Research Journal of Applied Sciences, Engineering and Technology 15(1): 1-6, 2018
DOI:10.19026/rjaset.15.5285
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: October 31, 2016 Accepted: February 13, 2017 Published: January 15, 2018

Research Article

Improved RC4 Algorithm Based on Multi-Chaotic Maps

[1]Naji Mutar Sahib, [1]Ali Hussein Fadel and [2]Noora Shihab Ahmed
[1]Department of Computer Science, University of Diyala, Baghdad,
[2]Department of Computer Science, University of Halabja, Kurdistan, Iraq

Abstract: The aim of this study is to overcome on the weakness points in RC4 (Rivest Cipher 4) algorithm, there are some blemishes in the Key Scheduling Algorithm (KSA) of RC4. This study presented improved RC4 key generation based on multi-chaotic maps. The new version of KSA coined as improved KSA (IKSA), the permutation of the S array modified to depend on the random numbers generator based on three chaotic maps and the proposed algorithm outputs as follows: Output = M XOR Generated key XOR Random value from IKSA (R3w) The improved RC4 with IKSA is tested for its secrecy, randomness and performance over the variable key length and different plaintext size with respect to those of the original RC4.The results show that the improved RS4 with IKSA is better than the original RC4 with KSA.

Keywords: Average secrecy, key scheduling algorithm, multi-chaotic maps, RC4

INTRODUCTION

RC4 is the vastly stream cipher and used in many internet protocols such as wired equivalent privacy (WEP), Skype, Wireless Protected Access (WPA) and Secure Socket Layer, Transport layer security (SSL/TLS) (Crainicu, 2015). The important factors in RC4 algorithm over such an extensive domain of applications have been its speed and simplicity; efficient implementation in both software and hardware were very easy to develop. RC4 is very simple and fast compared to other encryption algorithms. Weerasinghe (2012) presented the analysis of a simply modified RC4 algorithm and tried out a simple modification of RC4 PRGA, where we can mention it like this: Out Put = M XOR Generated key XOR j.

Hameed and Mahmood (2016) present a new version of KSA is suggested in an attempt to increase the security of RC4 and get rid of the weakness related to the initial permutation of the S array and the permutation process of the S array.

In Fluhrer *et al*. (2001) we analyzed the KSA which derives the initial state from a variable size key and describe two significant weaknesses of this process. The first weakness is in the existence of a large number of bits of the initial permutation (KSA output). The second weakness is related to key vulnerability, which applies when part of the key presented to the KSA in exposed to the attacker.

In this study we present a new improvement of the KSA depend on the randomness of the three chaotic maps (Logistic, tent and Chebyschev). The chaotic maps have many good features such as allergy on primary condition and system parameter, periodicity and mixing properties. In this study, we invest these interesting properties of chaotic maps to generation random number. The S array permutation is suggested to depend on the generated random key.

MATERIALS AND METHODS

RC4 Algorithm: Ron Rivest (Stallings, 2011), one of the inventors of RSA inserted the RC4 algorithm in 1987. RC4 is an acronym for" Rivest Cipher 4", it is also known as "Ron's Code 4". The algorithm is based on the use of a random permutation. The RC4 algorithm is simple and relatively easy to explain (Mao, 2003; Rahma and Hussein, 2015).

- **Algorithm (RC4 Stream Cipher Algorithm)**
  Input [plaintext] and [key]
  Output [cipher text]
  Step 1: /Initialize /
  for i = 0 to 255
  S[i] = i;
  T[i] = K[i mod key];
  Next i;
  Step 2: /Perform IP of S/
  Set j = 0;
  For i = 0 to 255
  j = (j + S[i] + T[i]) mod 256;
  Swap (S[i], S[j]);

Corresponding Author: Naji Mutar Sahib, Department of Computer Science, University of Diyala, Baghdad, Iraq

