



**Research on Blockchain Technology and its associated
Cybersecurity Issues**

Capstone Project Report

Presented by

RAHUL BAKSHI

**University of Alberta
Master of Science in Internetworking
Edmonton, Canada**

Under the guidance of
SANDEEP KAUR

ABSTRACT

The Blockchain is the decentralized and distributed chain of blocks that contains information, which was designed to timestamp the digital data so that it cannot be tampered. Since its inception in 2008 by Satoshi Nakamoto, Blockchain technology has drawn the attention of several financial and governmental sectors. Blockchain technology has the potential to be extremely important for the future of our financial system and other important industries. It has practically endless applications, and its ideas can be applied to various sectors where security, scalability, and efficiency are essential. Due to its decentralized nature, encryption concepts such as public-key cryptography, digital signatures, hashing and various algorithms Blockchain made it possible to cut down the involvement of the trusted third party and maintain the security of the shared databases.

The objective of the report is to conduct comprehensive research on Blockchain Technology. This research on the Blockchain is part of a sequence that outlines the history that led to the present-day system, mechanisms, and key features of Blockchain technology. We will study the layered architecture of the technology and different types of Blockchains and compare some consensus protocols used in the Blockchain. Furthermore, the security mechanism and the challenges of the technology are discussed. Finally, we implement the Distributed Denial of Service attack on blockchain technology and propose the possible mitigation steps for the same.

ACKNOWLEDGEMENT

I would like to extend my gratitude to my mentor Ms. Sandeep Kaur, who guided me throughout this project. Her constant support, guidance and suggestions helped me to expand my knowledge base and to complete the project. In addition, she offered me the freedom to work on my project while ensuring that I stayed on track and did not stray from my project's core.

I would also like to express my sincere gratitude to my program coordinator, Mr. Shahnawaz Mir, and my program director, Dr. Mike McGregor, for providing such an excellent opportunity to experience this project.

At last, I would like to express my heartfelt gratitude towards my family, classmates, professors, and the University of Alberta, for always supporting me throughout the journey and helping me achieve my goal.

TABLE OF CONTENT

1. INTRODUCTION	10
1.1 BRIEF INTRODUCTION TO BLOCKCHAIN TECHNOLOGY?	10
1.2 WHY IS IT IMPORTANT?	11
1.3 HISTORY	12
2. UNDERSTANDING BLOCKCHAIN	15
2.1 WHAT IS BLOCKCHAIN TECHNOLOGY?	15
2.2 THE LAYERED ARCHITECTURE OF BLOCKCHAIN	18
2.2.1 <i>The Hardware Layer</i>	19
2.2.2 <i>The Data Layer</i>	19
2.2.3 <i>The Network Layer</i>	20
2.2.4 <i>The Consensus Layer</i>	20
2.2.5 <i>The Application Layer</i>	21
<i>Layer 0</i>	22
<i>Layer 1</i>	22
<i>Layer 2</i>	23
<i>Layer 3</i>	23
2.3 OVERVIEW OF THE BLOCKS	24
2.3.1 <i>Block Identifiers</i>	25
2.3.2 <i>Block Structure</i>	26

2.3.3 Merkle Root	29
2.4 BLOCKCHAIN NODES	32
2.4.1 Types of Blockchain Nodes	32
2.5 BLOCKCHAIN FORKS	37
2.5.1 Understanding Forks in Blockchain	37
2.5.2 Hard Fork	38
2.5.3 Soft Fork	39
2.5.4 Hard Fork v/s Soft Fork	41
2.6 TYPES OF BLOCKCHAIN	41
2.6.1 Permissioned Blockchain	42
2.6.2 Permissionless Blockchain	44
2.6.3 Hybrid Blockchain	46
2.6.4 Consortium Blockchain	48
2.7 FEATURES OF BLOCKCHAIN	50
2.7.1 Decentralization	50
2.7.2 Immutable	51
2.7.3 Faster Settlement	52
2.7.4 Consensus	52
2.7.5 Anonymity	53
2.7.6 Increased Capacity	53
2.7.7 Security	53
2.7.8 Transparency	54
2.7.9 Distributed	55

3. CONSENSUS MODELS	56
3.1 INTRODUCTION.....	56
3.2 PROOF-OF-WORK	58
3.3 PROOF-OF-STAKE.....	62
3.4 DELEGATED PROOF-OF-STAKE.....	62
3.5 PROOF-OF-AUTHORITY.....	63
3.6 PROOF-OF-ACTIVITY	64
3.7 PROOF-OF-IMPORTANCE.....	65
3.8 PROOF-OF-BURN	66
3.9 PROOF-OF-CAPACITY	67
3.10 PRACTICAL BYZANTINE FAULT TOLERANCE (PBFT).....	68
4. SECURITY IN BLOCKCHAIN.....	70
4.1 INTRODUCTION.....	70
4.2 CRYPTOGRAPHY	73
4.3 ASYMMETRIC CRYPTOGRAPHY	76
4.3.1 <i>Applications of Asymmetric Cryptography in Blockchain Technology</i>	78
4.4 DIGITAL SIGNATURES.....	79
4.4.1 <i>RSA</i>	83
4.4.2 <i>Elliptic Curve Cryptography (ECC)</i>	83
4.5 HASHING	84
4.5.1 <i>Properties of Cryptographic Hash Functions</i>	85
4.5.2 <i>Patterns of Hashing</i>	89

4.5.3 Hashing in Blockchain.....	92
4.5.4 SHA-256.....	95
5. ATTACKS ON BLOCKCHAIN NETWORK.....	99
5.1 51% ATTACK	99
5.2 ECLIPSE ATTACK.....	100
5.3 DDOS ATTACK	102
5.3.1 Implementation	103
5.3.2 Mitigation Measures.....	108
6. CONCLUSION	110
7. REFERENCES.....	112
8. GLOSSARY.....	117

TABLE OF FIGURES

FIGURE 1: AN EXAMPLE OF BLOCKCHAIN TECHNOLOGY	10
FIGURE 2: HISTORY OF BLOCKCHAIN TECHNOLOGY	14
FIGURE 3: CENTRALIZED NETWORK	15
FIGURE 4: DECENTRALIZED NETWORK	16
FIGURE 5: THE LAYERED ARCHITECTURE OF BLOCKCHAIN TECHNOLOGY	18
FIGURE 6: ESSENTIAL ELEMENTS OF CONSENSUS LAYER	21
FIGURE 7: BLOCKCHAIN STRUCTURE	25
FIGURE 8: STRUCTURE OF A BLOCK	27
FIGURE 9: STRUCTURE OF A BLOCK HEADER	28
FIGURE 10: CALCULATING MERKLE ROOT IN MERKLE TREE	30
FIGURE 11: TYPES OF BLOCKCHAIN NODES	33
FIGURE 12: A HARD FORK	39
FIGURE 13: A SOFT FORK	40
FIGURE 14: TYPES OF BLOCKCHAIN	42
FIGURE 15: AN EXAMPLE OF CONSORTIUM BLOCKCHAIN NETWORK FOR MALWARE DETECTION.	49
FIGURE 16: CALCULATING CORRECT HASH VALUE	61
FIGURE 17: GENERAL DATA FLOW IN BLOCKCHAIN SYSTEM	71
FIGURE 18: DATA FLOW IN BLOCKCHAIN NETWORK	71
FIGURE 19: CRYPTOGRAPHY PRIMITIVES	72
FIGURE 20: CONVENTIONAL CRYPTOGRAPHY PROCESS	73
FIGURE 21: SYMMETRIC KEY CRYPTOGRAPHY	75
FIGURE 22: STREAM CIPHER FLOW DIAGRAM	75
FIGURE 23: ASYMMETRIC KEY CRYPTOGRAPHY	77
FIGURE 24: ENCRYPTION/DECRYPTION USING PUBLIC/PRIVATE KEYS	78
FIGURE 25: ILLUSTRATION OF DIGITAL SIGNING (SIGN THEN ENCRYPT APPROACH)	81
FIGURE 26: VERIFICATION OF THE SIGNED MESSAGE	82
FIGURE 27: COLLISION OF HASH VALUE	86
FIGURE 28: AN EXAMPLE OF INDEPENDENT HASHING	89

FIGURE 29: AN EXAMPLE OF REPEATED HASHING PATTERN OF HASHING TECHNIQUES	90
FIGURE 30: AN EXAMPLE OF COMBINED HASHING PATTERN.....	91
FIGURE 31: AN EXAMPLE OF SEQUENTIAL HASHING.....	91
FIGURE 32: AN EXAMPLE OF HIERARCHICAL HASHING PATTERN	92
FIGURE 33: ONE ROUND OF A SHA COMPRESSION FUNCTION	97
FIGURE 34: COMPLETE PROCESS OF 64 ROUNDS OF A SHA COMPRESSION FUNCTION.....	97
FIGURE 35: ECLIPSE ATTACK.....	101
FIGURE 36: SOURCE CODE OF A SERVER.....	104
FIGURE 37: SOURCE CODE OF A CLIENT	105
FIGURE 38: LEGITIMATE USER SENDS A TRANSACTION TO A BLOCKCHAIN NETWORK	106
FIGURE 39: LEGITIMATE USER NOT ABLE TO CONNECT AFTER THE NETWORK CRASH	107

1. INTRODUCTION

1.1 Brief introduction to Blockchain Technology?

The *Blockchain*, also known as a distributed ledger, is a peer-to-peer, decentralized, and distributed chain of blocks that are securely linked together using cryptography [1] [2]. That is why it is named “Blockchain.” These blocks contain information designed to timestamp the digital data so it cannot be tampered. Every block in a chain contains data (depending on the type of Blockchain), the hash of the block and the previous block hash (which forms the chain of blocks). Since each block contains information about the previous block, they effectively form a chain [3]. Hash is used for security purposes, like a fingerprint (unique) that identifies the block. Once the data is recorded in a block, its hash value is calculated, and changing anything in a block will lead to a re-calculation of the hash value, hence will cause all the following blocks in the chain to be invalid.

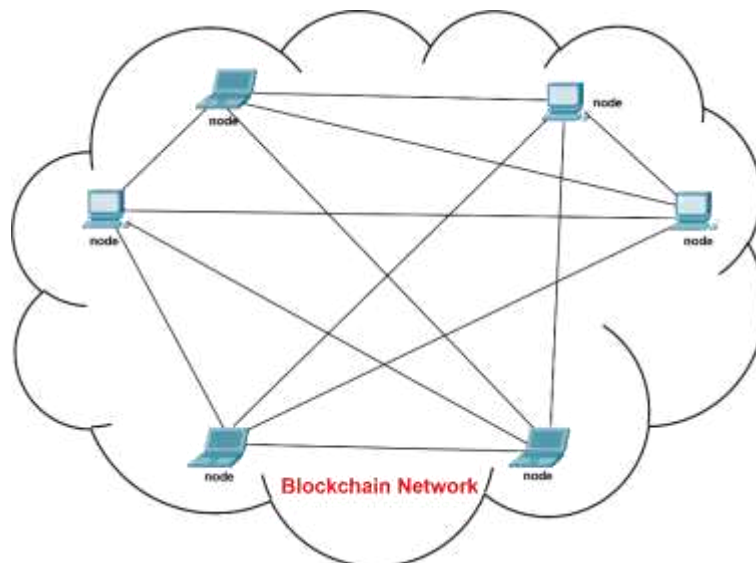


Figure 1: An example of Blockchain Technology [4]

Along with hash value, blockchain technology uses Proof-of-Work, a point-to-point network instead of a central entity, and smart contracts to stop tampering with the data. However, there are still some vulnerabilities associated with blockchain technology that led to past attacks, including 51% attacks, DDOS attacks, routing attacks, etc.

1.2 Why is it important?

Information is vital in the technological age we live in. The era in which the internet and physical worlds are increasingly interconnected to create environments that are more intelligent and integrated. If we talk about the most valuable organizations in the world, like Google, Meta, Apple, Amazon, Microsoft etc., they all serve as compelling evidence for the above statement. All these companies either operate solely as information brokers or bring physical things “online” to improve information generation and sharing. Even pure manufacturing organizations create, receive, and process information. Therefore, for best results, it is vital to have accurate and fast reception of information. Blockchain is considered one of the best technologies to do this as it provides utterly transparent information instantly, which is stored in an immutable ledger and can only be accessed by authorized network members. Blockchain networks can track orders, payments, accounts, production, and more [5]. Moreover, because members share a unified view of truth, they can see every transaction detail from start to finish, resulting in greater confidence/transparency and easier auditability, gains in speed and efficiency, automation and programmability, cost reductions, and opportunities [6].

1.3 History

In 1982, David Chaum, a cryptographer, first proposed a blockchain-like protocol in his dissertation titled “Computer System Established, Maintained, and Trusted by Mutually Suspicious Groups”. After this, in 1991, S. Haber and W.S. Stornetta published a paper on ‘How to time-stamp the digital document’, which proposed the method of time-stamping the data instead of the medium by using the computationally practical procedure. Again in 1993, S. Haber and W. Stornetta, along with D. Bayer, published a paper that suggested two schemes to improve the efficiency and reliability of digitally time-stamped data. After this, in 1997, the concept of hashcash was introduced by Adam Back, in which he put forward a solution to control spam emails. This led to the concept of creating money called “b-money” by Wei Dai based on a peer-to-peer network [7].

Based on all these concepts and solutions, a person or group named Satoshi Nakamoto first conceived Blockchain technology in 2008 when he published a paper on Bitcoin titled “Bitcoin- A Peer-to-Peer Electronic Cash System.” After which, on January 3rd, 2009, bitcoin was released in the market. Nakamoto’s paper and the technology behind bitcoin were based on years of research in the field of cryptography, e-cash, distributed ledger, the concept of hash-tree, etc.

In the paper, Nakamoto presented a method to make a direct online payment from one source to another without relying on a trusted third party. He explained how a peer-to-peer network using proof-of-work would enable the users to make an online payment without depending on a trusted financial institution. Through this paper, Satoshi Nakamoto also proposed a solution for the

problem of double spending. A cryptographic-based electronic payment system that makes it challenging to double-spend digital cash. A system where all users have access to transaction history, allowing them to check whether money has already been spent. This will also make it computationally impractical for an attacker to change if honest nodes control more CPU power than attacker nodes [8].

With the introduction of the Bitcoin network in 2009 and Satoshi Nakamoto's mining of the genesis block of bitcoin (block number 0), which contained a reward of 50 bitcoins, came the first use of blockchain technology. After several years, people began to realize the potential of blockchain technology, and numerous firms began investing in the full potential of the technology in various industries. It was then that Smart Contract was introduced, which was the most useful and popular advancement in the field at that time. Automation is quite important to increase profit, fast performance, and reduce efforts and expenses in any industry. Smart Contract offer automation to blockchain technology by enabling the users to write higher-level scripts and not worry about the actual blockchain implementation. All the parties involved finalize the smart contract rules, which are then stored in the ledger and will automatically be executed by the contract when the conditions are met. One such platform is the Ethereum platform, which was introduced in 2013 by Vitalik Buterin. Ethereum enabled blockchain to work with loans and contracts. Due to Ethereum's ability to offer a faster, safer, and more efficient environment, the technology became widely popular, leading to the development of a number of decentralized applications using the Ethereum platform, such as Corda, Hyperledger, Multichain and NEO [7] [9].

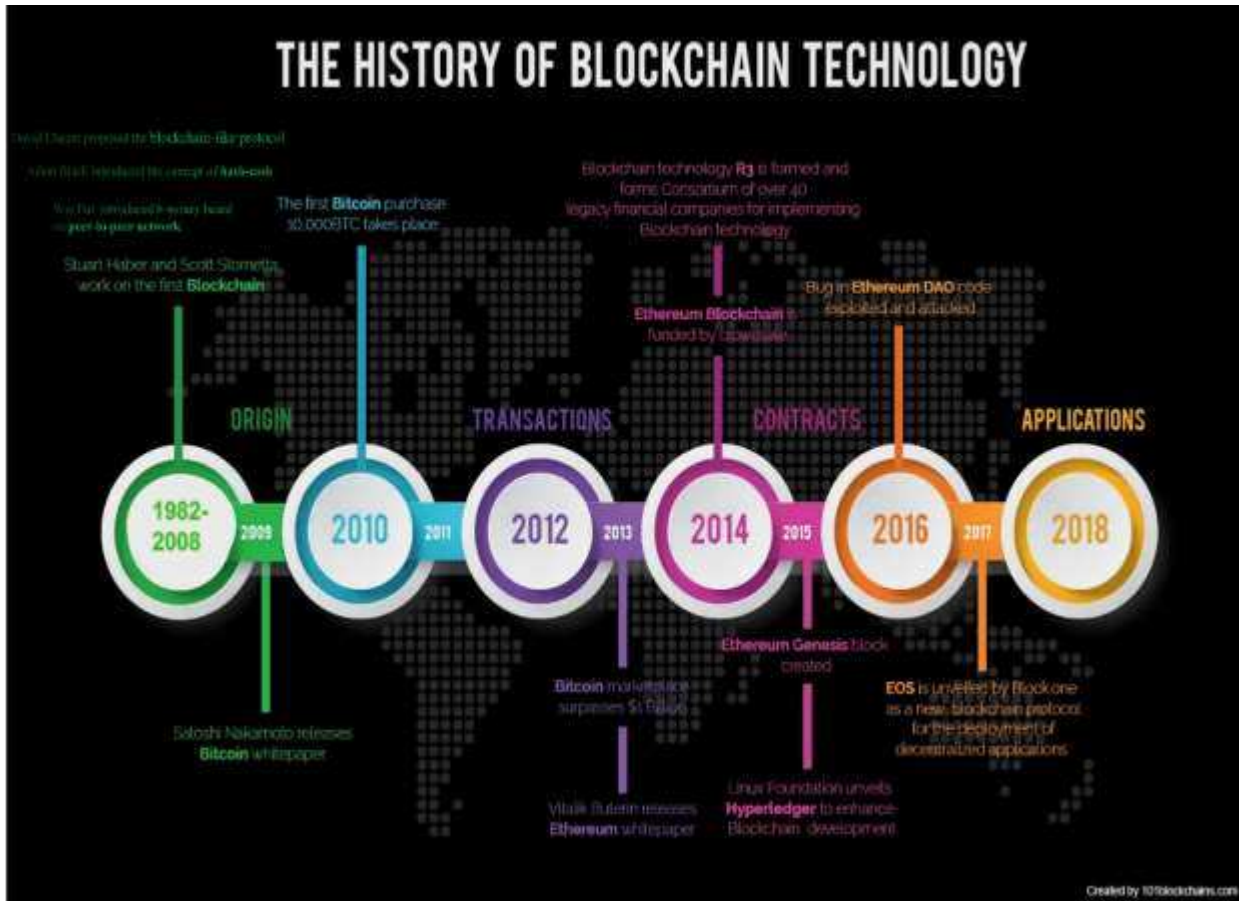


Figure 2: History of Blockchain Technology [10]

2. UNDERSTANDING BLOCKCHAIN

2.1 What is Blockchain Technology?

In its most basic form, a blockchain is a digital database to store information, which was designed to timestamp the digital data so that it cannot be tampered with. At its essence, blockchain is a distributed system/ledger with a decentralized or centralized structure. However, a decentralized platform is what a blockchain was first designed to be and what it is typically used as. So, to understand blockchain technology, we first need to understand the centralized and decentralized network and distributed ledger technology.

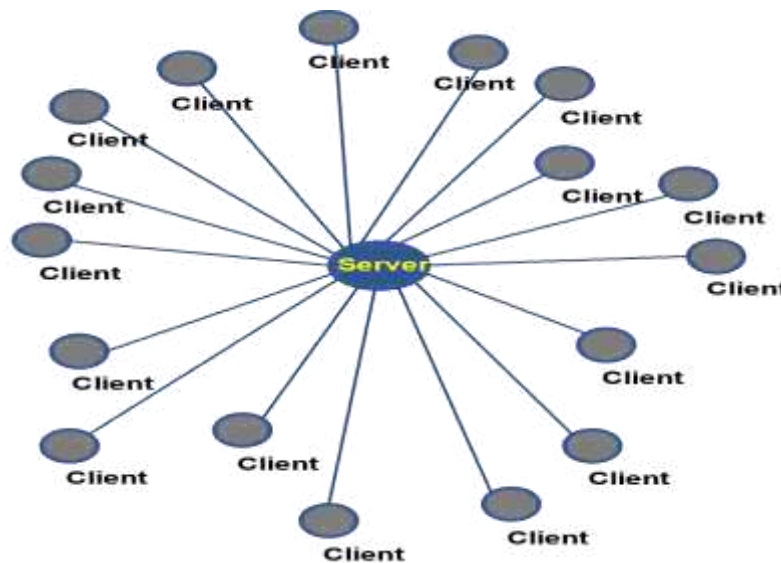


Figure 3: Centralized Network

The centralized network has a standard client-server architecture, where one node serves as the

server and every other node act as the client. As a result, one server node, which is the network's central authority, is required for the entire network to function. This node houses all the data, and client nodes solely rely on it for all the services. This traditional service delivery approach is used by the vast majority of internet service providers, including Google, Amazon, eBay, Apple's App Store, and others. On the other hand, a decentralized network is one in which every node is independent and is not reliant on a single server node. In this, the control is distributed among all the nodes in the network rather than relying on a central node. Therefore, it solves the problem of single-point failure of the conventional system.

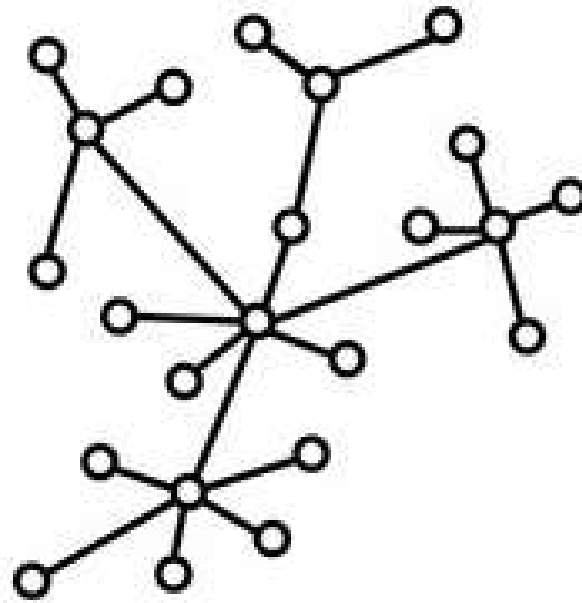


Figure 4: Decentralized Network [4]

Ledger, in simple terms, is nothing but the collection of accounts, and these accounts contain the record of all the transactions, whether debit or credit. For hundreds of years, humans have maintained the ledgers using various accounting tools, which have evolved. However, for all these

years, the objective has remained the same to keep the records safe and effective. Another thing that never changed was the third-party involvement as a validator (i.e. banks). Distributed Ledger Technology (DLT) is a significant advancement in this field and is a fast-evolving approach to recording and sharing data across multiple data stores or ledgers. Each has duplicate data records and is collectively maintained and controlled by a distributed network of computer servers called nodes [11].

