ISSN 2303-4521
Periodicals of Engineering and Natural Sciences *Original Research*
Vol. 9, No. 4, October 2021, pp.657-673

A proposed hash algorithm to use for blockchain base transaction flow system

Zainab Ali Kamal [1], and Rana F. Ghani [2]
[1,2]Computer Sciences Department, University of Technology-Iraq, Baghdad, Iraq

ABSTRACT

Blockchain technology introduces a new approach to storing information, implementing tasks and functions, and building trust between participating nodes. Although blockchain technology has received extensive attention in various application contexts in recent years, the issue of privacy and security remains the primary focus of discussions of the blockchain. The use of hash algorithms can provide secure blockchain integration, and many hash algorithms offer solutions to data integrity and security problems within the context of blockchain technology. However, they are also subject to problems related to time, lack of resources, and memory usage. In this research, an algorithm is proposed to generate a hash based on chaos theory (1D and 2D) logistic maps and the new Merkle-Damgård construction. Hash outputs are tested in terms of time, complexity, and collision. The proposed algorithm is evaluated according to Jaccard similarity and various coefficient measurements, and it was found that the similarity between the inputs and the outputs does not exceed 0.1932 percent. All outcomes indicated successful performance. The proposed algorithm was implemented on a blockchain-based transaction flow system, consumed fewer resources than other hash algorithms (such as SHA1, SHA2, and MD5), and requires mere milliseconds to implement.

Keywords: Blockchain, Data Integrity, Hah function, Merkle-Damgård construction, Logistic Map.

Corresponding Author:

Rana Fareed Ghani
Department of Computer Science
University of Technology- Iraq
Baghdad- Iraq
E-mail: 110016@uotechnology.edu.iq

1. Introduction

The blockchain was proposed for the first time in 2009 in a research paper presented by Satoshi Nakamoto to invent the fiat currency Bitcoin. Blockchain transactions are carried out among the users of the technology without any intermediary, making it a decentralized technology (that is, no one controls the operations that take place through it). No governmental bodies control the course of its affairs, nor do the companies that manage and regulate its functioning [1]. Even though the Bitcoin is the most popular application of block-chain technology, the blockchain may be applied to many contexts beyond digital currencies. It can be used in various financial services and transfers because it permeates a wide variety of the applications over numerous industries, including transformation, governance, and manufacturing. E-governance is a powerful and important tool, especially in developing countries that face challenges like weak public services, growing crime rates, and poor education [2-4]. The blockchain provides hash functions for each transaction which convert a random-length file into a fixed-length hash value, known as the message fingerprint. Any modification of the input data will cause a radical variation in the hash values. As a result, cryptographic hashes are frequently utilized in the security protocols and applications [5-7]. Cryptographic hash functions ensure the integrity of Authentication files in a connection. This research focuses on blockchain cryptography intended to provide database security and protect transactions against interference from outside parties. Blockchain technology enables the transfer of digital assets from peer to peer without any intermediary and with high security and is characterized by the major features of decentralization, transparency, stability, and audibility [8]. This research is divided into the following sections: (1) introduction to the blockchain, its types, and its use; (2.1) block-chain architecture; (2.2) characteristics of the block-chain; (2.3) secure hashing function (addressing the most important hash algorithms that provide high data security); (3) Merkle-Damgård construction, the most important schemes and structures adopted by modern hash algorithms; and (5) the proposed hash algorithm (Chaos Buffer Hash). The final section describes the method of generating a new algorithm that adds more protection against external attacks. Its goal is to generate high-randomness outputs that depend entirely on chaotic maps [9].

2. Blockchain types and theory

Blockchain is an innovative technology for building and using a ledger. Data are recorded on a decentralized node called a block. No third party is required to use this technology, which improves its reliability. While blockchain represents a database or a new way of organizing data, its method of dealing is different from central dealing; it is an electronic record system. Processing and recording transactions that allow all parties to track information through a secure network. Its features include decentralization; it does not belong to anyone entity but is rather the property of its users. It is also open-source and offers high levels of safety and protection, as penetrating the system is almost impossible. Blockchain guarantees its users a great deal of privacy and confidentiality; as people are only represented by encrypted codes in the system, no one can identify personal data about other users [10-12]. In general, blockchain technologies are divided into four types: public, private, consortium, and hybrid [13, 14]. A public blockchain is an unrestricted distributed ledger. Without needing to obtain permission, anyone can become a certified node by logging into the blockchain platform and becoming part of this type of network. The public blockchain allows the node or user to access current and previous records, verify transactions, prove work as an incoming block, and do mining. Examples of the public block-chain networks include Bitcoin and Ethereum. The private blockchain is restricted and requires permission because it is a closed network. The private blockchain is usually used within institutions. The participants in the blockchain network are members. The controlling organization is responsible for authorizations, the level of security, permissions, and accessibility. Hence, public blockchain is the same as private. However, the private blockchain has a small and restricted connection. Voting, digital identity management, and access ownership are examples of contexts in which it is advised to deploy private networks. Hyperledger projects (e.g., Fabric, Sawtooth), MultiChain, and Corda can be considered as examples of applied private block-chains. The consortium block-chain can be briefly defined as semi-decentralized. Consortium blockchains are run by multiple organizations, unlike other classes that are run by only one institution, such as private blockchain. It is possible to circulate more than one institution as a node point in the network. The consortium blockchain is typically used for mining or exchanging information by community and government organizations and banks. Examples of consortium blockchains include Energy Web Foundation and R3 [15, 16]. Hybrid blockchain is a type of private and public block-chain that allows the use of two kinds of blockchain features. Hybrid blockchains can have a private ear-based system. In addition to a permissionless system (public system). In addition, inside the network of the hybrid blockchain, users control who can access the stored data. The portion of the data or records that are allowed to spread is specific to the blockchain. The hybrid system is characterized by flexibility, security, and transparency. An example of a hybrid blockchain is Dragonchain [17]. One problem with the blockchain is the speed of performance in creating the hash function while providing high security and ensuring that input information cannot be discovered from the output of the algorithm. The algorithm must also be anti-collision—that is, it should be impossible to produce the Same output for two different inputs. At the same time, it is allowed to be used for any type of blockchain.

2.1. Blockchain architecture

A node registered in de-centralized block-chain begins by creating a transaction. An electronic signature with a private Encryption key is used to make that transaction. All of the transactions have been stored in a file for collecting the unconfirmed transactions and publishing them in the network with the use of a flooding algorithm that has been referred to as Gossip protocol. Peers require the validation of the transactions according to certain predefined criteria. For instance, the node attempts to authenticate the validity of transactions through the verification of whether the balance is sufficient for the initiation of a transaction, an attempt to deceive the system, or an attempt to use the same balance by entering the same transaction two or more times (double spending theory). Once the miners authenticate the transactions, they are combined in the block. A consensus mechanism is used to verify the new blocks, which helps access the decentralized network. After verification, it is included in the chain, and each copy of all peers is subject to change. The next block connects to the newly created file whenever a new block is attached to the chai [18]. The smallest task unit which is stored or saved in public records is a transaction.