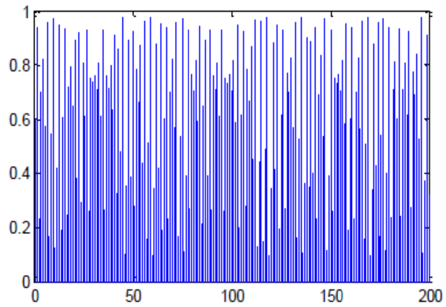

Fig. 1: Iterative sequence value of logistic mapping

Step 3: /Stream Generation/
Set [i, j] = 0;
while (true)
i = (i + 1) mod 256;
j = (j + S[i]) mod 256;
Swap (S[i], S[j]);
t = (S[i] + S[j]) mod 256;
k = S[t];
Step 4: /The process/
Step 4.1: Encryption C = P K
Step 4.1: Decryption P = C K
Step 5: /End/

Chaotic maps:

- **The logistic map:** Logistic map is a paradigmatic representation of chaotic mapping. In spite of the fact that logistic mapping is one dimensional, however, the control reaction is quite ideal. The following equation exemplifies the logistic formula:

$$x_{n+1} = \lambda x_n(1 - x_n)\ (n = 0,1,2, \dots) \quad (1)$$

In the equation, x_n is symbolized to the variable, also λ is an indication of system parameter whereas $\lambda \in (0,4]$, $x_n \in [0,1]$. In case that $1 \leq \lambda < 3$ the system takes the act of 'fixed point'. If λ = 3 the system starts the transmission stage. When λ = 3.5699456, the system periodically a chaotic condition. In case λ = 3.9, the starting value of x_n is 0.6. In logistic mapping extent, periodical the process for 200 times with chaotically ordered collection (sequence values) comes up with the products shown in Fig. 1.

- **Chebyschev map:** A particularly interesting candidate for chaotic sequences generators is the family of Chebyschev map, whose anarchism can be verified easily with many other properties is accessible to rigorous mathematical analysis. The independent binary sequences generated by a chaotic Chebyshev map (Prasadh *et al*., 2009) were shown to be not significantly different from random binary sequences. This map is defined by:

$$f(z_{n+1}) = cos\big(karccos(z_n)\big), -1 \leq z_n \leq 1, n = 1,2,3 \dots \quad (2)$$

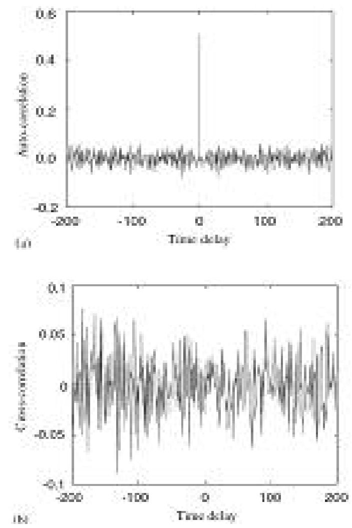

Fig. 2: The statistical correlation curves of a chaotic Chebyshev sequence; (a): Auto-correlation curve of the chaotic sequence when the initial value of 0.60000; (b): Cross-correlation curve of two chaotic sequences when their initial values are 0.60000 and 0.60001, respectively

Here, the map is chaotic for k ≥ 2 and we use k = 4 in this study. Figure 2 shows two time series of this map, with initial values, differed only by 10-5; indicating that the map can generate good chaotic (pseudorandom) sequences satisfying a basic requirement of a cryptosystem that demands such randomness. Figure 2 further shows its statistical correlation curves.

