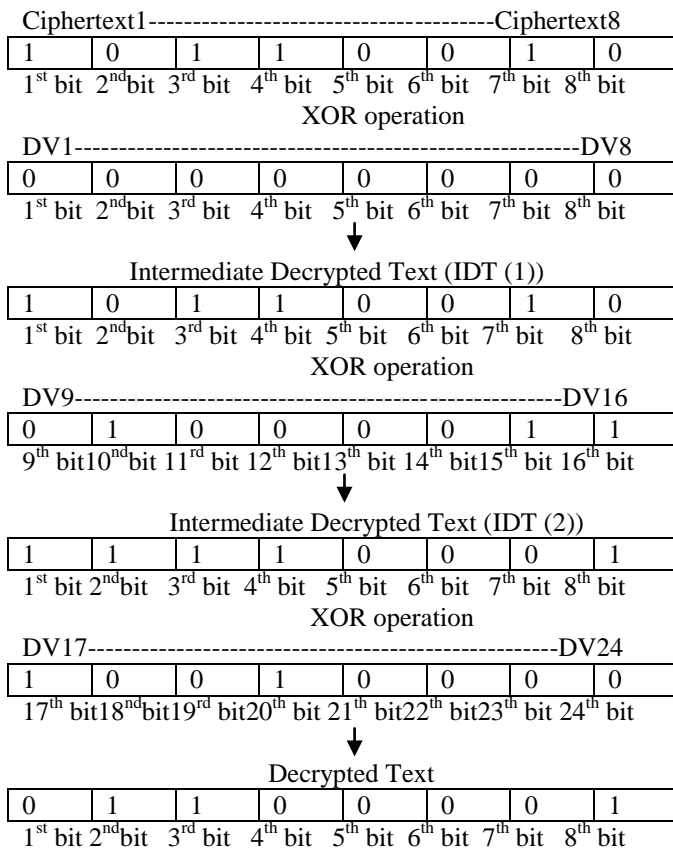


Figure 8: Generation of Decrypted text by Block Wise Cumulative XOR operation between Cipher Text's character and Secret Value derived from Private Key.

ASCII value in decrypted text is 97 and the character corresponding to that ASCII is 'a'. In this way all characters are decrypted.

VII. EXPERIMENTAL RESULT & DISCUSSION

The encryption of a plain text is done by using 1st item of 8th pair of amicable number set with a forward movement from the user defined base value 200. The value of the amicable number is 17296. EOFL cipher sequencing is used as 5th block's value in the private key is 11. The encryption or decryption has taken 131530 milliseconds. Table-2 demonstrates the content of the source files, encrypted files and the decrypted file.

Table 2. Corresponding content of source, encrypted, decrypted file

Content of Source File (a.txt)	Content of intermediate Encrypted File (a_jen.txt)	Content of final Encrypted File (a_en.txt)	Content of Decrypted File (a_de.txt)
AVKZ-- (@xpqt1058	??ç³ÔÄÄ©???? ØÛÛÑ	????ÔÄÄ©?? ç³ ØÛÛÑ	AVKZ-- (@xpqt1058

We have executed our program on different types of files (*.com,*.txt,*.exe,*.sys and *.dll). We derived 2nd item from 13th pair of amicable number set as a secret value which is 88730. That secret value is used for encryption and decryption. The Execution results are being displayed in the Table 3, Table 4, Table 5, Table 6 and Table 7.

Table 3. Execution Result for *.COM files

Source File name	Source File size (KB)	Encrypted File size (KB)	Encryption / Decryption time (Mille seconds)
diskcomp	9	9	891368
graphics	19.2	19.2	2112340
kb16	14.3	14.3	1383434
loadfix	1.10	1.10	133367
README	4.11	4.11	402355

Table 4. Execution Result for *.DLL files

Source File name	Source File size (KB)	Encrypted File size (KB)	Encryption / Decryption time (Mille seconds)
iconlib	2.50	2.50	249273
KBDAL	6.50	6.50	608651
MSMH	19.3	19.3	1923830
panmap	10.0	10.0	954704
tcpmib	14.5	14.5	1423771

Table 5. Execution Result for *.EXE files

Source File name	Source File size (KB)	Encrypted File size (KB)	Encryption / Decryption time (Mille seconds)
cacls	19.5	19.5	1907236
label	9.50	9.50	862988
mqsvc	4.50	4.50	406848
Shadow	14.5	14.5	1315419
WINSTUB	1	1	112897

Table 6. Execution Result for *.SYS files

Source File name	Source File size (KB)	Encrypted File size (KB)	Encryption / Decryption time (Mille seconds)
partmgr	19.2	19.2	1770073
rootmdm	5.75	5.75	527221
sffp_mmc	10.0	10.0	908839
Smclib	14.2	14.2	1308089
VIAPCI	2.64	2.64	244524

Table 7. Execution Result for *.TXT files

Source File name	Source File size (KB)	Encrypted File size (KB)	Encryption / Decryption time (Mille seconds)
AAB	19.5	19.5	1744230
LICENSE	4.71	4.71	432543
ReadMe	1	1	42826
ROMAN	14.0	14.0	1376958
TechNote	9.01	9.01	855701

Figure 9 represents that there is a proportionate relationship between execution times and the size of source file.