A record (or block) is the smallest data structure. Every previous transaction is confirmed and not modified. During transaction initiation, mining and consensus depend heavily on the hash function [19, 5, 12]. A hash function algorithm can take any type of input (number, string, or text) and produce a fixed-length output, and any variation in the input leads to a change in the output, termed a one-way function [20]. Figure 1 summarizes the most important stages of creating a transaction via the blockchain:
(I)Initial transaction: In the first stage, a transaction is uploaded to the blockchain (anyone registered on the network can upload a transaction).
(II)Verification of transaction: In the second stage, the sender's identity is verified, meaning that transaction processing between the sender and the recipient is required by the sender and not anyone else. Digital signatures (public and private) are used [21]. In addition to the digital signature, the data (i.e. the transaction amount) are sent, and the transaction is verified by the miners based on various criteria.
(III)New block: A blockchain consists of blocks that store information about all transactions. These blocks are linked so That each block has a hash reference. Each one of the block includes a block header and a block body. Metadata such as parent block hash, block version, Merkle tree root hash, time-stamp, n Bits, and nonce have been included in block header [22]. The block body includes several transactions and a transaction counter that indicates the number of the transactions that are being tracked, represented by a set of the transactions that have been recorded in the block. In the network, each participant owns a public and private key. A secret key has been utilized for encrypting and signing. The transaction, whereas public access, is distributed across network and offers a means for everyone to help decipher the code [23].

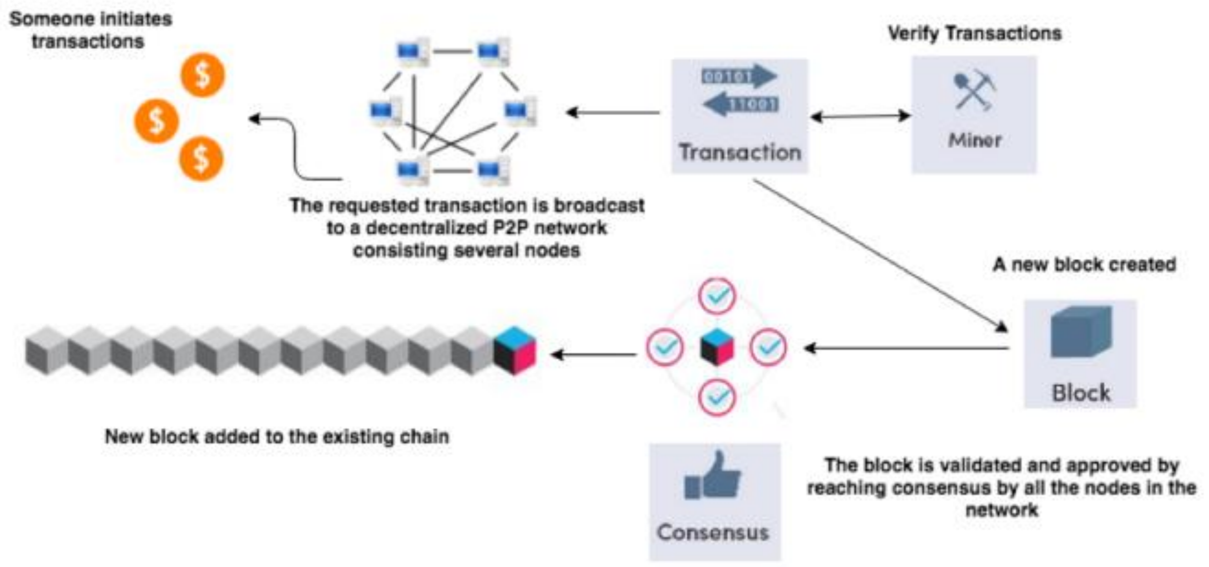
Someone initiates transactions
Verify Transactions
00101
11001
Transaction
Miner
The requested transaction is broadcast to a decentralized P2P network consisting several nodes
A new block created
Block
New block added to the existing chain
Consensus
The block is validated and approved by reaching consensus by all the nodes in the network

Figure 1. Transaction Propose Blockchain

2.2. Characteristics of the blockchains

The most important properties of the block-chain technology are: -
Immutability: The blockchain cannot be changed. This helps ensure that the technology remains the same: a permanent, immutable network [24]. Decentralization: The blockchain has no one governing authority or person. A group of nodes maintains the network, which makes it decentralized [25].Enhanced security: Since the blockchain removes the central authority, no individual entity can change any of the network properties to its advantage. Using cryptography guarantees another layer of security for the system through the use of complex mathematical algorithms that work with an attack firewall [26].
Distributed ledger: The ledger provides all information about the transaction and participants. Users can see what is happening in the ledger and, because the ledger is in the network, it is maintained by all other users in the system.
Consensus: The core of the blockchain is consensus algorithms. To help the network make decisions, each block has a consensus, defined as the decision-making process on the network for a set of the active nodes. This feature makes it possible for nodes to proceed to an agreement more quickly [27, 28].

2.3. Secure hashing blockchain

The most secure and reliable algorithms in block-chain technology are SHA-1, -2, and -256. The hash function is of unparalleled quality. It can create unique output when various inputs are given and are a unique key that is created for transaction identification. The United States Federal Information Processing Standard and the United States National Security Agency (NSA) were originally designed. The SHA. We designated these effective algorithms to verify the integrity of the files. Blockchain technology functions are reliable to use, which helps to create a suitable and robust hash code. Each proposed transaction in the blockchain is hashed. At the same time, the hash indicators connect every one of the Blocks to the next block and preserve the previous hash data. Any changes to blockchain transactions affect all the blocks involved. SHA1 and SHA2 are two versions of the same secure hash algorithm that vary in both syntaxes (how to create the resulting hash from the original data) and signature bit length. SHA2, the successor to SHA1, is considered a comprehensive improvement. Regarding bit length, SHA1 is 160 bits, whereas SHA 2 comes in various lengths but most commonly 256 bits. SHA 224, SHA 384, and SHA 512 all refer to SHA2. From 2011 to 2015, Sha1 was the primary algorithm used. However, a growing body of research has demonstrated weaknesses in SHA1 related to collision (i.e., two pieces of data originating the same hash). The algorithm used to generate the hash must have the following properties:

1- Deterministic: The hash does not change for the same message—that is, considering a certain input M, the function always calculates same output h(M).
2- Quick computation: Calculation speed.
3- Pre-image resistance: This refers to the impossibility of deducing the input through the hash. That is, considering hash value H, it's infeasible computationally to find input M such that h(M) = H.
4- A small change in input changes the hash: A small change in entered data leads to a change of the hash.
5- Collision resistant: No two messages have the same hash M and M′ such that h(M) = h(M′) and $M \neq M'$. which has been referred to as strong collision resistance as well.
6- Hashtags should be non-modifiable, not just encrypted.
7- No correlation: input bits M should not be connected to output bits h(M).
8- Random behavior: Random behavior has to be included in the hash function, making it useless to attempt to predict which output is associated with which input.

Figure 2(a) visualizes blockchain technology as the combination of a block included in its structure consisting of data and the previous block hash (except for the original one, which does not contain a previous hash). Figure 2(b) shows a blockchain network that includes several blocks linked to the previous blocks via hashing [29-32].

