Volume 2, No. 1, Jan-Feb 2011

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

TECHNICAL NOTE

Available Online at www.ijarcs.info

An RTL Based Design & Implementation of Block Cipher Through Time-Stamp-Keyed-Oriented Encryption Technique (TSK-OET)

Rajdeep Chakraborty*
Lecturer,
Dept. of Computer Science & Engineering,
Netaji Subhash Engineering College, Garia, Kolkata-700152, West Bengal, India
rajdeep_chak@yahoo.co.in

J.K. Mandal
Proffesor,
Dept. of Computer Science & Engineering,
University of Kalyani, Kalyani, Nodia,
West Bengal, India
jkm.cse@gmail.com

Abstract: This paper is an RTL based design and implementation of block cipher through Time-Stamp-Keyed-Oriented Encryption Technique (TSK-OET); it has been done where the plain text is divided into blocks, and substitutions/permutations are performed in each of the blocks during encryption. This technique has been successfully implemented in VHDL & C. The decryption is done in the similar manner, as the technique is symmetric. The block sizes are not restricted to 2^N, where N= 1,2,3, …… positive integers. Various tests & comparisons have been done with industrially accepted RSA, comparable result has been found after a number of TSK-OET iterations and a large number of variable sizes of blocks.

Keywords: VHDL, FPGA, RTL, Block Cipher, Session & Private Key, Cryptography, AD

I. INTRODUCTION

In this world of computer networks [2,7], it is important to make information [2,3,7] secure by protecting data and resources from malicious acts by program/software or being abused by intruders [2,3]. Cryptography [1,2,3,7,13,14] is one of the ways to protect data, which involves the study of mathematical techniques that allow the practitioner to achieve or provide the following objectives or services [1,2,3,5,7,13,14]:

A. *Confidentiality*

Service that keeps the data involved in an electronic transaction private.

B. *Data Integrity*

Service that requires that computer system assets and transmitted information be capable of modification only by authorized users.

C. *Authentication*

Service that is concerned with assuring that the origin of a message is correctly identified.

D. *Non-Repudiation*

This simply tells that the actions performed by the service user in an electronic transaction are non revocable so that they are legally binding.

E. *Access Control*

Requires that access to information resources may be control by or for the target system.

F. *Availability:*

Requires that the computer system assets be available to authorized parties are needed.

The technique proposed in this paper in very novel and simple. It has been implemented both in C & VHDL [8,10,11,12].

Moreover, this technique is a block cipher [2,3,4,5] based algorithm and a competent with comparable result has been found against RSA [2,3,4,5]. Therefore this, technique fully satisfies the primary goal of confidentiality. Moreover, in application-based scientific research on information security, role of ciphering technique with proven efficiency & security is inevitable; this paper presents one such newly developed technique.

Section II describes the technique, Section III deals with session key generation, Section IV illustrates the results, Section V draws the conclusion and future scope of the work, Section VI is the author's acknowledgement and finally in Section VII, the references are listed.

II. THE PROPOSED TECHNIQUE, TSK-OET

In this technique the block size [2,3,4,5] of different bits are considered and the swapping is done respectively. The orientation is done like 1st bit to nth bit. Then 2nd bit to (n-1) th bit and so on. A part of the session key [1,2,3,5,14] is generated from system time and is called time stamp based key, which is then XORED [1,7,14] with the resultant block and finally replacing the original block, generating the cipher text.

Suppose our source stream is S= 10110. So after orientation we get the target sub stream as S' = 01101. So, here the 1st bit is oriented with the 5th bit. Then, Then 2nd bit is oriented with the 4th bit and so on. Now a time stamp [3,5] of value 23 Hours (11.00 PM) is then EXORED to S' yielding S''= 11010, this is the final cipher text to be transmitted over unsecured channel. The decryption [1,2,14] is done in the similar manner. Moreover, to enhance the security further, a random orientation of bits is also proposed.

The Table I illustrates another example of Time-Stamp-Keyed-Oriented Encryption Technique (TSK-OET). Here 9-bit block size is used and the good thing in this technique is to use variable length block sizes, so the block sizes are not restricted to 2^N where N= {set of integer}.