Blockchain is a type of Distributed Ledger System with a decentralized structure which upholds a constantly growing list of records known as a block. The data, once stored or recorded in these blocks, is challenging to change. Each block is then ‘chained’ to the next block using a cryptographic signature [6]. Each block in the chain contains the cryptographic signature of the block before it, which allows blockchains to be used like a ledger that can be shared and verified by anybody with the right permissions. It can be thought of as a peer-to-peer distributed ledger that is append-only, immutable (very difficult to change), cryptographically secure, and can only be updated by consensus among peers. In layman’s terms, it is a continuously expanding, secure, shared record-keeping system where each user has a copy of the records, which can only be updated if all parties involved in a transaction agree to update [12]. Since blockchain can share information in an open point-to-point network without any central governing authority, it can be used in various industries other than the financial sector.

Although the blockchain is a digital database, it is not the same as a relational database in a way that the blockchain is a read and append-only, whereas the relational databases follow a **CRUD** (create, read, update, and delete) operational model.

2.2 The Layered Architecture of Blockchain

The architecture of Blockchain technology is based on its use case, as each application has distinct requirements. Furthermore, it is essential to remember that there are two methods to comprehend blockchain technology while discussing the layers of the blockchain. Understanding blockchain architecture is the first step. The five layers of blockchain technology include the hardware layer, data layer, network layer, consensus layer, and application layer.

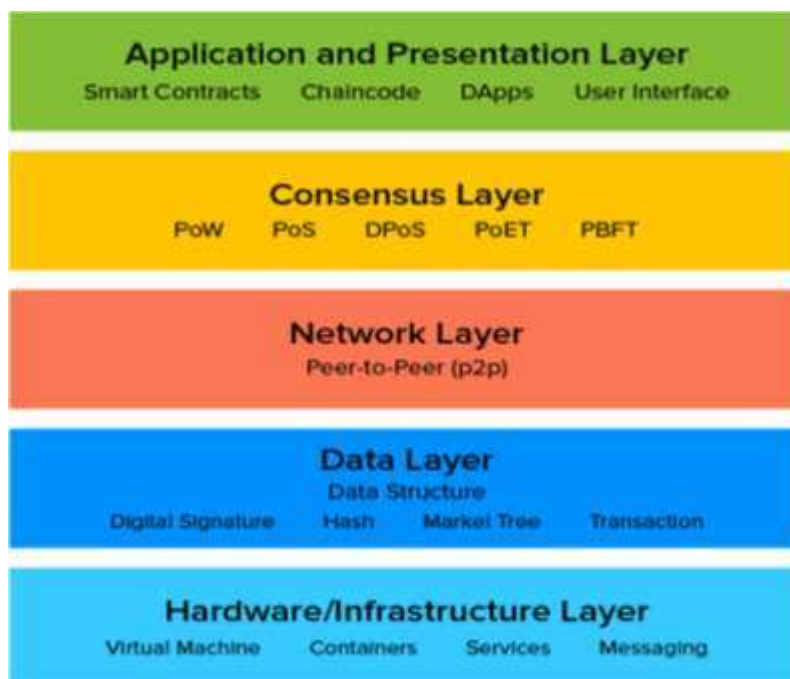


Figure 5: The Layered Architecture of Blockchain Technology [13]

The second is how the blockchain network is split up according to the protocol. Protocol refers to the set of rules that govern a network [14]. The blockchain protocol comprises four layers—Layer 0, Layer 1, Layer 2, and Layer 3.

The five layers of blockchain technology are:

2.2.1 The Hardware Layer

Hardware components, including network connections, computers, and data servers, make up the first layer of the blockchain. The hardware layer is responsible for securely hosting on a server, which the client requests while browsing the web or using any applications. This is commonly referred to as client-server architecture. However, clients can also communicate with one another and exchange data. Such a massive collection of computers sharing data is called a *P2P network* [13]. *Blockchain* is a peer-to-peer computer network that computes, authenticates, and accurately records transactions in a shared ledger [15]. The central components of this layer are nodes. A device is known and referred to as a node when it joins a blockchain network. Nodes are decentralized, distributed and are responsible for validating transactions, grouping them into blocks, disseminating them to the blockchain network, and so forth.

2.2.2 The Data Layer

The data layer, which is the second layer of the architecture, is where network-stored information is managed. The data structure of a blockchain is seen as a linked list of data blocks in the data layer. All the blocks in the blockchain network contains the hash of the previous block, except the first block, which is called the genesis block. The data about the block, transactions, and transacting parties are secured using cryptography. All the transactions on these blocks are digitally signed using a private key to ensure the security and integrity of the data on the blockchain

[15]. A private key is a digital signature used to authorize transactions that are only known to its owner, anyone with access to the public key can verify who has signed for the transaction. To put it simply, if someone has to send you some cryptocurrency, they need to know your public key for that and for you to receive it, and you have to use your private key to verify the transaction and demonstrate your ownership of your blockchain wallet [13].

2.2.3 The Network Layer

The network layer, also known as the point-to-point layer, is responsible for inter-node communication [16]. Network Layer, also referred to as a propagation layer, handles block propagation, transactions, and discovery. This layer makes sure that only valid transactions are shared on the network. Moreover, the blocks are created and added to the blockchain at this layer. The Network layer also ensures that nodes can find one another and interact, share information, and synchronize with one another to keep the blockchain network in a legitimate state.

2.2.4 The Consensus Layer

For blockchain platforms, the consensus layer is crucial. This is the layer that enables key functionality of blockchain, which is why it is fundamental to the existence of blockchain platforms, and the failure of this layer leads to the failure of the entire blockchain system. The consensus layer is in charge of ordering the blocks, authenticating them, and making sure everyone is in agreement [15].

This layer ensures that the rules are effectively applied to maintain uniformity inside the network. A transaction cannot be added to the blockchain by a single node; instead, all nodes in the network must concur on it and validate the transaction. This reduces the likelihood of fraudulent transactions being added to the blockchain.

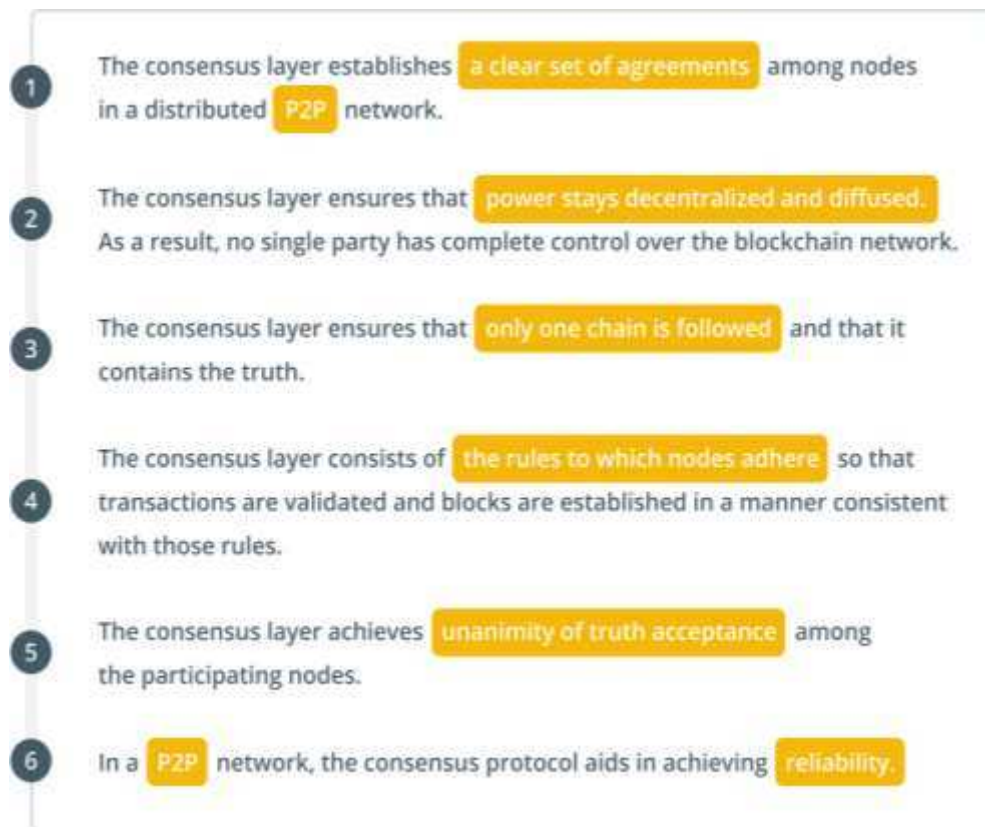


Figure 6: Essential Elements of Consensus Layer [17]

2.2.5 The Application Layer

The Application Layer is the layer that lies on top of the blockchain technology architecture. The application layer can be further subdivided into the application and the execution layers. The

application layer has the programs that end users use to communicate with the blockchain network, and it is comprised of scripts, API, user interface and frameworks. Applications interact with the blockchain network through APIs, which serve as the back-end technology for them [13].

The execution layer comprises smart contracts, underlying rules, and chaincode and it is the execution layer that has the code and rules to be executed. The transaction starts from the application layer and then moves to the execution layer, where its validation and execution occur.

Now, let's interpret the above-mentioned layers in terms of the standard terminology use in blockchain technology:-

Layer 0

The internet, hardware, and connections that will make it possible for the other layers of the blockchain to work smoothly makeup layer zero. These components form the technology that enables any blockchain to function and also enables inter-chain operability. in the terminology outlined above, layer zero is made up of the hardware infrastructure layer and the data layer [14].

Layer 1

Layer one typically refers to the compute and consensus layers bundled together in every blockchain system. Layer one of the blockchain protocol is the foundation layer, also known as the implementation layer, and consists of the different blockchains that can process transactions [14]. This layer is in charge of programming languages, block time, dispute resolution, consensus

processes (like proof of work and proof of stake) that guarantee the security of the blockchain, and the guidelines and parameters that maintain a blockchain network's basic functionality.

The issue with this layer is that layer one becomes overloaded as more people use blockchains. To secure the blockchain using the consensus mechanism, miners have to solve cryptographic algorithms using computational power. The workload on the layer one blockchain increases as the number of users grows, as more computational power and time are required, and hence, the processing speeds and capacities are slowed as a result.

Layer 2

We just read about the layer one scalability problem as the blockchain expands. The best method for addressing this scalability problem is Layer 2, often known as the execution layer. It is a third-party integration that works together with layer one to address the scalability issues and enhance the blockchain's productivity by enhancing the number of nodes. Examples of layer two architectures that have addressed blockchain difficulties include nested layers, rollups, and sidechains. To do this, certain interactions are removed from the base layers, all transactional validations are managed at layer 2, and layer 1 adds and creates new blocks to the blockchain network.

Layer 3

The application layer of the blockchain protocol, or layer 3, is the final layer of the blockchain ecosystem and is where blockchain-based applications and enabling protocols are hosted. Layer

three serves as a user interface while disguising the communication channel's technical details. Along with the user interface, it also offers usefulness in the form of intra- and inter-chain operability, including decentralized exchanges, liquidity provisioning, and staking applications. The developers gain real interoperability and blockchain utility from this layer. Decentralized apps (dApps) are a type of layer three interfaces that provide real-world applications for blockchain technology.

2.3 Overview of the Blocks

A block is made up of transactions, and a transaction is a record of an activity, such as the transfer of money from the sender's account to the beneficiary's account. All the valid transactions in the blockchain are collected and stored in groups called blocks. These blocks are then linked or chained together to form a blockchain. The formation of the blockchain is similar to the linked list where each block has a pointer, which is the identifier of the block and points towards the previous block in the chain, thus forming a link all the way to the first block in the blockchain known as genesis block as shown in *figure 7* [9]. This pointer is nothing but a hash value of the previous block in a sequence. The hash value is a unique identifier of fixed length, and each block contains the hash value of its own and the previous block in the chain, as all the blocks are linked together using the hash value. The previous block in a chain is called a parent block, and the subsequent block is called a child block. Every child can have only one parent block or vice versa.

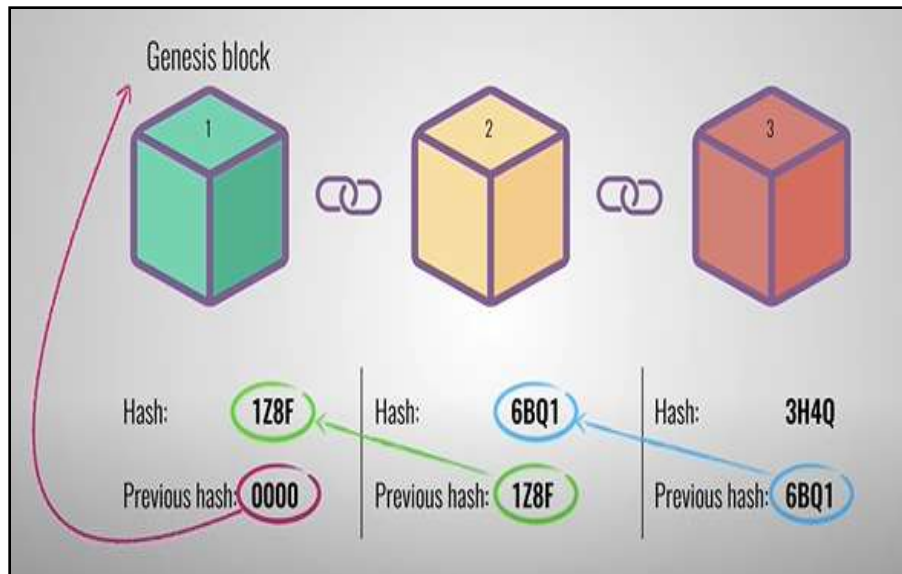


Figure 7: Blockchain Structure [18]

The height of the block is a crucial characteristic and a simpler method to refer to a block. It is just a plain number as opposed to the complex hash value and tells how far a block is from the genesis block. Therefore, the block hash value is not the hash value of the complete block but rather the hash value of the block header, which consists only of metadata.

2.3.1 Block Identifiers

There are two ways to identify the block in the blockchain: Block Header Hash and Block Height. A block in a blockchain is typically identified using the Block Header Hash, which is a cryptographic hash of a block. It is calculated by hashing the block header twice through the Secure Hash Algorithm 256 (SHA256 Algorithm). SHA256 is a part of the SHA-2 algorithm (Secure Hash Algorithm 2) designed by the National Security Agency (NSA) of the United States. SHA256

generates a 256-bit (32-byte) signature or hash, which is not sent through a network when a block is communicated over a network. Each node determines it as part of the validation process by merely hashing the block header [19].

The second way to identify the block in a blockchain is through a block height. The block height of a block is defined as the number of confirmed blocks preceding it in a blockchain [20]. The first block of a blockchain (genesis block) is at a block height 0, and each subsequent block after the first block is one position higher in the blockchain. Like, Block Header Hash, Block Height is also not part of a block's data structure.

2.3.2 Block Structure

Although the structure of a block depends on the application, type, and design of a blockchain. However, the general structure of a block includes few attributes that are essential for the block to function: ***Block Size, Block Header, Transaction Counter, and Transactions.***

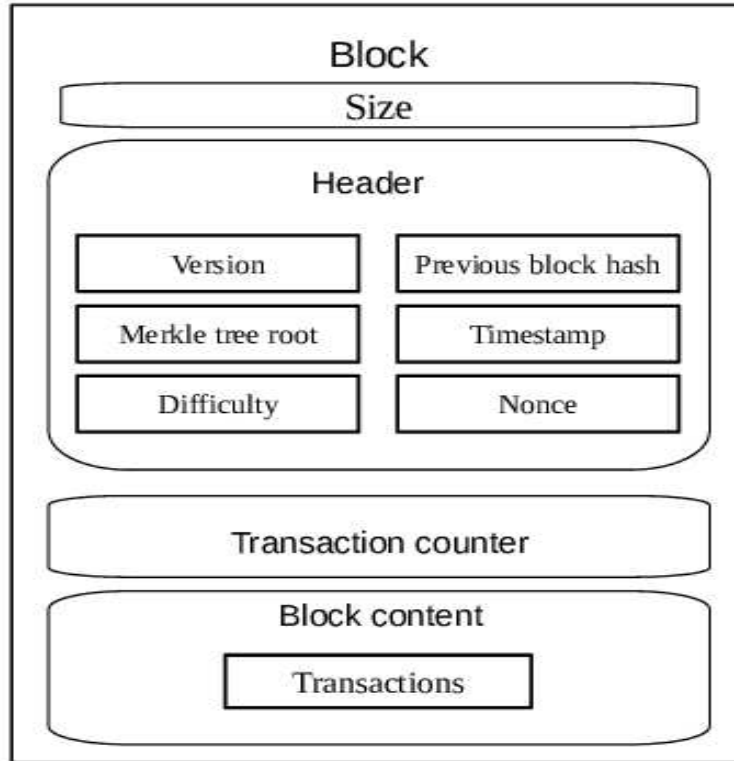


Figure 8: Structure of a Block [21]

The Block Size is the size of the entire block, which varies depending on the type and design of the blockchain in use [22]. The Block Header and Transactions are the essential parts of the block, as they are responsible for the hash value. The Block Header contains the block metadata, which includes the block version, hash of the previous block, merkle tree root, timestamp, difficulty target, and nonce. At last, all the valid transactions are stored. The size of the block as well as the size of each transaction determine the maximum number of transactions that can be stored in a block. Every node of the network validates each transaction before adding it to the block. Lastly, the Transaction Counter has the count of the transaction in the block.

Block Header

The Block Header consists of metadata of the block. The constituents of the block header are **Block Version, Previous Block Hash, Merkle Root, Timestamp, Difficulty Target, and Nonce**.

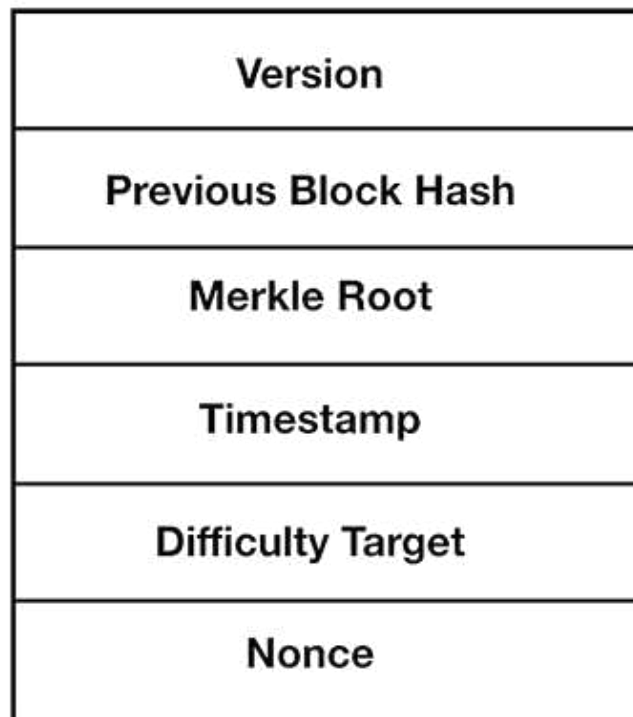


Figure 9: Structure of a block header [21]

Block Version: It states the version that particular block is using or the set of block validation rules to follow.

Previous Block Hash: The previous block hash is 256-bit hash value that refer to the previous block in the sequence/chain.

Merkle Root: A Merkle root is a hash of all the nodes of a Merkle tree, which verifies the integrity of the data using mathematical formulas to rule out corruption, hacking, or manipulation [23]. Merkle trees are usually used to validate large data structures and, therefore, are used in the blockchain to validate all transactions quickly and securely. All the transactions in the block combine to form one hash value, which is the block's hash value and is called the Merkle Root. Therefore, just by verifying the Merkle Root, all the transactions can be authenticated rather than authenticating them one by one.

Timestamp: The blockchain's timestamp is used as evidence that a specific block was used at a specific point in time and as a parameter to confirm any block's legitimacy [23].

Difficulty Target: The difficulty target is used, to simply specify the complexity and computational power required by the miners working to solve the block [24].

Nonce: It is a short form of “**Number only used once**”. As the name suggest it is a number which is generated and used only once. It is a 32-bit number that increases with every hash calculation and use for proof-of-work consensus algorithm and transaction replay protection.

2.3.3 Merkle Root

A *Merkle tree*, also known as a *binary hash tree*, is a data structure used for summarizing and validating large data structures [23]. As a result, they are used in the blockchain to validate all transactions quickly and safely. Merkle tree summarizes all the transactions in each blockchain block to form one digital fingerprint or hash value, providing a very efficient process to verify whether a transaction is included in a block. A Merkle tree is constructed by recursively hashing

pairs of nodes until there is only one hash, called the *root*, or *Merkle root* [25]. One can check to see if any data element is included in the tree using at most $2 \cdot \log_2(N)$ calculations, where N is data elements that are hashed and summarised in a Merkle tree.

