Blockchain Technology Overview: Architecture, Characteristic, relevant Attacks, and Applications

Fahad M. Senan, Firdous Kausar

Department of Electrical and Computer Engineering, College of Engineering,
Sultan Qaboos University, Muscat, Oman.

Abstract
The Blockchain-based applications, nowadays, has gotten insanely popular. It is a race out there; everyone is trying to build his application on top of the blockchain. Certainly, that is due to its unique characteristics and design. However, such characteristics are not well suited to all applications of blockchain technology. Not merely alteration of the blockchain's protocols would make it suitable, but rather redesigning its architecture is required to make it suitable for different applications. This paper describes the overview of blockchain technology in a very comprehensive manner, summarizes most of the popular attacks that surfaced against such a technology and the vulnerabilities exploited by them, which leads to design a better version of the blockchain that is well suited to the application in question.

Keywords:
blockchain; bitcoin; security, immutability, attacks; proof of work;

1. Introduction

Nowadays, blockchain technology has gotten insanely popular. Although it serves as the basis of the famous cryptocurrency – that is the "Bitcoin" which was created by Satoshi Nakamoto [1] in 2008 and although he or she didn't refer to the blockchain by the name per se, the blockchain idea goes far before when it was originally described as a technique to timestamp digital documents [2] in 1991. Ever since, it has been a race on innovating applications based on it, such as smart contracts, communication systems, healthcare systems, censorship resistance systems, electronic voting, national ID systems, and of course, cryptocurrencies hold the lion's share [3].

Such a rapid evolvement has developed a battle of wits between the applications' innovators and the adversaries. The applications' innovators are trying to build systems that meet the business requirements and protocols that assure the security of the made transactions. While the adversaries are trying to exploit the weaknesses of the blockchain architecture or its protocols and impose security risks that could cause the loss of faith in such systems, in other words, it harms its reputation, the loss of capital involved, or simply it could cause devaluation.

Over the past years, serious incidents have been recorded against Blockchain-based systems. For instance, in Jun. 2016, an attacker managed to drain $50 million from the decentralized autonomous organization (DAO) that operates on Blockchain-based smart contracts. In Jun. 2017, Bitfinex experienced a DDoS attack that led to its temporary suspension. In May, Aug., and Nov. 2017, memory pools of Bitcoin were flooded with dust transactions to create stale, delay in transactions verifications, and increase in Bitcoin mining fee. The reason for such an attack was to force the Bitcoin users to move to other cryptocurrencies with faster transaction processing time. In May and Jun. 2018, five Blockchain-based cryptocurrencies, namely, Monacoin, Bitcoin Gold, Zencash, Verge, and Litecoin Cash were targeted by the 51% attack, which performed double spending on valuable transactions, which led to a loss of $5 million [3]. Such incidents, consequently, led to the investment of serious thoughts and a fair amount of studies in the field of the blockchain and its applications.

Nevertheless, understanding the threats that arise from the weaknesses of the blockchain architecture or its protocols, understanding the blockchain architecture itself and how exactly its protocols are functioning, and understanding the recorded attacks against it and how did they exactly exploit those weaknesses, would help in a better realization of the applications built based on it.

Most of the research, however, has not gone beyond the blockchain's characteristics or its architecture and propose solutions that would overcome some of the threats or recorded attacks, or even some of the limitations or problems that would prevent from meeting the users' business requirements. Rather, the proposed solutions were targeting and enhancing the protocols employed by the blockchain system. Perhaps, that is due to the fact, the blockchain system is not fairly simple, its intelligence and interrelated concepts are hard to grasp or untangled, or as if, the blockchain system has set the framework that makes every application based on it is bounded by it. Anyhow, going forward without, at least, considering the redesigning of the blockchain system to furtherly tune its characteristics, would make blockchain potential application's space narrowed. Great efforts and extraordinarily complex protocols, otherwise, needed in case of realizing an application out of that space.

This paper aims to describe the blockchain system's architecture, its protocols, and its characteristics in detail,

Manuscript received January 5, 2025
Manuscript revised January 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.1.23

and explain the blockchain's surfaced common attacks, the weaknesses they have exploited, and the reasons behind it. Collectively, it would serve in a better understanding of such a technology, and the easiest tweaks that are needed to realize a new application.

The section-2 progressively explains the building blocks of the blockchain, its protocols, and characteristics. The third section gives a conceptual summary of the most common attacks arisen against blockchain technology. The fourth section nonetheless, concludes the paper by providing opening directions for the enhancements that could take place and thoughts that could be considered.

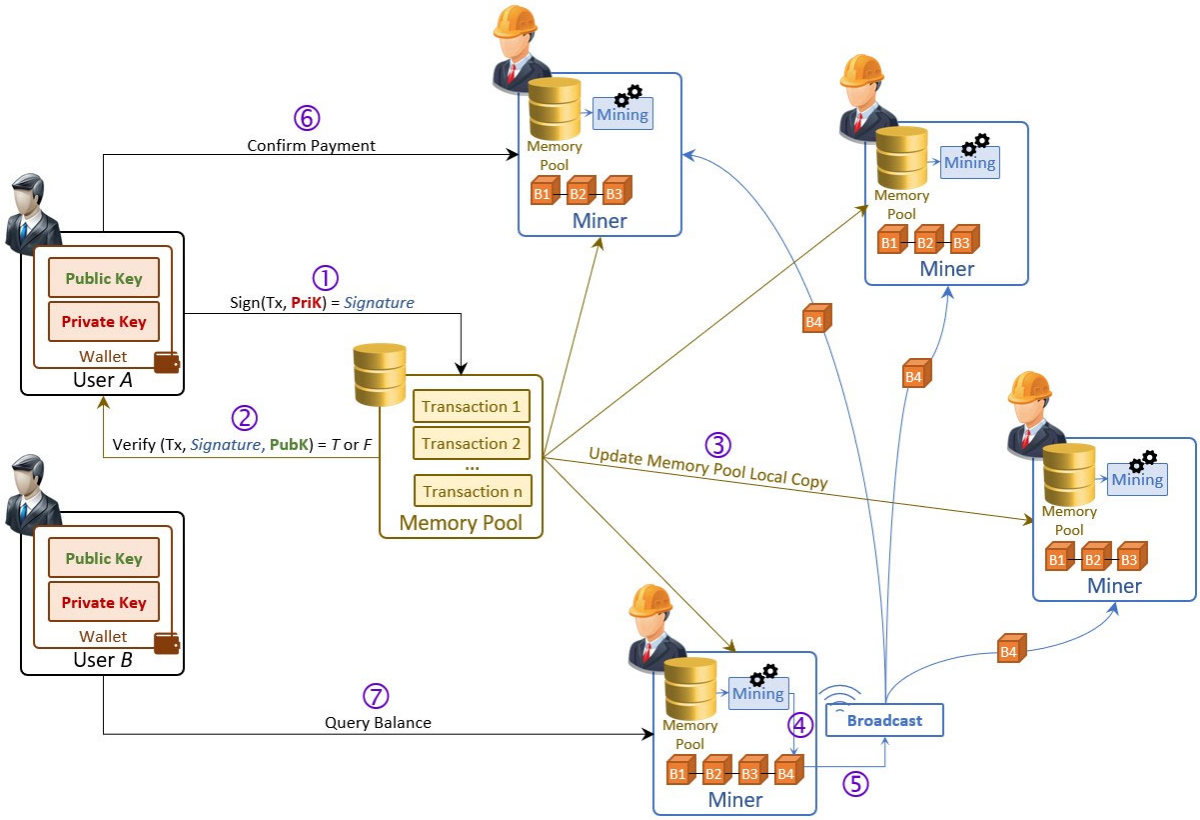

Figure 1: Transaction (Tx) Lifecycle

2. Blockchain Ecosystem

A Blockchain-based application is a peer-to-peer network architecture. Whether it is a cryptocurrency application, e.g., Bitcoin, Ethereum, Litecoin, or any other application, e.g., healthcare, supply-chain management, e-voting, national citizen ID, its ecosystem is constituted of two main things: architecture and a set of protocols, by which it demonstrates a set of characteristics.

