DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

SNOW3G Modified by using PLL Algorithms in Magic Cube

Rana Mohammed Zaki[1], Hala Bahjat Abdul wahab[2]
[1,2]Computer Sciences Department, University of Technology, Baghdad, Iraq
[1]Rana.M.Zaki@uotechnology.edu.iq,[2]hala.b.abdulwahab@uotechnology.edu.iq

Abstract— *Thomas and Patrik are working on a stream cipher called SNOW 3G. In 2006, it was chosen as the centerpiece of a new set of confidentiality and integrity algorithms for the Universal Mobile Telecommunications System (UMTS). In 2008, Böhm published an article named "Statistical Evaluation of Stream Cipher SNOW 3G." He put the randomness of the SNOW 3G key stream generator to the test. As a randomness test tool, Böhm uses the NIST test statistics package, which consists of three kinds of tests: lengthy key stream data, short key stream data, and initialization vector data. Only the short key stream set of data failed eight random chance test results out of three kinds of tests, according to the report's findings. The SNOW 3G suggestions, he claims, fail due to a flaw with in cipher's initialization. In this paper, we use the PLL algorithm to modify the SNOW 3G algorithm for key initialization and generation keystream. We employ the same itself Böhm employed. The modify SNOW 3G algorithm exceed whole of the statistic exam in a experiment. The findings show that PLL has an effect on algorithm randomness.*

Index Terms— *SNOW-3G, Stream Cipher, PLL Algorithms.*

I. INTRODUCTION

The Snow3G algorithm is one of the LTE algorithms used in mobile networks. It is a developed version of the SNOW 2.0 algorithm which was subjected to several forced attacks that led to its breaking [1]. So both Thomas and Patrik developed the algorithm in January 2006. ETSI/SAGE stands for European Telecommunications Standards Institute/Security Algorithms Group of Experts are evaluation SNOW3G and Three Generation Partnership Project (3GPP) is allocated as part of the service for confidentiality and integrity algorithms [2]. Böhm [3] investigated the SNOW 3G key stream generator's randomisation properties in 2008. Böhm used the NIST test statistics suite for randomness testing with a significance level of 0.01 [4] according to the results, the short key failed the test.

In this paper, we used Permutation of Last Layer (PLL) algorithm to modify SNOW3G to create a dynamic S-Box, not static, as in the traditional snow3G algorithm. The benefit of a dynamic S-Box is to increase the complexity of keystream generation in addition to increasing the randomness with less time during the generation of the S-Box. When used the statistical test NIST the modified SNOW3G algorithm succeeded.

II. SNOW3G ARCHITECTURE

The SNOW3G is stream cipher algorithm and the first kernel algorithm for the 128-EEA1 and 128-EIA1 LTE network systems [5-7].SNOW3G is generate key-stream has

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

length 32 bits called word by using Cipher Key (CK) and Initial Vector (IV) Each has a length of 128 bits, it is continent two layers: first layer is LFSR Divided into 16 stages, each stage has 32 bits)S0, S1,……..S14,S15). Second layer is FSM have two component three registers (R1, R2 ,R3) and two substitution boxes (S-Box). Where ⨁ represent XOR and ⊞ represent adder to arithmetic operator [4, 8-10]. Given that the standard SNOW-3G lacks robustness toward short keystream set of data attacks, We suggest enhancing their statistical properties and security level through overcoming them flaws by modified the S-Box that updates the initialize and keystream mechanisms whilst still maintaining the standard as shown in *Fig. 1*.

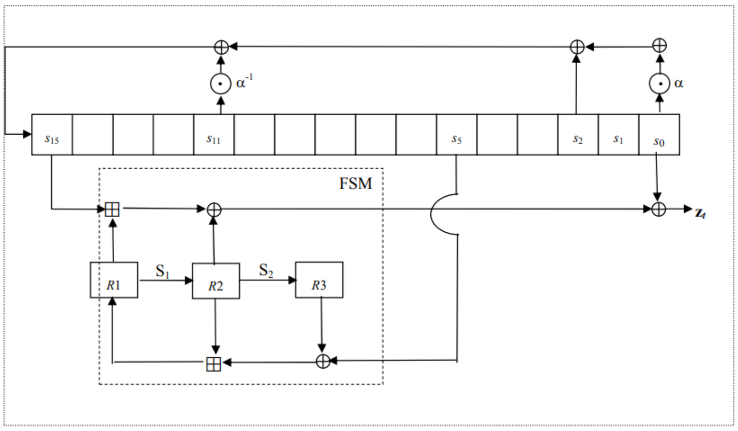


FIG. 1. SNOW 3G STANDARDS [4].

