Periodicals of Engineering and Natural Sciences ISSN 2303-4521
Vol. 8, No. 4, December 2020, pp.2379-2388

Modify Speck-SHA3 (SSHA) for data integrity in WoT networking based on 4-D chaotic system

Haider K. Hoomod [1], Jolan Rokan Naif[2], Israa S. Ahmed[2]
[1] 1 Computer Science Department, College of Education, Mustanisiryah University
[2] Informatic Institute for Postgraduate Studies, Iraqi Commission for Computers and Informatic

ABSTRACT

WoT has become one of the important fields that accompanied the development of technology in the present day, so that its income in all areas of work, including (industrial, agricultural, medical, industrial, etc.). This proliferation generated fears among users of the growing use of WoT without providing safe ways to preserve the information generated by its devices. So, many ways to maintain the security of data as well as integrity, through the use of light weight speed algorithms to encrypt and to validate the parameters that may be used in this area. In the field of information authentication, many algorithms have emerged that help to ensure the authenticity of information generated when sent from physical sensors to the user environment. These include SHA-1, SHA-2, SHA-3, etc. In WoT, the speed of encryption and authentication must be an important requirement and a duty to ensure the validity of the information and to continue monitoring the data. Therefore, providing fast algorithms is an important requirement to be provided with WoT. In this paper, a modification of the SHA-3 algorithm will be introduced by replacing the KECCAK function with another very fast algorithm which is SPECK which is produced a very fast algorithm with a reliable strong level of security in the data validation process produced by sensors. Also, the extended logistic system will be used to generate the initial values that the SHA3 algorithm uses to make these values unknown and which the intruder cannot guess or recognize.

Keywords: SHA-3, Keccak, SPECK, Chaotic System, WoT Integrity

Corresponding Author:
Haider Kadhim Hoomod
Computer Science Department, College of Education, Mustansiriyah University
Baghdad, Iraq.
Email:drhjnew@gmail.com

1. Introduction

In the past years there has been a great interest in the use of the Internet of Things in everyday life. Millions of people are using the Internet of Things in very different ways. It is estimated that by 2020, the number of devices connected to the Internet will be estimated to be between 50 to 100 billion devices [1].

This growing use of WoT has led users to start worrying about the future in terms of information security and how to maintain the privacy and integrity of data when transferred from the source to the final interface [2]. Therefore, many users have quit using the Internet of Things for this reason. Accordingly, there have been many techniques and suggestions for maintaining data integrity

So, Jolan Rokan Naif Al-Khazraji and et al was used modified SHA-3 to maintain data integrity in the proposed IOT system. This is done through the use of chaos keys and use within SHA-3. This is done through two modifications in the algorithm. The first is to rotate the data block to the left by 3 and make XOR with a selected chaos key. Second, it uses the SHA-256 algorithm with a select key switch to produce the KecHash used in the Keccak function [3].

Arlen Baker used SHA as a solution to information validation problems summarized by using two SHA functions and a shared key to produce hash message authentication code (HMAC) [4].

Martin Schlater et al., they have presented a study on the threats that IoT users may face and gave ways in which data security and integrity can be maintained. Through, the use of hashtags or signatures between parties, the Internet of Things to preserve the integrity of data from manipulation or modification of its contents [5].

In this paper, a quick authentication mechanism will be proposed through which it can be ensured the information that may be received from the WoT system used is not forged and its contents have not been modified. This is done, using the modified SHA-3 algorithm with the SPECK algorithm (SSHA) with the possibility of using an excessive chaos system, to provide a faster SHA-3 algorithm than the original algorithm in order to provide quick validation of information that may be gained from the system.

1.1. Secure hash function

Secure Hash Algorithm was created and developed by (NIST) which was used to encrypt or validate files to resist attacks that may occur on data in these files. Which was characterized by the introduction of different values of data to produce us numerical values as well as fixed size whatever the length of the data entered is different [6]. So, these algorithms were used as a criterion to determine the validity of data sent from the source to the final interface. Therefore, they agree on a type of data encrypted using one of the secure hashing algorithms agreed between the two parties to ensure the integrity of the data that may be sent and received between different systems [7].
So, many hashing algorithms have emerged that have helped to maintain data integrity, including RipeMD, Tiger, xxhash and more, but the most common hash types used in file safety tests are MD5, SHA-2, SHA-3, and CRC32.