Most of the cryptocurrency's applications, however, share almost a remarkably similar ecosystem, with differences that reside in some of the protocols, i.e., the consensus protocols. In this chapter, the document will focus on the Bitcoin ecosystem, which is considered one of the origins of the decentralized digital currency, if not the first, but the most associated with the blockchain. Moreover, understanding the Bitcoin ecosystem would, without doubt, set the firm basis for understanding any other existing Blockchain-based ecosystem, let alone innovating new Blockchain-based applications. Nevertheless, in some cases, it may point out those differences for verbosity.

2.1. Blockchain Architecture

Blockchain architecture, in this document context, refers to players/participants and the components that build the peer-to-peer network. Figure 1 illustrates the journey of the transaction through the various participants and components. This section lists those participants, i.e., user or miner, and components i.e., node, wallet, transaction, memory pool, block, and define their roles and responsibilities [4] [5].

Node: It is a computer that connects to the blockchain network. It can be the user's computer or the miner's computer, or any other computer that plays a role in the network. A node can be furtherly classed into two types: full node and SPV node [6].

Full Node: (aka Thick Client) is a node that downloads and validates the full chain of the blocks, starting from the genesis block to the most recently mined block. Although, with block file pruning, it may discard older parts of the chain to clear up disk space. A full node can validate transactions, relay blocks, or transactions to other nodes and mine new block.
Simplified Payment Verification (SPV): (aka. Thin Client) is a node that only downloads the headers of the blocks and validates the blocks and transactions by downloading the transactions as needed from the full nodes but does not mine new blocks.

Wallet: It is software that resides on the user's computer. Its purpose is to enable the user to send and receive bitcoins, or satoshis (the smallest unit of the bitcoins). A wallet generates and stores a mathematically related key pair, a public key (PubK) and a private key (PriK). The private key is used to sign the transaction and create a signature that assures the authenticity of the transaction. The public key, on the other hand, is used by other nodes to verify the authenticity of the transaction along with the concatenated signature. The hash of the public key (P2PKH) is used as an address for the user, so others can pay to this address. There are many kinds of wallet program, i.e., full-service wallets, signing- only wallets, offline wallets, hardware wallets, distributing-only wallets [7].

Full-Service Wallet: It is the simplest and it performs all the functions: it generates the private keys, derives the corresponding public keys, distributing those public keys as necessary, monitor for outputs spent to those public keys (it means, what other has paid to those public keys, such payments are considered outputs), creates and signs transactions (thus, spend those outputs), and broadcast the signed transactions. The main advantage of such a wallet is that it is easy to use and one program that does all. The main disadvantage, however, the private key, which is supposed to be secret to the user, is stored on the device that connects to the Internet. So, there might be a compromised application on the device that can be exploited by an attacker to steal and transmit the private key.

Signing-Only Wallet: To increase security, the wallet is segregated into two parts. One part will generate and store the private keys and operates in a more secure environment. For instance, a device that does not connect to the Internet thus called "Offline Wallet", or dedicated hardware that has, also, no other applications running on it, thus called "Hardware Wallet". The second part, which is called the networked wallet (often called an "Online Wallet" or "Watching-Only Wallet") is complementing the remaining functions.

Distributing-Only Wallet: It is a wallet program that runs on difficult-to-secure environments, such as web servers. It is designed for the public keys (including the P2PKH or P2SH addresses) and nothing more.

Transaction: It is the record that transfers the spent bitcoins or satoshis (fractions of bitcoins) from the payer to the payee [8] [9] [10] [11].

If a payer pays a payee, then that would be considered as an output transaction (TxOut), also called output for short. If the payee does not utilize that amount right away, then it will remain as output and be called Unspent Transaction Output (UTXO). Once a payee decided to spend a UTXO, then the payee should be called a payer, in this case. This payer should reference that UTXO as an input transaction (TxIn), and only then a payer can spend it (pays to another payee). A payer can only spend what he/she owns bitcoins.

Figure 2 shows that a transaction consists of 4 portions. In the first portion, the Version field defines the structure of the transaction and the applicable functions and rules that can be run on the transaction parts, and the Flag if present, then its value is always set to 0x0001 which indicates the presence of the witness data.

If LockTime is set in the fourth portion of the transaction, then the transaction will not be mined until a future time or a block height reaches [10] otherwise, it must be mined immediately. If the value of the LockTime is set to a value that is less than 500,000,000 then it will be interpreted as the number of block height required to be reached, if it is above, then it will be interpreted as the timestamp (it is the time represented by Unix epoch time) required to be reached.