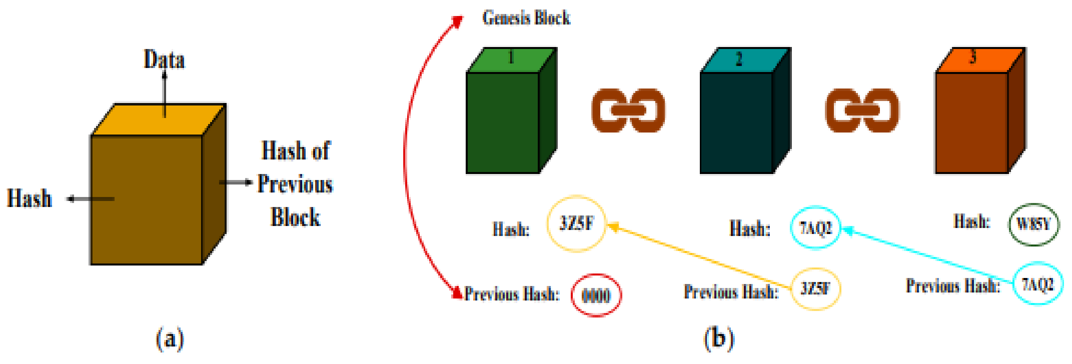

Figure2. (a) Hash block component. (b) Block-chain network.

The National Institute of Standards and Technology (NIST) was introduced in 1995 by SHA1 and was adopted for many years. Many researchers have proven after a while that it is possible under certain conditions to get conflicting messages. Thus, the safety level of SHA1 decreases. In 2001, it announced the Sha2 hash, but unfortunately, it did not meet high expectations. Therefore, hash algorithms are constantly evolving. In October 2008, the Institute announced an open discussion of a new hash function, SHA3 [33]. Therefore, researchers always seek to use hash algorithms that guarantee their first concept, which is that they are not reversible, while the second indicates that it is not easy to find 2 inputs for the same hash value [34].

3. Merkle-Damgard construction

Merkle-Damgård construction can be defined as the base that designers have relied on since the early days of cryptographic hash functions. Independently Discovered by Merkle-Damgård in 1989. It follows the MD iterative method and a pressure function that takes an input length value and outputs values with a constant output which is the core of this construct component.

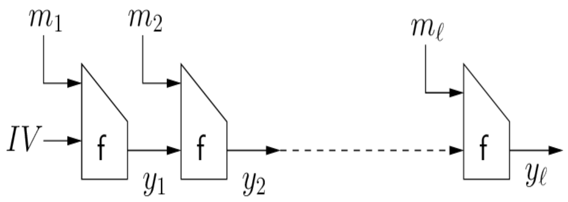

Figure 3. Merkle-Damgård construction

A chaining variable and a message block are the accepted input methods for the compression function. A description of strengthening Merkle-Damgård construction is provided in the following example. Assume that M is a binary representation of the message length M. The binary encoding of the length of the message has to be added as well for completing padding. Input M is divided consequently to t blocks, every bit-length b. The hash function h may be described then as:

$$H_0 = IV$$
$$H_I = F(H_{I-1}, M), I = 1 \dots T, \ \ldots\ldots(1)$$
$$h(M) = H_t$$

Where f represents compression function of h, H_i represents intermediate chaining that is variable between the stages i-1 and i, and H_0 represents a pre-defined starting value or an initial value IV. The diagram of iterative hash function utilizing compression function has been represented in Figure4. The calculation of the hash value has been based upon chaining variable, which has a fixed initial value at start of the hashing that has been assigned as part of the algorithm [35-37].

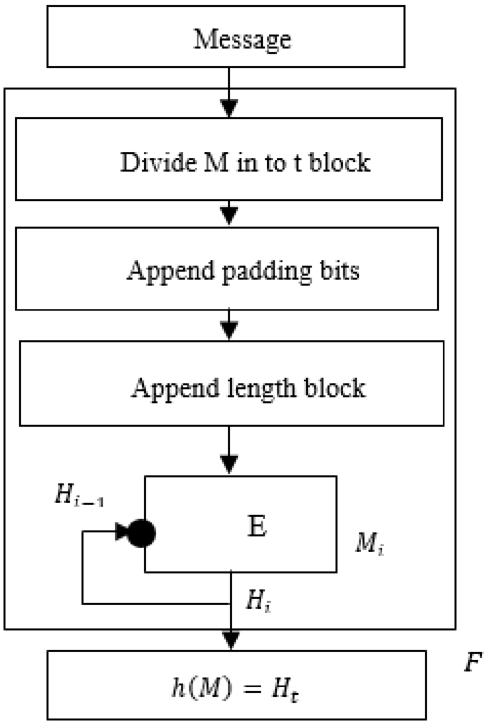

Figure 4. Detailed view of the Merkle-Damgård constructions

This operation proceeds in a recursive manner, with updating the chaining variable being in various parts of the message to the point where the whole message was utilized.

4. Logistic map

The hash function is similar to that of the random number generator concerning the random output. A dynamic system such as chaos has been found suitable for use as a hash function. Not all field maps are proper for the coding purposes. Logistical maps are simple, fast, sensitive to elementary conditions, unpredictable, and are a 1-way and iterative function [38, 39].

$$r_n = t(1 - r_{n-1})r_{n-1}$$
$$where\ t \in [0,4] r_n \in (0,1), and\ n \in N$$

And

$$x_{\ n+1} = t(3y_n + 1)\ x_n(1 - x_n)$$
$$y_{\ n+1} = t(3x_{n+1} + 1)\ x_n(1 - y_i\ )$$
$$where\ t \in [0,4]\ X_n\ and\ Y_n \in (0,1), and\ n \in N$$

Chaos theory has been used in a lot of scientific research (in the field of encryption and in the generation of hash algorithms) because it follows random behavior or defines a completely inevitable chaotic system [40].

5. Proposed hash algorithm (chaos buffer hash)

For various reasons, hash algorithms can experience certain problems, the most important of which are collisions. It is imperative to strengthen the power of algorithms so that they can withstand any problems they encounter. The hash algorithm was designed based on chaos theory (1D and 2D), logic, and Buffers with the adoption of a new structure based on Merkle-Damgård construction. The Chaos Buffer Hash algorithm deals with variable-length messages and converts them to fixed-length output (32 hexadecimals, 64 hexadecimals, and 128 hexadecimal). The algorithm steps are as follows:

1-The message is broken into chunks of 1,024-byte blocks.

If blocks are less than 1,024 bytes, the message is padded by the sum modulus 1,024 bits.

2- A 1D logistic map is used to generate 1,000 random numbers, of which the algorithm selects only 32 random numbers and only 4 numbers after (,).

The 32 numbers are converted to binary ($32 \times 4 \times 8 = 1{,}024$ bytes).

3- Made (XOR) between step 1 (1,024-byte message padding) and step2 (1,024 bytes from logistic map).

4- Using a 32-byte buffer, initial buffer values are generated from a 2D logistic map.

x_i Feeds the first 16 buffer and y_i feeds the second 16 buffer. Each buffer represents eight numbers after (,) from the 2D logistic map. Each buffer represents 64 bytes.

Initial vector: XOR between first 16 buffers.

initial vector': XOR between second 16 buffers.