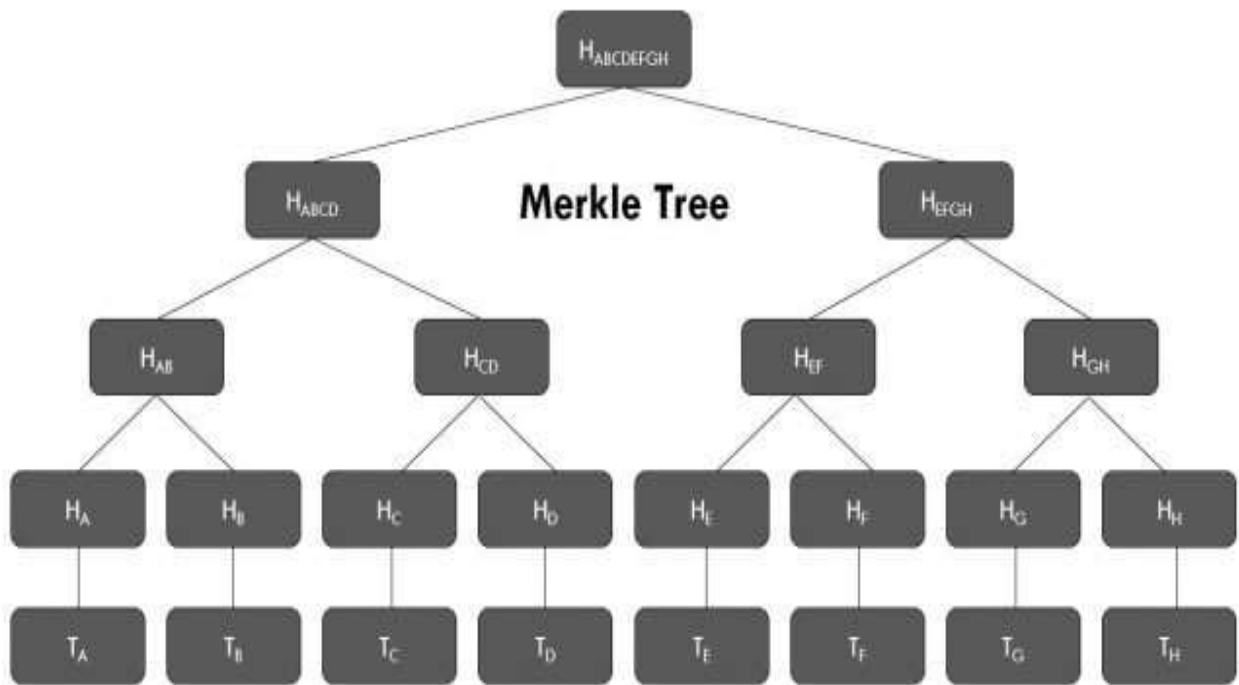


Figure 10: Calculating Merkle Root in Merkle Tree [26]

The Merkle tree is built from the bottom. Let us understand with the help of an example shown in *figure 10*. Let us assume that we have eight transactions (A, B, C, D, E, F, G, H) of one block, which are the leaves of the Merkle tree. All these transactions are hashed using SHA-256 algorithm, which results in a unique hash value of each transaction ($H_A, H_B, H_C, H_D, H_E, H_F, H_G, H_H$) and stored in a leaf node of a Merkle tree as shown in *figure 10*.

$$\text{Hash Value of a Transaction A} = \text{SHA-256}(\text{SHA-256}(\text{Transaction A})) = H_A$$

After this, pairs are made of consecutive leaf nodes, which are then hashed together, resulting in the parent hash. In this example, the hash of transactions A and B, C and D, E and F and a hash of transactions G and H are clubbed together to find the parent hash, i.e., H_{AB} , H_{CD} , H_{EF} , and H_{GH} as shown in *figure 10*. This process continues until only one node is left, called the Merkle Root, which is then stored in the Block Header.

$$H_{AB} = \text{SHA-256}(\text{SHA-256}(H_A + H_B))$$

Now, we are left with H_{AB} , H_{CD} , H_{EF} , and H_{GH} , which will be further coupled and hashed again, resulting in H_{ABCD} and H_{EFGH} . Finally, H_{ABCD} and H_{EFGH} will be hashed together, leaving only one node at the top ($H_{ABCDEFGH}$), called the Merkle root. The Merkle root is a 32-byte value which summarizes all the data in all eight transactions.

Therefore, it is beneficial to use Merkle tree for blockchain's as it significantly improves the efficiency of blockchain as less data is required to be communicated through the network to verify the data and transactions. This also reduces the amount of computing power required to validate the transactions. Furthermore, Merkle tree provides an advantage of tamper detection by determining whether any transaction(s) is tampered with [27]. It also allows for Simple Payment

Verification (SPV), where a node doesn't download the complete block, it just downloads the block header and verifies a transaction(s) by using Merkle Path.

2.4 Blockchain Nodes

On the most basic level, a node is simply a device running the software of a specific blockchain [28].

~ Till Wendler, co-founder of Peaq

A device is known and referred to as a node when it joins a blockchain network and runs the blockchain protocol's software. However, the precise definition of nodes varies based on their type, purpose, and the network. Nodes are decentralized, distributed, and are responsible for validating transactions, grouping them into blocks, disseminating them to the blockchain network, and so forth. However, every node does not perform the same function and one blockchain can have different types of nodes in its network. Every node on the blockchain network have the identical database. As transparency is one of the features of blockchain technology, which means anyone can even access the history of all previous transactions on the blockchain. To do that user actually communicate with a blockchain node to access the data. Nodes are very important part of the blockchain system as they maintain the data integrity.

2.4.1 Types of Blockchain Nodes

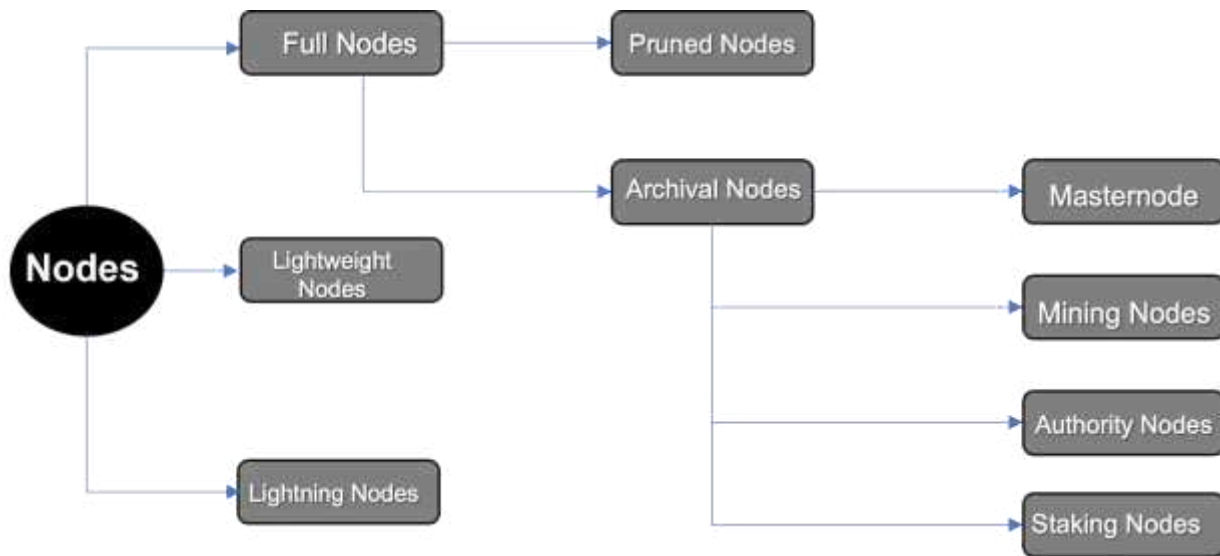


Figure 11: Types of Blockchain Nodes

There are broadly two categories of nodes- full nodes and lightweight nodes. Full nodes, also known as blockchain servers, store the blockchain's complete history. Lightweight Node or Simplified Payment Verification nodes are the ones which only download and store the block header and save time and storage. A full node is further divided into two types, Pruned Nodes and Archival Nodes. Out of these two, Archival Nodes are divided into four categories: Masternodes, Mining Nodes, Authority Nodes, and Staking Nodes.

2.4.1.1 Full Node

Full Nodes are also called servers of the blockchain. Full node contains the complete history of all the transactions on the blockchain and are responsible for maintaining consensus between other nodes. A full node is a blockchain node that stores the blockchain data and validate and authenticate the new blocks added to the blockchain. Full nodes are the ones that vote on proposals when deciding for the future of a network [29]. The proposal is skipped if more than 51% of the

community reject it, and in some situations, this can result in a hard fork where the community cannot agree on a particular modification and splits into two chains as a result. There are two types of full nodes, Pruned Nodes and Archival Nodes.

Pruned Node

The first category of Full Node is Pruned Node. As a full node, it is responsible for verifying transactions and is involved in consensus. A Pruned node downloads and stores the block in the memory; however, pruned nodes have only limited storage memory. Therefore, once the capacity is reached, it starts deleting the blocks to make space for the new blocks; this is called pruning. It deletes the older blocks from its memory but does keep their metadata and sequence. It keeps deleting the older blocks until it can store the most recent transactions in its limited storage space.

Archival Node

Another type of Full node is Archival Nodes. As a full node, they maintain consensus and validate transactions. It differs from pruned node in a way that it stores the complete blockchain from the beginning to the end. Due to this, archival nodes have large storage space as compared to pruned nodes. Archival nodes are further divided into four categories:

Masternode

A masternode is a full node that keeps the record of the transactions in a blockchain network and is responsible for validating the transaction, but it can not add a block to the blockchain. However, using a masternode in a network helps secure a blockchain and is relatively inexpensive to maintain. One can earn rewards for their service of running a masternode. However, the condition is that the one establishing a masternode must always be available and have to put a significant amount of money as collateral. Dash was the first blockchain to use masternodes in its network mechanism in 2014 [30].

Mining Node

Mining nodes are an essential part of the blockchain because they solve difficult mathematical problems to validate transactions on the network. A mining node may have one miner or a group of miners working together, known as a mining pool, and it is selected based on a consensus mechanism. To run a mining node or to be a miner requires highly powered computational devices to solve complex mathematical problems, which also consumes a substantial amount of electricity. In return, miners get compensated with coins for their service.

Authority Node

Anyone can join and become a node in a public blockchain by synchronizing blockchain data in their systems. There are many situations, nevertheless, where it's critical to maintain access to data and authorized parties in this scenario must govern the Blockchain [10]. Authority nodes are used in this situation. The organization or community managing a blockchain elects an authority node. These nodes get their name because they control or permit other nodes to join the Blockchain

network. Authority nodes can even define which nodes have access to a particular data channel in some Blockchain platforms.

Staking Node

A node in the blockchain that uses the Proof-of-Stake consensus technique to verify the transaction on the network is referred to as a staking node. As the name implies, to run a staking node, one needs to stake a certain amount and become an archival node. Then as per the set of rules, the system uses one of the staking nodes to validate the transaction on a network. If it is done successfully, the staker will receive a reward in return. Selection is subject to a set of predetermined guidelines, such as time spent on the blockchain, or the amount of funds staked. Similar to a mining node, a staking node may have one or a group of users called a staking pool.

2.4.1.2 Lightweight Node

Lightweight Node or Simplified Payment Verification nodes are the ones which only download and store the block header and save time and storage. Since they just download the block header and therefore have only the essential information. Hence, they rely on full nodes while communicating with the blockchain. The purpose of these nodes is to enable fast transactions, which is why they are called Simplified Payment Verification nodes. That is why they require minimal resources for the operation.

2.4.1.3 Lightning Node

Another type of blockchain node is the Lightning node. The lightning node is a beneficial concept that reduces the overall load of the blockchain network. The lightning node establishes a separate connection between the users outside the blockchain network, to which users have a separate key to access. Using this private connection, users can make the transactions which are agreed upon between them, and these transactions are done almost instantly. Once all the transactions are done, any user can close the connection, take the final balance sheet, and broadcast it onto the network. This way, users do not have to wait for each transaction to get validated first, reducing the wait time and the overall load from the blockchain network.

2.5 Blockchain Forks

2.5.1 Understanding Forks in Blockchain

As technology is evolving every single day, it is essential to continuously upgrade the systems to maintain efficiency, improve security and be able to use new features. Since blockchain is decentralized in nature and there is no central authority to make decisions about the updates in the system, it is done with the agreement of all the network nodes. However, since every block is linked through a practically unalterable set of protocols, it is nearly impossible to make changes to every block in the network. Therefore, the fork is a better way to modify or upgrade the blockchain network. Forking is vital to maintain the blockchain and support the participants on the

network. It is a way to make modifications in which the original code of the chain is used along with the relevant upgrades or proposed changes.

The forks are broadly classified into two categories: Hard Forks and Soft Forks. However, there is one more type of fork, known as the accidental fork. In reality, not all the forks are intentional; some can be accidental when some nodes are not replicating the same information [26]. Almost all accidental fork occurs when two or more miners find the same block and starts mining on different chains. An accidental fork is identified when any miner adds the subsequent block. This block decides which chain becomes the longer one leaving others abandoned, and the miners move to the longest chain. All blocks of these abandoned chains are known as orphaned blocks. In most cases, these forks are identified and resolved.

2.5.2 Hard Fork

The Hard Fork is a change to a consensus of the blockchain network, and every node on the network is required to upgrade to the latest protocol software. The Hard fork is a permanent divergence from the current blockchain version. However, there is always a possibility that not all nodes will agree to the proposed changes, leading to the blockchain split into two. Therefore, a hard fork refers to a radical change to the protocol of a blockchain network that effectively results in two branches, one that follows the previous protocol and one that follows the new version [31].

As in most cases, the hard fork leads to the splitting of the chain. The developers generally do not prefer using a hard fork to do the significant update in the network. This is because splitting the chain will make the network less secure and vulnerable to attacks like 51% attack. However, there are numerous essential reasons that may lead to the implementation of a hard fork by a developer, like adding some functionality, correcting the security risk, or resolving any disagreement within a community. One more important reason to implement a hard fork is to reverse a transaction on a blockchain; one such example is when the Ethereum blockchain implemented a hard fork to reverse the hack on the Decentralized Autonomous Organization (DAO) [32].

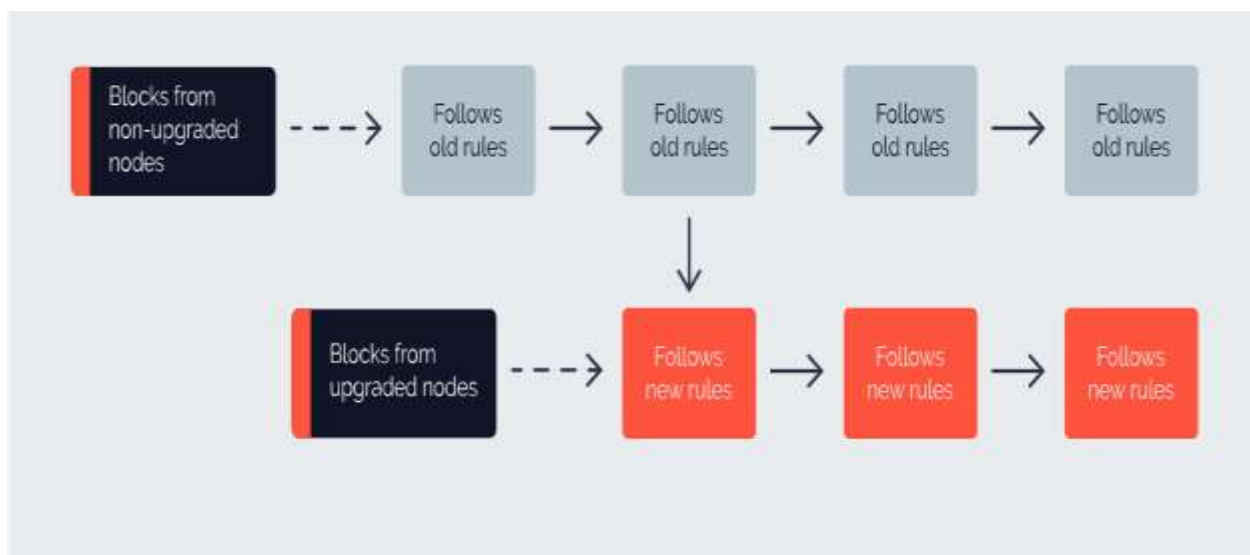


Figure 12: A Hard Fork [31]

2.5.3 Soft Fork

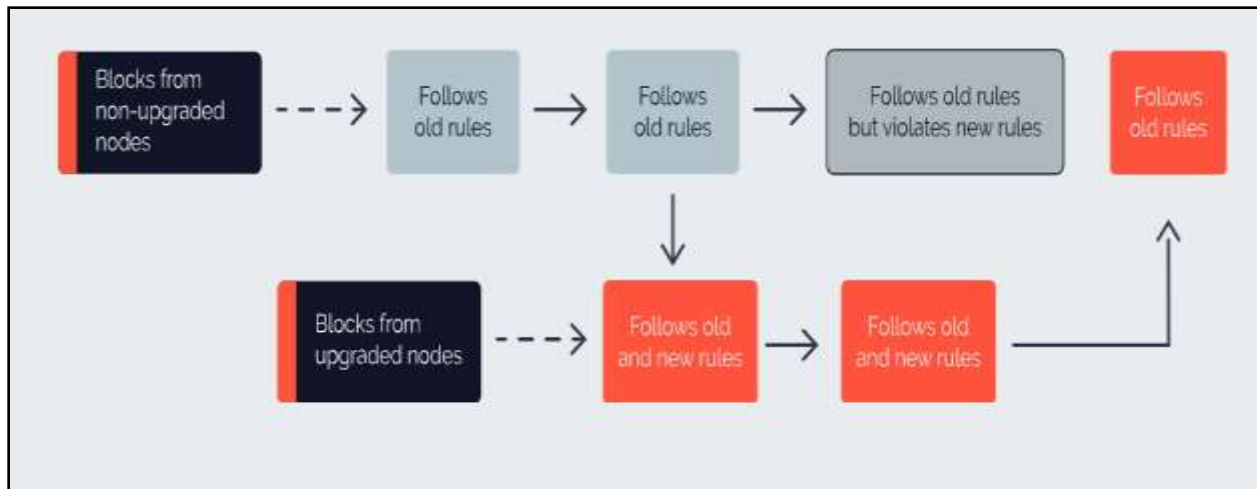


Figure 13: A Soft Fork [31]

A soft fork is another way to implement updates or changes in the network, but it is considered a safer option than a hard fork. Moreover, it is not at all compulsory for the nodes to upgrade to the latest version. Furthermore, as opposed to hard fork, soft fork is backward-compatible, which means the nodes that do not upgrade to newer versions will still see the chain as valid. However, this advantage is sometimes used by hackers to make malicious changes in the network.

A prominent example of a soft fork is Segregated Witness (SegWit) fork, which was introduced after the Bitcoin Cash split. Segregated Witness changed the format of blocks and transactions, but it was cleverly crafted in a way so that the old node could still validate blocks and transactions (the formatting did not break the rules), but they would not understand them [33]. Since it was a soft fork, it was not mandatory for all the nodes to upgrade to it, which is why, until today, not all the nodes are using this feature. There are advantages to doing so, but there is no real urgency since there is no network-breaking change [33].

2.5.4 Hard Fork v/s Soft Fork

	<u>Hard Fork</u>	<u>Soft Fork</u>
1	It is mandatory for all the nodes to upgrade to the latest protocol software to use it.	It is not mandatory for all the nodes to upgrade to the latest version.
2	It is not backward compatible.	It is backward compatible which means that the upgraded nodes will still be able to communicate with the non-upgraded nodes.
3	If all the nodes in the network do not agree to the proposed change or update, then it will lead to splitting of chain.	Under soft fork implementation, the chain does not split into two.
4	They often make a change to the original protocol, thus requiring a stronger consensus of nodes and miners.	The changes are made on the network level and can occur even when a small section of developers want to add small upgrades.
5	Hard forks require high computational power since a new side chain generates out of the original one.	Generally, 51% hash power is required for a soft fork.

2.6 Types of Blockchain

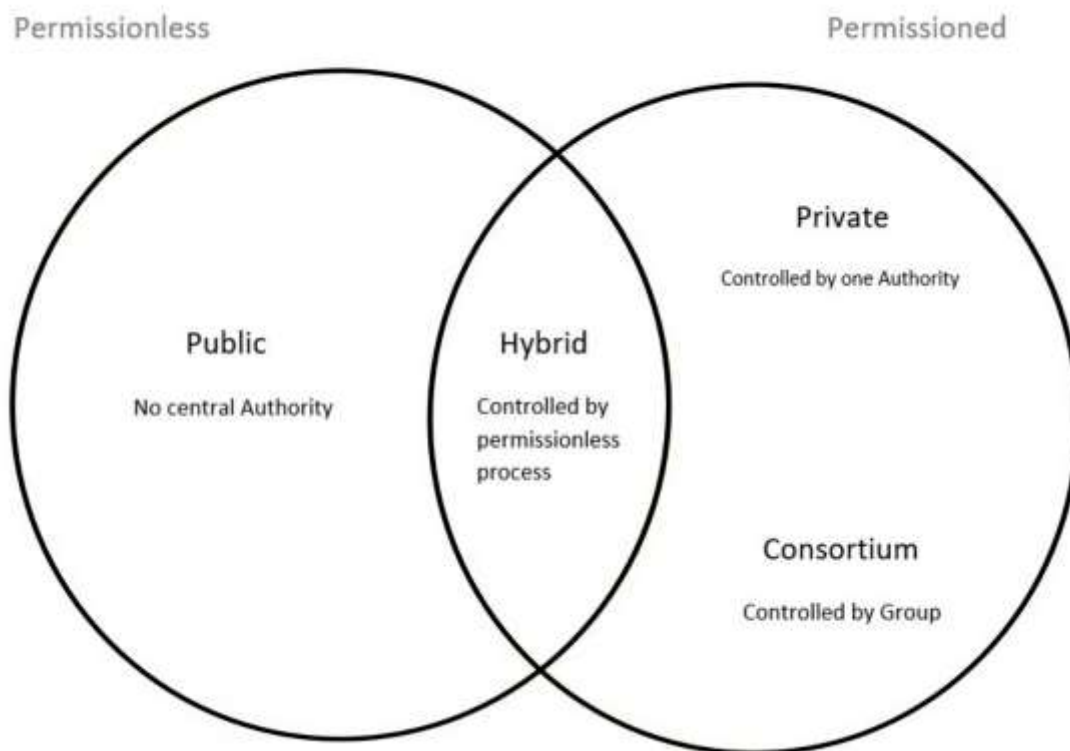


Figure 14: Types of Blockchain [34]

2.6.1 Permissioned Blockchain

A permissioned blockchain or private blockchain is one that can only be accessed by a select group of users who have been given permission; as a result, all the participants of the network are already known and trusted [35]. These are closed networks with limited decentralization in which the transactions are verified only by a few nodes, which are also pre-selected by the administrator. The users must have permission to access the ledger and are only allowed to operate within the strict confines of privileges granted to them by the administrator.

Since in permissioned blockchains, very few nodes are present, every change is visible to the administrator and can be easily tracked, and the user can be identified. Due to this extra layer of security, permissioned blockchains are becoming increasingly popular and adopted by big organizations. The permissioned blockchains tend to be faster with increased performance and are used in a network where high privacy and security are particularly important, such as verification of payments; smart contracts; by the bank to track money transfers; by companies for their internal auditing, voting and asset management. Examples of private blockchains are Hyperledger, Quorum, HydraChain, and Corda.

Features

- A key feature of permissioned blockchain is transparency based on the objective of the organization as the nodes or the users and their connections are known, and their transactions are visible [35]. Every action taken by the user is tracked and known.
- As a limited number of users are allowed in permissioned blockchains, lack of anonymity is another feature of permissioned blockchains [34]. The administrator is always aware of all the changes, and every change is tracked to the user.
- It does not have a central authority, but it is developed as a private authority which also authorizes decisions [36].

Advantages

- The main advantage of a permissioned blockchain is that no user can access the transaction information without permission, ensuring high levels of privacy and security.
- The permissioned blockchains are faster as they have only a few nodes for validation and therefore have increased performance and scalability [36].
- Customization is another advantage of permissioned blockchains. They are highly customizable as they can accommodate various configurations and integration as per the needs [35].

Disadvantage

- Due to a smaller number of nodes, permissioned blockchains tend to be faster with increased scalability and performance; however, this also increases the risk of corruption in permissioned blockchains.
- The owner or administrator can change the rules anytime as per the need, and therefore permissioned blockchains make it challenging to reach a consensus.

2.6.2 Permissionless Blockchain

A permissionless blockchain, also known as a public blockchain or trustless blockchain, is entirely the opposite of a permissioned or private blockchain. As the name suggests, the public blockchain is open to the public, which means anyone can participate in consensus and validate the data.