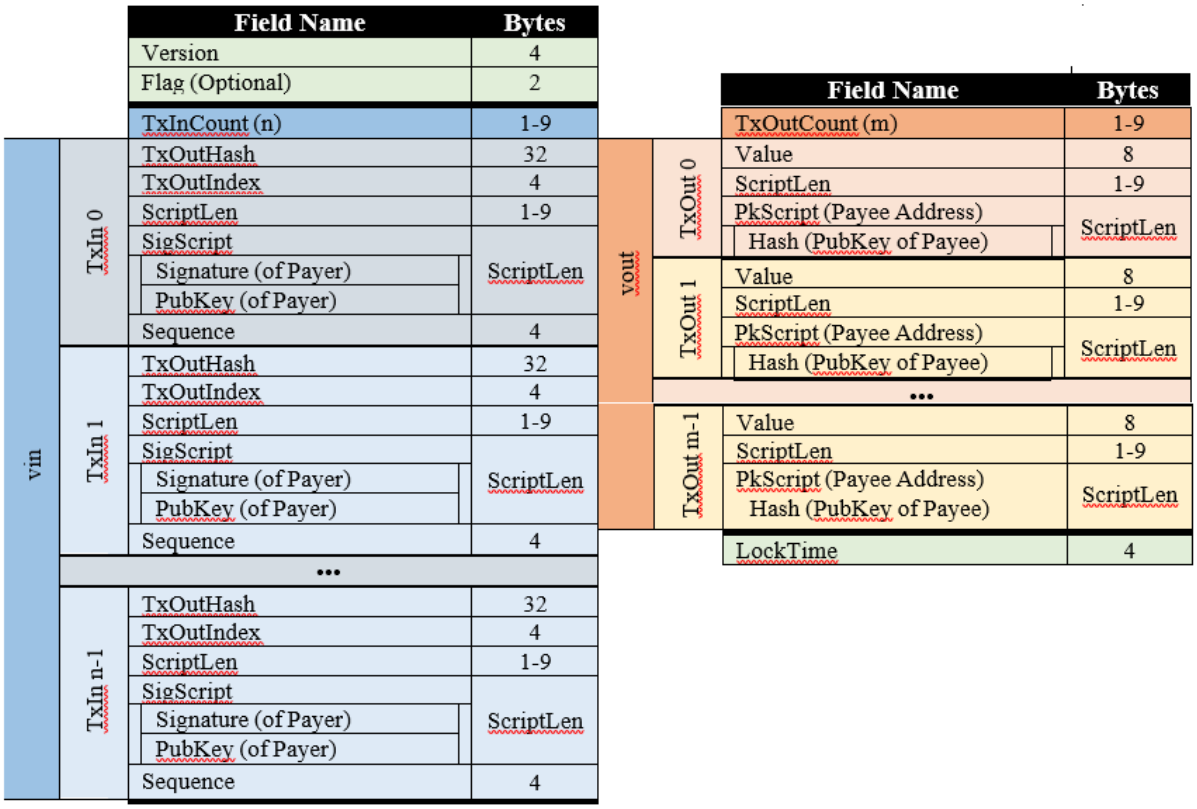

Figure 2: Transaction Structure

The second portion of the transaction lists the inputs that are required to accumulate the amount needed to be paid. So, a transaction can have an n number of inputs, and that number is stored in the field TxInCount. The index of the list of inputs (vin) starts from 0. The purpose of an input is to reference an output that once was paid to the current payer by a previous payer. So, an input consists of the following fields: the TxOutHash which refers to the hash of the previous payer's transaction, the TxOutIndex which refers to the index in the list of outputs in the previous payer's transaction, the ScriptLen states the length in bytes of the next field that is the SigScript, the SigScript consists of two parts, the signature of the previous payer which assures the identity of the previous payer and the public key of the previous payer which is used by the current payer to verify the signature and the referenced output, and the Sequence field is meant to be a counter that tracks the time or the block height, it will be incremented by one every time the timestamp increases by one or the block height increases by one, once it reaches 0xFFFFFF then the transaction is considered ready to be mined. Of course, setting it to 0xFFFFFF from the beginning means the LockTime is disabled and it will not be considered.

Outputs of the transaction are listed in the third portion. A transaction can have an m number of outputs, that number is stored in the field TxOutCount, and it means that the current payer can pay to one payee or more, where each output would correspond to one payee. The index of the list of outputs (vout) starts from 0. The output consists of the following fields: The Value which is the paid amount in satoshis (1 BTC = 100,000,000 Satoshi) [8], the ScriptLen states the length in bytes of the next field that is the PkScript, and the PkScript is the address of the payee which is usually the hash of the public key of the payee. Most often, what is meant to be paid to the payee(s) is less than the accumulated amount that is referenced by all the listed inputs, so the payer can deduct the change and send it to himself, and that is called Change Fee. Rarely or deliberately, there could be an extra amount that has not been paid to any payee and the payer did not consider it in his Change Fee, then it will be considered as Mining Fee (aka Transaction Fee or Miners Fee) and it will be given to the miner who mines that transaction.

The reason that some of the fields are tailed with the word "Script", i.e., PkScript (and it was just public key), because such a field has not exactly a simple value, but it

contains commands that are executed by the node. There is a language behind it; the commands must conform to a particular syntax and adhere to a certain grammar, that is why it is called a Script [12].

Memory Pool (or mempool): It is a temporary place for the unconfirmed transactions that are waiting to be confirmed and mined. Not only for the freshly generated transaction by the users, but also the added back transactions from stale blocks (stale blocks are the blocks which were successfully mined but which aren't included on the current best blockchain, more likely because some other block at the same height had its chain extended first) [13] [4].

Block: It is the data unit of a chain, or the chain itself; because there must be one chain that represents the Bitcoin peer-to-peer network, for instance. A block keeps a record of a transaction, or a set of transactions, by preceding them with the block header and protecting them by proof of work [14] [15] [16].

Field		Bytes
Magic Number		4
Block Size		4
Block Header		80
Version	4	
Previous Block Header Hash	32	
Merkle Root Hash	32	
Time	4	
Bits	4	
Nonce	4	
Transaction Count		1-9
Transaction(s)		
Transaction 0		
Transaction 1		
•••		
Transaction n		

Figure 3: Block Structure

Figure 3 shows the block structure where the Magic-Number field is always set to 0xD9B4BEF9. The Block Size is the number of bytes following up to the end of the block. The Transaction Count is the number of transactions. The Block Header consists of 6 fields: The Version is the block version number and it is changed when the software is upgraded to a newer version, the Previous Block Header Hash is a 256-bit hash to reference, or act as a point to, the previous block's header in the chain and to ensure that no previous block can be changed without also changing this block's header, the Merkle Root Hash is the hash that is derived from the hashes of all the transactions included in this block – thus, ensuring that none of the transactions can be modified without modifying the header – see figure 5, the Time is a Unix epoch time and it is set when the miner started hashing the header, the Bits is an encoded version of the target threshold that is this block's header hash must be less than or equal to, the Nonce is a number that a miner has invested his computation power to calculate it, it is a number that when it is hashed along with the other fields of this block's header, the resulted hash is considered as this block's header hash and it has to be less than or equal the target . All the values of the block's header fields are represented in little-endian, except for the hashes. Figure 4 shows an example of a block header in hex:

Block version: 2
02000000
Hash of previous block's header
b6ff0b1b1680a2862a30ca44d346d9e8
910d334beb48ca0c0000000000000000
Merkle root
9d10aa52ee949386ca9385695f04ede2
70dda20810decd12bc9b048aaab31471
Unix epoch time: 1415239972
24d95a54
*Target: 0x1bc330 * 256 ^ (0x18 – 3)*
30c31b18
Hash of previous block's header
fe9f0864

Figure 4: An Example of Block Header

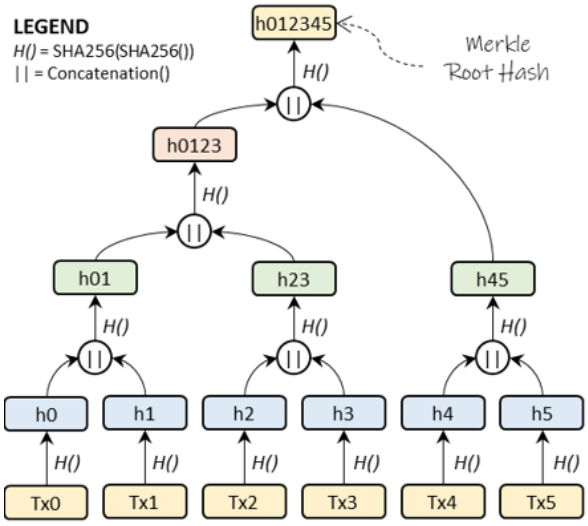

Figure 5: Merkle Tree

The Bits, as mentioned, that is an encoded version of the target is decoded as follow:

1) Considering the value of the Target that is in the example in figure 4, that is 0x30c31b18, which is in little-endian,
2) Transform it to big-endian → 0x181bc330,
3) Take the 1st byte as the exponent, and the rest as significand (mantissa), then apply:

0x1bc330 * 256 ^ (0x18 - 3)

Significand Base Exponent # of Bytes in Significand

= 0x1bc330000000000000000000000000000000000000000000

4) The result is the target threshold.

