Developing a New Hybrid Cipher Algorithm using DNA and RC4

Rami k. Ahmed
Dept. of Computer Science
College of Science, University of Baghdad
Baghdad, Iraq

Imad J. Mohammed
Dept. of Computer Science
College of Science, University of Baghdad
Baghdad, Iraq

***Abstract*—This paper proposes a new hybrid security algorithm called RC4-DNA-Alg. It combines the symmetric stream cipher RC4 algorithm with DNA-indexing algorithm to provide secured data hiding with high complexity inside steganography framework. While RC4 represent one of the widely used algorithms in network security protocols, such as Secure Sockets Layer (SSL), a DNA cryptography considered as a modern branch of cryptography that combines the traditional cryptographic techniques with the power of the genetic material The performance evaluation of the proposed algorithm is measured based on three parameters (conditional entropy, randomness tests and encryption time). The result shows outperformance in security and distorted in hybrid Cipher compared to the native RC4.**

Keywords*—Rivest Cipher 4 (RC4); Secure Sockets Layer (SSL); Deoxyribonucleic acid (DNA); Rivest Cipher 4-Deoxyribonucleic acid-Algorithm (RC4-DNA-Alg)

I. Introduction

As a society, we are relying increasingly on rapid access and information processing. The proliferation of cheap computers and computer network increased the problem of unauthorized access and data theft. It may be contributed in the need to take into account the security engineering mathematics qualities and physical security systems to develop more efficient approaches, Cryptography is one of them. The demand for cryptographic technologies has increased because of the need to transmit information privately and secrecy to public networks that can intercept. In the past, encryption was used only by governments and military. At present, Cryptography used available to anyone. It is considered one of the most important means of maintaining the confidentiality of information, privacy and access control. It also used in the field of identity authentication and many other fields [1].

This paper is organized as follows: Section II describes some of the research work related to this study. Section III describes a brief about background of encryption and presents the main cryptography algorithms used in this study. Section IV explains in details the proposed algorithm for hybrid algorithm cryptography, Including all components and technologies involved. Section V discusses the performance analysis and the results. Section VI describes in brief a conclusion of this study.

II. Related Work

In 2013, Naser and et al. [12] developed a hybrid algorithm by combining three algorithms AES, RC4, and a serpent. The hybrid algorithm provided protection against most of the attacks using encryption and tried to ensure the confidentiality and secrecy of information. In 2015, Rafael and Antonio [13] proposed a modification for RC4 algorithm (called RC4itz) using the properties of the Spritz algorithm. RC4itz outperforms AES, RC4 and Spritz algorithms in term of performance, secrecy and randomness. In 2014 Himanshu and Vishal [15] developed a new hybrid approach using two algorithms AES and DNA. In their algorithm, the information split into two segments: one encrypted with AES (128) and other segment used DNA scenography to hide the information. In 2016, Karandeep [14] developed a new novel technique which is a double security layer algorithm. This algorithm consists of RSA algorithm and Deoxyribonucleic Acid (DNA) using cloud environment. The former, DNA used to encrypt information followed by RSA algorithm to encrypt cipher text result from DNA algorithm before storing in cloud servers. In 2011, Xue Sun et al. [10] present a new hybrid encryption algorithm to protect the instant messaging system using the AES for encryption, SHA-1 for authentication and RSA algorithms for key exchange to implement a hybrid encryption system. This was implemented over an Extensible Messaging and Presence Protocol (XMPP) based IM server and Java based clients. In 2016, Tutt and et al. [15] Proposed an efficient secure end-to-end messaging system that Consists of a combination of symmetric key cryptography (AES 256 bit) with temporary keys for individual message security and using Elliptic Curve Diffie-Hellman cryptography for key exchange and message authentication (HMAC-SHA384).

III. Background of Encryption

A. *Symmetric and Asymmetric Encryption*

Symmetric encryption also called secret key encryption where both the sender and the receiver depend on the same secret key where the sender uses the key to encrypt the message while the receiver uses the same key to decrypt the cipher text. This method is faster than the public key encryption method and less complex [3]. Some examples of Symmetric encryption: DES, 3DES, 3DES, AES and IDEA. Asymmetric encryption also called public key encryption; it uses two keys instead of single key [2]. The public key can be published while the private key kept in secret. The information encrypted using the public key and then can be decrypted using private key [3]. Asymmetric encryption works slower than symmetric encryption and sometimes preferred to encrypt the secret key for symmetric encryption over digital networks safely. Some examples of Asymmetric encryption: PGP, DSA, Deffie-Hellman and RSA. Fig. 1 illustrates the two types of encryption.

