Blockchain Science and Technology

CS581: Course Project

Task:

Components of a Cryptocurrency

Group:

Aditya Gupta	210101009
Harsh Katara	210101045
Ridhiman Kaur Dhindsa	210101088
Shivam Agrawal	210101119

27 April, 2024

1 Introduction

Throughout the history of economies, humanity has explored various means of facilitating value exchange. From the ancient barter system and the use of coins and gold to more modern methods like cash, credit cards, and online payments, each form has had its limitations. Whether it's the inconvenience of bartering goods directly or the centralized control inherent in traditional currencies, there have always been drawbacks.

Cryptocurrency, built on blockchain technology, presents a revolutionary solution to these age-old challenges. By leveraging decentralization and cryptographic security, cryptocurrencies offer a new paradigm for currency exchange that is both convenient and resistant to centralized control.

Building a cryptocurrency involves several key components, each serving a crucial role in the functionality and operation of the digital currency:

- **Consensus Mechanism:** This is the algorithm or protocol that ensures agreement among participants in the network on the validity of transactions and the state of the blockchain. Popular consensus mechanisms include Proof of Work (PoW), Proof of Stake (PoS), Delegated Proof of Stake (DPoS), and others.
- **Blockchain:** The blockchain is a distributed ledger that records all transactions in a chronological and immutable manner. It consists of blocks linked together using cryptographic hashes. The blockchain serves as the backbone of the cryptocurrency, providing transparency, security, and decentralization.
- **Cryptographic Security:** Cryptocurrencies rely on cryptographic techniques to secure transactions and control the creation of new units. This includes digital signatures for transaction verification, hashing algorithms for securing the blockchain, and cryptographic puzzles in Proof of Work systems.
- **Mining (or Staking) Infrastructure:** In Proof of Work cryptocurrencies, mining involves using computational power to solve complex mathematical puzzles and validate transactions in exchange for rewards. In Proof of Stake cryptocurrencies, staking involves holding a certain amount of coins as collateral to validate transactions and earn rewards.
- **Economic Model:** The economic model determines the distribution, issuance, and supply of the cryptocurrency. It includes parameters such as block rewards, transaction fees, inflation rates, and monetary policies.

1.1 Motivation

In this project, we've incorporated three fundamental components essential for building a decentralized cryptocurrency: **the RSA cryptosystem** as a signature scheme, **the Dolev Strong Protocol** as a consensus mechanism, and **Merkle Trees** for organizing transactions within the block structure. These components collectively form the backbone of our cryptocurrency system, facilitating secure transactions, consensus among network participants, and efficient block validation.

The RSA cryptosystem serves as our signature scheme, enabling Public Key Infrastructure (PKI) assumptions crucial for the consensus mechanism. By employing RSA, we establish a secure method for generating and verifying digital signatures, ensuring the authenticity and integrity of transactions within the network.

Complementing the RSA cryptosystem, the Dolev Strong Protocol operates as our consensus mechanism, leveraging the capabilities of PKI assumptions to validate signatures and reach agreement among network nodes. This protocol plays a pivotal role in ensuring the integrity and consistency of the blockchain by verifying the validity of proposed blocks and achieving consensus among participating nodes.

Furthermore, Merkle Trees are employed to organize transactions within the block structure, facilitating efficient block validation mechanisms. Through the use of Merkle Trees, we streamline the verification process, allowing nodes to quickly confirm the integrity of transactions within a block without needing to access the entire transaction history.

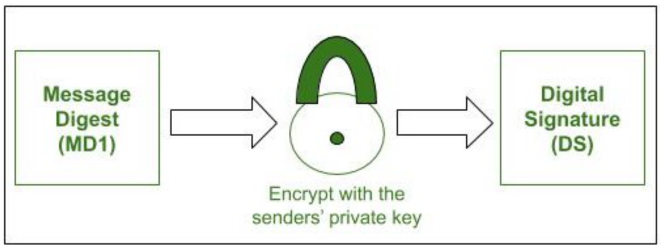
The integration of these three components is driven by their seamless compatibility and their critical roles in forming the foundation of our blockchain structure and process.