2.2 Blockchain Characteristics

This section summarizes some of the notable characteristics of the Blockchain-based application as follow:

High Provenance Assurance: The blocks are chained, the transactions are chained, and the ledger is open for the public, thus anyone can verify the provenance of any source of bitcoins, the legitimacy of the transactions, and the authenticity of the blocks. Consequently, this leads to several other characteristics, like:

Transparency: The blockchain is in plain sight to everyone in the network, even to those who join later. Blockchain provides a fully auditable and valid ledger of transactions.

Immutability: The blocks or any part of the blockchain cannot be changed or altered, since they are chained using the "Previous Block Header Hash". Therefore, if anyone thinks of changing a block, he needs to reflect the changes on the "Previous Block Header Hash" of every previous block to geneses block (the ever first block in the chain) by recalculating the hashes below the targets. Also, he needs to own most of the computation power and wins, every time, the mining process to outrace the other miners in the network to develop the longest chain, and that is likely impossible.

Irreversibility: Once the block is successfully mined, and the transactions are confirmed, then the amount is settled, like cash.

Censorship Resistant: Nobody can block or freeze a transaction of any amount at any point in time.

Decentralized Authority: There is no single authority, entity, or person that is stating the rules, governing the system or carrying out a particular model of control or ownership. Everyone is the bank!

Permission-less: There is no system for setting the permissions, privileges, or access rights where it dictates who can join or who has access to what functionalities or parts of information. Anyone can join or leave at any point in time, choose to perform or not any functionality, and read everything.

Anonymity: The users of Bitcoin are addressed by a hash value that does not reveal any information about the identity of the user. Therefore, they can freely make their transactions anonymous, without the fear of the invasions of their privacy.

2.3 Blockchain Protocols

Blockchain protocols can be thought of as a set of rules and communication mechanisms that enable the components of the blockchain architecture to interact with each other and together achieve the intended goals of their existence. For instance, in the case of Bitcoin, the goal is to have a version of electronic cash that would allow online payments to be sent from one party to another without going through a financial institution [1]. Such protocols can be technically implemented using a set of defined messages and defined remote procedure calls (RPCs), as detailed thoroughly in [17] and [18], respectively.

Sometimes, sophisticated protocols are developed, not for the sake of making things complicated, but rather overcoming the security threats and breaches that may arise from weak protocols and avoid the pitfalls as much as possible. The aim of this chapter, however, is to elaborate on the main protocols conceptually, not technically.

From Scratch: As is well known, electronic information are merely bits of zeros and ones, hence they could be easily replicated! Throughout this part of this section, we are going to develop a protocol that allows a coin to be paid electronically, and by every phase, the protocol should be adjusted to overcome its weaknesses. So, let us say that Alice would like to pay a coin to Bob electronically, then Alice needs to use a string of bits to represent a coin, and then give it to Bob. Now, who can prove that this coin is Alice's? What can prevent Bob, or anyone else from forging the same string of bits to pay whomever they like?

There exist tools that can provide Alice with two keys, the first is called a private key, which is the key that Alice must keep secret from everyone, and the second is called a public key, which she can distribute to anyone. Those keys are generated using overly complex mathematical cryptographic algorithms [19]. Alice could use her private key to sign that coin. Again, there exist tools that can apply an algorithm, i.e., Elliptic Curve Digital Signature Algorithm (ECDSA) [20] [21], to sign the bits that represent the coin with Alice's private key. Once signed using the two inputs {Alice's private key + the coin}, as a

Manuscript received January 5, 2025
Manuscript revised January 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.1.23

result, a unique string of bits is generated that is called a signature. Firmly, no one else can generate that signature by any means, unless he possesses Alice's private key. Later, Bob can use Alice's public key, along with the coin and the signature, input them all into a tool, and that tool would verify the coin is Alice's truly. Doing so, firstly, the coin has transformed into a crypto coin – thus the term "Cryptocurrency". Secondly, Alice cannot deny that that coin was not hers – thus the term "Non-repudiation". Moreover, no one else can forge that coin in her name. Alice, however, or even anyone else, can duplicate those strings of bits, and pay Bob multiple times with the same coin – thus the term "Double Spending", as if Alice is minting an infinite supply of money!

One would consider, how did we make sure that the coin is intended for Bob at the first place, and not someone else? The solution is quite simple; Bob can own a private key and a public key as well. The public key could act as the address of Bob. Similarly, for any other public key in the network, including Alice's public key. Therefore, Alice should perform the signing using three inputs {her private key + the coin + Bob's address (in other words, Bob's public key)}. So, looking back at the double-spending problem, one would think of serializing that coin, giving it a sequence number. To have that, we need to introduce a trustworthy third party, i.e., a Bank, which would take care of the serialization of all coins. Now, whenever Alice wants to pay Bob, she needs, firstly, to withdraw that serialized coin from the bank, the bank then would confirm that Alice has that serialized coin, and then deduct one coin from Alice's account. Secondly, Alice needs to sign using four inputs {her private key + the coin + the coin's serial number + Bob's address}. Yet, such a solution is against the goals of the network that is eliminating the need of the third-party financial institution. At this point, we have ended up with the two problems: the double-spending and the need of the bank, rather than solving one. So, how did the Bitcoin network solve these two problems?

Sources of Bitcoins: There are two ways only where one could earn bitcoins in the Bitcoin network. The first way is through mining; upon successful mining of a new block, either the network will reward the miner with bitcoins, such bitcoins are called Reward fees and appear as Coinbase in the first transaction of a block, the owner of the transaction pays a transaction fee as explained at the "Transaction" part section 2.1, or both. The second way can be as a form of payment.

Nonetheless, a transaction cannot spend (or output) bitcoins out of the thin air, and the source of money (input) must be referenced, most often the input refers to unspent transaction output (UTXO). Such a referencing acts as a ledger and creates a kind of chained transactions see figure 6.

