An Approach of Private-Key Encryption Technique Based on Multiple Operators And Nth Even or Odd Term For Even or Odd Bit Position's Value of A Plain Text's Character

Ramakrishna Das, Saurabh Dutta

Abstract: - Private-key cryptography is a cryptographic system that uses the same secret key to encrypt and decrypt messages. The problem with this method is transmitting the secret key to the person that needs it without it being intercepted

Here, we have proposed an idea to increase the security of private-key encryption technique. We have focused on the secret procedure to retrieve secret value from the private-key rather than securing the actual private-key value. The encryption is done by the secret value derived from the private-key. The secret value is being derived by making arithmetic operation between two decimal values. Those decimal values are derived by taking the binary values from even and odd bit position of a plain text's character respectively. The operators are supplied by the user. As the formations of '0' and '1' are different for each distinct character in the plain text file, so we get different or may be distinct secret private-key value for each of the distinct character in the plain text. Thus the security is increased.

Index Terms:- Even or odd bit position, Nth Even or Odd term, Private-key Encryption, Stream Cipher.

I. INTRODUCTION

Private-key cryptography refers to an encryption method where both the sender and the receiver share the same secret key or their keys are different, but they are related in an easily computable way. The Fig-1 represents a private-key encryption method where the same secret key is being used for encryption and decryption.

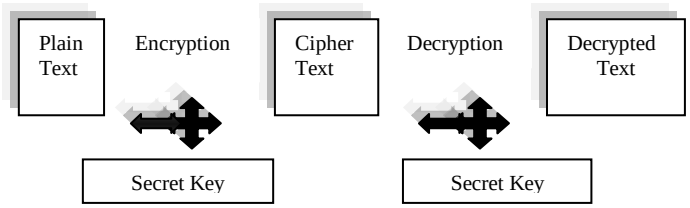

Figure 1: A Private-key Encryption Scheme

The main problem of private-key encryption technique is to distribute the private-key securely. All the existing private-key encryption systems suffer from lack of security. Here, we have proposed a scheme to increase the security of the existing private-key encryption system. We have developed a secret procedure which is responsible to retrieve the secret value from the private-key.

Manuscript received on April, 2013.

Ramkrishna Das, Department of Computer Applications, Haldia Institute of Technology, Haldia, INDIA.

Saurabh Dutta Department of Computer Applications, Dr B.C.Roy engineering College, Durgapur, INDIA.

The value is being used for encryption and decryption. The secret value is being generated by making different arithmetic operations (operations for odd values, operations for even values, base operations) on decimal values. Those decimal values are derived by taking the binary values from even and odd bit position of 8-bit representation of a plain text character respectively. The operators are given by the user. As the formations of '0' and '1' are different for each distinct character in different bit position, so we get different or may be distinct secret private-key value for each of the distinct character in the plain text. Combination of user defined operators and derived secret value from even or odd bit position from plain text construct the private-key. Hence the security is increased as we focus on securing the retrieving procedure rather than the private-key value. Secret value can't be retrieved without the knowledge of the retrieving procedure.

In this paper Section-B describes the encryption process; Section-C describes the decryption process. Experimental results are being described in section-E and section-F draws the conclusion.

II. ENCRYPTION PROCESS

A: Plain Text Formation:-

Let 'a' is a character which is present in the inputted file. Fig-2 represents its 8-bit binary representation through an array PLAINTEXT with dimension 8.

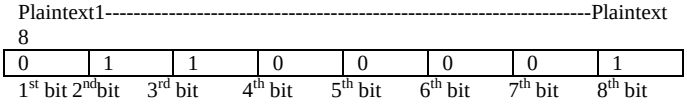

Figure 2: Formation of Plain Text

Step-A.1 Read one character at a time from the inputted file till we reached to the end of the file. Convert each character into 8-bit binary representation and store the value into the array PLAINTEXT with dimension 8 [1] [2] [3].

