Hindawi
Security and Communication Networks
Volume 2018, Article ID 9675050, 27 pages
https://doi.org/10.1155/2018/9675050

Research Article

A Survey of How to Use Blockchain to Secure Internet of Things and the Stalker Attack

Emanuel Ferreira Jesus, Vanessa R. L. Chicarino, Célio V. N. de Albuquerque, and Antônio A. de A. Rocha

Institute of Computing (IC), Fluminense Federal University (UFF), Niterói, RJ, Brazil

Correspondence should be addressed to Emanuel Ferreira Jesus; efjesus@id.uff.br

Received 27 October 2017; Revised 29 December 2017; Accepted 11 February 2018; Published 8 April 2018

Academic Editor: Félix Gómez Mármol

Copyright © 2018 Emanuel Ferreira Jesus et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The Internet of Things (IoT) is increasingly a reality today. Nevertheless, some key challenges still need to be given particular attention so that IoT solutions further support the growing demand for connected devices and the services offered. Due to the potential relevance and sensitivity of services, IoT solutions should address the security and privacy concerns surrounding these devices and the data they collect, generate, and process. Recently, the Blockchain technology has gained much attention in IoT solutions. Its primary usage scenarios are in the financial domain, where Blockchain creates a promising applications world and can be leveraged to solve security and privacy issues. However, this emerging technology has a great potential in the most diverse technological areas and can significantly help achieve the Internet of Things view in different aspects, increasing the capacity of decentralization, facilitating interactions, enabling new transaction models, and allowing autonomous coordination of the devices. The paper goal is to provide the concepts about the structure and operation of Blockchain and, mainly, analyze how the use of this technology can be used to provide security and privacy in IoT. Finally, we present the stalker, which is a selfish miner variant that has the objective of preventing a node to publish its blocks on the main chain.

1. Introduction

Internet of Things (IoT) and Blockchain are considered emerging concepts and technologies. At the same time they transform concepts and create new possibilities, each in their respective scenarios, and there is an opportunity to create applications that can share the intrinsic characteristics of both, exploring how the IoT can benefit from the decentralized nature of the Blockchain.

The IoT is a comprehensive term referring to ongoing efforts to connect a wide variety of physical things to communication networks. Currently, the Internet has not only conventional computers connected but also a significant heterogeneity of equipment such as TVs, laptops, fridges, stoves, electrical appliances, cars, and smartphones. In this new scenario, projections indicate that the Internet will have over 50 billion devices connected until 2020 [1]. Within the IoT domain, there are several types of applications, such as smart cities, smart healthcare, and smart home.

At the same time that the IoT can provide us with valuable benefits, it also increases the risk of exposure to various security and privacy threats; some of these threats are new. Before the advent of the IoT, information leakage and denial of service were the most security threats reported. With the IoT, security threats go far beyond the theft of information or denial of service. These threats can now be potentially related to the real lives, including physical security. Other concerns are related to privacy. IoT brought with it an increase in the amount of personal information delivered and shared between connected devices. Although it is not a new demand or unique in this new scenario, privacy is an important element.

Security solutions and privacy should be implemented according to characteristics of heterogeneous IoT devices. There is a demand for security solutions that are capable of providing equivalent levels of security for various types of devices and demands mechanisms capable of audit and access control in these environments.

In this context that Blockchain also falls, because this technology can be used to authenticate, authorize, and audit data generated by devices. Also, because of its decentralized nature, it eliminates the need to trust in the third party and does not have a single point of failure.

Blockchain (also known as "the protocol of trust") is a concept that aims to decentralization as a security measure, has a function to create a global index for all transactions that occur in a given network, and makes them immutable. It works as a public, shared, and universal ledger. It creates consensus and confidence in direct communication between two parties, without any third party. We also can use Blockchain in supply chain, smart contracts, and digital identity management and in some other applications [3].

This paper aims to familiarize newly interested, as well as updating the readers who have some prior knowledge of Blockchain, and this includes the recent applications in security and privacy, and how their use can leverage the IoT. The approach offered will be a survey of the state-of-the-art articles in which the Blockchain is used to provide some level of privacy and security to IoT and will present a variant of a selfish mining attack [4], which we call *stalker*. The stalker is a malicious mining that aims to block a specific miner to publish its blocks.

We structured this paper into five sections. Section 2 will present the theoretical foundations for the understanding of the proposed solution. Section 3 shall submit all the working mechanisms of Blockchain. Section 4 describes some cases of use for Blockchain to provide security and privacy at IoT. Section 5 presents the stalker. Finally, Section 6 presents the final considerations and open questions.

2. Theoretical Foundation

This section presents an IoT overview, approaching the classifications and taxonomies proposed for your infrastructure and applications followed by some common definitions in the security and privacy area and the main concepts needed to understand Blockchain vision.

2.1. Internet of Things. The IoT covers the processing of data and the communication between devices of different platforms and capacities of autonomic, without human intervention. In recent decades, this term emerged as an evolution of the Internet and presented itself as a new technological and social paradigm. The IoT is considered an extension of the current Internet, and it provides computing and communication to connect objects to the Internet. The connection to the worldwide computer network will enable the remote control of objects and allow the objects to be accessed as services providers, making them smart objects.

The first device connected to the Internet was presented in 1990 at *INTEROP '89 Conference* by John Romkey. He created a toaster that could be turned on and off by the Internet, connecting the toaster to a computer with network TCP/IP. In September 1999, Ashton, founder and executive director of the Auto-ID Center, delivered a lecture to the Procter and Gamble, presenting the idea of using electronic tags in the company's products, to facilitate the logistics of the production chain, through identification of radio frequency (RFID). To draw the executives attention, he placed in the title of the presentation the expression *"Internet of Things."* For this reason, he is considered the term's creator, to describe that objects can connect to the Internet, creating a more intelligent world. Ten years later, Ashton published an article where he introduced himself as the creator of the term [5].

From 2005, the discussion on the IoT became widespread, began to gain the attention of governments, and appear related to privacy and data security issues. In this year, the IoT became the agenda of the *International Telecommunication Union* (ITU), the United Nations agency for information and communication technologies, which publishes an annual report on emerging technologies.

The term IoT gained popularity quickly, between the years of 2008 and 2010, due to maturity of *Wireless Sensor Networks* (WSN) [6] and advances in home and industrial automation. In this period, techniques to explore the various limitations of the devices emerged such as memory, power, scalability, and robustness of the network. On October 28 of 2008, Rob Van Kranenburg published the book *"The Internet of Things,"* which addresses this term under a new paradigm in which the objects produce information that should be stored and protected. This book is one of the major theoretical references about the IoT [7]. In 2011, Gartner Inc. included the term "Internet of Things" like an emerging technology in his *Gartner Hype Cycle* [8] that provides a graphic representation of the maturity and adoption of technologies and applications, and how they are potentially relevant to solving real business problems and exploiting new opportunities.

Currently, there is not a single definition of IoT. However, several authors and institutions have contributed to the construction of his vision. Atzori et al. [9] described IoT as a variety of things or objects, such as tags for the radio frequency (RFID) identification, sensors, actuators, and cell phones. These devices interact with each other cooperating with its neighbors to achieving common goals. The author divides this visions into Internet-oriented (middleware), object-oriented (sensors and actuators), and semantics-oriented (the representation and information storage).

Some relevant institutions have emphasized the concept that the IoT should focus mainly on "things," and the way to its full implementation should begin with the increase in the things intelligence. Some definitions in the literature derived from this vision, one of them, a proposal by the research group in the IoT (European Research Cluster on the Internet of Things (IERC)). The IERC presents IoT as "a global network infrastructure and dynamic with capacities of autoconfiguration, based on communication protocols standardized and interoperable, where physical 'things' and virtual machines have identities, physical and virtual personalities and use intelligent interfaces, being integrated perfectly into the network" [10].

The ITU-T (Telecommunication Standardization Sector) proposed a model composed of four layers [11]:

(i) Application layer: responsible for providing services to customers, for example, health monitoring and smart home.

(ii) Application support layer: responsible for specific support, which meets the requirements of a particular application and generic, which are common and applicable to many applications, such as processing or storage.

(iii) Network layer: responsible for relevant functions to control network connectivity, such as mobility management, authentication, authorization, and accounting, as well as transport management information related to IoT.

(iv) Devices layer: represented by the devices and gateways contemplating its elements as processors, memories, firmware, sensors, and actuators and their features. Device features include the ability of devices to interact directly with the communication network; they are able to collect and send information directly, without using gateway capabilities, for network communication. Gateway features include support for multiple interfaces, allowing communication of IoT devices, even though different types of wired or wireless technologies, such as ZigBee, Bluetooth, or Wi-Fi.

This model includes management features and security features associated with the four layers. They are also categorized into generic and specific capabilities. Generic management features in IoT include device management, such as remote device activation and deactivation, diagnostics, firmware or software upgrades, device status management, network topology management, and traffic and congestion management. Generic security features include authorization, authentication, confidentiality, and application data integrity protection and signage and privacy protection.

There are numerous and diverse applications for IoT. These applications permeate people daily life, businesses, and society as a whole, transforming the world into a *smart world*, which allows the computation to become "invisible" for the user, through the relationship between man and machine, making the world more efficient and effective [12]. Figure 1 shows an overview of the work of IoT:

(i) Intelligent products: goods purchased by consumers, such as smartphones, smart house, smart car, smart TV, and wearables.

(ii) Smart health: fitness and health care, for example, monitoring and controlling heart rate during exercise and monitoring the conditions of patients in the hospitals or their homes. The prevention of health problems becomes more effective with a real-time collection of information from our body and diagnoses become more accurate, with a patient profile that has long-term records.

(iii) Intelligent transport: notification of traffic conditions, intelligent control of routes, remote monitoring of the vehicle, coordination of highways, and intelligent integration of platforms.

(iv) Intelligent power distribution (smart grid): monitoring of energy installations, smart substations, power distribution, and remote measurements of residential power meters.

(v) Logistics smart e-commerce: traceability, distribution, and inventory management.

(vi) Smart industry: energy savings, pollution control, manufacturing safety, monitoring products life-cycle, tracking goods in the supply chain, monitoring of environmental conditions, and production control processes.

(vii) Precision agriculture: quality management, environmental monitoring for production and cultivation, and production process management.

(viii) Smart cities: structural monitoring, monitoring of vibrations and conditions of materials in buildings, bridges, and historical monuments. Electrical energy: street smart lighting. Security: monitoring, fire control, and alarm systems. Transport: smart roads with warnings, messages, and deviations in accordance with the climatic conditions and unexpected events such as accidents or traffic jams. Parking: real-time monitoring of the parking spaces availability. Waste management: optimizing the route of garbage collection with trash levels detection in containers.

2.2. Fundamental Safety Principles. Security and privacy are fundamental principles of any information system. We refer to safety as the combination of integrity, availability, and confidentiality. Typically it is possible to obtain security using a combination of authentication, authorization, and identification. These concepts are defined below [13]:

(i) Integrity: it is the certainty that the information has not been altered, except by those who have the right to make these changes. In the Blockchain context, integrity provides the guarantee that transactions are immutable. Commonly, cryptographic mechanisms are used to check integrity.

(ii) Availability: it ensures that users of a given system will be able to use it whenever necessary. In other words, the service is always active when requested by a legitimate user, and this requires the communication infrastructure and the database. The Blockchain achieves this objective by allowing users to establish connections with multiple users and to maintain the blocks in a decentralized way with various chain copies on the network.

(iii) Confidentiality: it is the guarantee that the unauthorized persons will not obtain information. That is, only those with the rights and privileges will be able to access the information, whether it is in processing or transit. To ensure this principle, the Blockchain uses mechanism for pseudo-anonymization, like the use of hash functions to blind users identities.

(iv) Authentication, authorization, and auditing: this seeks to verify the identity of who performs a specific function in a system, check what rights that user owns, and store usage information for that user. The

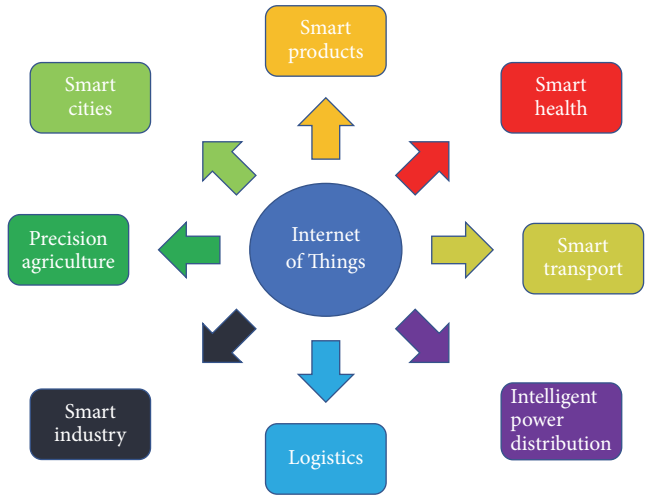


Figure 1: Applications of IoT.

structure of the Blockchain ensures these three functions, since only users who have the private keys can perform transactions, and all transactions are public and auditable.

(v) Nonrepudiation: it guarantees that a person cannot deny an action in a system. The nonrepudiation provides evidence that a user performed a specific action such as transferring money, authorizing a purchase, or sending a message. As all transactions are signed, a user cannot deny that he has done it.

The privacy can be defined as the right that an individual has to share their information. Users of Blockchain use a pseudonym (address) to perform their transactions. Usually, each user has hundreds of addresses. A transaction can be seen as a chain of signatures that prove the possession and transfer of values, so auditable way. One of the main concerns is that these transactions may disclose information from a user, such as buying habits and frequented locations or data usage.

The concept of privacy in Blockchain consists in keeping the anonymity and the untying of transactions. The anonymity of transactions requires that it is not possible to link a particular transaction to a user; for this reason, the user uses a different address for each new transaction. Untying assumes that both Blockchain addresses and transactions are not bound to the actual identities of the users; once the data of these transactions are routed to a random set of points in the network.

2.3. Hash Functions and Encryption. All the possession of resources and transactions on the network is made using the concept of keys and digital signatures. The keys used are generated by applying the concept of public key cryptography. A pair of keys is generated: a public key that can be shared and a secret that only the owner has access. The entire transaction requires a signature to be considered valid and to prove the ownership of the resources expended.

2.3.1. Hash Functions. Hash functions are mathematical functions that generate a summary, a data fingerprint. When applied to a given dataset, it generates an output, which is unique (there may be two data sets with the same hash, but the likelihood of occurrence is extremely low). One of the most frequent uses for the hash is verifying data integrity. The hash output size depends on the algorithm used, but what is important is that it is always the same size, regardless of input size. Examples of hash algorithms are the SHA-256 and the RIPEMD160 [14]. The hash algorithms must have specific characteristics:

(i) One way: it must be computationally very difficult to find the input from hash values.

(ii) Compression: it is desirable that the hash size represents a small fraction of data.

(iii) Ease calculation: the hash algorithm must not be costly to calculate the hash value.

(iv) Diffusion: to hinder the reverse engineering of the algorithm, when one bit of input is changed, the hash result should be changed from a number of bits next to 50%.

(v) Collision: it should be computationally difficult to find two inputs that generate the same hash.