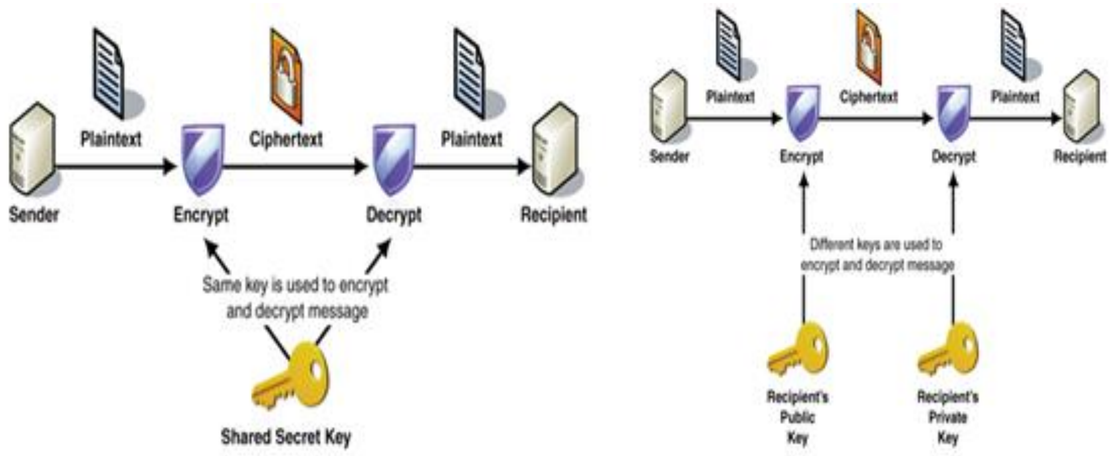


Fig. 1. Symmetric and asymmetric encryption [3].

B. Stream Cipher and RC4

Stream Cipher is a Symmetric key cipher where the plain text and the pseudo-random key (key stream) are combined by xor operation. In the encryption, each byte of plain text is encrypted with the corresponding character of the key stream. An alternate name is the cipher state, where the encryption of each character depends on the current state. RC4 no longer provides complete protection also it may generate a weak stream key as a result of user-defined weak key, but still widely used in different applications due to the ease of use as well as its speed in encryption and decryption [4], [7].

RC4 algorithm itself does not encrypt anything but generates pseudo-random values called the Key stream used for encryption using xor with the original text then decryption using xor operation with the cipher text [5]. RC4 algorithm divided into two basic phases (Algorithm 1 and Algorithm 2 in our context):

Phase 1: key-scheduling phase (KSA) as a preliminary stage that requires the following processes:

- A user-defined encryption key is specified between 0 and 255 characters (as length).
- A state Array (256 bytes) created and denoted by state. Initialized from 0 to 255 values, several substitutions performed on state array by combining the ASCII code of the encryption key with the state array cells. A new state array is produced that contains all possible cases, but arranged random [6].

The original version of RC4-KSA can be summarized in Algorithm 1.

Phase 2: Pseudo-Random Generation Algorithm PRGA. At this phase, a new state array created with Pseudo-random values [6]. When the process of generating Pseudo-Random bytes completed, the encryption process begins by executing xor operation between the key stream and the original text, the decryption process executed in the same manner by doing the xor operation between the key stream and the cipher text to produce the original text. The PRGA can be summarized in Algorithm 2.

Algorithm I. KEY SCHEDULING ALGORITHM (KSA)

```
For i = 0 to 255
   state [ i ] = i
Next i
j = 0
For i = 0 to 255
   j = ( j + state [ i ] + key [ i mod key.length ] )   mod 255
   Swap (state [ i ] , state [ j ])
Next i
call algorithm.2
```

Algorithm II. ALGORITHM 2: PSEUDO RANDOM GENERATOR ALGORITHM (PRGA)

```
i = 0
j = 0
For x = 0 to message.length - 1
    i = ( i + 1 ) mod 256
    j = ( j + state [ i ] ) mod 256
    Swap ( state [ i ] , state [ j ] )
    t = ( state [ i ] + state [ j ] ) mod 256
   // Pseudo-random values
   // Encryption step
        Cipher_text [ x ]  = ( Plain_text [ x ] xor state [ t ] ) mod 256
Next x
```

