REVIEW OF COMPUTER ENGINEERING STUDIES

ISSN: 2369-0755 (Print), 2369-0763 (Online)
Vol. 4, No. 2, June 2017, pp. 57-61
DOI: 10.18280/rces.040203

A publication of IIETA

International Information and Engineering Technology Association
Advancing the World of Information and Engineering

http://www.iieta.org/Journals/RCES

Symmetric key cryptography using digital circuit based on one right shift

Joyita Goswami (Ghosh)*, Manas Paul

Institute of Leadership Entrepreneurship & Development, Kolkata 700010, India

Email: joyitagoswami@gmail.com

ABSTRACT

A session based symmetric key cryptographic technique has been proposed in this paper and it is termed as 1RS. The plain text is considered as a stream of bits and is chopped into variable length blocks. Bit positions into the block are right shifted to generate the cipher text. Right shift means the bit is right shifted by one position. The data bit is set or reset depending on the previous bit. The MSB is initialized by reset condition. A session key is generated randomly from the chopping information of plain text. Results are generated using twenty files of twelve different types with varying file sizes. Analyzing the results with respect to different parameters, the proposed technique 1RS is compared with existing and industrially accepted symmetric key techniques Triple-DES (168bits) and AES (128 bits).

Keywords: 1RS, AES, Triple DES, Session Key, Chi-square.

1. INTRODUCTION

In modern era, every computer is connected virtually. It is very important to secure our information from eavesdroppers. So, data security gets priority in modern life. Cryptography is an important feature for secure communication to protect important data. As a result, continuous research works are going on in this field of cryptography to enhance the network security.

Section 2 of this paper explains the proposed technique. Section 3 deals with the algorithms for encryption, decryption and session key generation. Section 4 explains the proposed technique with an example. Section 5 shows the results and analysis on different files and the comparison of the proposed technique with TDES and AES. Conclusions are drawn in section 6.

2. TECHNIQUE

1RS considers the input file as a finite number of binary bits. The binary bits are split dynamically into blocks of length 8n where n ε N, N is the set of natural numbers. The block sizes are written into file to generate session based key. The ith position bit of the original block is mapped to the jth position of encrypted block. In nature, this mapping is bijective. The bit position value (say value i) of the original block having block length 2k, varies from 0 to (2k -1), is converted into k-bit binary number and the corresponding binary bits are sent into k-input digital circuit. For each 2k number of combination of inputs, the output of the circuit produces unique 2k number of k-bit binary numbers. Figures 1 and 2 show the block diagram of circuit for encryption and decryption respectively. For encryption, the input bits are identified by IE1, IE2, IE3, …, IEk (where IE1 is the MSB and IEk is the LSB) and output bits are identified by OE1, OE2, OE3, …, OEk (where OE1 is the MSB and OEk is the LSB). The output bits of the circuit for encryption are defined as

$$OE_j = IE_{(j-1)} \text{ for } j=2, 3, \ldots, k$$
$$= IE_k \text{ for } j=1$$

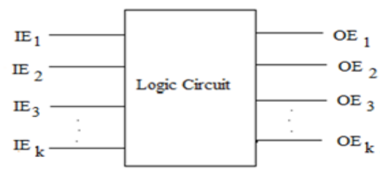

Figure 1. The block diagram of circuit for encryption

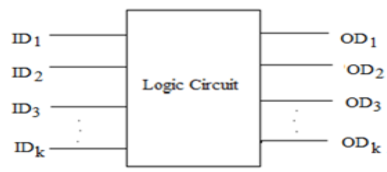

Figure 2. The block diagram of circuit for decryption

The bits are converted to the corresponding decimal to find the value of j. For decryption, the input bits are identified by ID1, ID2, ID3, …, IDk (where ID1 is the MSB and IDk is the LSB) and output bits are identified by OD1OD2OD3…ODk (where OD1 is the MSB and ODk is the LSB). The output bits for decryption are represented as

$OD_j = ID_{(j+1)}$ for j=1, 2, …, (k-1)
$= ID_1$ for j=k

3. ALGORITHMS

In this section Encryption, Decryption and Session key generation algorithms are explained in details.

3.1 Encryption algorithm

Step 1: The plain text i.e input file is considered as a finite number of binary bits.

Step 2: The bits are chopped dynamically into blocks of different lengths like 8 / 16 / 24 / 32 / 40 / 48 / 56 /….[i.e. 8n for n=1,2,3,4…] as follows
First n1 no. of bits is considered as x1 no. of blocks with block length y1 where n1 = x1 * y1. Next n2 no. of bits is considered as x2 no. of blocks with block length y2 where n2 = x2 * y2 and so on. Finally nm no. of bits is considered as xm no. of blocks with block length ym (= 8) where nm = xm * ym. So no padding is required.

