RSU Algorithm for Secured Communication

Sunil Shirsat
Research Scholar – Department Of Information Technology (M.E)
Smt. Kashibai Navale College of engineering, Pune-16, Maharashtra, India

Swapnil Sanap
Patent Analyst – Department Of Intellectual Property Rights
Legasis Services Pvt. Ltd, Pune-16, Maharashtra, India

ABSTRACT
The essential aspect of secured communications is secret key cryptography. The key selection mechanism and the encoding methodology determine the efficiency of the cipher text generated. In this paper, RSU algorithm is proposed which encodes the message by using reversible Boolean functions and combinatory. In this method, same keys are used for both encryption and decryption. The plaintext and secret keys are modified before encryption. This increases the complexity of deciphering the cipher text by intruders. Thereby it provides extremely better security for data.

Keywords:
Combinatorics; Cryptography; Decryption; Encryption; Reversible gates; RSU Algorithm; Secret key

1. INTRODUCTION
Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables to store sensitive information or transmit it across insecure networks so that data can only be accessed by the intended recipient. Data that can be read and understood without any special measures is called plain text. The method of distinguishing plain text in such a way as to hide its substance is called encryption. Encrypting plain text results in unreadable gibberish called cipher text. The process of reverting cipher text to its original plain text is called decryption.

A cryptographic algorithm [1, 3] works in combination with a key to encrypt the plain text. The same plain text encrypts to different cipher text with different keys. The security of encrypted data is entirely dependent on two things: the strength of cryptographic algorithm and the secrecy of the key. The strength of cryptographic algorithm is measured in time and resources it would require to recover the plain text.

In a classical computer, the logic gates other than the NOT gate are not reversible; for instance, in an AND gate, two input bits cannot be recovered uniquely rom the output bit. However, it is instructive to observe that reversible gates in classical computers are theoretically possible for input strings of any length; moreover, these are actually of practical interest, since they do not increase entropy. A reversible gate is a reversible function on n-bit data that returns n-bit data, where an n-bit datum is a string of bits x1, x2... xn of length n. The set of n - bit data is the space {0, 1} n, which consists of 2n strings of 0's and 1's. In this paper, a novel RSU algorithm is proposed, which uses reversible gates and Combinatorics for implementation. The entire algorithm is simulated using MATLAB.

2. BASIC REVERSIBLE GATES
Given a function ‘f ‘is reversible [4, 5] if and only if there exists a function ‘g’ such that x = g(f(x)) for all x in the domain of ‘f’. The corresponding function „g’ is usually referred to as f−1. Given n Boolean inputs, any multiple output Boolean function on such n Boolean inputs must have exactly ’n’ Boolean outputs so that it is reversible [7]. In n X n reversible gate there are ‘n’ Boolean inputs and n Boolean outputs.

2.1. Optimization issues
• The number of outputs of a reversible logic gate should be equal to the number of inputs.
• The output of the gate that is not used as a primary output or as input to other gate is called garbage outputs.
A heavy price is paid for every garbage outputs.
• The number of constant input to the gate should be as minimum as possible.
• In reversible logic, fan-out of more than one is not allowed; every output can be used only once [8].

2.1.1. Reversible AND gate:
Reversible AND gate and its corresponding functionalities are shown in fig 1. Rev_and acts as a 1-bit reversible AND gate if a constant input 0 is applied to input A. Here P is the required AND functionality, Q and R are the garbage bits. This is indicated in fig 2.