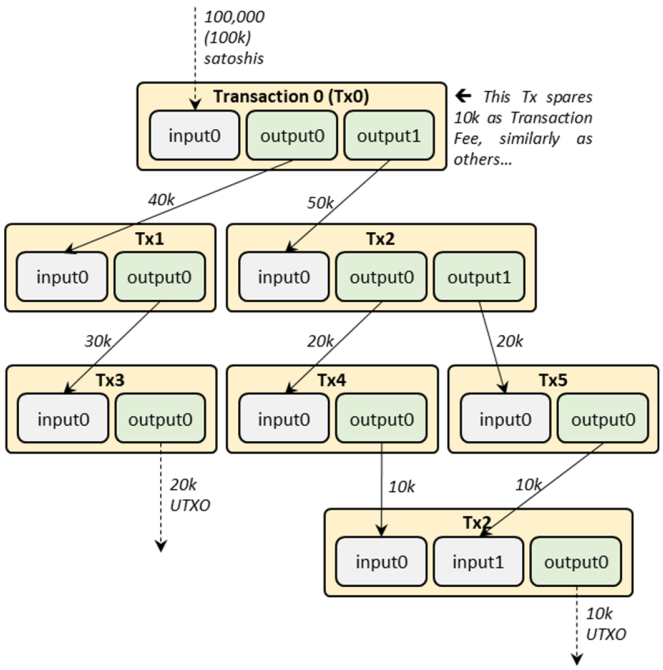

Figure 6: Transaction Propagation

Proof-of-Work (PoW): User A (payer) generates a transaction for user B (payee). The transaction is broadcasted to the entire peer-to-peer network where it is temporarily stored in the memory pool (mempool). A full node would store the transaction in its local copy of the memory pool and relay it to other peers. To maintain the state of the blockchain, the miners query the memory pool and select the transactions of their choice, confirm their legitimacy, put them into a block, add the block the chain by referencing the hash of the previous block, forming a kind of chained blocks, thus the name "Blockchain".

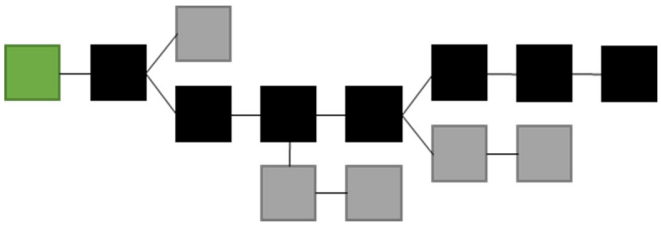

Figure 7: The Blockchain, the main chain is formed by the black blocks, the green is the geneses block, and the rest are stale blocks

Every 2016 blocks (≈ two weeks), the difficulty is adjusted such that the average mining time across the 2016 blocks sustains around 10 minutes. For instance, if the computer hashing rate power increases as time advances due to the

evolvement of the technology – meaning, its speed increases – then, the difficulty will increase. Difficulty is a number between "0" to "1", where "1" is the easiest, and "0" as shown in figure 8.

It is known that the target is a hash value, and a hash value is just a number, an excessively big number in a matter of fact – a 256-bit number. It is quite easy for anyone to find a hash value below the current target, but whenever the bar is lowered, then the difficulty increase, and vice-versa.

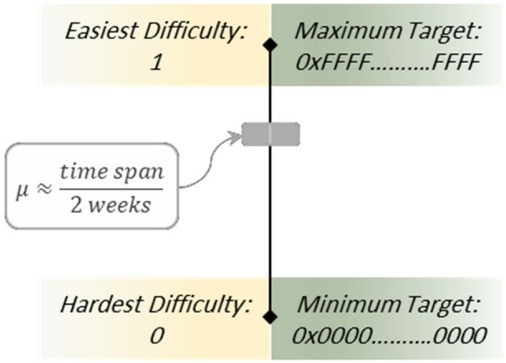

Figure 8: Difficulty Adjustment

The "time span" is the actual time taken to mine 2016 blocks. The ability of a miner to find a nonce such that when it is hashed with all the fields of the block's header will generate a hash of the block that is below the target is what has been referred to as the computing power. There is no means of finding that nonce, except through brute force attack, so the number of hashes it calculates per second is called the Hash Rate, and that is how the computation power is measured [22] [23] [24].

Due to the great incentive, the reward fee and the transaction fee, miners are racing to, successfully, mine new blocks and willing to invest their computation power for it, because only the winner, he who could find the nonce first, will be rewarded, so it is a race!

After a block is mined successfully, then and only then, it will be appended to the chain, and it will be broadcasted to other miners. The other miners will cease their work, verify the block – by, simply, hashing the block's header with its supplied nonce and make sure that the resulted hash value is less than the target, confirm the block by appending to its local copy of the chain and start working on mining a new block.

There could times where two miners successfully mine two blocks at the same. This will fork the chain, but since it is a network, and everyone is mining, the rule is always appending to the longest chain. Soon, the longest chain will be obvious due to the tremendous computation power needed, and the forked chain will be deserted. This will result in blocks called "Stale Blocks", which were successfully mined, but was not appended to the current best blockchain, see figure 7.

Moreover, there could be times where a miner receives a broadcasted block, but he could not append it to his local copy of the blockchain; because the "Previous Block Header Hash" of this received block points to an unknown block. In other words, the miner could not locate its parent. In this case, such a block is called "Orphan Block". Whether it is a stale block or an orphan block, after some time, all the transactions within them will be put back into the memory pool.

The huge work needed by all the miners, from confirming the transactions up to mining the blocks, broadcasting, verifying the blocks and the invested computation power, in that sequence is called the "Proof-of-Work (PoW)" protocol. The PoW is not the only consensus algorithm that exists in any of the Blockchain-based applications. Due to different needs, applications requirement, or security breaches, another version of it has emerged. Beside the proof-of-work (PoW), some of the notable consensus algorithms are proof-of-stake (PoS), proof-of-activity (PoA), proof-of-capacity (PoC), proof-of-burn (PoB), proof-of-knowledge (PoK), and the practical Byzantine fault tolerance (PBFT) [3].

Implementing such a consensus algorithm, where everyone participating in consenting, confirming, or verifying the blockchain and its parts, practically has made the blockchain a public ledger. It eliminates the need of having a single trusted third-party. Also, referencing the transactions using their hashes, as well as the block, eliminates the need of using serial numbers. Besides, if Alice has, for example, one bitcoin, and would like to pay it to Bob, then tell everyone else in the network (including Charlie) to verify and update their blockchain. Once that has happened, Alice would no longer fool Charlie. Let us assume that Alice exploits the brief window that she has, and creates two transactions, one for Bob and another for Charlie almost at the same time. Knowing that, in the Bitcoin network, a transaction will be confirmed upon the successful mining of other 6 blocks (≈ one hour) after its block, then Alice needs to outrace the network to maintain her version of the blockchain and keep up to pass the 6 blocks. Unfortunately, none of the cases will pass through, and the double transactions will be spotted, thus eliminating the threat of double-spending [25].

Other Protocols: Of course, the Bitcoin network contains other protocols as well that are used in the different locations of the network. It contains, but not limited to, protocols for managing contracts, i.e., Escrow and Arbitration, Micropayment Channel, CoinJoin, protocols for managing wallets, i.e., Hierarchical Deterministic Key Creation, Hardened Keys, protocols for managing payments, i.e., Verifying Payment, Issuing Refunds, Disbursing Income, protocols for managing the network, i.e., Peer

Discovery, Connecting to Peers, Initial Block Download, Block Broadcasting, Transaction Broadcasting, and many more.