Step 3: The bit position value (say value i) of the plain text block having block length 2k, varies from 0 to (2k - 1), is converted into k-bit binary number and the corresponding binary bits are sent into k-input digital circuit as IE1IE2IE3…IEk (where IE1 is the MSB and IEk is the LSB).

Step 4: The generated output bits of the digital circuit OE1OE2OE3…OEk are expressed as
$OE_j = IE_{(j-1)}$ for j=2, 3, 4, …,k
$= IE_k$ for j=1

Step 5: The output bits of the circuit are converted into decimal number to get the corresponding bit position value (say value j) in the encrypted block of length 8n.

Step 6: The ith bit of the plain text block is placed to jth bit of the encrypted block. The relationship between i and j for the block with block length 2k can also be expressed using the function given below
$j = f(i) = \{ i + (2k - 1)*(i \% 2) \} / 2$
Where / gives the integer part of the quotient and % gives the remainder part.
The cipher text is formed by converting the encrypted block to its corresponding characters.

3.2 Decryption algorithm

Step 1: The cipher text is considered as a finite number of binary bits.

Step 2: Processing the session key the binary bits are sliced into manageable sized block.

Step 3: The bit position value (say value i) of the cipher text block having block length 2k is converted into k-bit binary number and the corresponding binary bits are sent into k-input digital circuit as ID1ID2ID3…IDk (where ID1 is the MSB and IDk is the LSB).

Step 4: The generated output bits of the digital circuit OD1OD2OD3…ODk are expressed as

$OD_j = ID_{(j+1)}$ for j=1, 2, 3, …, (k-1)
$= ID_1$ for j=k

Step 5: The output bits of the circuit are converted into decimal number to get the corresponding bit position value (say value j) in the decrypted block of length 8n.

Step 6: The ith bit of the cipher text block is placed to jth bit of the decrypted block. The relationship between i and j for the block with block length 2k can also be expressed using the function given below
$j = f(i) = 2*i + (1 - 2k)(2*i / 2k)$
where / gives the integer part of the quotient
The plain text is regenerated by converting the decrypted block to its corresponding characters.

3.3 Session key generation algorithm

1RS generates a session based key for one time use in a particular session. The input bit stream is divided into 16 portions where 1st portion contains 20% of the total file size, 2nd portion contains 20% of the remaining file size and so on. Each portion is divided into x no. of blocks with block length y (=8n) where value of n is selected dynamically for first fifteen portions. Finally, last (i.e. 16th) portion is divided into x16 no. of blocks with block length 8 bits (i.e. y16 = 8). So, no padding is required. Total length of the input binary stream is = x1*y1+x2*y2+…….. +x16*y16. The value of n for each portion is stored as a character in the key file. So the key file contains sixteen characters.

4. EXAMPLE

Let consider the word "Ma". The 8 bit representation of the above characters "M" and "a" are '01001101' and '01100001' respectively. The bits are taken from MSB to LSB as 8 bit or 16 bit block length randomly. Now the position of 8 or 16 bit is converted into binary and following the above logic bits are changed to generate the new position. Figure 2 shows the encryption steps for the above example.

Case I: If block length is 8 then the encrypted string is '0010101101001001'. Two 8 bit binary numbers are '00101011' (=$[43]_{10}$) and '01001001' (=$[73]_{10}$) is encrypted from binary string and the corresponding characters are "+" and "I" respectively. So "Ma" is converted into "+I".

Case II: If block length is 16 then the encrypted string is '0010010010111001'. Two 8 bit binary numbers are '00100100' (=$[36]_{10}$) and '10111001' (=$[185]_{10}$) is encrypted from binary string and the corresponding characters are "$" and "¹" respectively. So "Ma" is converted into "$¹".

5. RESULTS

Results are generated using twenty files with different file sizes varying from 49 bytes to 134 MB (approx.) and eleven different file types (like .txt, .dll, .docx, .zip etc). Comprehensive analysis and comparison has been made between the proposed technique 1RS, Triple-DES (168bits) and AES (128bits) with respect to the following parameters.

5.1 Encryption and decryption times

The encryption and decryption times are taken the differences between processor clock ticks at the starting of execution and ending of execution. The minimum time indicates the highest speed of execution. Encryption and Decryption times (in milliseconds) of twenty different files are calculated for Triple-DES, AES and 1RS. Tables 1 and 2 show the encryption and decryption times respectively of TDES, AES and 1RS for different source files. Files are taken in ascending order of their size. Figures 3 and 4 indicate the graphical representation of encryption times and decryption times respectively for TDES, AES and 1RS of different source files.