5- Update buffers based on logic map:

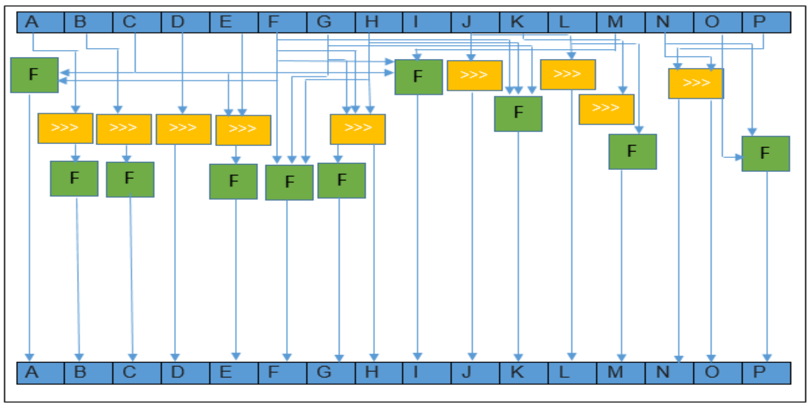

Figure 5. Logic map

Designed around more than 250 logic maps. Figure 5 was adopted because it generated the highest random percentage and had the greatest impact on the advanced stages.

6- Update initial vector: XOR between new values and first 16 update buffers.

update initial vector$'$: XOR between new values and second 16 update buffers.
7- New Merkle-Damgård construction is built (see Figure 6).
Parameters input in the new Merkle-Damgård construction:
The result from step 4 is divided into chunks of 1,024-byte blocks, each of which represents 64 bits: M1, M2… M16.

- Initial vector: XOR between first 16 buffers.
- $initial\ vector\ '$: XOR between second 16 buffers.
- Update initial vector and $initial\ vector'$ based on update buffers of each function expected in the function One Operation new Merkle-Damgård construction

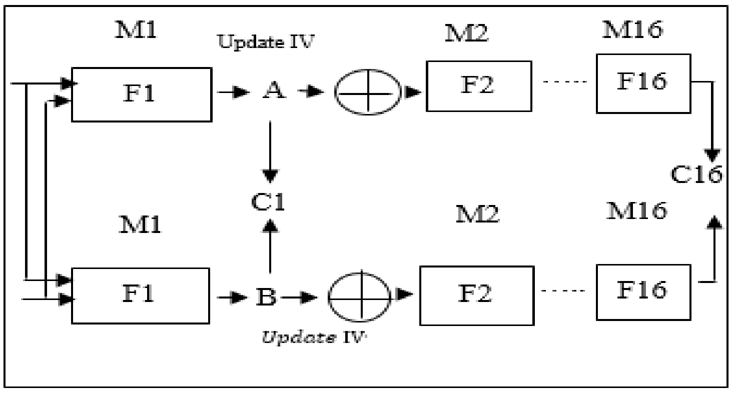

Figure 6. One Operation new Merkle-Damgård construction

$\mathrm{A{=}F(\ IV, IV^{\cdot}\ , M1)}$
$= \mathrm{(IV\ AND\ \ IV^{\cdot})\ XOR\ (IV\ AND\ NOT\ IV^{\cdot}\ AND\ M1)}$
$\mathrm{B{=}F(IV\ , IV^{\cdot}, M1)}$
$= \mathrm{(NOT\ IV\ AND\ NOT\ IV^{\cdot})XOR\ (IV\ AND\ M1\ )XOR}$
$\mathrm{(IV^{\cdot}\ AND\ M1)}$
$\mathrm{C1 = (A\ XOR\ B)}$

Need each function XOR between (a) and (b):-
A XOR WITH (UPDATE FIRST 16 BUFFER)
B XOR WITH (UPDATE SECOND 16 BUFFER)

Algorithm: generate chaos buffer hash

```
Input: Mmessage any size
Output : Hash ( fingerprint)   32, 64 and128  HEX
Begin
Step 1: The message is broken into chunk of 1024 bite blocks. If blocks less than 1024 bit
Made Expansion message by sum with mod 1024, x [1024] bite.
Step 2 : Generating state 1D random number
    2.1 : with the use of parameters chaos generator matrix 1D(a[1000])
    2.2 :Select only 32 random number
    2.3 : Convert 32 random number to binary y[1024] bite
Step 3:Made XOR between x[1024] bite and y[1024] bite
Step 4: Using 32 buffer. generated  initial value buffers from 2D logistic map
  4.1:with the use ofthe parameters chaos generator matrix 2D(R[$x_i$][$y_i$] and the
  First 16 buffer represent by the $x_i$ and the second 16 buffer represent $y_i$
```

4.2: Each buffer represent number from logistic map and convert to binary

4.3: Made XOR between first 16 buffer to get initial vector and made XOR between second 16 buffer to get initial vector’

step 5 : update buffers base on logic map

step 6 : update initial vector and initial vector ’ **go to step 8**

step 7: The result from step 4 divided in to chunks of (1024 bit) blocks each block represent 64 bit : M1,M2 ,………………,M16 and put in new Merkle-Damgård construction to get 1024 bite

End

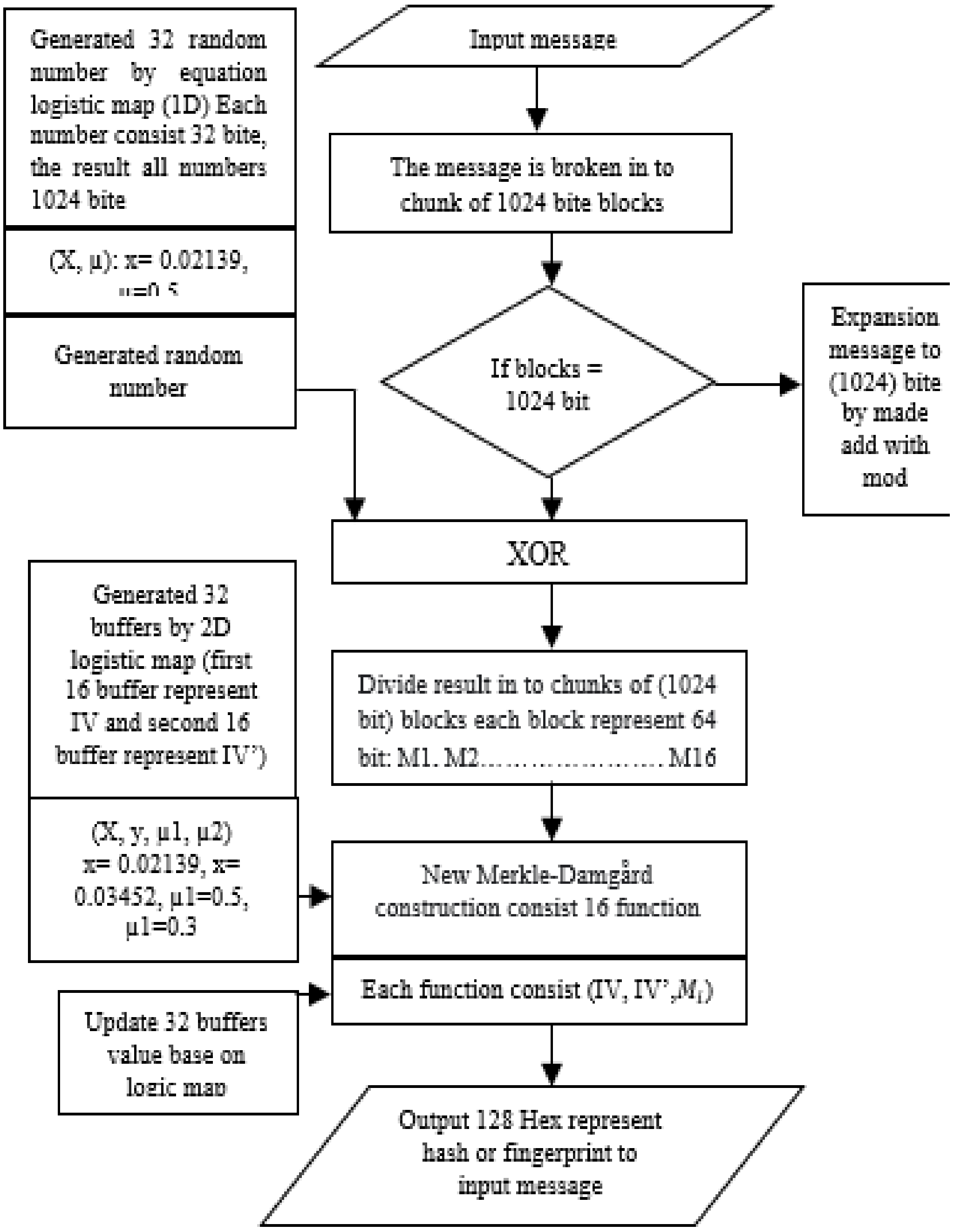
Generated 32 random number by equation logistic map (1D) Each number consist 32 bite, the result all numbers 1024 bite
(X, μ): x= 0.02139, μ=0.5
Generated random number
Input message
The message is broken in to chunk of 1024 bite blocks
If blocks = 1024 bit
Expansion message to (1024) bite by made add with mod
XOR
Generated 32 buffers by 2D logistic map (first 16 buffer represent IV and second 16 buffer represent IV’)
(X, y, μ1, μ2) x= 0.02139, x= 0.03452, μ1=0.5, μ1=0.3
Divide result in to chunks of (1024 bit) blocks each block represent 64 bit: M1. M2………………….M16
New Merkle-Damgård construction consist 16 function
Each function consist (IV, IV’,M_i)
Update 32 buffers value base on logic map
Output 128 Hex represent hash or fingerprint to input message