1.2. Secure hash algorithm 3 (SHA-3)

In 2007, NIST opened a public competition to develop the hash algorithm in new ways, to propose a new method of hashing under the name SHA-3. The objective of this competition is to increase the number of Hash algorithms of the Federal Information Processing Standard (FIPS). In 2008 NIST received sixty-one researches to develop the retail function Wu, the competition began and qualified for the second round 14 competitors in 2009, after which qualified 5 finalists in 2010 to progress to the last round, and the nominated teams are (BLAKE, Grøstl, JH and Keccak and Skein). After the five candidates were submitted for the final round of the competition, their work was submitted for evaluation for 18 months to finally be selected Keccak in 2012, to be announced to NIST that the Keccak algorithm is the winner of the SHA-3 contest [8]. The reasons for choosing the KECCAK algorithm are the winning algorithm is for the following
1. The Keccak can receive the greatest resistance to attacks that may occur on encrypted text, with the possibility of preserving the text from the sixth session and above. This is because its output was broken at just five out of 24 rounds due to a near-collision attack. This indicates that Keccak will remain safe in the future.
2. Keccak relies on hardware-oriented design for simple bit-oriented operations, which is quite different from the SHA-2 mechanism.
3. Keccak offers acceptable performance in software and good hardware, which shows that this algorithm has the ability to preserve text much more than other algorithms.

Keccak relied on a new method of producing fragmentation. This technique is called a sponge, which relies on the creation of a sponge on a large random method that allows input of a certain amount of data in a way called (absorption), and then outputs the data in a way called compression, which can act as a quasi-random function of all previous inputs. This leads to high flexibility in its work in producing a stream of encrypted texts [9].
However, Keccak lacks only one thing, it needs a higher speed device to produce coded texts. So, several algorithms have been proposed to ensure speed and here in this paper a new alternative to the Keccak algorithm will be proposed to ensure high speed in data encryption and security level near or higher [10].
As mentioned above, the SHA-3 algorithm uses a sponge structure, so that it absorbs data as much as possible and then outputs the result by compressing the output.
In the absorption phase, the message is transformed into a group of blocks, and these blocks are converted into sub-groups of the state. Later to the function f.
In the compression phase, blocks produced from the absorption phase are read directly, for modification to the symbol (R), which can be read and written, while the untouched part is called the input and output (symbol C).
As shown in Figure 1, the scheme of the SHA-3 algorithm works

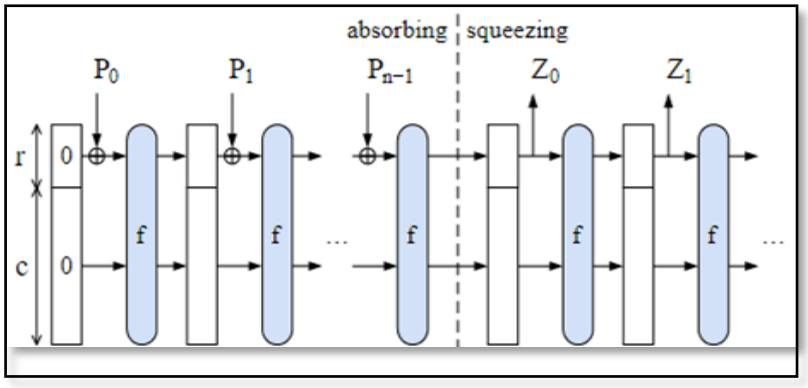

Figure 1. SHA-3 structure [3]

The flipping function f works on the mass of data that is determined by the width b, R is based on the initial values set before the start of the coding process, or the power is extracted by c = b – r. The sponge structure Z = sponge [f, cushion, r] (N, d), which produces a series of bits in Z with a length of d, operates as follows:

1. The clear text is passed to the pad function, to produce a series of bits of P that accept a division by R so that the results of dividing P by R are an integer.
2. P is distributed into a set of chunks and sequentially P0, ..., Pn - 1
3. Initialize the initial state of S so that all its contents are zero
4. It started with the process of absorbing the input from STATE: for each Pi produced:
1. Pi is extended by a series of c which is replaced by zero
2. XOR work with the case S which consists of a matrix of size (5 * 5)
3. The host f is applied to the result, which produces a new string of state S
4. Z initializes it to be the empty string
5. While the length of Z is less than d:
6. Append the first r bit from S to Z
7. If Z is still less than d bits, use f as S, giving a new state S
8. Truncate Z to d bits

The SHA-3 algorithms of sizes (224, 256, 384, 512) the size of r is greater than d, so there is no need for additional mass variations at the compression stage. However, SHAKE-128 and SHAKE-256 provide additional output length, which is useful in applications such as optimal padding for asymmetric encryption.