III. SNOW3G MODIFIED BY USING PLL ALQRITHM

Initially, we'll go over how to modify the SNOW 3G stream cipher by adding two new s-boxes that were created using PLL algorithms:

A. PLL Algorithms (Permutation of Last Layer)

It is an advanced technology of the Magic Cube which was Jessica Fridrich's progress entails memorizing a large number of algorithms, but there is a logical connection between them. The Petrus system and the Fridrich method (or full CFOP), which are used by the vast majority of speed cubes these days, must be mentioned when discussing advanced Rubik's Cube solving techniques. The advanced technique used by Jessica Fridrich divides the puzzle into layers. and that you must help in solving the cube layer after layer, using algorithms for each step, without destroying the pieces that are already in place as shown in *Fig. 2* [11-14].

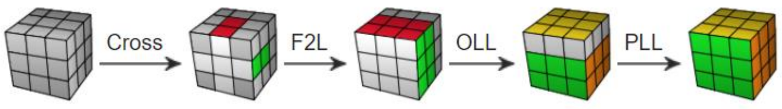


FIG. 2. CROSS, F2L, OLL AND PLL (CFOP) STRUCTURE [11].

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

B. Proposed Substitution Box (S-Box)

This article describes a new method for building S-Boxes that uses PLL algorithms to modification the positioning (permutation) on the pieces in the last layer without rotary them. After that, the cube would be solved. This step is split into two sections. As well:

– Permutation of edges: It must permute the edges, which means changing their position but not orientation. It is necessary to consider the number of well-positioned edges.

– Permutation of corners: Once the edges have been properly placed, only the corner remains to be solved.

To build S-Box 16 × 16 uses PLL algorithms represented by 3x4 as show in *Fig. 3*.

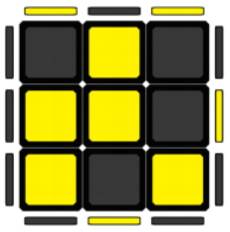


FIG. 3. A SPECIFIC EXAMPLE OF LAST-LAYER PREPARATION.

After each movement, the border limits will be checked. of position in orderly to save the output within the wanted, values [0, 255]. The permutation of position to guarantee a satisfying ownership with the aid a 256-value sequence with every item in the S-box is checked from start to finish prevent a certain item from being iterative in the S-Box and ensuring that a values are distributed within a large randomness with 21 algorithms to produce novel S-Box as shows in *Fig. 4*.

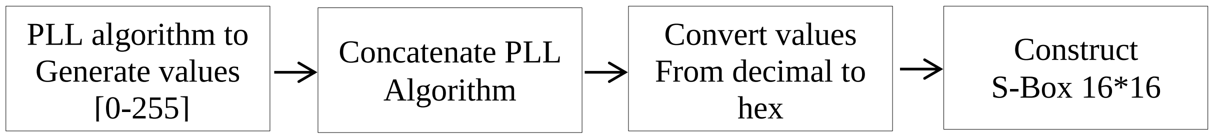

PLL algorithm to Generate values [0-255]
Concatenate PLL Algorithm
Convert values From decimal to hex
Construct S-Box 16*16

FIG. 4. PROPOSED CONSTRUCT NEW S-BOX.

The first and second stages in the above scheme use an algorithm PLL. It depends on solving the problem of different colors in the last layer of the cube by switching locations according to a certain movement and a certain direction as shown in algorithm 1.