Table I: Encryption & Decryption process of TSK-OET

<table>
<tr><td rowspan="2">Encryption</td><td colspan="3">Source Stream
101011001</td><td colspan="3">First Phase of TSK-OET Encryption
100110101</td><td colspan="3">Encrypted Stream
TSK=23Hrs08Min
(101111000)</td></tr>
<tr><td>101</td><td>011</td><td>001</td><td>100</td><td>110</td><td>101</td><td colspan="3">001001101</td></tr>
<tr><td rowspan="2">Decryption</td><td colspan="3">Receive Cipher Text
TSK=23Hrs08 Min
(101111000)</td><td colspan="3">First Phase of TSK-OET Decryption
100110101</td><td colspan="3">Target Stream
101011001</td></tr>
<tr><td colspan="3">001001101</td><td>100</td><td>110</td><td>101</td><td>101</td><td>011</td><td>001</td></tr>
</table>

III. THE IMPLEMENTATION OF TSK-OET AND KEY GENERATION

At first, this technique has been implemented in C programming language and then tested. A comparable result has been found. Then it has been implemented in VHDL for RTL design [8,9,10] to be embedded in the FPGA [8,9,10] based systems. A good synthesis and simulation been generated in XilinX software [8,9,10].

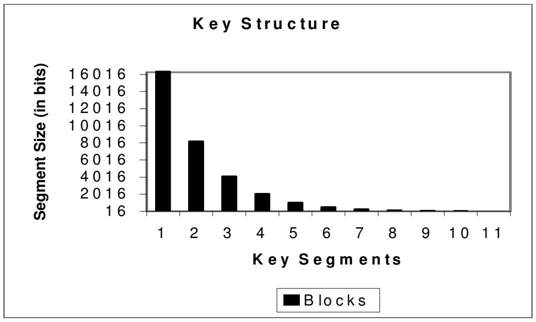

Figure I: 110-bit Key Format with 11 segments

In the proposed technique, for ensuring the successful encryption [1,2,14] and decryption with varying size of blocks, a 110 – bit key format consisting of 11 different segments has been proposed in this section. Moreover a time stamp key of size 56-bit is also generated from the system time. The concept of time stamp is that a unit is considered of 10^{-6} seconds. So, the time stamp is equal to the system time calculated in this unit since AD.

Therefore the total key length is 110 + 56 = 166-bit. Although, the system time is proposed here but to ensure the randomness of the key, user can use any time-stamp since AD. Now we will be considering the 110-bit key generation in details.

For the segment of the rank R [6,7,15], there can exist a maximum of N = $2^{15–R}$ blocks, each of the unique size of S = 2^{15-R} bits, R starting from 1 and moving till 11.

For different values of R, the following segments are generated:

→ Segment with R = 1 formed with the first maximum 16384 blocks, each of size 16384 bits.
→ Segment with R = 2 formed with the first maximum 8192 blocks, each of size 8192 bits.
→ Segment with R = 3 formed with the first maximum 4096 blocks, each of size 4096 bits.
→ Segment with R = 4 formed with the first maximum 2048 blocks, each of size 2048 bits.
→ Segment with R = 5 formed with the first maximum 1024 blocks, each of size 1024 bits.
→ Segment with R = 6 formed with the first maximum 512 blocks, each of size 512 bits.
→ Segment with R = 7 formed with the first maximum 256 blocks, each of size 256 bits.
→ Segment with R = 8 formed with the first maximum 128 blocks, each of size 128 bits.
→ Segment with R = 9 formed with the first maximum 64 blocks, each of size 64 bits.
→ Segment with R = 10 formed with the first maximum 32 blocks, each of size 32 bits.
→ Segment with R = 11 formed with the first maximum 16 blocks, each of size 16 bits.

With such a structure, the key space becomes of 110 – bits long and a file of maximum size around 44.74 MB can be encrypted using proposed technique. Figure I represents this structure.

Table II: Chi-Square Values

Source File	File Size (in Bytes)	Chi Square Value		Degree of Freedom	
		TSK-OET	RSA	TSK-OET	RSA
In01.TXT	16,530	191382	30148	255	64
In02.DOC	23,425	253470	185351	255	66
In03.TXT	33,221	410735	169424	255	73
In04.TXT	45,825	505121	334371	255	77
In05.TXT	48,410	638592	396128	255	75
In06.EXE	70,985	896405	761842	255	76
In07.EXE	126,465	1203665	1053183	255	88
In08.EXE	142,125	1692655	545752	255	73
In09.DLL	190,175	4250652	307565	255	10
In10.DLL	410,912	3922143	327510	255	11

This key generation process concatenated with the time-stamp-key will get a key size of 166 bits.

