# Hashing Algorithm and Blockchain: The Unbreakable Link

Mirkhon Mukhammadovich Nurullaev[a]
*Department of Information Communication Technology, Bukhara Engineering Technological Institute, Bukhara, Uzbekistan*
*nurullayevmirxon@gmail.com*

Keywords: Hashing algorithm, Blockchain, Digital security, Transaction.

Abstract: In the contemporary landscape of digital security and decentralized ledger technology, the unbreakable link between Hashing Algorithms and Blockchain has emerged as a pivotal cornerstone. This abstract delves into the symbiotic relationship between these two key elements and their profound impact on the world of cybersecurity and trustless transactions. Hashing algorithms, revered for their cryptographic robustness, have been instrumental in safeguarding data integrity, authentication, and confidentiality. They serve as the linchpin of blockchain technology by providing a secure and efficient means of processing and validating transactions. Conversely, blockchain technology has revolutionized various industries by introducing a transparent, immutable, and decentralized ledger. Through the inherent structure of blockchain, trust is established among participants without the need for intermediaries. This abstract explores how blockchain's architecture leverages hashing algorithms to validate transactions and secure data in a distributed ledger, creating an unbreakable link that is the bedrock of trust and transparency. The fusion of hashing algorithms and blockchain technology offers solutions to the perennial challenges of data security and trust in the digital age. Ultimately, the amalgamation of hashing algorithms and blockchain technology has redefined the way we perceive and engage with digital security and decentralized systems.

## 1 INTRODUCTION

In the realm of modern cryptography and digital security, hashing algorithms play a pivotal role in ensuring the integrity and immutability of data. One of the most prominent applications of hashing algorithms is in the domain of blockchain technology. This article explores the profound connection between hashing algorithms and blockchain, elucidating how they work together to create an unbreakable chain of trust and transparency.

Hashing Algorithms: The Building Blocks

Hashing algorithms are mathematical functions that take an input (or 'message') and return a fixed-length string of characters, which is typically a seemingly random sequence of numbers and letters. The essence of hashing is to take data of arbitrary size and convert it into a fixed-size string of characters, often referred to as a hash value or digest (Wang and Wu, 2017).

The key characteristics of a good hashing algorithm are as follows:

Deterministic: The same input will always produce the same hash value.

Fast Computation: Hashing algorithms are designed to produce hash values quickly.

Pre-image Resistance: Given a hash value, it should be computationally infeasible to determine the original input.

Collision Resistance: It should be exceedingly unlikely that two different inputs produce the same hash value.

Avalanche Effect: A small change in the input data should result in a significantly different hash value.

These attributes make hashing algorithms an integral part of securing data and verifying its integrity (Shrivas 2019).

The Role of Hashing in Blockchain

Blockchain technology, known primarily for underpinning cryptocurrencies like Bitcoin, is essentially a decentralized, distributed ledger that records transactions across a network of computers. Each new transaction is grouped into a block, which is then linked to the previous one, forming a chain of

[a] https://orcid.org/0000-0001-7510-7628

blocks – the blockchain (Agung and Handayani 2020).

Hashing plays a critical role in the security and immutability of the blockchain (Nurullaev and Aloev, 2020). Here's how:

Data Integrity: When a new block of transactions is added to the blockchain, its contents are hashed. This hash value is included in the header of the new block. Any tampering with the data inside the block would result in a completely different hash value, immediately alerting the network to foul play.

Linking Blocks: Each block contains the hash of the previous block in its header, creating a continuous chain of blocks. This linkage ensures that once a block is added to the blockchain, it cannot be altered without changing the data in all subsequent blocks, which is computationally unfeasible.

Immutability: The distributed nature of the blockchain, combined with hashing, makes it nearly impossible to alter data once it is recorded. This immutability is what makes blockchain a reliable ledger for various applications, from financial transactions to supply chain management.

Consensus Mechanisms: Hashing algorithms are also essential in various consensus mechanisms used in blockchain, such as Proof of Work (PoW) and Proof of Stake (PoS). These mechanisms rely on complex mathematical puzzles that involve hashing to validate and secure transactions (Miglani and Neeraj, 2020).