Algorithm 1: Creating decimal code

1. **Input**: PLL Algorithm
2. **Output:** Decimal codes

Begin

3. Create a number sequence using PLL) Tperm, Yperm, Hperm, Vperm ,Fperm ,Eperm,A1perm,A2perm,R1perm,R2perm,G1perm,G2perm,G3perm,G4perm,N1perm,N2perm,U1perm ,U2perm ,J1perm ,J2perm ,Zperm ,Sperm)
4. Concatenate [Tperm, Yperm, Hperm ,Vperm ,Fperm, Eperm, A1perm, A2perm,R1perm,R2perm,G1perm,G2perm,G3perm,G4perm,N1perm,N2perm,U1pe

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

rm ,U2perm,J1perm,J2perm,Zperm,Sperm]and save the outcomes in ns11//ns11 is a collection of one domination with decimal values called array

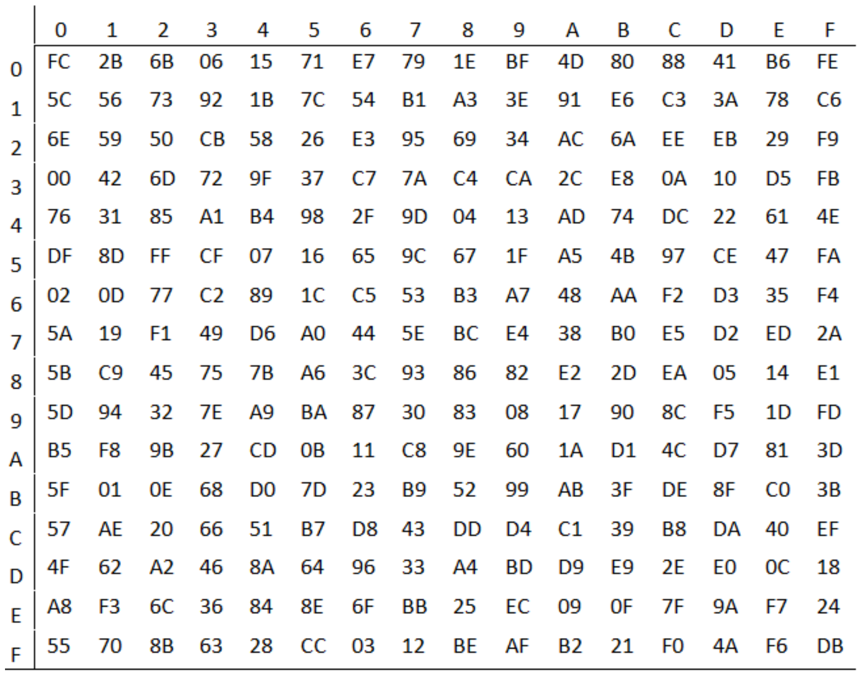
End

After creating the decimal code, it will be converted to hex code in order to create a new S-Box table for use in the encrypt process. S-Box building is represented by Algorithm 2. Table I shows the results.

Algorithm 2: Create S-Box (16*16)

```
1. Input: decimal code
2. Output : Novel S-Box contain (16 Row and 16 Column)
Begin
3. Convert decimal code to hex code by using dec2hex(ns11)
4. Reshape from matrix one diminution [0-255] to [16*16] and save the results in dx // dx this array of two domination with hex values.
4.1 For I =1 to 16
5. delete any space from dx to construct new S-Box (16×16) and store the result in dm
End
```

TABLE I. S-BOX (SR1) PRODUCED BY THE PLL ALGORITHM



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	FC	2B	6B	06	15	71	E7	79	1E	BF	4D	80	88	41	B6	FE
1	5C	56	73	92	1B	7C	54	B1	A3	3E	91	E6	C3	3A	78	C6
2	6E	59	50	CB	58	26	E3	95	69	34	AC	6A	EE	EB	29	F9
3	00	42	6D	72	9F	37	C7	7A	C4	CA	2C	E8	0A	10	D5	FB
4	76	31	85	A1	B4	98	2F	9D	04	13	AD	74	DC	22	61	4E
5	DF	8D	FF	CF	07	16	65	9C	67	1F	A5	4B	97	CE	47	FA
6	02	0D	77	C2	89	1C	C5	53	B3	A7	48	AA	F2	D3	35	F4
7	5A	19	F1	49	D6	A0	44	5E	BC	E4	38	B0	E5	D2	ED	2A
8	5B	C9	45	75	7B	A6	3C	93	86	82	E2	2D	EA	05	14	E1
9	5D	94	32	7E	A9	BA	87	30	83	08	17	90	8C	F5	1D	FD
A	B5	F8	9B	27	CD	0B	11	C8	9E	60	1A	D1	4C	D7	81	3D
B	5F	01	0E	68	D0	7D	23	B9	52	99	AB	3F	DE	8F	C0	3B
C	57	AE	20	66	51	B7	D8	43	DD	D4	C1	39	B8	DA	40	EF
D	4F	62	A2	46	8A	64	96	33	A4	BD	D9	E9	2E	E0	0C	18
E	A8	F3	6C	36	84	8E	6F	BB	25	EC	09	0F	7F	9A	F7	24
F	55	70	8B	63	28	CC	03	12	BE	AF	B2	21	F0	4A	F6	DB