B: key Generation: -

The size of the private-key is 32 bits having 7 numbers of blocks. 1st block having size of 6 bits represent the operator for the even decimal values where the decimal value is being derived from the even or odd position's binary value of the plain text.2nd block is having size of 6 bit represent the operator for odd value. 3rd block holds the value to determine

that 4th block's value is even or odd. Size of the 3rd block is 1bit. 4th block holding the value which represents the Nth even or odd term in the range of 0 to 2^6 -1.Nth term will be calculated from the even position's bit value of the plain text's character where N is a positive integer in the range of (1<=N<=∞). The size of the 4th block is 6 bits. 5th block holds the value to determine that 6th block's value is even or odd. Size of the 5th block is 1bit. 6th block's value represents the Nth even or odd term in the range of 0 to 2^6 -1. Nth term will be calculated from the odd position's bit value of a plain text's character. The size of the 6th block is 6 bits. 7th block holds the base operator. The size of the 7th block is 6 bits [1]. Fig-3 represents block diagram of the 32 bits Private-key.

1st block	2nd block	3rd block	4th block
Operator for even values o(ep)	Operator for odd values o(op)	Code for even odd (0/1)	Nth even / odd term for the even position's bit value of a plain text's character. n(ep)
6 bit	6 bit	1 bit	6 bit

5th block	6th block	7th block
Code for even odd (0/1)	Nth even / odd term for the even position's bit value of a plain text's character. n(op)	Base operator O(b)
1 bit	6 bit	6 bit

Figure-3: Block diagram of 32 bit Private-key

*Step B.**1.*** Read the user inputs for 1st, 2nd, 7th block from the user.

Step B.2. The value for the 3rd, 4th, 5th, 6th will be calculated from the plain text. Convert the input values into corresponding bit size of their respective blocks and store the values in an array KEY with a dimension 32.

C: Formation of Secret Value from Private-key for encryption:

Two Decimal values are derived from the plain text. Decimal value for even bit (d (ev)) position is the value which is generated by taking the binary value of the even bit position of the plain text's character. Another one i.e. decimal value for odd bit (d (ov)) position is generated by taking the binary value of the odd bit position of the plain text's character.

Step C.1. Determine the Nth even or odd term in the range of 0 to 2^6 -1 where Nth term is calculated from the value of d(ev).The d(ev) value is the Nth odd or even value in the range of 0 to 2^6 -1 that value is stored in n(ep).The value of n(op) is calculated in the similar way from the value of d(ov).The d(ov) value is the Nth even or odd term which is stored in n(op).

Step C.2. The value (n (ep)) is an even or odd value. We perform an operation between two n (ep) values and the result is stored in result1. The operation (operator for even/operator for odd) is selected depending on nature (even/odd) of the value n (ep). Similar activity is done for n(op) and result is stored in result2.

Step C.3. We will get the final derived value by performing the operation between result1 and result2. The operation will be operated by base operator.

Step C.4. Make 8 bit binary representation of that Final derived value and store that value in the array DERIVEDVALUE with dimension 8.

Example:-The example demonstrates the private-key Formation and the Secret Value generation procedure. Fig-4 represents the bit wise representation of a private-key for some specific value.

-Operator for even (*)-Operator for odd (+) - even/odd ----

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	1	0	1	0	1	0	1	0	1	1	0	0	0	0

6bit 6bit 1 bit 6 bit

Nth Even/odd term for even position's Value in PT (6) -Nth Even/odd term for even position's Value in PT (3)

17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	1	0	1	0	0	0	0	1	1	1	0	1	1	0	1

1 bit 6 bit 6 bit

Even/odd Base operator (-)-

Figure 4: 32 bit Private-key for Specific Value.

