International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Design and Implementation of Secure Stream Cipher Algorithm

Ali M. Sagheer [1], Salih S. Salih[1] and Sura M. Searan [1]

[1]*Department of Information Technology, University of Anbar, Baghdad, Iraq*

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: RC4 stream cipher is an encryption algorithm that is used in two domains of security realized for IEEE 802.11 LANs: one of them is wired equivalent confidentiality and the other is WI-FI Protected Access protocol. It is symmetric encryption, fast, and simple algorithm. It has different weaknesses such as the bias that occurs in the key stream bytes. Key scheduling phase is more intricate. It was prepared to be unnecessarily easy. The initial few bytes of keystream that are generated by pseudo random generation algorithm (PRGA) are biased to several bytes of the private key. Thus, the byte analysis makes it probable for attacking RC4 in some methods of operations. This paper presents a new stream cipher algorithm as a development for RC4, named double RC4 key generation, which is submitted to solve interconnection problems in the output of the inner state and solve the problem of weak keys. The new idea allows random access to the key stream. The keystream depends on all preceding state bytes. RC4 and the proposed algorithm are analyzed and proved that the new algorithm has no single or double byte bias in the key stream while RC4 has proved the same bias that is shown in the literature. The proposed algorithm introduces high resistance against different attacks that are applied to RC4 and the generated key stream has high randomness, large complexity, and good statistical properties.

Keywords: : RC4, Key Scheduling Algorithm (KSA), Pseudo Random Generation Algorithm (PRGA), double RC4 key generation, single bias, double bias.

1. Introduction

Encryption is a process that transforms plain text into cipher text to hide its contents and to prevent unofficial participants from retrieving plain text. Encryption is basically used to ensure privacy. Encryption uses data by symbolizing it as symbols or numbers by a specific encryption key. Both transmitter and receiver realize this key that is utilized for data encryption and decryption [1]. Cryptographic algorithms provide lower size, high speed of implementation, a larger level of security, and less complexity [2]. Classical cryptographic algorithms are very complex and take higher capacity. Since public key ciphers are yet not appropriate in tracer connections for many causes such as finite store [3]. Protection systems are based on the symmetric key cryptology especially in the equipment that has restricted hardware resource [4]. Encryption algorithms, used in wired networks, are classified into symmetric and asymmetric cryptology [5]. Stream cipher can be categorized into an asynchronous stream and the other is synchronous stream ciphers. In synchronous stream, a key series is got from the plain text and cipher text. The synchronous stream disadvantage is that both the transmitter and receiver must be synchronized for key uses. It can discover every bit of insertion or deletion by the active attack. Thus, this attack can cause instant loss of concurrence. Asynchronous cipher depends on the ciphertext that is generated previously to continue producing new ciphers and it cannot repeat generating the same ciphertext series [6]. RC4 is the most popular stream encryption. Ron Rivest proposed it in 1987. It was a commerce confidential to 1994. It is used in Oracle Secure SQL, WEP Protocol [7] and is used to conserve wireless connections as a portion of WPA protocols. Fluhrer, Shamir, and Mantin portrayed the attack on this. RC4 is an algorithm that uses symmetric encryption; it is substantially one of the encryption algorithms. They encrypt original data of one at a time [8]. This algorithm consists of two main combinations for key generation, the first is KSA and the other is PRGA [9]. RC4 begins with the alteration and utilizes a secret key for producing a random permutation with KSA based on the secret key. The next stage is PRGA that generates keystream bytes [10]. The operation performed between the key and plain text is equivalent in some regards to the Vernam cipher. The key is restricted

E-mail: ali.m.sagheer@gmail.com, Salihsami90@gmail.com, surasms917@gmail.com

to 40 bits because of missing of restrictions, but it is occasionally utilized as a 128-bit of key [11]. Different people analyzed RC4 and different weaknesses were detected. KSA is more problematic, it was prepared to be simple. Then, at the beginning few bytes of the output of PRGA are biased or attached to several bytes of the secret key; therefore, analyzing these bytes makes them probable for attacking RC4 [12]. There are different attack types classified by the amounts of information available to the adversary for cryptanalysis based on available resources [13].

The modern researches show that you can practically utilize the short-term and long-term biases for RC4 to acquire any part of the Internet traffic, depending on TLS (Transport Layer Security) with RC4 cipher option. This work covers analyzing RC4 based on its single and double byte biases by using new efficient algorithm. Single byte bias algorithm is designed to compute the frequency of each value in each position while double byte bias algorithm is designed to compute the frequency of each pair of bytes in each position. Also, it includes a new proposal as development for RC4 to reduce this weakness.

2. RELATED WORKS

Many researchers have analyzed RC4 algorithm based on its weakness and suggest different algorithms.