A permissionless blockchain is visible to everyone on the internet, and anyone can participate in current transactions as one individual or an organization does not own it. It is entirely decentralized; no administrator grants users access or gives them permission and rights to make the changes. Each user maintains a copy of the ledger on their local nodes, and they can all use a distributed consensus mechanism to determine the ledger's final state. Since a very large number of nodes are present, public blockchains have less user privacy with slow processing and high energy consumption. These are generally used in the network requiring high transparency; two eminent examples based on public blockchains are Bitcoin and Ethereum.

Features [35]

- A key feature of permissionless blockchain is that there is full transparency of the transactions to the users.
- Another characteristic of a public blockchain is that it is completely open source, as any user can easily access it and make the necessary changes.
- Unlike permissioned blockchain, public blockchain does not have any central authority. There is no administrator controlling the nodes in the network.
- It involves digital assets and heavy use of tokens.

Advantages [35]

- The major advantage of a public blockchain is security. As there is a large number of nodes present, there is greater distribution of records, and it is not possible for a bad actor to cause disruption because it requires corrupting 51% of the network, which is almost impossible.
- The second advantage of permissionless blockchain is that it has broader decentralization which allows more users to participate.
- Another advantage of a public blockchain is that it is censorship resistant due to a large number of nodes in the network.

Disadvantages

- Less user privacy.
- Due to large numbers of nodes, public blockchain tends to be slow and difficult to scale.
- It is not energy efficient as the proof-of-work and mining process to cryptographically add a block to a chain is highly energy consuming.

2.6.3 Hybrid Blockchain

We have studied both public and private blockchains up until this point. Private blockchains are closed networks with limited decentralization that only authorized users can access. Public blockchains, on the other hand, are open to the public and allow anyone to participate. The term "hybrid blockchain" refers to a system which is a combination of both private and public blockchains. In Hybrid Blockchains, a portion of the blockchain's data is managed by some organization, while the remainder of the data is accessible to the general public. As a result, a

business chooses Hybrid Blockchain because it offers the benefits of both permissioned and permissionless systems with very few drawbacks.

Due to their hybrid nature, these types of blockchains are generally used where the public accesses data but at the same time must be kept private. Therefore, it has a use case in healthcare, banking or financial, real estate, and government sectors.

Advantages

- The primary advantage of Hybrid Blockchain is that it is immune to 51% attack. Due to its hybrid nature, it works in a closed environment by making only a portion of the blockchain data public which prevents the hacker from execute the 51% attack on the Hybrid Blockchain network.
- In Hybrid Blockchains, only a few nodes carry out the verification of the transaction instead of all the nodes present in the network. Therefore, have less computational cost.
- Another advantage of Hybrid Blockchain is that it can be customized as per the needs. The rules can be changed in the Hybrid Blockchain anytime according to the need.
- Since a very smaller number of nodes are involved in the verification process, they offer high computational speed. Along with this, it offers good scalability.

Disadvantages

- Hybrid Blockchain is not completely transparent.
- It is very challenging to upgrade to Hybrid Blockchain. Moreover, it does not offer any incentive to the user for participating and contributing to the network.

2.6.4 Consortium Blockchain

Consortium Blockchain, also known as Federated Blockchain, seems quite similar to Hybrid Blockchain in some aspects. Consortium Blockchain is another creative approach to harness the features of both public and private blockchains. In this also some parts are public while other is private. Like private blockchain is controlled by a central authority, in a similar way, the consortium blockchain is also controlled but by more than one organization. Therefore, there is no single central authority making all the decisions. Due to this, the consortium blockchain is ideal for use at places where collaboration is required, like supply chain, food tracking, medicine, banks and other financial organizations, and research- for sharing the data and results of research outcomes. The consensus process in the consortium blockchain is controlled by pre-selected nodes. They ensure proper functionality, it has a validator node, which is responsible for initiating, receiving, and validating the transactions. Then there are members nodes in the network which can only initiate and receive transactions.

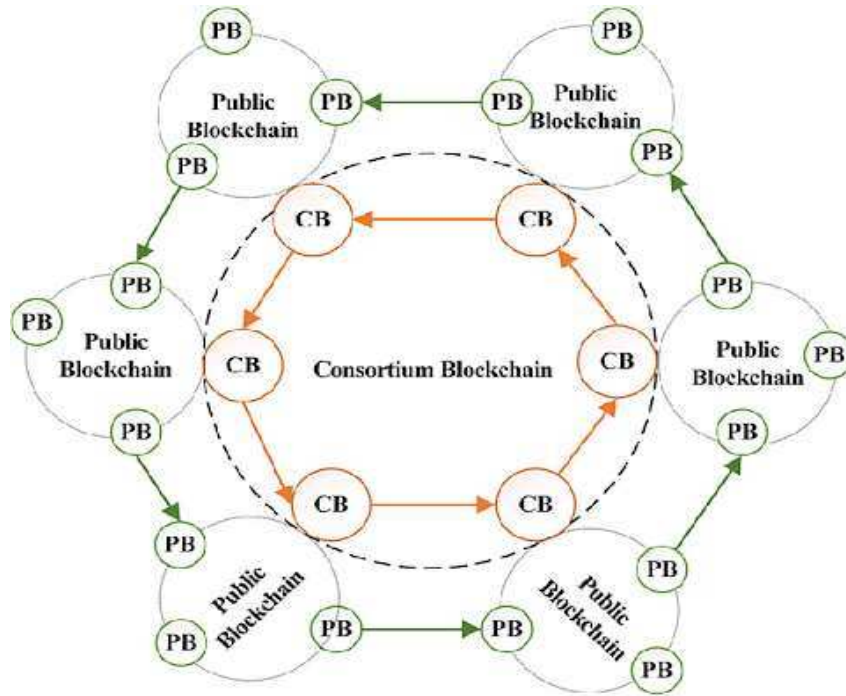


Figure 15: An example of Consortium Blockchain Network for Malware Detection [37]

Advantages:

- The Consortium blockchain is quite secure as it allows a limited number of nodes on the blockchain. Due to this, the data is untampered, secured and can only be modified if all the nodes agree.
- There is no transaction fee involved in consortium blockchain.
- It is also immune to 51% attack as the validation is done only by a trusted, preselected node.
- Another advantage of a consortium blockchain is that the central authority does not control it; instead, have multiple owners, which makes it more secure.

- It offers access control.
- Since there is a small number of users, which makes verification fast, making transactional speed very high.
- They have better scalability and are more efficient than public blockchain.

Disadvantages [34]

- As there are multiple owners from different organizations, this may lead to frequent disputes.
- There is lack of transparency.
- It is also less anonymous compared to other types of blockchain.

2.7 Features of Blockchain

2.7.1 Decentralization

One of the key features of blockchain technology is decentralization. The network is said to be decentralized when there is no single authority governing the system. In the blockchain, every node is independent and is not reliant on a single server node. The control is distributed among all the nodes in the network rather than relying on a central node. Therefore, it solves the problem of single-point failure of the conventional system.

Due to the decentralized nature of blockchain, a user can add their assets over the blockchain network, and no one can access it without their permission. It gives users full control over their property via a key linked to their account, which cuts down the involvement of the third party. Moreover, decentralization makes blockchain less prone to failure and offers transparency. All the participants on the network have a copy of the ledger, due to which every change is known to the administrator and traceable.

2.7.2 Immutable

Immutability is just another incredible feature of blockchain technology that only adds to its allure. That means that the blockchain network is unalterable, which makes it quite secure as it prevents data tampering and is one of the reasons for the increasing popularity of the technology.

The conventional databases follow the CRUD (create, read, update, and delete) operational model, which makes them prone to data tampering and hacking. However, in blockchain technology, every node has a copy of the ledger and to add a transaction, every node validates the transaction. After the validation from the maximum number of nodes, the transaction is added to the network, and every node gets the updated copy of the ledger. Moreover, once this transaction is added, no single node can delete or update it. That is why it is very hard to alter the data once added to the blockchain network. To change or delete, the intruder has to gain access and overwrite or delete the record from all or the maximum number of nodes before the next block is recorded, which is next impossible and expensive.

2.7.3 Faster Settlement

Another critical feature of blockchain technology which saves time and makes life easy for its users, is its ability to offer a faster settlement. The blockchain speeds up the process and facilitates faster settlement than conventional systems. This boosts the system's efficiency while also saving time and money in the long term.

The conventional systems, which are heavily paper based, require much time for processing and settlement. They are even prone to error and, most importantly, require third-party mediation. On the other hand, blockchain can complete the transaction much faster and efficiently, and all the documentation related to a transaction is stored on the blockchain itself, eliminating the need to exchange papers.

2.7.4 Consensus

One of the essential properties of blockchain technology is that the greater part of the nodes must validate all the transactions within the block before the block is recorded in the ledger. This reduces the likelihood of fraudulent transactions being added to the blockchain and builds trust in the technology. This is known as consensus and is a fundamental feature of the technology. Using the consensus mechanism, all actions are validated and confirmed on the blockchain.

The consensus is achieved through the algorithm, and there are different consensus algorithms for different blockchains. It can be compared to a voting system where the decision made by the

maximum number of nodes in the network is taken into account and must be accepted by all other nodes. Nodes might not trust other devices, but they can work smoothly by trusting the algorithm that runs at the systems' core, making the decision-making process more manageable.

2.7.5 Anonymity

Anonymity describes a situation where the acting person's name is unknown. Blockchain technology supports anonymity. Anonymity provides an efficient way of hiding the identity of users and keeps their identities private [38]. Anyone on a network can make the transaction without providing any identification; all that it needs is the receiver's payment address. Due to this feature, the use of blockchain for illegal works also increases. However, it is advantageous if use for right purpose like voting.

2.7.6 Increased Capacity

One of the significant features of blockchain technology is that it increases the entire system's capacity. In the blockchain, there is a large number of nodes/users/computers which together offer great power than the few devices in the centralized systems.

2.7.7 Security

Security is yet another important feature offered by blockchain technology. Firstly, blockchain does not have a central authority, which solves the problem of a single point of failure. Moreover,

no central authority means no one can simply add, delete, or modify the data for their benefit. This reduces the possibility of fraudulent transactions getting added to the network and the possibility of data tampering.

Blockchain also uses the technique of cryptography to maintain the security of the system. In the blockchain, all the records or information are encrypted, adding an extra layer of security. Everything is hashed cryptographically before it gets added to the network. All the blocks in the blockchain contain their hash value and the hash value of the previous block they are linked with. Therefore, changing anything will affect the current block and make all the other blocks invalid.

2.7.8 Transparency

The next important feature of the blockchain is transparency. Due to this, the complete history of all the transactions is publicly available. Anyone joining the network can view it and validate it. This also solves the problem of double spending and builds trust. In the blockchain, all the users have the copy of the ledger, which makes the working of the blockchain system more secure and transparent. All users of the blockchain have access to the ledger, making the system's operation more secure and transparent. However, the user's identity is not revealed through transparency. The methods of cryptography that are discussed in the report's concluding section are used to encrypt it.

2.7.9 Distributed

The blockchain is a decentralized and distributed ledger. This is a crucial aspect of the blockchain that makes it possible for the system to function quickly and effectively. This property of the blockchain enables users to verify their ownership in the network. In addition, every node has an equal weightage, and no single node has the power to add, delete or modify the block. A distributed ledger also makes it faster and easier to keep track of changes. As a result, in a distributed system, malicious activity detection and response times are reduced.

3. CONSENSUS MODELS

3.1 Introduction

Consensus means a general agreement has been reached, which is very important in blockchain technology. As in blockchain technology, there are multiple independent nodes, and it is essential that an agreement is reached between these nodes regarding the status of the ledger despite the failure of some nodes in a network. This process is called reaching consensus and is considered a backbone of the technology as it maintains the security of the network, addresses the problem of double-spending, and aims to make the network fault-tolerant. However, reaching a consensus in the blockchain is challenging as it is a decentralized and distributed system with numerous nodes participating in the consensus process. They need to reach to an agreement on a single version of the history of a chain.

There are many different types of consensus algorithms to establish trust between the nodes in a blockchain network. The type of consensus algorithm depends on the blockchain and its use. Apart from the method used to reach consensus, different consensus algorithms differ from each other in energy consumption or computational power required and security.

There are various requirements that must be met to provide the desired results in a consensus mechanism. The following describes these requirements: [12]

Agreement: All honest nodes must agree on the same value.

Termination: All honest nodes terminate execution of the consensus process and eventually reach a decision.

Validity: The value agreed upon by all honest nodes must be the same as the initial value proposed by at least one honest node.

Fault tolerant: The consensus algorithm should be able to run in the presence of faulty or malicious nodes called Byzantine nodes.

Integrity: Another requirement is that no node can make the decision more than once in a single consensus cycle.

These consensus algorithms can be broadly classified into two categories: Byzantine Fault Tolerance (BFT) based consensus algorithm and Proof-Based or leader election-based or Nakamoto consensus algorithm.

Byzantine Fault Tolerance (BFT) based consensus mechanism: It is a traditional mechanism that uses voting to reach consensus in a network. This mechanism consumes significantly less energy but has very low scalability. They work well with a limited number of nodes; therefore, the blocks and data inside the blocks get validated faster than the Nakamoto consensus mechanism.

Leader election-based or proof-based or Nakamoto consensus mechanism: The Consensus algorithm which is used in almost every type of blockchain network. It is well known for its use case in Bitcoin and Ethereum Blockchain. In this algorithm, nodes compete in an election where

they need to solve a complex computational puzzle. The first node to solve it, or the node that wins the election, gets elected as a leader. This node then validates the new blocks in the network. This mechanism works relatively slower than the BFT-based consensus mechanism and consumes much energy, but it is highly scalable.

Some of the well-known consensus mechanisms are:

3.2 Proof-of-Work

To understand the Proof-of-Work consensus mechanism, let's take a scenario where two nodes propose the blocks and broadcast them to the network. The receiving nodes will receive two proposed blocks. In this case, there is a probability that some will add a block proposed by node 1 and others will add a block proposed by node 2. Moreover, if the proposals are broadcasted simultaneously, the receiving nodes will not understand which one to add and will end up adding both blocks. This is a problem, as there is no consensus here. Some nodes have inserted a block proposed by one node, some have accepted from the second node, while others have inserted blocks proposed by both nodes. Another possibility in this scenario is that if those two nodes proposed the blocks simultaneously, then this could lead to the blockchain forks into two chains. That is, instead of a single chain or history of events, there are now two chains or a history of events.

To resolve this problem, let's introduce a rule in which when node encounters such a situation,

they can add a block to the longest chain they know and ignores the other chain(s). If there are two chains of a block with the same height, they can randomly select one, add a block to it, and propagate the decision onto the network. This is how the longest chain of blocks will take over, and there will be a single chain as the smaller chain(s) will be ignored by the nodes in a network.

Now there could be another problem. Let's assume that after the blockchain is forked into two, as per the rule, receiving node added the block to the longest chain and propagated the decision onto the network. However, there is a probability that other nodes do not hear this due to some reason and add the block proposed from another node. Again, there is no consensus, and this will again lead to different chains. The reason for this problem is that the nodes are validating and proposing the blocks quickly. The receiving nodes are receiving multiple proposed blocks from different nodes and therefore, not getting the time to converge.

Hence, a waiting timer is used to resolve this problem and allow the nodes to converge. This will slow down the process of block generation. Now, when one node proposes a block, the waiting timer will start, and another node will be proposed after the timer expires. This way, only one block will be added to the chain, and the above problem will be resolved. However, to make this solution work, the timer should be long enough, and the time of the timer should be increased as the number of nodes increases in the network.

However, in this case, there could be a possibility of an attack. If one of the nodes turns malicious and can find a way to expire its timer before the other nodes. Then this node will always be

validating and proposing the block onto the network. The solution to this problem is to cryptographically secure the timer so that no one can mislead by always expiring the timer first. The Proof-of-work consensus mechanism works on the same concept.

In the Proof-of-Work consensus mechanism, nodes are required to solve complex computation puzzle, which is time-consuming, and there is no alternative or easy way to solve the puzzle. Therefore, it is nearly impossible for a malicious node to mislead by always solving the puzzle first or, in other words, to expire its timer first and win the election. The Proof-of-Work is based on cryptographic hash functions in which the node is required to find a hash value of the block, which must be less than a specific value and must start with a certain number of zeroes [39].

$$\text{Proof of Work} = \text{SHA-256}(\text{SHA-256}(\text{Nonce} \parallel \text{Block Header})) < \text{target}$$

To find the correct hash value to win the right to validate and add a block to the ledger, the node needs to repeatedly try with different and all possible values of the nonce. A nonce is just a random number, changing the value of which node can reach to the correct hash value of the block. Once the node finds the correct value, it gets the right to validate the block. This node is called the miner node, it validates the block and broadcasts the block onto the network, and other nodes on the network verify the correctness of the hash value. Once the CPU effort has been expended to satisfy the proof-of-work, the block cannot be changed without redoing the work [8]. As later blocks are chained after it, the work to change the block would include redoing all the blocks after it [8].

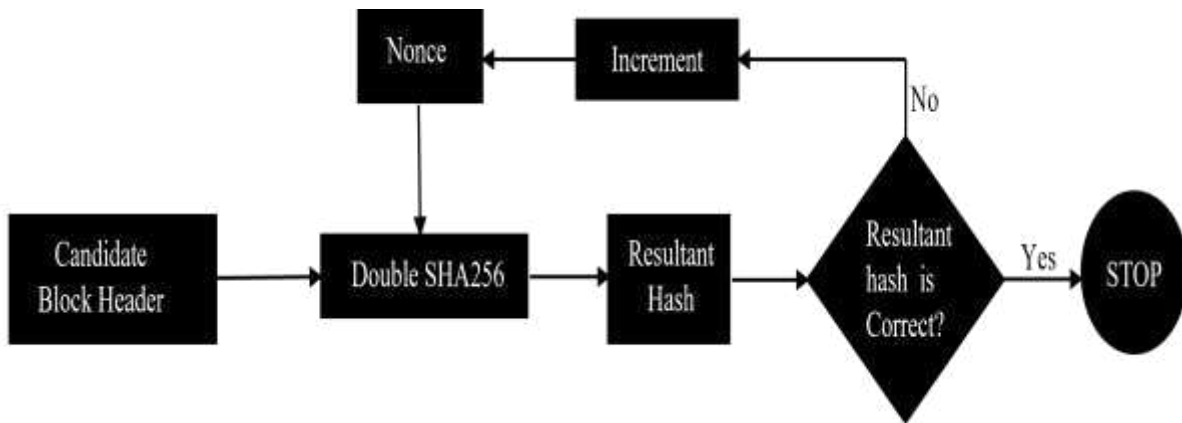


Figure 16: Calculating correct Hash Value

Now, there is a very good probability that two or more nodes will be able to find the correct hash value simultaneously. In this case, all these nodes can propose the block. This will lead to a fork in the blockchain. However, over time when more blocks are added to the blockchain, the algorithm will select the chain with the maximum proof-of-work or, in other words, the longest chain and all other chains will be ignored. This is how all the nodes will reach to an agreement regarding the state of the blockchain.

Therefore, proof-of-work consensus mechanisms introduce enough delay in the generation of the blocks, which resolves the problem of a fork and give time to nodes to converge. It also resolves the problem of malicious nodes being able to trick by always expiring its timer first and gaining the right to validate and add a block to the blockchain.

3.3 Proof-of-Stake

Another consensus mechanism in the list is the Proof-of-Stake consensus mechanism. Proof-of-Stake consensus algorithm consumes significantly less energy as compared to Proof-of-Work Consensus algorithm. In the Proof-of-Stake consensus mechanism, users need to stake a certain amount in order to have a chance to get selected as a validator. Then as per the set of rules, the system uses one of the staking nodes to validate the transaction on a network. The higher the amount, the higher the chance of getting selected, as it is believed that people with more currencies would be less likely to attack the network [22]. However, this could lead to centralization, as the rich user will dominate the network. Therefore, to resolve this, a few other things are taken into consideration in order to select the validator, like time spent on the blockchain, also called coin age, the lowest hash value, along with the staked amount. Along with this, the blockchain randomly selects the next validator; therefore, no user can predict their chance to get selected beforehand.

Proof-of-Stake consensus mechanism suffers a problem called Nothing-at-Stake [40]. In this mechanism, when the fork in the blockchain happens, there is no additional cost to users if they follow both chains to increase their chances of getting rewarded. Nothing-at-Stake problem is a big security concern as it makes the network vulnerable to attacks.

3.4 Delegated Proof-of-Stake

Delegated Proof-of-Stake (DPoS) consensus algorithm is a modification of the Proof-of-Stake consensus mechanism, which offer higher security. In the DPoS algorithm, the users of the network

who have staked their amount vote for their delegates, also called witnesses or block producers, to validate the transactions on their behalf and reach a consensus. However, only limited nodes with the most number of votes get the status of the witnesses and earn the right to validate the transactions in the blockchain. Therefore, with the limited number of nodes involved in the validation process, it offers faster transaction confirmation. Moreover, each vote by the node has a different weightage. The weightage of the vote depends on the amount node has kept on stake.

Now, the witnesses or the block producers validate the transactions in the block and receive rewards for this, which they have to share with the other nodes who voted for them. To maintain security in the system, the witnesses can be replaced anytime in Delegated Proof-of-Stake consensus mechanism. If any other node gets more votes, then it replaces the current witness with the least number of votes. Moreover, if the elected witnesses try to do a malicious activity like validating the fraud transaction(s), then also it can be replaced. The nodes with the amount on a stake can vote to elect another witness and vote out the malicious witness. It is used by Lisk (LSK), EOS.IO (EOS), Steem (STEEM), BitShares (BTS), and Ark (ARK) [41].

3.5 Proof-of-Authority

Another consensus mechanism which is a combination of the Proof-of-Stake and Proof-of-Work consensus mechanisms, is the Proof-of-Authority consensus mechanism. Proof-of-Work offers high security but consumes a significant amount of energy, whereas Proof-of-Stake is energy efficient, but security is an issue in the algorithm. Proof-of-Authority has features of both. It is more energy efficient and offers a good level of security. It was proposed by the co-founder and

former CTO of Ethereum, Mr. Gavin Wood, in the year 2017 [42]. The consensus mechanism uses Proof-of-Work in the first stage and later shifts to a Proof-of-Stake consensus algorithm. In this algorithm, nodes are neither required to solve any complex computational puzzle nor they have to stake any amount. Rather the validator is selected on the basis of their reputation. All the nodes interested in becoming a validator have to invest money and put their reputation at stake. A tough process reduces the risks of selecting questionable validators and incentivizes long-term commitment to the blockchain [43]. Since there is nearly no computational power required, this is one of the cheapest and most highly scalable consensus algorithms. Proof-of-Authority consensus mechanism offers a high transaction rate and fault tolerance. However, it has a high risk of corruption and manipulation of a third party, as the original identity of the validator is visible to everyone.