- **The tent map:** The tent map is real-valued formula based on μ parameter and is denoted be fμ. Tent map formula can be expressed by:

$$f_\mu = \mu \min\{y, 1 - y\}$$

The reason behind its naming is the likeness of its graph to tent shape. By setting the parameter μ with values from 0 up to 2, fμ charts the:

$$y_{n+1} = f_\mu(y_n) = \begin{cases} \mu y_n & for\ y_n < \frac{1}{2} \\ \mu(1 - y_n) & for\ \frac{1}{2} \leq y_n \end{cases} \quad (3)$$

where, μ is a positive real variable (constant). The setting, for example, the parameter μ = 2, the outcome of the function f_μ is possibly been discernible as the product of the process of bending the unit duration in twin, thereafter extending the product duration [0, 1/2] to get back the duration [0, 1]. By Repeating the process, each point proposes the new upcoming locations as mentioned above, making a sequence yn. The μ = 2 state of tent mapping is a non-linear transmission of bit shift mapping and state r = 4 of logistic mapping as well (Ahmed, 2016). Figure 3 shows the Orbits of unit-height tent map.

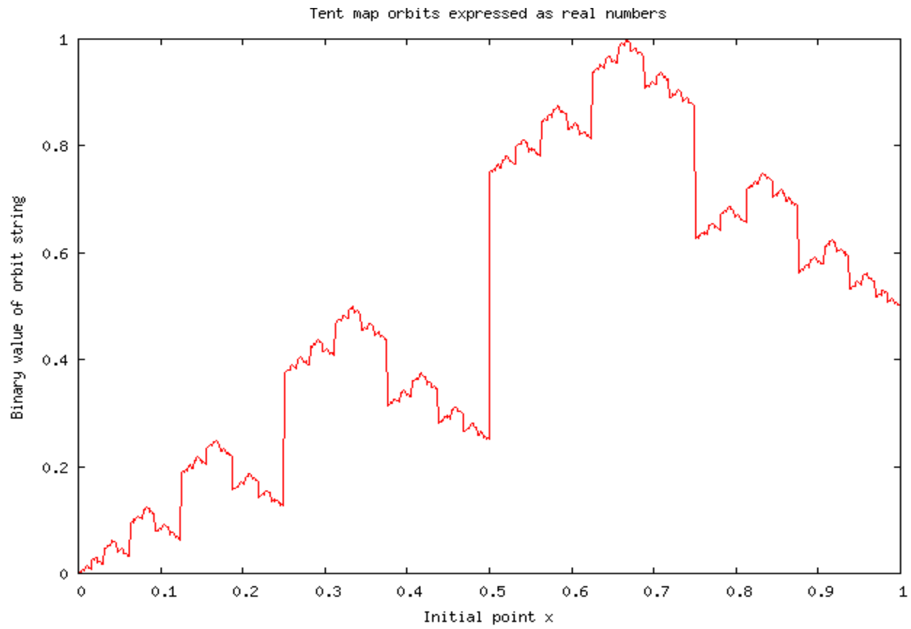

Fig. 3: Orbits of unit-height tent map

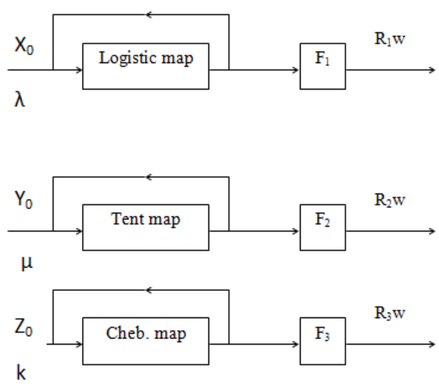

Fig. 4: The proposed algorithm IKSA

Proposed algorithm: The purpose of this part is to establish the improvement RC4 algorithm (IRC4) basically through two stages:

- Improved key scheduling algorithm, a new version of KSA called IKSA is proposed. In this proposal as shown in Fig. 4, we choose three chaotic maps (Logistic, Chebysehev and tent) and their Eq. are (1), (2) and (3), respectively. The secret key is SEED, which is the initial condition of each map. The algorithm generated by each iteration (w = 0 to 255: the number of iterations) sequences of 24 bits (8-bit blocks for each chaotic maps). R1w, R2w and R3w are extracted from chaotic maps as follows:
  Logistic map generate R1w
  Tent map generate R2w
  Chebysehev map generate R3w
  In the following way:

$$F_n(t_{m+1}) = \begin{cases} 0 \; if \; 0 < t_{m+1} \le 0.5 \\ 1 \; if \; 0.5 < t_{m+1} < 1 \end{cases}, \qquad n = 1,2,3$$