IV. THE RESULT AND SIMULATION

This section gives the results found on various parameters. The main results that are described here, frequency distribution graph [6,7,15], chi-square [6,7,15] test for non-homogeneity time complexity analysis, the avalanche ratio [2,3,5] and RTL [8,9,10] based results. These are all described in respective sub sections.

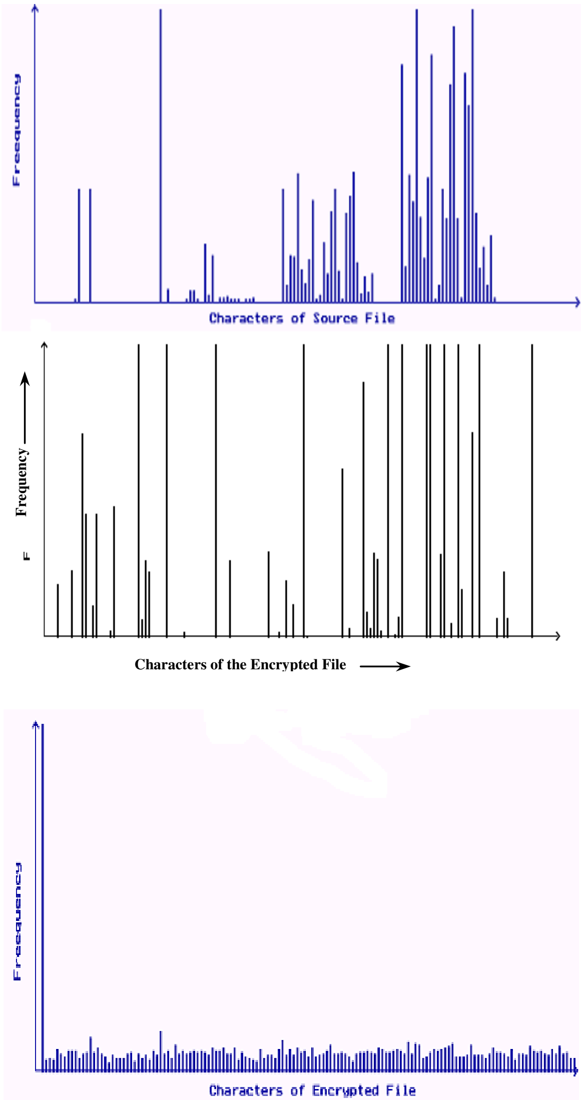

Figure II: The Frequency Distribution graph of Source, RSA encrypted & TSK-OET encrypted Files

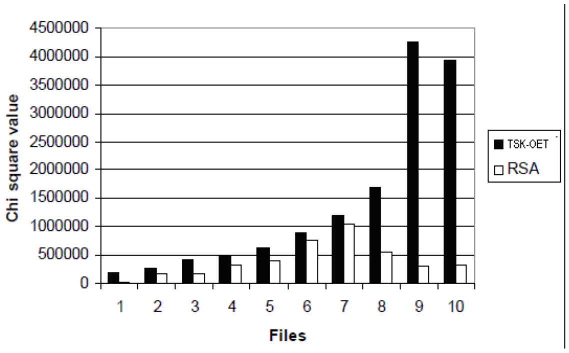

Figure III: Graphical representation of Chi-Square values

A. *The Frequency Distribution Graph*

The Figure II: gives the frequency distribution graphs of source file, RSA encrypted file and TSK-OET encrypted file. This result illustrates the frequency of the proposed technique is well distributed than that of RSA, this infers the statistical cryptanalysis [2,3,4,5] is quite difficult.

B. *The Chi-Square Test*

Table II gives the Chi Square values of the proposed technique (TSK-OET) and that of RSA. Figure III illustrates the same graphically. So observing the above table & figure it has been seen that chi-square values of the proposed technique is quite higher than RSA and also the degree of freedom [1,6,14,15] comes to be at a value of 255 in TSK-OET which says a well distribution of characters present in the TSK-OET encrypted files than that of source file. Therefore the chi-square values are at par with that of frequency distribution graph result illustrated in section A.