When we permutated of locations, we will get a completely different S-Box from the first since the algorithm has the ability to create 2^{21} S-Boxes as shown in Table II.

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

TABLE II. S-BOX (SR2) PRODUCED BY THE PLL ALGORITHM

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	06	0B	03	04	05	09	0A	02	01	0C	08	07	0D	11	0F	10
1	0E	12	13	17	15	16	14	18	1F	1A	1B	1C	23	22	21	1D
2	19	24	20	1E	2B	2C	27	28	29	30	25	26	2D	2E	2F	2A
3	31	38	39	3A	35	36	37	32	33	34	3B	3C	3D	41	3F	40
4	3E	46	43	44	45	42	47	48	49	4A	51	52	50	4E	4F	4D
5	4B	4C	53	54	5B	56	55	60	59	58	57	5C	5D	5E	5F	5A
6	61	68	63	64	6B	66	67	62	69	6A	65	6C	75	74	6D	78
7	71	72	73	77	6F	70	6E	76	79	7A	7B	7C	7D	82	81	83
8	7F	7E	80	84	85	86	8B	8A	89	8E	8D	8C	87	88	8F	90
9	99	92	97	96	95	94	93	98	91	9C	9B	9A	9D	A7	9F	A0
A	9E	A2	A3	A4	A5	A6	A1	A8	AB	AA	A9	B4	B3	AE	AF	B0
B	B1	B2	AD	AC	B5	B9	B7	B8	BF	BA	BB	BC	BD	BE	B6	C0
C	C1	C2	C7	C6	CB	C4	C3	C8	C9	CA	C5	CC	CD	D7	CE	D0
D	D1	D6	D5	D4	D3	D2	CF	D8	D9	DD	DF	DE	DA	DC	DB	E0
E	E1	E2	E3	E4	ED	E9	E5	F0	EF	EA	EB	EC	E7	E8	E6	EE
F	F1	F2	FA	F9	F5	F6	F7	FB	F4	F3	F8	FC	FF	00	FD	FE

C. SNOW3G Modify

In key initialization, the modified SNOW 3G stream cipher algorithm consists of 16 stages, each stage consisting of 32 bits and represented in $F(2^{16})$. LFSR (r0, r1, r2,..., r14, r15), the polynomial equation for LFSR is:

$$R(x) = x^{16} + x^{12} + x^{4} + x^{3} + x^{1} + 1 \qquad (1)$$

32-bit registers R1, R2, and R3, and two new S-Boxes SR1 and SR2 that are used to update registers R2 and R3.as shown *Fig. 5* and According to the following equation

$$V=((r_{0,1}||r_{0,2}||r_{0,3}||0x00)\oplus MUL\alpha(r_{0,0})\oplus(r_2)\oplus(0x00||r_{11,0}||r_{11,1}||r_{11,2})\oplus DIV\alpha(r_{11,3})\oplus F \oplus(r_3)) \qquad (2)$$

In *Fig. 6* the key stream generator mode scheme of Modified SNOW 3G.The structure of FSM is modified by using the permutation algorithm to design two different boxes, where the two boxes were created using the PLL algorihms to produce an output F. According to the following equation to the keystream generation process.