Mantin I. and Shamir A. (2001) [14] presented a major statistical weakness in the keystream of RC4 by analyzing this algorithm. This weakness makes it important to discriminant between random strings and short outputs of RC4 by analyzing the second bytes. They observe that the second output byte of RC4 enjoys a very strong bias that takes the value 0 with twice the expected probability (1/128 instead of 1/256 for n = 8). The main result is the detection of a slight distinguisher between the RC4 and random ciphers that requires only two output words under many hundred unrelated and unknown keys to make robust decision. In this work, their bias is also proved. Fluhrer S.R. & McGrew D.A. (2001) [1] were the first researchers that detected a new type of bias in the RC4 keystream described as double byte bias in a pair of bytes following each other and illustrated a new method to distinguish 8-bit of RC4 from random bits. The long-term biases that are detected for RC4 are composed of ten conditions stating the positive biases that have probability higher than the meant value and two conditions stating the negative biases that means their likelihood is less than the intended value. Their bias is endorsed in this work. McKague (2005) [15] illustrated RC4 security and suggested new cipher, Chameleon uses a similar organization to RC4 but in different ways and develops a new cipher, RC4B, for increasing the security of RC4. Maitra and Paul (2008) [10] analyzed Rivest Cipher 4 based on weakness in biases and proposed additional layers over key scheduling phase and pseudo-random generation phase. Beck and Tews (2008) [16] illustrated two attacks on IEEE 802.11 based wireless LANs. The first is a key recovery attack on WEP to reduce the average number of packets an attacker intercept to recover the private key, the other is the first practical attack on WPA secured wireless networks. AlFardan et al. (2013) [9] determined RC4 security in transport layer safety and Wi-Fi protected arrival and determined countermeasures and determined single and double bias in the key stream. They retrieved plain text bytes from cipher text using single and double bias attacks. Jindal and Singh (2014) [2] described many weaknesses of an RC4 algorithm based on newly suggested enhancements, it is proved that survey works are demanded developing safe RC4 algorithms to reduce the RC4 impairment and the biased bytes and described key-retrieval attacks on Wi-Fi preserved access. Hammood, Yoshigoe, and Sagheer (2015) [6] worked on enhancing security and speed of RC4 by suggesting algorithms as enhancement for RC4, solving weak keys problem and making it robust by using random initial state. They also suggested RC4 cipher with two states as enhancement for RC4 to resolve the attachment problem between public recognized outputs of the inner state using commutation between State1 and State2. In addition, they developed RC4 using random elementary state in order to increase randomness and to eliminate the attachment between the output bytes of the inner state. The last suggestion was RC4 using two states for generating four keys in every round to increase randomness. Searan, Sagheer, and Hammood (2016) [17] were designed new algorithms for measuring single and double byte biases in RC4 key stream bytes, also they were attacked RC4 algorithm based on its single byte bias and retrieved the first 32 bytes of RC4 plain text. These weaknesses of RC4 remain to present an open challenge to the developers.

3. DESCRIPTION OF RC4 CIPHER

RC4 is the quicker used software of stream cryptology. It is very fast and it is simple in design [18]. It is a set of ciphers represented by integer number n, which shows the size of the word in bits. The inner state is a table (State) of (2n word) [19]. Many of stream cryptographic ciphers use linear feedback shift records particularly in the hardware but RC4 design does not use it. It has a variable length of keys in the domain (0 and 255) of bytes to initialize the array of 256-byte in the elementary state [17].

RC4 is carried out in two phases: The initial step is key scheduling phase, it initializes the inner state to make a replacement of {0, 1, 2..., N − 1} using a variable size of key [8].

Algorithm 1. KSA of RC4

Input: Secret Key

Output: State Table

```
1.   For ($a = 0$ to 255)
     1.1. State[a] = a
2.   Set b = 0
3.   For ($a = 0$ to 255)
     3.1. $b = (b + \text{State}[a] + \text{Key}\ [a \bmod \text{key-length}]) \bmod 256$
     3.2. Swap (State[a], State[b])
4.   Output: State Table
```

The second is a PRNG. It generates the output key stream.

Algorithm 2. PRGA of RC4

Inputs: State, Plaintext $_a$

Outputs: Key sequence (K sequence)

```
1.   $a = 0, b = 0$
2.   For ($a = 0$ to Plaintext length)
     2.1. $a = (a + 1) \bmod N$
     2.2. $b = (b + \text{State}[a]) \bmod N$
     2.3. Swap (State[a], State[b])
     2.4. K sequence = State [State[a] + State[b]] mod N
3.   Output: K sequence
```

The output sequence of key K is XORed with the plaintext

$C_a = K_a \oplus \text{Plaintext}_a$ [20].

