COUNTERMEASURES FOR THE MAJORITY ATTACK IN BLOCKCHAIN DISTRIBUTED SYSTEMS

Fredy Andrés Aponte Novoa

Universidad del Norte
Departamento de Ingeniería de Sistemas
Doctorado en Ingeniería de Sistemas y Computación
Barranquilla, Colombia
2023

COUNTERMEASURES FOR THE MAJORITY ATTACK IN BLOCKCHAIN DISTRIBUTED SYSTEMS

Fredy Andrés Aponte Novoa

A dissertation submitted in partial fulfillment of the requirements for the degree of Ph.D. in Systems Engineering and Computing.

Advisor: Ricardo Luis Villanueva Polanco, Ph.D.
Co-advisor: Ana Lucila Sandoval Orozco, Ph.D.
Co-advisor: Pedro Mario Wightman, Ph.D.

Evaluating committee:
José Duván Márquez Díaz, Ph.D.
Miguel Ángel Jimeno Paba, Ph.D.
Lihki José Rubio Ortega, PhD.
Alfredo J. Perez, Ph.D.
Mauricio Barrios, Ph.D.

Universidad del Norte
Departamento de Ingeniería de Sistemas
Doctorado en Ingeniería de Sistemas y Computación
Barranquilla, Colombia
2023

To God for giving me the necessary strength to not give up on achieving this goal.

To my mother for her love and unconditional support.

To Jimena, my beloved wife, for never letting go off my hand all this way.

To Elizabeth, the Light of my Eyes, her love, joy and her whole being, are the fuel to carry on day by day.

To my sisters Nubia, Paola, and Mónica, for their constant support and motivation.

To the memory of my beloved Father who from heaven, continues to care for us.

To the memory of Ines Meriño, excellent friend and companion.

Acknowledgements

I sincerely thank my director, Professor Ricardo Villanueva Polanco, for his guidance and unconditional support. His teachings and advice were essential for my doctoral formation. I also thank the members of the Research Analysis, Security and Systems Group (GASS) of the Computer Science Faculty of the Complutense University of Madrid (Spain) for letting me join their team during my internship. A special thanks to Professor Ana Lucila Sandoval Orozco for her co-direction; her guidance, support, and advice were important in the development of the degree work. I also thank Daniel Povedano Álvarez for his collaboration, talks, and friendship; his fellowship was essential during the internship. I thank my teachers for their teachings and dedication, especially to professor Daladier Jabba Molinares. I thank my partners and friends for their sharing and teachings.

I thank my parents in-law Bernardo and Myriam, for their unconditional support, and my brothers and sisters-in-law, especially Deiby and Natalia, for constantly keeping an eye on me during this process.

I thank the Government of Boyacá (Colombia) for the financial support to carry out my doctoral studies through the program "Convocatoria 779 de 2017 - Convocatoria Nacional de Doctorado para la Formación de Capital Humano de Alto Nivel para el Departamento de Boyacá - 2017". Also, I thank the Universidad del Norte for managing the academic and administrative processes related to my studies.

Finally, and most important, thanks to the women in my life, my mother Bernarda, my daughter Elizabeth, my wife Jimena, and my sisters Nubia, Paola, and Mónica, for their unconditional support and motivation.

Abstract

In recent years, blockchain has become a disruptive technology to protect the integrity of information, especially in open and collaborative information systems. Its main advantage is the possibility of reaching a consensus on the new data blocks to be added to the chain, even with anonymous actors. The applications that use blockchain are cryptocurrencies, decentralized finance applications, video games, and many others. Most of these applications trust that the blockchain will prevent issues like fraud, thanks to the built-in cryptographic mechanisms provided by the data structure and the consensus protocol. However, blockchains suffer from what is called a 51% attack or majority attack, which is considered a high risk for the integrity of these blockchains, where if a miner, or a group of them, has more than half the computing capability of the network, it can rewrite the blockchain. Even though this attack is possible in theory, it is regarded as hard-achievable in practice due to the assumption that, with enough active members, it is very complicated to have that much computing power; however, this assumption has not been studied with enough detail.

Consensus protocols are a fundamental part of any blockchain; although several protocols have been in operation for several years, they still have drawbacks. For instance, some may be susceptible to a 51% attack, also known as a majority attack, which may suppose a high risk to the trustworthiness of the blockchains. Although this attack is theoretically possible, executing it in practice is often regarded as arduous because of the premise that, with sufficiently active members, it is not 'straightforward' to have much computing power. Since it represents a possible vulnerability, the community has made efforts to solve this and other blockchain problems, which has resulted in the birth of alternative consensus protocols, e.g., the proof of accuracy protocol.

Cryptojacking or illegal mining is a form of malware that hides in the victim's computer and takes the computational resources to extract cryptocurrencies in favor of the attacker. It generates significant computational consumption, reducing the computational efficiency of the victim's computer. This attack has increased due to the rise of cryptocurrencies and their profitability and their difficult detection by the user. The identification and blocking of this type of malware have become an aspect of research related to cryptocurrencies and blockchain technology; in the literature, some machine learning and deep learning techniques are presented, but they are still susceptible to improvement.

This work presents four main contributions: - A new characterization of the consen-

sus algorithms. - A detailed characterization of the miners in the bitcoin and crypto ethereum blockchains. - A detailed proposal of a proof-of-accuracy protocol. - A exploration of multiple Machine Learning classification models for detecting cryptojacking on websites. These contributions have already been included in scientific articles published n high-impact journals. The related articles can be found at [Aponte-Novoa et al., 2022b, Aponte-Novoa and Villanueva-Polanco, 2022b, Aponte-Novoa et al., 2021, Aponte Novoa et al., 2021, Aponte et al., 2021a]

Keywords

51% attack, alternative consensus protocols, bitcoin, blockchain, cryptojacking, consensus protocol, double-speding, ethereum, hash rate, illegal mining, malware; machine learning, proof of accuracy

Contents

List of Figures

List of Tables

1 Introduction

Blockchain technology promises to become an excellent opportunity to provide different solutions for society's problems, "... Like the Internet reinvented communication, blockchain may similarly disrupt transactions, contracts, and trust – the underpinnings of business, government, and society" [Piscini et al., 2016]. It is defined as "a perpetually updated record of transactions independently saved by users across the internet"; in other words, it is an immutable distributed ledger [Wessel and Olson, 2016]. The basic operation of a blockchain consists of the secure administration of a shared ledger, where transactions are verified and stored in a network of anonymous nodes that does not have a central authority. A blockchain can be public or private, where permissions to read or write can be configured. Some mathematical tools, like cryptographic hash functions, and computational ones, like a p2p network and consensus algorithms, allow the blockchain to work not only to execute transactions but also to protect the integrity and anonymity of the users. However, blockchain, despite its strong data structure and other benefits, has some shortcomings, like the computational cost to run the blockchain's consensus algorithms, which usually requires the solution of complex mathematical problems in parallel by a large number of users, all competing to finish first in a global race [Piscini et al., 2016].

Although the effort required to solve the problems is high, there are users with enough computing power who could not only solve them quickly and in a distributed manner, but also try to take over the network by generating a new version of the blockchain, violating one of the core design principles of this structure.

The 51% attack (also called Majority attack) is categorized into hash-based vulnerabilities ("hash-based attack"). It entails one or more miners taking control of at least 51% of the mined hash or computation in the blockchain network [Dean, 2015]. It can be more formally defined as a hash-based attack that occurs in a blockchain when one or more miners take control of at least 51% of all the mining of hash or the computation in the blockchain network. With this computational power, a miner may alter transactions in a blockchain network and hence hinder the process of storing a new block [N and M, 2019].

By executing a 51% attack, a miner can arbitrarily manipulate and modify the informa-

tion on the blockchain. Specifically, an attacker can exploit this vulnerability to carry out the following attacks: a) reverse transactions and initiate a double-spending attack; that is, spending the same coins multiple times; b) exclude and modify the order of transactions; c) hamper the normal mining operations of other miners; and d) prevent the confirmation operation of normal transactions [Dean, 2015].

If a few miners gather the mining power in a blockchain that uses the Proof-of-Work (PoW) consensus mechanism, then fear of an inadvertent situation may occur, such as one group controlling more than 50% of the computing power hash [Li et al., 2020]. In January 2014, the mining group ghash.io reached 42% of the total hash power in bitcoin, which caused several miners to voluntarily leave the group, while ghash.io, in a press release, assured the bitcoin community that would avoid reaching the 51% hash power threshold [Hajdarbegovic, 2014]. In this case, there was a self-control mechanism based on honor; however, this kind of issue cannot be left to chance if the blockchain would like to become a more widely accepted infrastructure for transactions.

Since blockchain is an emerging technology, its applicability to new domains and challenges is constantly growing. [Le and Hsu, 2021] presented a taxonomy of blockchain-based applications and an analysis of blockchain challenges regarding security and performance. Covered 96 papers categorized into 7 application domains: finance (e.g., [Wang and Kogan, 20 achievement records (e.g., [Le et al., 2022]), energy (e.g., [Gao et al., 2018]), healthcare (e.g., [Javed et al., 2021]), manufacturing (e.g., [Kurpjuweit et al., 2021]), supply chain (e.g., [Kurpjuweit et al., 2021]), shipping and delivery (e.g., [Wu et al., 2017]), and sustainability (e.g., [Saberi et al., 2019]). Regarding security challenges, they highlighted majority attacks [Aponte-Novoa et al., 2021], DDoS attacks, selfish mining, and others [Le and Hsu, 2021]. Concerning performance challenges, they focused on throughput and latency in some blockchains, as well as resource and energy management issues [Le and Hsu, 2021].

Another problem related to the applications of blockchain consists of the unauthorized mining activity on the victim's computer, using the computational power of the victim's computer to extract cryptocurrencies, which generates large computational consumption, reducing the computational efficiency of the victim's computer. This is an illegal practice called Cryptojacking. Detection techniques for this problem, such as browser extensions and antiviruses, provide a partial solution to the cryptojacking problem since attackers can avoid them by employing obfuscation techniques or renewing domains or malicious scripts frequently. [Tekiner et al., 2021]. Cryptojacking boomed with the birth of service providers that offered ready-to-use implementations of mining scripts in web browsers. This way, attackers can reach many more victims through websites. These service providers are coinhive [Coi,] and cryptoloot [Cry, c]. Moreover, this attack may be used by a powerful attacker to increment their computational power, posing a risk to

any blockchain based on mining [Carlin et al., 2020, Aponte et al., 2021a, Tayyab et al., 2022, Wu et al., 2022, Bijmans et al., 2019, Aponte-Novoa et al., 2021, Aponte-Novoa and Villanueva-Polanco, 2022b].

Cryptojacking on websites uses JavaScript code to mine cryptocurrencies. This technique does not require installing JavaScript code to perform the mining process. All it takes is for the user to load an infected website in their browser for the illegal mining code to execute in the browser of the victim's computer [Cry, a]. According to [Cry, b], many websites have been infected by cryptojacking, such as personal blogs up to Alexa-ranking websites. Also, it noted that as of January 2022, there were around 3000 sites websites that offered online cryptojacking scripts.

1.1 Goal of this thesis

This thesis work seeks to propose a majority attack detection and mitigation strategy in a blockchain distributed system. For this, the general objective is to Design and implement majority attack detection and mitigation strategies (51% attack) in a blockchain distributed system, based on the characterization of the behavior of miners which is supported by the following three specific objectives:

1. Characterize the historical behavior of miners and their computational power in the most popular blockchain systems bitcoin and crypto ethereum.
2. Design and implement a consensus protocol to control the computing power of miners and mitigate majority attacks on distributed blockchain systems
3. Build a strategy to detect and mitigate malicious software of the "Cryptojacking" type in end user devices

1.2 Contributions

This dissertation presents four main contributions:

1. A new characterization of consensus algorithms, that can be used to find families of mechanisms using cluster-based classification. Using the Ward Method and Spearman's Rank Correlation analysis, new clusters of consensus mechanisms were identified. The results describe the behavioral patterns not seen before in the literature.
2. A detailed characterization of the miners in the bitcoin and crypto ethereum blockchains, with the aim of proving the computing distribution assumption and

the creation of profiles that may allow the detection of anomalous behaviors and prevent 51% attacks. The results of our analysis show that, in the last years, there has been an increasing concentration of hash rate power (computing power) in a very small set of miners, which generates a real risk for current blockchains. Also, there is a pattern in mining among the main miners, which makes it possible to identify out-of-normal behavior.

3. A detailed proposal of a proof-of-accuracy protocol. It presents the proposed protocol progressively, starting with an initial blueprint, which is improved further concerning security. The last protocol version removes the need for a coordinator and combines the proof-of-work feature with access to random locations to improve the protocol's resistance to majority attacks. It aims to democratize the miners' participation within a blockchain, control the miners' computing power, and mitigate the majority attacks.

4. An exploration of multiple Machine Learning classification models for detecting cryptojacking on websites, such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Classifier, k-Nearest Neighbor, and XGBoost.The results suggest that simple models such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, and k-Nearest Neighbor models, can achieve success rate similar to or greater than that of advanced algorithms such as XGBoost and even those of other works based on Deep Learning.