3.6 Proof-of-Activity

Similar to Proof-of-Authority, the Proof-of-Activity consensus mechanism is a combination of Proof-of-Work and Proof-of-Stake consensus mechanisms. It utilizes a new concept called “*Follow the Satoshi*” [12]. The Proof-of-Activity aims to combine the best features of Proof-of-Stake and Proof-of-Work consensus mechanisms. Therefore, it consumes less energy than the Proof-of-Work algorithm and offers better security than the Proof-of-Stake algorithm.

In the Proof-of-Activity consensus algorithm, the first step is the same as the Proof-of-Work protocol, wherein nodes have to solve a complex computational problem to get the right to validate

the block. The first node that solves the problem or calculates the correct hash value validates the block and proposes it onto the network. The process is the same till here. After this, the Proof-of-Activity consensus mechanism switches to the Proof-of-Stake algorithm in order to reduce energy consumption. Once the block is validated, the group of validators is selected at random. These validators are the nodes that have kept their amount at stake. Similar to the Proof-of-Stake algorithm, the higher the amount, the higher the chances of getting selected as a validator. These validators are then required to sign/validate the block proposed by the node at the first stage. Only after the block is signed by all the validators, it gets the status of the complete block and gets added to the blockchain. In the Proof-of-Activity consensus mechanism, the rewards earned are shared between the mining node and all the validators based on their roles in the process.

Although it consumes less energy as compared to the Proof-of-Work consensus mechanism, the overall energy consumption is still high, which is the disadvantage of the Proof-of-Activity consensus algorithm. It is used by Decred (DCR) and Espers (ESP) blockchain projects [41].

3.7 Proof-of-Importance

Proof of Importance (PoI) is another consensus mechanism used in blockchain technology. It was first used in NEM (New Economy Moment), an easy and inexpensive blockchain-based platform to manage data or assets. In the Proof-of-Importance consensus mechanism, the node is selected based on the importance score of the node. The higher the importance score, the higher the chances of getting selected, similar to the Proof-of-Stake algorithm, where the node with a high stake has a higher chance of getting selected as a validator node. The validate node then gets the chance to

validate the block, which is broadcast onto the network for the other nodes to verify and add it to the blockchain.

The importance score is based on various factors like the number of transactions done by the user in the last month and the size of those transactions. Along with this, overall usage and movement of the user are also monitored to establish trust, which also affects the importance score. Many other factors are considered for the overall importance score of the user. Like in NEM technology, the amount vested by the user is also considered for the importance score. NEM uses XEM cryptocurrency, so whenever a node receives the XEM, 90% of the amount gets added as the unvested balance, whereas 10% of the amount gets added as the vested balance. This vested amount is considered for the importance score. The higher the amount, the higher the chances of getting selected.

3.8 Proof-of-Burn

Proof-of-Burn (PoB) is an alternative to the Proof-of-Work consensus mechanism, which consumes less energy. POB is often called a POW system without energy waste [44]. In the -of-Burn consensus algorithm, the nodes need to burn their coin to gain the right to validate the block. To do so, they send their coins to a verified address, and the burned coins are not returned to the node. The node can burn either the currency used in that particular blockchain or another currency, depending on the implementation of the consensus mechanism in the network. Similar to the other consensus mechanisms, the higher the number of coins burnt, the higher the chances of getting

elected to write the next block. However, in the Proof-of-Burn consensus mechanism, the validator is elected for a certain timeframe which is proportional to the number of coins burned.

Although the Proof-of-Burn is an energy-saving alternative to the Proof-of-Work consensus mechanism, it still wastes resources. The major setback is that the validating rights will only go to those nodes which are ready to burn more money. Cryptocurrencies that use the proof of burn protocol include Slimcoin (SLM), Counterparty (XCP), and Factom (FCT) [41].

3.9 Proof-of-Capacity

Proof-of-Capacity is one of the most efficient consensus mechanisms first used in Burstcoin Cryptocurrency. In Proof-of-Capacity, nodes do not get elected based on the amount staked or burned, nor are they elected based on their information score or reputation. The nodes in this consensus mechanism use their hard-drive space for mining.

In the Proof-of-Capacity consensus mechanism, nodes prepare the list of all the possible hashes for a block header before the consensus process starts. Using the different nonce values, possible hash values are calculated and stored in a node hard drive. Therefore, the more the hard-drive space node has, the more values it can store. Once the consensus mechanism starts, they simply match all these possible hash values to check which one is correct. Like Proof-of-Work, the first one to find the correct hash value is elected to validate the next block. Therefore, the greater

number of possible hash values the node has in the hard drive, the higher the chances of finding the correct hash value and getting elected.

Since the Proof-of-Capacity consensus mechanism does not require expensive equipment except the large capacity hard drive and does not require nodes to invest or put anything at stake, more users in a network can participate and get a chance to be a miner. Some well-known examples of blockchain using the Proof-of-Capacity consensus mechanism are Burstcoin, Chia, and SpaceMint.

3.10 Practical Byzantine Fault Tolerance (PBFT)

The Practical Byzantine Fault Tolerance consensus algorithm was introduced by Barbara Liskov and Miguel Castro. The mechanism was designed to work in a system where there are malicious nodes. The aim was to solve the problems associated with existing Byzantine Fault tolerance solutions. It came out as an effective solution that achieves the practical Byzantine state machine replication to reach the consensus in the presence of malicious nodes. The condition for the PBFT algorithm to work well is that the total number of nodes should be more than three times the number of malicious nodes in the system (if m is the number of malicious nodes, then the total number of nodes should be at least $3m + 1$ or more).

In the PBFT consensus mechanism, the nodes participating in the consensus process are divided into two categories: primary nodes and secondary (backup) nodes. The primary nodes are changed

in every consensus round, called a *view* in the PBFT algorithm. The client communicates with the primary node by sending the request, which is further broadcast to all the secondary (backup) nodes in the network. All the nodes in the network then work on the client's request and send a reply back to the client. If a primary node tries to perform any malicious activity, then honest nodes can vote to replace it with the next leader node. The PBFT consensus mechanism is energy efficient but has high latency and low scalability because of high communication overhead, which increases with the increase in the number of nodes in the network.

4. Security in Blockchain

4.1 Introduction

The era in which the internet and physical worlds are increasingly interconnected to create environments that are more intelligent and integrated. The number of people or businesses using technology is continually growing. In today's world, where technology is rapidly evolving and expanding, security and privacy are the biggest concerns. To securely store and protect user information or data is quite challenging. However, in recent years, the blockchain has gained considerable appeal and has been employed in various sectors other than digital currency due to its ability to offer high security. Blockchain is considered one of the best solutions when data needs to be stored and distributed securely and efficiently.

We have examined the blockchain's architecture, features, types, consensus mechanisms, and other aspects up to this point. However, what makes blockchain so secure is the most important question about technology. To comprehend this, we need to understand the basic data flow in blockchain technology.

As we can see in the *figure 17*, generally, the user (node) connects with the blockchain via some channel. This channel could be distributed applications (dApps), online web-based wallets with MultiSig authentication, or third-party organizations or exchanges [45]. The user first gets authenticated, and only after that does it get connected to the blockchain network.

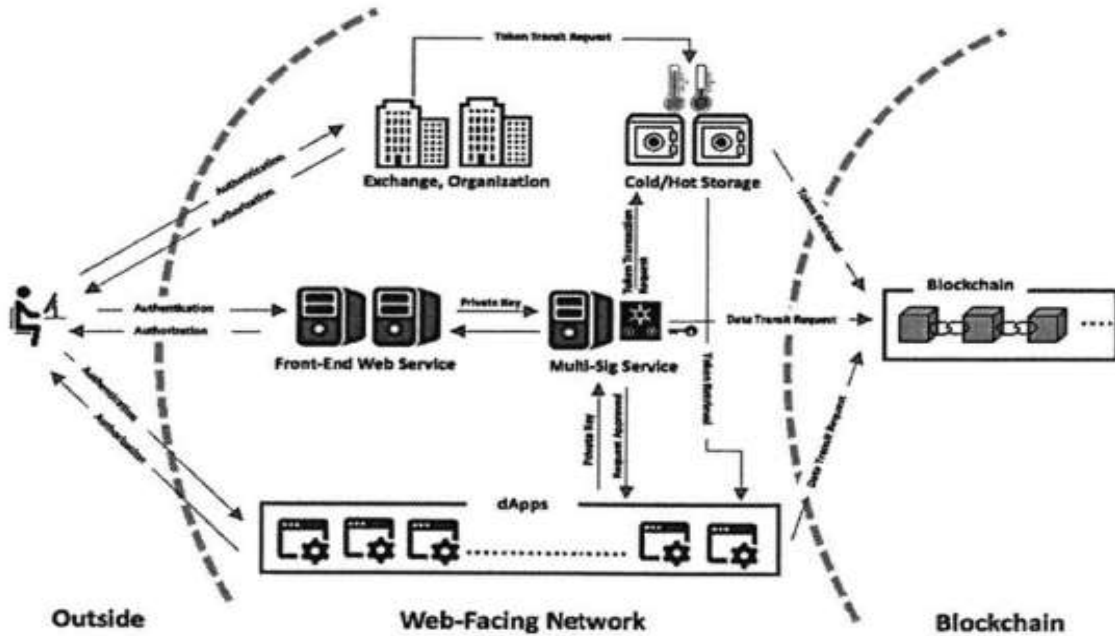


Figure 17: General data flow in Blockchain System [45]

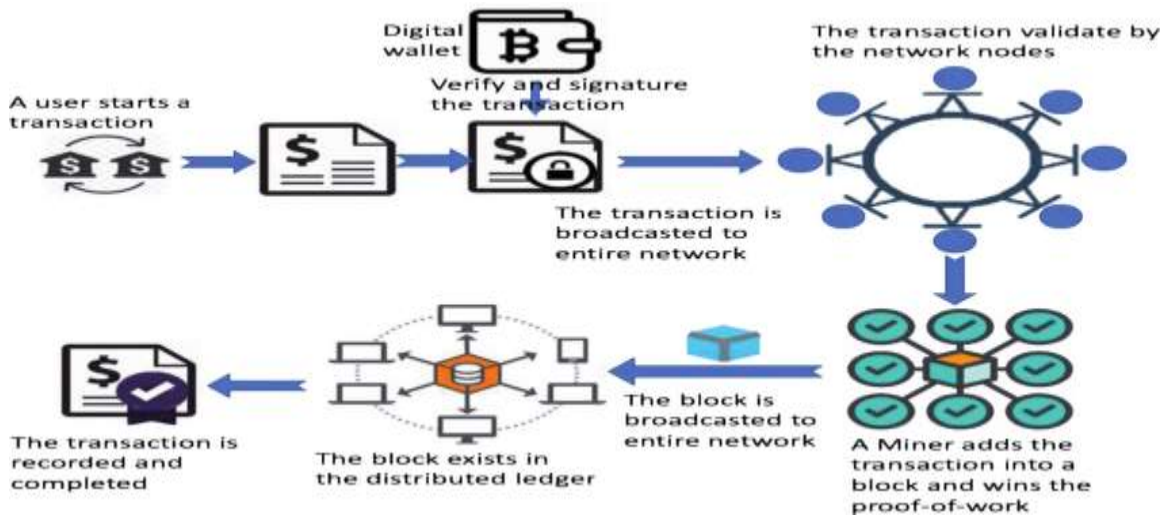


Figure 18: Data Flow in Blockchain Network [45]

As we can see in the *figure 17* and *figure 18* that once the user is authenticated, no security control filters or data protection parameters are in place for the data coming from the node. This is due to

the nature of blockchain technology to give open access to the user to operate autonomously. Therefore, blockchain security is built on cryptography-based authentication and hashing.

Blockchain technology's security relies heavily on cryptography. Most of the cryptographic primitives are used in blockchain technology to secure the data, like asymmetric key cryptography for the identification of users and authorization of transactions. In the application layer, digital signatures are used for validation purposes. Hashing is used in consensus mechanisms like proof-of-work, maintaining the integrity of the block and much more.

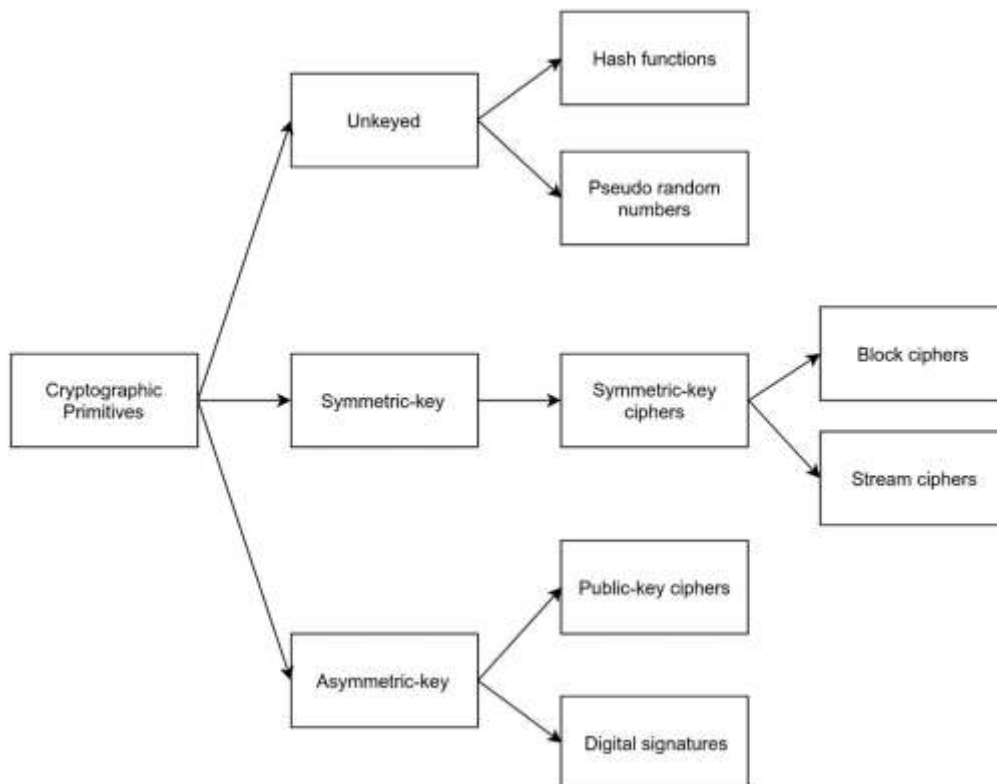


Figure 19: Cryptography Primitives [9]

4.2 Cryptography

Cryptography is the process of hiding or coding information so that only the person a message was intended for can read it [46]. The word cryptography is derived from two Greek words, Kryptos meaning “hidden,” and Graphein, meaning “to write”. The primary aim of using cryptography is that it enables people to share information over an insecure channel without the fear that the information will be stolen or tampered with. When the sender starts the communication, cryptography is used to convert a plaintext message to an encoded message called ciphertext before sharing it over an unsecured medium. Cryptography uses certain algorithms to encode messages. Cryptographic key generation, digital signing, and verification are all performed with these algorithms to safeguard confidential transactions and the privacy of online data and web browsing [47]. Only the intended recipient will be able to decipher the ciphertext or encoded messages.

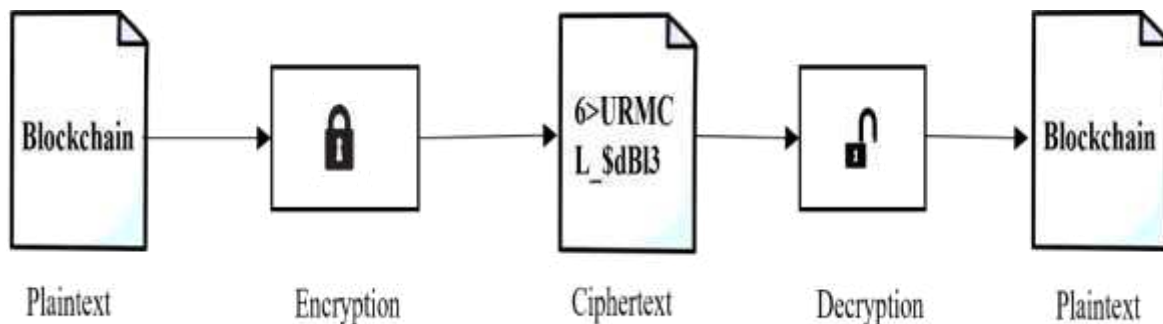


Figure 20: Conventional Cryptography Process

Figure 20 illustrates the conventional cryptography process, where plaintext is first converted to ciphertext using an encryption algorithm and shared over the unsecured channel. Only the user for

whom the message is intended can decrypt it back to the original text.

The technique of cryptography has been in use for millennia and has constantly advanced during these years. Cryptography is used in a variety of applications today, including secure web browsing, digital signatures, end-to-end encryption, authentication, and blockchain. In the blockchain, cryptography plays an important role. It is used in the consensus and application layer of the blockchain. On the blockchain network, it is utilized to authenticate users, preserve their security, and safeguard transactions. Apart from this cryptography also provides integrity, accountability, and non-repudiation. Cryptography is broadly divided into two categories Symmetric Key Cryptography and Asymmetric Key Cryptography.

Symmetric Key Cryptography doesn't play any role in the security of the blockchain. The symmetric-key cryptography, also called secret-key cryptography, is a key-based cryptography algorithm in which the same keys are used to perform the encryption of plaintext to ciphertext and decryption of ciphertext to plaintext. The participants must communicate these keys through a secure channel. Therefore, it is also known as shares-key cryptography. It is fast, simple, and requires less computational power. However, the algorithm's only issue is securely sharing keys among participants. Data Encryption Systems (DES) and Advanced Encryption Standards (AES) are two popular examples of a symmetric-key cryptography algorithms. There are two types of cryptography algorithms: stream cipher and block cipher.

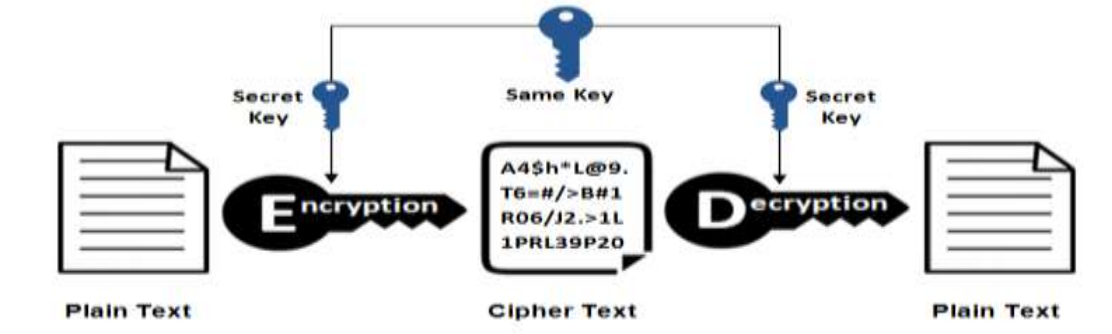


Figure 21: Symmetric Key Cryptography

In Stream cipher, each plaintext character is encrypted one at a time to convert to ciphertext. It uses a pseudorandom string that serves as a keystream. The Digital shift registers are used to create this pseudorandom string from a random seed number. This seed value is used as a cryptographic key for decrypting the ciphertext stream to plaintext. For a stream cipher to be secure, its pseudorandom generator should be unpredictable, and its seed value used to generate the keystream should never be reused to reduce possible attacks [9]. Stream Ciphers are of two types of synchronous stream ciphers and asynchronous stream ciphers. In synchronous stream ciphers, the keystream is only dependent on the key whereas in asynchronous stream cipher, the keystream is dependent on the encrypted data.

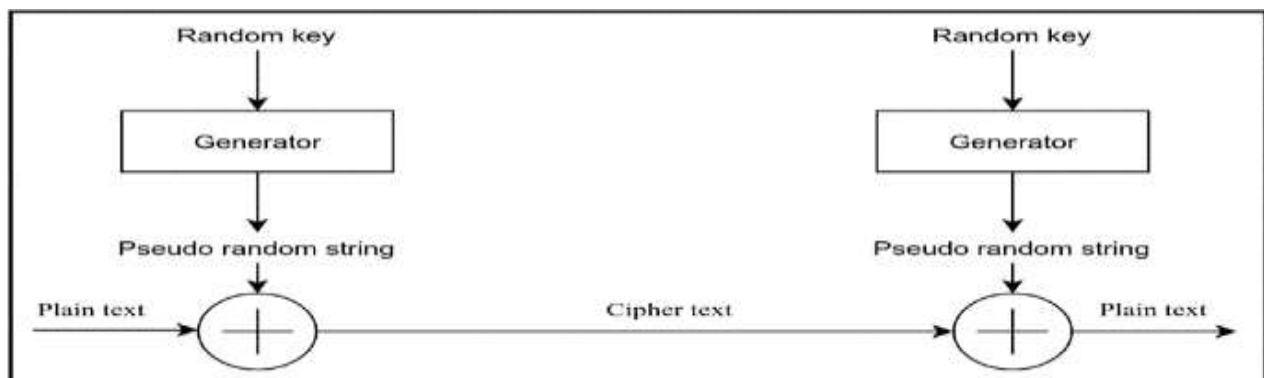


Figure 22: Stream Cipher flow diagram [9]

In a Block Cipher, as a name suggest, one block of information is encrypted at a time. It is built using a design strategy called Feistel Cipher. These ciphers are also the building blocks of other cryptographic protocols, such as hash functions and random number generators.

4.3 Asymmetric Cryptography

The same keys are used for encryption and decryption in symmetric key cryptography. Communicating these keys between the participants over a secured channel is the biggest problem of symmetric key cryptography. Moreover, if we have a secured channel for sharing the keys, then there is no point in encrypting the data in the first place; the data can be shared over this secure medium. This is what asymmetric key cryptography intended to solve. In asymmetric key cryptography, two different keys are used for encryption and decryption. It is called public-key cryptography and takes a long time for execution.

Let's look at an example to understand the process of asymmetric key cryptography. Let us say two friends, Rahul and Varun, have to share confidential documents over an unsecured medium. In asymmetric key cryptography, both of them will first generate the keypairs called the public key and private key. The public and private keys are mathematically linked together in such a way that the private key can not be derived from the public key. The public key is used for encrypting, and the private key is used to decrypt the data. The private key is private to the owner, and it is the owner's responsibility to keep it confidential because anyone with knowledge of it has the ability to decrypt and read all the private information intended for you. So, in order to share the sensitive documents over an unsecured channel using asymmetric key cryptography,

Rahul and Varun will share their public keys with each other. Public keys can be shared over the unsecured channel, as it is used for encrypting the data. However, only the authorized user can decrypt the data using the private key.

