Security and Vulnerability in Networks Assignment 3

BLOCKCHAIN

Abstract

This project presents a simplified blockchain model that highlights core elements of decentralized ledger technology, including secure transaction verification, data immutability, and a basic consensus mechanism. Through the implementation of key components (Wallet, Transaction, Block, and Blockchain) this system demonstrates how distributed data management can overcome the limitations of centralized systems, such as single points of failure and lack of transparency. Findings indicate that the blockchain's cryptographic techniques, particularly SHA-256 hashing and ECDSA, ensure data integrity and authenticity, supporting tamper-resistant, trustless interactions. While effective in a controlled setting, this model could be enhanced with a full consensus mechanism and smart contract capabilities to align more closely with real-world blockchain applications. The project concludes that blockchain's decentralized structure holds significant potential for industries requiring secure, transparent, and autonomous transaction management, recommending further development to realize its full capabilities.

1. Introduction.

Blockchain technology represents a transformative approach to data management, offering a decentralized, secure, and transparent way to record and verify transactions. Unlike traditional centralized systems that rely on a single authority for control, blockchain distributes this responsibility across a network of participants, eliminating single points of failure and reducing the risk of manipulation. By recording data in an immutable, shared ledger, blockchain fosters trust among users without the need for intermediaries, laying a foundation for secure, tamper-resistant interactions in various industries, from finance to supply chains and beyond.

Background:

The motivation behind creating a decentralized ledger system lies in addressing the limitations of traditional centralized systems, which are prone to single points of failure, data breaches, and limited transparency. In centralized models, a single authority manages data and verifies transactions, making the system vulnerable to hacks, downtime, or internal manipulation. Additionally, centralized systems often lack transparency, as control is restricted to a few trusted parties, raising concerns over data privacy and accountability.

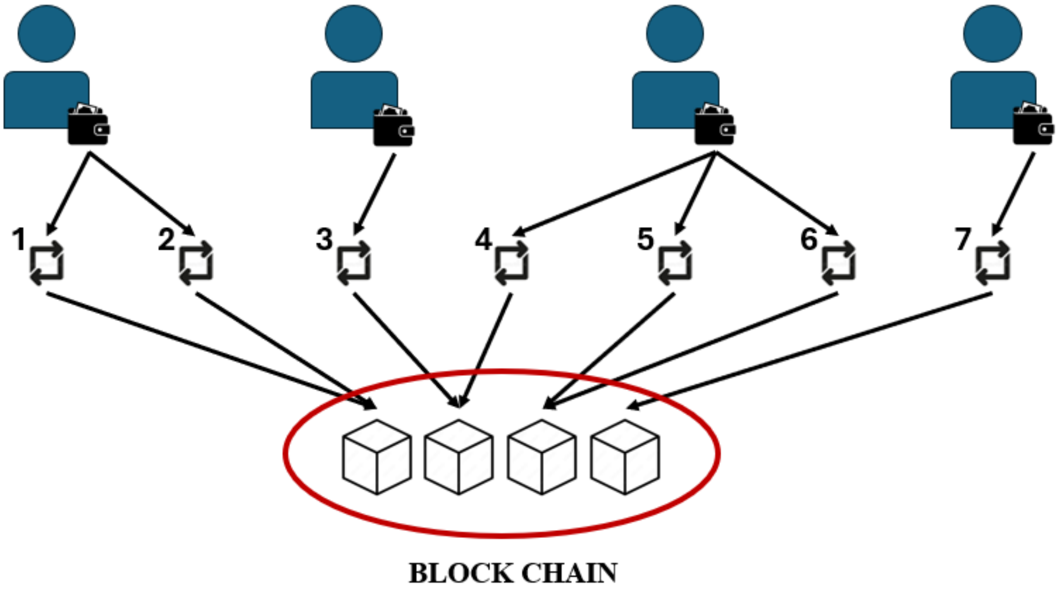
A decentralized ledger system, like blockchain, overcomes these limitations by distributing control across a network of independent participants. Each transaction is verified, recorded, and encrypted in blocks that link together to form an immutable chain. This decentralized structure not only minimizes dependency on a single authority but also provides a secure, transparent, and tamper-resistant way to manage digital transactions.

Objective:

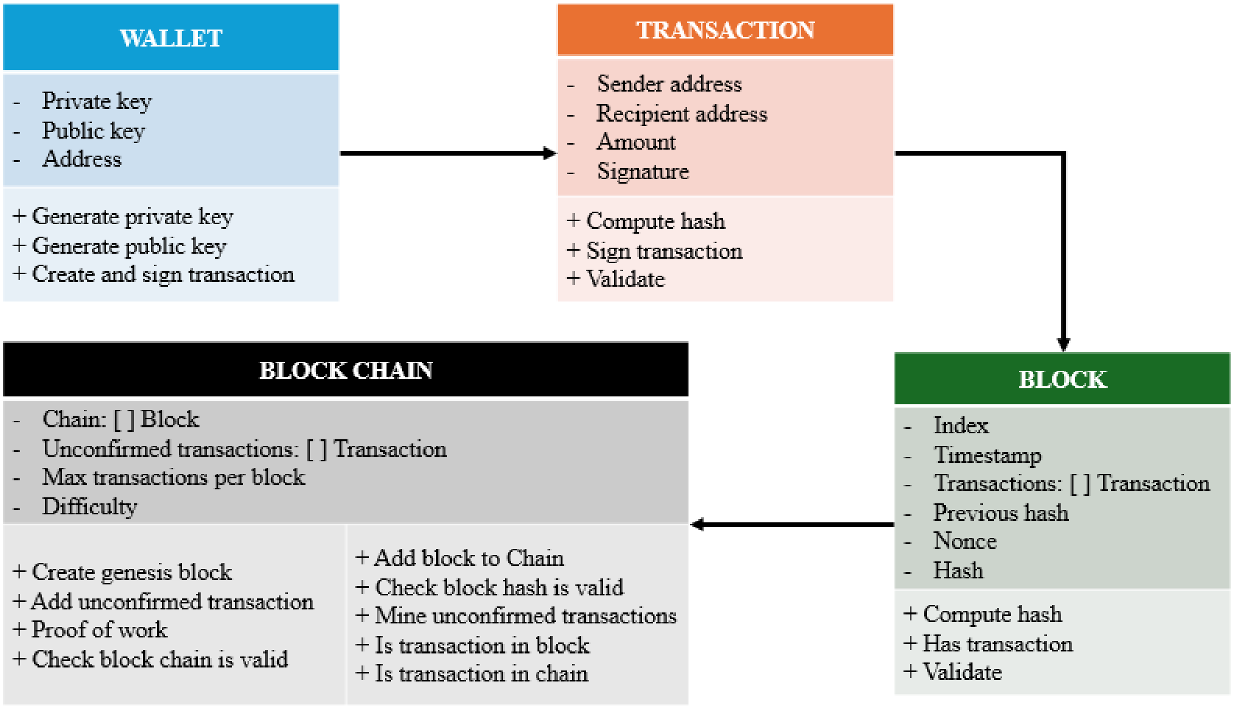
Therefore, the goal of this blockchain system is to enhance security, ensure transparency, and enable trustless transactions, where users can interact without needing to rely on a central authority. Through cryptographic techniques for transaction verification and consensus, this model aims to create a secure and transparent ledger. Trustless interactions allow participants to independently verify and authenticate data, making the system resilient against unauthorized modifications. Ultimately, this blockchain project aims to lay the groundwork for a reliable, decentralized platform that strengthens data integrity, fosters trust, and demonstrates the potential for blockchain technology to reshape data management across diverse applications.

2. System Architecture

To understand the architecture of the program, one must first comprehend the organization of the elements that make up the blockchain. Users generate transactions through their wallets. These transactions form blocks, which in turn form the chain.



The system architecture for this blockchain project consists of four main components: **Wallets**, **Transactions**, **Blocks**, and the **Blockchain** itself. Together, these components create a robust, decentralized ledger system that records value transfers in a secure, immutable, and tamper-resistant way. Each component serves a specific function, contributing to the overall integrity and functionality of the blockchain.



WALLET

- Private key
- Public key
- Address

+ Generate private key
+ Generate public key
+ Create and sign transaction

TRANSACTION

- Sender address
- Recipient address
- Amount
- Signature

+ Compute hash
+ Sign transaction
+ Validate

BLOCK CHAIN

- Chain: [ ] Block
- Unconfirmed transactions: [ ] Transaction
- Max transactions per block
- Difficulty

+ Create genesis block
+ Add unconfirmed transaction
+ Proof of work
+ Check block chain is valid

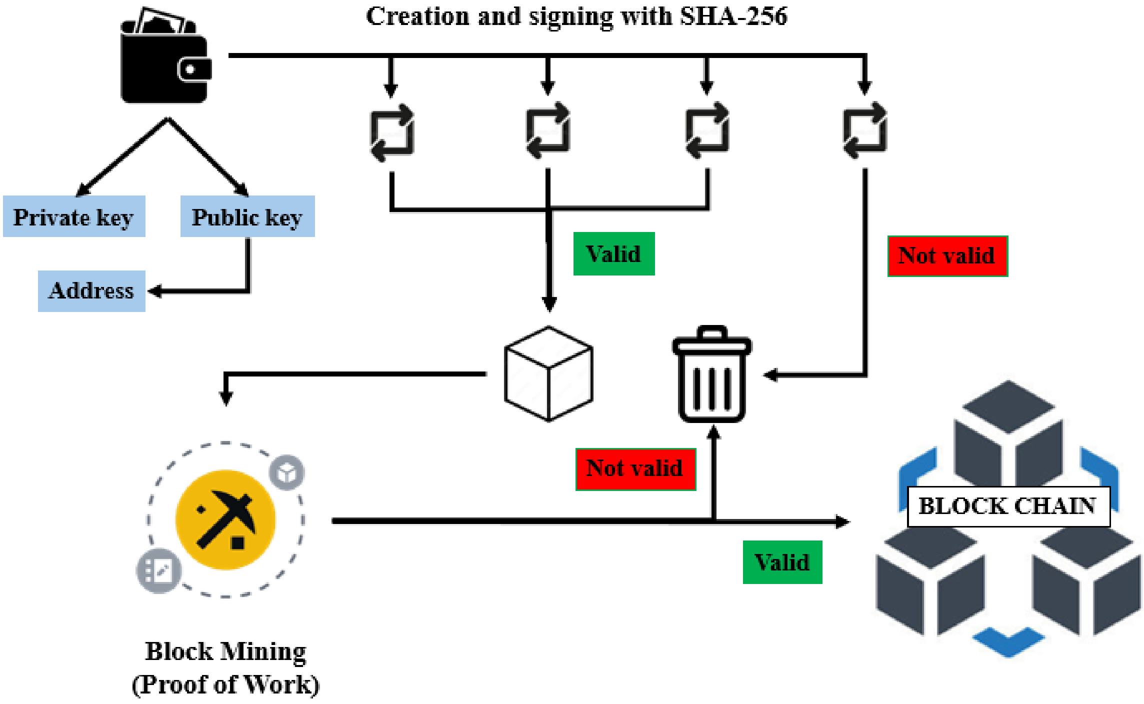
+ Add block to Chain
+ Check block hash is valid
+ Mine unconfirmed transactions
+ Is transaction in block
+ Is transaction in chain