3. Attacks on Blockchain

As nothing is perfectly secured, the same is true for blockchain technology. It has a fair share of different kinds of attacks. The motivation behind those attacks cannot be determined. The vulnerabilities that enabled such attacks to take place can be determined though. Sometimes, such attacks could be arisen from mistakes within the deployments, weaknesses within the system, negligence from the users, or seldomly, turning the system strengths on itself.

This chapter summarizes some of the main attacks surfaced for blockchain and classify them according to their orientation into two classes: blockchain's core-oriented attacks and blockchain's client-oriented attacks. Henceforth, it could lead to an even better design of future versions of the blockchain or applications based on it.

3.1 Core-Oriented Attacks

This section surveys the attacks that exploit blockchain architecture, its components, and its protocols. It cascades furtherly to the lower layers, such as the machines on which they run and the industry network protocols through which those machines are communicating.

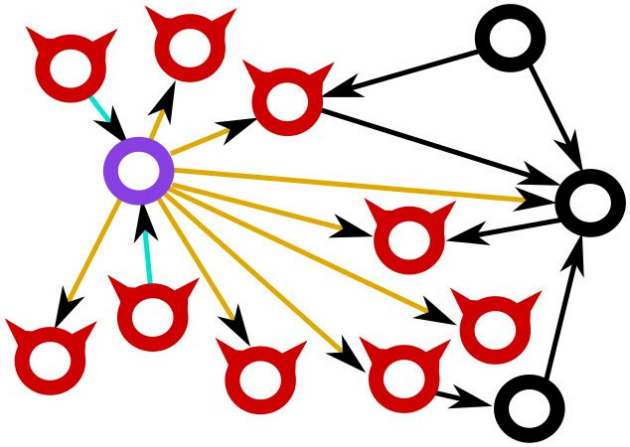

Figure 9: Eclipse Attack

Eclipse Attack: Any node in the Bitcoin network will not be able to stay connected with every other node in the network, rather stay connected to those who are the closest. That node, then, will always seek its ledger's view from those nodes that are connected to it. So, if an attacker managed to surround that node with alienated nodes, see figure 9, then it can present its manipulated version of the ledger and feed incorrect data to that node, and make that node unable to see the true ledger, thus eclipsing [27]. Consequently, it could lead to fake spending. For example, let us assume that the victim node is a merchant who sells cars, the attacker goes to him for buying a car, he pays, and the alienated nodes confirm his payment, not the actual network, which brings false satisfaction to the merchant and hands over the car's key to the attacker! It could lead also to a Timejack attack or double-spending attack.

Sybil Attack: While the eclipse attack aims to hide the true view of the ledger from one user or node, the Sybil attack targets the whole network. The attacker will overwhelm the network with a large number of alienated nodes. By which, they can refuse to receive or relay blocks, effectively blocking other users from a network, or pave the way for the 51% attack or double-spending attack [28].

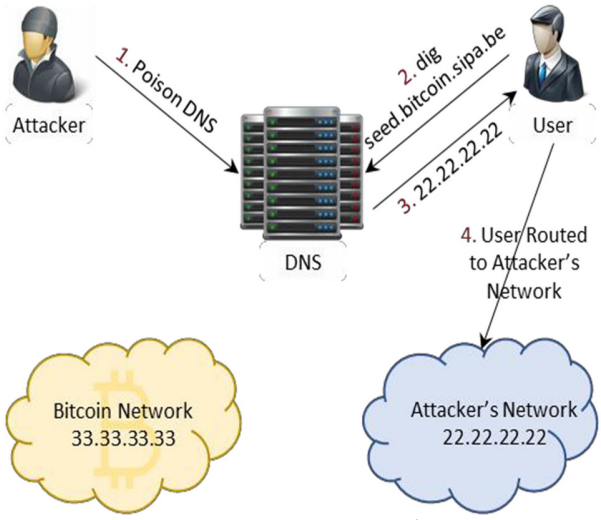

Figure 10: DNS Attack

DNS Attack: For the first time a node tries to join a network, it queries a DNS server to discover the active peers. Before that time, an attacker might have either injected an invalid list of seeder nodes or poisoned the DNS server at the resolver. Thus, the node would think that it is connected to the real Bitcoin network, but the reality is that the node is connected to the attacker's network, see figure 10 [3]. Such an attack could lead to an eclipse attack or a Sybil attack.

BGP Hijacking and Spatial Partitioning: There are the full nodes and the SPV nodes who draw their view of the ledger from the full nodes. When a full node is compromised, all its associated SPV nodes are also compromised. The full nodes are spatially distributed across

the globe, see table 1. The Internet Service Providers (ISPs) controls the flow of traffic on the Internet, which owns one or more Autonomous Systems (ASes), responsible for handling traffic routing. An adversarial AS can hijack the traffic for a target AS that hosts most of the full nodes. Consequently, isolate more than 50% of the network's hash rate, which could enable the 51% attack or delay the block propagation. Such delays increase the probability of other attacks like Blockchain fork, consensus delay, and double-spending.

Table 1: Full Nodes Distribution at Top 10 Countries, as of Dec. 22, 2020 [29]

RANK	COUNTRY	NODES
1	n/a	2833 (24.67%)
2	United States	1983 (17.27%)
3	Germany	1771 (15.42%)
4	France	581 (5.06%)
5	Netherlands	457 (3.98%)
6	Canada	380 (3.31%)
7	United Kingdom	343 (2.99%)
8	Singapore	255 (2.22%)
9	Russian Federation	227 (1.98%)
10	Japan	224 (1.95%)

Distributed Denial of Service (DDoS) Attack: One of the most common attacks on online services is the DDoS attack. Similarly, in the Blockchain realm. DDoS attacks manifest themselves in several ways. For example, 51% attack could lead to DDoS by which the adversaries can prevent other miners from adding their mined blocks, invalidating ongoing transactions, and cause hard forks [3].

Stress Testing Attack: It is a form of DDoS attack. Consider the maximum size of a block is 1MB, the average size of the transaction is 500B, this means that there are about 2000 transactions per block. Also, consider the fact that the Bitcoin network mines 1 block per 10 minutes, this leads to a throughput of 200 transactions per minute (or about 3.33 transactions per second). Such throughput is considerably low compared to the Visa credit network, which verifies up to 2000 transactions per second. An attacker could exploit such knowledge, introduce many Sybil identities, each identity control multiple wallets, use them all, and flood the network with dust transactions (e.g., 0.0001 BTC per transaction – so small transactions). As a result, this congestion could lead to a double-spending attack.

Memory Pool Flooding Attack: Another form of DDoS attack. It is practiced that miners prioritize the transactions in the memory pool based on the mining fee that they offer. In other words, the transactions with the highest mining fees are picked up first and mined. So, users tend to pay more mining fees as an incentive for miners to pick up and mine their transactions. An attacker could again flood the memory pool with dust transactions, which will create panic among the legitimate users who are tempted to pay higher mining fees. So that is the purpose of such an attack, is to increase the mining fee.

Block Withholding Attack: It is the act of malicious nodes when they deliberately mask, forge, or withhold important information they need to be relay across the network [3].