4. RC4 Weaknesses

The RC4 algorithm has several weaknesses. Some of these are simple but other weaknesses are dangerous that can be exploited by attackers. One of these weaknesses in initialization state is statistical biases occurring in distributing words of the first byte of output [15]. This bias makes it slight to differentiate between many different shortened outputs of RC4 and random series by analyzing the second word of them. This impairment is used to make an effective cipher-text-only attack on this algorithm [16]. The key stream beginning algorithm swaps the entry of the s-box exactly one time [21]. Roos found RC4 weaknesses of the high correlation between the first state table values and produced value. The essential cause is the state array that began in series (0, 1… 255) and at minimal one out of each 256 potential keys The first generated byte of the key is highly attached with few key bytes [22]. The attack aims to retrieve the main key, the inner state, or the final key stream to arrive at the original messages [23]. Mantin and Shamir found the main weakness of RC4 in the second round the likelihood of zero output bytes [14]. Fluher found a large weakness if anyone knows the private key portion then potential to attack fully over the RC4 [24]. Paul and Maitra found the private key by using the elementary state array, generated equations on the elementary state array bases, and selected some of the secret key bytes based on assumption and kept private key discovery by utilizing the equation that used. So, the safety of RC4 is based on a private key security and the internal states. Various attacks focus on getting the private key of the inner states of S-box. For making RC4 secure and capable of standing against the attack, many researches were done on RC4 for enhancing RC4 security [25]. Also, there are some weaknesses of RC4 that is the biased bytes, distinguishers [26], key collisions, and key retrieving from the state [27]. The attack goal is to recover the key, the internal state, or the final key sequence [28].

5. The Proposed Algorithm (Double RC4 Key Generation Algorithm)

There are many weaknesses were detected in RC4, they are in the KSA phases and in PRGA phase. Many developers have attempted to solve these issues by proposing different suggestions; however, there remains an issue in the randomness of these algorithms and resulted in additional delays. Our proposed algorithm aims to increase the randomness as well as the same implementation time of RC4 and solve the connection between publically known outputs of the inner state and got a key stream with high randomness. The proposed algorithm is composed of two phases that is KSA and the other is PRGA as shown below. All operations are carried out in modulo state length. The major concept of this suggestion is to use the private key with separation into two portions and implement the KSA and PRGA with the first part of the secret key with additional operations. The generated key stream is used as a new state table in the second tour with the second part of the key stream to get the final key stream that is used for encryption by using XOR operation with the plain text to get the cipher text. In the phase of key generation, a random number is used and should be kept secret and only sender and receiver can know it. In the decryption, the same random number that is used for encryption is used. Algorithm steps is determined below:

1. Get the data and private key to use it for encryption.
2. Generate random number between 0 and 255 to be initial value.
3. Generate two arrays.
4. Fill the first array with numbers from 0 to 255.
5. Put the key in the other array.
6. Separate the key array into two arrays.
7. Permute the first array based on the first part of the key array.
8. A permutation of the first array through itself and initial value to get the keystream.
9. Obtain generated key stream and use it as a new state table.

10. Repeating the above steps by using the new state and the second part of the key.

Generating the final key sequence that is XOR-ed with plain text to obtain the cipher text.

The proposed algorithm is robust against any attack applied on RC4 such as Klein/PTW attack and FMS attack, this algorithm is free from bias that is a main weakness found in RC4 and cause it vulnerable to the attack.

The first phase is KSA:

Algorithm 3. KSA for Double RC4 Key Generation Algorithm

Input: Secret Key

Output: State Table

```
1.  r = random number
2.  Set b = 0
3.  For (a = 0 to 255)
      3.1.  b = (b + State[a] + Key [a mod key length]) mod 255
      3.2.  Swap (State[a], State[b])
4.  Output:  State Table
```

The other is PRGA phase as shown below:

Algorithm 4. PRGA for Double RC4 Key Generation Algorithm

Inputs: State Table, Plaintext

Outputs: Key sequence (K_i), Ciphertext (C_i)

```
1.  For (a = 0 to StateLength-1)
      1.1.  State[a] = a
2.  For (l = 0 to key Length / 2)
      key1[l] = key[l]
3.  For (l = key Length / 2 to key Length)
      key2[l - key Length / 2] = key[l]
4.  a = 0, b = 0

5.  Call Algorithm 3: KSA (State, key1, r)

    Call Algorithm 3: KSA (New-State, key2, r)
6.  while (a < Plaintext length)
      6.1.  a = (a + 1) mod N
      6.2.  b = (b + State[a]) mod N
      6.3.  n= (b + State [r mod N]) mod N
      6.4.  Swap (State[a], State[n])
      6.5.  $K_i$ = State [ (State[a] + State[(State[r] + State[b]) mod N] + State [(r + n) mod N) mod N]
      6.6.  if (0 < i and i < 257)
      6.7.  New-State [i - 1] = ($K_i$ mod N)
      6.8.  $C_i = P_i \oplus K_i$
7.  Output:  $K_i$ and $C_i$
```

The model of double RC4 key generation algorithm is shown in figure 1.