BLOCK

- Index
- Timestamp
- Transactions: [ ] Transaction
- Previous hash
- Nonce
- Hash

+ Compute hash
+ Has transaction
+ Validate

This architecture enables a seamless interaction between all components, establishing an immutable and distributed ledger that is secure, transparent, and resistant to tampering. Cryptographic principles, such as digital signatures and hashing, work in tandem with the PoW mechanism to guarantee the blockchain's reliability and decentralization. Together, these components create a trustworthy platform for secure, peer-to-peer transactions without relying on intermediaries.



3. Cryptographic Foundations

The security and integrity of this blockchain system are built upon two primary cryptographic algorithms: **SHA-256** and **ECDSA** (Elliptic Curve Digital Signature Algorithm) **with the SECP256k1 curve**. These algorithms enable data integrity, authenticity, and immutability within the blockchain.

3.1 SHA-256: Hashing for Data Integrity

The SHA-256 (Secure Hash Algorithm 256-bit) hashing function is used extensively in the blockchain to ensure data integrity. A hash is a fixed-size string of characters that is generated from input data of any size. The SHA-256 algorithm produces a 64-character hexadecimal hash, which acts as a unique identifier for data within each block.

This hashing mechanism is crucial for data integrity and immutability within blockchain due to the following properties:

- **Deterministic**: SHA-256 always produces the same output for a given input. In blockchain, this property ensures that each block's data consistently generates the same hash, providing a reliable digital fingerprint for block validation. Determinism guarantees that all nodes will compute the same hash for identical block data, achieving consensus.

- **Pre-image Resistance**: Given a SHA-256 hash, it is computationally infeasible to reverse-engineer the original input data. This one-way property protects sensitive data in blockchain, as it prevents anyone from deducing transaction details or other block contents solely from the hash. Pre-image resistance secures transactions by ensuring that hashed data cannot be deciphered without authorization.

- **Collision Resistance**: Collision resistance ensures that it is practically impossible for two different inputs to produce the same hash. In blockchain, this prevents attacks in which a malicious user could substitute a valid block with a fake one by crafting different data that hashes to the same value. This property thus guarantees the uniqueness of each block's hash, securing the blockchain's immutability and data uniqueness.

- **Speed and Efficiency**: SHA-256 can hash large amounts of data quickly, making it well-suited for blockchain applications, where each block may contain numerous transactions and metadata. The efficiency of SHA-256 contributes to the timely validation and addition of blocks to the blockchain, supporting network performance.

These cryptographic properties of SHA-256 form the basis for the key uses of SHA-256 in this blockchain:

- **Block Hashing:** Every block in the blockchain contains a unique hash generated by applying SHA-256 to the block's header data, including the index, timestamp, transactions, nonce, and hash of the previous block. This unique hash serves as a "digital fingerprint" of the block. Any alteration to the block data would result in a completely different hash, alerting the system to potential tampering.

- **Linking Blocks:** The hash of each block is included in the header of the subsequent block, linking them in a cryptographic chain. If a block is modified, its hash changes, breaking the link with the next block and invalidating the blockchain. This chain-linking mechanism ensures immutability.

- **Proof of Work (PoW):** SHA-256 is also integral to the Proof of Work mechanism. To add a new block, miners must find a nonce (a variable added to the block data) that, when hashed with the block header, produces a hash with a specific number of leading zeros. This difficulty adjustment regulates the rate of block addition, making it computationally difficult to alter data or tamper with the blockchain.

3.2 ECDSA with SECP256k1: Digital Signatures for Authentication

The Elliptic Curve Digital Signature Algorithm (ECDSA) is employed within the blockchain to secure transactions and authenticate users. This project uses the SECP256k1 elliptic curve due to its efficiency and strong security properties. ECDSA with SECP256k1 is used to generate private and public keys, allowing users to sign and verify transactions. Here's how this works within the system:

- **Private and Public Key Generation:** Each wallet generates an ECDSA key pair using SECP256k1. The private key is kept secure by the wallet owner and is used to sign transactions. The public key, derived from the private key, serves as the user's blockchain address. This address is openly shared, allowing other nodes to verify the authenticity of signed transactions without revealing the private key.