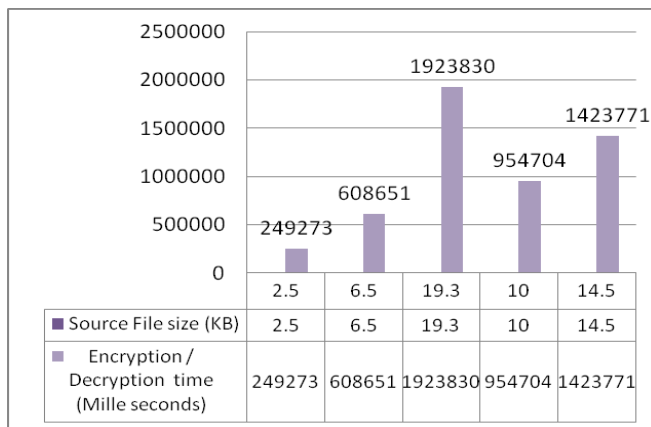


Figure 9: Relationship between Encryption Time and Source File Size

The Pearsonian Chi-square test is used here to decide that the degree of arisen of characters of encrypted files from a specified population. The Pearsonian Chi-square is defined as follows: $X^2 = \sum \{(f_0 - f_e)^2 / f_e\}$ where f_e and f_0 respectively stand for the frequency of a character in the source file and that of the same character in the corresponding encrypted file. [7]

High Chi-square value means there is a higher frequency in source file and a lower frequency in encrypted file for a specific character. Table VIII, Table IX, Table X, Table XI and Table XII indicate that the proposed encryption procedure having highly satisfactory chi-square values for all types of files.

Comparison between proposed technique and AES-256 bit encryption technique based on Chi Square test result value on different types of inputted files are represented in Table 8, Table 9, Table 10, Table 11 and Table 12.

Table 8. Execution Result for *.COM files

Source File name	Source File size (KB)	Chi-square Test Value for proposed Technique	Chi-square Test Value for AES-256 bit Encryption Technique
diskcomp	9	59162.722656	9216.000000
graphics	19.2	90349.453125	19694.000000
kb16	14.3	49740.839844	14710.000000
loadfix	1.10	1849.756104	1131.000000
README	4.11	10640.603516	4217.000000

Table 9. Execution Result for *.DLL files

Source File name	Source File size (KB)	Chi-square Test Value for proposed Technique	Chi-square Test Value for AES-256 bit Encryption Technique
iconlib	2.50	6672.274902	2560.000000
KBDAL	6.50	44474.777344	6656.000000
MSMH	19.3	34499.722656	19768.000000
panmap	10.0	75107.367188	10240.000000
tcpmib	14.5	96802.875000	14848.000000

Table 10. Execution Result for *.EXE files

Source File name	Source File size (KB)	Chi-square Test Value for proposed Technique	Chi-square Test Value for AES-256 bit Encryption Technique
cacls	19.5	127799.695313	19968.000000
label	9.50	76094.953125	9728.000000
mqsvc	4.50	29264.796875	4608.000000
Shadow	14.5	66915.531250	14848.000000
WINSTUB	1	634.412964	578.000000

Table 11. Execution Result for *.SYS files

Source File name	Source File size (KB)	Chi-square Test Value for proposed Technique	Chi-square Test Value for AES-256 bit Encryption Technique
partmgr	19.2	112557.718750	19712.000000
rootmdm	5.75	26591.115234	5888.000000
sffp_mmc	10.0	58059.800781	10240.000000
Smclib	14.2	58216.992188	14592.000000
VIAPCI	2.64	7212.227051	2712.000000

Table 12. Execution Result for *.TXT files

Source File name	Source File size (KB)	Chi-square Test Value for proposed Technique	Chi-square Test Value for AES-256 bit Encryption Technique
AAB	19.5	7609.227051	20000.000000
LICENSE	4.71	6572.482910	4829.000000
ReadMe	1	242.766479	296.000000
ROMAN	14.0	22061.707031	14423.000000
TechNote	9.01	13500.411133	9232.000000

Figure 10 graphically shows that the proposed scheme having very highly satisfactory chi-square values compare to AES-256 bit encryption scheme.