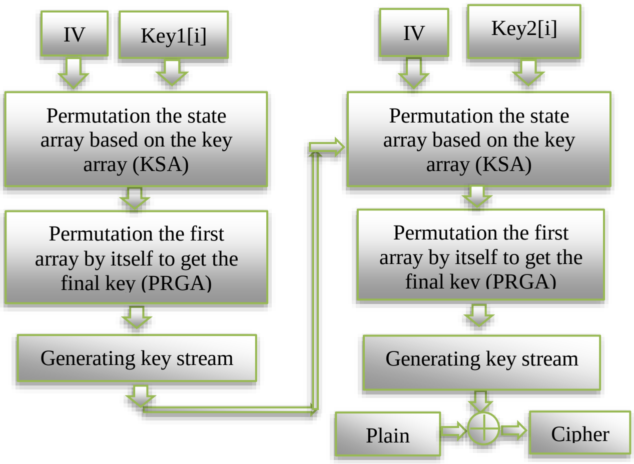
IV
Key1[i]
Permutation the state array based on the key array (KSA)
Permutation the first array by itself to get the final key (PRGA)
Generating key stream
IV
Key2[i]
Permutation the state array based on the key array (KSA)
Permutation the first array by itself to get the final key (PRGA)
Generating key stream
Plain
Cipher

Fig. 1. The model of developed RC4 encryption algorithm.

6. The Analysis of RC4 and Double RC4 Key Generation Algorithm Based on Single Byte Bias

Mantin and Shamir were the first researchers that denoted the bias in the key stream of RC4, their result was highly accurate. Sen Gupta et al. determined a key-length-based bias in the output key of RC4 and work with 256-byte keys [14]. AlFardan et al. (2013) denoted additional biases in the key stream of RC4 that do not have theoretical observations. In this work, RC4 and the proposed algorithm have been analyzed. Double RC4 key generation algorithm has no bias in key distribution bytes as determined below because of using additional operations that cause no attachment between internal state and the output sequence. Algorithm 5 is used for measuring the distribution of key stream bytes.

Algorithm 5. Measuring distributions of key stream bytes

Input: Key [k_1, k_2, …., k_{16}].

Output: Key position (Kp), Key value (Kv), and the number of Frequents in each position (Kf).

```
1.  For (z = 1 to $2^{34}$) Do
      1.1.  a = 0, b = 0
      1.2.  Call Algorithm 1: KSA (Key Scheduling Algorithm)
      1.3.  Call Algorithm 2: PRGA (Pseudo-Random Generation Algorithm).
      1.4.  Deducting new key with length =16 from each generated key.
2.  For (a = 1 to N)
      For (b = 1 to values. Count)
      2.1.  If (values [a] == value)
      2.2.  Increment count by 1
      2.3.  Frequents = (count / ($2^{34}$ * 16))
3.  Return Kp, Kv, and Kf for each position of key stream bytes.
```

The state table is analyzed with 32 positions to reduce the search space and 2^{34} secret keys, each one with length 16, producing 32 positions to calculate the frequency of each 32 values in each position. The key distribution bytes of RC4 and double RC4 key generation algorithm are shown in figures 2, 3, 4, and 5. The X-axis shows the values in each position from 0 to 31 and Y- axis shows the frequents of each value.

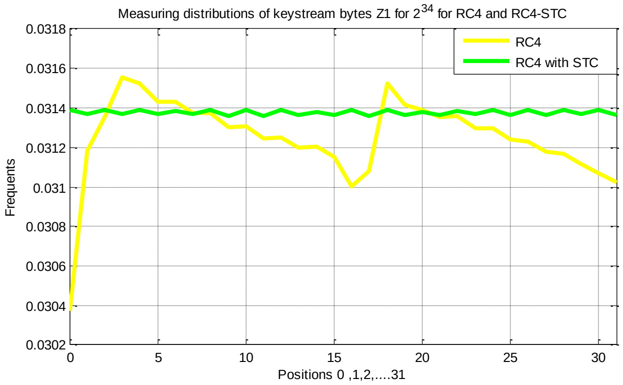

Fig. 2. Key distribution byte in the 1st position for RC4 and RC4-STC with 2^{34}.