2.3.2. Encryption. Encryption is the set of techniques that transform intelligible information into something that an outside agent is unable to understand. Encryption systems work as follows: given a message and a key, the system generates a new ciphered message to be transmitted over unprotected channels, without running the risk of being understood by others who do not possess the decryption key. The system will only be complete if the encrypted message can be de-encrypted, usually through the same (symmetric) or another (public/private) key.

It uses key pairs, one public and one private. The first to encrypt and the second to de-encrypt and vice versa; this

is possible due to the use of some mathematical functions that have the property of being irreversible. The most mathematical functions used are prime numbers factorization (IFP, *Integer Factorization Problem*); elliptical curves (ECDLP, *Elliptic Curve Discrete Logarithm Problem*); and discrete logarithms (DLP, *Discrete Logarithm Problem*). The efficiency of an encryption scheme can be measured by considering the following:

(i) Computational load: it measures the efficiency with which the algorithms can implement the changes with the keys.

(ii) Key size: the NIST indicates the use of key pairs (public, private) with sizes, in bits, for each type of implementation: RSA (1088, 2048), DSA (1026, 160), and ECC (161,160). The ECC has a significant advantage in this aspect.

(iii) Size of band: it matches the number of bits required to transmit a message, after encoding or signing.

The paper [15] compared the ECC with the RSA and concluded that for the same security level ECC has a lower computational load, lower key size, and smaller size of the band. For these reasons, Bitcoin has adopted the elliptic curves system as defined in a standard called *secp256k1*, established by the National Institute of Standards and Technology (NIST). For more information about elliptic curves, we recommend [16].

2.3.3. Digital Signature, Address, and Wallet. A digital signature can be defined as an encryption of a document hash, using a private key to sign, and public key to prove who signed that document. The Bitcoin adopts the *Elliptic Curve Digital Signature Algorithm (ECDSA)* to perform signatures. It is a version based on elliptic curves. The difficulty of the logarithm does not allow third parties to sign a document without having the person private key. Thinking conversely, if it is impossible to forge the signature, then a valid signature cannot be refuted by the key owner. Usually, the process of signing a document is performed on its resume. An advantage of using these functions is that they always generate as output a few bits of the same size. The signature must be able to provide integrity, nonrepudiation, and authenticity.

In the Bitcoin, the private key is obtained by generating a random number with 256 bits length, a public key by performing the multiplication of the private key by one point in the curve known as "generator point." It is always the same for all users of Bitcoin and is defined in the specification secp256k1. The result of the multiplication of the private key by point generator is another point on the curve; this point is the public key. The nodes store only their private keys because they can generate the corresponding public key at any time.

From this point, the node already has a pair of keys that can generate the address. The address, not to be confused with IP address, is a number obtained using the public key. It is used to tell the system which is the owner of that transaction because only those who possess the private key that generated that address can unlock the transaction value. The node must perform a double hash to generate the address, first using SHA-256 after RIPEMD160.

Users of Bitcoin have keys that allow proving possession of transactions. These keys need to be stored, usually, in a digital wallet. The wallet has the function of generating the keys and stores them. There are two types of wallets: the deterministic and random. The deterministic wallets use one initial key, called a seed, to create the others through a hash function, and store only the first key, because all the others may be recalculated. The random must use an algorithm to generate random numbers with 256 bits. These numbers are the keys. This type of wallet needs to store all created keys.

2.4. Peer-to-Peer Network (P2P). The Blockchain network was developed to be a decentralized consensus network. A crucial point of Blockchain's mentality is the decentralization. So a P2P network best fits its mentality, where all the network participants are equal, there is not a central node, and all are burdened to keep the network running. All nodes interconnect in an overlay network.

A node can perform four functions: routing, database, mining, and wallet. A full node has all four functions, but all nodes have at least the routing function. A typical user, for example, that seeks only a payment way has only the wallet and routing. In this way, he can connect to a network and do transactions using a smartphone, without the need to store the entire chain of blocks.

To enter in the network is necessary to know at least one node. Each node can start until 8 *(Outbounds)* connections and accept up to 117 *(Inbounds)* connections. The core of Bitcoin has recorded a list of some nodes, known as *Seeders*, which has the objective of delivering a list of other active nodes in the network, so that the new node can establish initial connections. All connections are TCP. The node performs a *HandShake* to establish the initial connection. Once the node establishes one connection he sends a *GETADDR* message requesting a list of known neighbors IP addresses. Using this list, the node starts this process again to new neighbors, to become well connected. After the first time that the node is connected, it saves a list of all nodes that he has established connection recently on disk. So the node does not need the aid of *Seeders* on the next time he connects to the network.

To store the addresses the nodes use two tables: a table of successful connections, where the information of all connections made, inbound and outbound, is stored; and a table of addresses provided by others nodes, requested or not. The former is called *Tried Table* and the latter *New Table*.

(i) Tried Table: it is formed by 64 containers that can store 64 addresses each. The containers are selected as follows: when the node is started, it chooses a random value SK and calculates

$$\text{Cont} = \text{Hash}\left(\text{SK}, \text{Group}, \text{Hash}\left(\text{SK}, \text{IP}\right) \%4\right) \%64, \quad (1)$$

where the group is the /16 prefix of the IP address. When a node establishes a connection, it maps the IP address of the new neighbor to a container. If the

container is filled, the node then invokes a function to remove addresses from the container. We randomly chose four addresses and move the oldest to *New Table*.

(ii) New Table: it is formed by 256 containers and each one holds up to 64 addresses. It is populated by addresses removed from the *Tried Table,* addresses provided by *DNS Seeders,* or *ADDR* messages, which are messages to inform new addresses to neighbors. Similar to the *Tried Table*, there is a function to map containers and a function to remove old addresses from containers.

When a node needs to establish a new connection, he will choose an address from one of the two tables: *Tried or New*. For this, it uses the following formula, which gives the probability of choosing the *Tried*:

$$P_{\text{tried}} = \frac{\sqrt{\theta\,(9-\epsilon)}}{(\epsilon+1)+\sqrt{\theta\,(9-\epsilon)}}, \tag{2}$$

where θ is the ratio between the number of addresses stored in *Tried* on *New* and ϵ is the number of initiated connections.

Besides *ADDR* messages the protocol specifies messages to exchange data, which are used for transactions and blocks dissemination.

Some network nodes are simplified nodes, which have only routing and wallet functions. These nodes do not have a complete view of the network and need help from other nodes to do routine checks: for example, to receive a payment a node needs to know if the value received is valid. So, the protocol specifies that the full nodes can perform these checks and respond to simplified nodes. To do this, they provide an RPC *(Remote Procedure Call)* to help simplified nodes.

3. Blockchain

Blockchain's concept begins to make clear that it goes far beyond technological innovation. It is having a significant impact, primarily by shifting the business way centrally to a decentralized form, conferring trustworthiness on unreliable agents transactions, without the need for an intermediate entity trusted by both. Besides, it can change the way of realizing all transactions types and enable a wide range of possibilities in other areas, such as Multi-Party Computation (MPC) [17], use in Decentralized Autonomous Organizations (DAC) [18], and government applications [19].

It can divide its evolution into three stages [20]: Blockchain 1.0, 2.0, and 3.0. Blockchain 1.0 is the commercial use with money transfer, remittance, and digital payment systems, widely diffused by the use of Bitcoin and derivatives. Blockchain 2.0 is its use with contracts, the entire list of economic issues, market, and financial applications that use it in a more extensive way than simple cash transactions such as stocks, bonds, loans, mortgages, and smart contracts. Blockchain 3.0 refers to its use in applications beyond currency, finance, and markets, particularly in the areas of government, health, and science.

3.1. Blockchain Definition. Nakamoto [21] (original Bitcoin developers nickname) introduced Blockchain as a mechanism to ensure auditability, immutability, and nonrepudiation to provide security to electronic transactions, serving as a giant distributed ledger. This mechanism is the main innovation introduced by Bitcoin. It represents a way to reach consensus among unreliable participants. Usually, institutions like banks or notary offices are responsible for the guardianship and security of the transaction record; they are called trusted third parties. The system proposed by Nakamoto eliminates the necessity of these entities, since all the registries are, besides public, maintained in a decentralized way by several participants of the network. Figure 2 is a network simplified view, where can observe the main functions that each node can use. It is an overlay network. An overlay network is a network that is built on top of another network, creating layers of network abstraction providing new applications or security benefits.

In a simplified way, Blockchain is a data structure that stores transactions in an ordered way and linked to the previous block, serving as a distributed system of records. This structure is divided into two parts, header and transactions, and stores detailed information about the transactions it contains. So it can associate a transaction with its source and destination address. Each block has a unique ID generated from a cryptographic digest as explained in the previous section. The header has a field that stores the hash of the immediately preceding block so that we can establish a connection, a "link," between the blocks. For this reason, this structure was called Blockchain (see Figure 3). Another feature of this connection is that this hash is a partial collision, which will be explained in more detail below; this process requires a tremendous computational power to find the correct hash. As each block references its predecessor, if we change one bit of the previous block, its hash will change, and consequently, it will be necessary to recalculate the hash of all descending blocks. For this reason, it is assumed that the existence of a long chain of descendants makes the block immutable, ensuring the security of the stored transactions.

3.2. Block's Structure. The main parts of a block are the header and the transactions. Transactions are the data stored in the block. In turn, the header has several fields, of which the most important for its operation are hash of the previous block, difficulty, nonce, and the Merkle tree root. Besides these, it is also necessary to understand two metadata: block height and header hash, which are stored to identify the block and its position in the chain. These fields will be detailed below for Blockchain's correct understanding.

3.2.1. Block Header

(i) Height: the blocks are linearly included in the chain in chronological order, each new block receives an order number, the difference between the number of the last block and the first one is called height. This field is not always used to identify a block, as there may be momentarily two or more blocks with the same height. In this case, a fork occurs in the chain.

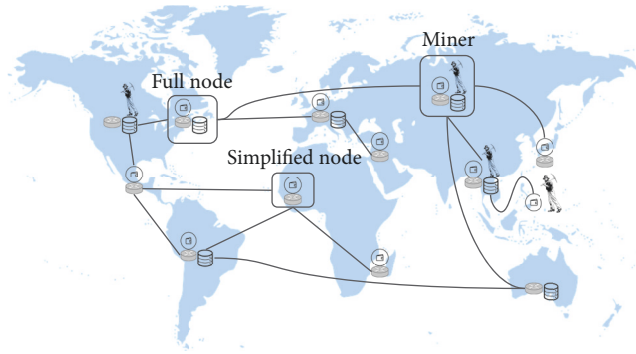


Figure 2: Bitcoin network overview.

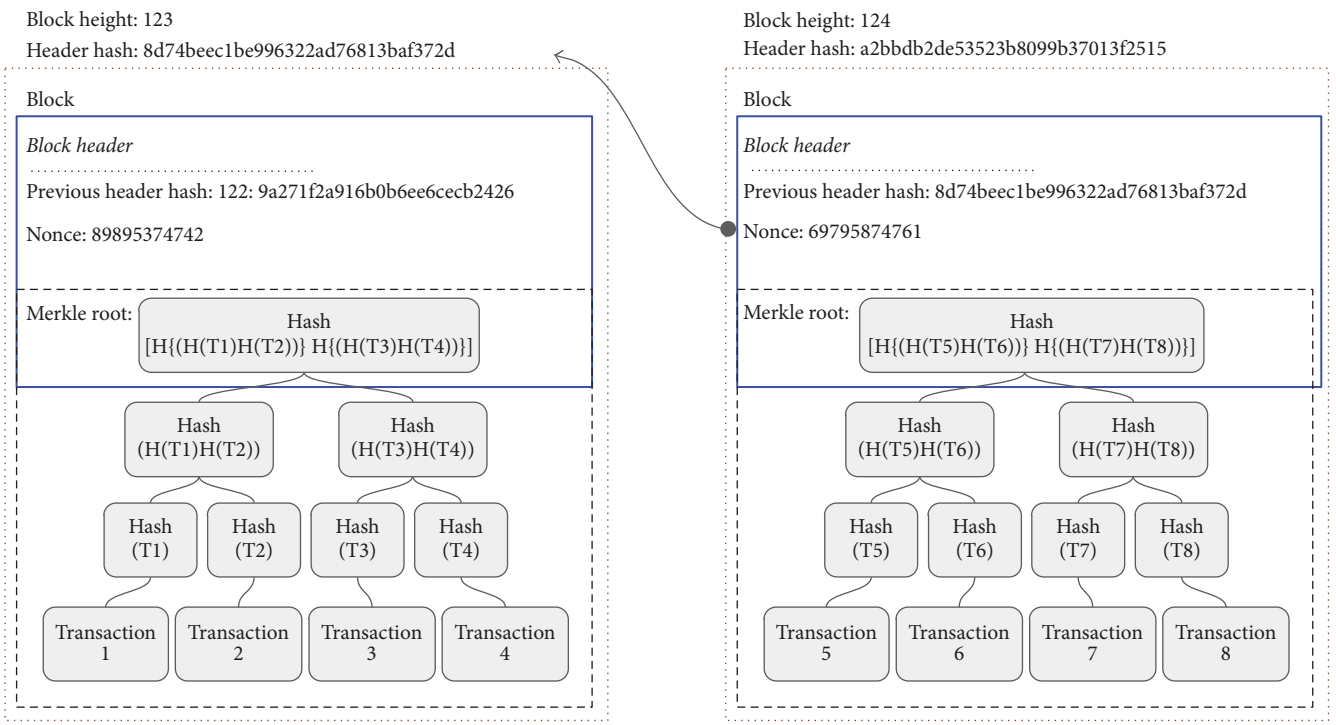


Figure 3: Simplified block structure.

(ii) Header hash: it is the principal block identifier. It is a cryptographic digest operation using the block header as input. It is not part of the block's data structure and is also not sent over the network. Each complete node computes it upon receipt of a new block. After that, they store it in a separate database as part of the block metadata. Unlike the height, the header hash can be used to identify a block unambiguously.

(iii) Hash of the previous block: this field is included in the header to allow the block connection with previous one. As we saw in Figure 3, block 236 has, in its header, the hash of block 235. The complete nodes store the block's metadata. Thus, all nodes have the hash of block 235, as soon as block 236 is received by a complete node, it will check this field and determine that block 236 is the child of 235.

(iv) Nonce: this is a number used as a variable to modify the header hash output. In conjunction with the difficulty field is used to prove that a miner has performed a work. If the difficulty imposes that the header hash starts with a sequence of three zeros, the miner will iterate the nonce until the header hash meets that requirement. Upon receipt of the new block, the complete nodes will calculate the header hash only once, to see if the nonce is valid.

(v) Difficulty: the difficulty is nothing more than a partial hash collision: that is, as previously described, a hash algorithm always generates the same digest for a given

input. If a bit is changed from this input, the resulting hash will be completely different. So it depends on the computational power of the mining node to find a hash that satisfies this partial collision. The mechanism used to generate the collision is the nonce. As it is a header field, the miner will change it until reaching the partial collision. When the difficulty is set to 1 bit (zero), it is sufficient to find a hash that starts with a zero and any value for the other 255 bits, that is, $2^2$55 possibilities, will be considered valid. If set to 2 bits, the possibilities will be reduced to 254 bits or $2^2$54, where 10 bits will be $2^2$46 possibilities and so on. It is possible to observe that reducing the possible space of values that satisfy the collision implies a higher difficulty in finding a hash that satisfies the difficulty. Therefore, more computation is required, or more time of mining, and higher expense with energy.