Figure 7. Block Diagram for Propose Chaos Buffer Hash

Example: input "ali"

Input Message = ali, Convert to ASCII Code 97,108,105 and Expansion message

Sum (97 + 108) mod 1024=205	Sum (924+ 113) mod 1024 = 13
Sum (97 + 205) mod 1024 = 302	Sum (519 + l3) mod 1024= 532
Sum (108 + 302) mod 1024 =410	Sum (112+ 532) mod 1024= 644
Sum (205 + 410) mod 1024 =515	Sum (113 + 644) mod 1024= 757
Sum (205 +515) mod 1024 =720	Sum (13 + 757) mod 1024=770
Sum (302+720) mod 1024 =1022	Sum (532 +770) mod 1024= 278
Sum (410+1022) mod 1024 =408	Sum (644+ 278) mod 1024= 922
Sum (515+ 408) mod 1024 =923	Sum (757+ 922) mod 1024= 655
Sum (720+ 923) mod 1024 =619	Sum (770 + 655) mod 1024= 401
Sum (1022+ 619) mod 1024 =617	Sum (278+ 401) mod 1024= 679
Sum (408 + 617) mod 1024 = 1	Sum (922 + 679) mod 1024= 577
Sum (923 + 1) mod 1024 = 924	Sum (655 + 577) mod 1024= 208

Message padding
97,108,105,205,302,410,515,720,1022,408,923,619,617,1,924,519,112,113,13,532,644,757,770,278,922,655,401,679,577,208,609,264

Each number (x) in the message padding represents four bytes. If x consists of one rank, must add another 3 digits, if x consists of two ranks must be added two other digits, and if x consists of one rank must be added with three digits.

Message padding
9787,1082,1053,2054,3020,4101,5155,7209,1022,4086,923C,619B,617D,1AEF,9248,5197,1122,1133,1340,5321,6445,7579,7706,278C,922B,655D,401A,679F,577E,2088,6097,2642

A second expansion of the message (only when needed), by using a special table containing HEXA numbers.

Generating state 1D [32] random number

Example: 1.0008995953645272
- Take only 8 numbers after (,) =000899593645272
- Take only the last four numbers out of the eight numbers=9959
- Made reverse to last four numbers = 9599

Made XOR between message padding binary and random numbers binary
Using 32 buffer. Generated initial value buffers from 2D logistic map
- *Initial vector:* XOR between (first 16 buffers)
- $initial\ vector\,'$: XOR between (second 16 buffers)
- Update buffers base on the logic map
- The result from message padding and chaos 1D divided into chunks of (1024 bit) blocks each block represent 64 bit: M1, M2,………………, M16

By this parameter put in the new Merkle-Damgård construction to get message-digest consist (32, 64, 128 HEXA, 256 HEXA)

128 HEXA
B556AA9551373935ECB326FF2F3D8CA67D58575D996957594A5D3CCD0FB6B8A952D2FA5
DCB2662DA9F996B5512D2E9A3CB335BAA7C775754971C4CA565A3A56B9F5A94A4DCB92
D76A69A6A6DF69BBAD2A66F0EDCD353D3154DBAB1B4537ED51529B3E956BA69546AD9
BD99D429B394AAEA6D32BBA66E6D73A5366468B4D35D55

64 HEXA
7D58575D996957594A5D3CCD0FB6B8A952D2FA5DCB2662DA9F996B5512D2E9A3

32 HEXA
CB335BAA7C775754971C4CA565A3A56B

6. Experimental results

The output of the proposed hash algorithm (Chaos Buffer Hash) was tested according to several metrics, including time, complexity, collision, and resource consumption.

➢ **Time:** The algorithm takes only a few milliseconds to implement. The time was calculated using C#'s built-in Timer class.

Table 1. The Time It Takes to Generate A Hash

message	Execution time
Rana	**0.190 millisecond**
zainab	**0.188 millisecond**

➢ **Complexity**: The complexity of the algorithm is a result of using the logistic map parameters. Changing the initial values of the logistical equations changes the results of the hash, so it requires all parties using the algorithm to prove the same initial values of the logistical equations used.

Table 2. Complexity in Chaos Buffer Hash Alorithm

message	Initial value (1D,2D)	Output Hash 128 HEXA

ali	Logistic map 1D : R=2
	X=0.0009
	
	Logistic map 2D:
	X= 0.02139,
	Y= 0.02139
		B556AA9551373935ECB326FF2F3D8CA6
		7D58575D996957594A5D3CCD0FB6B8A
		952D2FA5DCB2662DA9F996B5512D2E9
		A3CB335BAA7C775754971C4CA565A3A
		56B9F5A94A4DCB92D76A69A6A6DF69
		BBAD2A66F0EDCD353D3154DBAB1B4
		537ED51529B3E956BA69546AD9BD99D
		429B394AAEA6D32BBA66E6D73A53664
		68B4D35D55

ali	Logistic map 1D : R=7
	X=0,006
	
	Logistic map 2D:
	X = 0.345
	Y = 0.87
		52AA92AA2DF9C8D6A2B267D19EA306D
		96156B4A2CABD92914F29C376B5A29523
		BF2B454CCB6ADD9965A811174A8FA333
		755BB4EF5D5111C5956A8874BAEDAA96
		293DD723556155A64DABA2AD758A27B4
		6C91567B3142ACEA8DA2EDDA854156F
		AAA58AAAA9AD7B999A202AA712EE15
		2BDB3DB6BCF02AC64B54551AF75