2. Keccak [r, c]

The function of the sponge is based on the use of the Keccak-f stirring function, which in turn depends on 7 types of permutations, which are scaled according to the needs of the environment. These ranges from light weight to very high mass. These permutations can be limited to the following: B {25, 50, 100, 200, 400, 800, 1600} [9]. For example, when taking two examples of the Keccak-f flipping functions, which may use light and high relays, we find that:

Instance SHA-3: r = 1088 and c = 512

- permutation width: 1600
- security strength 256: After enough quantity

Lightweight instance: r = 40 and c = 160

- permutation width: 200
- security strength 80: The same (initially expected from) SHA-1

The state: an array of) 5 * 5 *2 (bits

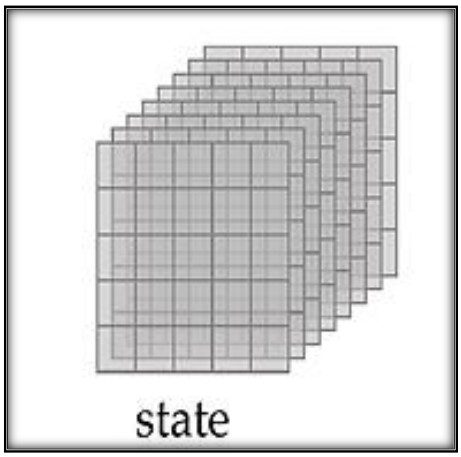

Figure 2. State Block in Keccak Function [10]

So, the state that is relied upon in the sponge process, will consist of three dimensions, these dimensions are dealt with by filling the blocks sequentially. Fill five columns and five rows and then fill the entire slide. Continue this way until the mass is fully filled. The filling process that follows the sponge can be summarized as follows

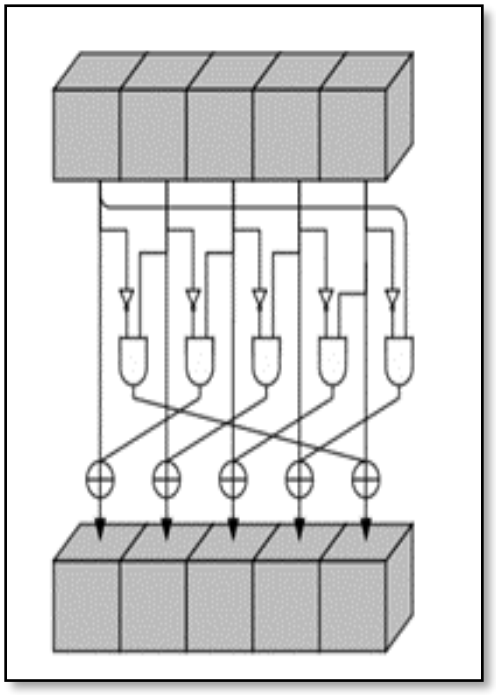

Figure 3. Block diagram shows the nonlinear mapping in Keccak-f [10]

As shown in Figure 3, keccak-f operates in a non-linear fashion so that this method can be utilized in its bit-flipping capability if the model contains bit 01, and also works independently and in parallel with the 5 rows parallel to it [9].

3. SPECK

Speck is a family of lightweight block ciphers publicly released by the National Security Agency (NSA) in June 2013.[11] Speck has been optimized for performance in software implementations, while its sister algorithm, Simon, has been optimized for hardware implementations. Speck is an add–rotate–xor (ARX) cipher.[12]

5.1 . Cipher description

Speck supports a variety of block and key sizes. A block is always two words, but the words may be 16, 24, 32, 48 or 64 bits in size. The corresponding key is 2, 3 or 4 words. The round function consists of two rotations, adding the right word to the left word, xoring the key into the left word, then xoring the left word into the right word. The number of rounds depends on the parameters selected [11]

5.2 . Round function

SPECK uses 3 basic operations on n-bit word for each round:

– bitwise XOR, $\oplus$,
– addition modulo 2n,
– left and right circular shifts by r2 and r1 bits, respectively.