C. DNA Algorithm

Biological DNA can be used in steganography and cryptography as the storage material. Molecular computations can be performed with biological DNA structures and then applied on the classical ciphers [8]. Several projects in genome sequencing offer the possibility to exploit digital DNA databases (NCBI) for the cryptographic purposes. Deoxyribose Nucleic Acid (DNA) has a helical shape, included of 2 long strands of nucleotides. A nucleotide has one of 4 bases: A – adenine, C – cytosine, T – thymine or G – guanine. Utilizing coding method, any digital data can be transformed easily into DNA sequence using Table 1 [8].

TABLE I. DNA GENETIC CODE

Binary	DNA Chromosome
00	A
01	C
10	G
11	T

a) Principle of DNA algorithm [9]:

- Each character of text information converts to ASCII code, for example : - original message is “ramy” in ASCII will be : 114 97 109 121
- Convert ASCII code to binary sequence: 01110010 01100001 01101101 01111001
- Convert binary sequence to DNA sequence using DNA Genetic Code (Table 1): CTAG CGAC CGTC CTGC.

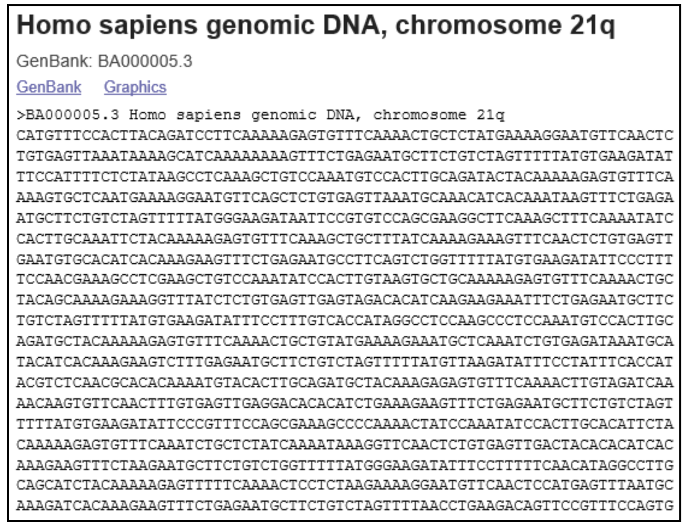
IV. PROPOSAL ALGORITHM (RC4-DNA-ALG)

It is hybrid algorithm, called RC4-DNA-Alg, a modified version of RC4 applied by adding new state called (new-state) that supports DNA indexing and consequently scenography technology.

A. DNA Indexing Algorithm

DNA Indexing is a stream cipher and symmetric algorithm that encrypts one byte at a time. The bases of DNA cryptography are to transform one byte of data to a series of four chromosomal. The next stage is to find four series out of the chromosomal series selected as a key for encryption. The chromosomal series selected from public available databases (NCBI database website) and used in the implementation of our algorithm, an instance of the selected DNA database presented in Fig. 2. DNA Indexing can regard as homophonic substitution cryptography.

The fundamental of homophonic substitution is to make an array where every Byte of the letters has a specific number of replacement values. The substitution operation of DNA Indexing cipher is simple to achieve, but the reverse process is complex without knowing the key because the distribution pattern of the cipher text is completely different from that of the plaintext [11]. Each byte of plain text has a set of values used in the replacement. The number of replacement values for a byte will depend on its appearance in the chromosome. If someone wants to reuse the same key for many encryptions, then it will be useful to transform this cipher into a homophonic one [10].