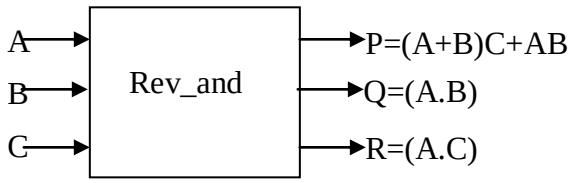


Fig 1. Rev_and gate

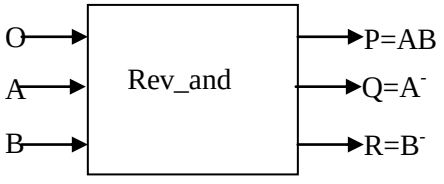


Fig 2. Rev_and as ANDgate

2.1.2. Reversible OR gate:

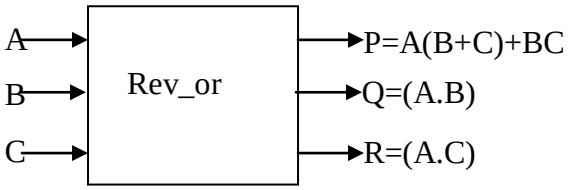


Fig 1. Rev_or gate

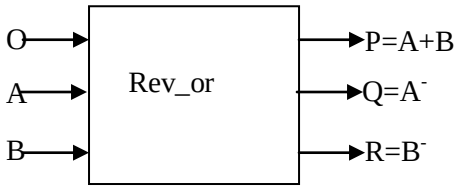


Fig 2. Rev_or as ORgate

Reversible OR gate and its corresponding functionalities are shown in fig 3. Rev_or acts as a 1-bit reversible OR gate if constant input 0 is applied to input A. Here P is the required OR functionality, Q and R are the garbage bits. This is indicated in fig 4. In these gates, length of the critical path can be varied. In this design, the length of the critical path in both the cases is maintained as three.

3. COMBINATORICS

2	7	6
9	5	1
4	3	8

Figure 5. Magic Square of order 3

Combinatorics is a branch of mathematics concerning the study of finite or countable discrete structures. Magic square [5] belongs to the class of algebraic Combinatorics. A magic square of order n is an arrangement of n2 numbers, usually distinct integers in a square, such that there are n numbers in all rows and all columns. Each row, each column and both diagonals sum to the same constant. A normal magic square contains the integers from 1 to n2. Magic squares exist for all orders n _ 1 except n = 2. The smallest nontrivial case of order 3 is shown in fig. 5.

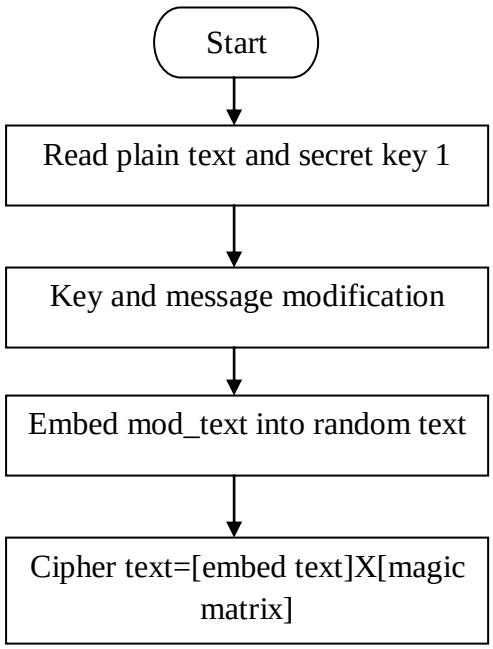
The constant sum in every row, column and diagonal is called the magic constant or magic sum *S*. The magic constant of a normal magic square depends only on order *n* and has the value given in equation 1.

For normal magic squares of order n = 3, 4, 5... the magic constants are: 15, 34, 65, 111, 175, 260 …The number of different $n \times n$ magic squares for n from 1 to 5, not counting rotations and reflections is 1, 0, 1, 880 and 275305224. The number for $n = 6$ has been estimated to 1.7745×1019. As the value of n increases, number of magic squares that can be constructed in the respective order of n increases enormously and is unpredictable.

4. PROPOSED RSU ALGORITHM

4.1. Encryption:

The brief overview of the proposed encryption algorithm is depicted in fig.6.