Left half n-bit word is denoted by $X_{r-1,L}$ and right half n-bit word is denoted by $X_{r-1,R}$ to the r-th round and n-bit round key applied in the r-th round is denoted by kr. $X_{r,L}$ and $X_{r,R}$ denotes output words from round r which are computed as in Eq (1) and (2) :

$$x_{r,L} = \left(\left(x_{r-1,L} \gg r1 \right) \boxplus x_{r-1,L} \oplus k_r \right) \quad (1)$$

$$x_{r,R} = \left(\left(x_{r-1,R} \ll r2 \right) \oplus x_{r,L} \right) \quad (2)$$

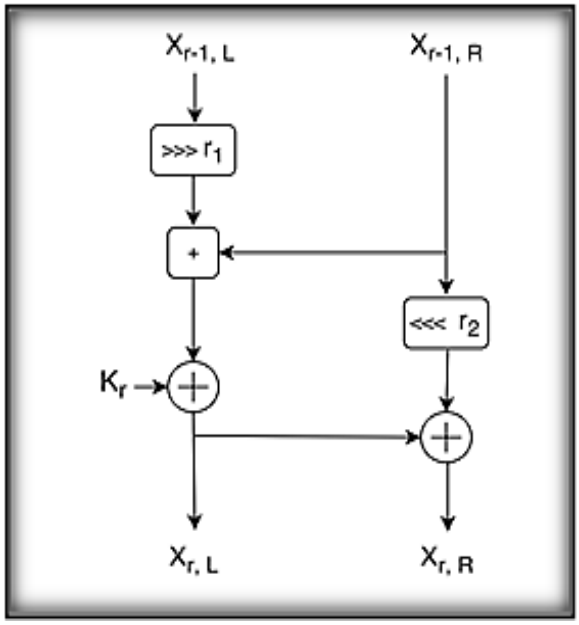

Figure 4. The round function of SPECK [14]

Different key sizes have been used by several instances of the SPECK family and the total number of rounds depends on the key size. The value of rotation constant r1 and r2 are specified as: r1 = 7, r2 = 2 for SPECK32 and r1 = 8, r2 = 3 for all other variants. Parameters for all variants represented in Table 1[15].

Table1. SPECK parameters [15]

Variant	Block Size(2n)	Word Size(n)	Key Size	Round
SPECK32	32	16	64	22
SPECK48	48	24	72	22
			96	23
SPECK64	64	32	96	26
			128	29
SPECK96	96	48	96	28
			144	29
SPECK128	128	64	128	32
			192	33
			256	34

4. 4-D chaotic system

The chaotic generate chaos keys were including two parts (A and B) and named Extended Logistic Chaotic System (EL). Both parts consist of two stage for generate random chaos keys by apply chaotic system. The first stage applies 2D-logistic-sine chaotic map (2D-LASM) to take the output it as initials value for second stage to make it complex and hard to predictable.

The second stage was modified logistic map and named 2D-modified logistic map (2D-ML) based on two equations as shown in equations (3) and (4) with two control parameters and initial values are fed from first stage to generate chaos keys as shown in figure (5) [16].