ali	Logistic map 1D : R=3
	X=0,0098
	
	Logistic map 2D:
	X = 0.08658,
	Y = 0.768
		6D2954A54B375934D8DD35FC69762833
		AEE9E3535D695C72E638CCCBB8CA63C
		D571B298AB5CCD8B5D0B187A69F8E29
		D6AE66B5D4EAAC65898AB38C5A56AD
		DD1524AAF592AAAB2B74D4B57E566B4
		69B87765CAAF563656B562D2A6ED52A8
		AEFD2B52B959515BA7533B11951E5576B
		B256DB4EB5CE6569232B2A95BB

ali	Logistic map 1D : R=3
	X=0,0034
	
	Logistic map 2D:
	X = 0.2457,
	Y = 0.9768
		2AB52A4AA2AFCE265A99A67D3CEB006
		EEAC57529A57ADCA4E9D32C2BBB6F19
		2696EAB4C9B11AABBCD6BA04D2D4D6
		A66D67555DC73AAD5138E4D556A9A3C
		D2D96B2493F2C9555A2D356A5ABB16ED
		8227E4EAB262D57BC545B6AC5A2FEEA
		D48B67A55A5C5A5A88B5FB39E8845B32
		EE8B69AADB3E6AE7116CE4762D3A9752
		76CA796B9E52D923554B4AEEA

- **Collision resistance:** It’s not possible to generate 2 different strings for the same output or to generate more than one output for one input at a time. About 8,000 hashes were generated using the proposed algorithm and did not result in any collision.

- **Input/output size:** Any size of data may be used as input, while the output length is fixed.

6.1. Statistical Analysis of Experimental Results

Statistical analysis was applied to find the percentage of similarity between the input and output of proposed algorithm. We used the Jaccard similarity coefficient, a statistical parameter that measures similarity and diversity in sample groups by finding the ratio of intersection on the union. This was used for binary data that provides an accurate solution and methods for testing hypotheses using Jaccard’s modulus.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cup B|}$$

This paper improves blockchain security by presenting a major challenge to exchanging information through the blockchain network, thus providing security and trust, which is extremely important. Thus, it achieved sharing of data resources among nodes and authentication between different users.

- A= input message (binary)
- B= message Digest (binary)

M_{11}:- total number of attributes where A and B both have 1
M_{10}:- total number of attributes where A=1 and B =0
M_{01}:- total number of attributes where A=0 and B =1
M_{00}:- total number of attributes where A and B both have 0

- TEST 1:

A= input message (ali) made an expansion to 1024 bits
B= message Digest output from (Chaos Buffer Hash) 1024 bits

Table 3. Jaccard Similarity and Differently Measure/Test1

Chaos Buffer Hah			
Message	Jaccard coefficient	Similarity	Jaccard differently coefficient
ali	0.2110		0.789

TEST 2:

A= input message (xyzasgjkk) made expansion to 1024 bits
B= message Digest output from (Chaos Buffer Hash) 1024 bits

Table 4. Jaccard Similarity and Differently Measur/Test2

Chaos Buffer Hah			
Message	Jaccard coefficient	Similarity	Jaccard differently coefficient
xyzasgjkk	0.1923		0.8077

TEST 3:
A= input message (khuyewqasdfg) made an expansion to 1024 bits
B= message Digest output from (Chaos Buffer Hash) 1024 bits

Table 5. Jaccard Similarity and Differently Measure/Test3

	Chaos Buffer Hah	
Message	Jaccard Similarity coefficient	Jaccard differently coefficient
khuyewqasdfg	0.1924	0.8076

TEST 4:
A= input message (تانبي يناغثصشئ) made expansion to 1024 bits
B= message Digest output from (Chaos Buffer Hash) 1024 bits

Table 6. Jaccard Similarity and Differently Measur/Test4

Chaos Buffer Hah			
Message	Jaccard coefficient	Similarity	Jaccard differently coefficient
تانبي يناغثصشئ	0.179		0.8121

TEST 5:
A= input message (Ghyt74892) made expansion to 1024 bits
B= message Digest output from (Chaos Buffer Hash) 1024 bits

Table 7. Jaccard Similarity and Differently Measur/Test5

Chaos Buffer Hah			
Message	Jaccard coefficient	Similarity	Jaccard differently coefficient
Ghyt74892	0.189		0.811

The first test recorded a percentage of similarity and difference with a value 0.2110 and 0.789.
The first test recorded a percentage of similarity and difference with a value 0.1923 and 0.8077.
The first test recorded a percentage of similarity and difference with a value 0.1924 and 0.8076.
The first test recorded a percentage of similarity and difference with a value 0.179 and 0.8121.
The first test recorded a percentage of similarity and difference with a value 0.189 and 0.811.
The results of this test were confirmed by the large difference between the input values and the output values that were measured with binary.

Table 8. Samples Experimental Results Comparing the Hash Values with the Execution Time Between the Md5 & Propose (Chaos Buffer Hash)

file size (bits)	BasicMD5	Execution time	Propose algorithm		
			256-bithash length	512-bit hash length	Execution time
200	6A6EB0ABFCCE28BCD 53F5B553537AAFA	0.02	5ED57D4CF52AAD353E95F1F9B D4B5AAA9F1548A5A9B369A5F CEED717BCD77035	555626E7CCA48F34B4D5C6A9DA0D9BF35E D57D4CF52AAD353E95F1F9BD4B5AAA9F15 48A5A9B369A5FCEED717BCD770356A6EB0 ABFCCE28BCD53F5B553537AAFAE995A2D 6AECD254B42ABB566B4AAD916AB7B5DD5 A97D53D2CD576ADE1776DE6B52AD71146A ACEA5BEDEDD356EDD78526AB722AEBED 62375557DB5AA557AB6BB5	0.0923
480	91fc2e 01d4d6 31a8ff dae68b a2cb84 2b	0.092	AA6F9AA553A93194A9F4 AF8D51AD528ED4F8AA45 2D4D9B4D2F53BD642AD6 A954	53B8A5A5549192DD5255EA0AAA377522AA6 F9AA553A93194A9F4AF8D51AD528ED4F8A A452D4D9B4D2F53BD642AD6A9545B5354D D6157AA6A2AD19542DAAF2EA6F55F5B1A DD2953489A92B7492B782A47AF152A4BED6 8B549C652C55DAB785D496A94DAB5C451A AB3A96EAFE0D692B515AAB722AEBA4D26 65521DAABB57AB6BB5	0.198
512	ee60ff27ad9b7e8a378 7b6cb0 10d468 a	0.098	CB12CAB6F096353A5CCE 2B290AA99A7E955AA5425 67352A559AAF56D26535B A9	B4B1A95B0B374A96D998DE5955252566 CB12CAB6F096353A5CCE2B290AA99A 7E955AA542567352A559AAF56D26535B A96BD6912B38B132AD2C56309794A5A D88E5B5356969D1A9B5751C518AAA5B 514DA54E6F42554B6D58DA34E4A644A A96CFD2AA5A9CA914A5A5B9A634AA 9CA7FA8A5B495DB2A733EC154B648D 55391A9454	0.123
1024	8adbbe6fba07fb5551a66976 fc72d276	0.1986	B54925533D4AAB4D4FA57C75 37569B6DA9F1548A5A9B369A 5EA77AC855AEA6BC	A6C4AD34E750AB534B35CD4ED06CAEA6B 54925533D4AAB4D4FA57C7537569B6DA9F1 548A5A9B369A5EA77AC855AEA6BCB24A9 BAC2AFA99E279753F5B553537AAFAEC6AA D1552D5AAACAA955555574D5B22D2DAAD 52A4954F4B355DAB785D56DACF6AD71146 AACEA5BABF835ABCB4A98AADC8ABAEA BD14E9757DB5AA557AB6BB5	0.2987
2038	f89ceff4772f1575984c8 ff7341917819		6CB12CAAEA5B454A2B937 A189352589555732B16F55D 9542D467352A59A7A68F	BAAC96B5526D4D595334AF57134C54566CB12 CAAEA5B454A2B937A189352589555732B16F5 5D9542D467352A59A7A68FE6535BAA6354E8A	