The method of adding new blocks to the chain is called mining, and the nodes that do the job of generating a new block are called a miner. The rate at which new blocks are included in the chain is defined by the developers of each Blockchain project. In the Bitcoin network a target of 10 minutes was established: that is, the difficulty is adjusted by all the complete nodes and miners so that, on average, every 10 minutes a new block is included in the chain. New miners are expected to join the network, and new, more powerful equipment is launched, so on average, the inclusion time of new blocks tends to decrease. To prevent new blocks from being included at intervals shorter than 10 min, the difficulty is adjusted by increasing the number of bits for the collision. Thus, as it will be harder to find the new hash, the inclusion time of new blocks will adjust until it is close to the 10-minute target. Each mining node independently recalculates the new difficulty every 2016 new blocks by performing the following mathematical calculation:

$$\text{New}_{\text{Diff}} = \text{Old}_{\text{Diff}} \times \frac{\text{Time } n \text{ Blocks}}{(\text{Time Target} \times n \text{ Blocks})}, \quad (3)$$

where New_{Diff} is the new difficulty calculated and Old_{Diff} is the old difficulty in the Bitcoin network.

(vi) Transactions: in Bitcoin, a transaction is a transfer of values. In a simplified way, it is a set of inputs (addresses from where the values will be taken) and outputs (addresses where the values will be sent). A node after creating a transaction sends it to all its neighbors. The nodes that received the transaction relay it to their neighbors and so on and so forth, so that the transaction reaches all the nodes of the network. When a miner receives the transaction, he will save it so that it is included in a next block that will be mined. When this block is included in the chain, the transaction becomes public and immutable. Transactions are signed with a public key system. To send a value to someone it needs to have the private key to sign the transaction, proving ownership of the value. It is also necessary to know the public key of the user that will receive the value, to encrypt the transaction so that only the holder of the private key, which matches the target public, will be able to decipher it. In this way, it is possible that the system is public and yet only whoever owns the transaction can use it.

There are two other types of transactions in the Bitcoin network, the smart contracts, which will be explained better throughout the section, and the data storage called *OP_RETURN*. The *OP_RETURN* is a custom transaction used to store 40 bytes. This is enough for a SHA-256 checksum (32 bytes) with 8 bytes of prefix or for a shortened URL. It is addressed in the same way as a financial transaction. Multiple use cases exist, like proving existence of some file; transferring other types of assets than monetary value; and colored coins, other coins on top of Bitcoin.

(vii) Merkle trees: a Merkle tree [22], or binary hash tree, is defined as a complete binary tree with a k-bit value associated with each tree node. The value of inner node is a one-way function of the values of its children. They are designed so that a leaf value can be checked against a publicly known root value by supplying the values of the corresponding pairs in the leaf path to the root.

In Blockchain, it is used to efficiently summarize transactions. Using it, is necessary to produce $2 * \log_2 N$ hashes, where N is the number of transactions. Therefore, it provides a very efficient process for checking whether a transaction is in a block. To build this tree, you must start with the leaves, which contain the transactions hash. As it is a full binary tree, in which all internal nodes have two children and all leaves are at same level, if there is an odd number of transactions to summarize, the last transaction hash will be duplicated to create an even number of leaf nodes. Then the leaves are grouped by two and their hash produces a parent node. The parent nodes are then grouped into pairs and experience the same process so that this process continues until there are no more pairs, thus generating a root node called the Merkle root, according to Figure 4.

To prove that a transaction is included in a block, we just provide the path that the transaction will go through in the tree. This path consists of the complementary nodes with the same height in the tree. This hash enables us to perform this scan quickly in the middle of thousands of transactions. This is particularly useful because to verify if a transaction is in a particular block, it is not necessary to request the entire block from the network, just the block header and the path to the transaction. As we saw earlier, a simplified node does not have the stored Blockchain. If this node needs to confirm a transaction, it needs the complete node help. For example, in Figure 4, each leaf corresponds to a transaction hash. The gray values correspond to the path to prove that this

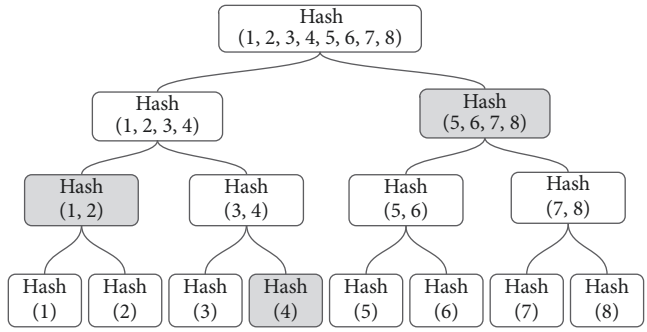


Figure 4: Merkle tree.

transaction is on the block. To prove that transaction 3 is on the block the complete node will send the block header and the *hash(4)*, *hash(12),* and *hash(5678)* to simplified node. With this data, it is possible to calculate the Merkle tree root and compare it with the Merkle root value on the block header. The simplified node will calculate the *hash(3)*, which together with the *hash(4)* will calculate the *hash(34)*. Take the value of the *hash(12)* and get the *hash(1234)* and finally use the *hash(5678)* to calculate the root whose value is *hash(12345678)*.

3.3. Mining. Mining is the process responsible for updating Blockchain, whereby some particular nodes, called miners, include transactions in a block and generate a valid header for those transactions. The miners spend much energy to perform the proof of work, which is why they need to be rewarded. The first transaction of the block is always a special transaction called Coinbase. It has two purposes, to include new coins in the system and reward the miner. In the Bitcoin network, mining has two purposes. First, include new currencies into the system and secondly protect the transactions made. To generate this heading, the miners must calculate the Merkle tree of the transactions, check the difficulty established, including the timestamp, and perform a series of calculations in order to find a nonce that satisfies the difficulty in force. This process will describe the importance of the difficulty and how it adjusts automatically, as well as showing a step-by-step process of the mining process.

Mining consists in generating a new block. For this, the miner first creates a "draft" of a block. It is in this draft that it will work until it gets a viable block to be sent to all nodes in the network. The draft is the data structure that will hold the header data and the transactions. After creating this blank structure, the miner fills in some header fields: hash from the previous block, timestamp, version, and difficulty. The miner then also calculates the root of the Merkle tree and the nonce and groups the transactions.

Transactions, when generated by a given node, flood the network, sending via broadcast to all neighboring nodes and these nodes forward to their neighbors, and so on and so forth. When miners receive a message with a transaction, they store these transactions in a database of transactions that have not yet been mined. Transactions remain temporarily in a sort of priority queue, based on fee taxes and arrival time, until they are removed to be included in a new block. Each miner has a different queue of transactions and can select which transactions it will include in that new block. After selecting which transactions to include, it will generate a Merkle tree and include the value of its root in the header.

Now it is missing the value of the nonce that will be part of the new block; this is the time-consuming stage of the process, requiring a tremendous computational power from the miners and consequently a considerable energy expenditure, as explained in the previous section. Currently, devices that specialize in calculating hash are marketed; these devices reach the 9TH/s mark: that is, they can calculate nine trillion hashes per second. To have an idea of the time to find a valid hash, with this equipment and the current Bitcoin network difficulty, it would take 13 years to find a valid hash.

For instance, to find the nonce that produces a valid hash for "Security and Communication" with the target difficulty of "000," the hash has to start with 12 bits zero in sequence. To make this possible, a nonce is concatenated with the message "Security and Communication," and a hash function is applied to it (e.g., sha256 "Security and Communication Network + nonce"). The nonce is incremented after each failure until a valid hash is found. An example result is as follows:

(i) Nonce: 1969

(ii) Hash: 000575dece1b23c16ebac44a9ed2a73eaded969 80c0d9d1292c4e0636776f917

In this example, the hash function applied of the message concatenated with the nonce 1969 generates a hash value that meets the target difficulty. It is important to note that there are other nonce values that generate valid results, such as 8715. From this point on, anyone with the same hash implementation can compute the hash of "Security and Communication Network +1969" and compare it with the provided hash, thus demonstrating that the result is valid.

The draft is complete when the nonce is found and therefore a new block is ready to be sent to all other nodes. They receive, validate, and then propagate the new block. Upon receiving a new block, all nodes initiate a series of checks to validate the block and to reach a consensus in the case of bifurcations ("forks"). As soon as the block is disseminated

in the network, each mining node adds it to its own chain, extending it to a new height. As the mining nodes receive and validate the block, they stop their efforts to find a block of the same height and immediately begin computing the next block.

We will see in Section 3.4 that the mining process is crucial to the consensus mechanism. Full nodes only accept new valid blocks, and the miners remove the validated transactions from temporary queue. In this way, a distributed mechanism for synchronizing the nodes is implemented.

3.4. Consensus and Proof of Work. The Blockchain does not have a central authority. Blocks are created independently by network miners. The nodes using information transmitted through insecure connections can reach the same conclusion and fabricate the same public record as all other nodes, achieving a global consensus. The complete nodes store the entire chain with the blocks that have been validated by it. When several nodes have the same blocks in their main chain, they are considered to have reached consensus. This subsection describes the validation rules of each block and how consensus is reached and maintained. We also explain some other consensus mechanisms that are currently used.

The consensus mechanism consists of two steps: block validation and the most extensive chain selection. These two steps are performed independently by each node. The blocks are broadcast on the network, and each node receiving a new block retransmits it to its neighbors. But, before this retransmission, the node performs a block validation to ensure that only valid blocks are propagated. There is an extensive checklist to follow including the following:

(i) Block structure

(ii) Verifying if the header hash meets the established difficulty

(iii) Block size within projected limits

(iv) Verification of all transactions

(v) Checking the timestamp

By definition of Blockchain, each block has only one parent, but there may be a situation where one or more miners generate new blocks almost at the same time, causing one or more children to have one parent. In this case, it is understood that a fork, a bifurcation, occurred in the chain. The last consensus mechanism step is to select which of these blocks will be part of the main chain and which will be discarded. This is possible because of the proof of work, which will be discussed in this section, fundamental to the consensus mechanism adopted because, as we saw earlier, to generate the block, miners spend much energy in search of a valid block.

Since it is possible for bifurcations to occur, the nodes store the blocks without a parent (orphan) (Rare and temporary situation) and maintain two chains, one main and one secondary. Orphan blocks occur when two blocks are generated in short time frames and arrive in reverse order: that is, a block has been received and does not refer to a block in the chain. It is stored for a period; if the node receives a block that is the parent of the orphan, it will be included in the chain in its correct order. Note that in this case there was no bifurcation; the blocks were only received out of order.

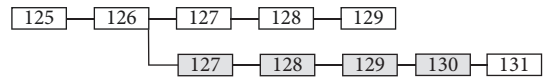


Figure 5: Fork.

As there are several miners generating blocks in a decentralized way, the new blocks sent by them can reach different nodes at different times, which can result in different views. When two miners generate blocks with reference to the same parent the fork occurs, and the other miners must choose which block they will adopt as a reference. If one part of the miners adopts one block and another part adopts the other, these two chains will coexist until one becomes larger than the other. To resolve this situation, nodes that behave honestly, according to the consensus mechanism, will always adopt the largest chain and the fork will be solved. The mainstream is the most extensive chain, the one where there is the highest amount of work accumulated. In Figure 5, the gray blocks branched out of the main chain; as they reached a higher height, they became the main chain. White blocks 127, 128, and 129 are discarded, and their transactions are considered unconfirmed and should be included in other blocks in the future.

One of the most common concerns for digital coin systems is the possibility of double spending when one malicious user spends the same value on two different transactions in the chain. Note that a bifurcation is necessary to cause a double expense attempt because if the expense occurs in the same chain when the new block is created, it will not pass in the initial checks of consistency and will be discarded. With the fork, the malicious user makes an expense and sends it to the network, spends the same amount again elsewhere, and starts mining on that expense. In this way, there is the possibility that he can mine a block and perform the fork. From this moment, the network will be divided, and as previously mentioned, there will be a race that will be won by the biggest chain. One of the transactions will be discarded, and the double spend will be rejected. As one of the strings will be accepted by the network and the other discarded, eventually the double expense will be detected. It is usually accepted in the Bitcoin network that a transaction is considered confirmed when there are six new blocks with a higher height than yours because it will take much effort to change it.

An attack scenario against the consensus mechanism is called the "51% attack." In this scenario, a group of miners, controlling a majority (51%) of the total hash power of the network, conspire to attack Bitcoin. With the ability to mine most blocks, attacking miners can spawn deliberate bifurcations in Blockchain, generate double-spend transactions, or perform denial of service attacks (DoS) against specific addresses or transactions. A bifurcation attack or double-spend attack is an attack where the attacker causes already confirmed blocks to be invalidated by bifurcating a level below them, with a later reconvergence in an alternate chain. With enough power, an attacker can invalidate six or more blocks in

a sequence, invalidating transactions that were previously considered immutable (with six acknowledgments). Note that double spending can only be done on the attacker's transactions, for which the attacker can produce a valid signature. Making a double spend of the transaction itself is profitable when, by invalidating a transaction, the attacker can receive an irreversible payment or product without having to pay for it.

Achieving consensus in a distributed system is a challenge. Consensus algorithms must be resilient to node failures, network partitioning, message delays, and messages that arrive out of order and corrupted. They also have to deal with selfishly and deliberately malicious nodes. Several algorithms have been proposed to solve this, each realizing the set of necessary assumptions regarding synchrony, message transmissions, failures, malicious nodes, performance, and security of the exchanged messages. For a Blockchain network, achieving consensus ensures that all nodes in the network agree on a consistent global state of the Blockchain.

According to [23, 24], a consensus protocol has three fundamental properties by which its applicability and effectiveness can be determined:

(i) Security: a consensus protocol is determined to be secure if all nodes produce the same result (agreement) and the results produced by the nodes are valid according to the protocol rules *(validity)*; this is also referred to as shared state consistency.

(ii) Liveliness: a consensus protocol guarantees the liveliness if all the nodes that follow the protocol, eventually, produce a value *(termination)*; that is, if a node generates a transaction and sends it to all nodes of the network at some point a miner will include it in one block.

(iii) Fault tolerance: ability to continue to operate and reach consensus, correctly, even after the failure of some network nodes.

The impossibility result of Fischer Lynch Paterson (FLP) states that a deterministic asynchronous consensus system can have at most two of these three properties. This is a proven result. Any consensus system distributed on the Internet should sacrifice one of these properties [25].

Most of the existing Blockchain platforms, more than 90% of the total market capitalization of digital currencies, use the consensus mechanism, in its original and computationally expensive form, which is proof of work. However, there are a number of other mechanisms that offer certain desired advantages over the original model: for example, the proof of stake (PoS) [26], practical Byzantine fault tolerance (PBFT) [27], and the proof of elapsed time (PoeT) [28] appear as other and will be briefly explained below:

(i) Proof of work: the main idea of proof of work is to try to avoid cyberattacks. A system is used where the user must prove that he has spent some time to find some answer that satisfies some requirement that the verifier asks for, to achieve that goal. The task of finding such an answer is based on two principles. Firstly, PoW has to be difficult and laborious, but not impossible; and secondly, the verification of that evidence should be much faster and easier to perform. This concept was first proposed by Back [29] and is used by several test systems and also by Bitcoin.