This manuscript is based on the following articles:

1. Aponte-Novoa, F.A.; Povedano Álvarez, D.; Villanueva-Polanco, R.; Sandoval Orozco, A.L.; García Villalba, L.J. On Detecting Cryptojacking on Websites: Revisiting the Use of Classifiers. Sensors 2022, 22, 9219. `https://doi.org/10.3390/s22239219` [Aponte-Novoa et al., 2022b]

2. Aponte-Novoa, F.A.; Villanueva-Polanco, R. On Proof-of-Accuracy Consensus Protocols. Mathematics 2022, 10, 2504. `https://doi.org/10.3390/math10142504`. [Aponte-Novoa and Villanueva-Polanco, 2022b]

3. F. A. Aponte-Novoa, A. L. S. Orozco, R. Villanueva-Polanco and P. Wightman, "The 51% Attack on Blockchains: A Mining Behavior Study," in IEEE Access, vol. 9, pp. 140549-140564, 2021, `doi:10.1109/ACCESS.2021.3119291`. [Aponte-Novoa et al., 2021]

4. Aponte-Novoa, F. A., Jabba-Molinares, D., and Wightman-Rojas, P. M. (2021). Uso y Aplicaciones de la Integración Entre Computación Cuantica y Blockchain:

Revisión Sistemática Exploratoria. Mundo FESC, 11(21) [Aponte Novoa et al., 2021]

5. F.Aponte, L.Gutierrez, M.Pineda, I.Meriño, A.Salazar and P.Wightman, "Cluster-Based Classification of Blockchain Consensus Algorithms," in IEEE Latin America Transactions, vol.19, no.4, pp.688-696, April 2021, doi:10.1109/TLA.2021.9448552. [Aponte et al., 2021a]

Paper [Aponte et al., 2021a] is the base for chapter 3, paper [Aponte-Novoa et al., 2021] is the base for chapter 4, paper [Aponte-Novoa and Villanueva-Polanco, 2022b] is the base for chapter 5, and paper [Aponte-Novoa et al., 2022b] is the base for chapter 6. Some parts of the paper [Aponte Novoa et al., 2021] are used in chapter 1 and chapter 2 of this manuscript.

1.3 Overview

This document is organized as follows:

Chapter 2 describes different consensus algorithms from the original Proof-of-Work (PoW) to some of the alternative protocols. Also, different techniques for the 51% of attack prevention are presented.

Chapter 3 presents a new characterization of the consensus algorithms that can be used to find families of mechanisms using cluster-based classification.

Chapter 4 presents a detailed characterization of the miners of the bitcoin and crypto ethereum blockchains, to test the assumption of computer distribution and create profiles to detect anomalous behavior and prevent 51% attacks.

Chapter 5 presents a detailed proposal for the formalization of what is called a Proof-of-Accuracy protocol. The main contribution of that chapter is to introduce a Proof-of-Accuracy protocol.

Chapter 6 presents an exploration of machine learning models (comparatively simpler than deep learning models) for cryptojacking classification to identify which of these machine learning models may render desirable results.

2 Generalities of Blockchain

2.1 Consensus algorithms

This section presents the consensus algorithms Proof-of-Work, Proof-of-Stake, Hybrid form of these algorithms, and other proof-based consensus algorithms.

2.1.1 Original Proof of Work

In a blockchain, when a new block is added, an agreement between the nodes is required. The most popular agreement process is called Proof-of-Work (PoW). It consists in each node trying to solve a puzzle, whose difficulty can be adjusted, so that the node solving it first will get the right to add a new block to the current chain. The effort made by the node for the solution of the puzzle is called PoW and is payed to the winning node using the internal currency. These nodes are called mining nodes or miners, and the action of trying to solve the puzzle is called mining [Nguyen and Kim, 2018a].

The most common version of PoW is to calculate a SHA-256 hash value with certain characteristics; for example, the hash value must start with a given number of zeros. This is not an easy task and implies the exploration of multiple combinations of values in order to find a viable solution. The average work required is exponential in relation to the number of required zero bits and can be verified by executing a simple hash operation [Nakamoto, 2008].

In PoW implemented by bitcoin, the difficulty of the puzzle is adjusted every time that 2016 blocks are added, so that the average speed to add one new block in the chain is one (1) block every ten minutes [Nguyen and Kim, 2018a]. When a new block is created, the header information is combined and sent as an input parameter to the SHA-256 hash function [Bitcoinwiki, 2016]. If the output of this function is below a threshold T (which depends on the difficulty), then the value sought is accepted. Otherwise, the node must continue calculating the secret value until the output of the SHA-256 function is accepted. The difficulty of the puzzle increases as the value T becomes smaller [Nguyen and Kim, 2018a].

2.1.2 PoS-based

The Proof-of-Work algorithm is not fair for all miners, because not all have the same hardware configuration. Some have modern equipment, while others have very basic equipment to process data and information, therefore the former will have an advantage, given that solving the puzzle is computationally intensive. The algorithms based on Proof-of-Stake (PoS) seek to deal with this inequality. The basic principle of the PoS algorithms is to use the idea of a bet or participation magnitude, to define which node will have the opportunity to mine the next block in the chain. Using participation as evidence has an advantage: any node that has had a lot of previous participation is more reliable, and thus it is expected that this node will not perform any fraudulent activity to attack the chain that contains a large part of its profits. Also, the use of PoS implies that there has to be at least 51% of all bets in the network, to perform a double-cost attack, which is very difficult. There are currently two popular types of consensus that use PoS: those that use pure participation to obtain consensus and the hybrids that combine PoS and PoW [Nguyen and Kim, 2018a].

2.1.3 Hybrid form of PoW and PoS

[Sunny and Scott, 2012] proposed a new concept called the coin age of each miner, which is calculated by his bet multiplied by the time the miner owns it. For a node to get the right to add a new block to the chain, it creates a special block called coin stake, which contains many transactions, but also includes a special one from that miner to itself. The amount of money spent on the transaction gives the miner more possibilities to mine a new block, then solve the puzzle, as in PoW. The more money is spent on the transaction, the easier it is to solve the puzzle. When the puzzle is solved, the mining node gets 1% of the amount of the coins that they have spent in the transaction, but the accumulated coin age by these coins is reset to zero (0) [Nguyen and Kim, 2018a].

Unlike the previous proposal, [Vasin, 2014] does not use the coin age in his blockchain, because it is assumed that, by making use of the coin age, the attacker can be given the possibility to accumulate enough value to deceive the network. Another problem is the possible existence of some miners who keep their bet until they have a large number of coins, while they remain outside the verification system; therefore, the proposal by [Vasin, 2014], is to use pure participation in exchange for the age of the currency to offer miners the possibility of mining a new block. This may encourage more nodes to be online to obtain profits. When the existence of off-line miners is untied, [Ren, 2014] proposes to use an exponential decay function with the coin age, in which, when the miner waits for the increase in the coin age, less is the speed of increase. [Duong et al., 2017] propose a method that combines PoW and PoS, which will be explained in the

next section as a way to also mitigate the 51% attacks.

2.1.4 Other kinds of proof-based consensus algorithm

One of the main problems of PoW is the excessive energy demands required to find the nonce, besides the fact that this calculation is disposable and does not provide any long-term benefit to the users. This was presented by [Blocki and Zhou, 2016] and by [Sunny, 2013]. To address this issue, [Blocki and Zhou, 2016] proposed the use of some types of puzzles for education and social activities, which were easy to solve for computers but difficult for people to solve; thus, the effort to solve the puzzle to mine a new block corresponds to people and not in using hardware. This is fairer for all because not all miners can invest in modern hardware [Nguyen and Kim, 2018a].

Different authors have proposed other evidence-based consensus algorithms that do not use the idea of PoW and PoS. Examples of these are: Proof-of-Burn in [P4Titan, 2014], Proof-of-Space in [Park et al., 2018], Proof-of-QoS (PoQ) in [Yu et al., 2019] and A Fair Selection Protocol [Liu et al., 2020]. In Proof-of-Burn, the miners send their coins to a direction to be burned, in this way these coins can not be used by others, so the miner, who burns most coins, earns the right to mine a new block. On the other hand, the miners of Proof-of-Space must invest in hard disks for their computers, which in comparison with the hardware required in PoW is much cheaper. The Proof-of-Space algorithm generates large data sets called plots on the hard disk, so the more data a node has, the more likely it will be able mine a new block. In PoQ the network is divided into small regions; each of these chooses a node based on its QoS. Then, a deterministic Byzantine Fault Tolerance (BFT) consensus is run between all the chosen nodes. The goal of PoQ is to achieve very high transaction throughput as a permissionless protocol and to provide a fairer environment for participants. The Fair Selection Protocol is composed of two main phases: the mining process and the confirmation of the new nodes list. More consensus techniques can be found in [Aponte et al., 2021a], where a new classification method was proposed.

2.1.5 Alternative Protocols

Alternative consensus protocols have originated as responses to different efforts to improve the traditional consensus protocols. The works carried out on these alternative protocols are little known. This is evidenced by the number of recent scientific publications. In [Oyinloye et al., 2021], they made a general description of alternative consensus protocols proposed between 2019 and 2021. They classified the alternative consensus protocols according to how the winner node was selected as follows: Consensus Proto-

col based on Effort or work (CPE), Consensus Protocol based on Wealth or resources (CPW), Consensus Protocol based on Past Behavior or reputation (CPPB), and Consensus Protocol based on Representation (CPR). Furthermore, the authors of [Saad et al., 2021] presented a modular version of PoS-based blockchain systems called e-PoS, which resisted the centralization of network resources by expanding mining opportunities to a more extensive set of participants. In addition to the few publications on the subject, the interest in studying these protocols lies in their design features, which can guide future research.

2.2 Techniques for 51% attack prevention

This section presents some works focused on mitigating the 51% attack on Blockchain networks.

The double-spending attack considers a high risk for the security of a blockchain, when the miners own more than 51% of the mining power, generating an alert. To mitigate this problem, [Duong et al., 2017] proposed a method combining PoW and PoS (2-hop Blockchain). The objective of this method is to make sure that, even if a miner owns more than 51% of the mining power, he or she will not have many possibilities to carry out a fraudulent action. To achieve this, the authors propose using a PoW first to choose a winning node, which is the first to solve the puzzle. Next, this node, in addition to adding a block called PoW Block to the chain, provides a basis for choosing another miner who has a bet. If the return value of the hash function that has input parameters of the newly added PoW Block and the private key of the owner of the bet is below a threshold, the chosen miner will have the possibility to add the PoS Block to the chain.

[Scicchitano et al., 2020] proposes an encoder-decoder deep learning model to detect anomalies in the use of blockchain systems. The contributions of this work are the following: a) the identification of a relevant set of characteristics calculated in blockchain registers that describe the state of the network in certain time steps; and b) The use of a sequence-by-sequence neural network model to recognize anomalous changes in the blockchain network.

Due to the uniqueness of the attacks and how this makes them hard to identify and detect, [Scicchitano et al., 2020] adopt an unsupervised approach to address this problem. Also, they propose as future work to study the use of hybrid architectures based on the combination of Recurrent Neural Networks (RNN) with convolutional neural

networks [O'Shea and Nash, 2015] to perform the feature-selection process and assess eventual improvements. In addition, they consider defining models capable of predicting attacks before they occur, improving network security.

[Yang et al., 2019] proposes a method to mitigate 51% attack on Proof-of-Work blockchains based on weighted history information. In this approach, the authors use the frequency rate of miners in historical blocks and calculate the total weighted historical difficulty to establish if a branch change is required. The proposed protocol is called "Proof of Work based on historical weighted difficulty" (HWD-PoW), and its analysis indicates that the cost of the attack increases by two orders of magnitude when the new technique is implemented.

[Dey, 2018] proposes a methodology in which intelligent software agents can be used to monitor stakeholder activity in blockchain networks to detect anomalies, such as collusion, by making use of a supervised machine learning algorithm and algorithmic game theory, to mitigate the majority or the 51% attack.

Horizon proposes a delayed block sending penalty system [Garoffolo et al., 2018]. This proposal suggests modifying the Satoshi consensus protocol (PoW) to secure a network against 51% attack. The sanction applied is determined based on the time the attacking node is hidden from the network. This technique notifies the entire network about the fork, and during that period, participants, miners, and exchanges cannot transact until the delay period is removed. The penalty system is a research prototype technique that has not yet been implemented in a real network, and it also includes several limitations. According to [Rosenfeld, 2014], when an attacker owns 51% of the network's hash, he will always succeed regardless of the imposed delay. Consequently, the possibility of carrying out the 51% attack when this security mechanism is in place is very large. Also, the delay process slows down the general transactions of the network and strongly impacts the usual transactions because the delay blocks will not be confirmed until the penalty is lifted. This fact makes this technique not very appropriate to be implemented in a real network, and it is not completely effective against a 51% attack.

[ChainZilla, 2019] proposes a Kodomo security solution called "delayed proof of work" (dPoW). This solution is implemented for cryptocurrencies based on UTXO. This security technique is already implemented in some blockchains to safeguard against double-spend attacks. The main attribute of dPow is that it does not recognize the rule of the longest chain; consequently, attacks that are intended to be carried out in private cannot gain an advantage to double spend. For its operation, dPow chooses 64 special nodes each year to acquire information from Komodo and store it on the bitcoin blockchain. The strengthened security orientation of this proposal intends for the attacker to rewrite the Komodo chain and bitcoin checkpoints. Likewise, the attacker

must also be able to influence the majority of the notary network. This makes the technique robust. However, the limited number of special nodes makes the security technique centralized, which leads to the known problem of a "single point of failure", where attackers know exactly what to attack. Another limitation that dPoW presents is that it is only implementable in cryptocurrencies based on UTXO and it is not profitable since it requires an implementation fee. In addition, the participating nodes of the network must wait an explicit amount of time for the notarization process to be completed, which can discourage certain participants who intend to make a faster transaction. The notarization process is carried out every 10 minutes, which gives attackers a window of time to carry out the 51% attack in cryptocurrencies given that the block confirmation time only needs a few seconds [Komodo, 2018].