- **Transaction Signing:** When a user initiates a transaction, the wallet uses the private key to create a digital signature for the transaction data, specifically the transaction's hash. This signature is unique to both the transaction and the user's private key, making it impossible for anyone without the private key to generate a valid signature for that transaction. The signature serves as proof that the transaction originated from the rightful owner.

- **Signature Verification:** Each node in the blockchain verifies the digital signature of every transaction before including it in a block. The node uses the sender's public key to verify the signature against the transaction data. This ensures that only transactions signed by the authorized private key can be accepted by the blockchain, preventing unauthorized access or modifications.

The combination of ECDSA for transaction signing and SHA-256 for hashing enables a highly secure system. ECDSA ensures that only legitimate transactions are recorded, while SHA-256 maintains the immutability of the blockchain by preventing any alteration of block data without detection.

4. Core Components and Functionality

As already said, this blockchain system comprises four essential components: **Wallet**, **Transaction**, **Block**, and **Blockchain**. Each component is designed to perform specific tasks that ensure secure, verified transactions, chain integrity, and decentralized trust.

4.1 Wallet

- Function:

The Wallet serves as the user's interface with the blockchain, managing cryptographic keys (private and public) that allow secure transaction signing and verification. Each wallet represents a unique identity on the blockchain and enables users to initiate transactions.

- Operation:

- **Key Generation:** The wallet generates an ECDSA key pair (private and public keys) using the SECP256k1 curve. The private key, stored securely within the wallet, is used to sign transactions, while the derived public key serves as the wallet's address.

- **Transaction Creation:** Users initiate transactions by specifying a recipient's address and the amount to transfer. The wallet signs the transaction with the private key, generating a digital signature that guarantees the transaction's authenticity and integrity.

- **Digital Signature:** When a transaction is signed with the private key, it produces a unique signature tied to both the transaction and the wallet. This signature allows nodes to verify the transaction's validity using the wallet's public key without needing access to the private key itself.

4.2 Transaction

- Structure:

Transactions record the details of value transfers between wallets. Each transaction includes the following fields:

✓ **Sender Address:** The address (public key) of the wallet initiating the transaction.

✓ **Recipient Address:** The address of the wallet receiving the transaction.

✓ **Amount:** The quantity of value being transferred.

✓ **Signature:** A unique digital signature created by the sender's private key, proving the transaction's authenticity and linking it to the sender's wallet.

- Validation:

- **Signature Verification:** Nodes validate each transaction by checking the signature with the sender's public key. This ensures that only the legitimate owner of the private key could have signed the transaction, confirming its authenticity.

- **Sufficient Funds Check:** Before including a transaction in a block, nodes also check that the sender has enough funds for the transfer. This prevents double-spending and ensures that only valid transactions enter the blockchain.

- **Transaction Storage:** Validated transactions are stored in the pool of unconfirmed transactions, awaiting inclusion in a block through the mining process.

4.3 Block

- Composition:

Each block is a container that organizes multiple validated transactions and stores essential metadata for linking within the blockchain:

- ✓ **Index:** A unique identifier indicating the block’s position in the chain.
- ✓ **Timestamp:** The time at which the block was created.
- ✓ **Transactions:** A list of validated transactions, grouped together to form a complete block of data for the chain.
- ✓ **Nonce:** A number used to complete the Proof of Work (PoW) challenge, which is essential for block validation.
- ✓ **Previous Hash:** The hash of the preceding block, which links the current block to the last block in the chain.

- Hashing:

The Block component uses SHA-256 hashing to secure each block’s data, ensuring immutability and integrity in the blockchain. SHA-256 generates a unique 64-character hash for each block by combining key elements (transactions, timestamp, nonce, and the previous block’s hash). This "digital fingerprint" changes completely if any block data is altered, making modifications immediately detectable.

The **Proof of Work (PoW)** process also relies on SHA-256. To mine a block, miners adjust the nonce until the SHA-256 output meets a specific difficulty level (typically a certain number of leading zeros in the hash). This requirement adds computational complexity, making it resource-intensive to alter block data, thus securing each block against tampering and reinforcing the blockchain’s integrity.

By using SHA-256 for hashing, each block in the blockchain is cryptographically secured, linked to the previous block, and validated through PoW, ensuring the immutability and resilience of the entire blockchain system.

4.4 Blockchain

- Chain Formation:

The blockchain is a sequence of blocks arranged in chronological order, with each block containing a reference to the previous one. This structure starts with the Genesis block, which has no predecessor and acts as the foundation of the chain. Every node on the network maintains a complete copy of the blockchain, ensuring decentralized distribution and preventing reliance on a central authority.

- **Mining Process:** Miners create new blocks by gathering a subset of unconfirmed transactions, validating them, and performing the PoW required to find a valid hash. Once the PoW challenge is solved, the block is validated by other nodes and added to the chain.

- **Addition of Blocks:** When a new block is mined and validated, it is appended to the chain with a reference to the previous block's hash. This sequential linking maintains the chain's structure and confirms the chronological order of transactions.

- Integrity Maintenance:

The integrity of the chain is maintained thanks to the following factors:

- **Validation of Blocks:** Each node in the blockchain independently validates blocks by verifying that the block's hash matches its computed hash and that each block references the correct hash of the previous block.

- **Consensus Mechanism:** In this decentralized system, nodes work together to reach consensus on the chain's validity. Nodes follow the same PoW rules and verify blocks independently, ensuring consistency across the network.