Operator for even value, operator for odd value and the Base operator are taken from user which are *, + and – respectively. Convert these operators into corresponding ASCII codes and corresponding binary value is stored in 1st block, 2nd block and 7th block respectively. We have read a character from the plain text. Let 'r' is the character. The ASCII code of 'r' is 114. Now we have converted that ASCII code into 8 bit binary representation. This is 01110010.Then we take the even position's value from this binary representation. that is 1100 and also odd positions value that is 0101.Now convert these two binary value into decimal which is 12 and 5.12 is the 6th even number and 5 is the 3rd odd number in the range of (1-2^6). So we perform 6*6 where * is operator for even value and we perform 3+3 where + is the operator for odd value. Now we perform – between these two results where – is the base operator. So the final result is (6*6)-(3+3). That is 30. We convert it into 8 bit binary representation and use it for encryption and decryption. The secret value is derived by using the private-key. The secret value is stored in an array DV with dimension 24. Fig-5 represents the entire procedure

Example- one character from plain text is taken at a time. Let 'r' is the character.

	1	2	3	4	5	6	7	8	position
Binary representation of 'r'	0	1	1	1	0	0	1	0	bit value

Even position value-1100=12. 12 is the 6th even value in the even group (2,4,6,……..2^8).odd position value -0101=5 5 is the 3rd odd value in the odd group (1,3,7,…….. 2^8)

result1=n(ep)o(ep)n(ep) so that is 6*6=36 and result2= n(op)o(op) n(op) so that is 3+3=6. Now final value= result1 o (b) result2. So that is 36-6=30 (in case of negative result absolute value is taken)

Then 8 bit binary representation of 30 is done and we can also convert it into 16 bit or 24 bit depending upon the range of the value. Fig-5 represents the 8 bit representation of the derived secret value.

1bit	2bit	3bit	4bit	5bit	6bit	7 bit	8bit
0	0	0	1	1	1	1	0

----------8 bit----------

DV1----------DV8

Figure 5: 8 bit Secret Value derived from key.

D: XOR operation and Formation of Cipher Text file:-

Plain text is encrypted by the secret value. XOR operation is performed bitwise between the plain text and the secret value.[5] Fig-6 represents the encryption process

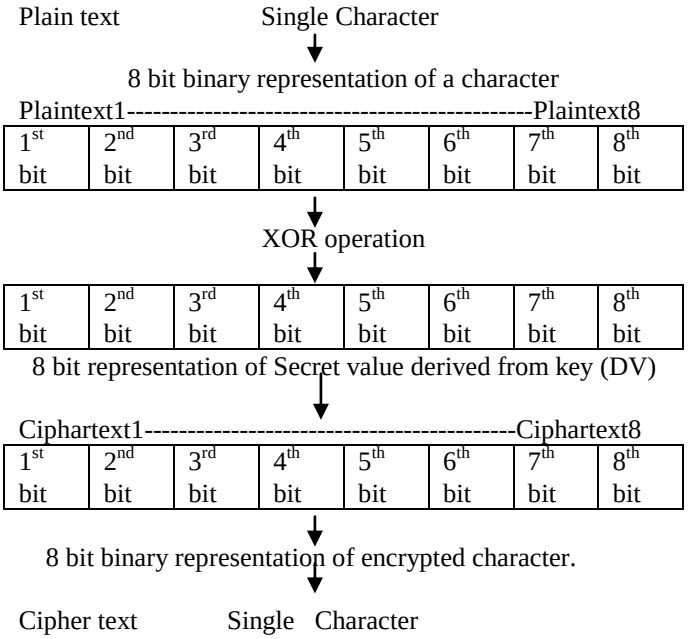

Figure 6: XOR operation between Plain Text and Secret Value.

XOR operation is performed between plain text block and the block of the secret value derived from the Private-key (DV). Then we get the final encrypted block. And from there we get the final encrypted single character. In this way all the characters of the inputted file are encrypted and those encrypted characters are stored into the cipher text file. The file is sent to the receiver with the secret private-key file. Fig-7 demonstrate the total XOR procedure between plain text and the secret value derived from the private-key where PT is the plain text, DV is the Secret Value derived from the private-key, and CT is the cipher text [5].