$$x_{n+1} = b * x_n(1 - x_n)(1 - x_n^2) \quad (3)\ [16]$$

$$y_{n+1} = a * y_n(1 - y_n)(1 - y_n^2) \quad (4)[16]$$

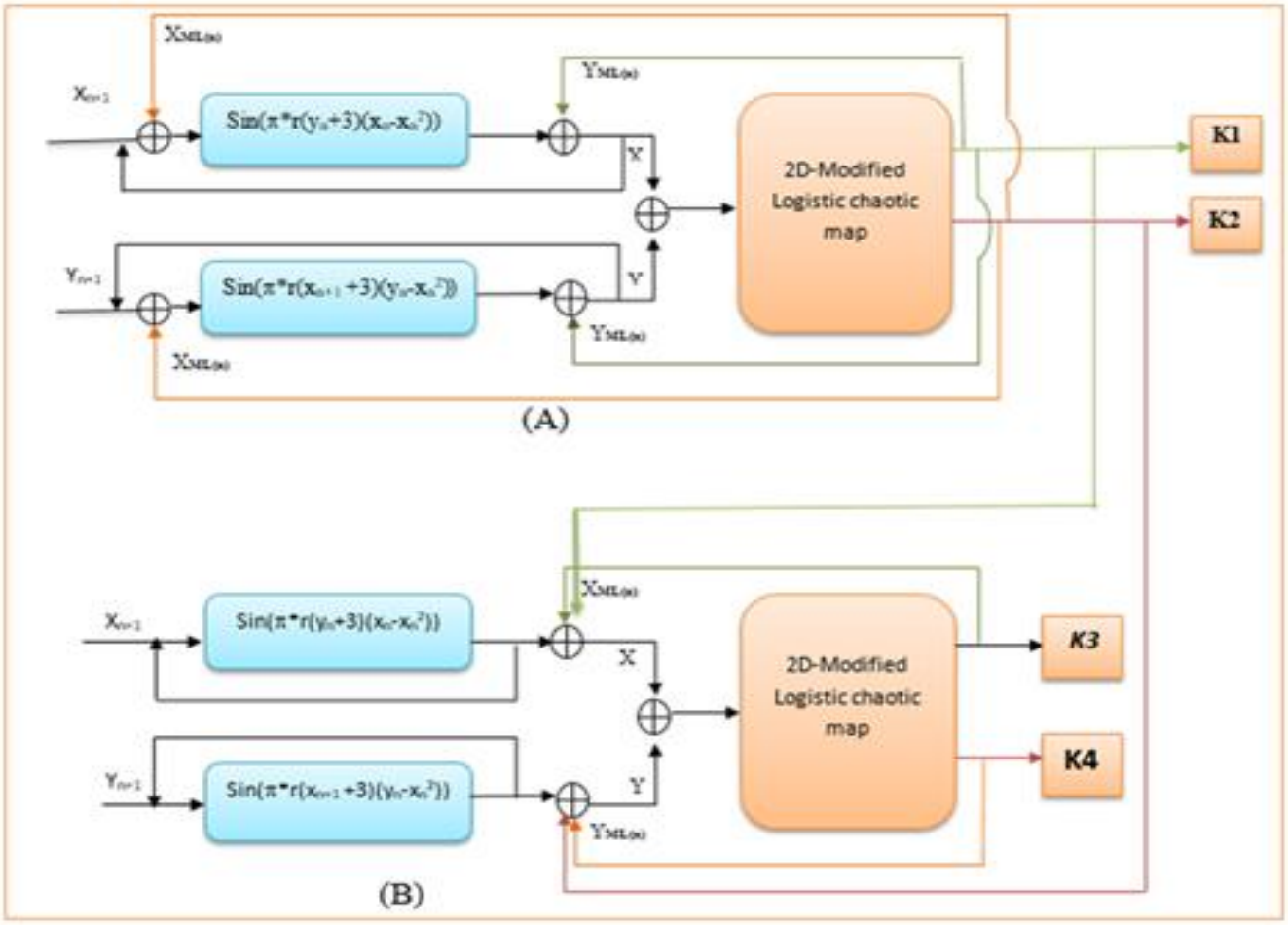

Figure 5. The 4-D chaotic system that generates the Chaos Keys [16]

5. The Proposed method

As mentioned above, the SHA-3 algorithm lacks the speed needed to achieve fragmentation at constant lengths, this disadvantage makes the process of using SHA-3 to validate information in the Web of Things slow down the work of the Web of Things, because the algorithms that are used must be fast and light by converting data from one situation to another. So, in this work, a new way of making the SHA-3 algorithm is proposed so that the algorithm works more quickly and effectively. Also change the initial values of r and c from constant values, to values that track the type of data entered. Values are generated by a proposed system of chaos, which has been conducted a series of tests to ensure the resulting chaos. Figure 6 shows the modified SHA-3.