Table 1. Encryption times for TDES, AES and 1RS

Sl. No.	File type	Encryption time (in m.sec)		
		TDES	AES	1RS
1	txt	0	0	0
2	zip	0	0	0
3	txt	15	0	0
4	txt	0	0	15
5	jpg	14	0	0
6	docx	45	15	30
7	exe	15	0	30
8	jpg	15	0	61
9	rar	30	0	106
10	dll	45	14	181
11	exe	121	31	545
12	docx	211	30	1073
13	dll	258	75	1240
14	jpg	574	91	3147
15	pdf	726	121	3918
16	avi	1300	196	6431
17	rtf	2572	408	13784
18	doc	6915	1119	38011
19	rar	12317	1998	67322
20	avi	23166	3677	125610

Table 2. Decryption times for TDES, AES and 1RS

Sl. No.	File type	Decryption time (in m.sec)		
		TDES	AES	1RS
1	txt	0	0	0
2	zip	0	0	0
3	txt	0	0	0
4	txt	0	0	0
5	jpg	0	15	14
6	docx	0	0	31
7	exe	14	0	45
8	jpg	14	14	60
9	rar	30	14	91
10	dll	44	30	181
11	exe	120	60	559
12	docx	226	75	1104
13	dll	257	91	1271
14	jpg	696	195	3207
15	pdf	876	226	3964
16	avi	1407	362	6566
17	rtf	3011	877	14041
18	doc	8201	2556	38707
19	rar	14723	4267	68850
20	avi	27374	8382	128924

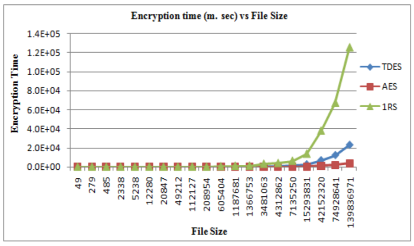

Figure 3. Graphical representation of encryption times against file size in logarithmic scale

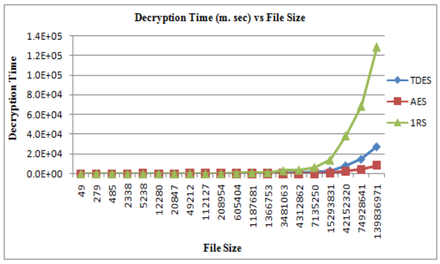

Figure 4. Graphical representation of decryption times against file size in logarithmic scale

5.2 Avalanche, strict avalanche and bit independence criterion

Table 3. Avalanche values for TDES, AES and 1RS

Sl. No.	File type	Avalanche achieved		
		TDES	AES	1RS
1	txt	0.9608	0.9634	0.2425
2	zip	0.9561	0.9684	0.8773
3	txt	0.9658	0.9639	0.9342
4	txt	0.9696	0.9695	0.9611
5	jpg	0.9697	0.9696	0.2425
6	docx	0.9699	0.9699	0.9667
7	exe	0.9697	0.9697	0.9193
8	jpg	0.9700	0.9697	0.9689
9	rar	0.9699	0.9700	0.9684
10	dll	0.9699	0.9699	0.9159
11	exe	0.9700	0.9700	0.9470
12	docx	0.9700	0.9700	0.9698
13	dll	0.9700	0.9700	0.9432
14	jpg	0.9700	0.9700	0.9468
15	pdf	0.9700	0.9700	0.9646
16	avi	0.9700	0.9700	0.9515
17	rtf	0.9700	0.9699	0.9434
18	doc	0.9699	0.9691	0.9342
19	rar	0.9700	0.9700	0.9687
20	avi	0.9700	0.9700	0.9669

The degree of security of cryptographic technique is measured by Avalanche, Strict avalanche and Bit Independence cryptographic test mechanisms. The bit changes among encrypted bytes for a single bit change in the original message sequence for the entire or a large number of bytes. The high degree of security is indicated by the values of Avalanche and Strict Avalanche if it is closer to 1.0. Tables 3, 4 and 5 show the Avalanche & Strict Avalanche values and Bit Independence values respectively for Triple-DES, AES and 1RS which are closer to 1. Figures 5, 6 and 7 represent the graphical representation of Avalanche and Strict Avalanche and Bit Independence values respectively with respect to different files where files are taken in ascending order of its sizes. This analysis indicates that 1RS may provide good security.