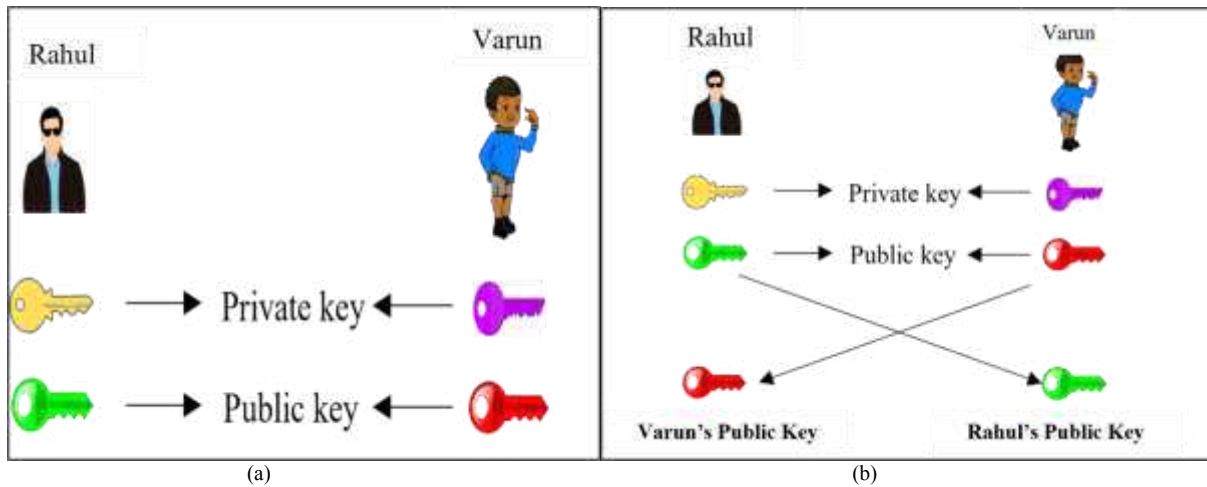


Figure 23: Asymmetric Key Cryptography (a: Generation of Keypairs, b: Sharing Public Key with the participants)

Now Rahul will encrypt the document with Varun's public key, digitally signed it and share it with Varun. We will discuss the concept of digital signatures in the next section. Now only Varun can decrypt the document using its private key. This can be seen as a mailbox whose address can be known to anyone, and anyone can drop/send mail to that mailbox. So, the mailbox address is like a public key in asymmetric key cryptography. However, only the mailbox owner has the key and can open the mail, similar to a private key in asymmetric key cryptography. In the same way, Varun can share the sensitive document with Rahul over an unsecured medium.

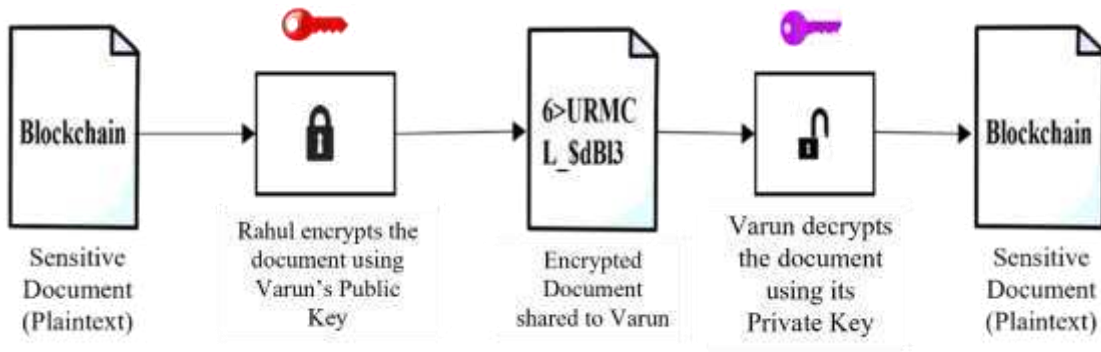


Figure 24: Encryption/Decryption using Public/Private keys

4.3.1 Applications of Asymmetric Cryptography in Blockchain Technology

Asymmetric Cryptography has two applications in blockchain technology:

1. Identity Management
2. Authentication

➤ Identity Management

Users are able to use the blockchain or create accounts on the blockchain without disclosing their true identities because of the pseudonymous nature of the technology. However, to map owners to their properties, blockchain, on the other hand, needs to identify the users. In blockchain technology, asymmetric cryptography is used to identify user accounts and for identity management.

The account numbers in the blockchain are known to the general public as public keys. Therefore, only public-private key pairs with the relevant address can be used to create valid accounts. As a result, users can remain anonymous because the private key is nothing more than a random number that has no connection to the users' actual identities [48]. In addition, the public cryptographic key is used by transaction data to identify the ownership transfer accounts.

➤ **Authentication**

Asymmetric Cryptography is also used to authorize transactions in blockchain technology. When transactions occur in a blockchain, asymmetric cryptography is used for authorization. Whenever a transaction occurs, the sender always includes data that proves the account owner made the transaction and the owner has agreed to transfer the ownership. The transaction data includes the sender address, receiver address, amount, and much more. This is then signed using the owner's private key. Now, when other participants who are intended to receive this can verify the transaction using the public key and hence, authorize the transaction and add it to the block.

4.4 Digital Signatures

We have been using handwritten signatures for millennia to verify documents. They are used for making agreements, contracts, transferring assets and many more things. Digital Signatures are similar to handwritten signatures but are more secure as it is nearly impossible to forge someone's digital signature. They are used to provide proof of ownership of digital documents.

Digital Signatures are one of the cryptography primitives that play a crucial role in blockchain technology. The primary aim of blockchain is secure communication between participants without a trusted third party. Therefore, it is critical to have a system that creates trust between the participants. To comprehend it better, let's return to our previous scenario, where Rahul wants to send a sensitive document to Varun. There are certain problems in this scenario:

- How can Varun trust that this document comes from Rahul?
- Once Varun has received the message, Rahul could always claim that he has sent that document.
- Another possibility is that the document could be tampered with by the hacker in between (Man in the Middle attack).

Therefore, digital signatures are used in blockchain technology to authenticate messages. The sender of the transaction digitally signs the message using the private key before broadcasting it onto the network. The recipients of the message can then authenticate it using the public key of the sender and the hash function. This is how digital signatures provide the solution to the problem of authentication, non-repudiation, and man-in-the-middle attack. By guaranteeing the sender's authenticity and message integrity, it fosters trust among participants. Moreover, digital signatures are unforgeable and non-reusable, improving the blockchain network's overall security. This means anyone's digital signature can not be forged. Therefore, only the sender can sign the document using its private key. It is also non-reusable, meaning no one can separate it from a message, document, or transaction and reuse it with another message, document, or transaction. The process of Digital Signature consists of two parts: Creating a signature and Verification.

Creating a signature

In blockchain technology, we have two approaches to use a digital signature. The first approach is to sign and then encrypt. In this approach, the sender first digitally signs the data using its private key, and then the data and the digital signatures are encrypted using the receiver's public key. The second approach is the opposite of the first. In this approach, the sender first encrypts the data using the receiver's public key and then digitally signs the encrypted data using its private key. The first approach is considered more secure than the second one.

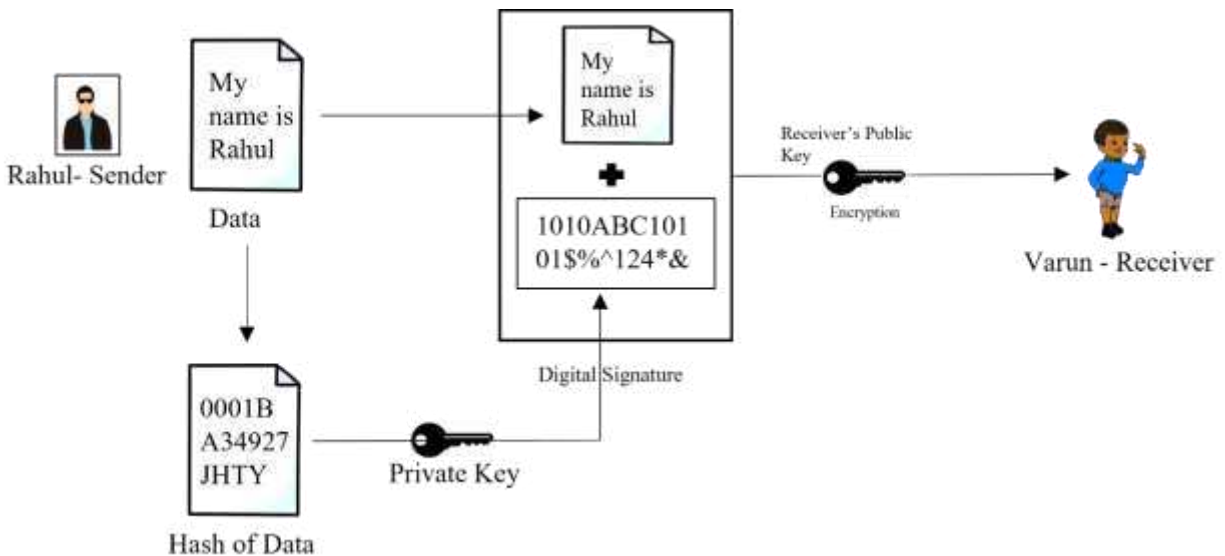


Figure 25: Illustration of Digital Signing (Sign then Encrypt approach)

Let's say Rahul wants to send a message to Varun which says, "My name is Rahul". So, first of all, Rahul and Varun will use asymmetric-key cryptography to generate keypairs and share their public key with each other. Now, Rahul will first calculate the hash value of the data using the hashing function. Then the hash of the data will be digitally signed by Rahul using its private key. Now,

the digital signature is combined with the original message, which is called a digitally signed message and is shared onto the network. The digital signature is created by the formula [49]:

$$S = F_s (F_h (m) , dA)$$

Here F_s - signature function, F_h - hashing function, m – message, and dA – private key of sender

Verification Process

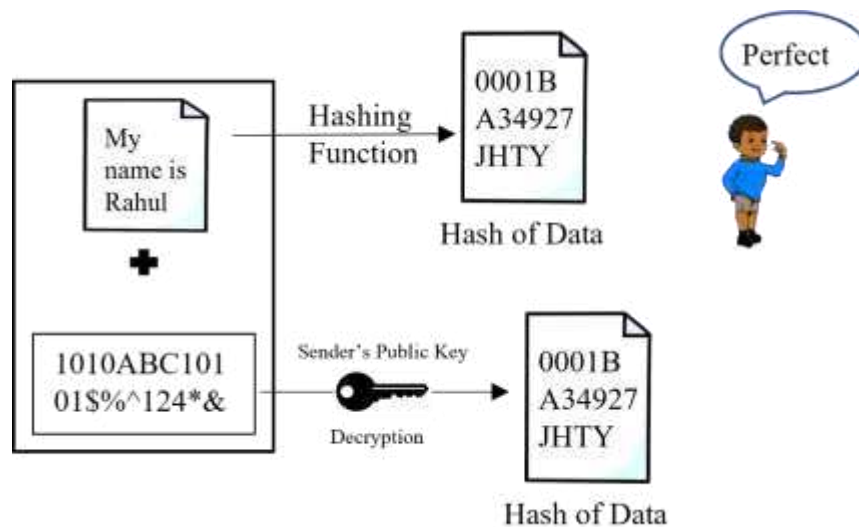


Figure 26: Verification of the Signed Message

The digitally signed message by Rahul is sent onto the network. Now, anyone who possesses public information can verify it. In this case, Varun wants to verify this, for which Varun will first calculate the hash of the data. After this, Varun will decrypt the attached encrypted message using Rahul's (sender) public key, which will give him the hash value of the data. If both the hash values

are same, then the Varun (recipient) will conclude that the message is signed by the original owner (Rahul) and the message is not tampered with. However, let us say that before Varun received it, the hacker was able to alter the message, and the hacker changed the message to “My name is Ankit”. When Varun receives this, again, he will calculate the hash value of the data and decrypts the cipher text using Rahul’s (sender) public key. However, this time both the hash values will be different. Therefore, Varun will conclude that Rahul did not authorize this message or that the message was tampered with; hence, Rahul is not accountable for this.

4.4.1 RSA

Rivest Shamir Adleman Algorithm (RSA Algorithm) is one of the earliest and the most popular asymmetric cryptographic algorithms. Though RSA is mainly used for encrypting messages but can also be used for digitally signing a message in the blockchain. It uses the principle of prime factorization to produce the keypairs, which act as a trapdoor function [50]. The two distinct large prime numbers are selected to compute the public and private keys. The private key is use for decryption and kept secret. The public key is available to everyone and is use for encryption. Along with the private key, the prime numbers are also kept secret. It is nearly impossible to compute the private key from the public key, provided that the two selected prime numbers are large.

4.4.2 Elliptic Curve Cryptography (ECC)

Elliptic-curve cryptography (ECC) is an alternative technique to RSA used in digital signatures. The difference between the ECC and RSA is that ECC requires smaller keys and signatures for the

same level of security. For example, a 256-bit ECC key provides the same level of security as a 3072-bit RSA key. Due to this, it provides fast signatures, fast key generation and agreement. In ECC, the public-private keys are generated based on the elliptic curve equation instead of computing it from the large prime numbers. It requires less space and processing power to operate, and therefore it is gaining popularity in embedded systems and other devices having limited storage or processing power.

4.5 Hashing

A hash function is another primitive of cryptography which is used in blockchain technology. The process of producing an output "hash value" or "hash code" or "hash digest" of a fixed length for any input data, regardless of its size or length, is known as "hashing" [51]. The hash value can be compared to the fingerprints of a human being. As the fingerprint of every human is unique and is impossible to alter, similarly, the hash value generated using a hash function for every data in the blockchain is nearly unique and impossible to change.

Once the data is recorded in a block, its hash value is calculated. Every block in a chain contains data (depending on the type of blockchain), the hash of the block and the previous block hash. Since each block contains a hash value of the previous block, they effectively form a chain. Changing anything in a block will lead to a re-calculation of the hash value, hence will cause all the following blocks in the chain to be invalid. Therefore, hashing maintains the integrity of the blocks in a blockchain network. The hash value of any input data (single character, an audio file,

n page word document, etc) will be of the same length; therefore, it might contain leading zeroes to generate the output of a required length.

Hashing is a one-way function, unlike symmetric and asymmetric cryptography, which are two-way functions. This means that when the data is encrypted using symmetric or asymmetric cryptography and sent to another person, the person can decrypt the data to the plaintext. That is why these two cryptography functions are two-way functions. However, hashing is a one-way function, which means the generated hash value is not intended to be decrypted to the original data.

There are various hash functions available that differ from each other based on resulting digest size (length of hash value they generate), implementation or construction technique etc. The few quite eminent hash functions are Message Digest (MD), Secure Hashing Algorithm (SHA), RACE Integrity Primitives Evaluation Message Digest (RIPEMD), and Whirlpool.

4.5.1 Properties of Cryptographic Hash Functions

Hashing is backbone of blockchain technology. Therefore, cryptographic hash functions should possess the following properties.

1. Collision Resistance

One of the major properties of a hash function is that it should be collision resistant. A collision of hash means that there are at least two distinct pieces of data that produce the same hash value. A hash function is considered safe if it is infeasible to determine two values, x and y , such that $x \neq y$, yet $H(x) = H(y)$ [52]. x and y are two distinct inputs, but when input into the hash function produces the same hash value. It is like two human beings having the same fingerprint.

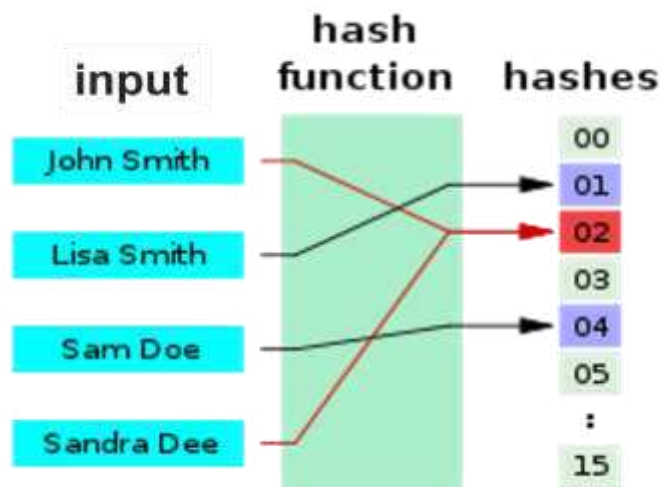


Figure 27: Collision of hash value [53]

It can be seen in the *figure 27* that for two distinct inputs, John Smith and Sandra Dee, the hash function produces the same hash value, i.e. 02. Though it is required that hash functions are collision resistant, the collision does happen. The reason behind this is that the hash function input contains all the inputs of different lengths, but as we know, the hash function produces the output of a particular length. This means the possible output values are less than the possible inputs. The input space is interminable, whereas the output space is limited. Therefore, there will be a very large number of possible input sources that lead to

a particular output. To find a collision, if we calculate the hash value of 2^{130} inputs, there is a 99.8% possibility that the hash function will produce the same output for two inputs. In reality, no hash function has been proven collision-free, but they should be computationally impractical to find. This is where the concept of the **avalanche effect** comes into play. The avalanche effect states that the slightest change in the input will result in an entirely different output hash value. This means that just by changing a single character in an input, the hash value can be changed, which helps when the hash function collides.

2. Preimage Resistance

Preimage resistance means that there is no way to track the input from the output. In other words, it should be infeasible to find the input value from the hash value. It is a very important property of a hash function. The output hash function should not say anything about the content of the input data.

If $H(x) = m$, where h is the hash function, x is the input value, and m is the output hash value. Then the preimage resistance property says that it should not be possible to reverse compute the output hash value “ m ” to the input value “ x ”. The preimage resistance property is also known as the one-way property.

3. Second Preimage Resistance

Given an input value x and hash $h(x) = h(x)$. The second preimage resistance property states that there should be nearly impossible to find an input value y , such that $h(y) = h(x)$ or $h(x) = h(y)$ provided that $y \neq x$. This property is also known as weak-collision resistance [54].

4. Quickly provide the hash value of any type of data

Given an input value of x , then its hash value $h(x)$ should be calculated very quickly. In other words, the hash function should be capable of calculating the hash values of input messages of any size or length quickly. However, at the same time, the hash function should not return garbage or error messages. Then only the application can be used to generate and compare the large volume of hashes for their trueness [55].

5. Random Oracle or Pseudorandom

This property states that the hash functions should be designed in such a way that their hash values are unpredictable and random. Pseudorandom means that the output hash value should change unexpectedly with a slight change in the input value. It should not be possible to predict the hash value from the input data. One can send any input to the hash function, and the output will always be completely random and unexpected.

4.5.2 Patterns of Hashing

There are 5 types of patterns of hashing techniques [56]:

1. Independent Hashing
2. Repeated Hashing
3. Combined Hashing
4. Sequential Hashing
5. Hierarchical Hashing

INDEPENDENT HASHING

The first pattern of hashing techniques is called Independent Hashing. In this pattern of hashing techniques, each piece of data is individually fed as an input to the hash function. An example of Independent Hashing is shown in *figure 28*, where “Blockchain” and “technology” are two individual pieces of data and are hashed separately.

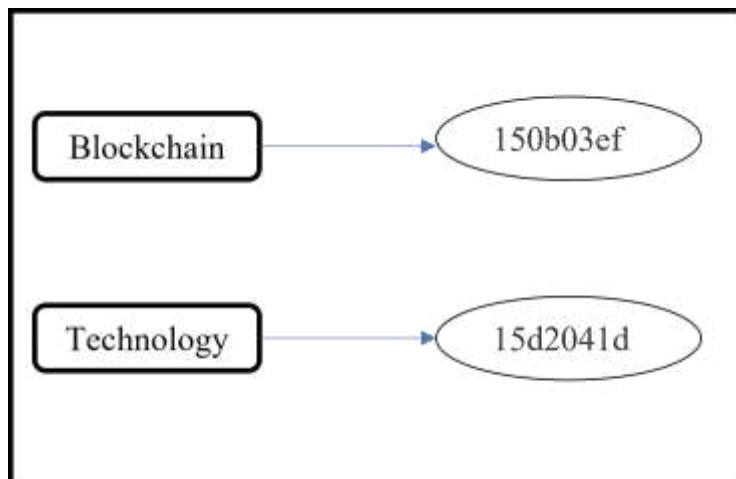


Figure 28: An example of Independent Hashing

REPEATED HASHING

In this pattern of hashing techniques, the input value is first hashed using the hashing function to produce a hash value. This output hash value is again fed as an input to the hash function to produce the final output hash value. That is why it is called a repeated hashing pattern. An example of a repeated hashing pattern is shown in the *figure* below.



Figure 29: An example of Repeated Hashing Pattern of hashing techniques

In the above example, “**Blockchain**” is fed as an input to the hashing function, which produces “**150b03ef**” as an output. This is again hashed to produce a final hash value of “**08960227**”.

COMBINED HASHING

The third pattern of hashing techniques is Combined Hashing. In this pattern, more than one piece of data is combined to produce a single hash value. This pattern of hashing techniques is useful when individual pieces of data are very small. Combining them reduces processing power, time, and space. It can be seen in the *figure 30* that the two individual pieces of data are combined into a single data input and given to the hash function to produce a single hash value. The hash value obtained is the same as the first output of the repeated hashing pattern shown in *figure 29*.

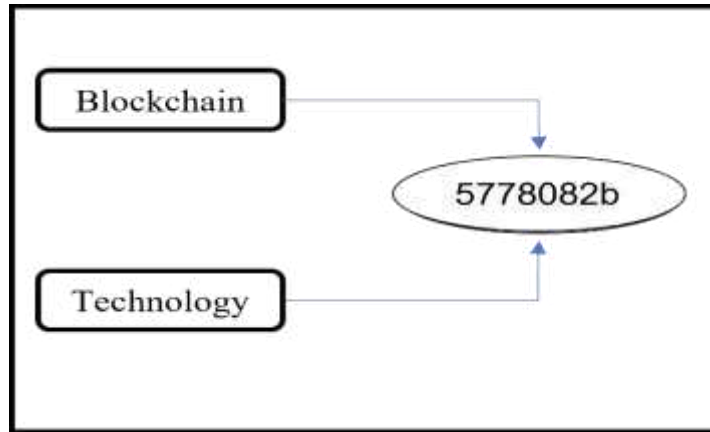


Figure 30: An example of Combined Hashing Pattern

SEQUENTIAL HASHING

Another type of pattern of hashing technique is Sequential Hashing. In this type, the hash value is updated as soon as a new piece of data comes in. It uses the Combined and Repeated Hashing pattern simultaneously. Sequential Hashing combines the current hash value output with the new piece of data and gives it as an input to the hash function to produce an updated hash value. An example of sequential hashing is shown in the below *figure*.