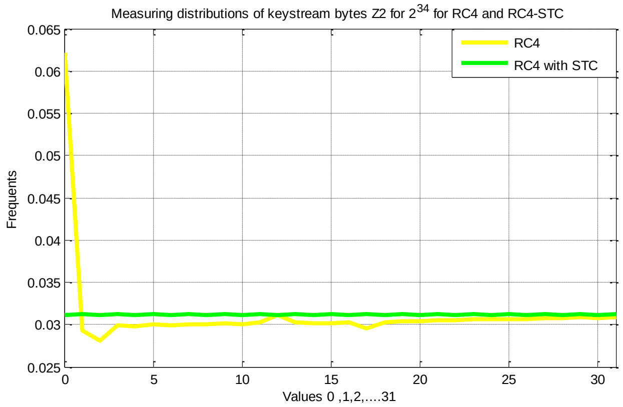

Fig. 3. Key distribution byte in the 2nd position for RC4 and RC4-STC with 2^{34}.

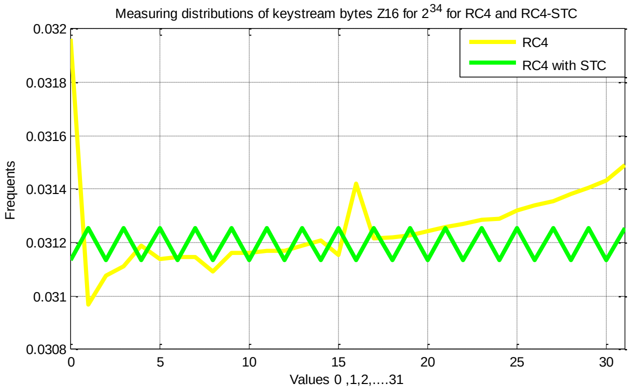

Fig. 4. Key distribution byte in the 16th position for RC4 and RC4-STC with 2^{34}.

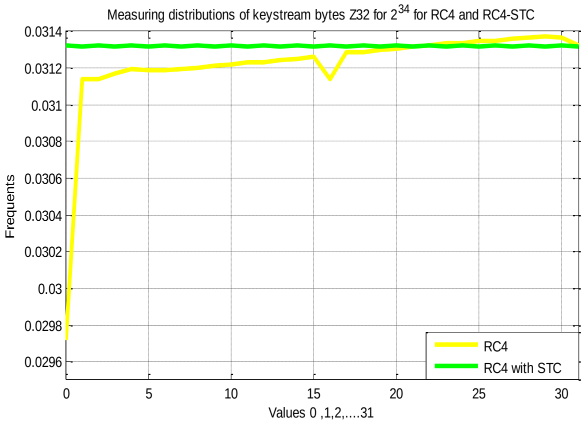

Fig. 5. Key distribution byte in the 32nd position for RC4 and RC4-STC with 2^{34}.

7. The Analysis of Rc4 and Double Rc4 Key Generation Algorithm Based on Double Byte Bias

After explaining single-byte biases that have great benefit to the cryptographic society, the attack simply can be discarded by ignoring the initial bytes. Thus, RC4 with additional configuration is still resistant to the single-byte attack. However, the researchers have studied and investigated biases beyond initial bytes and different multi-byte biases have been discovered in the key output of RC4. (Fluhrer & McGrew, 2001) [1] were the first researchers that discovered the biases in a pair of bytes (K_i, K_{i+1}) and detected long-term biases in the key. They discovered ten positive biases that their mean probability was higher than the desired value and detected two negative biases whose mean probability was lower than the desired value. Hammood, Yoshigoe, & Sagheer (2015) [6] replicated biases of Fluhrer and McGrew [1]. They found two new positive biases not mentioned by Fluhrer and McGrew.

The Fluhrer and McGrew biases and Hammood bias with 1024 keys of 16 bytes are reproduced to generate 2^{32} keystream bytes after discarding the first 1024 bytes. Each key from the 1024 keys generates 2^{32}; therefore, the whole amount of generated keys is 2^{42}. The developed double RC4 key generation algorithm didn't generate any statistical bias and its output in the range is only $\pm 2^4$ from the predicted occurrences. Algorithm 6 below determines the measuring of double byte bias. The main idea of this algorithm is to measure the appearance of the pair (Z_i, Z_{i+1}) in each position of the output of RC4.

Algorithm 6. Measuring distributions of key stream bytes (K_a, K_{a+1})

Input: K [k_1, k_2, …., k_{16}]

Output: 3-Dimentions array

```
1. a = b = a1 = q = 0
2. For (z = 1 to $2^{10}$)
        2.1. Call Algorithm 2.1: KSA
        2.2. For (R= 1 to $2^{32}$)
                2.2.1. a = (a + 1) mod N
                2.2.2. b = (b + State[a]) mod N
                2.2.3. Swap (State[a], State[b])
                2.2.4. Generated Key = State[(State[a] + State[b]) mod N]
        2.2.5. A[q] [Generated Key] [a1] = A[q] [Generated Key] [a1] + 1
                2.2.6. Deducting new key with 16 bytes from each generated key.
                2.2.7. q = Generated Key
                2.2.8. a1 = (a1 + 1) mod N
3. Return A[q] [Generated Key] [a1]
```