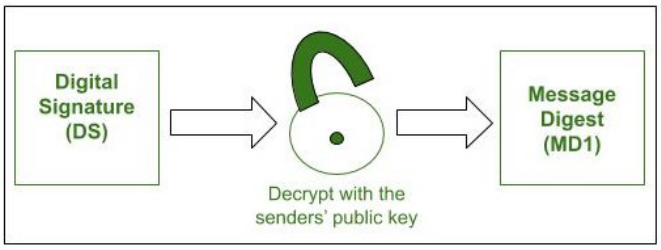
2 Cryptographic Primitives

2.1 Digital Signature Scheme

The RSA algorithm (named after its founders, Ron Rivest, Adi Shamir, and Leonard Adleman) has become almost synonymous with public key cryptography. It is an asymmetric cryptography algorithm which works on two different keys i.e. Public Key K^+ and Private Key K^-. These always satisfy the property that $K^-(K^+(m)) = m = K^+(K^-(m))$. The LHS represents encryption and the RHS represents authentication, since we can verify a sender's identity if he has signed the message with his private key.

To generate the public and private RSA keys, we perform the following steps:





1. Choose two large prime numbers, p and q. The larger the values, the more difficult it is to break RSA, but the longer it takes to perform the encoding and decoding. RSA Laboratories recommends that the product of p and q be on the order of 1,024 bits.
2. Compute $n = pq$ and $z = (p - 1)(q - 1)$.
3. Choose a number, e, less than n, that has no common factors (other than 1) with z. (In this case, e and z are said to be relatively prime.) The letter e is used since this value will be used in encryption.
4. Find a number, d, such that ed - 1 is exactly divisible (that is, with no remainder) by z. The letter d is used because this value will be used in decryption. Put another way, given e, we choose d such that $ed \equiv 1 \pmod{z}$.
5. The public key that is available to the world, K^+, is the pair of numbers (n, e); the private key, K^-, is the pair of numbers (n, d).

The encryption and decryption is performed as follows:

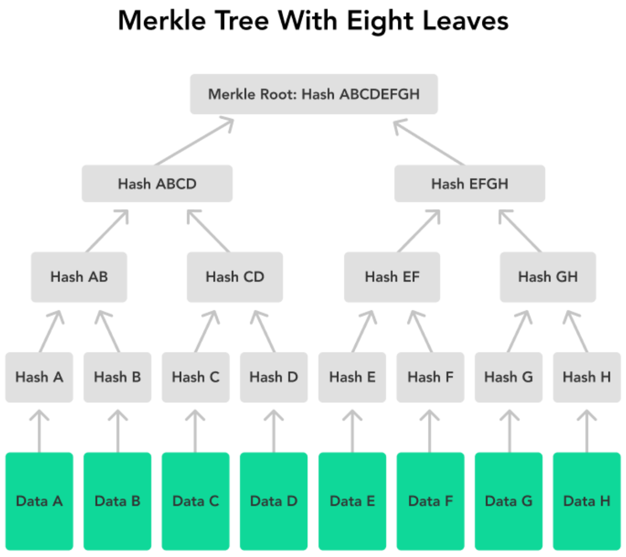
1. Suppose a bit pattern represented by the integer m is to be sent, the encrypted value (ciphertext) calculated will be: $c \equiv m^e \pmod{n}$ using the receiver's public key.
2. To decrypt the received ciphertext message, c, the receiver computes $m \equiv c^d \pmod{n}$ using his private key.
3. To verify a signature, the message signed with the sender's private key is checked by applying the sender's public key to it, such that $K^+(K^-(m)) = m$.

3 Block Structure

3.1 Merkle Trees

A Merkle tree, also known as a hash tree, is a tree in which each leaf node is labeled with the cryptographic hash of a data block, and each non-leaf node is labeled with the cryptographic hash of its child nodes' labels. It's a mathematical data structure made up of hashes of various data blocks that summarize all the transactions in a block. It also enables quick and secure content verification across big datasets and verifies the consistency and content of the data. The Merkle Root is stored in the block header. The block header is the part of the bitcoin block which gets hash in the process of mining. It contains the hash of the last block, a Nonce, and the Root Hash of all the transactions in the current block in a Merkle Tree. So having the Merkle root in block header makes the transaction tamper-proof. As this Root Hash includes the hashes of all the transactions within the block, these transactions result in saving disk space as well as creating a tamper-proof ledger. Our implementation supports 2 operations:

- Membership Proof: The membership proof consists of a path from the leaf node containing the data element to the root of the Merkle tree. The prover calculates the hash of the leaf node containing the data element and then combines it with the hash of its sibling node at each level of the tree, moving upwards towards the root. The resulting hash chain, along with the hashes of other nodes along the path, forms the membership proof. The verifier can then independently verify the validity of the Merkle proof by comparing the calculated root hash with the root hash of the Merkle tree.