- **Decentralization and Security:** The blockchain's decentralized structure enhances its resilience, as each node holds a full copy of the chain. If a node tries to alter the blockchain, other nodes can easily identify the discrepancy and reject the manipulated version. This resilience reinforces the integrity and security of the entire blockchain.

Summary

Each core component (Wallet, Transaction, Block, and Blockchain) supports the functionality and security of the blockchain system. The **Wallet** enables secure transaction creation and signing, while the **Transaction** component documents and validates the transfer of value between wallets. The **Block** component groups validated transactions, using hashing and PoW to secure the block and link it immutably within the chain. Finally, the **Blockchain** itself manages the orderly addition of blocks, verifies the chain's integrity, and maintains a decentralized network through consensus. Together, these components establish a secure, transparent, and tamper-resistant environment for peer-to-peer transactions on the blockchain.

5. Consensus Mechanism (Simplified)

Given the assignment's scope, a full, decentralized consensus mechanism is excluded. Instead, this simplified model assigns the responsibility of block creation to a single miner. This miner alone validates transactions, solves the Proof of Work (PoW) puzzle, and appends blocks to the blockchain. In this trusted environment, where no other participants validate blocks, the blockchain can function without requiring agreement from multiple nodes. This approach enables a more straightforward setup, ideal for experimental use, though it lacks the security of decentralized consensus.

- Steps in the Single Miner Process:

- **Transaction Collection and Validation:** The miner gathers pending transactions from the pool of unconfirmed transactions. Each transaction undergoes a validation process to ensure authenticity and integrity, including verifying digital signatures to confirm that only the owner of a wallet's private key could have authorized the transaction.

- **Proof of Work (PoW) Execution:** Once transactions are validated, the miner groups them in a new block and begins the PoW process. The block also includes the previous hash from the last block in the chain, ensuring it is linked securely. PoW requires finding a nonce that, when combined with the block's data (including transactions, previous block hash, and timestamp), produces a hash (with SHA-256) meeting the required difficulty criteria (a certain number of leading zeros). The miner continuously adjusts the nonce and rehashes the block's data until a valid hash is produced. This computational challenge, even when done by a single miner, deters tampering by making each block addition resource-intensive.

- **Block Addition:** After solving the PoW puzzle, the miner encapsulates the block and the computed hash, called 'proof'. After checking the validation of the previous hash of the block and the proof (only checked by the single miner), the block is finally appended to the blockchain, becoming the latest part of the chain.

- **Immediate Acceptance Without Decentralized Verification:** Since this model assumes a trusted environment with a single miner, the new block is accepted immediately without further verification by other nodes. In typical decentralized systems, consensus among nodes would be required to validate and accept the block, but here, the single-miner model streamlines block acceptance without this additional layer of security.

This simplified consensus mechanism is effective for environments where the miner is assumed to be trustworthy, and the risk of tampering is low. While efficient, this setup lacks the robustness of decentralized consensus mechanisms found in production blockchains, where multiple nodes independently verify blocks to prevent single points of failure and resist attacks.

- Potential Extensions for Decentralization:

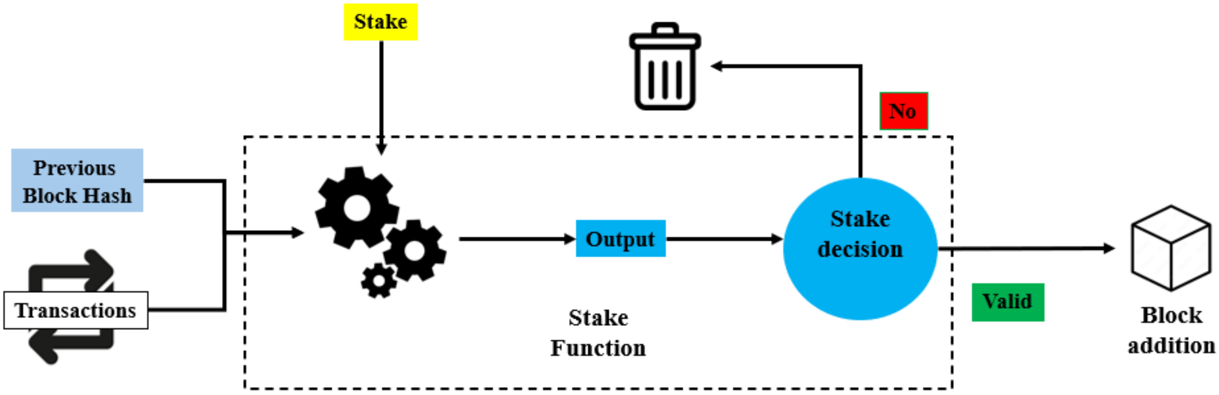
To enhance security and resilience, future iterations of this system could incorporate a more decentralized consensus mechanism:

- **Distributed Proof of Work (PoW):** Adding multiple miners would increase security by distributing control. Miners would compete to solve the PoW puzzle, making it difficult for any single entity to control block validation and increasing the network's resistance to tampering. Also, the validation and addition of a block to the chain would require consensus among nodes.

- **Proof of Stake (PoS):** PoS could be implemented to select validators based on their stake or holdings within the network, reducing the computational load of PoW.