Figure 6 shows the distribution of (Z_r, Z_{r+1}) for all the first 32 bytes of RC4 where Z_r = i and Z_{r+1} = i to discover possible double-byte biases. Y-axis determines the frequents of each pair of values while the X-axis contains each pair of values

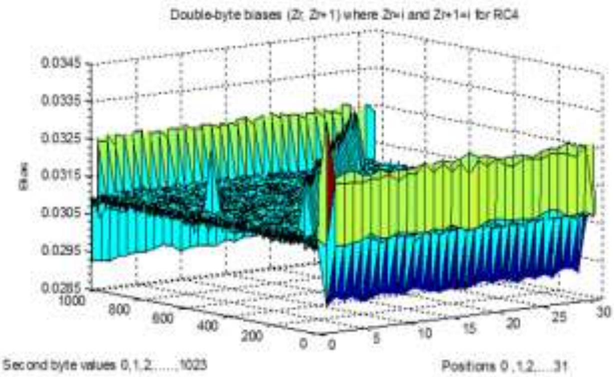

Figure 6. Double-byte biases (Z_r, Z_{r+1}) for RC4 where Z_r=i and Z_{r+1}=i.

Figure 7 shows the distribution of (Z_r, Z_{r+1}) for all the first 32 bytes of developed RC4 where Z_r = i and Z_{r+1} = i.

Y-axis determines the frequents of each pair of values while the X-axis contains each pair of values

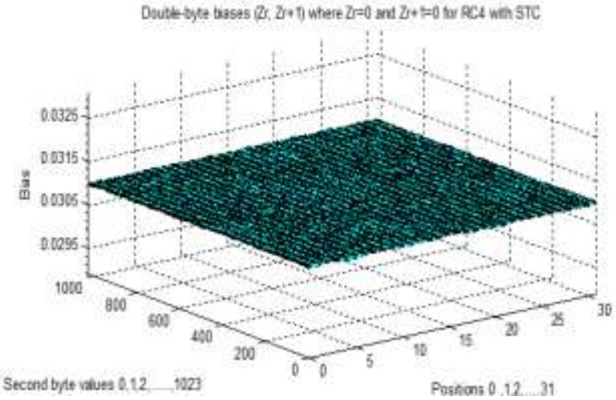

Fig. 7. Double-byte biases (Z_r, Z_{r+1}) for RC4-STC where Z_r=i and Z_{r+1}=i.

8. IMPLEMENTATION

RC4 and the proposed algorithm were implemented in C# language. The generated key stream of RC4 and modified RC4 were examined by NIST statistical test suite. NIST is the statistical combination for random numbers generator check that includes sixteen trials for measuring the output sequences randomness of pseudo random generator or true random generator of numbers as shown below. The examination of this PRNG was made by using NIST STS-1.6.

9. RESULTS AND DISCUSSION

Implementation time of the proposed algorithm is roughly the same time as that is required for RC4 implementation. The randomness and complication of developed algorithm key are higher than those of RC4 key stream. The randomness test was done by using sts-1.6 of NIST. The likelihood of perfect random number creator is determined by P-value. In the test, P-value is matched with 0.01. If the value is higher than 0.01 then the series is accepted and the obtained sequences are random and uniformly distributed, otherwise, the series is rejected and there is no randomness in the series. The SUCCESS means that the series has good randomness and is acceptable, where FAILURE indicates that the series is not random and not acceptable. In this program, a12.5 KB is generated from each secret key. This series was tested and the p-values average calculated from these examinations, as shown in the Table I.

TABLE I. RESULTS OF RUNNING NIST ON GENERATED KEY BY RC4 AND RC4-STC.

Test No.	Statistical Test Name	RC4		RC4-STC	
		P-VALUE	*Conclusion*	*P-VALUE*	*Conclusion*
1	Approximate Entropy	0.805578	Passed	0.859596	Passed
2	Block Frequency	0.742455	Passed	0.772057	Passed
3	Cumulative Sum(Forward)	0.739164	Passed	0.728672	Passed
4	Cumulative Sum (Reverse)	0.854066	Passed	0.677266	Passed
5	FFT	0.279715	Passed	0.890366	Passed
6	Frequency	0.898580	Passed	0.451117	Passed
7	Lempel-Ziv compression	0.889521	Passed	0.804724	Passed
8	Linear Complexity	0.407918	Passed	0.984154	Passed
9	Longest Runs	0.767817	Passed	0.584260	Passed
10	Non periodic Templates	0.5407084	Passed	0.516712	Passed
11	Overlapping Template	0.497550	Passed	0.786861	Passed
12	Random Excursions	0.528198	Passed	0.614838	Passed
13	Random Excursion Variant	0.525591	Passed	0.587629	Passed
14	Rank	0.610871	Passed	0.997436	Passed
15	Runs	0.115965	Passed	0.925681	Passed
16	Serial	0.646168	Passed	0.180612	Passed
17	Universal Statistical	0.380374	Passed	0.357331	Passed

The implementation time of double RC4 key generation algorithm is roughly the same time as that of RC4 implementation when implemented on the same size of secret keys, as shown in figure 8.