Homo sapiens genomic DNA, chromosome 21q

GenBank: BA000005.3

GenBank Graphics

```
>BA000005.3 Homo sapiens genomic DNA, chromosome 21q
CATGTTTCCACTTACAGATCCTTCAAAAAGAGTGTTTCAAAACTGCTCTATGAAAAGGAATGTTCAACTC
TGTGAGTTAAATAAAAGCATCAAAAAAAAGTTTCTGAGAATGCTTCTGTCTAGTTTTTATGTGAAGATAT
TTCCATTTTCTCTATAAGCCTCAAAGCTGTCCAAATGTCCACTTGCAGATACTACAAAAAGAGTGTTTCA
AAAGTGCTCAATGAAAAGGAATGTTCAGCTCTGTGAGTTAAATGCAAACATCACAAATAAGTTTCTGAGA
ATGCTTCTGTCTAGTTTTTATGGGAAGATAATTCCGTGTCCAGCGAAGGCTTCAAAGCTTTCAAAATATC
CACTTGCAAATTCTACAAAAAGAGTGTTTCAAAGCTGCTTTATCAAAAGAAAGTTTCAACTCTGTGAGTT
GAATGTGCACATCACAAAGAAGTTTCTGAGAATGCCTTCAGTCTGGTTTTTATGTGAAGATATTCCCTTT
TCCAACGAAAGCCTCGAAGCTGTCCAAATATCCACTTGTAAGTGCTGCAAAAAGAGTGTTTCAAAACTGC
TACAGCAAAAGAAAGGTTTATCTCTGTGAGTTGAGTAGACACATCAAGAAGAAATTTCTGAGAATGCTTC
TGTCTAGTTTTTATGTGAAGATATTTCCTTTGTCACCATAGGCCTCCAAGCCCTCCAAATGTCCACTTGC
AGATGCTACAAAAAGAGTGTTTCAAAACTGCTGTATGAAAAGAAATGCTCAAATCTGTGAGATAAATGCA
TACATCACAAAGAAGTCTTTGAGAATGCTTCTGTCTAGTTTTTATGTTAAGATATTTCCTATTTCACCAT
ACGTCTCAACGCACACAAAATGTACACTTGCAGATGCTACAAAGAGAGTGTTTCAAAACTTGTAGATCAA
AACAAGTGTTCAACTTTGTGAGTTGAGGACACACATCTGAAAGAAGTTTCTGAGAATGCTTCTGTCTAGT
TTTTATGTGAAGATATTCCCGTTTCCAGCGAAAGCCCCAAAACTATCCAAATATCCACTTGCACATTCTA
CAAAAAGAGTGTTTCAAATCTGCTCTATCAAAATAAAGGTTCAACTCTGTGAGTTGACTACACACATCAC
AAAGAAGTTTCTAAGAATGCTTCTGTCTGGTTTTTATGGGAAGATATTTCCTTTTTCAACATAGGCCTTG
CAGCATCTACAAAAAGAGTTTTTCAAAACTCCTCTAAGAAAAGGAATGTTCAACTCCATGAGTTTAATGC
AAAGATCACAAAGAAGTTTCTGAGAATGCTTCTGTCTAGTTTTAACCTGAAGACAGTTCCGTTTCCAGTG
```

Fig. 2. Chromosomal sequences from a genetic database (NCBI).

B. DNA Indexing Process

Encryption process can be summarized in the following pseudo code 1:

Step 1: Key dictionary calculation:

- Convert every Byte of plain text into DNA series, for example:- Current state byte=114=01110010 ➔ 11=T, 01=C, 00=A, 10=G ➔ 114=01110010=CTAG
- A search performed for every 4 chromosomal DNA out of the NCBI chromosomal series, the index of that position saved in the key dictionary.
- Finally, each byte of plaintext may have multi indexes in the key dictionary, for example (Table 2):

TABLE II. KEY DICTIONARY OF GENETIC DATABASE (NCBI)

DNA	Key indexing
AAAA	221, 1036, 5002, 32654 ….
AAAT	12, 566, 9354 ….
AAAG	856, 6549, 22354 ….
AAAC	23, 66, 647, 6985, 63745 ….
......	
CCCC	478, 6324, 26583 ….

Step 2: The encryption process performed by replacing one byte of plain text with a value from the key dictionary as illustrated in Table 2. For example, the substitutions or the pattern (CTAG) could be replaced with any one of (65, 3154, 4687, 9637, 13586, 25697 or 36548).

Step 3: The final output of the encryption process consists of key index as integer value, for example: Current new-state byte selected from the substitutions = 36548.

Fig. 3 depicts the block diagram of the proposed algorithm (RC4-DNA-Alg).