- Encryption/Decryption process
  Encryption: C = (M⊕ Generated key⊕R_{3w})mod_{256}
  Decryption: M = (C⊕ Generated key ⊕R_{3w})mod_{256}

Algorithm IRC4
Input [plaintext] and [key]
Output [cipher text]
Step 1: /Initialize /
for i = 0 to 255
S[i] = i;
T[i] = K[i mod key];
Next i;
Step 2: / Perform IP of S /
for w = 0 to 255
R_1w = Location: generate from the Logistic map
R_2w = Location: generate from the Tent map
j = (R_2w + S[R_1w] + T[R_1w]) mod_{256}
Swap (j, S(R_1w])
Next w;
Step 3: /Stream Generation/
Set [i, j] = 0;
while (true)
i = (i + 1) mod 256;
j = (j + S[i]) mod 256;
Swap (S[i], S[j]);
t = (S[i] + S[j]) mod 256;
k = S[t];
R_3w: generate from the Chebysehev map
Step 4: /The process/
Encryption C = (M ⊕ K ⊕R_3w) mod_{256}
Decryption M = (C ⊕ Generation key ⊕ R_{3w})
Step 5: /End/

RESULTS AND DISCUSSION

Algorithm:

Secrecy of ciphers: Secrecy of ciphers is calculated in terms of the key equivocation (conditional entropy of key given cipher):

$$H(k/c) = \sum_{j=1}^{L} \sum_{i=1}^{n} q_i P_{ij} \log P_{ij} \qquad (4)$$

where,

- q_i = Pr (C = ci)
- P_{ij} = Pr (K= ki/C = ci)
- L = The key length
- n = The cipher text length

- **Average secrecy: A variable plaintext size, Fixed key length:** As shown by the Table 1 and Fig. 5a to 5d, improvement RC4 algorithm with IKSA has better average secrecy than the original RC4 algorithm with KSA, using a variable plaintext size (128,256,512 and 1024 bits) and fixed key length for each phase (32, 64, 128 and 256 bits).
- **Average secrecy: A variable key length, Fixed plaintext size:** As shown by the Table 2 and Fig. 6 (a to d), improved RC4 algorithm with IKSA has better average secrecy than the original RC4 algorithm with KSA, using a variable key length (32.64, 128 and 256 bits) and fixed plaintext for each phase (128, 256, 512 and 1024 bits).

Table 1: Average secrecy value vs. plaintext size

		Algorithms	
Keys Length\Bits	Plaintext Size\Bits	Original RC4 with KSA	Improvement RC4 with IKSA
32	128	0.260459373	0.740856729
	256	0.203040633	0.498915456
	512	0.20944977	0.406738053
	1024	0.214365643	0.43235869
64	128	0.363815483	0.740856729
	256	0.245275562	0.531832803
	512	0.174139579	0.544481966
	1024	0.161067288	0.448314224
128	128	0.329087567	0.740856729
	256	0.249318629	0.531832803
	512	0.180433057	0.43880481
	1024	0.197202989	0.503334883
256	128	0.295187289	0.740856729
	256	0.247261403	0.74999756
	512	0.153576778	0.585268432
	1024	0.177807869	0.455159783