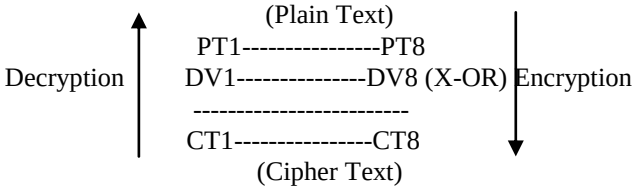

Figure 7: Block wise XOR operation between Plain Text and Secret Value derived from the Private-key.

Step D.1. Perform XOR operation between array PLAINTEXT and array DERIVEDVALUE and the final encrypted value is stored in array ENCRYPTED with dimension 8.

Step D.2. The binary value of the array ENCRYPTED is converted into corresponding ASCII value. Then the corresponding character is generated from the ASCII code and the character is stored into cipher text file.

Example- We read a character 'r' from the inputted file and generate the plain text in step-1 and secret value has been derived from the private-key in the Step-3. Now we perform the XOR operation between the plain text and the secret value derived from the private-key. Fig-8 represents the XOR procedure

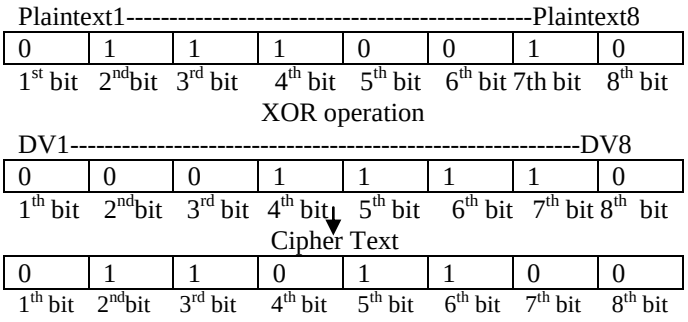

Figure 8: Generation of Cipher Text by XOR operation between Plain Text and Secret Value.

Corresponding ASCII value which is 108 is generated from this 8 bit binary string (Cipher Text).Then corresponding character 'l' is generated from this ASCII value. In this way all the characters of the plain text are encrypted and stored into the cipher text file. The encrypted file is sent to the receiver with the secret private-key file.

III. DECRYPTION PROCESS

E. Conversion of Cipher Text into Predefined Format: -

Read each of the encrypted character from the cipher text file and store it into 8 bit binary format into an array CIPHARTEXT with dimension 8.

F. Formation of Secret Value from Private-key for decryption:-

Derive the secret value from the private-key by using *step-C* and store the 8 bit binary value in the array DERIVEDVALUE with dimension 8.

G. XOR operation And formation of Decrypted Text file

Step-G.1. Here array CIPHERTEXT is used for decryption. Perform XOR operation between CIPHERTEXT and DERIVEDVALUE and the final decrypted value is stored in array DECRYPTED with dimension 8.

*Step-G.*2. The binary value of the array DECRYPTED is converted into corresponding ASCII value. The corresponding character is generated from the ASCII code and the character is stored into decrypted text file.

Example- We read a character 'l' from the cipher text file. The ASCII value of 'l' is 108. We convert 108 in 8 bit binary representations and store it into array CT with dimension 8. The secret value has been derived from the private-key in the

Step-6. Now we perform the XOR operation between the cipher text and the secret value derived from the private-key. Fig-9 represents the XOR procedure

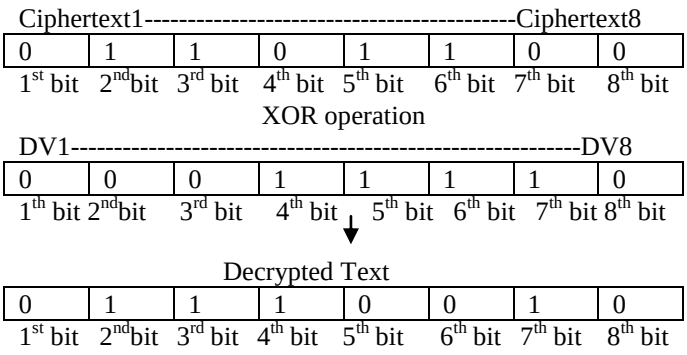

Figure 9: Generation of Plain Text by XOR operation between Cipher Text and Secret Value derived from the key.