Finney Attack: It is a kind of block withholding attack and double-spending attack where a miner who wants to buy a product from a merchant with an amount x, does the following:

1) The miner mine a block but introduce privately in this block a transaction that sends that amount x to himself.
2) Once the block is mined, the miner does not broadcast, rather hold it locally within his machine,
3) In the meanwhile, the miner releases another transaction to the network that sends the same amount x to the merchant,
4) After the merchant accepts the transaction and delivers the product,
5) The miner (attacker) broadcasts the block he mined at the first step, which will invalidate the block that was mined later for the merchant, thus double-spending.

The Finney attack has a very low success rate should the merchant waits for more confirmations (i.e., 6 confirmations as recommended); because the pay to himself transaction was locally introduced.

Fork after Withholding (FAW) Attack: To explain this kind of attack, we need to introduce an advanced component of the Blockchain architecture that is called "Mining Pools". As the value of the bitcoin rises, so does the mining industry, technology, and techniques. A mining pool is an organized group of miners who collectively mine a block – thus, collectively achieve a much higher hashing rate and share the profit. In every mining pool, a pool manager is appointed, and he behaves as a single miner in the eye of the Bitcoin network. Each member (miner) either partially (PPoW) or fully (FPoW) finds the proof-of-work, either way, the pool manager will receive any gained reward, then he splits it among all the members of that pool. Since

most of these mining pools are open, miners can freely join or leave these pools [30].

The FAW attack is carried as follows. An infiltration miner joins a target pool, while is still being an innocent miner of his preferable pool! At his preferable pool, if he mines a block through FPoW, then he gladly submits it to the pool manager immediately and gains legitimate profit. However, as the target pool, if he mines a block through FPoW, then he withholds the block; waiting for another miner, who is not a member of the target pool to append a valid block, only then he submits the block to the pool manager hoping to cause a fork, invalidating other's blocks created afterward, and gain all the profit. On the other hand, if it did not go as planned, he will be rewarded as any other member in the target pool at least. Still, such an attack is bounded by the honesty and the rational behavior of the pool manager of the target pool. The pool manager may discard the submitted block by the infiltration miner if he noticed the valid appended block after all. In addition, if another member of the target pool finds the block, and submits it to the pool manager, there is no use for furtherly withhold the block [31].

51% Attack (**aka Majority Attack**): It is where a single entity or an organization is able to control the majority of the hashing rate – computation power – a 51% or more. In such a scenario, the attacker will always win the mining. Consequently, he could exclude or modify the ordering of the transaction, prevent some of the transactions from being confirmed – thus, transaction denial of service, or prevent some or all other miners from mining – thus, mining monopoly [32].

Selfish Mining Attack: Blockchain considers the longest chain is the true chain where every miner should append its mined blocks to it. A selfish miner would mine his blocks privately without releasing them to the public upon discovery. After mining a good number of blocks privately which made the local copy of the blockchain longer than the published one, the miner releases it which will be validated by the other miners in the network and consider it the longest true blockchain. Such an attack can disturb the network by invalidating the blocks of honest miners, and that selfish miner (the attacker) will gain all the rewards from the invalidated blocks. Furthermore, there is a window for "Double Spending" and "Fork after Withholding" [3].

Consensus Delay: In this kind of attack, the aim is to introduce latency in the network, which will lead to rejection of blocks, prevent peers from reaching consensus about the state of the blockchain, or any other sabotage purposes. An attacker would do so in many ways. For instance, injecting false blocks, i.e., stale blocks or blocks that contain double-spending transactions. The problem can be magnified if the attacker controls Sybil nodes – that would add significant delays among the legitimate nodes.

Timejacking Attack: The nodes in the Bitcoin network maintain an internal counter that denotes the network time. During the bootstrapping phase, the node obtains the network time by requesting it from the neighboring nodes, calculate and store the median. If the median is greater than 70 minutes, then it will refer to its system time. An attack would carry on, an eclipse attack, for instance, to let the neighboring nodes, supply false timestamp. Such an attack could let the node rejects the blocks because it sees that their timestamp exceeds the network timestamp by 120 minutes. Eventually, such a targeted node will be isolated from the network [3].

3.2 Client-Oriented Attacks

This section surveys the attacks that affect the Blockchain clients' applications, i.e., applications run on the machines of the users or the miners.

Wallet Theft: If malware is installed on a user's machine, then his private keys might be in danger of being stolen, or the malware might have the ability to do transactions on his behalf. Unless those keys are encrypted and safely guarded.

Cryptojacking Attack: Such an attack is targeting the web and cloud-based services to mine a block illegally without consent. In other words, the attacker recruits the webserver without its owner's awareness and steal its computation power to mine the blocks. In other cases, turn it into a mining pool.

Double Spending Attack: The double-spending refers to the use of a one-time transaction twice or even multiple times. If for example, Alice wanted to pay Bob and Charlie with the same coin can be recalled, see section 2.2. The solution was for Charlie is to wait for six more blocks (1 hour) to be sure that that coin was not paid for by someone else. The problem is when Charlie delivered the product or the service before that waiting period, perhaps he was optimistic, or it was out of trust. Nonetheless, such a carried-out action by Alice is called a double-spending attack.

Transaction Malleability: It is a kind of denial-of-service attack. The signature protects the transaction, but nothing protects the signature. An attacker's intent might be only to sabotage the transaction by making an arbitrary modification to the signature, which will render the transaction invalid. Consequently, the transaction is dropped.

4. Blockchain Applications

Blockchain does not sound easy. Moreover, its complexity keeps growing every day. However, while it has a distinguishing property of immutability, it can be suitable for every application out there, as it is. For example:

National ID: It is an application that provides a unique identity for every citizen in the country. It might as well store the citizen's necessary information. Such an application could serve as the basis for other applications, i.e., Electronic Voting. Furthermore, such an application might go viral and tries to cover all the people on earth.
One of the characteristics of blockchain is anonymity. How is it going to work with the National ID application? If it is chosen that, the users must declare themselves and stay not anonymous anymore, who is going to define them and verify the integrity of their information? If a third-party trust-worthy government entity is chosen to do so, then here goes decentralized authority merit.

On the other hand, what is representing the coin in this case? If it is the citizen record, how it can be timestamped? Who is going to mine this transaction? What is the incentive for the miners? Should there be one chain or multiple chains? How each country could assure the integrity of its citizens' records, or other countries' citizens? How big the size of the chain would be in terms of bytes?
If somehow, it is managed to develop a newer version of the blockchain that can resolve the aforementioned wonders, what are the kinds of attacks that can affect this version negatively? Will the blockchain ingestion and the anonymity, explained in [3], take advantage of such a version?

Electronic Voting: It is that kind of a system that seeks public votes to elect a member of the parliament, for instance. Such a system is very crucial and too politically sensitive; one vote could make a big difference. So, perhaps that is why most of them are still relying on the paper manual process because they do not tolerate the technology! With characteristics, like transparency, immutability, and irreversibility, offered by the blockchain, it may soften their opinion about the technology, but not the anonymity. Whoever is voting, the public – the citizen – must be known and counted, thus implementing the National ID system as the basis of such a system is needed.