PirlGuard is a security protocol developed to mitigate the 51% attack, this approach modifies the consensus algorithm to protect itself from a 51% attack [Minchev, 2018]. This protocol is based on the attributes of the Horizen penalty protocol (system delay block send penalty), but it is built primarily for Ethash. When the network detects longer blocks extracted privately, PirlGuard abandons the node instantly by penalizing the extraction of x number of blocks, based on the total number of blocks extracted secretly. The PirlGuard approach employs notarial contracts that are controlled by master nodes; these master nodes are in charge of notarizing the blockchain and penalizing malicious nodes by regaining legitimate consensus on the Pirl blockchain. As in the Komodo solution, PirlGuard also employs master nodes, a feature that makes the security technique centralized, leading to the known problem of a "single point of failure". Another limitation of this solution derives from the fact that the penalty is not a final solution to protect against the 51% attack, as there is a probability that attackers with a hash rate of 51% will be able to overcome that penalty.

ChainLocks [Block, 2018] is another security technique developed to protect DASH, based on the implementation of "long-living master node quorums" (LLMQs) to mitigate the 51% attack. This technique includes a network-wide voting process that comprises a "first-look" policy. For each of the blocks, an LLMQ of a large number of master nodes is approved. All participating nodes in the network are required to sign the designated block to extend the active chain. While at least 60% of the network participants verify a block, they generate a P2P message (CLSIG) to notify all other nodes about the event. This CLSIG message cannot be generated unless enough members of the network comply, so it implies a valid signature of authenticity and verifiable by the nodes within the network. Because only one confirmation is required for the publication of a block, attackers with at least 51% hashing power have a chance to double-spend, making the Dash blockchain vulnerable. In addition, the main disadvantage of ChainLocks is that it is designed only for the Dash cryptocurrency, which has

a low network hash. This feature, in addition to the master node approach, makes it a weak security approach allowing the possibility of a 51% attack simply by renting the required hashing power [Ng, 2018].

Merged Mining is not a security technique, but a method that allows merging several cryptocurrencies with mining at the same time. Low hashing power cryptocurrencies that share the same consensus can benefit from merged mining to improve their security [Cryptocompare.com, 2015]. The merged mining process makes it possible to increase the hashing power by starting in the other coin that comprises a higher hashing power. Although cryptocurrencies take advantage of merged mining, transactions on both networks can run in sequence. Blockchains are classified as main and auxiliary. In addition to improving security, another benefit is the ability for miners to mine more than one block simultaneously. Although merged mining increases security on blockchains, the process is not straightforward and is very often neglected by miners. The main limitation of this method is that the cryptocurrencies that take advantage of this approach must be in the same consensus protocol and mining algorithms [BiXBiT, 2018]. Another limitation is that, if two low-hashed cryptocurrencies are combined, it is possible to exploit them as long as the attacker achieves the required hashing power. Consequently, merged mining is only a process to increase the cost of the attack by merging hashing power, and does not provide an effective solution to the 51% attack.

Sayeed & Marco-Gisbert present a novel technique called "Proof-of-Adjourn" (PoAj), whose objective is to mitigate the main blockchain attacks and the problem of delay in the processing of transactions with large transactions in cryptocurrencies passed in UTXO. This proposal does not recognize the longest chain to verify the authenticity of the chain; instead, a deferral period is imposed to regulate the verification of the block, although miners with high hashing power could have an advantage in the mining process, the transmission of more than one block will disqualify the block from being included in the chain by abstaining from mining activities for some time. The security of this proposal lies in eliminating the possibility of block reversion. PoAj confirms transactions with just one confirmation eliminating the waiting time of six confirmations brought by PoW. This leads to a much faster transaction confirmation rate compared to many existing consensus protocols. Similarly, PoAj introduces a unique approach that is activated when there is more than one block transmitted within a predefined period of time. This approach is unique and, so far, it is the first approach to fully solve the problem. This proposal is not found in any cryptocurrency; the authors implemented a proof of concept of the PoAj consensus protocol in the Python programming language.

The table **2-1** presents the advantages, risks, vulnerabilities, implementation cost, and the working of the techniques for 51% attack prevention presented in the analyzed works.

Table 2-1: Main characteristics of the analyzed Techniques for 51% attack prevention

Technique	Advantage	Risks	Vulnerability Identify	Cost	Working
2-hop Blockchain - [Duong et al., 2017]	Even if a malicious node manages to control more than 50 hash rate, honest nodes may still defend the blockchain through honest participation	higher centralization and resource consumption	May not identify the vulnerability in advance	Its implementation may involve a cost	TwinsCoin
Encoder-decoder deep learning mode - [Scicchitano et al., 2020]	Use of a sequence-to-sequence neural network model to recognize anomalous changes in the Blockchain network	Wrong detection	Detects anomalous situations and trigger an alert	Its implementation may involve a cost	Any Blockchain with sequential data
Proof of work based on historical weighted difficulty (HWD-PoW) [Yang et al., 2019]	Increase the cost of the attack by two orders of magnitude	Is not effective against a slow gradual increase of hash rate, may affect the privacy and security of miners	May not identify the vulnerability in advance	Its implementation may involve a cost	Blockchain-based on Proof-of-Work (PoW)
Methodology with intelligent software agents - [Dey, 2018]	Implementation of an intelligent agent in the application layer of the blockchain network system	Wrong deducting the level of importance of the product	Could be identified before it is adopted by the main chain	It is just a proposed methodology	Blockchain-based on Proof-of-Work (PoW)
Delayed block sending penalty - [Garofolo et al., 2018]	It forces the attacker to privately mine a large number of consecutive blocks before being able to join the main chain, consequently carrying out the attack is much more expensive	It is not enough to completely mitigate the attack, it is a very weak solution for a cryptocurrency with low hashing	It is identified before it is adopted by the main chain	It does not require, the technique is a prototype	Blockchain-based on Proof-of-Work (PoW)
Delayed Proof-of-Work (DPoW) - [ChainZilla, 2019]	Ignore the longest chain rule, the use of notarial nodes adds security to the protocol	A malicious node with at least 51% hash rate could execute an attack within the notarial period	May not identify the vulnerability in advance	There is a cost to adopt this technique	Any blockchain-based on Unspent transaction output (UTXO)
PirlGuard [Minchev, 2018]	Employ a penalty system for attack nodes	Using master nodes for notarization weakens the network due to a single point of failure	Identify vulnerability as soon as it is recognized	Its implementation may involve a cost	Blockchain based on Ethash Algorithm
ChainLocks - [Block, 2018]	Very fast transaction confirmation	Confirmation with a single block may enable double-spending with a low hash rate	Lock the first block as a legitimate block, discarding any other blocks	Require a cost for its implementation	Dash CryptoCurrency
Merged Mining - [Cryptocompare.com, 2015]	Merges the hashing power of two blockchains, making the attack more costly	A malicious node with a high hash rate may perform an attack	May not identify the vulnerability in advance	No cost for implementation	Auxiliary Proof of Work (PoW)
Proof-of-Adjourn - [Sayeed and Marco-Gisbert, 2020]	Offers sufficient protection to the network regardless of the hash rate of the attackers. shorter confirmation time for very large transactions	Participants with ample hash rate may get an advantage through the mining process	Is identified before it is adopted by the main chain	It is a proposal, a proof of concept was made	Any blockchain-based on Unspent transaction output (UTXO)

All these techniques assume a wide decentralization of the nodes, but the reality of this assumption has not been evaluated in the literature and will be addressed in this work.

3 Cluster-Based Classification of Blockchain Consensus Algorithms

In recent years, blockchain has become a disruptive technology to protect the integrity of information, especially in open and collaborative information systems. Its main advantage is the possibility of reaching a consensus on the new data blocks to be added to the chain, even with anonymous actors. The most common consensus mechanism is Proof of Work, but it has been proven very inefficient in terms of energy spent by the blockchain members. In the literature, there are many other techniques that pretend to become the new popular mechanism. However, the number of these techniques is growing too fast to differentiate among all the options. This chapter proposes a new characterization of consensus algorithms that can be used to find families of mechanisms using cluster-based classification. Using the Ward Method and Spearman's Rank Correlation analysis, new clusters of consensus mechanisms were identified. The results describe the behavioral patterns not seen before in the literature. In addition, some open problems of current consensus algorithms are discussed.

The study of blockchain consensus algorithms is a growing field in which research can be carried out in their operation, organization, problems, and other aspects. Therefore, in this chapter, a theoretical categorization based on the working mechanisms is carried out. After that, a detailed explanation of how the information was labeled and organized is presented. Finally, statistical analyses such as correlations, ordering, and clustering analysis were performed to understand not only how the attributes describe each consensus algorithm, but also propose a new categorization based on the results obtained.

3.1 Related work

[Du et al., 2017] present the basic principles and characteristics of consensus algorithms: *Proof of Work, Proof of Stake, Delegated Proof of Stake, Practical Byzantine Fault Tolerance and Raft.* The features they compare are: *byzantine fault tolerance, crash fault tolerance, verification speed, throughput, and scalability.* Additionally, they describe the

limitations of these algorithms. The authors suggest which algorithm should be used according to the type of blockchain, in the case of public blockchains: PoW, PoS, and DPoS. In private blockchains: PBFT and RAFT and in authorized blockchains: PBFT. The authors do not present a classification proposal, they rely on the existing literature about the types of blockchain.

On the other hand, [Bach et al., 2018] describe the alternatives to solve the consensus problem in distributed systems. They address the problem of Byzantine generals, BFT, and dBFT. The authors present the ten most profitable cryptocurrencies until the year 2018. They also make a description of the algorithms *PoW, Ripple, PoS, Stellar, dPOS, and Proof of Importance*. The comparison criteria they use are: *energy saving, tolerated power of adversary, and scalability*. They claim that PoW and PoS algorithms are the most widely used, however, they mention that hybrids of PoW and PoS can be used. Additionally, they introduce two new algorithms that were not in the public domain in 2018: *Proof of Luck (PoL)* and *Proof of eXercice (PoX)*. This work is oriented towards the evaluation of the most popular algorithms according to the criteria *energy saving, tolerated power of adversary and scalability*. The authors do not present a categorization of the algorithms.

[Nguyen and Kim, 2018b] describe the concept, architecture and characteristics of blockchain. They also classify consensus algorithms into two groups: proof-based and voting-based, the latter subdivided into *Byzantine fault tolerance-based consensus and Crash fault tolerance-based consensus*. The algorithms of the first group that they present are: original and variants of *Proof of Work, Proof of Stake*, hybrid of PoW and PoS, *Proof of burn, Proof of Space, Proof of Elapsed Time, Proof of Luck and Multichain*. Nguyen and Kim compare PoW, PoS, and hybrid PoS and PoW based on the criteria: *energy efficiency, modern hardware, forking, double spending attack, block creating speed, and pool mining*. The algorithms of the second group that they describe are: *Hyperledger with practical Byzantine fault tolerance, Ripple, Stellar and Chain*.

In the same way as [Nguyen and Kim, 2018b], [Zheng et al., 2017] describe the concept, architecture and characteristics of blockchain. They present three types of blockchain: public, consortium and private. The comparison criteria used for the blockchain types are: *Consensus determination, Read permission, Immutability, Efficiency, Centralized and Consensus process*. In this study they address the consensus algorithms: *Proof of Work, Proof of stake, Practical byzantine fault tolerance, Delegated proof of stake, Ripple and Tendermint*. They compare consensus algorithms based on the following criteria: *Node identity management, Energy saving, and Tolerated power of adversary*. They also list the challenges facing this technology. The objective of the work is not to present a categorization of the consensus algorithms but to compare them according to three criteria.

[Sankar et al., 2017] presents the three types of blockchain: public, consortium and private. The aim of the authors is to present the *Stellar* consensus protocol and compare it with the *Corda and Hyperledger Fabric* platforms. The comparison criteria they mention are *view transactions and latency*. SCP uses the concept of quorum segments which grants users more freedom to decide which participants are trusted. Corda maintains records of various business and financial contracts. The Hyperledger project allows various blockchain technologies to interconnect and ensures a secure *plug and play* environment for them. The hyperledger does not provide users with as much freedom as the SCP. The focus of [Sankar et al., 2017] is to present how the *Stellar* consensus protocol works and how it interacts with *Hyperledger Fabric*. It is not intended to classify consensus algorithms.

For their part, [Chaudhry and Yousaf, 2019] present a generic architecture and the categorization of consensus mechanisms in distributed systems. Based on criteria such as: *scalability, communication model, category and failure models*. Additionally, they express that the specific categorization of blockchain is divided into two groups: *proof-based consensus and vote-based consensus*. The paper describes a proposal to evaluate consensus algorithms taking into account the parameters of: blockchain *type, transaction rate, scalability, adversary tolerance model, experimental setup, latency, throughput, bandwidth, communication model, communication complexity, attacks, energy consumption, mining, consensus category and consensus finality*. Based on the above criteria, they evaluate the algorithms: *ELASTICO, Leader-free Byzantine Consensus, Implicit Consensus, Proof of trust (PoT), DBFT Consensus Algorithm, PoPF, Ripple, Proof of Vote (PoV), and Proof of Work (PoW)*.

[Hao et al., 2018] propose a method to evaluate PBFT and PoW algorithms on *Ethereum and Hyperledger Fabric* platforms. The testing architecture is made up of three modules integrated into *Yahoo Cloud System Benchmark (YCSB)*. The two performance evaluation metrics used are: *Latency and Throughput*. Hao et al. conclude that PBFT adapted for *Hyperledger* performs better than PoW for *Ethereum* in terms of average throughput and delay. The focus of this work is the evaluation of two consensus algorithms in private platforms, not the categorization of the algorithms.