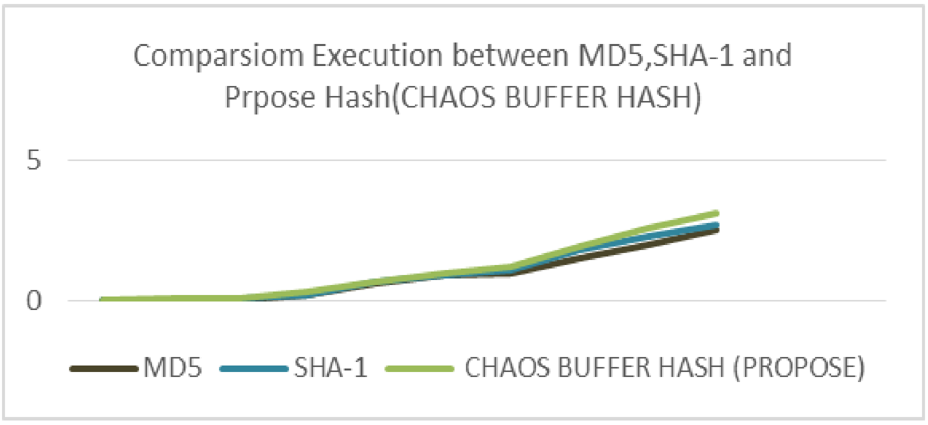

Figure 8. Comparison time between MD5, SHA-1, and CHAOS BUFFER HASH

6.2. Analysis of the seven stages of the proposed algorithm

The seven stages of the Chaos Buffer Hash algorithm were applied to more than 2,000 text files, and we found that all seven phases were collision-resistant. The algorithm was hybridized using logistic maps, logic, and new Merkle-Damgård construction and refined to increase its complexity while preserving temporal efficiency. The proposed technique was implemented in C# and run several times on a machine containing a 64-bit Intel Pentium i76500U processor running at 2.50 GHz and 2.70 GHz with 8GB RAM on the Microsoft Windows 10 OS.

6.3. Blockchain applying

The proposed algorithm was applied to a blockchain-based transaction flow system. Each transaction is sent by more than one node. The transaction must be verified through the hash generated for it. Each time data are sent and received between the nodes, the transaction is verified through the generated hash, which must give the same hash every time.
Node1 suggests a transaction and enters it into the Chaos Buffer Hash algorithm to give it a hash of its own (generated from the information recorded in the transaction). Node2 checks the hash value before accepting it. If the results give the same hash value as that sent from the first node, it is accepted and sent to the other nodes. Once the transaction passes through more than one node, a decision is made after the validity of the transaction has been verified every time it is sent or received. It is then placed in the blockchain. Upon any inquiry about the transaction, it will be retrieved from the blockchain. The nodes are connected through a TCP/IP protocol in the same network, representing the number of employees in a network.
Nodes: - It represents the number of employees within the network.

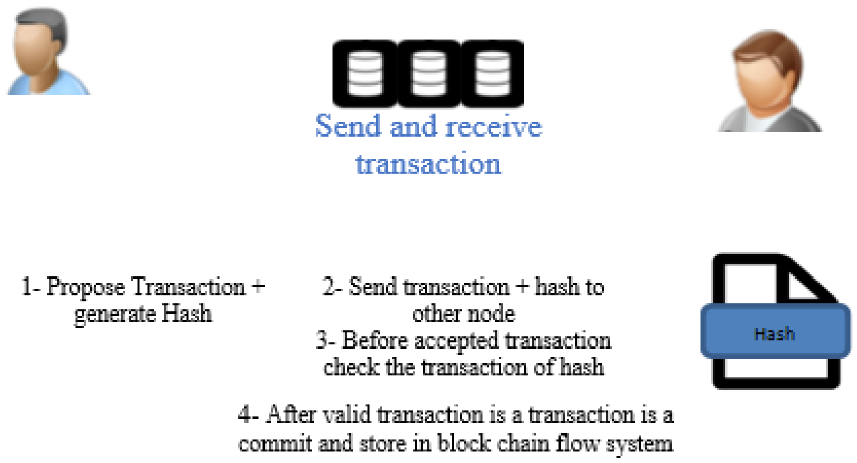
Send and receive transaction
1- Propose Transaction + generate Hash
2- Send transaction + hash to other node
3- Before accepted transaction check the transaction of hash
Hash
4- After valid transaction is a transaction is a commit and store in block chain flow system

Figure 9. Each node check the hash transaction

Regarding the importance of the proposed blockchain system, it does not allow for the modification of transactions and their hash values. If a change is made to the transaction through a malicious node, the hash values must be changed and altered in all the data stored at the node. The change is known because each node uses the same blockchain system and there will thus be discrepancies in the hash values for the same transaction.

7. Conclusion

This work provides a secure decentralized ledger of transactions that are saved and distributed among the nodes in the network using a secure blockchain-based on a hash algorithm. A new algorithm was proposed for generating a hash that works on any size of data and gives four outputs of different lengths without any impact on the speed of the algorithm. The algorithm was strengthened against possible attacks. Every transaction is proposed by a node within the network, and a decision is only made about that transaction after it is confirmed by all parties participating in the transaction system. Every sending and receiving movement is recorded in the blockchain. the algorithm’s efficiency was proven through the various measures applied, and it was found to be It is not prone to manipulation from outside parties. The proposed system has been applied to a municipality (the municipal work function) to identify whether a citizen is a beneficiary of land from the municipality. A form is opened for citizens and passed through the employees to decide it. The decision is thus a consensus between more than one party, and any sending and receiving process is recorded in the blockchain. This provides strong authentication between nodes depending on the proposed hash.

References

[1] Y. Zou, T. Meng, P. Zhang, W. Zhang, and A. Li, "Focus on blockchain: A comprehensive survey on academic and application," *I*EEE Access, vol. 8, pp. 187182-187201, 2020.

[2] W. J. Luther, "Bitcoin and the future of digital payments," *The Independent Review,* vol. 20, no. 3, pp. 397-404, 2016.

[3] M. Sharples and J. Domingue, "The blockchain and kudos: A distributed system for educational record, reputation and reward," in *European conference on technology enhanced learning,* 2016, pp. 490-496: Springer.