Therefore, all the risks and security threats that apply to the national ID system are relevant to the electronic voting system. Like a national ID system, electronic voting may bring new attack avenues and a whole another set of vulnerabilities.

Healthcare System: It is a system that maintains the integrity of health records while establishing a single point of truth. For example, doctors, hospitals, and laboratories can all request patient information that has a record of origin. Conversely, patients can request doctors, hospitals, and laboratories information.

However, while the patients' identities must be protected so they can benefit from the anonymity characteristic of the blockchain, the identities of the doctors, hospitals, and laboratories cannot benefit from the anonymity and must be known. So, attacks like eclipse, Sybil, or 51% attack, could be carried out on such a system just to create confusion and sabotage the integrity of the information.

5. Conclusion

Blockchain technology is extraordinarily great. It has brought forward some of the characteristics that the world is trying to have for a long noticeable time. Nevertheless, the enhancement does not have to be restricted within its protocols or be bounded by its characteristics. Each application has its own requirements, surely it could benefit some of the key characteristics of the blockchain, but there might be changes needed to be carried on the architecture of the blockchain to modify the other characteristics that do not suit the application's requirements. Besides, thoughtful consideration and risk-based thinking must take place for preventing the attacks and avoiding the vulnerabilities within the Blockchain-based applications.

The paper has, comprehensively, explained the architecture, the main components, the protocols, and the key characteristics of blockchain technology. It did highlight the common attacks that can be carried on the Blockchain-based system and the vulnerabilities that they could have exploited. All for the aim that it could help during the development of any new application that is based on the blockchain.

The paper could be enhanced by thoroughly explain the attacks, and how they exactly work, perhaps programmatically. Additionally, it could be enhanced by explaining the rest of the blockchain's protocols or explain the other versions of the consensus algorithm like proof-of-stake (PoS), proof-of-activity (PoA), proof-of-capacity (PoC), proof-of-burn (PoB), proof- of-knowledge (PoK), or the practical Byzantine fault tolerance (PBFT). Furthermore, one could fork another paper that focuses on a particular new application and explain the changes needed to the blockchain architecture to fulfill its requirement.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.
[2] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," Journal of Cryptology, no. 3, p. 99–111, January 1991.
[3] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang and A. Mohaisen, "Overview of Attack Surfaces in Blockchain," in Blockchain for Distributed

Systems Security, 2019, pp. 51-66.
[4] B. Project, "Glossary," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/glossary.html [Accessed 14 Dec 2020].
[5] B. Wiki, "Bitcoin Core 0.11 (ch 1): Overview," Bitcoin community, 24 Apr 2019. [Online]. Available: https://en.bitcoin.it/wiki/Bitcoin_Core_0.11_(ch_1):_Overview. [Accessed 14 Dec 2020].
[6] B. Project, "Operating Modes," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/devguide/operating_modes.html
[7] B. Project, "Wallets," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/devguide/wallets.html
[8] B. Project, "Transactions," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/devguide/transactions.html
[9] B. Wiki, "Transaction," Bitcoin community, [Online]. Available: https://en.bitcoin.it/wiki/Transaction#General_format_.28inside_a_block.29_of_each_input_of_a_transaction_-_Txin.
[10] B. E. K. Seguias, "Bitcoin Transactions (pre-segwit)," 11 January 2020. [Online]. Available: https://delfr.com/bitcoin/bitcoin-transaction-pre-segwit/. [Accessed 16 Dec 2020].
[11] KLmoney, "Part 1: Transaction Basics," [Online]. Available: https://klmoney.wordpress.com/bitcoin-dissecting-transactions-part-1/ . [Accessed 14 Dec 2020].
[12] B. Wiki, "Script," Bitcoin community, [Online]. Available: https://en.bitcoin.it/wiki/Script.
[13] B. Project, "P2P Network," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/devguide/p2p_network.html
[14] B. Wiki, "Block," Bitcoin community, [Online]. Available: https://en.bitcoin.it/wiki/Block.
[15] B. Wiki, "Block hashing algorithm," Bitcoin community, [Online]. Available: https://en.bitcoin.it/wiki/Block_hashing_algorithm
[16] B. Project, "Block Headers," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/reference/block_chain.html.
[17] B. Project, "P2P Network," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/reference/p2p_networking.html
[18] B. Project, "RPC API Reference," Bitcoin Developer, [Online]. Available: https://developer.bitcoin.org/reference/rpc/index.html
[19] B. E. K. Seguias, "Bitcoin – Private key, Public key, and Addresses," delfr.com, 20 July 2018. [Online]. Available: https://delfr.com/bitcoin/bitcoin-private-key-public-key-addresses/.
[20] B. E. K. Seguias, "Bitcoin - Elliptic Curve Digital Signature Algorithm (ECDSA)," 8 September 2018. [Online]. Available: https://delfr.com/bitcoin/bitcoin-ecdsa-signature/
[21] B. E. K. Seguias, "Digital Signature and Other Prerequisites," Delfr.com, 5 April 2018. [Online]. Available: https://delfr.com/prerequisites/digital-signature-prerequisites-monero-part-1-10/.
[22] B. Wiki, "Protocol rules," Bitcoin Community, 23 June 2020. [Online]. Available: https://en.bitcoin.it/wiki/Protocol_rules
[23] B. Wiki, "Difficulty," Bitcoin community, 17 December 2019. [Online]. Available: https://en.bitcoin.it/wiki/Difficulty
[24] B. Community, "Network Difficulty," Blockchain.com, [Online]. Available: https://www.blockchain.com/charts/difficulty
[25] M. Nielsen, "How the Bitcoin protocol actually works," michaelnielsen.org, 6 December 2013. [Online].Available: https://michaelnielsen.org/ddi/how-the-bitcoin- protocol-actually-works/
[26] B. E. K. Seguias, "Consensus in distributed systems," Delfr.com, 16 November 2020. [Online]. Available: https://delfr.com/bitcoin/consensus-in-distributed-systems/
[27] A. Marketing, "10 Blockchain and New Age Security Attacks You Should Know," Hewlett Packard Enterprise Development LP, 22 January 2019. [Online]. Available: https://blogs.arubanetworks.com/solutions/10-blockchain-and-new-age-security-attacks-you- should-know/
[28] B. Academy, "Sybil Attacks Explained," Binance Academy, [Online]. Available: https://academy.binance.com/en/articles/sybil-attacks-explained
[29] BITNODES, "GLOBAL BITCOIN NODES DISTRIBUTION," BITNODES, [Online]. Available: https://bitnodes.io/ [Accessed 22 December 2020].
[30] Eyal, "The Miner's Dilemma," in 2015 IEEE Symposium on Security and Privacy, San Jose, CA, 2015.
[31] Y. Kwon, D. Kim, Y. Son, E. Vasserman and Y. Kim, "Be Selfish and Avoid Dilemmas: Fork After Withholding (FAW) Attacks on Bitcoin," in CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017.
[32] B. Academy, "What Is a 51% Attack?," Binance Academy, [Online]. Available: https://academy.binance.com/en/articles/what-is- a-51-percent-attack