Common Hashing Algorithms in Blockchain

Several hashing algorithms are commonly used in blockchain technology. Two of the most prominent are:

SHA-256 (Secure Hash Algorithm 256): Used in Bitcoin, SHA-256 is known for its reliability and security. It produces a 256-bit (32-character) hash value.

Ethash: Ethash is the hashing algorithm used in Ethereum. It is designed to be memory-intensive and resistant to specialized hardware, promoting decentralization. And we describe the O`zDSt 1106:2009 hash function algorithm below (O`z DSt 1106:2009, 2009).

# 2 RESEARCH METHODOLOGY AND RESULTS

## 2.1 The O`ZDST 1106:2009 algorithm

The following parameters and functions are used for the hashing algorithm:

- **uzunlik** - is the penultimate block of input data to the hashing function, containing the length of the source data in bits;
- **NY** - is the last block of input data to the hashing function, containing the sum of the values of the source data in decimal notation;
- **e0** - is the total number of stages of the hashing procedure equal to **(b +2)10**, for input 256 bit blocks;
- **Qosh(holat, holatn)** is a transformation used in the hashing procedure over elements at the half-byte (byte) level of the current value of the **holat** array and the holatn array modulo p based on the exponentiation operation with the parameter (**A, B, R**).

Stages of hashing function execution:

- the hashed message is divided into **b** blocks 256 bits long, i.e. **1,...,b** blocks at the half-byte (byte) level, and the missing part of the last (**b**-th) block is filled with a sequence of half-bytes (bytes) of zeros - "**0**"; as a result, the main part of the input data of the hashing function is determined consisting of from **b** blocks;
- the total length of the main part is calculated in bits modulo $\mathbf{2^{256}}$, which is a 256-bit uzunlik block;
- the sum of the values of the blocks of the main part is calculated modulo $\mathbf{2^{256}}$, which is a 256-bit checksum block (**NY**); the main part, the **uzunlik** block and the **NY** block in the form of two-dimensional arrays of elements at the half-byte (byte) level of **b+2** blocks represent the input data of the hashing function.
- The initial stage is completed by copying the 256-bit hashing key to a two-dimensional array key.

Hashing processes in relation to each block of input data begin with the execution of a chain of a pair of transformations **Qosh (holat, holatn)**, **BaytZichlash (holat, holatn)** over two blocks holat and holatn and ends with the formation of the current hash value holat within 10 stages. At the very beginning of hashing processes, 1 block is used as the initial hash value as the **holat** block, and 2 block - as the **holatn** block; if the input data consists of only one block, then the **uzunlik** block is used as the 2 block.

- then another block is copied to the **holatn** array and 10 stages of the hashing procedure are implemented over the result of a pair of transformations **Qosh(holat, holatn), BaytZichlash (holat, holatn),** over the current hash value of **holat** and **holatn**, etc. The last block copied to the holatn array is the **NY**

block. Thus, the total number of stages of the hashing function is **(b+1)10**.

Each stage (round) of the hashing procedure relative to blocks, along with the initial pair of transformations **Qosh(holat, holatn), BaytZichlash(holat, holatn)**, contains transformations **Aralash(holat, $k_e$), Qosh(holat, holatn), SurHolat(holat)), SurKalit($k_e$), TuzilmaKalit($k_e$,k),** carried out in a cyclic order.

The data processing process in the hashing algorithm can be interpreted as the execution of algorithm operations on a two-dimensional array of half-bytes (bytes) **holat**.

The **holat** array consists of four rows (rows) and eight columns of half-bytes (bytes), with each row containing 32 bits.

**Qosh(holat, holatn, R)** is a transformation in which each half-byte (byte) of the holat and holatn array is replaced by its **x=-3** degree modulo **p** based on the exponentiation operation with the parameter **(A, B, R)** at $\mathbf{R} \equiv \mathbf{k}[\mathbf{1}]$**.**

The following transformations are performed on each half-byte (byte) of the **holat** and **holatn** array:

For $\mathbf{0 \le s < 4},\ \mathbf{0 \le u < 8}$ **,**

$h[s,u]^{\setminus -1} \equiv -A_{1h}h[s,u](B_{1h} + R\text{x}h[s,u])^{-1}(mod\text{p})$**,**