  It is a consensus mechanism that selects validators based on the amount of cryptocurrency they hold (their "stake") rather than their computational power, making it a more energy-efficient alternative to Proof of Work (PoW). In PoS, users become validators by locking up a portion of their funds as collateral. This stake incentivizes validators to act honestly, as dishonest behaviour could result in losing part or all of their staked funds.

  Validators are selected to add new blocks either randomly or proportionally to their stake, so those with a higher stake have a better chance of being chosen. Once selected, validators earn rewards, such as transaction fees or new tokens, for confirming a block. Unlike PoW, PoS does not require heavy computations, drastically reducing energy usage.

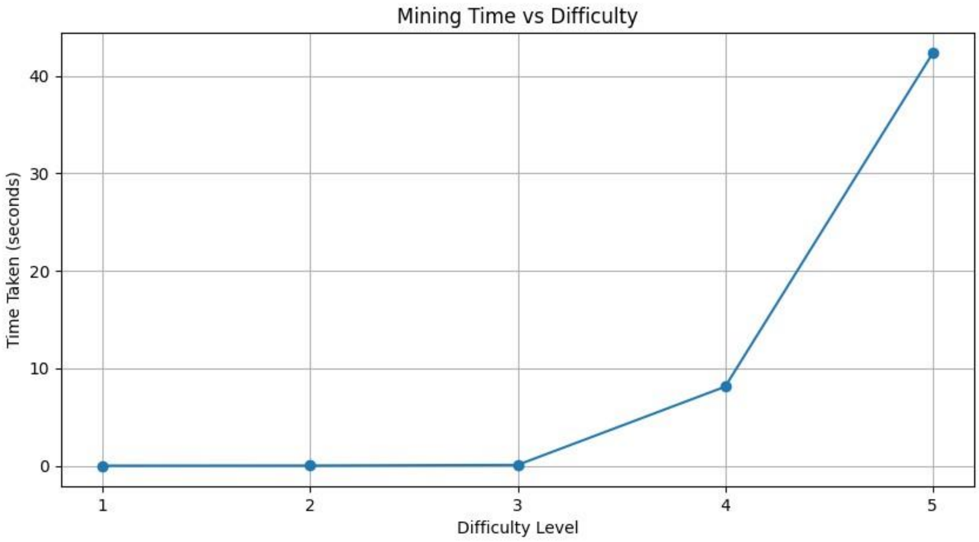


Proof of Stake Flow

6. Experimental Analysis

6.1 Mining Speed:

To understand how the difficulty level affects the time required to mine a block, a mining speed test has been implemented. This test performs the mining process on the same block with different difficulty levels (from 1 to 5) and finally displays the results in a graph.



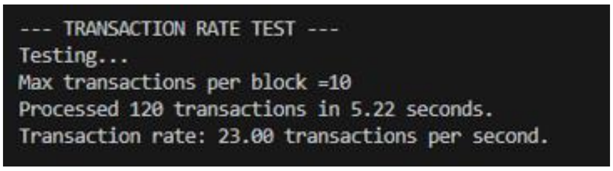
The results of the mining speed test reveal a clear pattern in the relationship between the difficulty level and the time required to mine a block. Starting with difficulty **1**, the mining time is almost negligible, which indicates that finding a valid hash at this level is very straightforward. When the difficulty is increased to **2 and to 3**, the mining time slightly increases but it is a negligible difference, demonstrating a minor impact on mining speed.

The transition from difficulty **3** to difficulty **4** reveals a dramatic increase in time. This increase represents a greater computational effort required to find a valid hash, illustrating that the rise in difficulty has a compounding effect. Mining time continues to increase, hitting more than **40 seconds** at difficulty **5**, further demonstrating the exponential nature of the relationship between difficulty and mining time.

This analysis highlights that as difficulty increases, the time required to mine a block grows in a non-linear manner. This trend is consistent with the concept of proof of work, where a higher difficulty implies finding a hash with more leading zeros, necessitating more iterations and computational effort.

6.2 Throughput:

A function has also been programmed to measure the number of transactions processed within a given time and, based on this, calculate the number of transactions processed per second. In this way, we obtain a measure of the blockchain's performance:

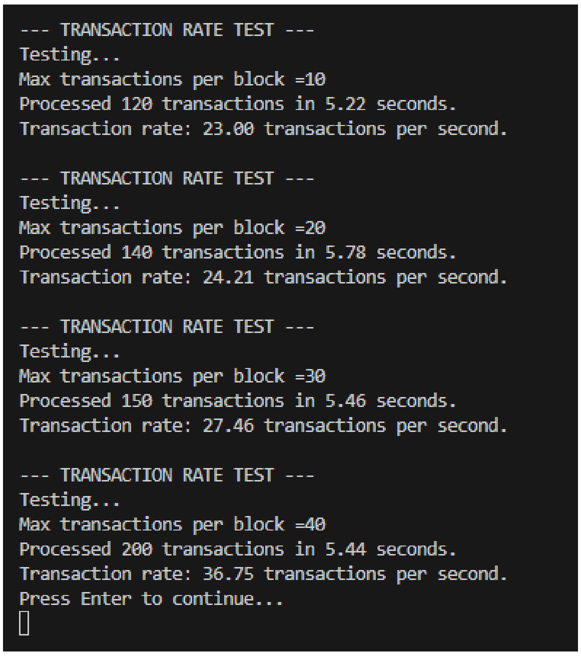


```
--- TRANSACTION RATE TEST ---
Testing...
Max transactions per block =10
Processed 120 transactions in 5.22 seconds.
Transaction rate: 23.00 transactions per second.
```

The results of the transaction rate test provide insight into the blockchain's capacity to process transactions efficiently. In this test, the system processed **120 transactions in 5 seconds**, yielding a transaction rate of **23.00 transactions per second**.