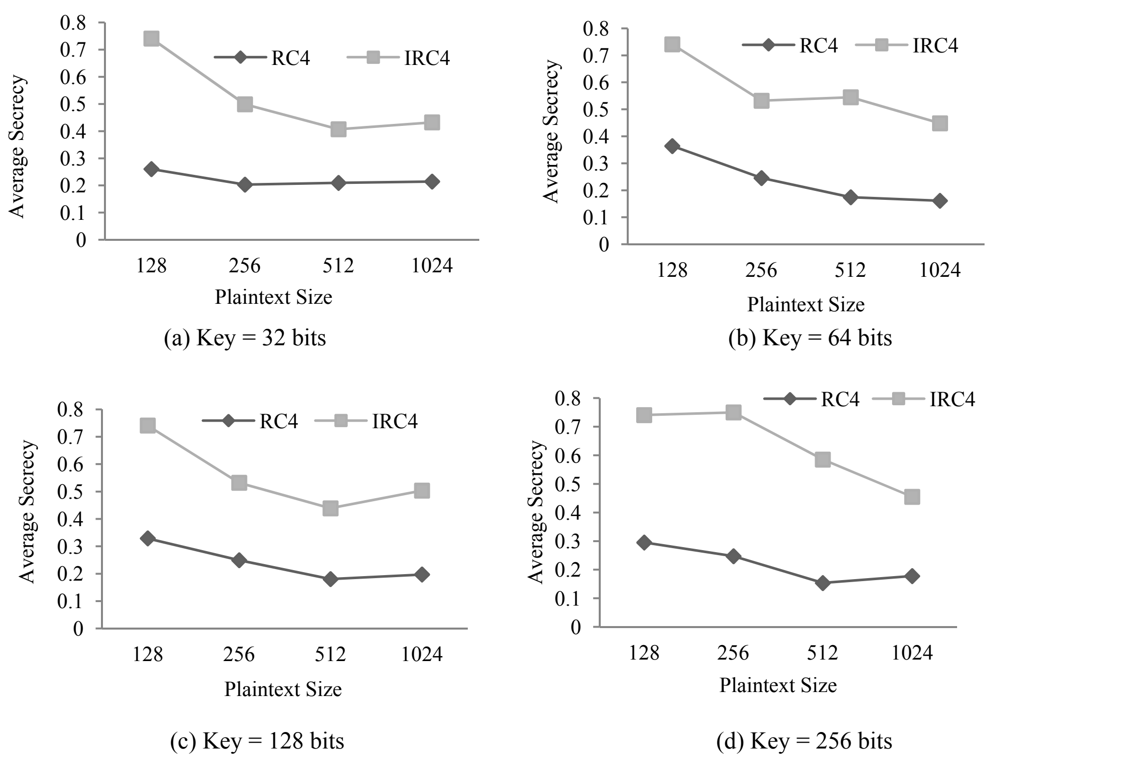

Fig. 5: Average secrecy value vs. plaintext; (a): key = 32 bits; (b): key = 64 bits; (c): key = 128 bits; (d): key = 256 bit

Table 2: Average secrecy value vs. key length

		Algorithm	
Plaintext size/Bits	Keys Length/Bits	Rc4	IRC4
128	32	0.260459373	0.740856729
	64	0.363815483	0.740856729
	128	0.329087567	0.740856729
	256	0.295187289	0.740856729
256	32	0.203040633	0.498915456
	64	0.245275562	0.531832803
	128	0.249318629	0.531832803
	256	0.247261403	0.74999756
512	32	0.20944977	0.406738053
	64	0.174139579	0.54448966
	128	0.180433057	0.43888481
	256	0.153576778	0.585268432
1024	32	0.214365643	0.43235869
	64	0.161067288	0.448314224
	128	0.197202989	0.50334883
	256	0.177807869	0.455159783