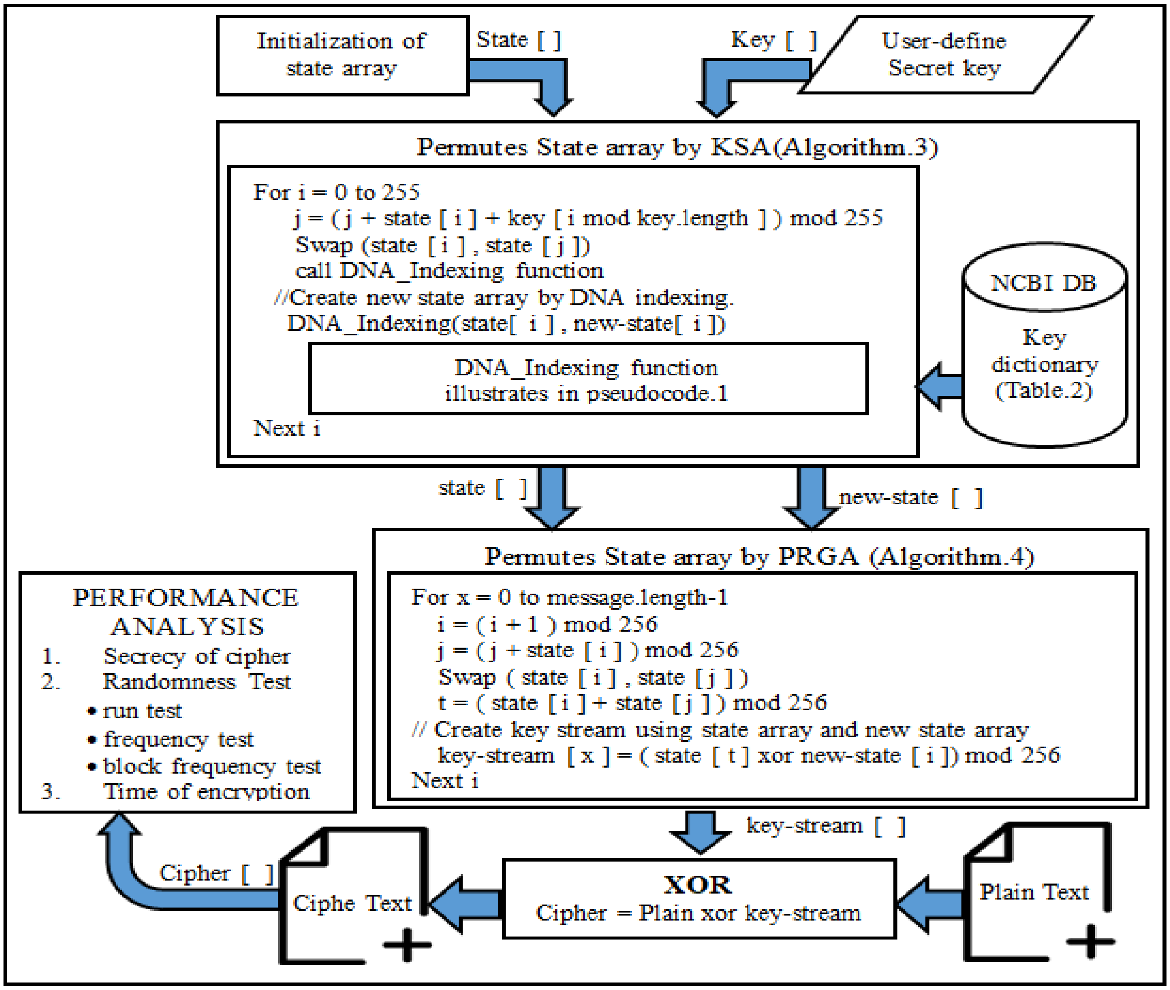


Fig. 3. Block diagram of the proposed algorithm (RC4-DNA-Alg).

In our proposed design, the modified KSA can be summarized in Algorithm 3.

Another modification suggested improving the original PRGA as appears in Algorithm 4.

The decryption process uses the same chromosomal sequence of the encryption process as part of symmetry criteria.

- Every integer value of cipher text used as an index into the key dictionary.
- Retrieve the four chromosomes from the indicated index of key dictionary.
- Every Byte of plain text reconstructed by convert every four chromosome into a binary then converts it to ASCII code (of plain text).