$hn[s,u]^{\setminus -1} \equiv -A_{1hn}hn[s,u](B_{1hn} + Rhn[s,u])^{-1}(mod\text{p})$**,**

where:

$A_{1h} \equiv J_{1hn} + 2hn[s,u] + \Delta(mod\text{p})$**,**
$A_{1hn} \equiv J_{1h} + 2h[s,u] + \Delta(mod\text{p})$**,**
$B_{1h} \equiv K_{1h} - 2h[s,u](mod\text{p})$**,**
$B_{1hn} \equiv K_{1hn} - 2hn[s,u](mod\text{p})$**,**
$J_{1h} \equiv \sum_{s=0,u=0}^{s=3,u=7} h[s,u](mod\text{p})$ **,**
$K_{1h} \equiv \prod_{s=0,u=0}^{s=3,u=7} h[s,u] + \Delta(mod\text{p})$,
$J_{1hn} \equiv \sum_{s=0,u=0}^{s=3,u=7} hn[s,u](mod\text{p})$ **,**
$K_{1hn} \equiv \prod_{s=0,u=0}^{s=3,u=7} hn[s,u] + \Delta(mod\text{p})$.

In the above expressions:

- For $A_{1h}J_{1hn}(mod2) \equiv 1$ or for $A_{1hn}J_{1h}(mod2) \equiv 1$, that is accepted ***Δ=0*** **,** otherwise accepted ***Δ=1***;
- For $K_{1h}h[s,u](mod2) \equiv 0$ or for $K_{1hn}hn[s,u](mod2) \equiv 0$, that is accepted $h[s,u] \equiv h[s,u] + 1(mod\text{p})$ *and* $hn[s,u] \equiv hn[s,u] + 1(mod\text{p})$ accordingly;
- For $K_{1h}\prod_{s=0,u=0}^{s=3,u=7} h[s,u] \neq \pm(J_{1h} + 1)(mod\ \text{p})$ or for $K_{1hn}\prod_{s=0,u=0}^{s=3,u=7} hn[s,u] \neq \pm(J_{1h} + 1)(mod\ \text{p})$ that is accepted ***Δ=0*** **,** otherwise accepted ***Δ=10***;

where:

***h*** – element of the ***holat*** array, ***hn*** – element of the ***holatn*** array.

$h[s,u]^{\setminus -2} \equiv -A_{2h}h[s,u]^{\setminus -1}(B_{2h} + Rh[s,u]^{\setminus -1})^{-1}(mod\text{p})$**,**

If $h[s,u]^{\setminus -2} \equiv 0(modp)$**,** then it is calculated $h[s,u]^{\setminus -2} \equiv A_{2h} + B_{2h}(mod\text{p})$

$hn[s,u]^{\setminus -2} \equiv -A_{2hn}hn[s,u]^{\setminus -1}(B_{2hn} + Rhn[s,u]^{\setminus -1})^{-1}(mod\text{p})$**,**

If $hn[s,u]^{\setminus -2} \equiv 0(mod\ \text{p})$**,** then it is calculated $hn[s,u]^{\setminus -2} \equiv A_{2hn} + B_{2hn}(mod\text{p})$**,**

$h[s,u]^{\setminus -3} \equiv -A_{2h}h[s,u]^{\setminus -2}(B_{2h} + Rh[s,u]^{\setminus -2})^{-1}(mod\text{p})$**.**

$hn[s,u]^{\setminus -3} \equiv -A_{2hn}hn[s,u]^{\setminus -2}(B_{2hn} + Rhn[s,u]^{\setminus -2})^{-1}(mod\text{p})$**,**

where:

$A_{2h} \equiv J_{1hn} + J_{2hn} + 2hn[s,u]^{\setminus -1} + \Delta(mod\text{p})$**,**
If $A_{2h} \equiv 0(mod\text{p})$**,** that is accepted $A_{2h} = 1$**,**
$A_{2hn} \equiv J_{1h} + J_{2h} + 2h[s,u]^{\setminus -1} + \Delta(mod\text{p})$**,**
If $A_{2hn} \equiv 0(modp)$**,** that is accepted $A_{2hn} = 1$**,**
$B_{2h} \equiv K_{1h}\,K_{2h} - 2h[s,u]^{\setminus -1}(mod\text{p})$**,**
$B_{2hn} \equiv K_{1hn}\,K_{2hn} - 2hn[s,u]^{\setminus -1}(mod\text{p})$**,**
$J_{2h} \equiv \sum_{s=0,u=0}^{s=3,u=7} h[s,u]^{\setminus -1}(mod\text{p})$ **,**
$K_{2h} \equiv \prod_{s=0,u=0}^{s=3,u=7} h[s,u]^{\setminus -1} + \Delta(mod\text{p})$.
$J_{2hn} \equiv \sum_{s=0,u=0}^{s=3,u=7} hn[s,u]^{\setminus -1}(mod\text{p})$ **,**
$K_{2hn} \equiv \prod_{s=0,u=0}^{s=3,u=7} hn[s,u]^{\setminus -1} + \Delta(mod\text{p})$.

In the above expressions:

- For $A_{2h}$ or $A_{2hn}J_{1h} + J_{2h}(mod2) \equiv 1$ **,** $J_{1hn} + J_{2hn}(mod2) \equiv 1$ **,** accordingly, then **Δ=0** is taken, otherwise **Δ=1**;
- For $K_{2h}h[s,u]^{\setminus -1}(mod2) \equiv 0$ **,** or $K_{2hn}hn[s,u]^{\setminus -1}(mod2) \equiv 0$ **,** then it is accepted $h[s,u]^{\setminus -1} \equiv h[s,u]^{\setminus -1} + 1(mod\text{p})$ or $hn[s,u]^{\setminus -1} \equiv hn[s,u]^{\setminus -1} + 1(mod\text{p})$ accordingly.
- For $K_{2h}\prod_{s=0,u=0}^{s=3,u=7} h[s,u]^{\setminus -1} \neq \pm(J_{2hn} + 1)(modp)$ or for $K_{2hn}\prod_{s=0,u=0}^{s=3,u=7} hn[s,u]^{\setminus -1} \neq \pm(J_{2h} + 1)(modp)$ then it is accepted **Δ=0**, otherwise **Δ=10**.

The results of the transformations are copied to the array **holat** and **holatn**.

***BaytZichlash(holat, holatn)*** **–** the transformation used in the hashing procedure over elements at the half-byte (byte) level of the current value of the **holat** array and the **holatn** array using the **XOR** operation if the module p=16, or based on a linear compression array into a single array if the module p=256.

If the length of the hashing key and the length of the input block are 256 bits, then in the **BaytZichlash(holat, holatn)** transformation, each of the **holat**, holatn arrays is transformed into the form of a linear array holat1, holatn1 at the byte level and bytes are replaced with half-byte elements of a linear

compression array of the order 1x256; the latter is a linear form of a compression table, columns, the rows and diagonal of which contains various elements from 0 to 15. The replacement procedure consists in replacing each byte of the same value with the address from 0...255 of the linear compression array with the cell element at this address.

As a result of the transformation, a pair of linear arrays **holat$_2$, holatn$_2$** of the order of 1x256 with elements at the half-byte level is formed. The conversion is completed by performing the concatenation **holat$_3$=holat$_2$||holatn$_2$** and converting **holat$_3$** into an array of order 1x256, with elements at the byte level, which are copied to the **holat** array of order 4x8.

**Aralash(holat, k$_e$)** is a transformation used in the hashing procedure based on the operation of multiplying diamatrix.

The multiplied diamatrix correspond one-to-one to the right and left halves of the square shape of the two-dimensional arrays **holat** and the step key **k$_e$**, respectively.

The transformation of **Aralash(holat, k$_e$)** is reduced to the following actions:

- copying a two-dimensional **holat** array into a pair of two-dimensional arrays $\boldsymbol{H_{ch}[4,4]}$ and $\boldsymbol{H_o[4,4]}$;
- calculating $\boldsymbol{H_{ch}' \equiv H_{ch} \otimes k_{ech}(mod p)}$ and assigning the result $\boldsymbol{H_{ch}'}$ to the array $\boldsymbol{H_{ch}}$;
- defining the result for $\boldsymbol{s,u \in 0,1,2,3}$ in the form:

$\boldsymbol{h_{ch}' \equiv h_{ch}[4,4]x\sum_{i=0}^{3} k_{ech}[i,u] - \sum_{i=0,i\neq u}^{3} h_{ch}[i,i]xk_{ech}[i,u](modp)}$,

$\boldsymbol{h_{ch}'[s,u]_{s\neq u} \equiv h_{ch}[s,u]x\sum_{i=0}^{3} k_{ech}[i,u] + k_{ech}[s,u]x\sum_{i=0}^{3} h_{ch}[i,u] - \sum_{i=0,i\neq s,u}^{3} h_{ch}[s,i]xk_{ech}[i,u](modp)}$,

where: $\boldsymbol{h_{ch}[s,u]}$ is the $\boldsymbol{H_{ch}}$ element; $\boldsymbol{k_{ech}[s,u]}$ is the $\boldsymbol{k_{ech}}$ element that belongs to the left half of the array $\boldsymbol{k_e[4,8]}$;

- calculation of $\boldsymbol{H_o' \equiv H_o \otimes k_{eo}(mod p)}$ and assignment of the result of $\boldsymbol{H_o'}$ to the array of $\boldsymbol{H_o}$;
- defining the result for $\boldsymbol{s,u \in 0,1,2,3}$ in the form:

$\boldsymbol{h_o' \equiv h_o[u,u]x\sum_{i=0}^{3} k_{eo}[i,u] - \sum_{i=0,i\neq u}^{3} h_o[i,i]xk_{eo}[i,u](modp)}$,

$\boldsymbol{h_o'[s,u]_{s\neq u} \equiv h_o[s,u]x\sum_{i=0}^{3} k_{eo}[i,u] + k_{eo}[s,u]x\sum_{i=0}^{3} h_o[i,u] - \sum_{i=0,i\neq s,u}^{3} h_o[s,i]xk_{eo}[i,u](modp)}$.

where: $\boldsymbol{h_o[s,u]}$ is the $\boldsymbol{H_o}$ element; $\boldsymbol{k_{eo}[s,u]}$ is the $\boldsymbol{k_{eo}}$ element that belongs to the right half of the array $\boldsymbol{k_e[4,8]}$;

- copying the $\boldsymbol{H_{ch}}$ and $\boldsymbol{H_o}$ arrays to the **holat** array.

**SurHolat(holat)** - the transformation used in the hashing procedure, which is carried out over the **holat** array, consists in cyclic shifts of all four rows of the **holat** array horizontally and vertically by different shift values.

The zero row of the **holat** array is cyclically shifted to the right by 1 half-byte (byte), the first row by 2 half-bytes (byte), the second row by 3 half-bytes (byte), the third row by 4 half-bytes (byte), and the 0 and 4 columns of the resulting array are cyclically shifted down by 1 half-byte (byte), the first and the fifth column is 2 half-bytes (bytes), the second and sixth columns are 3 half-bytes (bytes);

**SurKalit (k$_e$)** - the transformation used in the hashing procedure, which is carried out over the **k$_e$** array, consists in cyclic shifts of all four rows of the **k$_e$** array horizontally and vertically by different shift values.

The zero row of the **k$_e$** array is cyclically shifted to the right by 4 half-bytes (bytes), the first row by 3 half-bytes (bytes), the second row by 2 half-bytes (bytes), the third row by 1 half-byte (bytes), and the first and fifth columns of the resulting array are cyclically shifted up by 3 half-bytes (bytes), the second and the sixth column is 2 half-bytes (bytes), the third and seventh columns are 1 half-byte (bytes);

**TuzilmaKalit(k$_e$, k)** is a transformation used at the end of each stage of the hashing procedure, which is performed over each half-byte (byte) of the **k$_e$** array in order to bring its structure to the structure of the original hashing key **k**; the result of this transformation satisfies the conditions of reversibility of each of the square parts of the **k$_e$** array.

The **TuzilmaKalit(k$_e$,k)** transformation is performed over each half-byte (byte) of the **k$_e$** array in order to bring its structure to the structure of the original hashing key **k**.