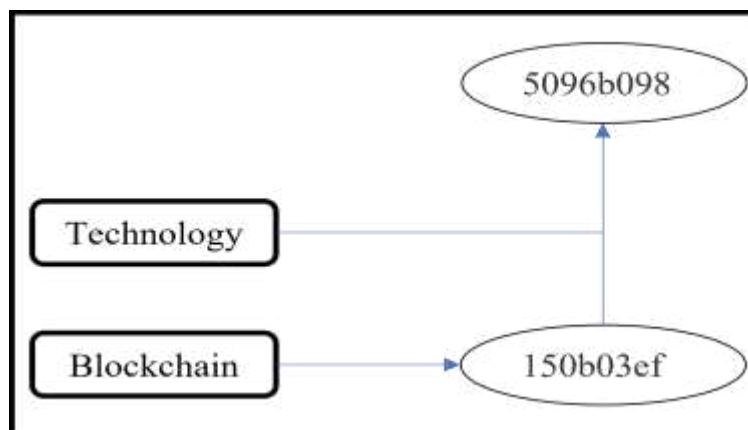


Figure 31: An example of Sequential Hashing

HIERARCHICAL HASHING

The last pattern of hashing techniques is hierarchical hashing. In this, the pair of hash values are combined to produce the single hash value at the end, forming a hierarchy of hash values. It is similar to Combined Hashing, as the aim of both patterns is to produce a single hash value at the end. However, a hierarchical hashing pattern is better than a combined hashing pattern because, in a hierarchical hashing pattern, the two hash values of the fixed and same length are combined as opposed to the combined hashing where two independent arbitrary size data are combined.

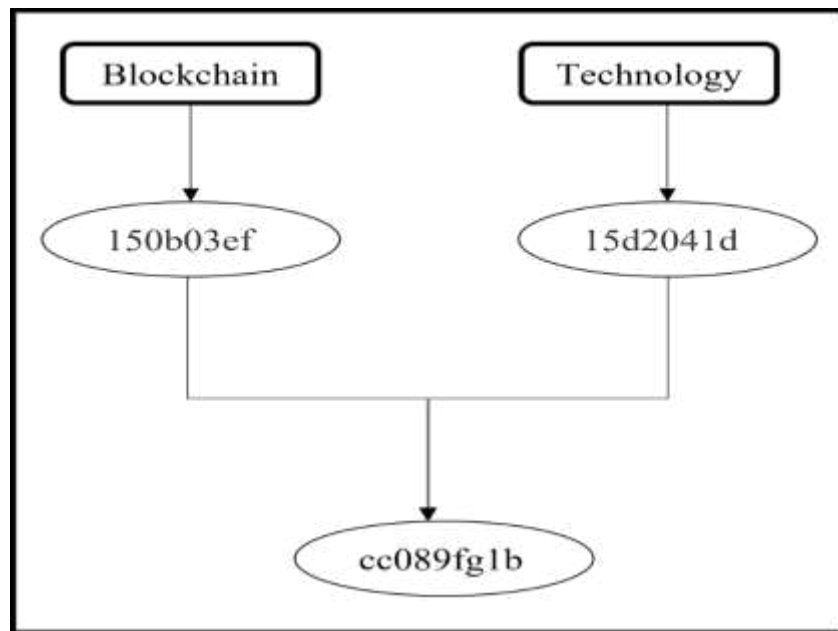


Figure 32: An example of Hierarchical Hashing Pattern

4.5.3 Hashing in Blockchain

As we have discussed earlier, the hashing algorithm is a one-way function, which makes it impossible to solve the hash puzzle using the reverse engineering method. In addition, it is quick,

deterministic, and safeguards data integrity. Consequently, hashing is regarded as the technology's foundation and has numerous applications in blockchain technology.

- [Proof-of-Work Consensus Mechanism](#)

In the Proof-of-work consensus mechanism, a node is required to find the correct hash value of the block to become a miner. Since the hashing function is a one-way function, it is impossible to track the input from the output. The node needs to try all the possible values one by one until it finds the correct hash value. The process consumes a lot of computational time and power, and no malicious node can solve the hash puzzle quickly using any trick. Hence, hashing is an important concept in a proof-of-work consensus algorithm. The complete concept is explained in detail in the consensus models section of the report.

- [Linking of Blocks in a Blockchain](#)

In the blockchain, all blocks have a hash value calculated during the consensus, as discussed in the consensus model section of the report. These hash values are the unique identity of the blocks in the blockchain network. Along with its own hash value, every block also contains the hash value of a block added to a ledger just before it. This is how the blocks get connected to the previous block forming a chain of blocks. Moreover, it also provides security to the network. Due to the characteristic of a hashing function, changing

anything in a block will make the block's current hash value invalid, thus causing every block in the chain to be invalid.

- [Merkle tree](#)

Merkle tree summarizes all the transactions in each blockchain block to form one digital fingerprint or hash value, providing a very efficient process to verify whether a transaction is included in a block. They take advantage of hashing operations to make it impossible to locate two Merkle trees with the same root hash. In this manner, the integrity of the transactions included within the block's body is also safeguarded, along with maintaining the integrity of the block header. The complete concept is explained in detail in the above section of the report.

- [Digital Signatures](#)

Digital Signatures are used in the blockchain to maintain the integrity of the message. The message is first hashed to summarize the data to a compact value while preserving integrity [57]. After the message is hashed, it is encrypted with the sender's private key to generate a signature.

4.5.4 SHA-256

Secure Hashing Algorithm, also known as SHA, was developed by the American National Institute for Standards and Technology and published as a FIPS standard in 1993 [58]. SHA is a modified version of MD5. SHA 0, a 160-bit hash function, was the initial iteration of the SHA algorithm. Later, in 1994, it underwent modification and was released as SHA-1. Following this, a new version of SHA known as SHA-2 was released. This hash function is more secure than SHA-1 and contains a variety of versions, including SHA-224, SHA-256, SHA-384, SHA-512, etc. Another iteration of SHA, dubbed SHA-3, was released in 2012.

Out of all the hash functions mentioned above, the most eminent is SHA-256 which is a variant of the SHA-2 algorithm. Blockchain technology uses the SHA-256 algorithm, which was created by the National Security Agency in 2001. The MD(Merkle-Damgard)-construction and Davis-Meyer mode are used to build it. The number 256 in the SHA-256 refers to the hash value's size. No matter how large the input message is, SHA-256 generates a hash value of 256 bits.

Design of SHA-256 algorithm [12] [59]

The designing of the SHA-256 algorithm is divided into two sections Pre-processing and Hash Computation.

1. Pre-processing:

1. Padding some extra bits to the input message to make the total length of the block 512 bits.
While padding, the first bit is one, and the rest are zeroes.
2. After this, the complete message (message + padding) is divided into blocks of size 512 bits each.
3. After this, the eight buffers are initialized with the default value. It consists of the eight 32-bit words obtained by taking the first 32-bits of the fractional parts of the square roots of the first eight prime numbers. These initial values are randomly chosen to initialize the process and provide confidence that no backdoor exists in the algorithm.

2. Hash computation:

1. Each message block is processed in a sequence, requiring 64 rounds to compute the full hash output. Each round uses slightly different constants to ensure that no two rounds are the same.
2. The message schedule is prepared.
3. Eight working variables are initialized.
4. The intermediate hash value is calculated.
5. Finally, the message is processed, and the output hash is produced:

In the below figure *a, b, c, d, e, f, g, and h* are the eight registers. *W_j* and *K_j* are the round constants and Σ_0 and Σ_1 performs bitwise rotation,

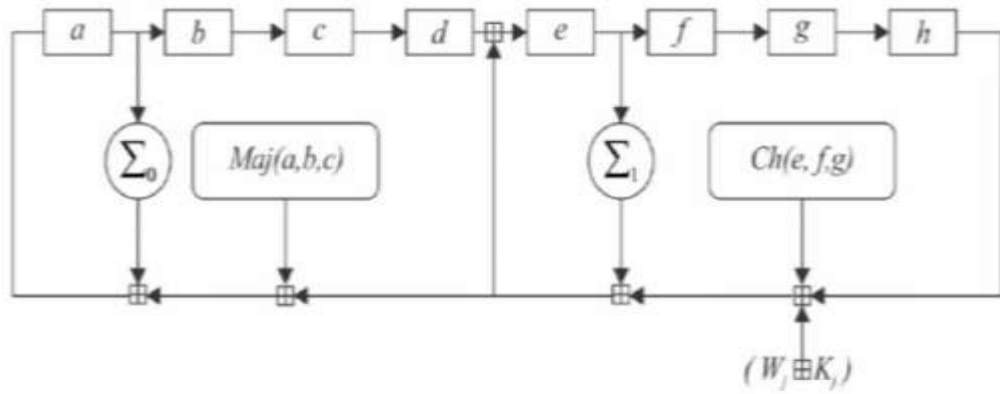


Figure 33: One round of a SHA compression function [60]

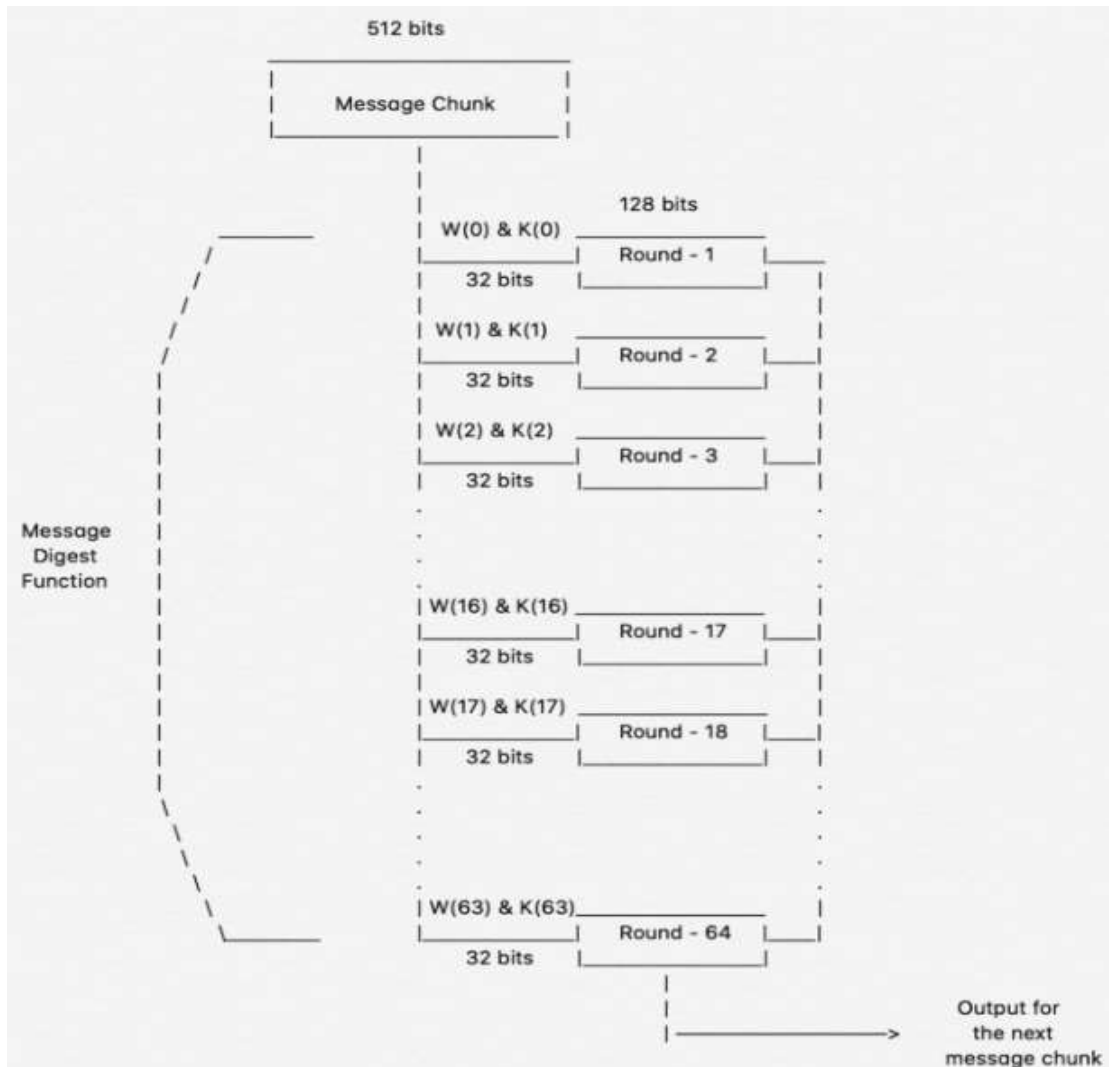


Figure 34: Complete process of 64 rounds of a SHA Compression function [61]

It can be seen in *figure 34* that the entire message is divided into blocks of 512 bits each, and each block goes through 64 rounds of operation. The output of one block is fed an input of the next block. This process continues until we reach the last block, the output of which is the final hash value, 256 bits in length.

5. Attacks on Blockchain Network

Blockchain is among the most secure technologies. However, nothing in this world is entirely secure. Every application is susceptible to some sort of attack, and the blockchain is no exception. It is challenging to attack the blockchain network because of its security paradigm. However, there are certain security risks that have the potential to do a great deal of harm, and recent times have seen a lot of attacks on various blockchain networks.

Here are some most common attacks that have challenged the security paradigm of blockchain technology.

5.1 51% Attack

In a blockchain network, every node has a copy of the ledger and to add a transaction, every node validates the transaction. After the validation from the maximum number of nodes (51% or more), the transaction is added to the network, and every node gets the updated copy of the ledger. More than 51% of the nodes must validate any transaction to add, delete, or alter it. However, if any participant in the network is able to get control of the majority of the network (51% of the network or more), then it can manipulate the block creation, add, delete, or alter the transactions. This type of attack is called a 51% attack or majority attack.

The majority gives the bad actor power to double-spend the currency, block new transactions, prevent other miners from mining the valid blocks, etc. However, carrying out a 51% attack is

exceedingly difficult since it costs a lot of money and resources and requires a lot of computing power to manage the vast majority of the network. Due to this, the profits gained by the attacker are usually significantly less than the expense of maintaining the majority of the network. The expense increases with the size of the network. Therefore, as the number of nodes in the network grows, the likelihood of a 51% attack decreases. So, anyone with this much mining power will therefore make more profit from utilizing it to mine the block legitimately than to attack the network.

There are a few 51% attacks that have taken place in the recent past on different blockchain networks. In 2018, more than \$18 million were double spent in a 51% attack on Bitcoin Gold (BTG). Another such attack on Ethereum Classic (ETC) in January 2019 resulted in over \$1 million of currency being double spent by an attacker. In 2018, the Vertcoin (VTC) network was attacked four times, resulting in double spending of over \$100,000 VTC. It is important to note that most of the 51% attacks that have taken place were on small blockchain networks.

5.2 Eclipse Attack

The blockchain is a distributed ledger in which one node is not connected to all the other nodes in the network. Instead, all the nodes are connected to their neighbouring nodes. The nodes always seek its ledger's view from those nodes connected to it. In an eclipse attack, the attacker tries to infect one node by creating an artificial environment. An attacker surrounds the node with the attacker-controlled nodes, thus secluding the node from the actual blockchain network. The

attacker can now present the manipulated version of the network and make that node unable to see the true ledger, thus eclipsing.

By isolating the node from the legitimate state of the blockchain network, the attacker can confirm the illegitimate transactions, which leads to fake spending or double-spending. The eclipse attack can also cause mining power disruption. Let us understand an eclipse attack with the help of an example. Suppose the infected node is a seller who sells mobile phones. The attacker will buy a phone from him and make a fake transaction. This fake transaction will be confirmed by the attacker-controlled nodes surrounding the victim node, and the seller will give the phone to the attacker.

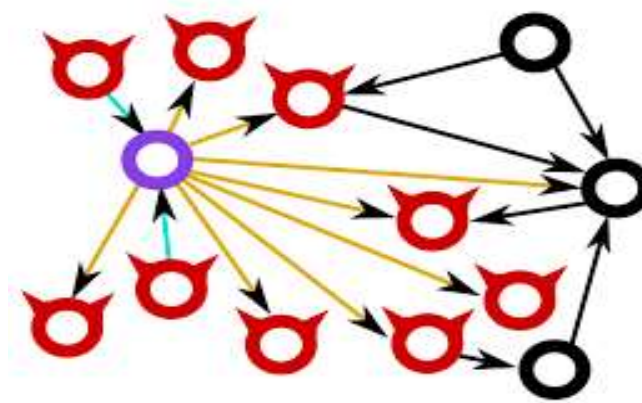


Figure 35: Eclipse Attack [62]

One way to mitigate the eclipse attack is by blocking/restricting the inbound connections and making outbound connections only with the trusted nodes. However, this is how it will become difficult for the new nodes to join the blockchain network. Therefore, the better approach to

mitigate the eclipse attack is 'Random Node Selection'. In this approach each node connects with a random and different set of IP addresses (nodes) every time it syncs with the blockchain network.

5.3 DDOS Attack

A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt the normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic [63]. The attacker repeatedly sends random traffic that the server cannot handle through different compromised sources, which causes a network failure. However, due to its decentralized nature, blockchain technology is protected from single-point failure. Any node in the network can go offline due to a DDoS attack or because of any other reason without causing harm to the entire network, and the transactions can continue. However, blockchain technology is not wholly resilient to DDoS attack.

In blockchain networks, the DDoS attack is performed via the transaction flooding method. The number of blocks that may be created and added to the network is limited for all blockchains due to their fixed, predefined capacities. As a result, the total number of transactions that can be recorded is limited. In the transaction flooding method, the attacker floods the false and spam transactions from multiple sources, filling the block. Therefore, the legitimate transaction(s) will not get added to the block. The transaction(s) which does not get added to the current block is kept in the memory called Mempool. Since the attacker will keep sending the spam transactions until the capacity of the blockchain is reached. As a result, the legitimate transaction(s) will not get

added to the block, resulting long waiting time for the users for their legitimate transactions to get verified and added to the ledger.

The DDoS attack can cause mass destruction to any organization. Only one hour of downtime can cause an organization to lose thousands of dollars and can harm its reputation. An attack on the Solana network is one of the most recent and well-known examples of a DDoS attack on a blockchain network. It happened in 2021, not long after it was launched. The attacker flooded the network with approximately 400,000 transactions per second, causing the memory to fill up, and the Solana network crashed. It was down for several hours before getting fixed by performing the hard fork. Another such DDoS attack was performed in February 2021 on the EXMO Cryptocurrency exchange causing the network to be down for around 5 hours. It is important to note that attackers attempt to launch thousands of DDoS attacks on the system, website, blockchain network, etc., each day. As a result, having a DDoS attack prevention strategy is critical.

Now, we will try implementing Distributed Denial of Service attack on the blockchain network.

5.3.1 Implementation

The goal is to build a blockchain network and attack it with a Distributed Denial of Service attack. To begin, we have developed a client-server model and are assuming that it will function on a blockchain network. We have assumed that server here is RB Crypto Coins network which uses blockchain technology and has a number of users. The transactions done by the users are recorded over a blockchain network.

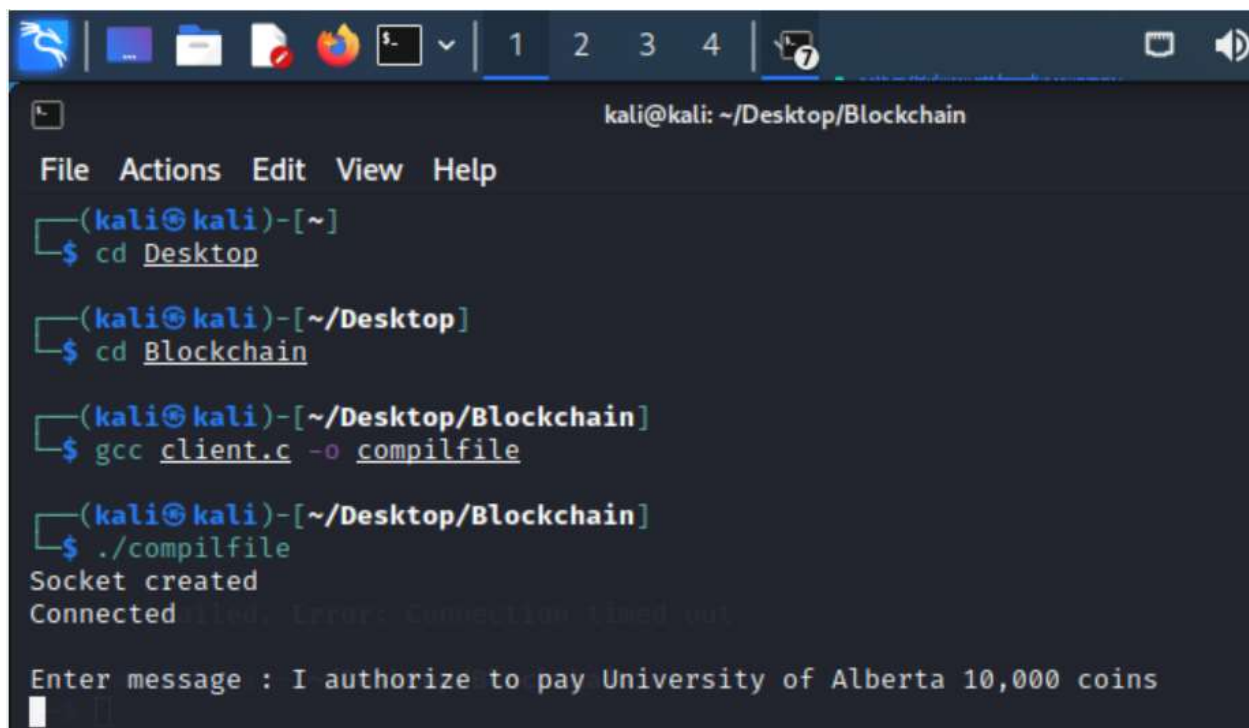
```
~/Desktop/Blockchain/server.c - Mousepad
File Edit Search View Document Help
1 #include<stdio.h>
2 #include<string.h> //strlen
3 #include<sys/socket.h>
4 #include<arpa/inet.h> //inet_addr
5 #include<unistd.h> //write
6 #include<stdlib.h>
7
8
9 int main(int argc , char *argv[])
10 {
11     int socket_desc , client_sock , c , read_size;
12     struct sockaddr_in server , client;
13     char client_message[2000];
14
15
16     //Create socket
17     socket_desc = socket(AF_INET , SOCK_STREAM , 0);
18     if (socket_desc == -1)
19     {
20         printf("Could not create socket");
21     }
22     puts("Socket created");
23
24     //Prepare the sockaddr_in structure
25     server.sin_family = AF_INET;
26     server.sin_addr.s_addr = INADDR_ANY;
27     server.sin_port = htons( 24459 );
28
29     //Bind
30     if( bind(socket_desc,(struct sockaddr *)&server , sizeof(server)) < 0)
31     {
32         //print the error message
33         perror("bind failed. Error");
34         return 1;
35     }
36     puts("bind done");
37     printf("%d", sizeof(server));
38     //Listen
39     listen(socket_desc , 3);
40
41     //Accept and incoming connection
42     puts("Waiting for incoming connections...");
43     c = sizeof(struct sockaddr_in);
44
45     //accept connection from an incoming client
46     client_sock = accept(socket_desc, (struct sockaddr *)&client, (socklen_t*)&c);
47     if (client_sock < 0)
48     {
49         perror("accept failed");
50         return 1;
51     }
52     puts("Connection accepted");
53
54     //Receive a message from client
55     while( (read_size = recv(client_sock , client_message , 2000 , 0)) > 0 )
56     {
57         //Send the message back to client
58         write(client_sock , client_message , strlen(client_message));
59
60         if(read_size == 0)
61         {
62             puts("Client disconnected");
63             fflush(stdin);
64         }
65         else if(read_size == -1)
66         {
67             perror("recv failed");
68         }
69         printf("%d",read_size);
70
71         return 0;
72     }
73 }
```

Figure 36: Source Code of a Server


```
~/Desktop/Blockchain/client.c - Mousepad
File Edit Search View Document Help
server.c client.c
1 #include <stdio.h> //printf
2 #include <string.h> //strlen
3 #include <sys/socket.h> //socket
4 #include <arpa/inet.h> //inet_addr
5 #include <unistd.h>
6
7 int main(int argc , char *argv[])
8 {
9     int sock;
10    struct sockaddr_in server;
11    char message[1000] , server_reply[2000];
12
13    //Create socket
14    sock = socket(AF_INET , SOCK_STREAM , 0);
15    if (sock == -1)
16    {
17        printf("Could not create socket");
18    }
19    puts("Socket created");
20
21    server.sin_addr.s_addr = inet_addr("127.0.0.1");
22    server.sin_family = AF_INET;
23    server.sin_port = htons( 24459 );
24
25    //Connect to remote server
26    if (connect(sock , (struct sockaddr *)&server , sizeof(server)) < 0)
27    {
28        perror("connect failed. Error");
29        return 1;
30    }
31
32    puts("Connected\n");
33
34    //keep communicating with server
35    while(1)
36    { printf("Enter message : ");
37      scanf("%s" , message);
38
39      //Send some data
40      if( send(sock , message , strlen(message) , 0) < 0)
41      {
42          puts("Send failed");
43          return 1;
44      }
45
46      //Receive a reply from the server
47      if( recv(sock , server_reply , 2000 , 0) < 0)
48      {
49          puts("recv failed");
50          break;
51      }
52
53      puts("Server reply :");
54      puts(server_reply);
55    }
56
57    close(sock);
58    return 0;
59 }
```

Figure 37: Source Code of a Client

A transaction made by a legitimate user will be confirmed in accordance with the consensus model in use and stored in the block over a blockchain network. However, as stated above, every blockchain has a limited capacity; as a result, the total number of transactions that can be recorded is limited. In our example, the network can handle 100,000 transactions per second. The below figure demonstrates that a legitimate user got connected to the blockchain network and sent a transaction. As per the consensus protocol, this transaction was verified and included to the block.