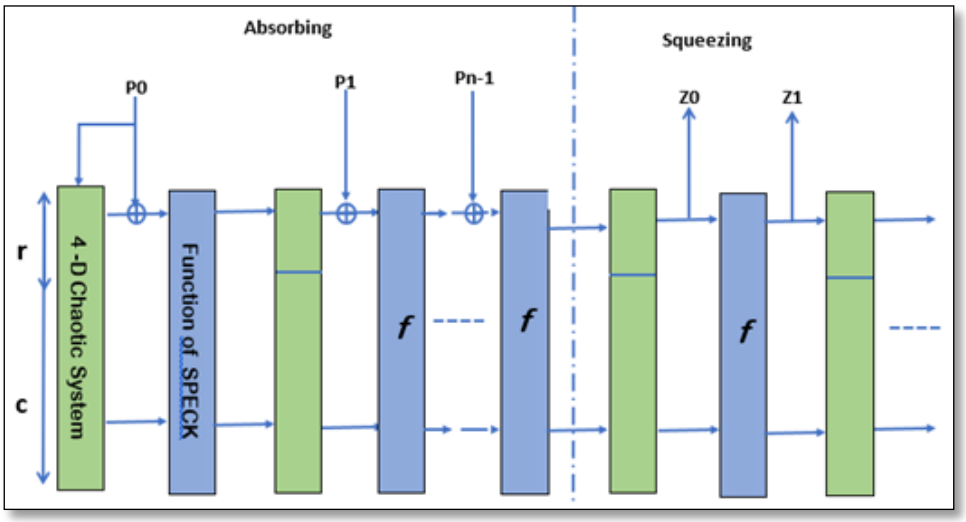

Figure 6. Block diagram show SPECK-SHA-3(SSHA)

The main function that was replaced is by replacing all the equations found in the absorption and compression phases. In the proposed system we have used the equations that are used in the SPECK algorithm with simple modifications to be suitable for use in the core of the SHA-3 algorithm.
As for the initial values of (r, c), these values are generated by the use of 4-D chaotic system based on the initial values of the data entered into the SSHA algorithm before being padded, to produce a more secure system It was suggested. So that the equations in (1) are used in the production of random values that produce two different arrays of values for (r, c).
As mentioned earlier, the state block consists of a 3D block consisting of (5 * 5 * 2), this block contains the data that is absorbed by the 24 cycles that were performed on the same algorithm to fill the state with a set of data and then compressed to produce kechash. Here in this paper the size of the state block of the situation is quite different from the size of the block in the former and it consists of (5 * 5) and this state block does not need much time to fill the state of data during the process of absorption. This gives them flexibility and speed in moving between and filling the cells of this state.
This is due to the way in which it was implemented by relying on the SPECK algorithm as well as the significant downsizing of the block size of the situation that accompanied the change in the function of keccak and Non-linear mapping in SSHA as shown in Figure 7.

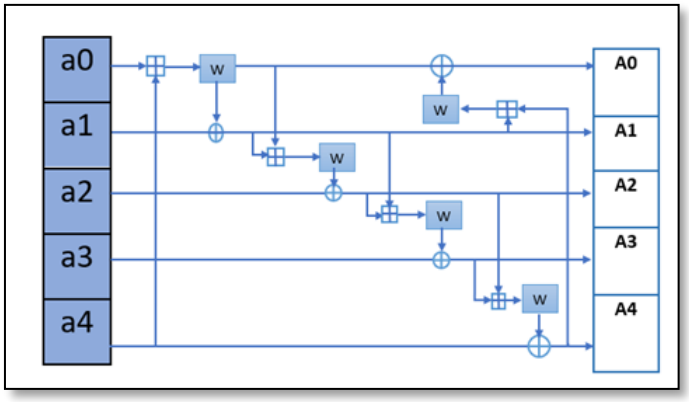

Figure 7. Block diagram shows the nonlinear mapping in SSHA

6. Results and discussion

The proposed SSHA algorithm was implemented using a messy system to randomly increase the primitive masses that are working with the modified procedures of the SSHA algorithm that have been tested by several tests including test results of the data after extracting the value of SSHA and whose results are shown in Table (2), and also the tests of NIST whose results are shown in Table (3). Table (4) shows the results of the HASH value from the proposed SSHA algorithm and compares it with the traditional SHA-3 algorithm. Table (4) illustrates the test results after modifying one letter of the text, and also comparing the results between the extracted results between SSHA and the original SHA-3.
Table 2 presented results compared between the original algorithm SHA-3 and the modified SSHA algorithm, were created, the first used to verify data information at a high speed So the time it takes to extract the hash result when applied to Sasha takes up to half the time algorithm much less than the original SHA-3. This indicates that the modified SSHA algorithm in terms of speed of implementation as well as in terms of productivity obtained after modifications.

Table 2. The Hashing Time comparison

File Size	SSHA (24 round) (sec)	SHA-3 (sec)
128B	0.001	0.002
1KB	0.014	0.023
10KB	0.17	0.284
100KB	0.92	1.41
1MB	6.49	11.66

Table 3 shows that when applying any text to the proposed SSHA algorithm with different data changes between numbers and letters, the modified algorithm added more confusion to the clear text with the possibility of verifying the data after applied to it and this indicates that the modified SSHA algorithm contains more security features and provides better data distribution density when compared to the original algorithm. Also, if one character is replaced by plain text, this will affect the change of all output of the SSHA algorithm.