$$Z_t = F + r_0 \qquad (3)$$

When generating the keystream, there is no entry from FSM to LFSR .That is, there is no F, so the equation V will be as follows:

$$V=((r_{0,1}||r_{0,2}||r_{0,3}||0x00)\oplus MUL\alpha(r_{0,0})\oplus(r_2)\oplus(0x00||r_{11,0}||r_{11,1}||r_{11,2})\oplus DIV\alpha(r_{11,3})) \qquad (4)$$

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

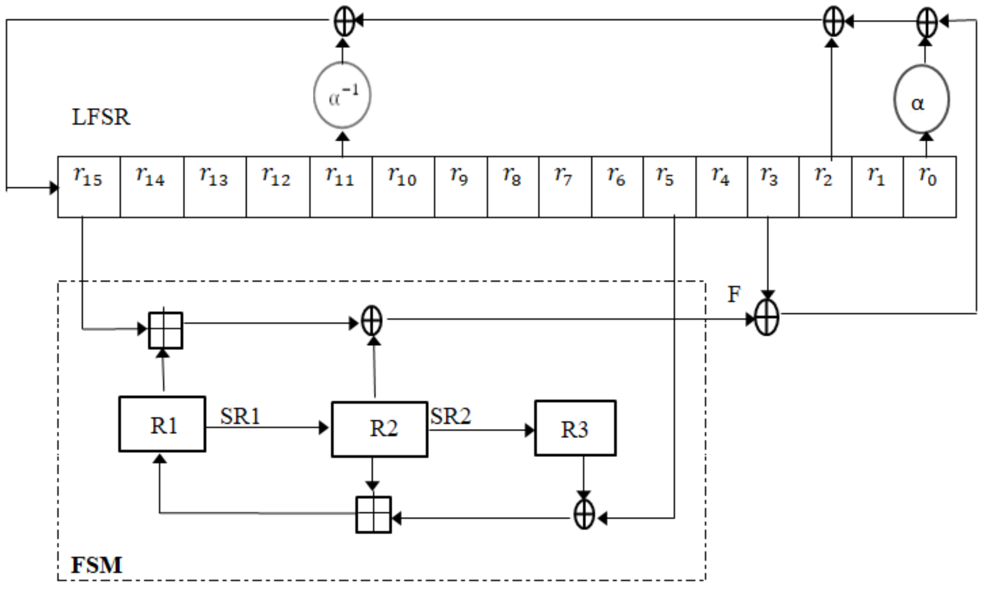

LFSR
α^{-1}
α
r_{15} r_{14} r_{13} r_{12} r_{11} r_{10} r_9 r_8 r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0
F
R1
SR1
R2
SR2
R3
FSM

FIG. 5. KEY INITIALIZATION IN MODIFIED SNOW 3G.