In Bitcoin, when a transaction is initiated, the transaction data is fitted into a block with a maximum capacity of 1 megabyte and then duplicated across multiple nodes called miners on the network. The miners verify the legitimacy of the transactions in each block. To carry out this verification, the miners need to solve a computational puzzle, known as the proof of work problem. The first miner to decrypt each block transaction problem gets rewarded with coin. Once a block of transactions has been verified, it is added to the Blockchain.

The PoW is generated as follows: the sender adds an arbitrary number to the message (called a nonce) and applies a mathematical hash function to the message. The SHA-256 [30] is used by Bitcoin. The goal is to find an answer with a number of advanced zeros that meets the network's current difficulty target (cf. Section 3.2.1, difficulty bullet point, and Section 3.3). He repeats the procedure by varying the nonce until he finds this answer. As it is relatively difficult to find such an answer, upon receiving the message, every user will be able to verify that there has been a great effort by the sender to generate it. When deciphering the problem, the miner generates a new block. The difficulty of the proof of work is adjusted every 2016 blocks, to generate on average one block every ten minutes. PoW's security is based on the principle that no entity should collect more than 50% of the network's processing power because that entity can effectively control the system by manipulating the longer chain.

(ii) Proof of stake (PoS): PoS is a category of consensus algorithms for public Blockchains that depend on a validator's economic stake in the network. Its concept states that a node can mine or validate block transactions according to how many coins it holds; this means that the more currency owned by a miner, the more mining power it has. PoS is a proposed alternative to replace the PoW that requires a great deal of computing power to run different cryptographic calculations to unlock its computational challenges. The PoS solve this issue by attributing mining power to the proportion of coins held by a miner. Thus, instead of utilizing energy to answer PoW puzzles, a PoS miner is limited to mining a percentage of transactions that is reflective of his or her ownership stake. The creator of the next block is chosen in a probabilistic way, and the chance of a node being chosen depends on its "wealth" (i.e., possession).

In PoS encryption, blocks are usually validated rather than mined, and it works in this way: the Blockchain keeps track of a set of validators, and anyone who

holds cryptocurrency can become a validator by sending a special type of transaction that locks up their cryptocurrency into a deposit. The process of creating and agreeing to new blocks is then done through a consensus algorithm that all current validators can participate in. There are many kinds of consensus algorithms implementation and one of this is the chain-based proof of stake. In chain-based proof of stake, the algorithm pseudo-randomly selects a validator during each time slot (e.g., every period of 10 seconds) and assigns that validator the right to create a single block, and this block must point to some previous block (normally the block at the end of the previously longest chain), and so over time most blocks converge into a single constantly growing chain.

Several different selection methods were planned. Nxt [31] and BlackCoin [32] use randomization to predict the next block generator, using a formula that looks for the lowest hash value in combination with the size of participation. Since bets are public, each node can predict, with reasonable accuracy, which account will gain the right to validate a block.

(iii) Practical byzantine fault tolerance (PBFT): the function of a consensus protocol is to maintain the order of transactions in a network of block strings, despite the threats to that order. One of these threats is the simultaneous arbitrary failure, one of Byzantine fault types, of multiple network nodes. Using PBFT, a network of Blockchain nodes can tolerate faulty nodes up to f, where f is a known arbitrary fraction of the total number of nodes, with a state machine replicated on different nodes (a replica being defined as primary). The PBFT algorithm works as follows:

(a) A client sends a service request to the primary machine.
(b) The primary replicates the request for the backups.
(c) Replicas execute the request and send responses.
(d) The client waits for $f+1$ identical responses from different replicas to consider a correct result.

As the total number of nodes needs to be known, the PBFT is not suitable for public systems and is only used in private systems. A PBFT network ensures data consistency and integrity when Byzantine failures occur in up to 1/3 of network nodes. For example, using PBFT, a Blockchain's network of nodes N can support f number of Byzantine nodes, where $f = (N - 1)/3$. In other words, PBFT ensures that a minimum of $2 * f + 1$ nodes reach consensus on the order of transactions before attaching them to the shared ledger. The rule $2 * f + 1$ has the following implications:

We need a minimum of $2 * f + 1$ nodes to reach a consensus before proceeding to the next block. The ledger on any additional node (beyond $2 * f + 1$) will be temporarily delayed. This delay in synchronization of the general ledger shared across all nodes is an unavoidable limitation on any PFBT network.

(vi) Proof of elapsed time (PoET): a consensus algorithm, designed by Intel. PoET uses a random election model of a leader, who will validate or mine the blocks. It essentially works as follows: there is a specialized hardware for generating a random time value. Each validator or miner requests a timeout for this hardware. The validator with the shortest waiting time for a given block is elected the leader and waits this given time to validate the block. After this, the block will be included in the chain and the process repeats itself.

This model is proposed for use in private Blockchains since in theory, the validators are honest. PoET uses these features to ensure the security and randomness of the leader election process, without requiring an expensive investment in energy; it occurs in PoW.

The PoET leader election algorithm meets the criteria for a good lottery algorithm and the probability of election is proportional to the resources contributed (for example, processing power). Randomness in generating waiting times ensures that the leader function is evenly distributed among all validators. The low cost of participation makes it feasible the participation of large numbers of validators, increasing the robustness of the consensus algorithm. A disadvantage of this algorithm is the specific hardware dependency.

3.5. Blockchain Categories Based on Data Access. Blockchain can be classified based on data access and participation of the consensus mechanism on any proposed changes in its ledger as follows:

(i) Permissionless Blockchain (public): the consensus mechanism is open to all. The purpose of a chain without permission is to allow anyone to contribute data. This creates the so-called censorship resistance, which means that no actor can prevent a transaction from being added to the chain. Participants maintain chain integrity by reaching consensus on their status. Anyone can join the network and participate in the block verification process to build consensus and also create smart contracts. Having a system without permission implies that there may be no trust between nodes, so a strongly distributed consensus mechanism must be enforced. In such a system, there is the possibility of a Sybil attack [33], where a network node tries to appear as several distinct nodes creating a large number of pseudo-identities. A disproportionately large influence by a single node is a threat, so the introduction of PoW in transaction validation is logically justified and necessary.

(ii) Permissioned Blockchain (private): participants in the consensus process are preselected. When a new record is added, the integrity of the ledger is verified by a consensus process conducted by a limited number of trusted actors; this makes keeping a shared

record much simpler than the consensus process without permission. Allowed Blockchain provides highly verifiable data sets because the consensus process creates a digital signature, which can be seen by all parties. The features that derive from reliable systems may open the possibility of avoiding a computationally demanding consensus protocol such as PoW.

Many projects were started to do Blockchain more popular and viable for different business models and applications, leveraging existing categories. Table 1 summarizes the key features of some Blockchain-based applications. Bitcoin and Ethereum [34] are examples of Blockchain permissionless and Hyperledger [35] and Ripple [36] are examples of permissioned Blockchain. It is possible to check a critical difference between these two categories which is the underlying mining model. Blockchains permissionless use the PoW where the power of hashing is offered to create trust. Permissioned Blockchains do not need to use computational energy-based mining to reach consensus. Since all actors are known, they end up using consensus algorithms like PBFT that can be used to achieve consensus without PoW mining, leading to a block processing time much lower compared to Blockchain's time permissionless, being practically considered realized in real-time.

4. Cases of Use for Providing Security and Privacy at IoT Using Blockchain

The devices in the IoT collect, generate, and process data and send this information via the Internet, producing a considerable mass of information to be used by various services. Despite the benefits, critical issues related to privacy may emerge. The Blockchain can play a crucial role in the development of decentralized applications that will run into billions of devices. Understand how and when this technology can be used to provide security and privacy is a challenge, and several authors point out these problems [37–39]. The authors have been discussing the applicability of connecting Blockchain and IoT, specifically regarding the following issues:

(i) Typical IoT devices have limited capabilities.

(ii) Transaction costs might inhibit interactions.

(iii) IoT endpoints are often sleepy.

(iv) IoT generated information might need to be kept private.

Therefore, there is need for investigating when both technologies can be applied appropriately. In that sense, the literature [9, 37, 39–43] has been addressing the following:

(i) A cost-effective Blockchain that fits low-capability devices

(ii) Micropayments between sensors for paying for data

(iii) Computation and knowledge extraction from sensitive data

(iv) Integration on smart homes, smart cities, or enabling shared economy

All of the above discussion is about applicability and solutions for connecting Blockchain and IoT. In this way, it becomes necessary to know the main weaknesses to which Blockchain is exposed and to keep it in mind when developing new applications.

In this section, we will explore how Blockchain can be used to benefit security applications for the IoT. Such as decentralized applications which enable the smart objects to interact with security, establish payments mechanisms [44], create public key infrastructure (PKI) services [45, 46], perform Multiple Secure Computation (MPC) [17], support Smart Ambient [43], and provide privacy in storage systems [47].

Also, we will describe how the block propagation latency and the block rate [48] may influence the safety of the consensus mechanism and present the most common attacks discussed in the literature, such as selfish miner [49, 50]; double-spend [48]; and Eclipse [51]. Finally, we will introduce the stalker attacks.

4.1. Use of Blockchain to Provide Anonymity and Access Control to IoT. Providing privacy remains a challenge for IoT, since "things" spread sensitive personal data and reveal the behavior and preferences of their owners. Developing IoT applications that use an existing and stable Blockchain is one of the proposals [37, 52], in which PoW and a large number of honest miners would guarantee integrity and privacy.

Firstly, it is worth mentioning that the anonymity provided by the use of Blockchain is not absolute, so it is commonly called pseudo-anonymity. It is possible, in certain circumstances, to deanonymize the transaction owner or its IP address. To deanonymize transactions there are some specific techniques, according to [37]which can be divided into four types:

(i) Multiple entries: in some cases to realize a certain transaction is necessary to gather balance from various accounts. In other cases, it is needed to save the total wallet balance in a single account. It is possible to carry out the transfer of lowers balance to a single account; this procedure is called multiple entries transaction; to accomplish this transaction, it is necessary to have the private keys of each input. So, we can assume that all accounts belong to the same user. From this moment we can associate the addresses to a user. This approach was used in [53–55].

(ii) Change address: by protocol definition, it is mandatory to spend all balance associated with a given key. If the value of the transaction is less than the balance assigned to the key, this transaction will generate change. The change value has to return to the owner. This is done by indicating the change as an output to himself. If a node always uses the same address to receive the change, we can associate this address with input addresses and describe exactly all user's expenditures. Also, it is possible to correlate with secondary sources of information such as social networking sites. These are the approach used in [53–55], to deanonymize transaction and users.

Table 1: Comparison between Blockchain systems.

Blockchain	Bitcoin	Ethereum	Hyperledger	Ripple
Nature	Permissionless	Permissionless	Permissioned	Permissioned
Validation	PoW SHA-256	Ethash PoW	PBFT	BFT customized (RPCA)
Purpose	cryptocurrency	Smart contract	Chaincode	Cryptocurrency
Language	Stack based scripts	Internal code Turing complete	Go, Java	C++
Block processing time	~600 s	~15 s	~Real time	~Real time

(iii) IP association: the Bitcoin is an overlay network on the Internet. Most network's messages are transmitted in *BROADCAST* to direct neighbors of each node. Many neighbors allow a node to extract some network's knowledge, as its topology, which are the nodes miners, node's location, and their IP address. In [56], the author listens to network traffic and uses a clustering algorithm and was capable of associating the IP address with the user.

(iv) Use of centralized services: users, for various reasons, do not save and manage their private keys, delegating this function to outsourced services. Some authors [54, 57] think this is a privacy risk. These outsourced services can leak identities or resources. Even more, they can use the resources of all user's balances.

According to [37], extra care is needed to mitigate these problems. The IoT devices must always be configured to use a different address to receive change, always generate a new address for each receiving resources, and do not use outsourced services. These measures are not sufficient to provide total anonymity but will give a degree of security to preserve identities.

We can also use the Blockchain in data storage and to provide access control. Suppose that a presence sensor wants to save daily history in the Blockchain. It will generate a transaction with the data to be stored and will sign this transaction, so everyone will know which sensor produced this data. The sensor will indicate as transaction output the public keys with the right to data read. It sends this transaction to network miners, which authenticate and include it in the next block. As the Blockchain is public, all users have access to transactions and know that a particular user has the right to read the history produced by the presence sensor. However, only those who have the private keys will be able to read the daily history which was released by the sensor.

Ouaddah et al. [52] proposed the FairAccess, a *framework*, which uses the Blockchain to enable users to control their data. He reuses the code of Bitcoin and introduces some new types of transaction used to provide data access control, such as *"grant" and "revoke"* access. The model has some actors: the shared resource; the resource owner; and the users. The transactions are used to provide access control, and the Blockchain uses it for storing and reading the permissions. The authors did a proof of concept with a *Raspberry Pi* and a camera ("the resource"). The owner controls resource access through transactions. So, to grant access to a user, it makes a *grant access* transaction specifying a user who has the right to access the camera, as if he were selling a product using Bitcoin. One miner will include this transaction in the Blockchain. From this point, the user will directly access the resource, so it will verify in the Blockchain if there is a transaction that ensures his access, in which case the user will be able to use the camera.

One of the main criticisms to storage in the Blockchain is the use of data structures that were not designed to store large amounts of information. Thus, if we use the block for this purpose, we will get several copies of the same file in the network. To use the security provided by Blockchain, Zyskind et al. [47] combined the use of data storage outside of the chain with the access control in the chain of blocks. The storage uses a DHT (distributed hash table), where there are a set of nodes, selected beforehand, responsible for maintaining it. The data is replicated efficiently to ensure high availability. No node has the entire file. The Blockchain is then used to manage where these data is, and who has access to them. For this reason, two new types of transaction are generated, one to provide access control and another to control the data distribution in the DHT.

As the Blockchain has no central point of failure and is not governed by a single entity, it enables a new class of applications and decentralized services, for example, a DNS root server or an enterprise root certification authority. These benefits have motivated Ali et al. [46] to use the Blockchain to build a new decentralized PKI and an identity system, called *Blockstack ID*. *The Blockstack* decouples the name record and property from the availability of associated data, separating the control and data. The control plane defines a protocol for name registration, creating links (name, hash). The control plane consists of a block and a layer logically separated from the control plane, being responsible for the storage. All data stored shall be signed by the name owner key.

4.2. Use of Blockchain on Economic Scenarios to Ensure Electronic Transactions in IoT. The IoT future is to become a network of autonomous devices that can interact with each other and with their environment, making intelligent decisions without human interaction. In this place, the Blockchain can help leverage the IoT and form a foundation that will support the shared economy, based on machine-to-machine (M2M) communications.

There is a vast set of proposals, prototypes, and proofs of concept which pointed out how IoT can take advantage

of the Blockchain qualities and use it to trade goods and data [2, 41, 42, 58–61]. Blockchain technology can provide a way to track the unique history of each device, recording data exchange. It can also allow intelligent devices to become independent agents that autonomously conduct a variety of transactions. Applications for IoT can use Blockchain benefits: reliable, fast, and without intermediaries transactions; absence of single point of failure; trust in the predefined rules execution; and transparency and immutability. Sun et al. [58] say that Blockchain will support all transactions processing and coordination between devices. Each device will manage its roles and behaviors, resulting in the Internet of Decentralized, Autonomous Things. In [42], the authors described a prototypical implementation of data exchange by electronic money, between a sensor and a client, using the Bitcoin network. The system is composed of three parts:

(i) IoT device: it needs to fulfill the following tasks: write a data request when receiving payment, it can create and publish a transaction containing the requested data.