Table III: The Time Complexity Analysis

Source File	File Size (in bytes)	Encryption time (in Seconds)		Decryption time (in seconds)	
		TSK-OET	RSA	TSK-OET	RSA
In01.TXT	16,530	0.02	0.01	0.10	0.28
In02.DOC	23,425	0.00	0.03	0.00	0.30
In03.TXT	33,221	0.10	0.21	0.10	1.67
In04.TXT	45,825	0.20	0.35	0.11	3.51
In05.TXT	48,410	0.25	0.40	0.20	5.06
In06.EXE	70,985	0.35	0.39	0.35	4.34
In07.EXE	126,465	0.40	0.65	0.40	8.37
In08.EXE	142,125	0.50	0.61	0.42	6.59
In09.DLL	190,175	0.52	0.75	0.50	10.15
In10.DLL	410,912	0.60	0.95	0.55	11.70

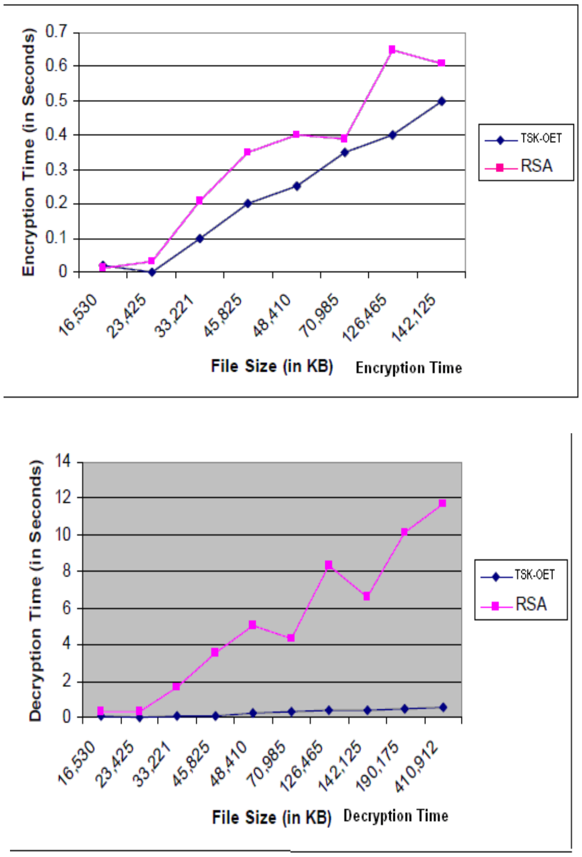

Figure IV: Encryption Time & Decryption Time

Table IV: The Avalanche Ratio

Source File Name	**File Size in Bytes**	**Avalanche Ratio (in Percentage)**	
		RSA	OET
In01.TXT	16,530	58.0	77.7
In02.DOC	23,425	60.0	80.0
In03.TXT	33,221	75.0	88.8
In04.TXT	45,825	78.9	89.0
In05.TXT	48,410	80.9	87.0
In06.EXE	70,985	58.0	77.0
In07.EXE	126,465	58.9	76.0
In08.EXE	142,125	67.0	77.0
In09.DLL	190,175	67.9	82.9
In10.DLL	410,912	68.0	88.5