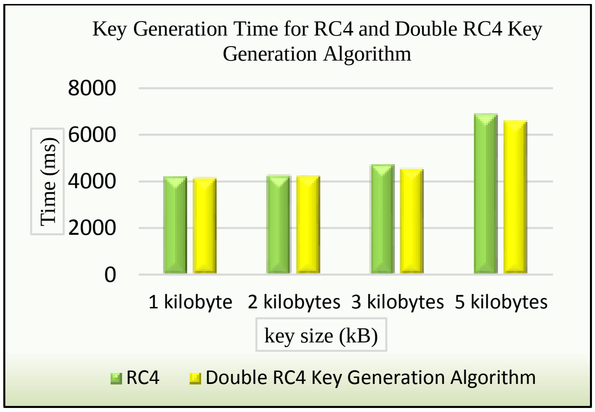

Fig. 8. The key generation time for RC4 and developed RC4.

10. CONCLUSIONS

RC4 is one of the widely used symmetric algorithms. The key generation includes a weakness in the distribution of key stream bytes that are biased to many values. In this paper, the new algorithm is proposed as an enhancement for the RC4 algorithm. The proposed algorithm idea is based on the idea of cipher block chaining. The key stream K_i depends on all preceding state bytes. The proposed algorithm implementation time is roughly the same time as that is required for RC4 implementation and is faster than RC4 when implemented on parallel processors. The proposed algorithm passed the NIST statistical tests. The proposed algorithm doesn't produce anyone of single byte biases and double byte bias while RC4 proved the same single and double bias that was shown previously. As a future work, parallel processors may be used for implementing analysis of RC4 and the proposed algorithm with more key stream bytes (256 and more).

REFERENCES

[1] S. R. Fluhrer, and D. A. McGrew, "Statistical Analysis of the Alleged RC4 Keystream Generator," Lecture Notes in Computer Science, Springer- Berlin Heidelberg, vol. 1978, pp. 19-30, 2001.

[2] B. Jindal, and B. Singh,"RC4 Encryption- A Literature Survey," Procedia Computer Science, vol. 46, pp. 697-705, 2015.

[3] K. K. Wong, G. Carter, and E. Dawson, "An Analysis of the RC4 Family of Stream Ciphers against Algebraic Attacks," In Proceedings of the EACIS. Australian Computer Journal Society, vol. 105, pp. 67-74, 2010.

[4] M. M. Hammood, K. Yoshigoe, and A. M. Sagheer, "RC4-2S: RC4 Stream Cipher with Two State Tables," Information Technology Convergence, Lecture Notes in Electrical Engineering, Springer Science Business Media Dordrecht1, pp. 13-20, DOI: 10.1007/978-94-007-6996-0_2, 2013.

[5] P. Prasithsangaree, and P. Krishnamurthy, "Analysis of Energy Consumption of RC4 and AES Algorithms in Wireless LANs," In Proceedings of Global Telecommunications Conference, IEEE, vol. 1443, no. 3, pp. 1445-1449, December 2003.

[6] M. M. Hammood, K. Yoshigoe, and A. M. Sagheer, "Enhancing Security and Speed of RC4," International Journal of Computing and Network Technology, vol. 3, no. 2, pp. 2210-1519, 2015

[7] G. Paul, Structural Weakness of the Key Scheduling of RC4. Jadavpur University: IFW 2007.

[8] L. Stosic, and M. Bogdanovic, "RC4 Stream Cipher and Possible Attacks on WEP," International Journal of Advance Computer Science Applications, vol. 3, no. 3, 2012.

[9] N. J. Al-Fardan, D. J. Bernstein, K. G. Paterson, B. Poettering, and J. C. Schuldt, "On the Security of RC4 in TLS and WPA," In Presented as Part of the 22nd USENIX Security Symposium,13, pp. 305-320, 2013.

[10] S. Maitra, and G. Paul, "Analysis of RC4 and Proposal of Additional Layers for Better Security Margin," Lecture Notes in Computer Science, International Conference on Cryptology, vol. 5365, pp. 27-39, 2008.

[11] S. S. Gupta, "Analysis and Implementation of RC4 Stream Cipher," (Doctoral Dissertation), Indian Statistical Institute Kolkata, Kolkata, West Bengal, India, 2013.