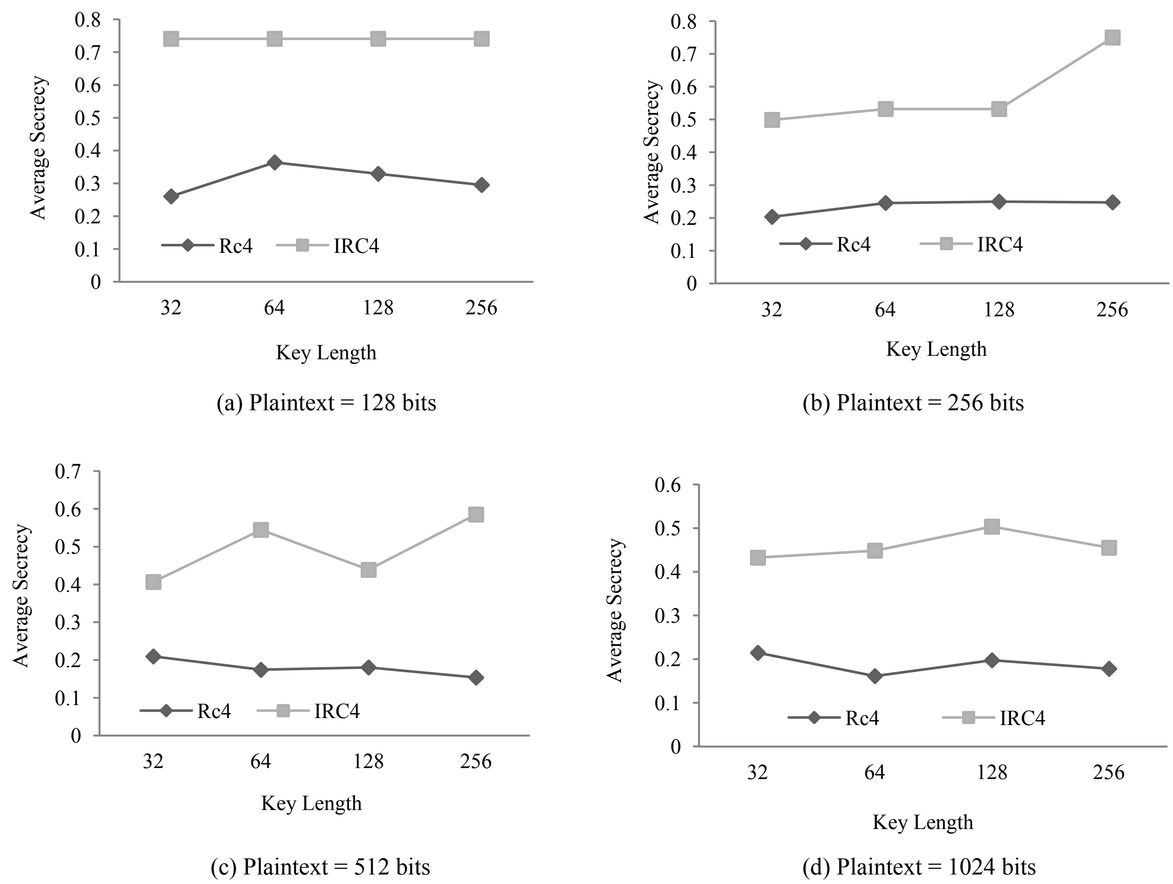

Fig. 6: Average secrecy value vs. key length; (a): Plaintext = 128 bits; (b): Plaintext = 256 bits; (c): Plaintext =512 bits; (d): Plaintext = 1024 bits

Analysis of randomness: The following, many different trials are performed to test the statistical properties of the cipher text generated from improved RC4 algorithm with IKSA. And it is sensitivity to elementary conditions. The four different statistical tests (frequency test, serial test, poker test and run test) on several binary sequences of key size (32, 64, 128 and 256 bits) and plaintext size (128 and 1024 bits). These binary sequences pass all four tests successfully. Results are shown in Table 3.