- Non-membership Proof: It is also possible to test non-membership in logarithmic time and space using a sorted Merkle tree. That is, it is possible to show that a given transaction does not belong in the Merkle tree. This can be done by displaying a path to the transaction that is immediately before the transaction in question, as well as a path to the item that is immediately following it. If these two elements in the tree are sequential, this proves that the item in issue is not included, else it would have to go between the two items shown.

4 Consensus Algorithms

4.1 Dolev Strong Protocol

We implement the result from 1983 by D.Dolev and H.Strong on reaching an agreement in a distributed system between a set of nodes ”N”, even in the presence of faulty (Byzantine) nodes ”F”. Achieving Byzantine Broadcast using Dolev-Strong Protocol is a gateway to understanding and implementing how consensus protocol works, including more complicated models used in the blockchain space and distributed systems. Goals:

- Validity: If all correct processes propose the same value v, then any correct process that decides, decides v.
- Agreement: No two correct processes decide differently.
- Termination: Every correct process eventually decides,i.e. the protocol terminates at some point of time.

The first two properties are safety properties, i.e., properties that say that some “bad thing” cannot happen, while the last is a liveness property, i.e., a property that states “good things” that must happen.

Assumptions:

- Permissioned Settings: We assume we have a priori know set of nodes 1, 2, . . . , n before the start of the protocol.
- Public Key Infrastructure: Each node i has a distinct public-private key pair. PKI is known to all nodes at the start of the protocol.
- Synchronous: This protocol operates as a synchronous model. This includes a shared global clock as well as the assumption that every message sent by one node at time t arrive to recipient by time step t+1.

The algorithm runs in rounds, and requires clocks to be synchronized between all participating nodes. Each round begins at a constant time-frame in which receivers can pass along a message. The algorithm must run for t+1 number of rounds - where t is the number of adversary or faulty nodes we can have in our network without failing to reach consensus upon a message (t is the constant that describes the basic network security assumption).

In our implementation, each node is a sender and also a receiver for other agreement attempts upon messages from other eligible nodes. We employ a standard modern cryptographic signature scheme to authenticate message sender and to prevent message content modifications by man-in-the-middle kinds of attacks. Whenever a sender wishes to reach consensus on a certain payload within a network of other receiving nodes, the payload and the sender's public key are signed using the sender node's private cryptography key and the signature is added alongside the payload. This enables receiving nodes to validate the content and origin of this message.

Each node that receives a message validates the content using the signatures found chained after the payload, and then signs the payload with its own private key. After chaining the node's signature to all other signatures, the node then sends the message to all nodes in the layer that have not yet signed it yet.

If a receiver receives two contradicting messages from the same sender, this sender is considered malicious, in which case the receiver will forward the message to all other nodes to notify them of the malicious sender so they can stop processing messages from it. A message is also considered invalid if the number of signatures is less than the current round number. At the end of a t+1 round iteration, all nodes should reach consensus on a message, unless the sender was faulty, in which case the consensus is that its messages are invalid.

5 Conclusion

We have implemented and integrated three foundational components crucial for constructing a decentralized cryptocurrency system: the RSA cryptosystem, the Dolev Strong Protocol, and Merkle Trees. Together, these elements establish the framework for secure transactions, consensus among network participants, and efficient block validation. Further scope would include alternate implementations of these components such as:

- Alternatives to RSA: Elliptic Curve Cryptography (ECC), Diffie-Hellman Key Exchange, Lattice-based Cryptography could be implemented. The trade offs with using these alternatives compared to RSA include differences in key size, computational efficiency, resistance to quantum attacks, and the maturity of the implementations.
- Alternatives to Dolev Strong: On modifying appropriate assumptions and introducing a permission less setting, partially synchronous or asynchronous network, we can come up with a variety of other consensus algorithms. These include Tendermint Protocol, majority based consensus, longest chain consensus etc.
- Alternatives to Merkle Trees: Several alternatives could be explores, such as- flat structure (list of transactions), Radix tree, Patrichia Tree, Verkle trees (require less data for proofs) etc.

The choice of transaction storage method depends on factors such as the specific use case, scalability requirements, data structure complexity, performance considerations, and the desired trade-offs between decentralization, efficiency, and storage costs. The choice of consensus algorithm depends on factors such as use case, scalability, trust assumptions, fault tolerance, and performance. PoW provides security but demands resources, while PoS enhances scalability but may risk centralization.