Algorithm III. MODIFIED KEY SCHEDULING ALGORITHM (KSA)

```
For i = 0 to 255
   state [ i ] = i
Next i
j = 0
For i = 0 to 255
   j = ( j + state [i] + key [ i mod key.length ] ) mod 255
  Swap ( state [ i ] , state [ j ])
//added step to the original KSA (illustrates in pseudocode.1).
  DNA_Indexing( state [ i ] , new-state [ i ] )
Next i
call Algorithm.4
```

Algorithm IV. THE MODIFIED PSEUDO RANDOM GENERATOR ALGORITHM (PRGA)

```
i = 0
j = 0
For x = 0 to message.length-1
   i = ( i + 1 ) mod 256
   j = ( j + state [ i ] ) mod 256
   Swap ( state [ i ] , state [ j ] )
   t = ( state [ i ] + state [ j ] ) mod 256
//modified step of original PRGA algorithm by adding a new state.
   key-stream [ x ] = ( state [ t ] xor new-state [ i ] ) mod 256
  Cipher_text [ x ] = ( Plain_text [ x ] xor key-stream [ x ] ) mod
256
Next x
```

V. Performance Analysis

This section introduces the performance analysis of RC4-DNA-Alg based on three main parameters; Cipher Secrecy, Randomness, and Entropy. In this paper, the same inputs are used to examine the original and the modified version of RC4 algorithm in terms of Average secrecy, Randomness and encryption time using 100 random plaintexts for of variable length (128,256,512 and 1024 byte) and 100 random key of variable length (32,64,128 and 256 byte).

A. Secrecy of Cipher

Measurement is entropy. It represents the amount of information exist in a random variable, the exchanged information, and the amount of information shared between two random variables. The key equivocation H (K|C) defined as "the amount of information about the key used that is revealed by the cipher text observed" As shown in the following equation:

$$H(K \mid C) = \sum_{j=1}^{l} \sum_{i=1}^{n} P_k(j) * P_{k,c}(i,j) \log_2 P_{k,c}(i,j)$$

Where, H denotes entropy, K for key, C for cipher text, P for probability, l for length key, n for length of cipher text.

Table 3 shows RC4 compared to RC4-DNA-Alg on average. It is observed that RC4-DNA-Alg provides more secrecy compared to RC4.

TABLE III. The Proposed RC4-DNA-ALG Compared to RC4 based on Average Secrecy

Plaintext Size	Key Size	Average Secrecy Value	
		RC4	RC4-DNA-Alg
128	32	0.32	0.422
	64	0.68	0.789
	128	0.907	0.998
	256	0.875	0.984
256	32	0.321	0.424
	64	0.583	0.691
	128	0.773	0.983
	256	0.809	0.99
512	32	0.351	0.424
	64	0.626	0.793
	128	0.825	0.986
	256	0.891	0.991
1024	32	0.361	0.425
	64	0.686	0.793
	128	0.815	0.986
	256	0.862	0.993
Wins / total test		0/16	16/16

TABLE IV. The Proposed RC4-DNA-ALG Compared to RC4 based on Average Randomness

Plaintext	Key Size	Run Test		Frequency (Monobit)		Frequency Block	
		RC4	RC4-DNA-Alg	RC4	RC4-DNA-Alg	RC4	RC4-DNA-Alg
128	32	0.417	0.502	0.491	0.488	0.399	0.503
	64	0.419	0.504	0.418	0.492	0.408	0.503
	128	0.502	0.5	0.497	0.496	0.452	0.502
	256	0.404	0.498	0.49	0.49	0.473	0.507
256	32	0.42	0.502	0.429	0.495	0.401	0.502
	64	0.503	0.499	0.495	0.49	0.41	0.502
	128	0.435	0.499	0.498	0.491	0.5	0.499
	256	0.496	0.505	0.441	0.493	0.468	0.504
512	32	0.397	0.497	0.42	0.5	0.449	0.504
	64	0.414	0.502	0.406	0.497	0.503	0.5
	128	0.393	0.5	0.454	0.502	0.46	0.507
	256	0.399	0.501	0.501	0.498	0.478	0.507
1024	32	0.396	0.5	0.502	0.5	0.42	0.505
	64	0.504	0.497	0.501	0.494	0.453	0.507
	128	0.405	0.501	0.498	0.497	0.471	0.507
	256	0.5	0.497	0.436	0.5	0.454	0.509
Wins / total test		4\16	12\16	9\16	7\16	2\16	14\16

B. Randomness Test

Randomness Test is a set of algorithms that find out whether the distribution of data is random. In this paper, three algorithms of Randomness (run test, frequency test and block frequency test) are implemented to evaluate the randomness of cipher text (Table 4). It can be observed that RC4-DNA-Alg provides more randomness values compared to native RC4 (shadow color).

C. Time of Encryption

Encryption time (Table 5) has been tested on a machine using CPU Intel core i5 2.4 GHz and RAM 8 GB under Microsoft windows 10 enterprise 64-bit.