(ii) Client: it needs to be able to send payment to the sensor and must monitor changes in the Blockchain to detect the transaction with the data sent by the device IoT.

(iii) IoT device repository: it is a local where sensors are registered and may be found by clients. An entry in the sensors repository must contain at least the sensor address, what data he offers, the price, and additional metadata like the location.

In [2], the authors propose an architecture for electronic commerce explicitly designed for IoT devices, based on the Bitcoin protocol. Distributed Autonomous Corporations (DAC) was used as a transaction entity to deal with data from IoT devices. In this model, the users can negotiate with DACs, using cryptocurrencies.

As shown in Figure 6, there are four proposed layers for the IoT e-commerce model, which are basic technical layer, infrastructure layer, content layer, and exchange layer. The basic technical layer includes the module of the goods classification mechanism, the credit algorithm module to manage the portfolios, and the Blockchain Bitcoin module, which was the cryptocurrency adopted by the project. The infrastructure layer contains the IoT information service platform and the smart contracts platform. The content layer includes two parts: participant entities and IoT commodities. Entities consist of DACs and human beings. DACs run automatically without human interference, and each DAC can buy products from other DACs as customers; meanwhile, everyone can issue their own IoT commodities. Commodities are smart properties and data collected from sensors. The smart properties can be works of art, durable goods such as cars, homes, and energy as electricity, water, gas, and oil that can be controlled and quantified by digital devices via electronic keys or access control systems. The exchange layer includes the P2P transaction system that is at the core of IoT's business model along with the chosen cryptocurrency that is Bitcoin.

Some proposals addressed the use of Blockchain for the functionality of economic transactions for IoT, including the following:

(i) ADEPT [59]: automated decentralized P2P telemetry is a decentralized IoT system created by a partnership between IBM and Samsung that uses elements from Bitcoin to build a network of distributed devices, allowing billions of devices to transmit transactions to each other and perform self-maintenance, providing secure identification and authentication. The ADEPT uses the Blockchain to provide the system backbone, using a mix of proof of work and proof of stake for secure transactions. This platform was tested in several scenarios, including one that involves a smart washing machine that can automatically buy and pay for detergent with Bitcoin or Ether and can negotiate the best price of cleaning products based on the owner preferences. This washing machine uses smart contracts to issue commands to a detergent reseller when it needs supplies. These contracts provide the device the ability to pay for their own order and receiving a message from the dealer that the soap was paid and sent; then the washer owner's smartphone receives this information.

(ii) Filament [41]: it is a system designed to allow devices have unique identities and can discover, communicate, and interact autonomously with each other. Also, the devices involved can directly exchange value. For example, they could sell data about environmental conditions for a forecasting agency. The goal is to create a directory of smart devices that allow the IoT Filament devices to communicate securely, send microtransactions, and execute smart contracts. The Filament uses five technologies: blockname; TeleHash; smart contracts; Pennybank; and BitTorrent. Devices can create a unique identifier that is stored in a built-in chip and recorded in the block. The TeleHash, in turn, provides encrypted communications from end-to-end devices, and BitTorrent allows the file share. The smart contracts are responsible for dealing with the payments for the devices use. The Filament uses a protocol based on the Bitcoin for microtransactions, called Pennybank. Due to specific restrictions of IoT devices, the Pennybank creates a warranty service between two devices IoT, allowing them to settle transactions when they are online.

(iii) Watson IoT platform [60]: this platform from IBM allows IoT devices to push data into a private Blockchain. All business partners, who have this Blockchain, can access and provide the device's data without a central management. Each transaction can be checked, avoiding disputes and ensuring that each partner is responsible for their roles in the global transaction. They provide a Blockchain network infrastructure that replicates the data to the device and validates the transaction through smart contract insurance. The Watson platform offers APIs that translates the device's data into the contract format.

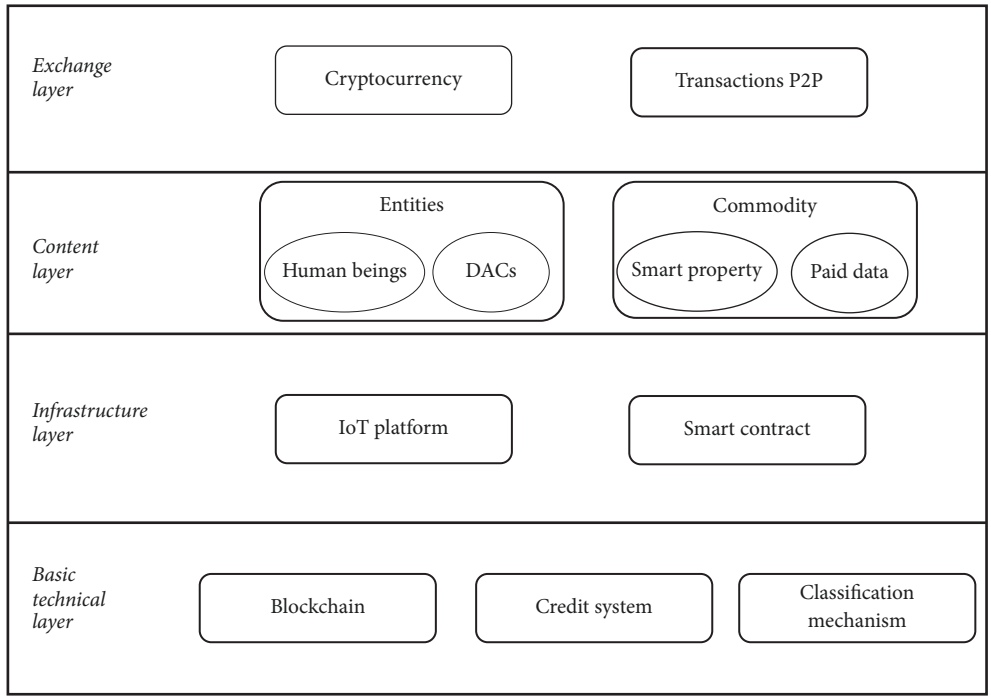


Figure 6: Business model for IoT using Blockchain. Adapted from [2].

(iv) IOTA [61]: it is a cryptocurrency explicitly developed for the selling of data from devices IoT. Instead of using a global Blockchain, the IOTA uses a DAG (Directed Acyclic Graph), the edges are the transactions, and the weights the number of times were confirmed. The main idea is that a node must first execute a series of transaction checks to approve them and only then carry out a transaction. There is no differentiation between nodes. All of them are responsible for approving the transactions. According to the author, this ensures a higher scalability: the higher the number of transactions, the more efficient it becomes.

In the IoT platforms that use Blockchain, there are some different proposals regarding their design. ADEPT [59] is an open-source framework and utilizes proven technologies, like BitTorrent, TeleHash, and Ethereum, which facilitates market adoption. But, it is still a proof of concept with several challenges to overcome, including scalability and the nature of cryptocurrency development. Filament [41] focuses on the industrial infrastructure, to make it smart and connected. Its main feature is the adoption of a secure element on each device, with a host set of keys that get burned into a write-once or one-time-programmable (OTP) memory. Thus, the Filament IoT device is naturally more expensive due to its secure tamper-proof capabilities. Watson [60] is a "Blockhain as a service" product. It has an API to provide its services for to IoT devices, but it works within a cloud infrastructure. The main advantage is the ability to provide the use of Blockchain for heterogeneous devices. Iota [61] has a huge disadvantage since it does not support smart contracts. Since there is no nodes differentiation, all of them have the burden of transaction validation. In this system, in order to perform a transaction, a node has to validate at least two other transactions and with the network growth, the system is expected to provide good scalability.

There are some other use cases involving data monetization with Blockchain and IoT devices. Nasdaq and Chain of Things lead the research on applications that can help make renewable sources of energy available to the general public, where the energy produced by solar IoT panels generates cryptocurrency registered in the Blockchain. So, anyone who joins the network can make investments in renewable energy technology.

4.3. Use of Blockchain in Secure Multiparty Computation. Consider the following problem: two millionaires interested in knowing which of them has the largest fortune without revealing their own to another or to third parties. This is the famous millionaire's problem proposed by Yao [62], which uses a protocol for secure two-party computation to solve it. The MPC is the generalization of this solution for multiple participants. We can define it as the problem of N participants to calculate a function with private entries in a safe manner, where security means ensuring the correctness and privacy of entries, even with the presence of some malicious participants. In the end, each participant will get only the result function and will not be able to know the entries of other participants. It opens the way for a variety of applications like Internet vote, data mining, and data sharing.

Starting from the principle that with additive and multiplicative circuits we can perform any function, we merely

need building these MPC blocks and then use these blocks for any other arithmetic functions. Thus, the proposed protocols for MPC seek to accomplish these two main functions, usually using Yao's circuits [62] or Shamir secret sharing [63] or its variants. The participants exchange messages to perform these functions on the additives circuits. The number of this message grows linearly with the number of participants, but on the multiplicative circuits, $O(n^2)$ communications are needed. This fact makes the MPC implementation restricted to few participants and specific scenarios. Over the years there have been proposals to optimize the solutions and increase the number of participants [17, 64, 65]. In the problems formulation, two types of protocols are commonly adopted: the semi-honest and malicious models.

(i) Semi-honest model: parties follow the protocol correctly but record all intermediate computations steps for later analysis, to achieve and infer other party secret information.

(ii) Malicious model: in this model, the malicious participant does not need to follow the protocol and may act arbitrarily; it may execute or abort the execution at any moment, using false information and storing the intermediate steps for further analysis.

Enigma [17] is a platform for MPC with privacy guarantee. It uses the Blockchain as a network controller, managing access control, and serving as log event to secret sharing. It can compute functions in both models and is scalable. Each node receives and records his inputs using Blockchain. There exist groups for each task, so each parcel performs a job, and at the end joining it. This partitioning allows a greater data replication control, improving the system scalability and allowing a more substantial number of participants.

Other work [66] uses the Blockchain to perform access control and storage of patient data. The author believes that the use of the data of patients without their consent is a privacy problem but also describes the importance of the use of these data for medical research. He sorts the data into two types: public and private. Any researcher or governmental entity may use the public data. To use the private data they have to do it via MPC. Thus, the use of MCP makes it possible to know, for example, the number of patients who have AIDS and belong to risk group. This makes it possible to extract data knowledge without revealing the patient privacy.

Chakravorty et al. [67] drew the attention to provide assistive services to older adults through data analytic technologies. However, the received data from smart homes represent personal and sensitive information and can often disclose the complete living behavior. Ideally, analysis of encrypted data would be a perfect solution for preserving privacy. However homomorphic encryption [68] scheme has computation and storage overhead and has to be carefully evaluated. It becomes necessary to devise a system that would allow execution of analytic data algorithms while preserving the privacy of monitored individuals. One of its possible solutions for using IoT devices that deal with sensitive data is the scheme like *Enigma* [47]. It uses the Blockchain to perform computation and extract knowledge from sensitive data generated without revealing it.

4.4. Use of Blockchain to Ensure Safety in Smart Home. Approaches based on Blockchain offer decentralized security and privacy but involve excessive consumption of energy and delays, which are not suitable for most IoT devices with limited resources. Dorri et al. [38, 43] offer a lightweight Blockchain solution IoT. This work proposes a method to adopt Blockchain in the context of IoT, eliminating the proof of work and the currencies mentality.

The author uses it to exemplify a smart home implementation, consisting of three main structures: cloud storage, an overlay layer, and smart home. Each smart home is equipped with higher power computer that is always online. This device is a type of "Miner" and is responsible for dealing with all communications inside and outside the house. This computer maintains a private Blockchain, which is used to control and audit the communications and provide access control between devices. All the IoT devices are in the smart home layer, which are managed by a miner. In this scenario, the PoW becomes unnecessary, because only one device will have the job of keeping the Blockchain. The others house devices receive a key pair so that they can perform transactions. As an example, if a presence sensor wants to turn the lamp on it will send a transaction to the lamp, which will check into Blockchain if that sensor is allowed to light it.

The overlay network consists of the smart home layer along with Service Providers (SP), cloud storages, and smartphones. The overlay network is grouped into clusters to minimize latency and each cluster elects its cluster head (CH). The miners maintain all the transactions in an immutable ledger which is the private Blockchain for each smart home network. There are different kinds of transactions like store, access, monitor, genesis, and remove which handle different operations and data sharing in the network.

This work mostly focuses on data store and access use cases, by IoT devices. The transactions in the Blockchain are data storage and access transactions. The public keys are fixed with the cluster heads and are immutable. In their security analysis, they analyze their model for DDOS attack and linking attack. They also measure the overhead for using their model over traditional message exchange.

A smart home is an excellent example of how to combine IoT and Blockchain. Blockchain-based sharing services can evolve and contribute to smart cities and shared economies. Shared economy is an economic-social model in which diverse population sectors can share underutilized assets [58]. Citizens, objects, and assets would connect transparently to exchange assets and status share. In this paradigm, people seek trust, access rather than ownership, the reliability of shared services, security, and privacy.

4.5. Attacks on Blockchain. Beyond the natural protection of stored data and typical attacks on distributed systems, Blockchain needs specific security mechanisms. Blockchain, by itself, can be considered secure and guarantees the block integrity and availability. But, the rest of the process before transaction validation or even a block (if there are attacks that fork the chain) is not safe in a natural way. Blockchain is not itself capable of detecting fraudulent activity.

Any system or network can suffer an attack; Blockchain-based systems are no different. The types of attacks that makes Blockchain vulnerable are a bit different: in most cases, we can perform attacks on the consensus mechanism to change the chain's history, prevent blocks or transactions from including the chain, or obtain greater revenue. The most common attacks on the consensus mechanism are the 51% attack and selfish mining attacks. In this way, it becomes necessary to know the main weaknesses to which Blockchain is exposed.

As we argue in Sections 4.1, 4.2, 4.3, and 4.4, there are much research of how to use Blockchain in conjunction with IoT. This fact arouses the interest in attacks on both technologies. By this way, it is important to know which IoT solutions will be affected, so we can exhaustively test the applications with a safe development process to mitigate potential vulnerabilities.

In the Bitcoin world, transactions are considered valid when they are in a block and confirmed when there are some blocks with higher height in the chain. The accumulated PoW in the chain does not permit us to change that transaction without a substantial computational power. However, bifurcations may arise. We choose the most extended chain to revolve the bifurcations. The majority bifurcations occur naturally, with no evil intention, causing delay to validating discarded transactions. This approach works well, under the crucial premise that no attacker must be able to gather as much computational power that can forge and publish a "chain" which has higher accumulated difficulty. In this case, the consensus rules do adopt the alternative chain instead of the main, from the point of bifurcation. This is theoretically possible and is called 51% attack [69]. As widely discussed, up to now the security of the Bitcoin depends on the consensus reached by distributed proof of work. We assumed that there is no single miner, nor a coordinated group of miners, nor a collusion of miners that has more than 50% of network computing power. However, this assumption is questionable. First, the miners began to be organized in groups, called mining pool. They join forces and share the rewards. Secondly, there is no regulatory entity, and neither miner is required to follow the protocol. A mining pool with a majority computational power can change the consensus. By doing this, the miners who do not participate in the cooperative will probably be forced to join it when their revenues start to fall. For example, a cooperative with more than 50% of computational power could choose to accept blocks of other miners in a ratio of 2 : 1; from every two blocks sent by honest nodes only one will be accepted, and this is possible because the mining pool shall have the power to manipulate the consensus. The honest miners will have their blocks ignored and, therefore, lose the payments. The mining pool behavior can perform a denial of service to any miner or any transaction. Because they have the power not to include these transactions in any blocks, and if other miners do, they can generate forks, thus rejecting a transaction.