Table 3. The NIST tests results comparison

NIST statistical tests Results Name	SSHA (24 round)	SSHA (12 round)
Frequency (Mono bit) test	1.987	1.874
Runs test	5.561	5.210
Discrete Fourier transform	0.778	0.611
Block frequency	0.987	0.977
Longest runs test	0.965	0.951
Cumulative sums test	0.898	0.884
Serial test	4.662	4.507
Matrix rank test	1.782	1.310
Overlapping template test	0.997	0.971
Linear complexity test	2.876	2.854
Non overlapping template test	1.309	1.165
Random excursions variant test	1.023	1.001
Random excursions test	1.264	1.122

From the results obtained, the modified algorithm was tested in two cases can be applied according to the need that is working, so that the results that obtained from proposed algorithm was acceptable with a slight difference between the results obtained and the original algorithm.

Table 4 shows an examples of hashing results in block size 256 of plain text. However, this algorithm provides a security randomness when working with larger files, because their work in the deployment of randomness depends more on the heterogeneity of data, the larger encrypted data was providing larger randomness. This is because the proposed algorithm does not work with the masks in which the SHA-3 algorithm works, it depends only on the size of the entered text. Since the basic purpose of this algorithm is to provide reliability when dealing with very large files that generate the result of the work of a large number of entities with the approved WoT system. Table 5 illustrates Differential attack test comparison. Table 6 shows Entropy test comparison

Table 4. Examples of hashing result in block size 256

Plain text	SSHA (24 round)	SSHA (12 round)	SHA-3
00000000000000000	9a34c87968a4c60b86f4bc2129206c2075f4f8d83a20bd8f09bc99524daff18e	e6636068bb0f620a97c2cbc0237abc6b475bc855a5dabf9e3ae939c0f6433ffc	3575571563d08eba465361ce5d77d18cfa525516992b87a63aded5ea1aff50c8
0123456789ABCDEF	c5a76c5ac39f44cc601785b8be82425026b7b000b784eb6304249f534033481c	408d2d48babd184316cb2135ac9d10b57894c8b86af7362302bebdb854b487e9	b95964a8108df9f282fd8eec246d1cc0f4d44ad360d01afc55242a25b4fc8986
FFFFFFFFFFFFFFFF	89ae0cb3d5eb7b5b4ace072222b8bcf0e48435ce63cf2ea7dac3622e04ef99e3	680248222e9a2eca3ecfe783e9575997e2e79b1a68ccc9448f2279771ff967d1	108755c524307e2727a88c0a66b8e90ece83fea32360befd468fe376741dfa44
1111111111111111	4eae34ac404519edb79d37761a0aca7eddadbdcdc77f7ccc4489d0b3a3a6c0d4	f55a701da95a801ab14428c9cb090270bb3047945bc87ddb415a6592cc958d28	d3a932d7b753c1e2607f0141a82c444ac78315df500f65c0a978d1368058e460
1010101010101010	4a92beee152e28a3f114bb637c96c4d65073c85dd7f2820c91024c6d8230e4a8	1df07c932f694f7a619af75bb1b6705cd9543bc7eb7a7d456c5c29a16905151a	a3aab6efe08656472ab37ef5631d7ec53ea3c063f3ae8757872fd62b2862006e
0000000011111111	18a6ed4ec87d6bc0588d6d352be4c61b89282df27dd8874be2fded3f5643689f	8d142a5a833cf9855a5586bcaa699c60a2524c0c934779a2866ca393d61d9bd4	4cc3b66d9b366181a269ded0a4ce8f9a4db52e2f91120f5a6324b1671f18cff0

Table 5. Differential attack test comparison

File Size	SSHA (24 round)	SSHA (12 round)	SHA-3
128B	0.356	0.3365	0.369
1KB	0.299	0.299	0.378
10KB	0.3610	0.3142	0.3643
100KB	0.379	0.2985	0.3972
1MB	0.438	0.394	0.434

Table 6. Entropy test comparison

File Size	SSHA (24 round)	SSHA (12 round)	SHA-3
128B	7.9980	7.9650	7.911
1KB	7.9982	7.9941	7.992
10KB	7.9970	7.9931	7.991
100KB	7.9981	7.9925	7.993
1MB	7.9920	7.9937	7.990

7. Conclusion and future works

After reviewing the results obtained from the tests carried out on the SSHA algorithm, we can come up with a set of conclusions that can determine whether the modification process is appropriate for the environment for which it was developed. In this paper, the SHA-3 algorithm and the SPECK algorithm are studied separately, and the processes on which the SHA-3 algorithm are based are interconnected and modified by replacing the KECCAK function with SPECK and changing the initial values of SHA-3 with the extended Lorenz system. These changes made the SSHA algorithm produced much faster than the SHA-3 algorithm with the possibility of providing a good security level when compared to the security level provided by the original SHA-3 algorithm. When compared to the original SHA-3 algorithm.