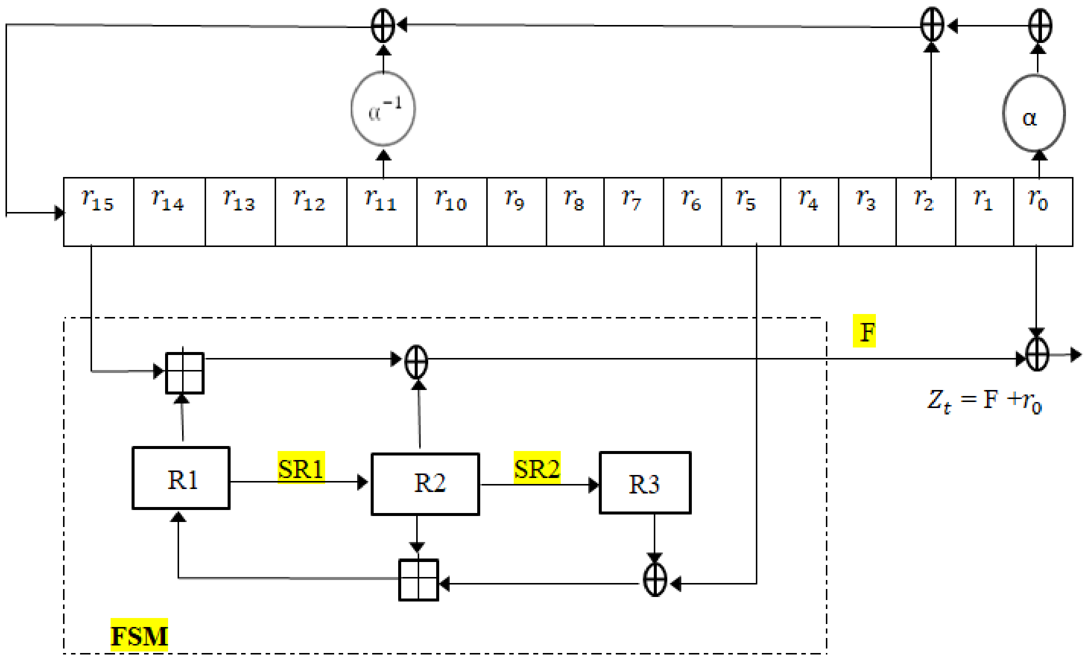

α^{-1}
α
r_{15} r_{14} r_{13} r_{12} r_{11} r_{10} r_9 r_8 r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0
F
Z_t = F +r_0
R1
SR1
R2
SR2
R3
FSM

FIG. 6. GENERATION KEYSTREAM IN IN MODIFIED SNOW 3G.

IV. RESEARCH METHOD

Our proposed stream cipher plan is subjected to statistical analyses in order to demonstrate its robustness, such as randomness. NIST Test (National Institute of Standards and Technology) has a number of statistical trails that can be applied to assess the randomness of the series, including NIST [15-17], DIEH. [18] and NESS. [19]. The SNOW3G modified stream cipher is evaluated using NIST statistical tests. We iterate the 3 statistical trials in [20] to evaluate a pass of the produced keystreams to demonstrate the importance of the proposed design.

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

We used the same sample size sequences and the same weighted importance level applied in the NIST trial to compare the results with previous work. As a result, the significance level in all of the samples was fixed at 300 sequences, and the size of the experiment was reduced to 0.01. The following are the definitions of the tests:

a. Long keystream set of data test: To obtain 300 sequences of 1048576 bits, generate bits for the entire 300 random key (put the initial vector to zero in all cases). The result sequences are then evaluated using the 15 NIST tests.
b. Short keystream set of data test: Generate bits for each of the 307200 random keys (in all cases, set the initial vector to zero).The concatenation of all the bit keystreams yields 300 sequences with a total length of 1048576 bits. The result sequences are then evaluated using the 15 NIST tests.
c. The initial vector set of data starts by generating 256 (4096bits) sequences for 300 random keys. It's worth noting that each key's first sequence is generated with IV put to zero. It is then increased to the next 255 sequences. As a result, we have 300 sequences totaling 1048576 bits for NIST's 15 tests to evaluate.

There are two accesses to interpret the NIST exam score, the ratio of sequences that exceed in to a test statistic while allocation of P-value for examination for solitude. The acceptable ratio range is defined using a confidence interval defined as:

$$\widehat{\mathrm{P}} \mp 3\sqrt{\frac{\widehat{\mathrm{P}}\,(1-\widehat{\mathrm{P}})}{\mathrm{S}}} \tag{5}$$

Let α = 0.01 $\hat{P}$ =1 - 0.01 S=300

If the ratio does not fall within this range, the data is not random.

The uniformity of P-value is determined using the X^2 goodness of fit test, which involves computing:

$$X_{test}^{2} = \sum_{i=1}^{t} \frac{O_{i-E_i}}{E_i} \tag{6}$$

The number of levels is represented by l, the key stream was tested to see if it was distributed uniformly. With a total of 300 keystreams, a minimum threshold ratio value of 0.97 is acceptable.

V. EXPERIMENTAL RESULTS

When we implemented the SNOW3G modified algorithm, since uniformly distributed P-values are , a keystream is random and the P-values ratio is accepted as shown in Table III. Finally, the proposed SNOW3G algorithm was compared with the SNOW3G traditional algorithm in Table IV.