The malicious miners can divert their behavior not to disclose immediately newly mined blocks. This attack is called "selfish mining" [4, 50]. First, it is needed to understand how the blocks propagation latency and the time target for the inclusion of new blocks affect the consensus mechanism.

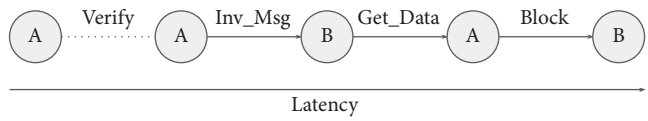


Figure 7: The latency of block spread.

Table 2: Impact of block interval on fork rate.

Blocks interval	Fork rate (%)	Median propagation time
0,5 s	38,15	0,82
1 s	26,74	0,82
2 s	16,65	0,84
5 s	8,64	0,89
10 s	4,77	1
20 s	3,2	1,21
30 s	2,54	1,43
1 m	2,15	2,08
2,5 m	1,82	4,18
10 m	1,51	14,7
25 m	1,72	35,73

(i) Analysis of the latency of block diffusion: upon receiving a new block, a node, transmits it to its neighbors. Before starting the transmission, he makes extensive checks to ensure the propagation has only valid blocks. Each node that receives a new block makes these verifications. After that, the node sends an inventory *(INV)* message to informing their neighbors who have a new block and its height. In case the neighbors do not have this block, they will respond with a request message *(GET_DATA)*. Only then the transmission of new block will start (Figure 7). The sum of all checking and spreading times, during the spread of a block, is the latency.

Decker and Wattenhofer [70] made a time analysis of 10,000 blocks with different sizes. The author found that the median latency time was 6.5 seconds and the average was 12.6 seconds. Another interesting observation is that after 40 seconds, 5% of nodes still had not received the new block.

(ii) Analysis of time for new blocks inclusion: the interval for the inclusion of new blocks is crucial for the number of *forks* observed in the network. The smaller this interval, the greater the number of blocks generated and consequently the greater the probability of forks occurrence and orphans blocks. Decker and Wattenhofer [70] observed an occurrence of 169 forks at 10,000 blocks in the Bitcoin chain, that is, 1.69% of discarded blocks. Gervais et al. [48] examined the time reduction impact; they varied the time for new blocks inclusion from 0.5 seconds to 25 minutes, observing 10,000 blocks in NS-3 simulations, as shown in Table 2.

If an attacker decides to deviate from standard behavior and keep mining in a secret chain, he needs to adopt

a heuristic for choosing the best moment to unveil these blocks. At the moment, everyone, including the attacker, is mining on the block n. When performing this action, if the attacker is able to produce the next block, he has an advantage, even without having power majority, because he can start next mining process (block $n + 1$) before everyone. As he began before, there is a high probability of releasing the block $n + 1$ before the other miner, generating a fork with higher height than the main chain. As the remaining nodes behave honestly, they will also adopt this chain and the attacker will reach his goal. Otherwise, if he receives a block, he may decide to adopt this block and throw away his work or ignore the received block and continued mining in the private chain. Figure 8 shows a simple fork scheme, where after releasing the blocks n and $n + 1$ by the attacker the honest nodes have embraced this chain and produced the block $n + 2$. This attack is known as *selfish mining* [4], and the attacker is called "selfish miner." Being more specific, the attacker has four stats. Suppose the attacker's portion of the network hash power is α, β are the honest nodes mining on top of the public chain, and ω is the portion of the network that picks up on the attacker's chain.

(i) State 0: if the attacker's private chain and the public chain have the same height, the attacker mine on the private chain. With probability α, the attacker finds a new block and go to state 1 (private chain one block ahead). With probability $1 - \alpha$, the public network discovers a block, and the attacker resets his private chain to the public chain.

(ii) State 1: if the attacker's private chain is one block longer than the public chain, mine on the private chain. With probability α, the attacker advances to state 2 (private chain two blocks ahead). With probability $1 - \alpha$, the public network discovers a block, setting the system to state $0'$.

(iii) State $0'$: the attacker unveils his chain. There are now two competing chains, both one block long. With probability α, the attacker will discover another block, converging the network to private chain. The attacker gains a revenue of 2, and the system resets to state 0. With probability $(1-\alpha)(\omega)$, the network finds a block on top of the attacker's block. The attacker and the network gain a revenue of 1, and the system resets to state 0. With probability $(1 - \alpha)(\beta)$, the honest finds a block on top of public chain, the network gains a revenue of 2, and the system resets to state 0.

(iv) State 2: with probability α, the attacker advances to state 3 and earns a revenue of 1. With probability $1-\alpha$, the network finds a block, so the attacker publishes his 2-block private chain, which is still one block longer than the public chain so that the network will switch to the attacker's chain. The attacker earns a revenue of 2.

(v) State n ($n > 2$): with probability α, the attacker advances to state $n + 1$ and earns a revenue of 1. With probability $1 - \alpha$, the attacker falls back to state $n - 1$.

A large part of latency time is due to the block checking obligation by every node. If an attacker controls some nodes, he can amplify the selfish miner attack. The slave nodes can be configured not to undertake blocks verification mined by the attacker and retransmit them as soon as they arrive. So the attacker's blocks latency will be shorter than honest nodes ones; this can be an advantage. The attacker could make the honest nodes work for him by unveiling his blocks as soon as he receives a new block. By this way part of honest nodes, who have received the attacker's block, will work on it, augmenting the attacker's power. Another observation about this attack is that the total blocks added to the chain are the produced blocks sum by the honest and the attacker. However, the occurrence of forks generates dropped blocks, *stale blocks*. Thus, the number of blocks included is smaller than the total produced blocks, this is important because, ignoring new miners entrance, when nodes recalculate the new difficulty, this will be smaller than the previous difficulty.

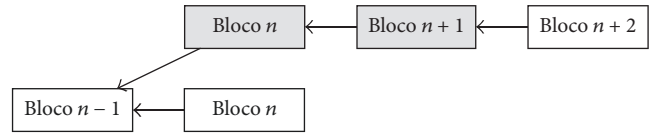


Figure 8: Fork.

In [4], Eyal and Sirer make a mathematical analysis and propose a model state transition with the aim of figuring out the best moment of release attacker's blocks. They analyze the occurrence probability of each state and conclude that, for the attacker to achieve success, that is, publish more blocks than the honest ones, attacker's mining power must satisfy the following:

$$\frac{1-\gamma}{3-2\gamma} < \alpha < \frac{1}{2}, \tag{4}$$

where α is the attacker's mining power and γ is the ratio of honest nodes mining in the attacker's chain. This is an attacker advantage because he will need less power to be able to supplant the honest nodes. So, from this, the lowest value of α is obtained and in the worst case when no honest node adopts its chain, it is necessary that the attacker has one-third of the network mining power.

Nayak et al. [50] expand the Eyal research and verify that the proposed attack earlier is not optimal. He proposes new strategies to increase the attacker's revenue, taking in account, not only the size of the chains, but also its computational power. For example, even if the attacker is losing the race if he possesses a significant mining power, it is better to continue mining in a private chain, because it will have a great chance to reach and exceed the honest chain. Another work contribution shows that if the *selfish mining* is combined with the *Eclipse* [51], when the attacker controls all connections to a given node, the attacker will increase your winnings and surprisingly, with certain parameters, the eclipsed node will also be able to publish more blocks, in relation to honest nodes.

The author uses a Markov Decision Process which uses the state transitions information to discover the best moment to unveil his blocks. When an honest node mines a new block,

it publishes it immediately, while the attacker maintains the blocks hidden. Then, the attacker identifies how many blocks are in each chain, which chain is leading, and how many honest nodes are mining on the honest or malicious chain. Their strategies, called *Stubborn Mining*, are as follows:

(i) Lead stubborn: as seen before, γ is an important factor, because of the more honest mining in the malicious chain; less α is necessary. We have also seen that the latency strongly influences which block each node receives first. Thus, one of the strategies adopted by the author defines that if the attacker is leading to one block, as soon as the honest nodes release a block, the attacker also sends one. The goal is that the attacker block reaches a portion of honest nodes that will adopt the block as a reference for mining, increasing γ and the likelihood of the attacker winning the race. If the attacker is winning by two or more, it keeps the chain hidden, only revealing the blocks when the difference reaches one.

(ii) Trail stubborn: when the attacker private chain is behind the public chain. The attacker continues mining in private, instead of leaving it, in the hope of reaching and exceeding the public chain. This strategy shows promise if the attacker possesses a certain amount of computational power.

(iii) Equal fork stubborn: the attacker uses this strategy when the honest nodes equalize the race, and the attacker continues mining in its chain until it is one block ahead when he releases his chain.

The *Eclipse Attack* [51] is an attack on the network level which occurs when an attacker monopolizes all connections of a given node, isolating the victim and filtering all messages sent and received. As a result, the victim has a different chain view. The victim can have blocks prevented from being included in the chain and can be forced to work in the attacker chain. In the Bitcoin, the nodes maintain up to 125 connections with its neighbors, being 8 outbound connections, and 117 inbound connections. Outbound connections are those initiated by the node itself, and the inbound connections are solicited by other nodes, as we have seen in Section 2.4 that deals with the P2P network.

The attack consists of filling up the *Tried Table* with addresses controlled by the attacker and fills the *New Table* with invalid addresses. In this way whenever the victim is seeking a new connection it connects to an address controlled by the attacker. The nodes only accept valid IP to connect; then *New Table* is populated with invalid addresses so that the attacker saves IPs. So, it simply performs two routines repeatedly to populate the victim's tables. First, to establish connections, the attacker requests a connection and then disconnects and solicits for a new connection with other addresses; this is enough to fill the *Tried Table*. To fill the *New Table*, it is necessary to send many *ADDR* messages with bogus IP addresses to the victim. Each *ADDR* can contain up to 1000 addresses. They use Class C addresses or reserved IP, as the multicast needed at least 16384 addresses to populate the *New Table*. In their experiments, with a *botnet* of only 400 machines, he was capable of fully populating the *New Table* and 60% of *Tried Table*, achieving success in controlling all connections of the victim at 80% tries.

The *balance attack* [71] occurs when an attacker disrupts communications between subgroups on a network. During the time that the network is partitioned, he releases transactions in one subgroup and mines blocks in another one. With high probability, the chain of the block subgroup outweighs the chain of the transaction subgroup. This strategy allows the attacker to mine a branch possibly in isolation of the rest of the network before merging its branch to one of the competing Blockchains to influence the branch selection process. The author shows that the GHOST consensus algorithm is prone to this attack.

To improve the throughput (transactions per second), the Ethereum uses the GHOST (Greedy Heaviest Observed Subtree) consensus algorithm. The Bitcoin generates one block every 10 minutes, while Ethereum generates one block every 12–15 seconds. Besides this improvement, the Ethereum generates much more forks. To avoid wasting large mining efforts while resolving forks the GHOST protocol iteratively selects, as the successor block, the root of the subtree that contains the largest number of nodes.

Kiayias and Panagiotakos [72] propose the liveness attack, which delays, as much as possible, the transaction confirmation. They also present two instantiations of such attack on Bitcoin and Ethereum.

Liveness attack consists of three phases, namely, attack preparation phase, transaction denial phase, and Blockchain retarder phase:

(i) Attack preparation phase: just like selfish mining attack, an attacker builds a private chain, which is longer than the public chain.

(ii) Transaction denial phase: the attacker privately holds the block that contains transaction, in order to prevent transaction from being written into the public chain.

(iii) Retarder phase: the transaction will no longer be able to be privately held. In this case, the attacker will publish the block that contains it. In some Blockchain systems, like in Ethereum, when the depth of the block that contains the transaction is greater than a constant, the transaction will be regarded valid. The attacker will continue building private chain to build an advantage over the public chain. After that, he will publish the blocks into public chain to slow down the growth rate of public chain. The liveness attack will end when transaction is verified as valid in the public chain.

The Sybil attack was first described by a Microsoft researcher Douceur [33]. Sybil's attack implies a situation where one node in the network acquires several identities. It is based on the fact that peer-to-peer networks cannot reliably distinguish between members in some Internet services that provide one IP address for all their users.

In Blockchain networks [73], an attacker may try to fill the network with nodes controlled by him. This allows him to launch the following rogue schemes:

(i) Refusing to transmit and receive blocks

(ii) 51% attack and double spending

In centralized networks, Sibyl attacks are usually avoided using a set of heuristic rules. For example, the system may require that only a limited number of accounts can be created from the same IP address within the allotted time interval. In Bitcoin Blockchain, Sybil attacks are eliminated by special requirements that rule the generation of new blocks. Because an attacker can only create a limited number of blocks, this provides reliable cryptographic protection against Sybil attacks. It turns out that the fraudster needs to have the actual computing power, which cannot be faked.

Quantum algorithms, like Shor's algorithm [74, 75], in theory, will be able to break the elliptic curve signature scheme and, consequently, digital signatures used in Blockchain networks. Aggarwal et al. [76] find that the proof of work used by Bitcoin may still be resistant to quantum computers in the next 10 years. Then, the development of quantum computers poses a serious threat to almost all of the cryptography and, therefore, to Blockchain. This algorithm can be used in two ways to attack the Blockchain. The first is that it can be used to search for hash collisions which can be used to carry out a 51% attack, replacing blocks without disturbing the chain integrity. The second is that it can speed up the nonce generation, consequently, recreating a new chain.

To best exemplify the *selfish mining* attack, we performed simulations using the NS-3 module developed by [48]. The module was developed with the objective of analyzing the impact on the stale block rate, network throughput, the block propagation time, and double spent gain. The nodes connections use point-to-point protocol, abstracting intermediary devices. To configure the channel characteristics (latency and bandwidth), statistical data from various sources were used, such as Verizon and https://testmy.net. To model the proof of work, values of mining power are assigned to nodes and statistically distribute the blocks generation. The data inputs are the block rate, block size, and the spent double value. Analyzing the major Bitcoin mining pools, we conclude that the 15 largest mining pools have 96.3% of mining power. For this reason, 16 nodes have been simulated, representing the 15 mining pools and the other miners grouped as one. The honest nodes adopt the standard protocol, while the attacker follows the heuristic proposed by the author:

(i) Adopt: the attacker adopts the honest chain; this corresponds to restarting the attack. The attacker infers that the honest nodes have a higher probability of winning the race.

(ii) Overlay: it occurs when the attacker has one block more than the honest chain. It is a good strategy when there is a portion of honest nodes mining on the attacker chain.

(iii) Match: the attacker publishes as many blocks as those published by honest. This action aims to make some honest nodes mining on the attacker chain. Following, the attacker can use the *Overlay* action.

(iv) Wait: the attacker is mining constantly in a private chain, without revealing it.

(v) Publish: it corresponds to unveiling its chain.

First 30 simulation rounds were done, generating 10,000 blocks in each. The simulations were run with 16 miners, and all of them followed the standard protocol, without attacker. This results in 0.13% of forks in the chain. In the second simulations round, one node was chosen as attacker. Initially, he has 20% of mining power, which was increased by steps of 5% until 50%.