The objective of [Arjun and Suprabha, 2020] is to present the results of a systematic literature review focused on three fronts: the first, identification of solutions for the banking industry. As theoretical models or empirical frameworks with potential for strategic change, operational advantages or functions of the stock market. The second approach was to find documents that highlighted the practical scope or challenges in platforms, legal, technical and organizational dimensions and the third front, to select documents oriented to specific applications that used experimental conceptual environments. The repositories they reviewed were: *Scopus, Web of Science, ACM, IEEE,*

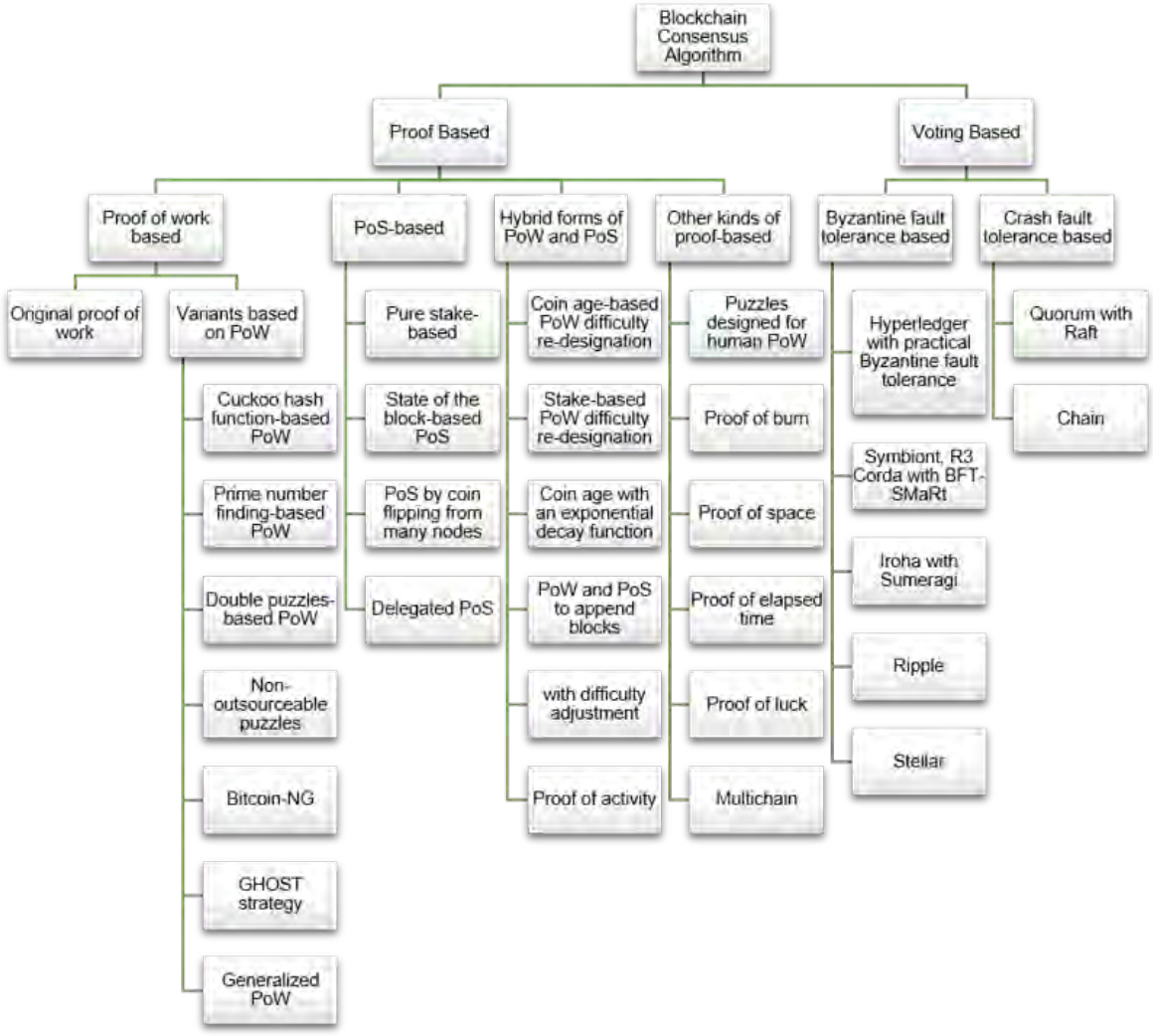


Figure 3-1: Organization of consensus algorithms based on [Nguyen and Kim, 2018b]

AIS. Among the most relevant findings are studies on: banking, information systems, innovation, law, finance, sustainability, entrepreneurship and digital infrastructure.

Taking into account the consulted literature, Fig. **3-1** proposes a basic organization of the consensus algorithms according to their working mechanism. The said organization responds to the categorizations, considering the bibliography. However, it does not consider the variations some derivatives have from the base algorithms from which they descended.

Table 3-1: Features and values used to cluster.

Feature	Description	Value
Energy efficiency	Efficient energy used to perform the tasks of the protocol	Yes = 0 No = 1
Modern hardware	Modern hardware requirements	No need = 0, Low need = 1, Need = 2, High = 3
Forking	Possibility to perform a forking of the main chain	Never = 0, Very difficult = 1, Difficult = 2, Probably = 3, Very Probably = 4, Never = 0
Double-spending attack	Possibility of a double-spending attack	Difficult = 1, More or less = 2, Easy = 3
Block creation speed	Speed to create a block	Very fast = 0, Fast = 1, Low = 2, Very low = 3
Mining Pool	Amenable to mining pool creation	Never = 0, Very difficult = 1, Can be prevented = 2, Difficult to prevent = 3, It occurs = 4
Number of participants	Number of participants in the protocol	Mostly unlimited = 0, Limited = 1
Decentralization	Decentralization of participants	Mostly high = 0, Low = 1
Trust	Trust of the network	More trustful = 0, Less trustful = 1
Node Identities	Node identity management	No = 0, Yes = 1
Security threat	Security threat to network	More serious = 0, Less serious = 1
Award-giving	Award-giving to miner nodes	Yes = 0, Mostly no = 1

3.2 Theoretical Categorization Based On Work Mechanisms

3.2.1 Organization of information

The characteristics of the consensus algorithms evaluated are qualitative variables proposed and described by several authors in previous studies [Du et al., 2017, Bach et al., 2018, Nguyen and Kim, 2018b, Zheng et al., 2017, Sankar et al., 2017]. Taking this information as a reference, an assessment was defined in numerical categories to apply statistical functions on them. We first labeled each of the characteristics of the consensus algorithms, giving a lower value than the desired condition based on the efficiency of each attribute; and increasing said value when the less favorable condition is obtained (Table **3-1**).

3.2.2 Correlation analysis between the characteristics evaluated

Statistical analyzes were generated using the Past program [Hammer Øyvind, Harper David A.T, 2001]. First, a Spearman rank correlation analysis was performed, which is a non-parametric method that quantifies the relationship between two descriptors; defining a perfect correlation of these when the ranks of all the objects are the same in both elements [Legendre, 2012, Franco et al., 2014]. Additionally, the Bonferroni [Cox and Donnelly, 2011] correction was applied to avoid type I error (finding significant differences when they do not exist). The results are shown in Fig. **3-2**, where all the blue ellipses signify a significant and positive correlation between the variables ($p < 0.05$), while the red ones indicate significant negative correlations between the variables.

Most of the variables show significant correlations. Specifically, the parameters *number of nodes executing, decentralization, trust, nodes identities are managed, security threat*, and *award*, show a significant correlation between them (green box in **3-2**), corresponding to features that have ideal efficiency conditions (zero values), from proof-of-work algorithms.

For their part, the parameters *energy efficiency, modern hardware, forking, double spending attack, pool mining, agreement making basement*, and *nodes can join freely*, also show a significant correlation between them (yellow box in Fig. **3-2**), corresponding to the characteristics that have the ideal efficiency conditions (zero values), of the vote-based algorithms.

Two parameters with few correlations are also evident. The first is *block creating speed*, which only shows a significant correlation with the *energy efficiency* and *forking* parameters, since although the block creation speed does not have ideal values in almost no algorithms (except *proof of luck*), show similar trends such that when a value is greater in one of them, it is also greater in the other. *Modern hardware*, despite also presenting its ideal conditions (zero-based values) in vote-based algorithms; It presents a variability of its efficiency in the algorithms based on tests. In some of them (*Pure PoS (Nextcoin), State of the block-based PoS, PoS by coin flipping from many nodes, Delegated PoS, Puzzles designed for human PoW, Proof of burn*), it even has the values ideals. For this reason, it presents few significant correlations with the other parameters.

To summarize, the ideal-valued features of evidence-based algorithms are directly related to each other, but are inversely correlated with the higher-efficiency variables of voting-based algorithms. Furthermore, it can be seen that the *block creating speed* feature is a variable that is only significantly correlated with the *energy efficiency* and *forking* features. Likewise, *modern hardware* only correlates with *energy efficiency*, *forking*, and *double spending attack*; while *pool mining* does not correlate with *energy*

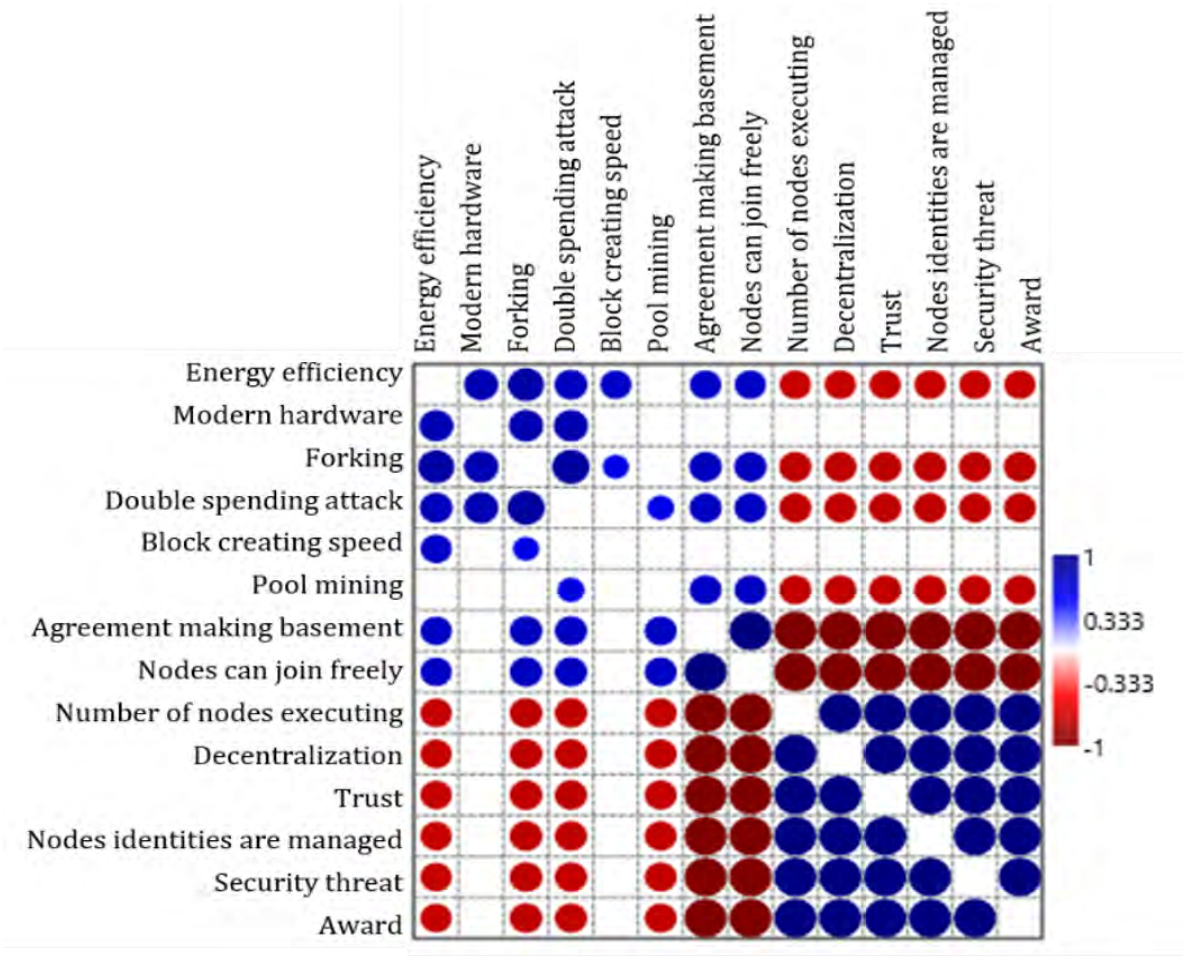


Figure 3-2: Results of Spearman's rank correlation analysis, blue ellipses indicating positive significant correlations, and red ellipses indicating negative significant correlations ($p<0.05$). Blank spaces indicate no significant correlations ($p>0.05$).

efficiency, *modern hardware*, *forking*, and *block creating speed* (Fig. **3-2**).