TABLE III. P-VALUES RESULTS BY NIST-TESTS

No. Test	NIST test	Standard SNOW3G Long	Standard SNOW3G Short	Standard SNOW3G IV	Modified SNOW3G Long	Modified SNOW3G Short	Modified SNOW3G IV
1	Monobit-test(frequency)	0.032203	0.911413	0.561227	0.501541	0.501481	0.501525
2	Frequency-within-block-test	0.314042	0.678686	0.574903	0.500141	0.500120	0.500131

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

3	Runs-test	0.883171	0.514124	0.015598	0.498093	0.498132	0.498142
4	Longest-run-ones-in-block-test	0.602458	0.462245	0.961593	0.487863	0.487918	0.487936
5	Binary-matrix-rank-test	0.746572	0.955835	0.630178	0.394870	0.527847	0.392232
6	discrete Fourier transform –test	0.678686	0.113151	0.540878	0.477568	0.477502	0.477515
7	Non-overlapping-template-matching-test	0.271167	0.345115	0.840081	0.448190	0.449863	0.440863
8	Overlapping-template-matching-test	0.262249	0.413032	0.616305	0.015570	0.750224	0.322039
9	Maurer-universal-test	0.547637	0.986515	0.249284	0.994031	0.993296	0.993690
10	Linear-complexity-test	0.706149	0.981767	0.245072	0.483309	0.270994	0.394591
11	Serial-test	0.989179	0.753185	0.70614	0.500713	0.500744	0.500708
12	Approximate-entropy-test	0.772760	0.739918	0.726503	0.495514	0.495523	0.495525
13	Cumulative-sums-test	0.003996	0.198690	0.020548	0.520670	0.520636	0.520671
14	Random-excursion-test	0.745791	0.231703	0.772760	0.599390	0.435679	0.452234
15	Random-excursion-variant-test	0.891010	0.904349	0.683283	0.374495	0.397919	0.397007

TABLE IV. COMPARED THE SNOW3G TRADITIONAL[21] WITH THE SNOW3G MODIFIED ALGORITHM USING NIST-TESTS

No. Test	NIST test	Standard SNOW3G Long	Standard SNOW3G Short	Standard SNOW3G IV	Modified SNOW3G Long	Modified SNOW3G Short	Modified SNOW3G IV
1	Monobit-test(frequency)	Pass	pass	pass	pass	pass	pass
2	Frequency-within-block-test	Pass	pass	pass	pass	pass	pass
3	Runs-test	Pass	fail	pass	pass	pass	pass
4	Longest-run-ones-in-block-test	Pass	fail	pass	pass	pass	pass
5	Binary-matrix-rank-test	Pass	fail	pass	pass	pass	pass
6	discrete fourier transform -test	Pass	fail	pass	pass	pass	pass
7	Non-overlapping-template-matching-test	Pass	fail	pass	pass	pass	pass
8	Overlapping-template-matching-test	Pass	fail	pass	pass	pass	pass

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

9	Maurer-universal-test	Pass	pass	pass	pass	pass	pass
10	Linear-complexity-test	Pass	pass	pass	pass	pass	pass
11	Serial-test	Pass	fail	pass	pass	pass	pass
12	Approximate-entropy-test	Pass	fail	pass	pass	pass	pass
13	Cumulative-sums-test	Pass	pass	pass	pass	pass	pass
14	Random-excursion-test	Pass	pass	pass	pass	pass	pass
15	Random-excursion-variant-test	Pass	pass	pass	pass	pass	pass

VI. CONCLUSIONS

The proposed SNOW3G modified algorithm is based on the basic structure of the traditional SNOW3G algorithm, but it has been updated to meet the latest hardware requirements. As well as offering the ideal combination of high security, high performance, and limited hardware resources.

The main goal of using permutation last layer algorithms is to generate more random S-boxes in order to generate a keystream that is more powerful and complex, thus increasing the cipher's complexity. In addition, Modified SNOW 3G's keystream exceed all of the tests for the long key stream data set, short key stream data set, and initialization vector data set successfully.

REFERENCES

[1] A. G. Sulaiman and I. F. Al Shaikhli, "Comparative study on 4G/LTE cryptographic algorithms based on different factors," International Journal of Computer Science and Telecommunications, vol. 5, no. 7, pp. 7-10, 2014.

[2] R. M. Zaki and H. B. A. Wahab, "4G Network Security Algorithms: Overview," International Journal of Interactive Mobile Technologies, vol. 15, no. 16, 2021.