5. The Stalker Miner

At the network layer, each node does TCP to its neighbors. All transactions have a special address, and only the node with the right key can unlock it. For this reason, we can say that Blockchain provides a certain degree of privacy. Discovering which nodes are doing specific transactions is very difficult, but not impossible. It is possible to infer the IP address and nodes identity through various techniques [53–57, 77].

It is difficult to imagine motivations for an attacker to spend a tremendous power to make a specific user to not publish his blocks. First, the main marketing in this system is confidence. If the miners are spending resources and do not receive their reward fatally, they will leave work, and the confidence will be broken. Second, in large networks, like Bitcoin and Ethereum, the attacker has to spend much money to buy specific hardware.

In systems that use proof of work, the attacker must possess enormous computational power and certainly will spend a lot of money to become a selfish attacker. But the future rewards may be greater. Imagine that two companies decide to acquire a large asset from a third party, and there is a smart contract with a priority clause for one of the companies. This clause says that the second company can acquire the asset only if the first one does not make the payment in a specific date. The second company can then carry out the attack in order to prevent this transaction being confirmed by the network. Ali et al. [46] announced the first known attack of selfish miner to a production network, proving that the attack is doable, despite the motivations or being very costly. The data collected show signs of attacker behavior: for example, miners were not accepting transactions; a long delay in blocks was noticed followed by blocks in rapid succession; and there were a lot of rejected blocks.

The stalker, detailed in Algorithm 1, is a variant of selfish mining. In this attack, the malicious node has the aim of not permitting that a specific miner publishes his blocks. The difference between the stalker and the selfish mining is the ultimate goal, while selfish mining seeks to increase the relative revenue, and the stalker seeks to deny a specific target, not worrying about gain. This attack only uses *Adopt, Wait, and Publish* heuristics (see Algorithm 1), because *Overlay and Match* have the objective of making honest nodes work to increase the relative revenue. All honest mining follows the

```
if next block == victim then
    if la > lh then
        publish attacker's chain;
    else
        switch ReadDecisionMatrix do
            case wait do
                continue mining
            end
            case adopt do
                restart attack
            end
        end
    end
else
    switch ReadDecisionMatrix do
        case wait do
            continue mining
        end
        case adopt do
            restart attack
        end
    end
end
```

Algorithm 1: Stalker attack.

protocol; then they reveal a block immediately after mining it. They accept the most extended chain and mine on top of it.

The stalker mining strategy consists of the following deviations from honest mining; see Figure 9:

(i) When there is not a fork and the victim mines the next block, accept the honest chain.

(ii) When leading and receiving a victim's block, unveil private chain and restart an attack.

(iii) When leading ≥ 2 and honest mining the next block, continue mining in a private chain and wait for a victim block.

(iv) When leading and the attacker mines the next block, continue mining in a private chain and wait for a victim block.

Two decisions define attacker mining strategy:

(i) The best moment to unveil the private chain

(ii) When to accept the public's chain

The attacker follows the following proposed heuristic:

(i) Adopt: the attacker adopts the honest chain. This corresponds to restarting the attack. The attacker infers that the honest will lose many blocks, which is not the objective.

(ii) Wait: the attacker mines constantly in a private chain, without revealing it.

(iii) Publish: this happens when the length of attacker chain *(la)* is greater than the honest one *(lh)*, and the victim publishes the next block. This corresponds to unveiling the attacker's chain and publishing it.

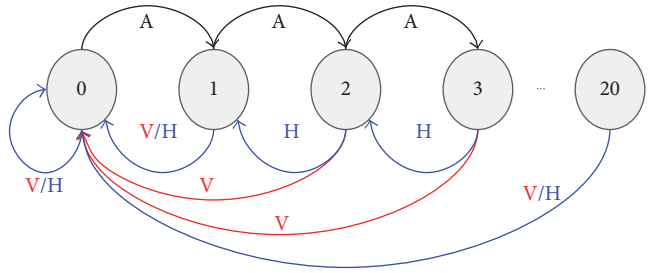


Figure 9: Stalker miner strategy. A is attacker, H is the honest nodes, and V is the victim.

The single-player decision problem cannot be modeled directly as an MDP because the attacker function is nonlinear. To use MDPs, we apply the Gervais technique [30], applying an MDP solver for finite state space MDPs, and use a cutoff value of 20 blocks. We use a decision matrix, Table 3, which gives two actions: adopt the honest chain or wait and continue mining. The attacker only unveils his chain when receiving a victim's block. However, the attack has the side effect of "waste" mining power of honest nodes. Sometimes the stalker only receives the victim block when the private chain is 3 or more blocks ahead, resulting in discarding honest blocks.

To evaluate the attack performance, we use the NS-3 module constructed by Gervais. We need to rely on simulations as the only workable alternative to realistically capture the Blockchain performance under this attack, since neither formal modeling nor the deployment of thousands of peers (e.g., currently there are 6000 reachable full nodes in Bitcoin) would be practical.

The NS-3 module evaluates different Blockchain parameters, such as the block interval, the block size, the propagation mechanisms by measuring the resulting stale block rate throughput, and block propagation times. We use the following parameters in our simulations:

(i) Total nodes: 16

(ii) Block interval distribution: 10 minutes

(iii) Block size distribution: variable

(iv) Nodes distribution: worldwide

(v) Connections per node: 15

(vi) Block request system: INV message

Each miner is set up with a mining power (PoW). Based on the block interval distribution, a new block is attributed to a miner. A miner mines on the first block he receives and uses the longest chain rule. The blocks in the fork are discarded. We do not consider difficult changes among different blocks; the longest chain is that with more blocks on it. Point-to-point channels establish the nodes connections, which abstracts away any intermediate devices like routers and switches. These channels have only two characteristics: latency and bandwidth. The block size implicitly simulates the transaction. In the simulator, we only use a miner node type. We use

Table 3: Decision matrix.

		lh										
		1	2	3	4	5	6	7	8	9	10	11
la	1	"w", "w"	"∗", "a"	-	-	-	-	-	-	-	-	-
	2	"w", "w"	"w", "w"	"∗", "a"	-	-	-	-	-	-	-	-
	3	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-	-	-	-	-	-
	4	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-	-	-	-	-
	5	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-	-	-	-
	6	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-	-	-
	7	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-	-
	8	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"∗", "a"	-	-
	9	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "w"	"w", "a"	"∗", "a"	-
	10	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"a", "w"	"w", "w"	"∗", "a"
	11	-	-	-	-	-	-	-	-	-	"a", "a"	"w", "w"

The rows correspond to the length, la, of the adversary's chain and the columns correspond to the length, lh, of the honest network's chain. The two values in each table entry correspond to the forks, first the attacker and second the honest one. w and a denote the wait and adopt actions, respectively.

the mining pool distribution from https://Blockchain.info, but the hash power is the main parameters that we modify in simulations.

Figure 10 exemplifies how the stalker acts. The gray blocks are from honest nodes, the blue corresponds to target, and the red ones to the attacker. At the instant T1, an honest node publishes the block 1, and the attacker then secretly starts mining its blocks chain waiting for a target block. At time T4, the attacker has a chain greater than the honest ones, and the target publishes a block when the stalker decides to publish his chain. As the attacker's chain is greater than the honest chain, all nodes will adopt it as the main chain, and the attacker achieves his goal, which is to prevent the blue blocks in the main chain. But, as we can observe, this attack had the side effect of discarding blocks 2 and 3 of honest knots. If the attacker's chain grows without the victim publishing a block, the attacker discards his chain and restarts a new one with the main chain top block as the reference.

Figure 11 shows the attacker influence on the target. The "X" axis represents the attacker's computational power over the rest of the network. The "Y" axis displays the target stale block. The attacker's computational power starts with 20% and is increased by 5% until it reaches a maximum of 40%. For each striker's power range is also varied, the target's computational power reaches up to 30%. The first conclusion obtained is that the higher the target power the lesser the influence of the attacker on it. This fact can be explained by the fact that the target will publish more blocks and the attacker cannot generate as many forks as they need. The second conclusion is that the higher the attacker's strength the greater the discarded block. The best result achieved by the attacker occurs when he has 40% of the hash power and the target has 5%, when 39% of the target blocks are discarded.

There is also a collateral damage of discarding some blocks from other nodes, this can be observed in Figure 12, and the computational power of these nodes is 50%. This effect can be explained by the fact that the attacker generates the forks before receiving a target block; in this way several other nodes blocks are lost as a consequence. These nodes

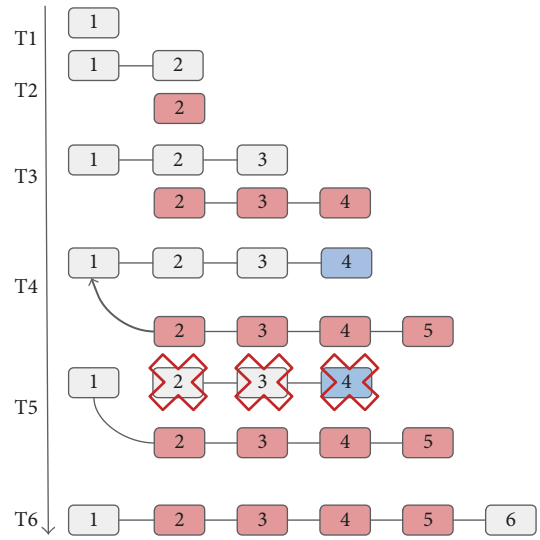


Figure 10: Stalker attack example.

even lose up to 16% of their blocks, which is above the observed natural forks rate of 1.69%.

In Figure 13, we can see that the attacker discarded blocks dim with increasing target strength; this is because the target publishes more blocks and the stalker only reveals its chain when there are target blocks. So, he will also publish more blocks, and with this, there is a decrease of its stale blocks. In Table 4, we compare stale block rate between the target and honest nodes.

6. Final Considerations, Future Prospects, and Open Issues

IoT processes and exchanges large amounts of data without human intervention, and this data often has information that can be critical to security and privacy. Therefore, they are attractive targets for attackers. Typically, these devices are

Table 4: Comparative of stale blocks.

(a) Attacker hash power 20%

Target Hash power	Target			Honest		
	Mined	Stale	%	Mined	Stale	%
5%	92	8	8,30%	1452	9	0,62%
10%	193	15	7,50%	1333	16	1,20%
15%	300	21	6,96%	1268	17	1,34%
20%	383	23	6,06%	1160	19	1,64%
25%	488	32	6,62%	1056	20	1,89%
30%	585	31	5,34%	984	23	2,34%

(b) Attacker hash power 40%

Target Hash power	Target			Honest		
	Mined	Stale	%	Mined	Stale	%
5%	96	38	39,63%	1083	152	14,04%
10%	195	69	35,54%	996	143	14,36%
15%	287	95	33,14%	878	151	17,20%
20%	394	121	30,61%	782	131	16,75%
25%	482	136	28,20%	675	115	17,04%
30%	572	152	26,58%	570	95	16,67%

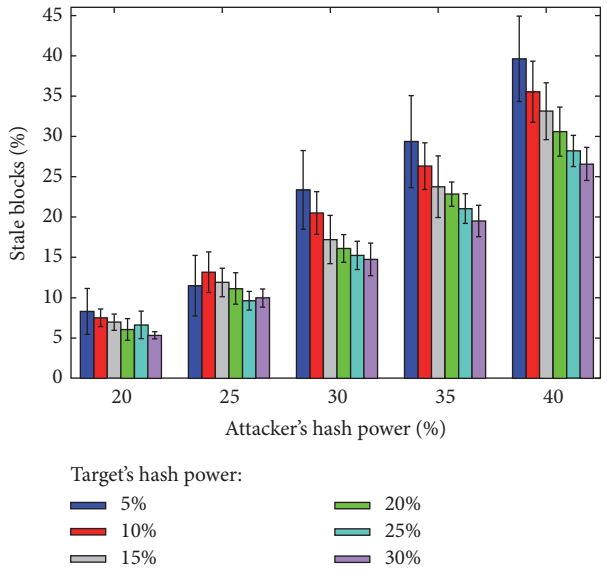


Figure 11: Target's stale blocks.

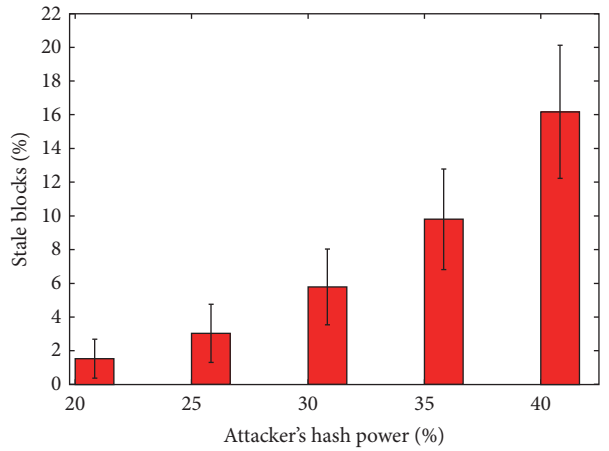


Figure 12: Honest stale blocks.

low-power and low-computing, and they should devote their few resources to their core activities, making the task of supporting security and privacy quite challenging. Traditional security methods tend to be expensive in computational and energetic terms. Also, many of the security frameworks are highly centralized and therefore not necessarily suitable for the IoT scenario because of the difficulty of scalability and the fact that it becomes a single point of failure. Consequently, IoT requires privacy and security protection that is light, scalable, and distributed. Blockchain technology, which supports Bitcoin, has the potential to overcome these challenges as a result of its distributed, secure, and private nature. However, it is not light, requiring adaptations and optimizations.

The combination of Blockchain and IoT can be quite powerful, as Blockchain can provide resilience to attacks and the ability to interact with peers in a reliable and auditable way. Blockchain's continued integration into the IoT domain will cause significant transformations across multiple industries, bringing new business models and making us reconsider how existing systems and processes are implemented.

The "Blockchain" not only enables the movement of money but can also be used to transfer information and allocate resources between devices, enabling the use of Blockchain as a service [78]. The connected world can usefully

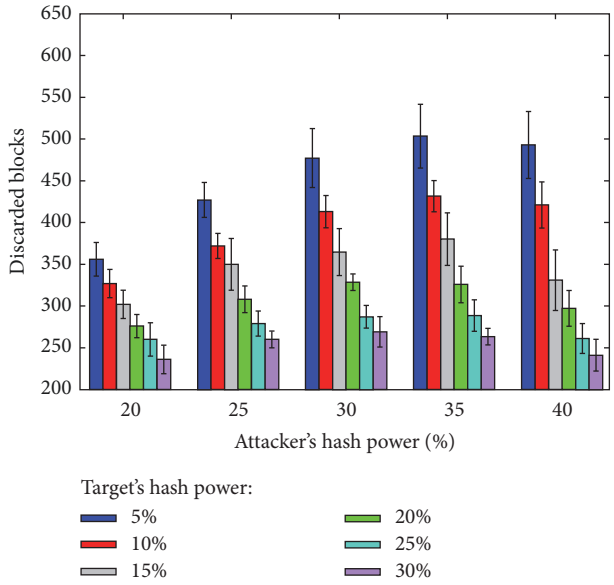


Figure 13: Attacker stale blocks.

include Blockchain technology as a layer for which more and more devices (wearables, sensors, IoTs, smartphones, tablets, laptops, homes, cars, and smart cities) can benefit from their characteristics.