For $\boldsymbol{0 \le s < 4,\ 0 \le u < 8}$ **,**

if $\boldsymbol{k_e[s,u] \equiv 0(mod\ p)}$ and $\boldsymbol{k[s,u] \equiv 0(mod\ 2)}$ then it is accepted $\boldsymbol{k_e[s,u] = 2}$

if $\boldsymbol{k_e[s,u] \neq 0(mod\ p)}$ and $\boldsymbol{k_e[s,u](mod\ 2) \equiv k[s,u](mod\ 2)}$ then it is accepted $\boldsymbol{k_e[s,u] = k_e[s,u]}$ otherwise accepted $\boldsymbol{k_e[s,u] \equiv k_e[s,u] - 1(mod\ p)}$.

The result of this transformation satisfies the conditions of reversibility of each of the square parts of the $\boldsymbol{k_e}$ array (Mukhammadovich and Djuraevich 2023).

## 2.2 Blockchain development

Blockchain development is a complex and multifaceted task.

The choice of consensus mechanism determines how transactions are verified and added to the blockchain.

The data structure for the blockchain has been developed, including how transactions are organized

into blocks, and the format of each block. As a rule, a block contains a header, a list of transactions and a link to the previous block ("hash").

You need to create the first block in the blockchain, known as the "genesis block". This block has no reference to the previous block and serves as the basis of the chain.

Implementation of cryptographic algorithms to protect transactions and blocks. This includes using hash functions to ensure data integrity, digital signatures to verify transactions, and other cryptographic methods.

Creating a peer-to-peer (P2P) network that allows nodes to interact and share the blockchain. Nodes in the network should be able to verify transactions and reach consensus on the state of the blockchain.

A mechanism for authentication and verification of transactions has been developed. This includes verifying digital signatures, ensuring that the sender has sufficient funds and confirming that the transaction complies with the rules of the blockchain. It is necessary to emphasize the mechanism for checking and adding blocks to the blockchain (Anusuya and Karthika, 2023).

Blockchain development is an important undertaking, and it often requires a team of qualified developers, security experts and network administrators. The steps described here are a basic roadmap, but blockchain development in the real world can be significantly more complex, especially for large-scale production systems (Zhang, 2021).

# 4 DISCUSSION OF RESULTS

In this section, we will delve into the findings of our study, which focused on the integration of hashing algorithms in blockchain technology and the resulting implications for data security, integrity, and performance. The aim of our research was to explore the unbreakable link between hashing algorithms and blockchain, and here, we present a comprehensive analysis of the results.

Our study reaffirmed the critical role that hashing algorithms play in ensuring data security on the blockchain. These algorithms, due to their mathematical properties, make it practically impossible for malicious actors to alter stored data, thus preserving the immutability of the blockchain.

The results of our study demonstrate that the use of hashing algorithms guarantees data integrity and immutability. Hashes created from data are unique, and even the slightest change in the original data results in a significantly different hash. This property, combined with the decentralized and distributed nature of blockchain, makes it an unbreakable link. Once data is added to the blockchain, it becomes practically impossible to tamper with, providing the trust and reliability that blockchain technology is known for.

The security implications of our findings are profound. Blockchain, with its reliance on hashing algorithms, offers robust protection against various security threats, including data breaches and unauthorized alterations. Our study revealed that the use of strong cryptographic hashes ensures the confidentiality, integrity, and authenticity of data, making blockchain one of the most secure methods for data storage and transfer.

In terms of scalability, our research found that the chosen hashing algorithms did not significantly hinder the performance or scalability of blockchain networks. While some computational resources are required to compute hashes, the trade-offs between security and efficiency were found to be manageable. The use of parallel processing and optimized algorithms can further mitigate resource consumption.

Several case studies showcased real-world applications of the unbreakable link between hashing algorithms and blockchain. Use cases in finance, supply chain, and healthcare demonstrated the practical benefits of integrating these technologies. Notable examples include the use of blockchain to secure medical records, enhance supply chain transparency, and provide secure financial transactions.

Our study revealed a clear advantage of blockchain over traditional databases in terms of data security and immutability. The integration of hashing algorithms in blockchain technology offers a superior level of trust and security, which is challenging to achieve in traditional database systems.

While our study provided valuable insights into the relationship between hashing algorithms and blockchain, there are avenues for further research. Future investigations could explore more efficient algorithms or delve into the ethical and regulatory aspects of data security in blockchain technology.