Table 3: Randomness test for improvement RC algorithm

	Plain size\bits	Statistical tests	Degree of freedom	Value test	Value table	Result
32	128	Frequency test	1	3.125	3.8415	Pass
		Serial Test	2	5.875	5.9915	Pass
		Poker Test	7	11.71428571	15.5073	Pass
		Run test	2	4.59496124	9.4877	Pass
	1024	Frequency test	1	0.31640625	3.8415	Pass
		Serial Test	2	5.070690524	5.9915	Pass
		Poker Test	31	24.07843137	82.5287	Pass
		Run test	8	16.91169031	82.5287	Pass
64	128	Frequency test	1	1.125	3.8415	Pass
		Serial Test	2	1.544291339	5.9915	Pass
		Poker Test	7	6.761904762	15.5073	Pass
		Run test	2	1.042635659	9.4877	Pass
	1024	Frequency test	1	2.640625	3.8415	Pass
		Serial Test	2	3.258690738	5.9915	Pass
		Poker Test	31	39.45098039	82.5287	Pass
		Run test	8	11.77336127	82.5287	Pass
128	128	Frequency test	1	0	3.8415	Pass
		Serial Test	2	2.291338583	5.9915	Pass
		Poker Test	7	6.761904762	15.5073	Pass
		Run test	2	5.06879845	9.4877	Pass
	1024	Frequency test	1	0.0625	3.8415	Pass
		Serial Test	2	1.003971163	5.9915	Pass
		Poker Test	31	24.70588235	82.5287	Pass
		Run test	8	12.2101587	82.5287	Pass
256	128	Frequency test	1	1.53125	3.8415	Pass
		Serial Test	2	1.201033465	5.9915	Pass
		Poker Test	7	2.952380952	15.5073	Pass
		Run test	2	1.727713178	9.4877	Pass
	1024	Frequency test	1	0.19140625	3.8415	Pass
		Serial Test	2	1.015827377	5.9915	Pass
		Poker Test	31	33.49019608	82.5287	Pass
		Run test	8	6.234876551	82.5287	Pass

CONCLUSION

In this study, the improved RC4 algorithm with IKSA based on chaotic maps is proposed (IRC4 algorithm). This algorithm overcome the weakness of the original RC4 with KSA. The average secrecy for the proposed algorithm is best than the original algorithm. The IRC4 algorithm is characterized by secrecy, performance and efficiency because of the permutation of the S array are modified to depend on the key random generation based on three chaotic maps (logistic, Chebysehev and tent).

CONFLICT OF INTEREST

This study is to provide a solution for bypass the RS4 weakness pointes by introducing improved RC4 key generation that based on multi-chaotic maps. The performance criteria secrecy and randomness were used to compare between the introduced improved algorithm with the original variant.

REFERENCES

Ahmed, N.S., 2016. Multi-image encryption technique based on permutation of chaotic system. Int. J. Video Image Process. Netw. Secur., 16(1): 9-18.

Crainicu, B., 2015. On invariance weakness in the KSAm algorithm. Proc. Technol., 19: 850-857.

Fluhrer, S., I. Mantin and A. Shamir, 2001. Weaknesses in the key scheduling algorithm of RC4. In: Vaudenay, S. and A.M. Youssef (Eds.), Selected Areas in Cryptography. SAC, 2001. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2259: 1-24.

Hameed, S.M. and I.N. Mahmood, 2016. A modified key scheduling algorithm for RC4. Iraq. J. Sci., 57(1A): 262-267.

Mao, W., 2003. Modern Cryptography: Theory and Practice. Prentice Hall, Upper Saddle River, NJ.

Prasadh, K., K. Ramar and R. Gnanajeyaraman, 2009. Public key cryptosystems based on chaotic chebyshev polynomials. J. Eng. Technol. Res., 1(7): 122-128.

Rahma, A.M.S. and Z.M. Hussein, 2015. Modified RC4 dual key algorithm based on irreducible polynomial. Int. J. Emerg. Trend. Technol. Comput. Sci., 4(2): 79-85.

Stallings, W., 2011. Cryptography and Network Security Principles and Practices. 5th Edn., Pearson Education Inc., Pearson Prentice Hall, USA.

Weerasinghe, T.D.B., 2012. Analysis of a modified RC4 algorithm. Int. J. Comput. Appl., 51(22): 12-17.