Table 4. Strict avalanche values for TDES, AES and 1RS

Sl. No.	File type	Strict Avalanche achieved		
		TDES	AES	1RS
1	txt	0.8763	0.8982	0.1776
2	zip	0.9159	0.9432	0.8620
3	txt	0.9595	0.9592	0.9184
4	txt	0.9674	0.9688	0.9564
5	jpg	0.9690	0.9690	0.1923
6	docx	0.9695	0.9692	0.9660
7	exe	0.9691	0.9694	0.9108
8	jpg	0.9696	0.9696	0.9687
9	rar	0.9698	0.9696	0.9683
10	dll	0.9698	0.9698	0.9085
11	exe	0.9699	0.9699	0.9347
12	docx	0.9699	0.9700	0.9698
13	dll	0.9699	0.9699	0.9341
14	jpg	0.9700	0.9699	0.9451
15	pdf	0.9700	0.9700	0.9642
16	avi	0.9700	0.9700	0.9502
17	rtf	0.9698	0.9697	0.9326
18	doc	0.9698	0.9684	0.9308
19	rar	0.9700	0.9700	0.9686
20	avi	0.9700	0.9700	0.9666

Table 5. Bit independence values for TDES, AES and 1RS

Sl. No.	File type	Bit Independence achieved		
		TDES	AES	1RS
1	txt	0.1517	0.2520	0.0273
2	zip	0.3817	0.3465	0.6216
3	txt	0.3994	0.3870	0.4401
4	txt	0.4657	0.4694	0.4896
5	jpg	0.9420	0.9460	0.7381
6	docx	0.9464	0.9429	0.9548
7	exe	0.6147	0.5915	0.6887
8	jpg	0.9676	0.9678	0.9601
9	rar	0.9678	0.9674	0.9676
10	dll	0.7307	0.7319	0.7447
11	exe	0.7237	0.7181	0.7521
12	docx	0.9611	0.9612	0.9608
13	dll	0.7037	0.7077	0.7728
14	jpg	0.9649	0.9649	0.9460
15	pdf	0.9461	0.9344	0.9634
16	avi	0.9635	0.9621	0.9495
17	rtf	0.3624	0.3288	0.3662
18	doc	0.3301	0.2141	0.5124
19	rar	0.9698	0.9697	0.9689
20	avi	0.9588	0.9582	0.9602

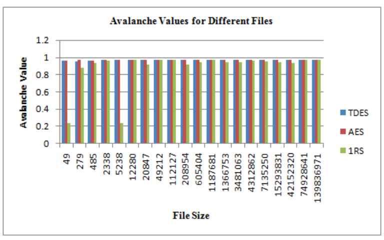

Figure 5. Graphical representation of avalanche value against file size in logarithmic scale

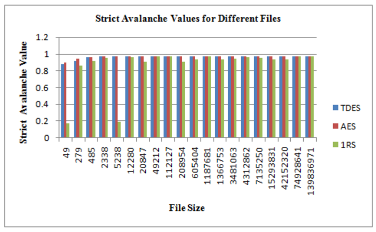

Figure 6. Graphical representation of strict avalanche value against file size in logarithmic scale

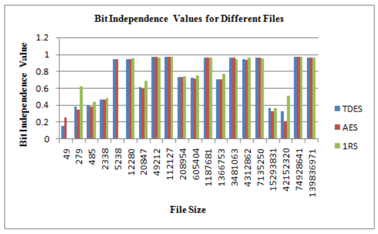

Figure 7. Graphical representation of bit independence value against file size in logarithmic scale

5.3 Chi-square values

A high degree of non-homogeneity among source and encrypted files may be indicated by the large Chi-square value compared with tabulated value. The Chi-square values for Triple-DES (168bits), AES (128bits) and 1RS is shown in Table 6. Average chi-square values of Triple-DES (168bits), AES (128bits) and 1RS are 34143114280, 32603653459 and 62483634354 respectively. Figure 8 shows the comparison of the Chi-square values of all three techniques against the twenty source files. From the figures, it is noticed that the degree of non-homogeneity of the encrypted files with respect to source files using the technique 1RS is very high. Hence it may conclude that 1RS provides good security.