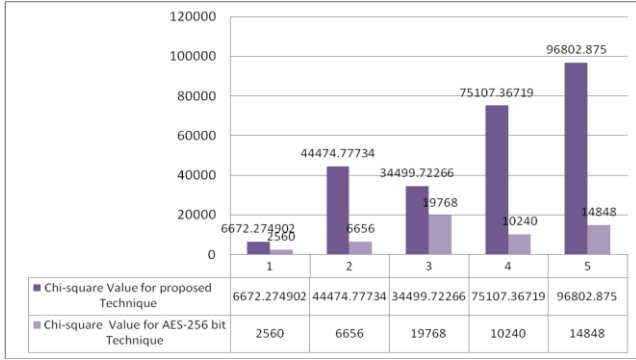


Figure 10: Comparison between proposed technique and AES-256 bit technique based on Chi Square test result value

Comparison between proposed technique and AES-256 bit encryption technique based on Degree of Freedom value on different types of inputted files are represented in Table 13, Table 14, Table 15, Table 16 and Table 17.

Table 13. Execution Result for *.COM files

Source File name	Source File size (KB)	Degree of Freedom Value for proposed Technique	Degree of Freedom Value for AES-256 bit Encryption Technique
diskcomp	9	255	245
graphics	19.2	255	254
kb16	14.3	255	255
loadfix	1.10	254	145
README	4.11	255	242

Table 14. Execution Result for *.DLL files

Source File name	Source File size (KB)	Degree of Freedom Value for proposed Technique	Degree of Freedom Value for AES-256 bit Encryption Technique
iconlib	2.50	255	113
KBDAL	6.50	255	227
MSMH	19.3	255	255
panmap	10.0	255	243
tcpmib	14.5	255	254

Table 15. Execution Result for *.EXE files

Source File name	Source File size (KB)	Degree of Freedom Value for proposed Technique	Degree of Freedom Value for AES-256 bit Encryption Technique
cacls	19.5	255	252
label	9.50	255	247
mqsve	4.50	255	208
Shadow	14.5	255	255
WINSTUB	1	232	45

Table 16. Execution Result for *.SYS files

Source File name	Source File size (KB)	Degree of Freedom Value for proposed Technique	Degree of Freedom Value for AES-256 bit Encryption Technique
partmgr	19.2	255	255
rootmdm	5.75	255	231
sffp_mmc	10.0	255	251
Smclib	14.2	255	254
VIAPCI	2.64	255	190

Table 17. Execution Result for *.TXT files

Source File name	Source File size (KB)	Degree of Freedom Value for proposed Technique	Degree of Freedom Value for AES-256 bit Encryption Technique
AAB	19.5	255	255
LICENSE	4.71	255	73
ReadMe	1	201	33
ROMAN	14.0	255	82
TechNote	9.01	255	78

Figure 11 graphically shows that the degree of freedom for the proposed system is very much higher than the degree of freedom value for AES-256 bit Encryption Technique. Degree of freedom defines the number of independent characters in encrypted file as related to source file. So higher degree of freedom value indicates the propose scheme is more secured as compare to AES-256 bit Encryption Technique.

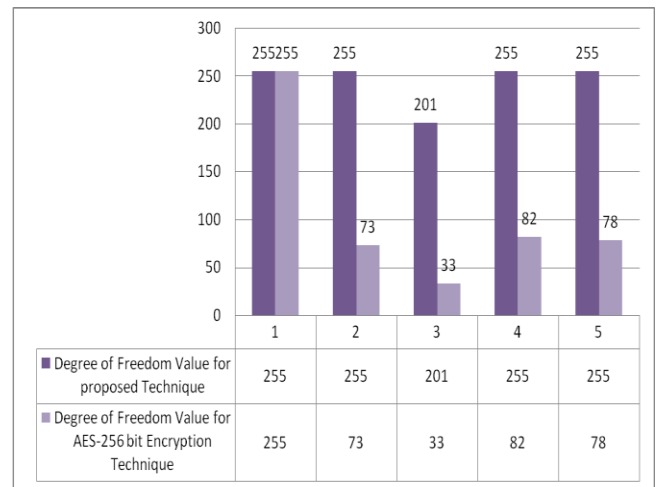


Figure 11: Comparison between proposed technique and AES-256 bit technique based on Degree of Freedom value

VIII. CONCLUSIONS

Bitwise XOR operation between the plain text and the secret value is performed in proposed private-key encryption scheme where the secret value is the either first or second amicable number from the Nth set, counted by making a forward or backward movement from the user-defined base value. Where N is user specified positive integer in the range of (1<=N<=1024). Changing of base value or Nth set will find a new amicable number. Thus the security is increased.