This transaction rate indicates the throughput of the blockchain under current conditions and serves as a baseline for evaluating scalability and performance. The transaction rate of **23.00 transactions per second** suggests that the blockchain can handle a moderate volume of transactions within a short time frame, but this rate may vary depending on system conditions, such as network load, block size limits, and the complexity of each transaction.

The previous test was conducted with a limitation of **10 transactions per block.** If this value is progressively increased, the following results are observed.



```
--- TRANSACTION RATE TEST ---
Testing...
Max transactions per block =10
Processed 120 transactions in 5.22 seconds.
Transaction rate: 23.00 transactions per second.

--- TRANSACTION RATE TEST ---
Testing...
Max transactions per block =20
Processed 140 transactions in 5.78 seconds.
Transaction rate: 24.21 transactions per second.

--- TRANSACTION RATE TEST ---
Testing...
Max transactions per block =30
Processed 150 transactions in 5.46 seconds.
Transaction rate: 27.46 transactions per second.

--- TRANSACTION RATE TEST ---
Testing...
Max transactions per block =40
Processed 200 transactions in 5.44 seconds.
Transaction rate: 36.75 transactions per second.
Press Enter to continue...
```

The transaction rate test results indicate that increasing the maximum transactions per block enhances the overall processing efficiency of the blockchain system. Starting with a 10-transaction limit per block, the system processed 120 transactions at a rate of 23 transactions per second. As the block capacity increased, the transaction rate also improved: with a limit of 20 transactions per block, the rate rose slightly to 24.21 transactions per second, and with 30 transactions per block, the rate increased further to 27.46 transactions per second. The highest transaction rate was achieved at a 40-transaction block limit, reaching 36.75 transactions per second.

These results suggest that allowing more transactions per block reduces the overhead associated with frequent block creation, thereby increasing transaction throughput. This scaling effect demonstrates the potential for higher transaction capacity to improve efficiency in blockchain systems, making it an effective strategy for enhancing performance. However, increasing block size should be balanced with potential impacts on network latency and storage requirements to maintain system scalability and responsiveness.

The values obtained may vary based on multiple factors, such as the computer's performance at a given moment.

7. Future Work

This simplified blockchain implementation provides a foundational understanding of blockchain mechanics, but it has several limitations compared to real-world blockchain systems. Below are some key enhancements that could be made to improve its robustness, scalability, and security, making it more comparable to full-fledged blockchain networks.

7.1 Decentralized Consensus Mechanism

As already mentioned, the blockchain relies on a single miner, limiting decentralization and exposing the system to potential trust issues. Implementing a more robust consensus mechanism, such as Proof of Work (PoW) with multiple miners or Proof of Stake (PoS), would allow for distributed block validation, enhancing security and resilience. Multiple nodes validating blocks would reduce the risk of tampering by any single entity and align the system more closely with decentralized blockchain principles.

7.2 Transaction Throughput and Scalability

As it stands, this blockchain has limited transaction throughput, as each block only contains a small, fixed number of transactions. Increasing transaction capacity per block and optimizing block confirmation times could help improve scalability.

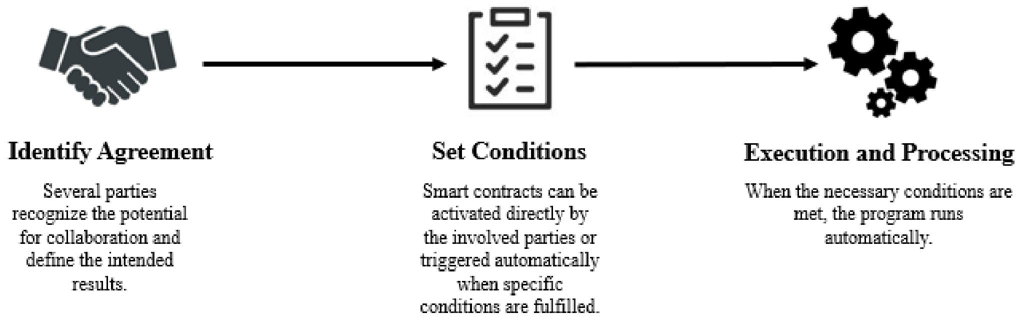
7.3 Smart Contract Functionality

A **smart contract** is a self-executing program stored on the blockchain that runs automatically when specific conditions are met. It defines rules and actions for an agreement. Once deployed, it becomes immutable, meaning its code cannot be altered.

When triggered by a transaction, the smart contract checks if the predefined conditions are satisfied. If they are, it executes actions like transferring funds or recording information. This process requires no intermediaries, as the contract itself enforces and executes the terms.

Stored on the blockchain, smart contracts inherit its security and transparency, making them a trusted, automated solution for managing agreements.