3.2.3 Variability of the characteristics of the working mechanisms of consensus algorithms in blockchain

Subsequently, a Principal Component Analysis (PCA) was carried out, which is a multivariate ordering technique that seeks to reduce the dimensionality of the data, based on the correlations, to find the factors that can explain the set of information. The ordering is defined through factors which are called "principal components". These components define the axes of the original rotation of the coordinate system, and cor-

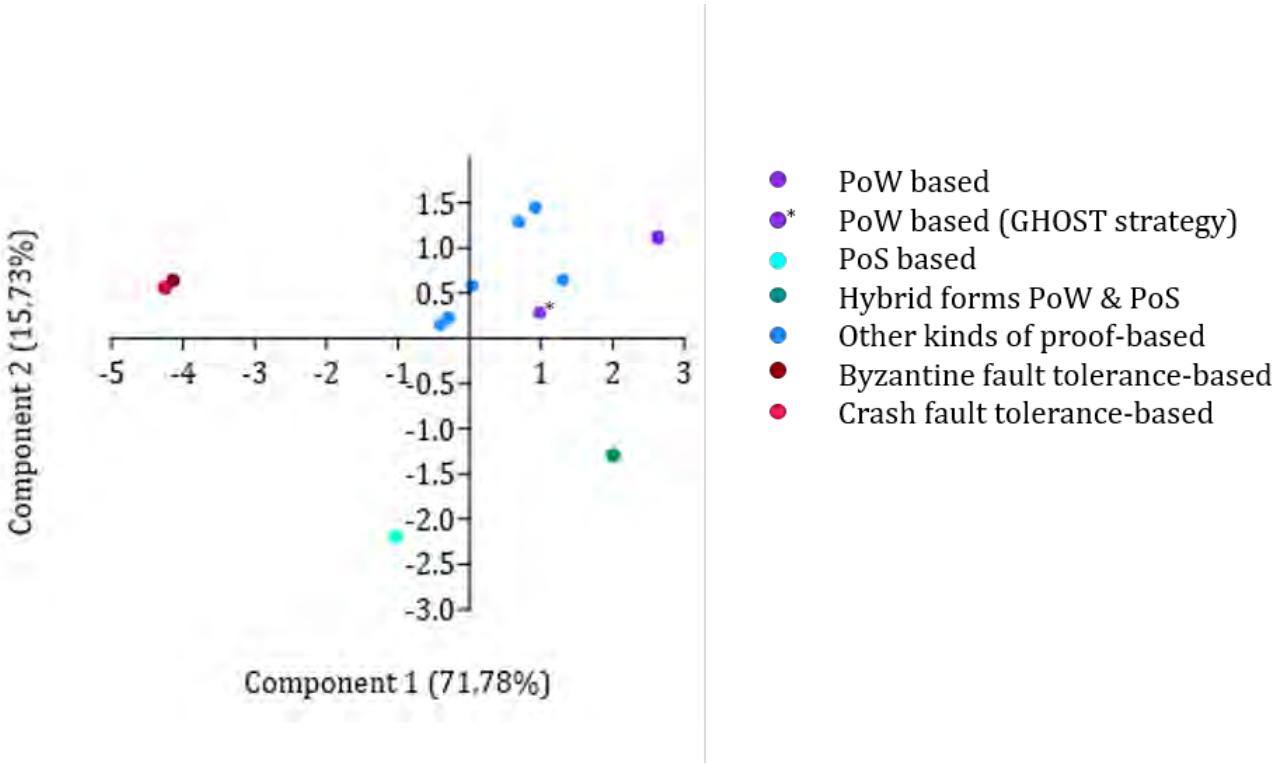


Figure 3-3: Analysis of principal components of the analyzed consensus algorithms.

respond to the successive directions of the maximum variances of the dispersion of the points; placing them in a new coordinate system [Legendre, 2012, Catena et al., 2003].

This analysis is visualized in two-dimensional graphs for ease of interpretation, where the first component will always explain the greatest variability of the data, and the explained variance will decrease as the component increases (Legendre and Legendre, 1998) [Carmen and Rafael, 2013]. The PCA allowed to identify the variability between the attributes that describe the working mechanisms of the consensus algorithms in the blockchain. The first two components (Fig. **3-3**) describe 87.5% of the variance among all the variables (PC1 = 71.78%, PC2 = 15.73%). The first component shows that the greater variability between the variables contained in each algorithm separates most of the PoW-based algorithms (purple dots) that are at the positive end of the first axis, from the voting-based algorithms (red dots), which are at the negative end of the axis. Hybrid algorithms (green dots) are closer to pure Pow algorithms (purple dots) and other proof-based algorithms (dark blue dots) (Fig. **3-3**).

The GHOST strategy algorithm (purple dot) is the only PoW algorithm separate from the others in the same category, located closer to another type of algorithm based on type tests (dark blue dots). On the other hand, PoS algorithms (light blue dots) are

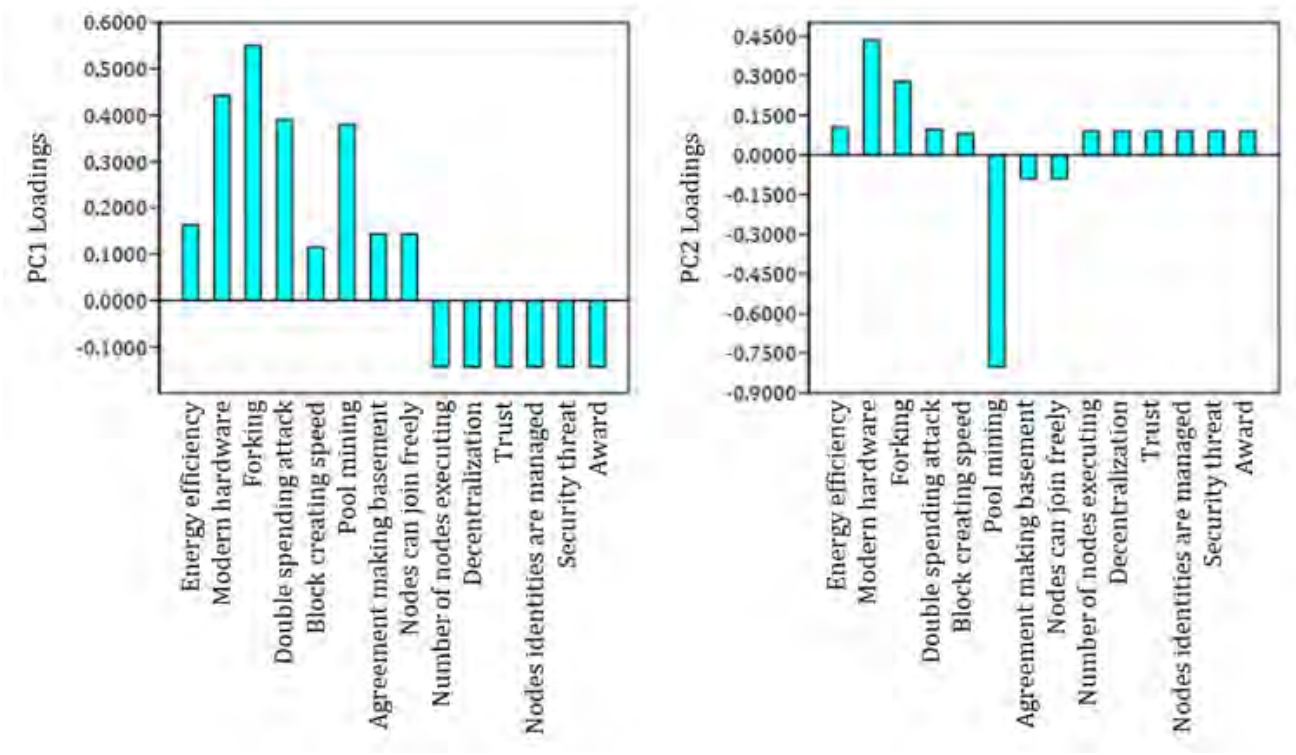


Figure 3-4: Attributes loads for the first (PC1, left) and second (PC2, right) components of the PCA

placed on the negative side of the first axis, they are placed closest to voting-based algorithms (red dots) among all proof-based algorithms (Fig. **3-3**).

The first principal component neatly separates job-based consensus algorithms (on the right) from voting-based algorithms (on the left). This separation is clearly demarcated by the marked difference in the attributes *number of nodes executing, decentralization, trust, nodes identities are managed, security threat, and award*; of which algorithms based on evidence have the highest efficiency with respect to those based on voting.

Additionally, the difference between the algorithms of the positive (*PoW based*) and negative (*voting based algorithms*) extremes of the first axis (Fig. **3-4**), shows that the differences are mainly due to the characteristics of *modern hardware, forking, and double spending attack*, in which the *byzantine fault tolerance based and crash fault tolerance based algorithms* (based on voting) have the highest efficiency values , while *PoW based* have the worst (supplemental material).

The second principal component, which explains 16% of the variance, makes it possible to separate the algorithms *PoS based and Hybrid forms PoW & PoS* (negative extreme), from the rest of the algorithms analyzed, depending on whether they have the worst

values of efficiency with respect to the *Pool Mining* parameter (Fig. **3-4**).

3.3 Proposed Categorization

Based on the information reviewed in previous analyses, a grouping analysis (also called cluster analysis) was developed, which seeks to classify similar individuals or variables among themselves, without an *a priori* classification criterion. The method consists of a multidimensional analysis that divides a set of objects or descriptors, under study. The aforementioned division is made into subsets in which the elements are grouped based on association matrices that depend on their characteristics. The method is used as a measure of object association [Legendre, 2012, Catena et al., 2003].

Within the grouping methods, working with the minimum variance method, also known as Ward's method; which seeks to obtain the least intracluster variability, in order to make each group as homogeneous as possible. This homogeneity is measured by the sum of the squares of the differences between the subjects within the cluster [Legendre, 2012, Catena et al., 2003, Sarabia Alegría et al., 2018]. With this method, the proposed categorization ordering of the consensus algorithms based on the working mechanisms analyzed in this document was obtained. The results of the categorization are shown in Fig. **3-5**, where it can be seen that most of the groups are similar to those currently proposed in the literature (Fig. **3-1**), but with some important differences.

According to the proposed categorization, PoW algorithms are correctly grouped into a single category, with the exception of the *GHOST strategy* algorithm, which differs from them by being much more efficient due to the bifurcation feature. The proposed categorization also maintains the grouping of the hybrid forms of PoW and PoS; allowing to identify that taking into account the characteristics analyzed, the hybrid algorithms are closer in their attributes to the pure PoW algorithms than to the pure PoS. The reason for this is that Proof-of-Stake (PoS)-based algorithms show a greater efficiency in their attributes: *energy efficiency, modern hardware, forking, Double spending attack, and block creating speed*, than the other two categories. (PoW and hybrids).

Likewise, the proposed classification preserves the current classification of the *PoS algorithms (Pure PoS (Nextcoin), State of the block-based PoS, PoS by coin flipping from many nodes and Delegated PoS).*

One of the main contributions of the proposed classification is based on the separation of the algorithms *Hyperledger with practical Byzantine fault tolerance and Symbiont, R3 Corda with BFT-SMaRt*, based on the less efficient block creation speed, with respect to the other grouping of algorithms made up of: *Iroha with Sumeragi, Ripple, Stellar,*

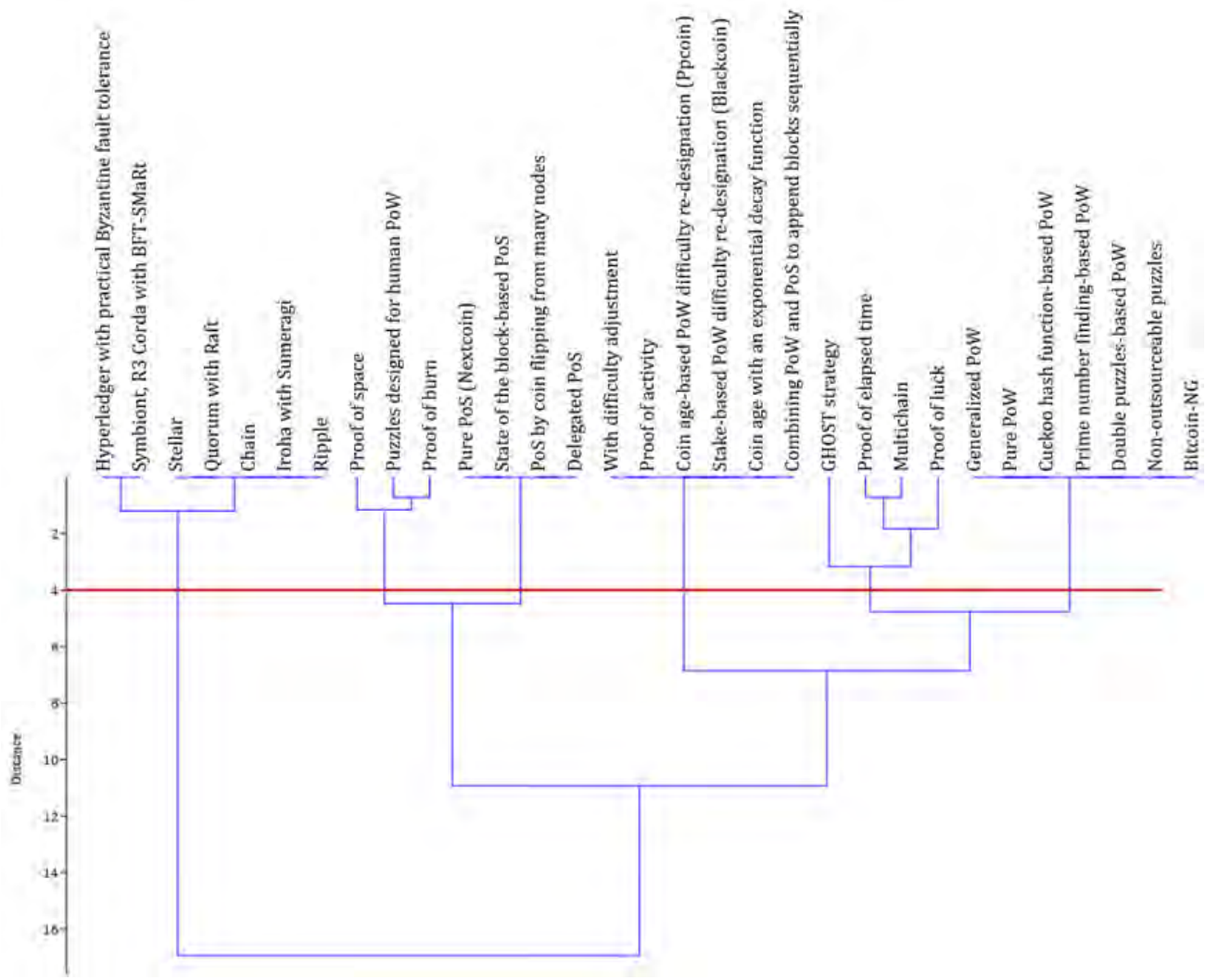


Figure 3-5: Cluster analysis of the blockchain consensus algorithms based on the work mechanisms attributes

Quorum with Raft, and Chain.