ASCII value 114 is generated from the decrypted text and the character 'r' is generated corresponding to this ASCII value. In this way all the characters of the cipher text file are decrypted and stored into the decrypted text file.

IV. EXPERIMENTEL RESULT

Here we have displayed the content of the plaintext, the corresponding encrypted content of the cipher text and the corresponding content of the decrypted file. Table-1 demonstrates the entire content of the source files, encrypted file and the decrypted file.

Table I: Corresponding content of source, encrypted, decrypted file

Content of the source file (f.txt)	Content of the Encrypted file (f_en.txt)	Content of the Decrypted file (f_de.txt)
Rgthfdtghhjmnbv cxzasdfghjklpoiuy trewq1245667890 -;l;/.,'[[	phyzDLF¾?æ-/? *Ö?Ï²?$üÛÑ Õ ÔïþÎÛÜý¥ßÄ?·¿ Ñã-?³?ºíàóýùÞ?ë¸ §á??ÈëßÕ°	Rgthfdtghhjmnbv cxzasdfghjklpoiuy trewq1245667890 -;l;/.,'[[

We have used *, +, - as operator for even value, operator for odd value and base operator respectively. The encryption is done by using the secret derived value from secret private-key which is dedicated and distinct for each character of the plain text. The Private-key value is depend on the formation of 'o' and '1' of a single character in the plain text. There is N numbers of different or may be distinct private-key value if a plain text has N number of characters. The encryption or decryption has taken 10047 milliseconds.

We have executed our program on 15 number of files of different types (*.com,*.txt,*.exe,*.sys and *.dll). We have taken 3 numbers of files of each type. The Execution results are being displayed in the Table II, Table III, Table IV and Table V, Table VI [4][5].

Table II: - Execution Result for *.com files

Source File name	Source File size (Byte)	Encrypted File size (Byte)	Encryption/ decryption time(Mille seconds)
loadfix.com	1131	1131	120031
README.COM	4217	4217	390141
diskcomp.com	9216	9216	906187

Table III: - Execution Result for *.txt files

Source File name	Source File size (Byte)	Encrypted File size (Byte)	Encryption/ decryption time(Mille seconds)
ReadMe.txt	286	286	37734
LICENSE.TXT	4829	4829	120360
TechNote.txt	9232	9232	244281

Table IV: - Execution Result for *.exe files

Source File name	Source File size (Byte)	Encrypted File size(Byte)	Encryption/ decryption time(Mille seconds)
WINSTUB.EXE	578	578	82968
mqsvc.exe	4608	4608	113407
label.exe	9728	9728	209969

Table V: - Execution Result for *.sys files

Source File name	Source File size (Byte)	Encrypted File size (Byte)	Encryption/ decryption time(Mille seconds)
VIAPCI.SYS	2712	2712	83235
rootmdm.sys	5888	5888	148156
sffp_mmc.sys	10240	10240	222594

Table VI: - Execution Result for *.DLL files

Source File name	Source File size (Byte)	Encrypted File size (Byte)	Encryption / decryption time(Mille seconds)
iconlib.dll	2560	2560	74312
KBDAL.DLL	6656	6656	161329
panmap.dll	10240	10240	266375

Figure 10 graphically shows how encryption time changes with size of the file being encrypted. It clearly indicates that the time required for encryption or decryptions do not depend on the type of the file, but depend only on its size.