As a result, the SSHA algorithm can be used in IoT systems that need ways to verify data integrity quickly and safely. Moreover, in the proposed algorithm was developed in two cases, the first is that the SSHA algorithm has 24 rondes: Which can be used to check the sensors data that are placed in the WoT system.

References

[1] G. Pradyumna, B. Omkar and Sagar" Introduction to IOT", IARJIS, Engineering and Technology, Vol. 5, Issue 1, January 2018.

[2] S. Singh and N. Singh, "Internet of Things (IoT): Security challenges, business opportunities & reference architecture for E-commerce," *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, Noida, India, pp. 1577-1581, 2015.

[3] V. Pevnev, Y. Novakov, M. Tsuranov and V. Kharchenko, "The Method of Data Integrity Assurance for Increasing IoT Infrastructure Security", Proceedings of the 31 th International Conference on Information Technologies (InfoTech-2017), pp. 27-36, 2017.

[4] J. R. N. Al-Khazraji, G. H. Abdul-Majeed and A. Khadhim Farhan "Design and Implementation of Secure IoT for Emergency Response System Using Wireless Sensor Network and Chaotic", Ph.D. dissertation, Iraqi commission for computers and informatics, informatics institute for postgraduate studies, 2019.

[5] M. Schlater, B. Andrew, L. Lu, M. Hamilton, N. Hanley, M. O'Neill, and W. P. Marnane:" A Hardware Wrapper for the SHA-3 Hash Algorithms", Journal in Science, Engineering and Technology, Vol. 7, No. 3, Feb. 2015.

[6] S. Gueron , S. Johnson , J. Walker, "SHA-512/256", Journal in Science, Engineering and Technology, Vol. 5, No. 1, 2018.

[7] N. R. Chandrana, and E. M. Manuelb, "Performance Analysis of Modified SHA-3", *Procedia Technology*, Vol. 24, pp.904-910, 2016.

[8] S.-J. Chang, R. Perlner, W. E. Burr, M. S. Turan, J. M. Kelsey, S. Paul, L. E. Bassham, "Third-Round Report of the SHA-3 Cryptographic Hash Algorithm Competition", NISTIR 7896, 2012.

[9] B. Guido, D. Joan, P. Michaël and V. Gilles, The Keccak SHA3 submission, *NIST (Round 3)*, Vol. *6*, No.7, p.16, 2011.

[10] L. Henzen, P. Gendotti, P. Guillet, E. Pargaetzi, M. Zoller, and F. K. Gürkaynak. "Developing a hardware evaluation method for SHA-3 candidates." In *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 248-263, Berlin, Heidelberg, 2010.

[11] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "SIMON and SPECK: Block Ciphers for the Internet of Things", IACR *Cryptol. ePrint Arch.*, Vol. 2015, p. 585, 2015.

[12] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK Families of Lightweight Block Ciphers", *IACR Cryptol. ePrint Arch*., Vol. 2013, p. 404, 2013.

[13] S. Kölbl, and A. Roy, "A brief comparison of Simon and Simeck." In *International Workshop on Lightweight Cryptography for Security and Privacy*, Springer, Cham, pp. 69-88, 2016.

[14] A. Bossert, S. Cooper, and A. Wiesmaier, A comparison of block ciphers SIMON, SPECK, and KATAN, Sematic Scholar, 2016.

[15] R. A. F. Lustro, A. M. Sison, and R. P. Medina, "Performance Analysis of Enhanced SPECK Algorithm." In *Proceedings of the 4th International Conference on Industrial and Business Engineering*, pp. 256-264, 2018.

[16] A. Majed, H. K. Hoomod, Secure Email of Things Based on Hyper Chaotic system, Mustansiriyah University, M.Sc. Thesis, Baghdad, Iraq, 2020.