Our study underlines the irrefutable link between hashing algorithms and blockchain, which results in a highly secure, immutable, and trustworthy environment for data storage and transactions. This unbreakable link paves the way for enhanced data security and integrity in a variety of industries. It is our hope that these findings will inform and inspire further developments in the ever-evolving field of blockchain technology (Aloev and Nurullaev, 2019).

The practical implications of our results are substantial. We recommend organizations and policymakers to embrace the integration of hashing algorithms in blockchain as a means to enhance data security and establish trust. Businesses should consider adopting blockchain technology to leverage the inherent security features, and its adoption while addressing potential ethical concerns (Mollah and Jun, 2020).

# 5 CONCLUSIONS

The unbreakable link between Hashing Algorithms and Blockchain represents a symbiotic relationship that underpins the core principles of data integrity, security, and trust within decentralized ledger technology. This synthesis of cryptographic hash functions and blockchain architecture has far-reaching implications for modern cybersecurity and the facilitation of trustless transactions. In conclusion, it is evident that this inseparable connection has ushered in a new era of digital security and transparency.

Hashing algorithms, renowned for their ability to transform data into fixed-length hashes, act as the linchpin for data integrity within the blockchain. Their cryptographic strength ensures that once information is recorded, it remains impervious to tampering, forging an immutable record of transactions. The integration of hashing algorithms into the blockchain's framework not only guarantees the security of data but also allows for efficient transaction verification, validating the authenticity of participants and their actions.

Conversely, blockchain technology, with its transparent, decentralized ledger, has revolutionized a myriad of industries by obviating the need for intermediaries and engendering trust among participants. It is the amalgamation of hashing algorithms that safeguards this trust. Through the process of hashing, transactions are cryptographically secured, and their immutability is assured. This ensures that the ledger's participants can transact, exchange assets, and execute smart contracts with confidence, mitigating the risks of fraud and data manipulation.

The practical implications of this unbreakable link extend across various domains, from cryptocurrency transactions to supply chain management, identity verification, and beyond. The promise of trustless, decentralized systems is realized, enabling innovative solutions in domains as diverse as finance, healthcare, and voting systems. This link creates a foundation for secure, transparent, and auditable systems that foster efficiency and confidence.

# REFERENCES


Wang, H., Wu, T., 2017. Cryptography on the Blockchain. Nanjing Youdian Daxue Xuebao (Ziran Kexue Ban). *Journal of Nanjing University of Posts and Telecommunications (Natural Science).* 37. 61-67.

Shrivas, M., 2019. The Disruptive Blockchain: Types, Platforms and Applications. *Texila International Journal of Academic Research*. 17-39.

Agung, A. A. G., Handayani, R., 2020. Blockchain for Smart Grid. *Journal of King Saud University - Computer and Information Sciences.* 34.

Nurullaev, M., Aloev, R. D., 2020. Software, algorithms and methods of data encryption based on national standards. *IIUM Engineering Journal.*

Miglani, A., Kumar, N., Chamola, V., Zeadally, Sh., 2020. Blockchain for Internet of Energy management: Review, solutions, and challenges. *Computer Communications.* 151. 395-418.

"Information technology. Cryptographic protection of information. Hash function," O`z DSt 1106:2009. 2009. Accessed: Mar. 16, 2023 [Online]. Available: https://lex.uz/files/5011555.zip

Mukhammadovich, N. M., Djuraevich, A. R., 2023. Working with cryptographic key information. *International Journal of Electrical and Computer Engineering.*

Anusuya, R., Dhanaraj, K.R., Kumar, L. A., 2023. Privacy-preserving in smart grids using Ethereum and Hyperledger blockchain.

Zhang, M., Zhu, H., Jiang, K., 2021. Secure Cross-Domain Data Sharing Technology Based on Blockchain.

Aloev, R. D., Nurullaev, M. M., 2019. Cryptography service provider – data encryption, *Conference on Complexity, Informatics and Cybernetics*, Orlando, pp. 127–131.

Mollah, M.B., Zhao, J., Niyato, D., Lam, K.-Ya., Zhang, X., Ghias, M., Amer M. Yu., Koh, L.H., Yang, L., 2020. Blockchain for Future Smart Grid: A Comprehensive Survey. *IEEE Internet of Things Journal.* 1-26.