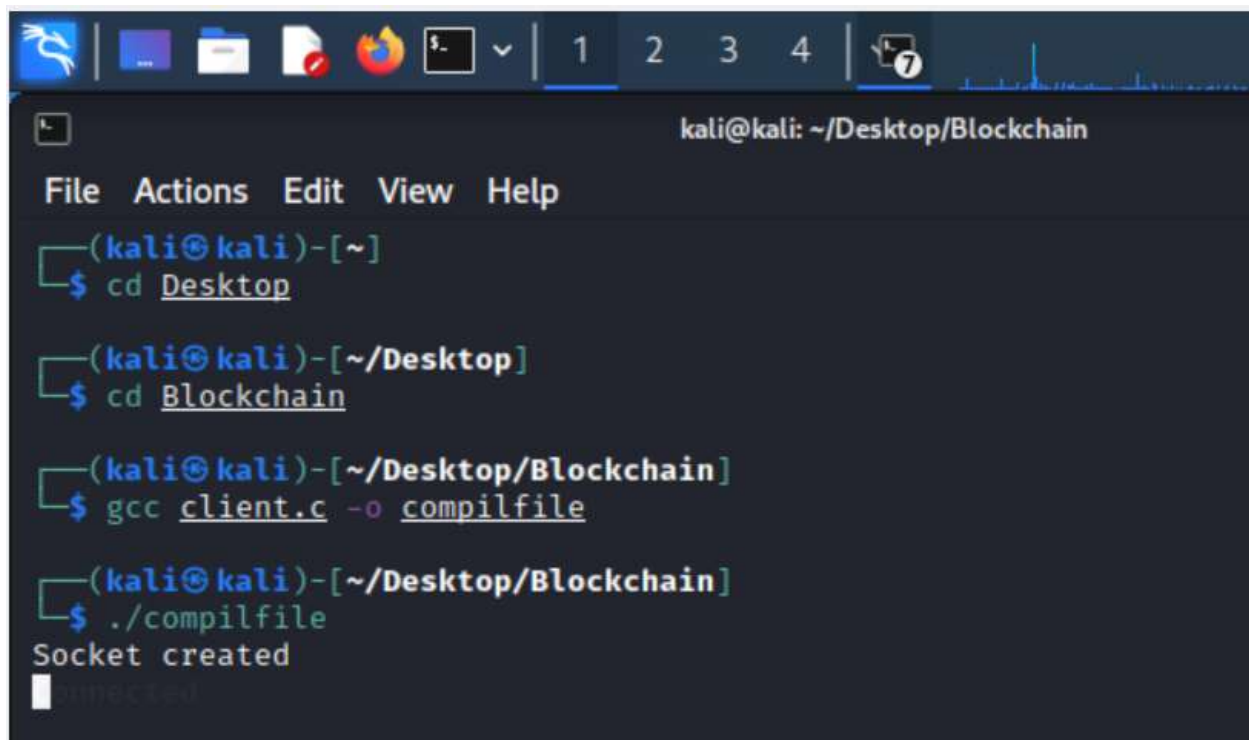


```
kali@kali: ~/Desktop/Blockchain
File Actions Edit View Help
(kali@kali)-[~]
└─$ cd Desktop
(kali@kali)-[~/Desktop]
└─$ cd Blockchain
(kali@kali)-[~/Desktop/Blockchain]
└─$ gcc client.c -o compilfile
(kali@kali)-[~/Desktop/Blockchain]
└─$ ./compilfile
Socket created
Connected
Enter message : I authorize to pay University of Alberta 10,000 coins
```

Figure 38: Legitimate user sends a transaction to a blockchain network

Now, we will join the RB Crypto Coins network as a malicious user and send spam and false transactions from a different source simultaneously. We will employ automated methods to flood the RB Crypto Coins blockchain network with tens of thousands of transactions per second for this purpose. The network tries to process these transactions, but because they are fake and spam, they

only clog up the memory and slow down the network. Because of this, the genuine transactions fail to be authenticated and put to the block, which ultimately brings down the blockchain network. A legitimate user attempting to make a legitimate transaction after the system has crashed won't be able to connect to the network. The figure below shows that the legitimate user cannot connect with the RB Crypto Coins network due to our Denial-of-Service attack on the network, which caused the network to crash.



```
kali@kali: ~/Desktop/Blockchain
File Actions Edit View Help
(kali@kali)-[~]
└─$ cd Desktop
(kali@kali)-[~/Desktop]
└─$ cd Blockchain
(kali@kali)-[~/Desktop/Blockchain]
└─$ gcc client.c -o compilfile
(kali@kali)-[~/Desktop/Blockchain]
└─$ ./compilfile
Socket created
[ ]
```

Figure 39: Legitimate user not able to connect after the network crash

5.3.2 Mitigation Measures

DDoS attacks are seen as "weapons of mass devastation" on the Internet [64]. DDoS attacks are increasingly challenging to protect against because they happen hundreds of times every day. Although modern security technologies can defend against most types of DoS assaults, a DDoS attack is still considered a significant attack on the Internet. In the blockchain as well, despite the fact that blockchain is innately resistant to many DDoS techniques, transaction flooding is still a big concern. Therefore, building preventive measures to protect against DDoS attacks is very important.

Firstly, it is important that all the nodes must have enough storage, bandwidth, and processing power for the network. The primary way to attack a blockchain is by transaction flooding; therefore, the nodes' storage, bandwidth and processing power play a vital part in protecting the blockchain against the DDoS attack. Moreover, the more decentralized the blockchain network is, the more secure it is against the Distributed Denial of Service attacks.

Another measure to protect the blockchain against DDoS attacks is rate limiting. That limits the number of transactions that can be processed at a particular instance. Another critical factor is to build a failsafe into the code of the smart contract to prevent the blockchain network from crashing instantly.

Along with all these, it is important to have a system to filter the transactions to clear the buffer space in order to avoid network congestion. Moreover, filtering potential spam or non-legitimate

transactions maintains the integrity of the blockchain. Following all these measures will help mitigating the Distributed Denial of Service attack on blockchain technology.

6. Conclusion

The world is going to be changed by blockchains. It is extremely important and is used in many facets of human life today. The future of the financial industry and secure information sharing is blockchain technology. It will enhance the various sectors by providing transparent information instantly, better security, better risk mitigation, fast processing, easier auditability, cost reduction, automation, and opportunities. As a result, numerous organizations are working to investigate the full potential of the technology in various industries.

In this project, we studied blockchain technology and its importance in detail. We have studied the technology's features, security mechanisms, and vulnerabilities. We have covered the different technical aspects of technology in four parts.

In the first part, we studied the underlying concepts, blockchain architecture, types, features, and the main components of the technology. We have tried to understand technology's fundamentals and its importance in today's world.

In the second part, we studied the different consensus protocols used in different types of blockchain, which play an important role in the security mechanism of blockchain technology.

In the third part, we studied asymmetric cryptographic techniques, including digital signatures and hashing, which control the security of the blockchain network.

In the final part, we have studied the different attacks that are still possible on blockchain technology, even after having high security, due to some vulnerabilities. We implemented the Distributed Denial of Service attack on the blockchain network and provided the countermeasures.

7. References

- [1] I. M. Abdelwahed, N. Ramadan and H. A. Hefny, "Cybersecurity Risks of Blockchain Technology," *International Journal of Computer Applications*, p. 7, 2020.
- [2] Xavier, "Simply Explained," [Online]. Available: https://www.youtube.com/watch?v=SSo_EIwHSd4.
- [3] [Online]. Available: <https://en.wikipedia.org/wiki/Blockchain>.
- [4] "Shutterstock," [Online]. Available: <https://www.shutterstock.com/search/decentralized-network>.
- [5] IBM, [Online]. Available: <https://www.ibm.com/topics/what-is-blockchain>.
- [6] "The World Bank," [Online]. Available: <https://documents1.worldbank.org/curated/pt/134831513333483951/pdf/WP-PUBLIC-Distributed-Ledger-Technology-and-Blockchain-Fintech-Notes.pdf>.
- [7] S. S. Sarmah, "Understanding Blockchain Technology," *Academia*, vol. 8, no. 2, pp. 23-29, 2018.
- [8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," pp. 1-9, 2008.
- [9] K. Raj, *Foundations of Blockchain: The Pathway to Cryptocurrencies and Decentralized Blockchain Applications*, Birmingham: Packt Publishing Ltd., 2019.
- [10] D. Geroni, "101 Blockchains: Blockchain Nodes," 05 April 2021. [Online]. Available: <https://101blockchains.com/blockchain-nodes/>.
- [11] L. Mazzucchelli, M. Bowman, R. Chandrasekharan, C. Reynolds and C. Freeberg, "EMERGING TECHNOLOGIES FOR FUTURE SKYIES: BLOCKCHAIN," CANSO Strategic Technology Workgroup, 2021.
- [12] I. Bashir, *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained*, 2nd Edition, Birmingham: Packt Publishing, Limited, 2018, p. 647.
- [13] V. Acharya, N. P. and A. E. Yerrapati, *Layered structure of the blockchain architecture*, Packt, 2019, p. 350.
- [14] K. Pandey, "JUMPSTART," 19 September 2022. [Online]. Available: <https://www.jumpstartmag.com/what-are-the-different-layers-of-blockchain-technology/>.
- [15] F. D. Currency, "Financial Express," 08 August 2022. [Online]. Available: <https://www.financialexpress.com/blockchain/the-layered-structure-of-the-blockchain-architecture/2621510/>.

- [16] S. S. Dang, "Forbes," 24 October 2022. [Online]. Available: <https://www.forbes.com/sites/sanjitsinghdang/2022/10/24/understanding-the-blockchain-layers-to-solve-the-scalability-challenges/?sh=57c50c6716e2>.
- [17] "cointelegraph," [Online]. Available: cointelegraph.com.
- [18] "Research Gate," [Online]. Available: https://www.researchgate.net/figure/Blockchain-block-structure_fig1_325136332.
- [19] R. G. IV, "https://robertgreenfieldiv.medium.com/," 20 July 2017. [Online]. Available: <https://robertgreenfieldiv.medium.com/explaining-proof-of-stake-flae6feb26f>.
- [20] A. M. Antonopoulos, *Mastering Bitcoin- Unlocking Digital Cryptocurrency*, O'Reilly Media, Inc., 2014, p. 298.
- [21] V. Tabora. [Online]. Available: <https://www.datadriveninvestor.com/2019/11/21/a-decomposition-of-the-bitcoin-block-header/>.
- [22] Z. Zheng, S. Xie, H. Dai, X. Chen and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," *2017 IEEE International Congress on Big Data (BigData Congress)*, pp. 557-564, 2017.
- [23] A. Kumar, "Geeks for Geeks," 11 May 2022. [Online]. Available: <https://www.geeksforgeeks.org/blockchain-and-block-header/>.
- [24] J. Frankenfield, "Investopedia," 22 September 2021. [Online]. Available: <https://www.investopedia.com/terms/b/block-header-cryptocurrency.asp#:~:text=Block%20headers%20identify%20individual%20blocks,metadata%20and%20multiple%20individual%20components..>
- [25] T. Brew, "Medium- Merkle Tree in Blockchain," [Online]. Available: <https://medium.com/techskill-brew/merkle-tree-in-blockchain-part-5-blockchain-basics-4e25b61179a2>.
- [26] S. S. Reddy, 18 May 2020. [Online]. Available: <https://www.forex.academy/understanding-merkle-tree-its-importance-in-blockchain/>.
- [27] V. Mathur, "Analytics Steps: What is Merkle Tree in Blockchain?," 16 November 2022. [Online]. Available: <https://www.analyticssteps.com/blogs/what-merkle-tree-blockchain>.
- [28] B. Becher, "Builtin," 29 September 2022. [Online]. Available: <https://builtin.com/blockchain/blockchain-node>.
- [29] J. Evans, "NODES.COM," [Online]. Available: <https://nodes.com/>.
- [30] "Origin Stamp: The 10 Different Types of Blockchain Nodes and How They Work," [Online]. Available: <https://originstamp.com/blog/the-10-different-types-of-blockchain-nodes-and-how-they-work/>.

- [31] J. Frankenfield, "Investopedia: Hard Fork: What It Is in Blockchain, How It Works, Why It Happens," 25 May 2022. [Online]. Available: <https://www.investopedia.com/terms/h/hard-fork.asp>.
- [32] S. Dutta, "Level Up Coding : How Ethereum Reversed a \$50 Million DAO Attack!," 17 December 2019. [Online]. Available: <https://levelup.gitconnected.com/how-ethereum-reversed-a-50-million-dao-attack-cee528d8c030>.
- [33] "Binance Academy: Hards Forks and Soft Forks," 28 November 2018. [Online]. Available: <https://academy.binance.com/en/articles/hard-forks-and-soft-forks>.
- [34] V. Gupta, "Geeks for Geeks," [Online]. Available: <https://www.geeksforgeeks.org/types-of-blockchain/>.
- [35] T. K. Sharma, "Blockchain Council," 3 November 2022. [Online]. Available: <https://www.blockchain-council.org/blockchain/permissioned-and-permissionless-blockchains-a-comprehensive-guide/>.
- [36] "Oracle Technical Articles," Oracle, 18 August 2022. [Online]. Available: <https://developer.oracle.com/learn/technical-articles/permissioned-blockchain>.
- [37] JINGJING GU¹, BINGLIN SUN¹, XIAOJIANG DU², JUN WANG¹, YI ZHUANG¹, AND ZIWANG WANG¹, "Consortium Blockchain-Based Malware," *IEEE Access*, vol. 10, 13 February 2018.
- [38] H. F. Atlam, A. Alenezi, G. B. Wills and . M. O. Alassafi , "Blockchain with Internet of Things: Benefits, Challenges, and Future Directions," *I.J. Intelligent Systems and Applications*, vol. 10, no. 6, pp. 40-48, June 2018.
- [39] I. Bashir, *Blockchain Consensus*, Berkeley: Apress, 2022.
- [40] A. Baliga, "Understanding Blockchain," Persistent Systems, Pune.
- [41] "Crypto : Consensus Mechanisms in Blockchain: A Beginner's Guide," 23 May 2022. [Online]. Available: <https://crypto.com/university/consensus-mechanisms-in-blockchain>.
- [42] I. B. Narayan Prusty, *Advanced Blockchain Development*, Packt, 2019.
- [43] M. Antolin, "CoinDesk : What Is Proof-of-Authority?," 02 June 2022. [Online]. Available: <https://www.coindesk.com/learn/what-is-proof-of-authority/>.
- [44] J. Frankenfield, "Investopedia: Proof of Burn," 22 September 2021. [Online]. Available: [https://www.investopedia.com/terms/p/proof-burn-cryptocurrency.asp#:~:text=Proof%20of%20burn%20\(POB\)%20is,%E2%80%9Cburn%E2%80%9D%20virtual%20currency%20tokens..](https://www.investopedia.com/terms/p/proof-burn-cryptocurrency.asp#:~:text=Proof%20of%20burn%20(POB)%20is,%E2%80%9Cburn%E2%80%9D%20virtual%20currency%20tokens..)
- [45] J. H. Lee, "Systematic Approach to Analyzing Security and Vulnerabilities of Blockchain Systems," Massachusetts Institute of Technology, 2019.

- [46] "Fortinet: What is Cryptography," [Online]. Available: <https://www.fortinet.com/resources/cyberglossary/what-is-cryptography>.
- [47] J. Kothari, "Geeks for Geeks: Cryptography in Blockchain," 22 November 2022. [Online]. Available: <https://www.geeksforgeeks.org/cryptography-in-blockchain/>.
- [48] D. Geroni, "101 Blockchains: Public Key Cryptography in Blockchain," 05 May 2021. [Online]. Available: <https://101blockchains.com/public-key-cryptography-in-blockchain/>.
- [49] J. Andress, *The Basics of Information Security*, Syngress, 2014.
- [50] J. Lake, "Comparitech: What is RSA encryption and how does it work?," 18 March 2021. [Online]. Available: <https://www.comparitech.com/blog/information-security/rsa-encryption/>.
- [51] S. P. Oriyano, *Cryptography InfoSec Pro Guide*, McGraw-Hill Business, 2013, p. 336.
- [52] "University of New Mexico : Intro to Cryptography and Cryptocurrencies," [Online]. Available: <https://www.cs.unm.edu/~saia/classes/591-Blockchains-s19/lec/lec1-crypto.pdf>.
- [53] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Hash_collision.
- [54] D. Wong, *Real-World Cryptography*, Manning Publications, 2021.
- [55] M. Bertaccini, *Cryptography Algorithms*, Packt Publishing, 2022.
- [56] "Medium: Patterns of Hashing in Blockchain," [Online]. Available: <https://medium.com/coinmonks/patterns-of-hashing-in-blockchain-5319a1b9cba8>.
- [57] M. Zubair, "Educative: How is SHA-256 used in blockchain, and why?," [Online]. Available: <https://www.educative.io/answers/how-is-sha-256-used-in-blockchain-and-why>.
- [58] H. Yoshida and A. Biryukov, "Analysis of a SHA-256 Variant," *Selected Areas in Cryptography*, pp. 245-260, 2005.
- [59] B. K. Jena, "Simplilearn: A Definitive Guide to Learn The SHA-256," 11 November 2022. [Online]. Available: <https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm>.
- [60] "behind bit coin," 15 March 2015. [Online]. Available: <https://behindbitcoin.wordpress.com/2015/03/15/sha-256-hash-functions-a-process/>.
- [61] B. K. Jena, "Simpli Learn," [Online]. Available: <https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm>.
- [62] "Tumblr," [Online]. Available: <https://ethanheilman.tumblr.com/post/114413288445/abstract-we-present-eclipse-attacks-on-bitcoins>.
- [63] "Cloudflare: What is a DDoS Attack," [Online]. Available: <https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/>.

- [64] Qaiser, "Geeks for Geeks: DDoS in Blockchain," 16 August 2022. [Online]. Available: <https://www.geeksforgeeks.org/ddos-in-blockchain/>.

8. Glossary

Terms	Definition
Authorized Network Members	Participants in a blockchain network who have been granted access to view or interact with the network.
Advance Encryption Standard (AES)	A symmetric-key cryptography algorithm (block cipher) developed by the National Institute of Standards and Technology of United States.
Bad Actors	An individual or an organization that that attempts to illicitly hack into systems, networks, or information and try to manipulate it.
Client Node	A node in a network that relies on a server node for all services.
CORDA	An open-source permissioned or private blockchain platform developed for enterprise use cases.
Cipher-text	The data or message that is encrypted after being converted from plain text using an encryption algorithm.
Double spending	To spent digital currency more than once in different transactions.
Distributed Ledger	A network of nodes that manages and shares a digital database. Without the need for a centralized authority, it provides a means for multiple parties to have a consistent view of the data.
Decentralized Applications	Applications that run on a blockchain network and do not store or process data from a central server or authority.

DES	Data Encryption Standard, a symmetric-key cryptography algorithm
Hyperledger	The Linux Foundation launched an open-source platform in December 2015 for building applications based on the blockchain.
Hashcash	Hashcash is a proof-of-work method that has been applied in a number of systems as a denial-of-service defence mechanism and to prevent email spam.
Miner	A network node that is selected to validate the new block and adding it to the blockchain network and in return get paid for it.
Mining	The process of performing proof-of-work to add a block to the blockchain network.
Malicious Node	Aa node in the blockchain network that violates consensus rules or launches attacks to attempt to disrupt the network's operation.
Man-in-the-middle attack	An attack where an attacker sits in between the two parties. It either manipulates the data shared by the one party to another or impersonate as one of the party and exchange information
Message Digest	A hash function used for data integrity and authentication.
Non-repudiation	The ability to prevent someone from denying that they have created or sent a message or document.
Peer-to-peer	A decentralized system in which users communicate with one another directly, without the aid of middlemen or centralised authorities.
Protocols	Protocols are the set of rules that govern a network.
Plain-Text	The original message or data before encryption.

Pseudorandom	A sequence of numbers that appears to be random but is actually produced by a deterministic algorithm.
Quorum	An enterprise-grade blockchain platform developed by JPMorgan Chase
Routing Attacks	A type of attack on a blockchain network where an attacker partitions the network into two or more preventing nodes in different sections to communicate with each other.
Smart Contracts	Smart contract is used to automate the process which needs to be completed by the parties and therefore eliminates the need of both parties trusting each other.
Server Node	A node in a network that serves as the central authority for all services in a network.
Timestamp	The process of associating a date and time with a digital document to prove that it existed at a particular time.