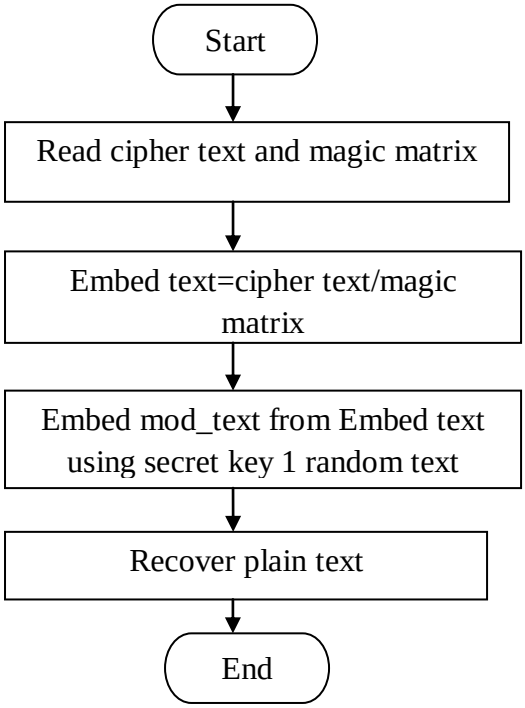


4.1.1. Steps followed in encryption algorithm:

• Read the plain_text from input file and store its ASCII code in plain_text.
• Consider the secret key1 and find its length=len.
• Convert secret key to ASCII code and store it in secret_key1
• Find the sum of ASCII equivalent of each character in secret key1=SUM
• Find the number of ones in each character of secret_key1=ONE, add the position of number of ones in the same character = sum_1. Find sum_1 mod ONE = R. If secret_key1 [i] is even, then secret_key1 [i] is rotated right by R bits, else secret_key1 [i] is rotated left by R bits to obtain mod_key.
• Evaluate SUM mod len=Y
• ASCII equivalent of first Y characters of the plain text is added with Y and next (8-Y) characters are subtracted with Y.
• ASCII equivalent of succeeding Y-1, Y-2 ……characters of plain_text is added with Y and (8-(Y-1)), (8-(Y-2))……characters are subtracted with Y to obtain mod_text.
• Perform reversible AND operation between mod_text and mod_key for every Yth character.
• Perform reversible OR operation between mod_text and mod_key for all the remaining characters to obtain the rev_mod_ text.
• Generate random text file.
• In mod_key, find the position of ones from the MSB side and replace those positions in random text file by rev_ mod_text to obtain embedded text.
• A magic square of order len is generated, which is considered as secret key2. This matrix is reshaped and extended till the length of embedded text. • Perform multiplication between individual elements of embedded text and extended magic square to obtain the cipher text.

4.2. Decryption

The brief overview of the proposed decryption algorithm is depicted in fig.7.



4.2.1 Steps followed in decryption algorithm:

• Get the secret key2 and extend it until the length of cipher text.
• Divide the cipher text by secret key2(magic matrix)
• Get the secret key1, find its length=len and sum of ASCII equivalent of key= SUM.
• Find the number of ones in each character of secret_key1=ONE, add the position of number of ones in the same character=sum_1 then take sum_1 mod ONE=R If secret_key1 [i] is even, secret_key1 [i] are rotated right by R bits, else secret_key1 [i] are rotated left by R bits to obtain mod_key.
• In mod_key find the position of ones from MSB side and extract the characters in those positions.
• Evaluate SUM mod len=Y.
• Perform reversible AND operation of the character present in rev_mod_text at the position which is multiple of Y with mod_key.
• Remaining characters are ORed with mod_key and convert the result to decimal form.
• ASCII equivalent of first Y characters of the plain text is subtracted with Y and next (8-Y) characters are added with Y.
• ASCII equivalent of succeeding Y-1, Y-2 ……characters of plain_text is subtracted with Y and (8-(Y-1)), (8-(Y-2))……characters are added with Y.
• The result is original plain_text.

4.3. Security

• It is very difficult to uniquely identify the secret key2. It is recommended to use secret key2 whose order is more than 6, since the number of different n × n magic squares generated for n>6 is unpredictable.
• In encoding, reversible OR gates are used. In general, reversible OR gates have two garbage bits for ach input. Here the number of garbage bits is reduced to one. Even in the bit stream, its difficult to differentiate a garbage bit from the output bit.
• In reversible gates, number of critical path is variable; hence security is increased.
• Obtained binary data is embedded with random text. Due to this, complexity is increased in predicting length of plain text or secret key.
• Before encoding, both plain text and secret key are modified by exchanging bits depending on secret key1. Secret keys are known only to transmitter and receiver; hence it‟ s difficult to identify the position of bits that got exchanged.
• Resultant cipher text is again modified by exchanging the bits in different position. The position value is again depending on secret key1. This multi level modification in bits will increase the security.