Table 6. Chi-square values for TDES, AES and 1RS

Sl. No.	File type	Chi-Square values		
		TDES	AES	1RS
1	txt	114	111	140
2	zip	503	529	520
3	txt	1470	1546	2495
4	txt	24059	20981	28721
5	jpg	936	946	869
6	docx	18333	9343	1076
7	exe	1044334	481174	114157
8	jpg	1373	1301	4175
9	rar	1030	1038	660
10	dll	530984	473027	360601
11	exe	2027105	1848171	2235771
12	docx	54964	55574	91023
13	dll	3219750	3139562	3138115
14	jpg	78927	79298	109954
15	pdf	413610	369563	1451572
16	avi	438208	442887	254523
17	rtf	6.8E+11	6.5E+11	12.4E+11
18	doc	288821670	267709342	801726632
19	rar	61298	61037	6915
20	avi	15912744	15646387	12154750
Average		3.4E+10	3.2E+10	6.2E+10

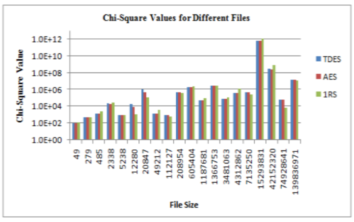

Figure 8. Graphical representation of bit independence value against file size in logarithmic scale

5.4 Other statistical measures

As a measure of non-homogeneity measure of Central tendency in terms of median, mode and measure of Dispersion in terms of standard deviation have been performed. Table 7 shows the values of median, mode and standard deviation of source stream and encrypted stream using 1RS for three different files. Using Karl Pearson's Product Moment Correlation Coefficient formula, the correlation coefficient between the source stream and cipher stream is measured. Product moment correlation coefficient of three types of source streams and the corresponding encrypted streams has been also presented in Table 7 from which it is observed that there is negligible correlation between the source stream and the cipher stream. This result indicates that 1RS may provide good security.

Table 7. Median, mode, standard deviation and correlation coefficient values using 1RS

Value of	Stream	S08.png	S10.dll	S17.rtf
Median (character with ASCII value)	Source	123	102	99
	Encrypted	124	102	87
Mode (character with ASCII value)	Source	0	0	92
	Encrypted	0	0	85
Standard Deviation	Source	93	2391	221568
	Encrypted	87	1658	151033
Correlation Coefficient	Source & Encrypted	0.79	0.89	0.13

6. CONCLUSION

The proposed technique 1RS is simple to comprehend and easy to implement using various high-level languages. Because of high processing speed and the measure of the degree of security is at par with Triple-DES and AES the performance of 1RS is quite acceptable. It is applicable in message transmission of any size and any form. Some of the salient features of 1RS can be summarized as follows:
(1) Session based key implementation
(2) Bock size independency
(3) High degree of security

ACKNOWLEDGMENT

Let us express our heartiest gratitude to all who are directly or indirectly provided all sorts of supports during the entire development process.

REFERENCES

[1] Mandal B.K., Bhattacharyya D., Bandyopadhyay S.K. (2013). Designing and performance analysis of a proposed symmetric cryptography algorithm, *International Conference on Communication Systems and Network Technologies (CSNT 2013)*, Gwalior, pp. 453-461.

[2] Paul M., Mandal J.K. (2013). A novel generic session based bit level cryptographic technique based on magic square concepts, *International Conference on Global Innovations in Technology and Sciences (ICGITS 2013)*, Kottayam, pp. 156-163.

[3] Niemiec M., Machowski L. (2012). A new symmetric block cipher based on key-dependent S-boxes, *4th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT 2012)*, St. Petersburg, pp. 474-478.

[4] Cheng H., Ding Q. (2012). Overview of the block cipher, *Second International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC 2012)*, Harbin, pp. 1628-1631.

[5] Paul M., Mandal J.K. (2012). A universal session based bit level symmetric key cryptographic technique to enhance the information security, *International Journal of Network Security & Its Application (IJNSA)*, Vol. 4, No. 4, pp. 123-136.

[6] Navin A.H., Oskuei A.R., Khashandarag A.S., Mirnia M. (2011). A novel approach cryptography by using residue number system, *6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT 2011)*, Seogwipo, pp. 636-639.

[7] Paul M., Mandal J.K. (2011). A novel generic session based bit level cryptographic technique to enhance information security, *International Journal of Computer Science and Network Security (IJCSNS)*, Vol. 11, No. 12, pp. 117-122.

[8] Som S., Chatergee N.S., Mandal J.K. (2011). Key based bit level genetic cryptographic technique (KBGCT), *7th International Conference on Information Assurance and Security (IAS)*, Melaka, pp. 240-245.

[9] Triple Data Encryption Standard. (1999). FIPS PUB 46-3 Federal Information Processing Standards Publication, Reaffirmed, Department Of Commerce/National Institute of Standards and Technology.