[12] L. L. Khine, A New Variant of RC4 Stream Cipher. Mandalay Technological University Mandalay 05052, Mandalay, Myanmar: World Academy of Science, Engineering and Technology, 2009.

[13] M. U. Bokhari, S. Alam, and F. S. Masoodi, "Cryptanalysis Techniques for Stream Cipher," International Journal of Computer Applications, vol. 60, pp. 0975 – 8887, 2012.

[14] I. Mantin and A. Shamir, "Practical Attack on Broadcast RC4," In Fast Software Encryption, Lecture Notes in Computer Science, Springer, vol. 2355, pp. 152-164, 2001.

[15] M. McKague, "Design and Analysis of RC4-Like Stream Ciphers," (Master Thesis), University of Waterloo, Canada, Ontario, 2005.

[16] M. Beck and E. Tews, "Practical Attacks Against WEP and WPA," In Proceedings of the Second ACM Conference on Wireless Network Security, pp. 79-86, 2008, Available at: http://dl.acm.org/citation.cfm?id=1514286.

[17] S. M. Searan, A. M. Sagheer, and M. M. Hammood, "Analyzing of RC4 Algorithm Based on Its Single and Double Byte Bias by Using New Algorithms," International Conference on Change, Innovation, Informatics and Disruptive Technology, London – UK, 11-12 OCT, 2016, Available at http://sriweb.org/londonconf/.

[18] S. Paul, and B. Preneel, "Analysis of Non Fortuitous Predictive States of the RC4 Keystream Generator," Springer Computer Science, vol. 2904, pp. 52-67, 2003.

[19] M. A. Orumiehchiha, J. Pieprzyk, E. Shakour, and R. Steinfeld, "Cryptanalysis of RC4 (n, m) Stream Cipher," In Proceedings of the 6th International Conference on Security of Information and Networks, vol. 178, pp. 165-172, 2013.

[20] S. Paul and B. Preneel, "A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher," Springer Computer Science, vol. 5, pp. 245-259, 2004.

[21] S. Mister and S. E. Tavares, "Cryptanalysis of RC4-like Ciphers," Springer Computer Science; vol. 1556, pp. 131-143, 1999.

[22] M. M. Hammood, K. Yoshigoe, and A. M. Sagheer, "RC4 Stream Cipher with a Random Initial State," Proceedings in 10th FTRA International Conference on Secure and Trust Computing, data management, and Applications, Lecture Notes in Electrical Engineering, pp. 407-415, 2013. Springer Netherlands.

[23] P. Sepehrdad, S. Vaudenay, and M. Vuagnoux, "Discovery and Exploitation of New Biases in RC4," Springer Computer Science, vol. 6544, pp. 74-91, 2010.

[24] M. Robshaw, and O. Billet, "New Stream Cipher Designs: The eSTREAM Finalists," Lecture Notes in Computer Science, Springer Berlin/Heidelberg, vol. 4986, pp. 1-6, 2008.

[25] B. Pardeep, and P. K. Pateriya, "PC 1-RC4 and PC 2-RC4 Algorithms: Pragmatic Enrichment Algorithms to Enhance RC4 Stream Cipher Algorithm," International Journal of Computer Science and Network, vol. 1, no. 3, pp. 2277-5420, 2012.

[26] A. M. S. Rahma, A. M. Sagheer, and A. A. Salih, "Development of RC4 Stream Ciphers using Boolean Functions," Journal of Economic Sciences, Baghdad University College, vol. 29, 2012.

[27] S. M. Searan, and A. M. Sagheer, "Modification of RC4 Algorithm by using Two State Tables and Initial State Factorial," International Journal of Computer Network and Information Security, vol.8, no.12, pp.1-8, DOI: 10.5815/ijcnis, 2016.

[28] M. Karahan, "New Attacks on RC4A and VMPC," (Doctoral Dissertation), bilkent university, 2015.

Ali M. Sagheer is a Professor in the Computer College at Al-Anbar University. He received his B.Sc. in Information System (2001), M.Sc. in Data Security (2004), and his Ph.D. in Computer Science (2007) from the University of Technology, Baghdad, Iraq. He is interested in the following fields; Cryptology, Information Security, Number Theory, Multimedia Compression, Image Processing, Coding Systems, and Artificial Intelligence. He has published many papers in different scientific journals.

Salih S. Salih has received his B.Sc. in Computer Science (2012) and M.Sc. in Computer Science (2016) from the College of Computer Sciences and Information Technology at University of Anbar, Baghdad, Iraq. He is interested in the following fields: Coding Systems, Database, and Data Mining.

Sura M. Searan has received her B.Sc. in Computer Science (2013) and M.Sc. in Information Security (2016) from the College of Computer Sciences and Information Technology at University of Anbar, Baghdad, Iraq. She is interested in the following fields: Cryptology, Information Security, and Coding Systems.