5. EXPERIMENTAL RESULTS

Consider the secret key1 as ‘sbm’ and plain text as „’quantum’. Length of secret key 1 is 3 i.e. len=3 and its SUM is 322. In secret key1 the number of ones in the first character„’s ‘is 5. Therefore ONE = 5 and sum_1 = 24. R = mod (24, 5) = 4. ASCII value of ’s ‘[115] is an odd number whose binary equivalent is 01110011. Rotate 115 by four bits left so that the key value is modified to 00110111. This is continued till the last character of secret key1 to obtain the mod_ key. Y= mod (322, 6) = 4. mod_text is obtained by, adding four to the first four ASCII values (‘quan’) of the plain text and subtracting four from remaining three characters (‘tum’). The obtained mod_text is “rt`mstl”.
Perform reversible AND operation between every fourth position of the mod_text with mod_key; and reversible OR operation is performed for the remaining characters. Assume the random text as “rr sP#Exy zx=e 5sdxR kuU_]2R Y # ”. Depending on the position of the number of ones in the mod_key, respective position of the random text is replaced by characters of the mod_text to obtain the embedded text. Embedding procedure is illustrated for mod_key of character„’s’ in fig 8.

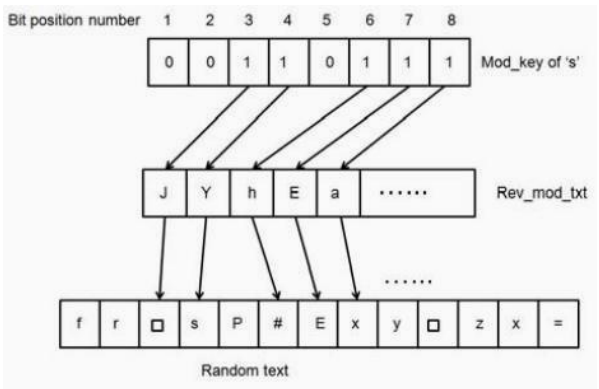


Fig 8. Embedding procedure

Now consider secret key 2. For illustration, magic matrix M of order 3 X 3 is used which is shown below.
Append and reshape the matrix to the size of embedded text.

$$M == \begin{bmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{bmatrix}$$

Perform matrix multiplication between embedded text and modified magic matrix to get the cipher text. The decoding algorithm decodes the cipher text to obtain original plain text as indicated in the fig. 9

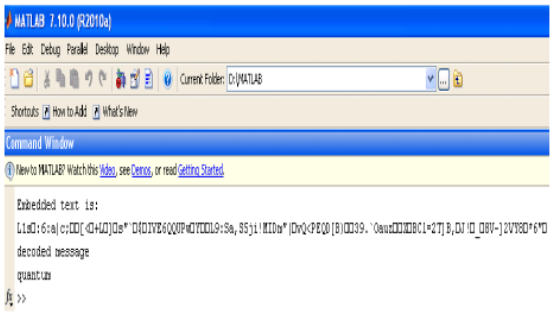


Fig.9 Decryption Result

6. CONCLUSION

The proposed RSU algorithm can be used to maintain the security of data. The algorithm is highly secured since the determinations of secret keys are very difficult. The hidden data cannot be accessed easily because of multiple level of encryption which provides greater level of security to the data.

7. REFERENCES

[1] Andrew S. Tanenbaum , "Computer Networks" (4th Edition), Prentice Hall PTR

[2] BANGALORE G.Tilak, JAIN Vinod Kumar, HEBBAR K.Shreedhar ,"Enciphered Data Steganography Using Secret Key" JEEE, University of Oradea Publisher, Volume 3, Number 2, October 2010

[3] Behrouz A.Forouzan, "Cryptography and Network Security", Special Indian Edition 2007, Tata McGraw-Hill Publication

[4] C.H.Bennett,1973. Logical reversibility of computation, IBMJ. Research and evelopment,17: 525532. [5] C.H.Bennett."The thermodynamics of computation – A Review",International journal of Theoretical Physics, 21:905-928,1982.

[6] http://en.wikipedia.org/wiki/Magic_square

[7] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information. Cambridge, U.K.: Cambridge Univ. Press, Dec.2000.

[8] Robert Wille, Rolf Drechsler "Towards a Design Flow for Reversible Logic", (9-13). Shreedhar H. K. et al. / International Journal of Engineering Science and Technology (IJEST)