Table 6 shows the result summary of encryption time.

TABLE V. THE PROPOSED RC4-DNA-ALG COMPARED TO RC4 BASED ON AVERAGE ENCRYPTION TIME

Plaintext Size	Key Size	Average Encryption Time (ms)	
		RC4	RC4-DNA-Alg
128	32	0.401	0.522
	64	0.402	0.529
	128	0.41	0.538
	256	0.414	0.584
256	32	0.554	0.624
	64	0.562	0.641
	128	0.567	0.681
	256	0.576	0.69
512	32	0.836	0.924
	64	0.843	0.993
	128	0.848	1.086
	256	0.857	1.091
1024	32	1.385	1.425
	64	1.435	1.793
	128	1.593	1.986
	256	1.612	1.993
Wins / total test		16/16	0/16

TABLE VI. SUMMARY OF RESULTS

Parameters / algorithms	RC4	RC4-DNA-Alg
	win	win
Secrecy of cipher	0 %	100 %
Run Test	25 %	75 %
Frequency (Monobit)	56.25 %	43.75 %
Frequency Block	12.5 %	87.5 %
Average encryption time (in ms)	0.832	1.007 (17.5 % overhead)

VI. CONCLUSIONS

Current research introduced a new hybrid algorithm (RC4-DNA-Alg) by combining RC4 and DNA algorithms. It is validated that RC4-DNA-Alg enhanced the secrecy level of the cipher text for all tested sizes of plaintext using four different key sizes (32, 64, 128 and 256 Bytes). Furthermore, got more stable randomness test than native RC4. The enhancement steps added an overhead around (17.50%) to the execution time of original RC4. In future work will focus on developing a key exchange algorithm in order to implement the instant messaging system based on this study.

REFERENCES

[1] Gampala, Veerraju, Srilakshmi Inuganti, and Satish Muppidi. "Data security in cloud computing with elliptic curve cryptography." International Journal of Soft Computing and Engineering (IJSCE) 2.3 (2012): 138-141.
[2] Schneier, Bruce. Applied cryptography: protocols, algorithms, and source code in C. john wiley & sons, 2007.
[3] Menezes, Alfred J., Paul C. Van Oorschot, and Scott A. Vanstone. Handbook of applied cryptography. CRC press, 1996.
[4] Imai, Hideki. "Wireless Communications Security. Artech House." Inc., Norwood, MA, USA 3 (2005).
[5] Mironov, Ilya. "random shuffles of RC4." Advances in Cryptology—CRYPTO 2002 (2002): 304-319.
[6] Lemke, Kerstin, Christof Paar, and Marko Wolf. Embedded security in cars. Springer-Verlag Berlin Heidelberg, 2006.
[7] Mousa, Allam, and Ahmad Hamad. "Evaluation of the RC4 algorithm for data encryption." IJCSA 3.2 (2006): 44-56.
[8] Borda, Monica, and Olga Tornea. "DNA secret writing Techniques." Communications (COMM), 2010 8th International Conference on. IEEE, (2010): 451-456.
[9] Soni, Er Ranu, Er Vishakha Soni, and Er Sandeep Kumar Mathariya. "Innovative field of cryptography: DNA cryptography." International Conference on Information Technology Convergence and Services. 2012.
[10] Tornea, Olga, et al. "Encryption system with Indexing DNA chromosomes cryptographic algorithm." IASTED International Conference. Vol. 680. No. 99. (2010): 12-15.
[11] Madhulika, Gadang, and Chinta Seshadri Rao. "Generating digital signature using DNA coding." Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014. Springer, Cham, (2015): 21-28.
[12] Aghajanzadeh, Naser, Fatemeh Aghajanzadeh, and Hamid Reza Kargar. "Developing a new hybrid cipher using AES, RC4 and SERPENT for encryption and Decryption." International Journal of Computer Applications 69.8 (2013).
[13] Álvarez, Rafael, and Antonio Zamora. "An Intermediate Approach to Spritz and RC4." International Joint Conference. Springer, Cham, 2015.
[14] Kaur, Karandeep. "A Double Layer Encryption Algorithm based on DNA and RSA for Security on Cloud." (2016).
[15] Chaudhary, Himanshu, and Vishal Bhatnagar. "Hybrid approach for secure communication of data using chemical DNA." Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference-. IEEE, (2014): 967-971.