Another contribution considers the recategorization of the algorithms previously classified in the category of "other evidence-based algorithms". The first grouping considers the similarity between the *Puzzles designed for human PoW, Proof-of-Burn and Proof-of-Space* algorithms, based on *modern hardware* which is much more efficient than the other algorithms. This efficiency attribute also places them closer to PoS algorithms. The other categorization proposes that the *Proof-of-Elapsed-Time, Proof-of-Luck and Multichain* algorithms, together with the *GHOST strategy* algorithm, form an independent grouping. This grouping is the one that should be taken with the greatest caution

within the proposed classification, since it is the one that considers the greatest distance in the analysis, and its data show high variability.

The work presented by Nguyen and Kim [Nguyen and Kim, 2018b] has a high degree of similarity with this work, because they divide consensus algorithms into proof-based and vote-based algorithms. There is also agreement on the comparison characteristics of the consensus algorithms. However, Nguyen and Kim do not present categorization in the way it is done in this paper. For their part, Chaudhry and Yousaf in their work [Chaudhry and Yousaf, 2019] propose a categorization of consensus algorithms in distributed systems and provide a list of relevant parameters to evaluate consensus algorithms, useful for the design and evaluation of algorithms.

3.4 Conclusions

This chapter analyzes the attributes of the working mechanism of the blockchain consensus algorithms in an attempt not only to understand, but also to identify the patterns that group them together or separate them from the others. Therefore, based on the published data, a new categorization of consensus algorithms based on their working mechanism attributes is proposed. The results presented a well-defined separation of real groups (PoW, PoS, hybrid and voting-based forms); however, it does reveal some patterns that were not apparent before. One notable finding is that even hard hybrid forms use attributes from the PoW and PoS algorithms. The cluster analysis Fig. **3-5**, as well as the PCA analysis Fig. **3-3**, show a similarity between the hybrid algorithms and PoW based on the attributes *energy efficiency, modern hardware, forking, double spending attack* .

Another important finding is the proposal to reclassify other types of evidence-based algorithms, separating them into two different groups. One closer to PoS algorithms, including Proof-of-Space, challenges designed for humans PoW, and Proof-of-Burn algorithms; while the other includes Proof-of-Elapsed-Time, multichain, Proof-of-Luck, and GHOST strategy; excluding the latter of the pure PoW algorithms. It is necessary to mention that this categorization could be optimized by including more information about other attributes of the work mechanisms or other types of attributes.

4 The 51% attack on blockchains: A mining behavior study

Blockchain applications include cryptocurrencies, decentralized finance applications, video games, and many others. Most of these applications rely on the blockchain to prevent problems like fraud, thanks to the built-in cryptographic mechanisms provided by the data structure and consensus protocol. However, blockchains suffer from what is called a 51% attack or majority attack, which is considered a high risk to the integrity of these blockchains, where if a miner, or a group of them, has more than half of the computing power of the network, you can rewrite the blockchain. Although this attack is possible in theory, it is considered difficult to achieve in practice due to the assumption that, with enough active members, it is challenging to have so much computing power; however, this assumption has not been studied in sufficient detail.

This chapter presents a detailed characterization of the miners of the bitcoin and Crypto Ethereum blockchains, to test the assumption of computer distribution and create profiles that detect anomalous behavior and prevent 51% attacks. The analysis results show that, in recent years, there has been an increasing concentration of hash rate power in a minimal set of miners, creating a real risk for current blockchains. Also, there is a mining pattern among the leading miners, which allows identifying unusual behavior. This chapter presents a characterization of the principal miners in bitcoin and Ethereum. In particular, it focuses on miners with a hash rate of more than 1% per defined period to create profiles that may allow the detection of anomalous behaviors. In addition, a validation of the theoretical work of [Rosenfeld, 2014] is performed, based on the actual transactions dataset of these blockchains.

This chapter is organized as follows: Section 4.1 describes the methodology used in this study. Concretely, it describes the data selection, preprocessing and mining characterization phases. The Section 4.2 presents the results obtained organized into multiple segments, namely the number of miners, hash rate/share of the miner, percentage of mined consecutively, the profile of miners, and analysis of double-spending. Finally, the conclusions are presented in Section 4.3

4.1 Methodology

In this section, each of the phases of the methodology is detailed, first, explain how the data were obtained and selected, then explain the process of pre-processing the data sets, and finally, the process of characterization of the miners.

4.1.1 Data selection

In this study, the complete historical data generated by the nodes of the bitcoin and Ethereum networks were acquired and processed.

On the one hand, the bitcoin dataset is hosted at [Ventrone, 2021a]. This data have been progressively collected through Web Scraping from the website [Blockchain.com, 2021] using a script written in Python, which is available hosted at [Ventrone, 2021b]. The corresponding file has a size of 243 MB, and it contains the data of each bitcoin block ranging from the Genesis block to the block number 682 676 mined on May 9, 2021. The data stored in the file is organized into the following fields: `Hash`, `Confirmations`, `Timestamp`, `Height`, `Number of Transactions`, `Difficulty`, `Merkle root`, `Version`, `Bits`, `Weight`, `Size`, `Nonce`, `Transaction Volume`, `Block Reward`, `Fee Reward`, `Miner Name`, `Date`, `URL Miner`, `Year`, `Month` and `Day`.

On the other hand, the crypto ethereum dataset is hosted on the Google cloud platform. It was queried and downloaded by using the Google BigQuery tool from the `blocks` table of the `crypto_ethereum` dataset
(bigquery-public-data:crypto_ethereum.blocks) [Google Cloud Platform, 2021].
All data in this dataset was extracted, transformed and loaded through a set of Python scripts available on [Blockchain ETL, 2021]. The dataset hosted on Google Cloud contains data on each crypto ethereum block ranging from the Genesis block to current date. This because it gets updated daily, and its size is than 13 Gb. It is organized into the following fields (columns): `Timestamp`, `number`, `hash`, `parent_hash`, `nonce`, `sha3_uncles`, `size` , `transactions_root`, `state_root`, `receipts_root`, `miner`, `logs_bloom`, `total_difficulty`, `difficulty`, `extra_data`, `gas_limit`, `gas_used`, `transaction_count` and
`base_fee_per_gas`.

For the analysis, a subset of data was taken from the main dataset, which corresponds to the records from the Genesis block until April 27, 2021, when the block with identifier 12 322 990 was mined. This subset only contains the fields `timestamp`, `number`, `miner`, `difficulty` and `gas_used` fields. This subset of data was extracted using the following BigQuery command

```
SELECT
timestamp,number,miner,difficulty,gas_used
FROM
bigquery-public-data.crypto_ethereum.block
```

of which result was saved into a file with a size of 1.2 Gb. Both bitcoin and crypto Ethereum data sets are saved in a flat comma-separated CSV file for later loading and pre-processing with a Python script.

4.1.2 Preprocessing

The next stage after selecting data was preprocessing each dataset. This task was carried out with Google Colaboratory, a cloud service offered by Google that provides a virtual machine environment to run Jupyter-based Notebooks. The notebook with the code makes can be found at [Aponte-Novoa and Villanueva-Polanco, 2021]. On one hand, on the preprocessing of the bitcoin data, the columns other than Height and Miner Name were removed. Additionally, the columns `Date`, `Year`, and `Year_Month` were created from the `Timestamp` field. On the other hand, for the preprocessing of the crypto Ethereum data, the column `miner` was renamed as `Miner Name` and the column number by `Height`. As similar as in the bitcoin file, the columns `Date`, `Year` and `Year_Month` were created from the field `Timestamp`, and the remaining columns were removed.

4.1.3 Mining characterization

This section provides an analysis of mining behavior on two popular blockchains: bitcoin and crypto ethereum. This analysis is based on real collected data that has been examined methodically and in detail using data sciences techniques.

4.2 Results

In the information analysis process, different algorithms were designed and coded in the Python programming language in order to identify aspects such as the number of miners and number of blocks in different periods of time, the hash rate of miners, and the percentage of blocks mined consecutively. In addition, own algorithms and clustering algorithms were used to create profiles of the miners. On the other hand, the model presented by [Rosenfeld, 2014] was also coded in the Python programming language. All of the above was done for the bitcoin and crypto Ethereum dataset.

Figure 4-1: Number of bitcoin blocks mined by year

4.2.1 Number of miners

Bitcoin

A first analysis of the bitcoin data indicates that 682 676 blocks were mined by 73 different miners in the analyzed period. In particular, Fig. **4-1** shows the number of mined blocks per year, and Fig. **4-2** shows the number of miners per year in the analyzed period. In this first analysis, it can be seen how the Unknown miners, those with Miner Name 'Unknown', are treated as a single miner, which differs from reality. So the miners were individualized by creating a new field in the data set by concatenating the fields 'Miner Name' and 'URL Miner', this dataset is published in Mendeley Data, [Aponte et al., 2021b]. As a result of this individualization process on the miners, 198 892 different miners were identified in the entire analyzed period in contrast to the 73 miners found first. Fig.**4-3** shows the number of known miners and the number of unknown miners for each of the years in the analyzed period.

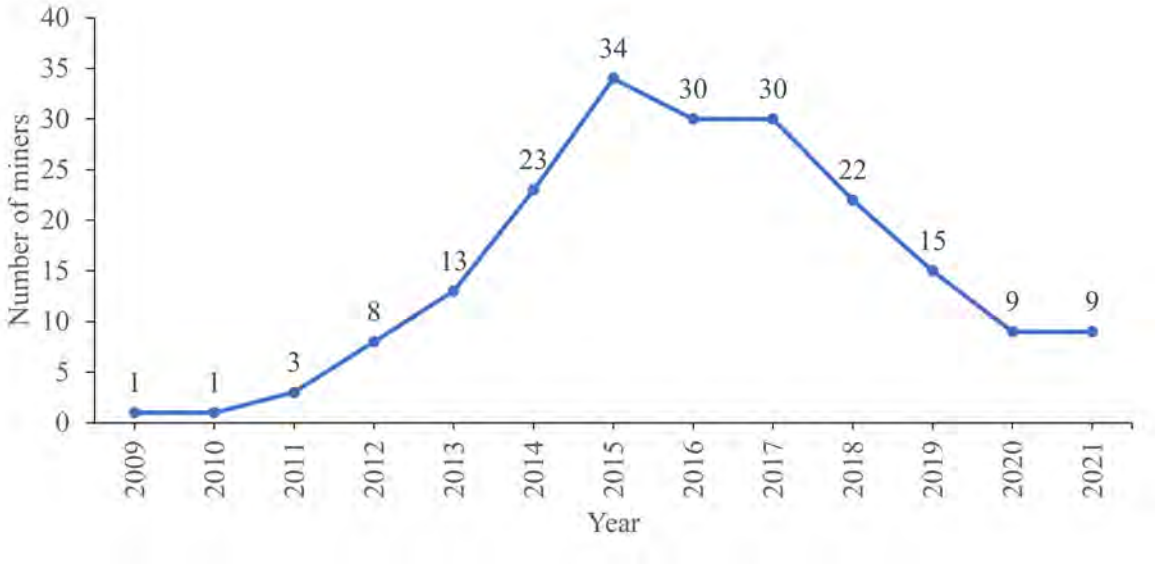


Figure 4-2: Number of unidentified unknown bitcoin miners by year

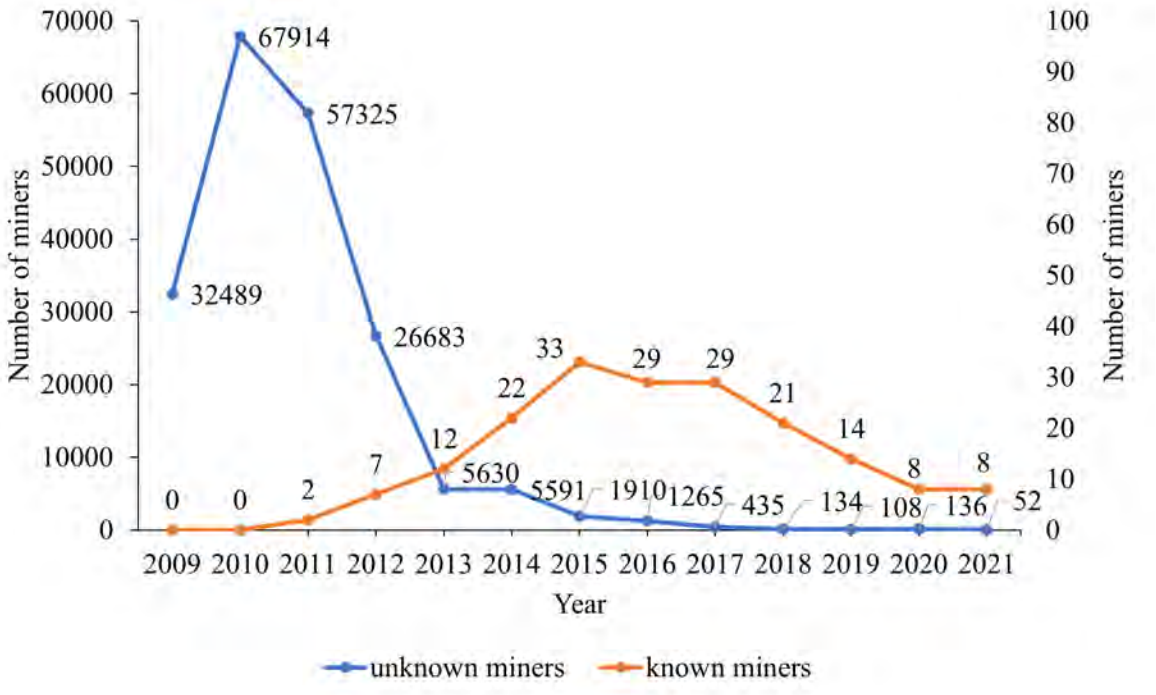


Figure 4-3: Number of bitcoin miners by year, after individualizing the unknown miners