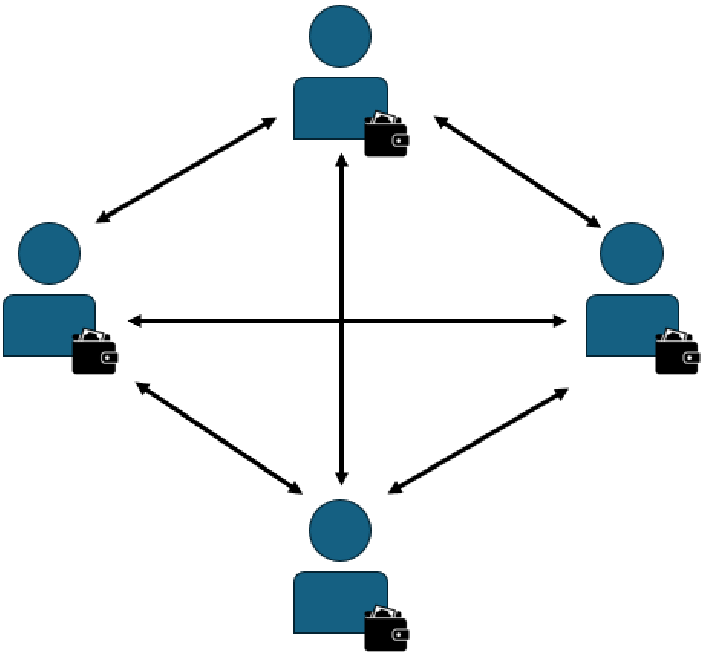
Adding smart contract support would extend the blockchain's utility, enabling programmable, self-executing agreements without requiring intermediaries.


Identify Agreement
Several parties recognize the potential for collaboration and define the intended results.
Set Conditions
Smart contracts can be activated directly by the involved parties or triggered automatically when specific conditions are fulfilled.
Execution and Processing
When the necessary conditions are met, the program runs automatically.

7.4 Improved P2P Communication

Peer-to-peer (P2P) communication is a decentralized network model where each participant, or "node," connects directly with others, sharing information without a central authority. In a P2P blockchain network, nodes communicate to validate and propagate transactions and blocks.

When a node initiates a transaction or receives new block data, it broadcasts this information to its connected peers. These peers validate the data and relay it to other nodes, ensuring the entire network is updated efficiently. This decentralized communication model increases resilience, as no single point of failure exists, and all nodes work collectively to maintain the network’s accuracy and security.



Peer-to-peer Communication

A more advanced peer-to-peer (P2P) communication protocol would allow nodes to better synchronize and propagate transactions and blocks. By enabling more efficient message broadcasting, network latency could be reduced, resulting in faster block validation and transaction processing.

7.5 Governance

To make the blockchain sustainable over time, implementing on-chain governance mechanisms would allow participants to vote on protocol upgrades and changes.

On-chain governance mechanisms are systems that allow blockchain participants to vote on changes to the protocol directly through the blockchain. With on-chain governance, proposals for updates, rule changes, or upgrades are submitted to the network, and token holders or designated nodes vote on these proposals.

Once a proposal reaches the required majority, the changes are automatically implemented on the blockchain. This approach ensures that protocol changes are transparent and that decisions reflect the consensus of the community, helping the network evolve while maintaining decentralized control.

By implementing these enhancements, the toy blockchain could evolve into a more secure, scalable, and feature-rich system, aligning it with the capabilities of modern blockchain networks used in real-world applications.

8. Conclusion

This blockchain project demonstrates the foundational elements of a decentralized ledger system, covering key aspects such as secure transactions, immutable records, and a simplified consensus mechanism. By combining cryptographic principles like SHA-256 hashing and ECDSA for transaction authenticity, this model showcases how data integrity and transparency can be maintained without a central authority. The implementation of core components (Wallet, Transaction, Block, and Blockchain) has provided a clear structure for understanding blockchain’s fundamental operations.

Although simplified, this system illustrates blockchain's potential impact across various industries, such as finance, supply chain, healthcare, and beyond. Its ability to facilitate secure, tamper-resistant record-keeping opens pathways for transforming how data is shared and verified. The journey of building this system has underscored the value of decentralization, setting the stage for innovations that could increase trust, efficiency, and autonomy in digital ecosystems.

As a stepping stone, this project highlights blockchain’s foundational role in advancing decentralized technologies. By exploring future enhancements, such as a full consensus mechanism and smart contract support, this model could evolve further, contributing to real-world applications and expanding the reach of blockchain technology.

References

Lectures 9, 10, 11, 13 and 16 of the course material.

- Blockchain workflow:

https://mlsdev.com/blog/156-how-to-build-your-own-blockchain-architecture

- Proof of Work VS Proof of Stake:

https://www.techopedia.com/proof-of-work-vs-proof-of-stake

- Smart Contracts:

https://binariks.com/blog/smart-contracts-blockchain-examples/

- Peer-to-peer Communication:

https://blog.cfte.education/what-is-p2p-network-blockchain/

- On-chain governance:

https://www.gemini.com/cryptopedia/blockchain-governance-mechanisms

All diagrams, graphs, and illustrations used in this document have been manually programmed or designed.