[3] P. Bohm, "Statistical evaluation of stream cipher snow 3G," in Constantin Brancusi University of Targu Jiu Engineering Faculty Scientific Conference with international participation, 2008, pp. 363-366.

[4] P. Ekdahl and T. Johansson, "A new version of the stream cipher SNOW," in International Workshop on Selected Areas in Cryptography, 2002: Springer, pp. 47-61.

[5] H. Bahjat and M. Ali, "Improvement Majority Function in A5/1 stream cipher Algorithm," Engineering and Technology Journal, vol. 34, no. 1 Part (B) Scientific, 2016.

[6] H. B. A. Wahab and M. A. Mohammed, "Improvement A5/1 encryption algorithm based on sponge techniques," in 2015 World Congress on Information Technology and Computer Applications (WCITCA), 2015: IEEE, pp. 1-5.

[7] H. M. Al-Mashhadi, H. B. Abdul-Wahab, and R. F. Hassan, "Data security protocol for wireless sensor network using chaotic map," International Journal of Computer Science and Information Security, vol. 13, no. 8, p. 80, 2015.

[8] S. Sahmoud, W. Elmasry, and S. Abudalfa, "Enhancement the Security of AES Against Modern Attacks by Using Variable Key Block Cipher," Int. Arab. J. e Technol., vol. 3, no. 1, pp. 17-26, 2013.

[9] P. Kitsos, G. Selimis, and O. Koufopavlou, "High performance ASIC implementation of the SNOW 3G stream cipher," IFIP/IEEE VLSI-SOC, pp. 13-15, 2008.

[10] S. Traboulsi, M. Sbeiti, F. Bruns, S. Hessel, and A. Bilgic, "An optimized parallel and energy-efficient implementation of SNOW 3G for LTE mobile devices," in 2010 IEEE 12th International Conference on Communication Technology, 2010: IEEE, pp. 535-538.

[11] S. Pochmann, "Analyzing Human Solving Methods for Rubik’s Cube and similar Puzzles," 2008.

[12] I. M. ALattar and A. M. S. Rahma, "A New Block Cipher Algorithm Using Magic Square of Order Five and Galois Field Arithmetic with Dynamic Size Block," International Journal of Interactive Mobile Technologies, vol. 15, no. 16, 2021.

[13] A. Y. Rahardian, R. Munir, and H. Harlili, "Rubikstega: A Novel Noiseless Steganography Method in Rubik's Cube," in International Conference on Information Technology, Engineering, Science & its Applications, 2018.

Received 31/October/2021; Accepted 22/December/2021

DOI: https://doi.org/10.33103/uot.ijccce.22.3.13

[14] V. Dan, G. Harja, and I. Naşcu, "Advanced Rubik's Cube Algorithmic Solver," in 2021 7th International Conference on Automation, Robotics and Applications (ICARA), 2021: IEEE, pp. 90-94.
[15] L. E. Bassham III et al., "Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications," ed: National Institute of Standards & Technology, 2010.
[16] R. M. Zaki, T. W. Khairi, and A. E. Ali, "Secure Data Sharing Based on Linear Congruetial Method in Cloud Computing," in Next Generation of Internet of Things: Springer, 2021, pp. 129-140.
[17] H. B. A. Wahab and T. A. Jaber, "Using Chebyshev Polynomial and Quadratic Bezier Curve for Secure Information Exchange," Engineering and Technology Journal, vol. 34, no. 5 Part (B) Scientific, 2016.
[18] G. Marsaglia and W. W. Tsang, "The 64-bit universal RNG," Statistics & Probability Letters, vol. 66, no. 2, pp. 183-187, 2004.
[19] B. Preneel, "New European schemes for signature, integrity and encryption (NESSIE): a status report," in International Workshop on Public Key Cryptography, 2002: Springer, pp. 297-309.
[20] G. Orhanou, S. E. Hajji, Y. Bentaleb, and J. Laassiri, "EPS confidentiality and integrity mechanisms algorithmic approach," arXiv preprint arXiv:1102.5191, 2011.
[21] A. Kircanski and A. M. Youssef, "On the sliding property of SNOW 3 G and SNOW 2.0," IET Information Security, vol. 5, no. 4, pp. 199-206, 2011.

Received 31/October/2021; Accepted 22/December/2021