[4] L. F. Jawad, B. H. Majeed, and H. Alrikabi, "Tactical Thinking and its Relationship with Solving Mathematical Problems Among Mathematics Department Students," *International Journal of Emerging Technologies in Learning (iJET),* vol. 16, no. 9, pp. 247-262, 2021.

[5] V. Morabito, "Business innovation through blockchain," *Cham: Springer International Publishing,* 2017.

[6] J. Al-Jaroodi and N. Mohamed, "Blockchain in industries: A survey," IEEE Access, vol. 7, pp. 36500-36515, 2019.

[7] B. Marr, "A very brief history of blockchain technology everyone should read," *Forbes: New York, NY, USA*, 2018.

[8] M. A. Uddin, A. Stranieri, I. Gondal, and V. Balasubramanian, "A survey on the adoption of blockchain in iot: Challenges and solutions," *Blockchain: Research Applications*, p. 100006, 2021.

[9] H. T. Salim, and N. A. Jasim, "Design and Implementation of Smart City Applications Based on the Internet of Things," *International Journal of Interactive Mobile Technologies (iJIM),* vol. 15, no. 13, pp. 4-15, 2021.

[10] M. Kouhizadeh and J. Sarkis, "Blockchain practices, potentials, and perspectives in greening supply chains," *Sustainability,* vol. 10, no. 10, p. 3652, 2018.

[11] B. Bitcoin, "BlockChain Technology," Tech. Rep2015.

[12] H. Th., and H. Tauma, "Enhanced Data Security of Communication System using Combined Encryption and Steganography," *International Journal of Interactive Mobile Technologies,* vol. 15, no. 16, pp. 144-157, 2021.

[13] D. Allessie, M. Sobolewski, L. Vaccari, and F. Pignatelli, "Blockchain for digital government," *Luxembourg: Publications Office of the European Union,* 2019.

[14] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE international congress on big data (BigData congress)*, 2017, pp. 557-564: IEEE.

[15] A. Razzaq, M. Khan, R. Talib, A. Butt, N. Hanif, S. Afzal, and M. Raouf, "Use of Blockchain in governance: A systematic literature review," *International Journal of Advanced Computer Science Applications,* vol. 10, no. 5, pp. 685-691, 2019.

[16] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops)*, 2017, pp. 618-623: IEEE.

[17] C. Brunner, F. Knirsch, and D. Engel, "SPROOF: A Platform for Issuing and Verifying Documents in a Public Blockchain," in *ICISSP*, 2019, pp. 15-25.

[18] G. Karame, E. Androulaki, and S. Capkun, "Two Bitcoins at the Price of One? Double-Spending Attacks on Fast Payments in Bitcoin," *IACR Cryptol. ePrint Arch,* vol. 2012, no. 248, pp. 1-17, 2012.

[19] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys Tutorials,* vol. 18, no. 3, pp. 2084-2123, 2016.

[20] E. Karaarslan and E. Konacaklı, "Data storage in the decentralized world: Blockchain and derivatives," *arXiv preprint arXiv:.10253*, 2020.

[21] G. O. Karame and E. Androulaki, *Bitcoin and blockchain security*. Artech House, 2016.

[22] Y. Yuan and F. Y. B. Wang, "Cryptocurrencies: Model, Techniques, and Applications," *IEEE Transactions on Systems, Man, Cybernetics: Systems,* vol. 10, 2018.

[23] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 2017, pp. 468-477: IEEE.

[24] H. Yu, Z. Yang, and R. O. Sinnott, "Decentralized big data auditing for smart city environments leveraging blockchain technology," *IEEE Access,* vol. 7, pp. 6288-6296, 2018.

[25] Q. Wang, X. Li, and Y. Yu, "Anonymity for bitcoin from secure escrow address," *IEEE Access,* vol. 6, pp. 12336-12341, 2017.

[26] D. Shrier, W. Wu, and A. Pentland, "Blockchain & infrastructure (identity, data security)," *Massachusetts Institute of Technology-Connection Science,* vol. 1, no. 3, pp. 1-19, 2016.

[27] J. Wang, P. Wu, X. Wang, and W. Shou, "The outlook of blockchain technology for construction engineering management," *Frontiers of engineering management,* pp. 67-75, 2017.

[28] S. M. Ali, and H. Salim "Finding the discriminative frequencies of motor electroencephalography signal using genetic algorithm," *TELKOMNIKA,* vol. 19, no. 1, pp. 285-291, 2021.

[29] A. M. Ali and A. K. Farhan, "A novel improvement with an effective expansion to enhance the MD5 hash function for verification of a secure e-document," *IEEE Access* vol. 8, pp. 80290-80304, 2020.

[30] H. S. Abdulah, M. A. H. Al-Rawi, and D. N. Hammod, "Message Authentication Using New Hash Function," *Al-Nahrain Journal of Science,* vol. 19, no. 3, pp. 148-153, 2016.

[31] M. A. a. Roa'a, I. A. Aljazaery, S. K. Al_Dulaimi, H. T. S. Alrikabi, and Informatics, "Generation of High Dynamic Range for Enhancing the Panorama Environment," *Bulletin of Electrical Engineering,* vol. 10, no. 1, 2021.

[32] A. S. Hussein, R. S. Khairy, S. M. M. Najeeb, and H. T. ALRikabi, "Credit Card Fraud Detection Using Fuzzy Rough Nearest Neighbor and Sequential Minimal Optimization with Logistic Regression," *International Journal of Interactive Mobile Technologies,* vol. 15, no. 5, 2021.

[33] F. Sheikh and L. Sousa, *Circuits and Systems for Security and Privacy*. CRC Press, 2017.

[34] L. Wang, X. Shen, J. Li, J. Shao, and Y. Yang, "Cryptographic primitives in blockchains," *Journal of Network Computer Applications,* vol. 127, pp. 43-58, 2019.

[35] H. Tiwari, "Merkle-Damgård construction method and alternatives: a review," *Journal of Information Organizational Sciences,* vol. 41, no. 2, pp. 283-304, 2017.

[36] T. Mieno, T. Yoshimura, H. Okazaki, Y. Futa, and K. Arai, "Formal Verification of Merkle–Damgård Construction in ProVerif," in *2020 International Symposium on Information Theory and Its Applications (ISITA)*, 2020, pp. 602-606: IEEE.

[37] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya, "Merkle-Damgård revisited: How to construct a hash function," in *Annual International Cryptology Conference*, 2005, pp. 430-448: Springer.

[38] A. F. Kadhim and Z. A. Kamal, "Generating dynamic S-box based on particle swarm optimization and chaos theory for AES," *Iraqi Journal of Science*, vol. 59, no. 3C, pp. 1733-1745, 2018.

[39] A. F. Kadhim and Z. A. Kamal, "Dynamic S-BOX base on primitive polynomial and chaos theory," in *2018 International Conference on Engineering Technology and their Applications (IICETA)*, 2018, pp. 7-12: IEEE.

[40] S. Arrag, A. Hamdoun, A. Tragha, and S. E. Khamlich, "IMPLEMENTATION OF STRONGER AES BY USING DYNAMIC S-BOX DEPENDENT OF MASTER KEY," *Journal of Theoretical Applied Information Technology,* vol. 53, no. 2, 2013.