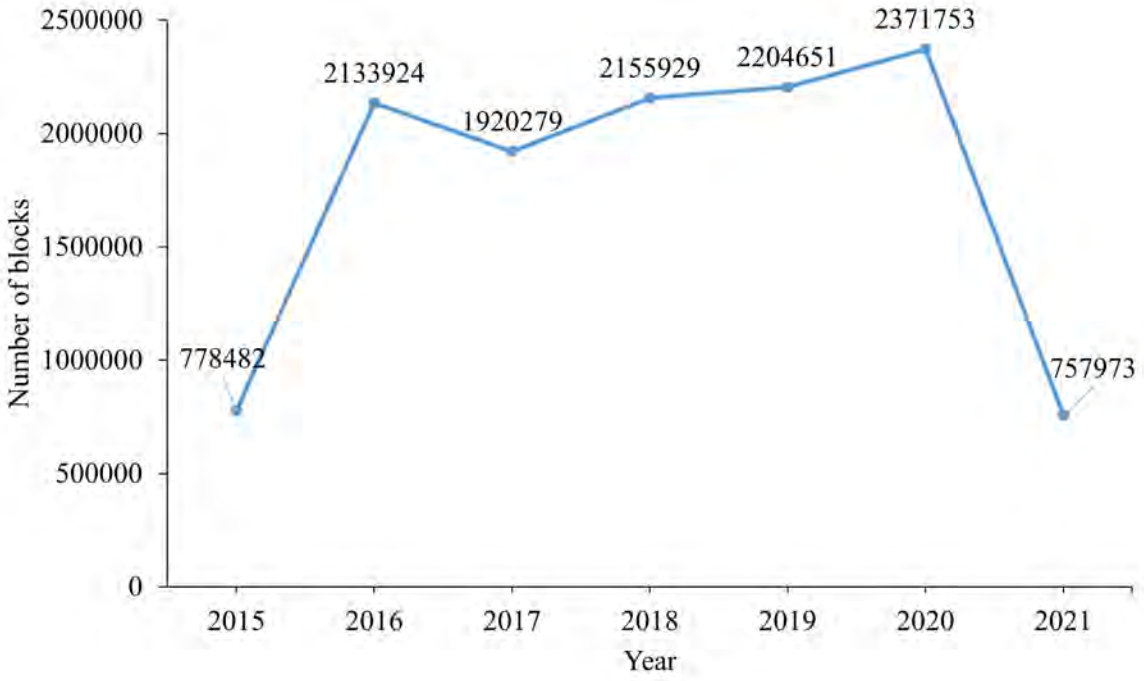


Figure 4-4: Number of crypto ethereum blocks mined by year

Crypto Ethereum

A similar analysis was carried out on the crypto ethereum dataset. In particular, this analysis shows that 12 322 991 blocks were mined by 5430 different miners in the analyzed period. Fig. **4-4** shows the number of mined blocks, and Fig. **4-5** shows the number of miners per year in the same time interval.

4.2.2 Hash Rate / share of miners

Bitcoin

For the 682 676 bitcoin blocks mined in the analyzed period, the hash rate distribution is made for that same period. Fig. **4-6** shows this distribution in which the unknown miners are taken as one. Because the number of miners presented is very large, only the miners that have a hash rate greater than or equal to 0.1% in the entire period are shown, resulting in 29 miners that represent 99.49% of the total hash rate in the observed period.

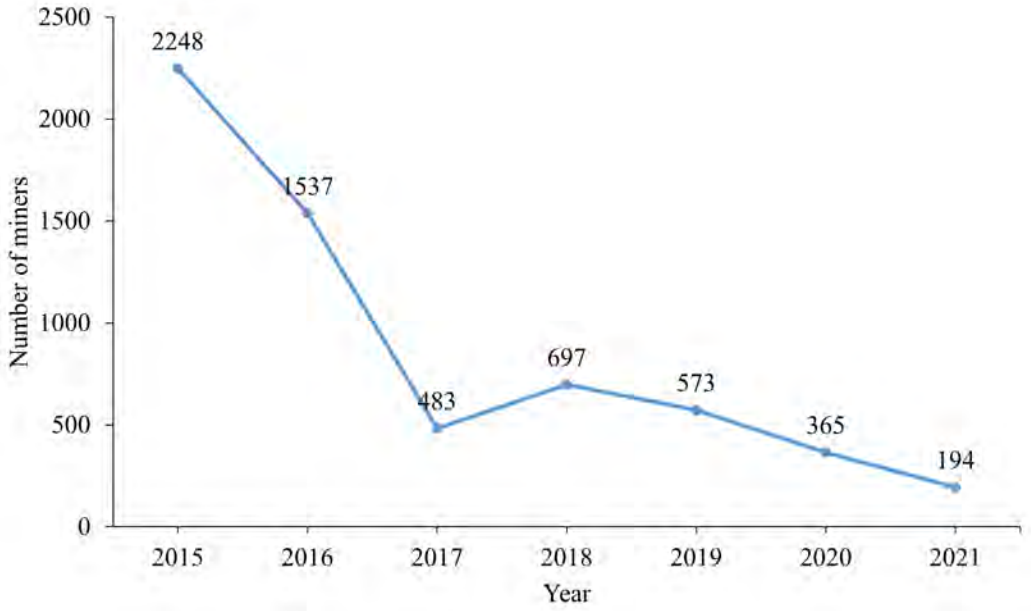


Figure 4-5: Number of crypto ethereum miners by year

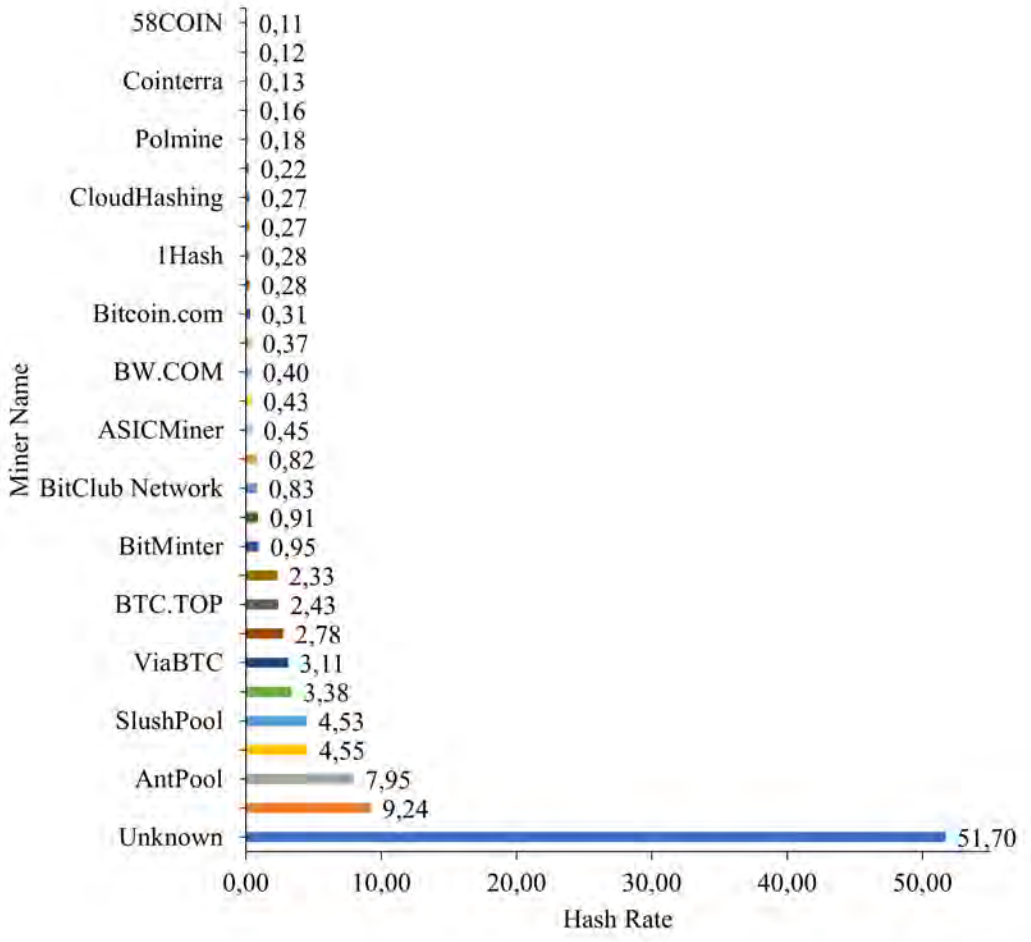


Figure 4-6: Bitcoin hash rate distribution throughout the period, miners with hash rate of at least 0.1%

Table 4-1: Hash rate distribution of bitcoin nodes throughout the observed period

Participation hash rate throughout the time period	**Number of miners in the period**	**Percentage of hash rate represented**
less than 0.1%	198 824	35.88%
at least 0.1%	68	64.13%
at least 0.3%	33	57.33%
at least 0.5%	22	53.37%
at least 0.7%	18	51.01%
at least 1%	11	45.22%

In the process of individualization of the unknown bitcoin miners, it goes from having 73 miners to 198 892 in the entire time. To analyze the hash rate distribution with all the miners already individualized in the period, this distribution is verified with the miners that present a minimum hash rate of 0.1%, 0.3%, 0.5%, 0.7% and 1% throughout the period. This hash rate represents the percentage of mined blocks by that particular miner in relation to all mined blocks in the entire time. Table **4-1** presents these results.

Based on the data in the table **4-1** and Fig. **4-7**, it can be seen that only 11 miners that represent 0.0053064% of their total number of nodes have a hash rate of at least 1% and together represent 45.22% of the hash rate in throughout the period. It is also identified that, of 198 824 miners that represent 99.965% of the total miners, present less than 0.1% of total hash rate in the entire period; in other words, only 68 miners, that represent 0.0341% of the total of miners have at least 0.1% hash rate in the entire period, and together represent 64.13% of the hash rate in the entire period analyzed. Fig. **4-7** presents the hash rate distribution for the entire period with the miners that have at least 1% total hash rate in the analyzed period.

Fig. **4-7** shows that of the 11 miners that have at least 1% of the total hash rate, two of those are unknown and the remaining nine individually represent mining pools. Taking as a starting point these 11 miners who individually possess at least 1% of the total hash rate, the hash rate distribution of the entire period is plotted year by year in Fig. **4-8**.

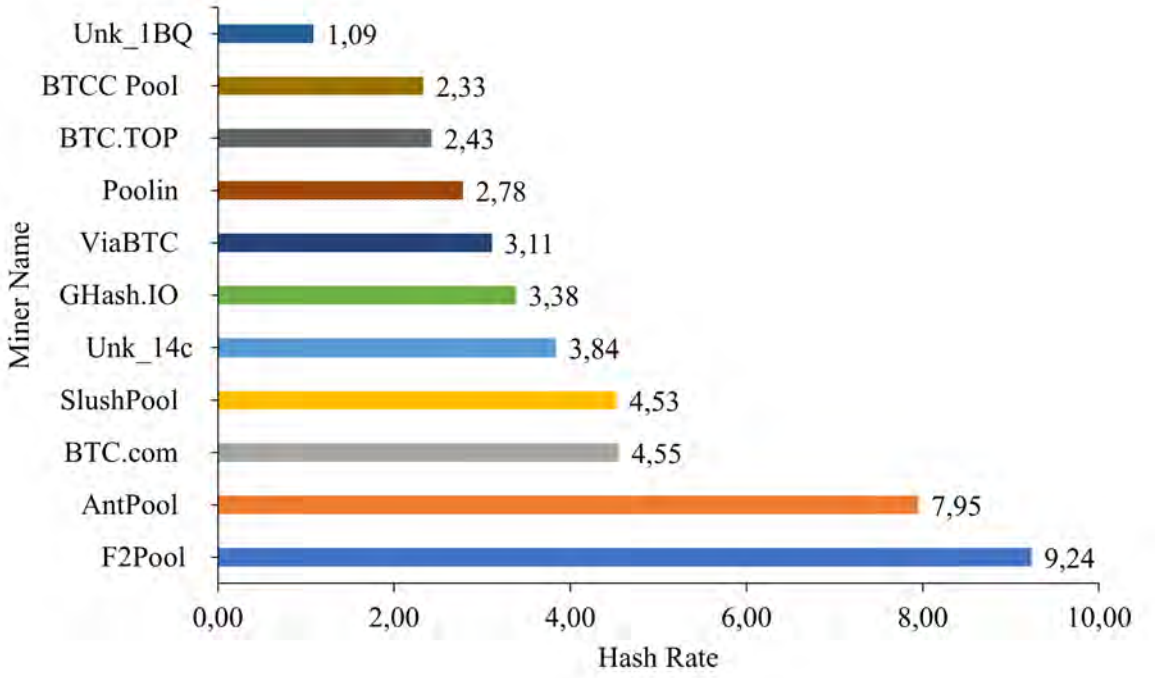


Figure 4-7: Bitcoin hash rate distribution throughout the period with hash rate of at least 1% after individualizing the unknown miners