Blockchain, therefore, presents many promising opportunities for the future of IoT. Challenges, however, remain, as consensus models and computational costs of transaction verification. However, it is still in the early stages of developing block chains, and these obstacles will eventually be overcome, opening the way to many possibilities.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank Brazilian Internet Steering Committee (CGI) and São Paulo Research Foundation (FAPESP) for the funding provided to the research (Grant no. 2015/24358-7) that has played a seminal role in providing scientific funding, leading to the preparation of this paper.

References

[1] D. Evans, "A internet das coisas: como a próxima evolução da internet está mudando tudo," *CISCO IBSG*, 2011.

[2] Y. Zhang and J. Wen, "An IoT electric business model based on the protocol of bitcoin," in *Proceedings of the 2015 18th International Conference on Intelligence in Next Generation Networks, ICIN 2015*, pp. 184–191, IEEE, France, February 2015.

[3] M. Pilkington, *Blockchain technology: principles and applications. research handbook on digital transformations*, F. X. Olleros and M. Zhegu, Eds., 2016.

[4] I. Eyal and E. G. Sirer, "Majority Is Not Enough: Bitcoin Mining Is Vulnerable," in *Financial Cryptography and Data Security*, vol. 8437 of *Lecture Notes in Computer Science*, pp. 436–454, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[5] K. Ashton, "That 'internet of things' thing," *RFID Journal*, vol. 22, no. 7, pp. 97–114, 2009.

[6] F. L. Lewis and et al., "Wireless sensor networks," *Smart environments: technologies, protocols, and applications*, pp. 11–46, 2004.

[7] X. Cui, "The internet of things," in *In Ethical Ripples of Creativity and Innovation*, pp. 61–68, Springer, 2016.

[8] J. Fenn and H. LeHong, *Hype Cycle for Emerging Technologies*, Gartner, July 2011.

[9] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[10] P. Guillemin, P. Friess, and et al., "Internet of things strategic research roadmap," The Cluster of European Research Projects, 2009.

[11] Y. Recommendation, "2060 ńoverview of internet of thingsż," *ITU-T, Geneva*, 2012.

[12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[13] W. Stallings, *Network and Internetwork Security: Principles and Practice*, vol. 1, Prentice Hall Englewood Cliffs, 1995.

[14] H. Dobbertin, A. Bosselaers, and B. Preneel, "RIPEMD-160: A strengthened version of RIPEMD fast software encryption," *Lecture Notes in Computer Science*, vol. 1039, pp. 71–82, 1996.

[15] N. Jansma and B. Arrendondo, "Performance comparison of elliptic curve and rsa digital signatures," *Efficiency Comparison of Elliptic Curve and RSA Signatures*, 2004.

[16] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer Science and Business Media, New York, NY, USA, 2006.

[17] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized Computation Platform with Guaranteed Privacy, 2015," https://arxiv.org/abs/1506.03471.

[18] M. Swan, "Blockchain thinking: The brain as a dac (decentralized autonomous organization)," in *Proceedings of the Texas Bitcoin Conferenc*, pp. 27–29, 2015.

[19] A. Andrea, Mastering BitCoin, 2014.

[20] M. Swan, *Blockchain: Blueprint for a New Economy*, O'Reilly Media, Inc., 2015.

[21] S. Nakamoto, *Bitcoin, A peer-to-peer electronic cash system*, 2008, https://bitcoin.org/bitcoin.pdf.

[22] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Proceedings of the Conference on the Theory and Application of Cryptographic Techniques*.

[23] J. Kim, "Safety, liveness and fault tolerance," *The Consensus Choices Stellar*, 2014.

[24] C. Natoli and V. Gramoli, "The Blockchain Anomaly," in *Proceedings of the 15th IEEE International Symposium on Network Computing and Applications, NCA 2016*, pp. 310–317, IEEE, November 2016.

[25] M. J. Fischer, N. A. Lynch, and M. . Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM*, vol. 32, no. 2, pp. 374–382, 1985.

[26] S. King and S. Nadal, "Peer-to-peer crypto-currency with proof-of-stake," *Self-Published Paper*, 2012.

[27] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance and Proactive Recovery," *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[28] Intel., "Proof of elapsed time (poet)," http://intelledger.github.io/.

[29] A. Back, Hashcash-a denial of service counter-measure, 2002.

[30] H. Gilbert and H. Handschuh, "Security analysis of SHA-256 and sisters," in *Selected Areas in Cryptography*, M. Matsui and R. J. Zuccherato, Eds., vol. 3006 of *Lecture Notes in Computer Science*, pp. 175–193, Springer, Berlin, Germany, 2003.

[31] T. H. Kim, "A Study of Digital Currency Cryptography for Bbusiness Marketing and Finance Security," *Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology*, vol. 6, no. 1, pp. 365–376, 2016.

[32] P. Vasin, Blackcoin's proof-of-stake protocol v2, 2014.

[33] J. R. Douceur, "The sybil attack," in *Peer-to-Peer Systems*, vol. 2429 of *Lecture Notes in Computer Science*, pp. 251–260, Springer, Berlin, Germany, 2002.

[34] G. Wood, "Ethereum, A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, 2014.

[35] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *In Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.

[36] M. Pilkington, "Blockchain technology: Principles and applications," *Browser Download This Paper*, 2015.

[37] M. Conoscenti, A. Vetro, and J. C. De Martin, "Blockchain for the Internet of Things: A systematic literature review," in *Proceedings of the 13th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2016*, IEEE, Agadir, Morocco, December 2016.

[38] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: Challenges and Solutions, 2016," https://arxiv.org/abs/1608.05187.

[39] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for IoT," in *Proceedings of the 2nd IEEE/ACM International Conference on Internet-of-Things Design and Implementation, IoTDI 2017*, pp. 173–178, ACM, Pittsburgh, PA, USA, April 2017.

[40] M. Conoscenti, A. Vetro, and J. C. De Martin, "Peer to peer for privacy and decentralization in the internet of things," in *Proceedings of the 39th IEEE/ACM International Conference on Software Engineering Companion, ICSE-C 2017*, pp. 288–290, IEEE, Buenos Aires, Argentina, May 2017.

[41] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, pp. 6–10, 2016.

[42] D. Wörner and T. Von Bomhard, "When your sensor earns money: Exchanging data for cash with Bitcoin," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2014*, pp. 295–298, ACM, September 2014.

[43] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 618–623, Kona, Big Island, HI, USA, March 2017.

[44] D. Wörner and T. Von Bomhard, "When your sensor earns money: Exchanging data for cash with Bitcoin," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2014*, pp. 295–298, September 2014.

[45] K. T. Nguyen, M. Laurent, and N. Oualha, "Survey on secure communication protocols for the Internet of Things," *Ad Hoc Networks*, vol. 32, article no. 1181, pp. 17–31, 2015.

[46] M. Ali, J. C. Nelson, R. Shea, and M. J. Freedman, "Blockstack: A global naming and storage system secured by blockchains," in *Proceedings of the USENIX Annual Technical Conference*, pp. 181–194, 2016.

[47] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *Proceedings of the IEEE Security and Privacy Workshops, SPW 2015*, pp. 180–184, IEEE, May 2015.

[48] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Čapkun, "On the security and performance of Proof of Work blockchains," in *Proceedings of the 23rd ACM Conference on Computer and Communications Security, CCS 2016*, pp. 3–16, Austria, October 2016.

[49] I. Eyal, "The miner's dilemma," in *Proceedings of the 36th IEEE Symposium on Security and Privacy, SP 2015*, pp. 89–103, IEEE, San Jose, CA, USA, May 2015.

[50] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn mining: generalizing selfish mining and combining with an eclipse attack," in *Proceedings of the 1st IEEE European Symposium on Security and Privacy*, pp. 305–320, IEEE, 2016.

[51] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoins peer-to-peer network," *In USENIX Security*, pp. 129–144, 2015.

[52] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "FairAccess: a new Blockchain-based access control framework for the Internet of Things," *Security and Communication Networks*, vol. 9, no. 18, pp. 5943–5964, 2017.

[53] M. Spagnuolo, F. Maggi, and S. Zanero, "Bitiodine: Extracting intelligence from the bitcoin network," in *Proceedings of the nternational Conference on Financial Cryptography and Data Security*, pp. 457–468, Springer, 2014.

[54] M. Moser, R. Bohme, and D. Breuker, "An inquiry into money laundering tools in the Bitcoin ecosystem," in *Proceedings of the 2013 APWG eCrime Researchers Summit, eCRS 2013*, IEEE, USA, September 2013.

[55] J. Herrera-Joancomartí, "Research and Challenges on Bitcoin Anonymity," in *Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance*, vol. 8872, pp. 3–16, Springer, 2015.

[56] P. Koshy, D. Koshy, and P. McDaniel, "An analysis of anonymity in bitcoin using P2P network traffic," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 469–485, Springer, 2014.

[57] L. Valenta and B. Rowan, "Blindcoin: blinded, accountable mixes for Bitcoin," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 112–126, Springer.

[58] Y. Sun, H. Song, A. J. Jara, and R. Bie, "Internet of things and big data analytics for smart and connected communities," *IEEE Access*, vol. 4, pp. 766–773, 2016.

[59] S. Panikkar, S. Nair, P. Brody, and V. Pureswaran, "Adept: An iot practitioner perspective," *IBM Institute for Business Value*, 2014.

[60] N. Kshetri, "Can Blockchain Strengthen the Internet of Things?" *IT Professional*, vol. 19, no. 4, Article ID 8012302, pp. 68–72, 2017.

[61] S. Popov, "The tangle, 2016," https://iota.org/IOTA_Whitepaper.pdf.

[62] A. C. Yao, "Protocols for secure computations," in *Proceedings of the Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium*, pp. 160–164, IEEE, 1982.

[63] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[64] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin, "Efficient multiparty computations secure against an adaptive adversary," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 1592, pp. 311–326, 1999.

[65] R. Gennaro, M. O. Rabin, and T. Rabin, "Simplified VSS and fast-track multiparty computations with applications to threshold cryptography," in *Proceedings of the 1998 17th Annual ACM Symposium on Principles of Distributed Computing*, pp. 101–111, ACM, July 1998.

[66] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control," *Journal of Medical Systems*, vol. 40, no. 10, article no. 218, 2016.

[67] A. Chakravorty, T. Wlodarczyk, and C. Rong, "Privacy preserving data analytics for smart homes," in *Proceedings of the 2nd IEEE Security and Privacy Workshops, SPW 2013*, pp. 23–27, USA, May 2013.

[68] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[69] J. A. Kroll, I. C. Davey, and E. W. Felten, "The economics of bitcoin mining, or bitcoin in the presence of adversaries," in *In Proceedings of WEIS*, vol. 2013, 2013.

[70] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," in *Proceedings of the 13th IEEE International Conference on Peer-to-Peer Computing, IEEE P2P 2013*, IEEE, Trento, Italy, September 2013.

[71] C. Natoli and V. Gramoli, "The Balance Attack or Why Forkable Blockchains are Ill-Suited for Consortium," in *Proceedings of the 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2017*, pp. 579–590, IEEE, June 2017.

[72] A. Kiayias and G. Panagiotakos, "On trees, chains and fast transactions in the blockchain," *IACR Cryptology ePrint Archive*, vol. 2016, p. 545, 2016.

[73] G. Bissias, A. P. Ozisik, B. N. Levine, and M. Liberatore, "Sybil-resistant mixing for bitcoin," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, pp. 149–158, ACM, Scottsdale, Arizona, USA, 2014.

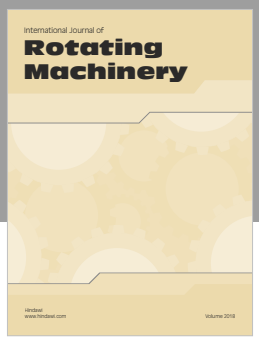
[74] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *35th Annual Symposium on Foundations of Computer SCIence (SANta FE, NM, 1994)*, pp. 124–134, IEEE Comput. Soc. Press, Los Alamitos, CA, 1994.

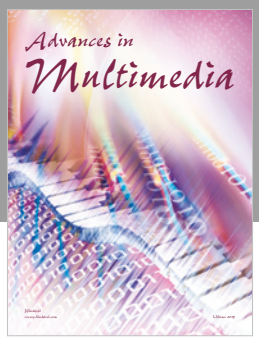
[75] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999.

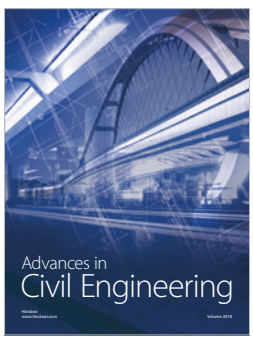
[76] D. Aggarwal, G. K. Brennen, T. Lee, M. Santha, and M. Tomamichel, "Quantum attacks on bitcoin, and how to protect against them, 2017," https://arxiv.org/abs/1710.10377.

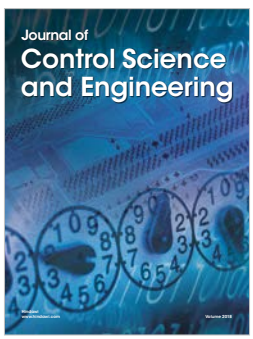
[77] A. Miller, J. Litton, A. Pachulski et al., Discovering bitcoin's public topology and influential nodes, et al., 2015.

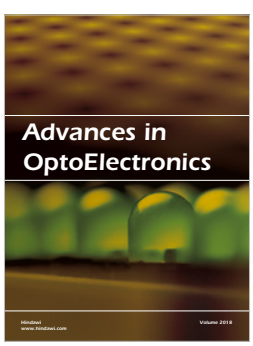
[78] T. Swanson, "Consensus-as-a-service: a brief report on the emergence of permissioned, distributed ledger systems," *Report, available online*, 2015.

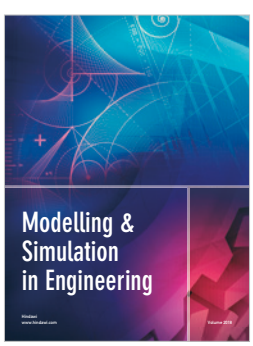


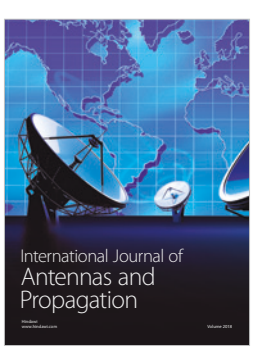


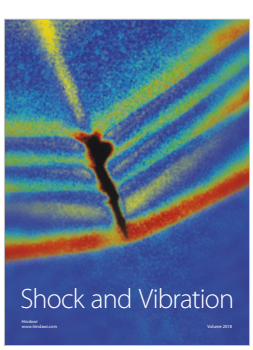


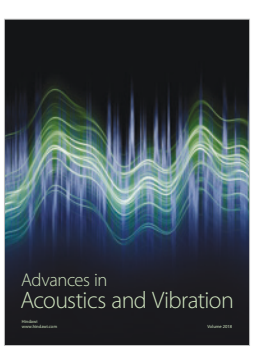












Hindawi

Submit your manuscripts at
www.hindawi.com