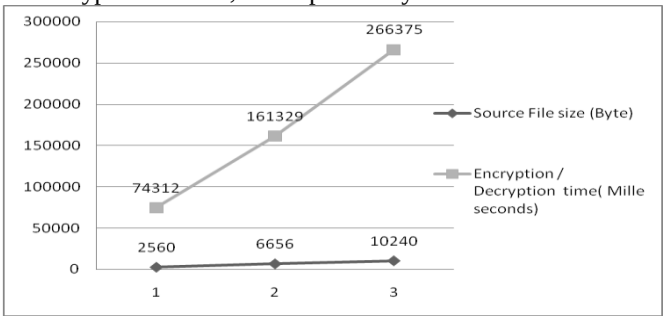

Figure 10: Relationship between Encryption time and source file size

The Pearsonian Chi-square test has been performed here to decide whether the sample of encrypted files may be supposed to have arisen from a specified population. The Pearsonian Chi-square is defined as follows: $X2 = \Sigma \{(f\ 0 - f\ e)2 / f\ e\}$ Here 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.[5]
If the Chi-square value is a higher one that means there is a higher frequency in source file and a lower frequency in encrypted file for a specific character. So, this proposed encryption procedure generate highly secured encrypted file. Table VII shows the Chi-square value for different types of file.

Table VII: Chi Square test Result on different files.

Source File name	Source File size (Byte)	Encrypted File size (Byte)	Chi-square Test Value
loadfix.com	1131	1131	31305486336.00
ReadMe.txt	286	286	276128064.00
WINSTUB. EXE	578	578	16863553536.00
VIAPCI.SYS	2712	2712	15576862720.00
iconlib.dll	2560	2560	18683279360.00

V. CONCLUSION

Here we proposed a new private-key encryption scheme based on bitwise XOR operation between the plain text and the secret value derived from the private-key.
The secret value is calculated depending upon the formation of '0' and '1' of a single character in the plain text and the different operators(base operator, operator for even value, operator for odd value) inputted by the user. There is a separate secret key value for each of the character in the plain text as the formation of '0' and'1' is different for each distinct character. Thus provide a great security.
The process of retrieving the secret value from the private-key is only being known by the receiver and the sender. So it is not possible for an unauthorised person to derive the secret value only with the presence of private-key. Thus the security is increased in a great entrance.
Besides this, a user can also do the encryption or decryption by using different types of operators in the private-keys allotted for specific numbers of user defined blocks in a plain text file. So the security will be increased as different keys are being used for encryption for different portion of plain text file,
The size of the encrypted or decrypted file is same as of the plain text file. So we don't need any additional memory for storing the encrypted or decrypted file.
The execution time is depends on the file size not on the type of the file as we have done the encryption in bit level.
So, the proposed scheme is better in respect of providing security for encryption, encrypted file size management, encryption or decryption time requirement.

REFERENCES

[1] J. K. Mandal, S. Dutta, "A 256-bit recursive pair parity encoder for encryption ", Advances D -2004, Vol. 9 nº1, Association for the Advancement of Modeling and Simulation Techniques in Enterprises (AMSE, France), www. AMSE-Modeling.org, pp. 1-14
[2] William Stallings, Cryptography and Network security: Principles and practice (Second Edition), Pearson Education Asia, Sixth Indian Reprint 2002.
[3] Atul Kahate (Manager, i-flex solution limited, Pune, India), Cryptography and Network security, Tata McGraw-Hill Publishing Company Limited.
[4] Mark Nelson, Jean-Loup Gailly, The Data Compression Book. BPB Publication
[5] Saurabh Dutta, "An Approach towards Development of Efficient Encryption Technique", A thesis submitted to the University of North Bengal for the Degree of Ph.D., 2004.

Ramkrishna Das, M.Tech (Computer Science of Engineering) MCA, BCA, Research experience 1 year,. Seminar Attend-2

Saurabh Dutta, PH. D. (COMPUTER SCIENCE), MCA, B. SC. (MATHEMATICS) **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- 12 years
Specialization: Information Security and Cryptology
Member-IACSIT, IJIST, IEDRC, , IJISMRD, 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