The security of the scheme depends on securing the private key value as well as the secret value derivation procedure. So unauthorized access is only been possible if both the knowledge of key and secret value derivation procedure is available. Thus the security is increased in a great entrance.

As the placement ordering sequence of encrypted characters corresponding to the source character sequence in plain text is specified by user choice, so prediction of an encrypted character corresponding to its source character is very difficult. Thus the security is increased.

Encryption of a source file may also be done by using multiple numbers of distinct private-keys where each key is applied on a specific block among multiple numbers of user-defined blocks in source file. So an attempt is made to increase the security.

As the encryption is done in bit level, so no additional memory is needed for encryption and the encryption time is depends on the file size not on the type of the file.

The only restriction is that if N^{th} term is very higher then encryption time is higher as it takes more time to compute N^{th} amicable number.

REFERENCES

- [1] J. K. Mandal and Mangalmy Das, "Fibonacci Based Position Substitution (FBPS) Encoder for Secured Message Transmission", IEEE International Advance Computing Conference (IACC) Patiala, India, pp.964-970, 6-7 March 2009.
- [2] Udepal Singh and Upasna Garg, "An ASCII value based text data encryption System", "International Journal of Scientific and Research Publications (ISSN 2250-3153)", Volume 3, Issue 11, pp. 1-5, November 2013.
- [3] Laiphrakpam Dolendro Singh and Khumanthem Manglem Singh, "Implementation of Text Encryption using Elliptic Curve Cryptography", "Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015)", "Procedia Computer Science", doi: 10.1016/j.procs.2015.06.009, Volume 54, pp. 73-82, 2015.
- [4] Nishtha Mathur and Rajesh Bansode, "AES Based Text Encryption Using 12 Rounds with Dynamic Key Selection", "7th International Conference on Communication, Computing and Virtualization 2016", "Procedia Computer Science", doi: 10.1016/j.procs.2016.03.131, Volume 79, pp. 1036-1043, 2016.
- [5] Snehal Sherkhane, Amit Waghmare, Suraj Dalvi, Shreya Bamne, "Hybrid Data Encryption using Colour code and Armstrong number", "International Journal of Engineering Science and Computing", Volume 7, Issue 4, pp. 10300-10305, April, 2017.
- [6] Kiran Kumar R and Bharathi Devi P, "A Novel Text Encryption Algorithm Using DNA ASCII Table With A Spiral Approach", "International Journal of Recent Scientific Research", Volume 9, Issue 1, pp. 23588-23595, DOI: 10.24327/ijrsr.2018.0901.1495, 2018.
- [7] J. K. Mandal, S. Dutta, "A 256-bit recursive pair parity encoder for encryption", Advances D -2004, Vol. 9 n°1, Association for the Advancement of Modeling and Simulation Techniques in Enterprises (AMSE, France), www. AMSE-Modeling.org, pp. 1-14.
- [8] William Stallings, Cryptography and Network security: Principles and practice (Second Edition), Pearson Education Asia, Sixth Indian Reprint 2002.
- [9] Atul Kahate (Manager, i-flex solution limited, Pune, India), Cryptography and Network security, Tata McGraw-Hill Publishing Company Limited.
- [10] Private Key cryptography on text [online] Available: <https://koolspan.com/private-key-encryption/>.

Authors Profile

Mr. Ramkrishna Das has achieved MBA, M.Tech (C.S.E.), MCA degree and currently pursuing PhD from Vidyasagar University. He has working experience over 9 years as Assistant Professor in Haldia Institute of Technology and 2 years as Teacher Government Administrative



Development Officer in Bankura University, INDIA. He has published 3 books from international publisher including Lambert Academic Publishing, GERMANY. He has published 20 numbers of research articles in international journals (including Scopus and UGC indexed journals) and conferences (including IEEE and Springer conferences).

Dr. Saurbh Dutta pursued Ph. D. (Computer Science), MCA, B. Sc. (Mathematics). His publications, experiences and achievements are furnished below:



Publication of book-2.
Research papers publications: 43, Foreign Journal Publications - 11 Peer-Reviewed International Conference Papers - 8, Indian Journal Publications - 2, Published/Accepted in International Conference - 7, others - 3. National Conference Publications - 12.
Research experience- 14 years.

Specialization: Information Security and Cryptology. Membership - IACSIT, IJIST, IEDRC, IJSMRD, IJMCAR, IJCSS, IJMER, ISOC, IAENG, IJCSI. Reviewer-(IJNS), Taiwan AMSE, France, IJACSA, ICCIA, ACCT12.

Award- Enlisted in the directory of Marquis Who's Who in the World in its 2010 edition. Biography selected for inclusion by ABI (American Biographical Institute, Inc.) in the list of International Profiles of Accomplished Leaders in its 2011 edition.