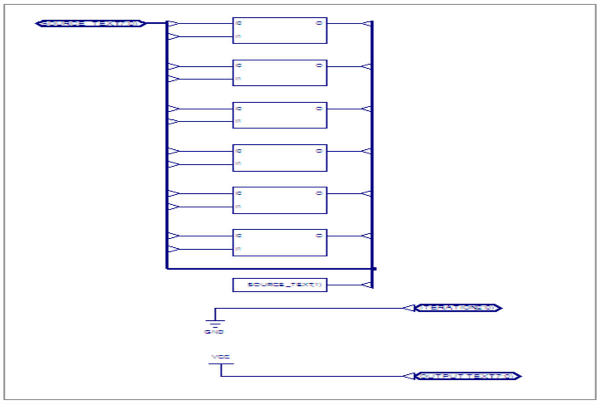

Figure V: RTL Diagram of the Proposed Technique

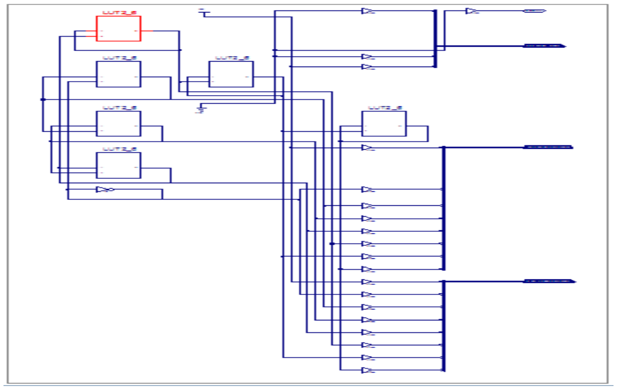

Figure VI: RTL Diagram for Spartan 3E

C. *The Time Complexity Analysis*

The main purpose of this paper is to develop an efficient and fast RTL design so; time complexity analysis is one of the major factors in developing the technique. The Table III shows the encryption time and decryption time of the proposed technique and that of RSA. Figure IV represents the same graphically. It is clearly seen that the time complexity of the proposed technique is quite less than that of RSA.

D. *The Avalanche Ratio*

The Avalanche is the ratio of difference between the simple encrypted file & one bit/one byte modified source/key file. The avalanche ratio is the degree of measure for cryptanalysis. More the ratio more difficult to analyses for known pain text – known cipher text pair. The Table IV clearly illustrates the result of avalanche ratio, which is found a satisfactory result for the proposed technique, TSK-OET.

E. *The RTL Simulation Based Results*

In this section gives some of the results found after implementing the proposed technique in VHDL. This code has been simulated & synthesized in Xilinx. The main objective is to find an efficient FPGA-based cryptographic technique for implementation in embedded systems [8,9,10]. The Figure V gives the RTL schematic of the proposed technique and the Figure VI gives the chip diagram for Spartan 3E [8,9,10]. If we closely observe the Figure VI, we can see that four Look-Up-Tables (LUTs) [8,9] are used here. So this implantation is synthesis able and can be burn into the Spartan 3E FPGA-chip. After synthesis of the design, the design translation, design mapping, placement of I/Os and routing has also been done successfully. The conclusion and the scope for the future work have been described in the next section.

V. THE CONCLUSION

The technique given here is easily implemented in high-level language and in VHDL. This technique is very easy and it's implemented in FPGA-based systems, the goal of fast execution and strong cryptanalysis requirements are also obtained here. Moreover this technique can be fabricated in chip to be used in embedded systems. The main goal of the author(s) is to develop an efficient FPGA-based crypto hardware and this paper is the first step towards this.

VI. ACKNOWLEDGEMENT

The authors express their deep sense of gratitude to The Department of Computer Science and Engineering, Netaji Subhash Engineering College, Garia, Kolkata, West Bengal, India and The Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nodia, West Bengal, India.

VII. REFERENCES

[1] Rajdeep Chakraborty, Dr. J.K.Mandal, "A Microprocessor-based Block Cipher through Rotational Addition Technique (RAT)", ICIT – 2006 18-21 December, 2006, Bhubaneswar, INDIA.
[2] W. Stallings, Cryptography and Network Security: Principles and Practices, Prentice Hall, Upper Saddle River, New Jersey, USA, Third Edition, 2003.
[3] B. Schneier. Applied Cryptography. John Wiley & Sons Inc., New York, New York, USA, 2nd edition,1996.
[4] U.S. Department of Commerce/National Institute of Standard and Technology. FIPS PUB 197, Specification for the Advanced Encryption Standard (AES), November 2001. Available at http://csrc.nist.gov/encryption/aes
[5] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Handbook of Applied Cryptography. CRC Press, Boca Raton, Florida, USA, 1997.
[6] A.M. Goon, M.K. Gupta, B. Dasgupta, Fundamentals of Statistics, Vol. 1, The World press Ltd.
[7] Dictionary of Computers and Information Technology Terms, 1^{st} edition, low point, Kolkata, India.

[8] FPGA- Based System Design by W. Wolf, Pearson Education.

[9] Embedded Core Design with FPGA's by Z. Navavi, TMGH.

[10] AVHDL: Premier by J. Bhasker, Pearson Education

[11] Programming in C by Balaguruswamy, India.

[12] Pointers in C by Y Kanitkar, India

[13] The software cryptographic tools for educational purpose available at http://www.cryptool.com/

[14] S. Mal, J.K. Mandal, S. Dutta, "A Microprocessor Based Encoder for Secured Transmission", Proceedings of the National Conference on Intelligent Computing on VLSI, Kalyani Govt. Engg. College, 16-17 February, 2001, pp 164-169.

[15] Number theory home page for secured key generations http://www.numbertheory.org/ntw/web.html