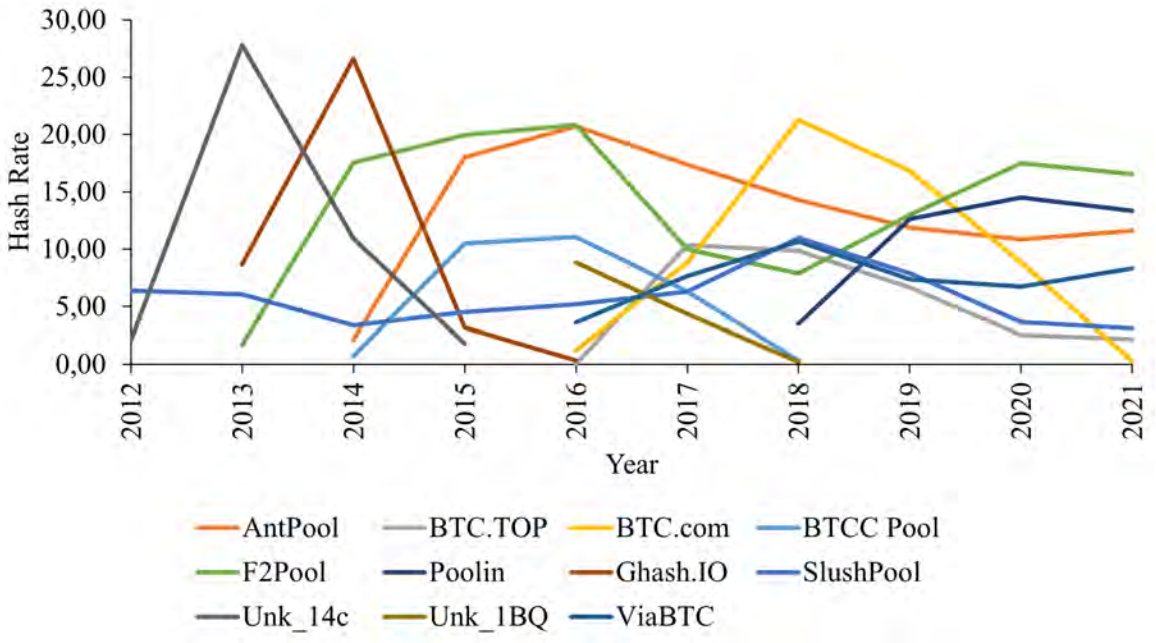


Figure 4-8: Bitcoin hash rate distribution over time with hash rate of at least 1%, after individualizing the unknown miners

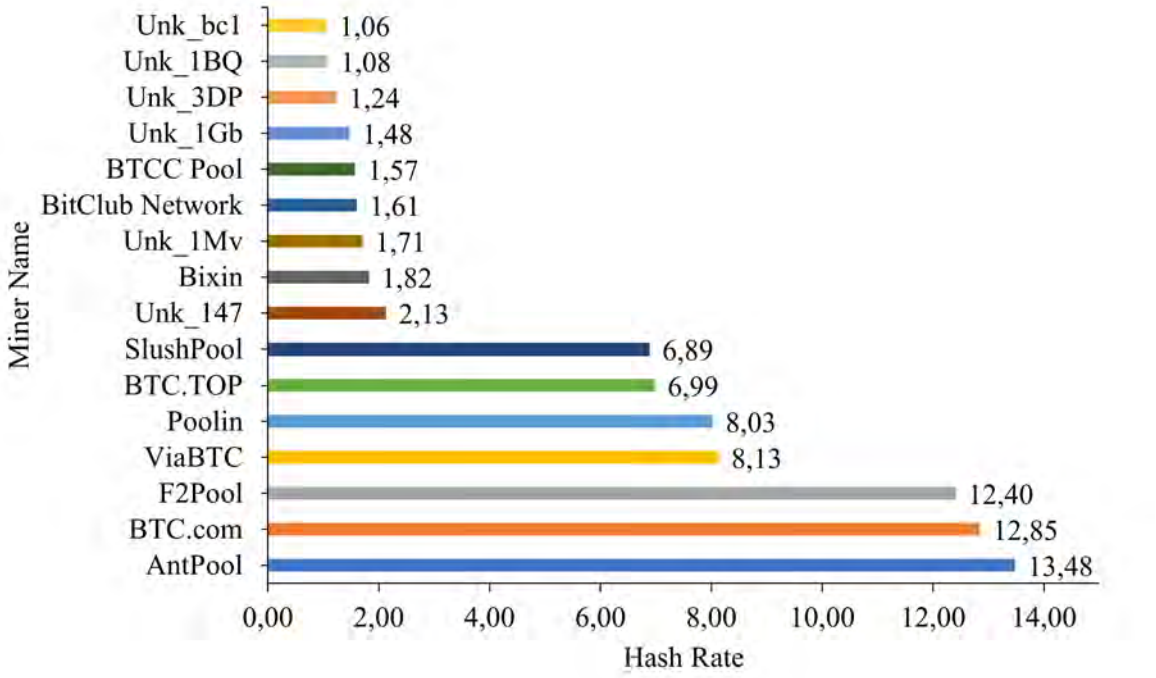


Figure 4-9: Bitcoin hash rate distribution with hash rate of at least 1% after individualizing the unknown miners from January 1, 2017 until May 8, 2021

In Fig. **4-8** it can be seen that all the miners that have a hash rate of at least 1% in the entire period are present in the bitcoin network since the year 2012, only one of them has remained active since 2012 to the present (May 2021), and 7 of the 11 miners in question are active in May 2021.

To verify the current status of the bitcoin network in terms of the hash rate distribution, and taking into account the number of active miners since 2017, the hash rate distribution of the entire bitcoin network is generated from January 1, 2017, until May 8, 2021 (Fig. **4-9**).

Fig. **4-9** indicates that for the last four years (2017-2020) in the bitcoin network only 16 miners that have a total hash rate of at least 1% for that period, represent 82.47% of the hash rate of the entire network. From these 16 miners, 10 have been identified and the remaining 6 correspond to unidentified miners. The Fig. **4-9** indicates that only 7 of the 11 miners exceed a hash rate of 5%, accumulating together 68.77% of the total hash rate.

Fig. **4-10** shows the hash rate distribution for these 7 miners in the period from January 1, 2017, to May 8, 2021, seen by years. In this figure, it can be seen that, of the 7 miners in question, 6 has been present on the network since 2017, and one of them since 2018.

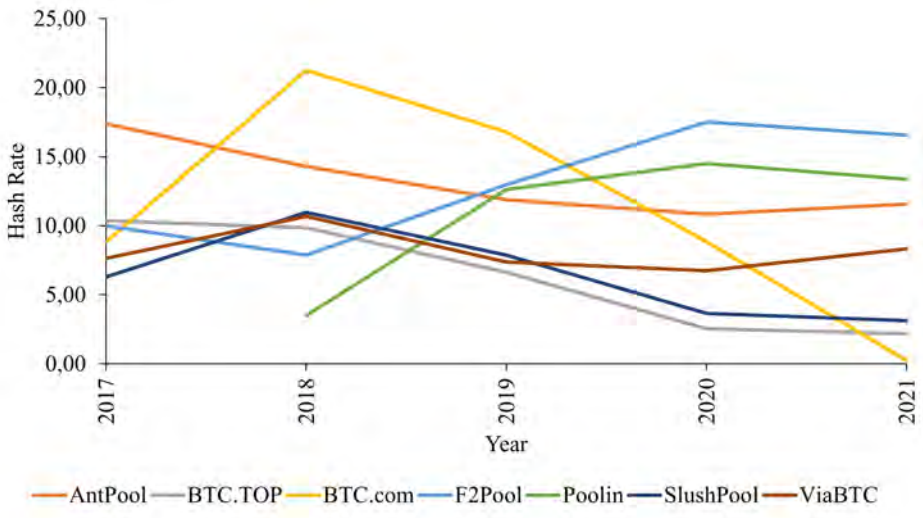


Figure 4-10: Bitcoin hash rate distribution over time with hash rate of at least 5% after individualizing the unknown miners from January 1, 2017 until May 8, 2021

The hash rate distribution is generated throughout the period for the 7 miners that have presented the most hash rate in the last four years, and it is shown in Fig.**4-11**. The results show that, of the 7 miners, "SlushPool" is the only one that has been present on the network since 2012, 'F2Pool' since 2013, 'AntPool since 2014' and miners the 'BTC.com', 'BTC.TOP', 'ViaBTC' since 2016 and for their part 'Poolin' since 2018.

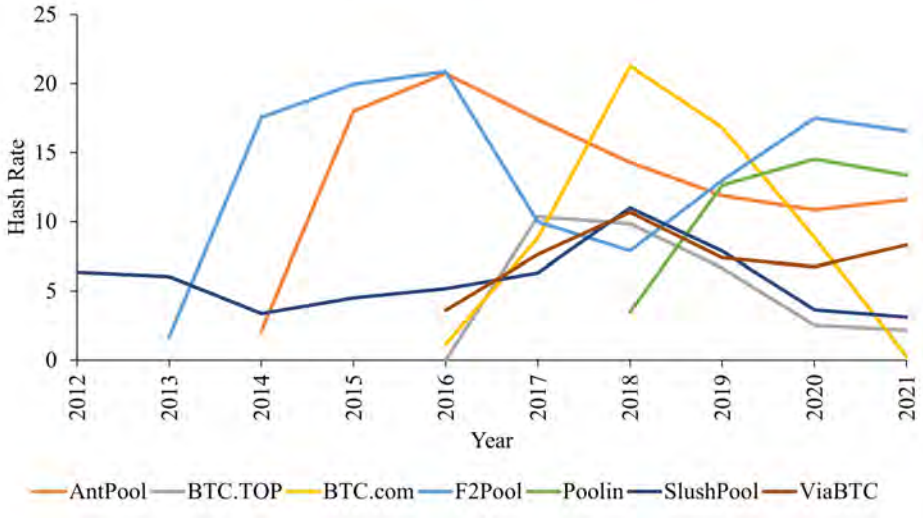


Figure 4-11: Bitcoin hash rate distribution of the top 7 miners over time after individualizing the unknown miners from January 1, 2009 until May 8, 2021

Crypto Ethereum

For the 12 322 990 crypto ethereum blocks mined in the analyzed period, the share distribution is made for that same period. This share distribution represents the percentage of mined blocks by that particular miner in relation to all mined blocks in the analyzed period. Fig. **4-12** shows this distribution. Because the number of miners presented is very large, only the miners that have a share greater than or equal to 1% in the entire period, resulting in 13 miners that represent 78.18% of the total share in the observed period, are analyzed.

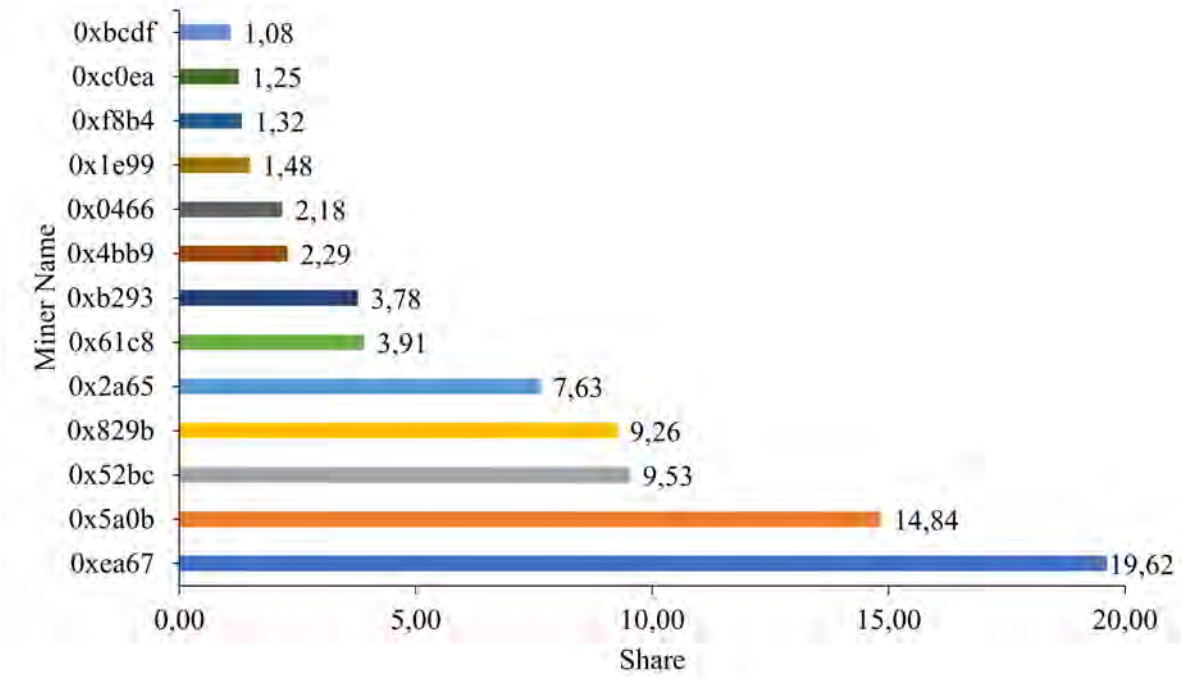


Figure 4-12: Crypto Ethereum share distribution throughout the period with share of at least 1%

To further analyze the share distribution of all the miners of the crypto ethereum network in the study period, this distribution is verified with the miners that present a minimum share of 0.1%, 0.3%, 0.5%, 0.7%, and 1% over the entire period, Table. **4-2** shows these results.

Based on the results in Table **4-2**, it can be said that only 13 miners that represent 0.23% of their total number of nodes and have a total share of at least 1% (Fig. **4-12**) represent 78.18% of the share in the entire period. It is also identified that 5366 miners, which represent 98.82% of the total miners, present less than 0.1% of the total share in

Table 4-2: Share distributions of some crypto ethereum nodes throughout the period

Participation share throughout the time period	**Number of miners in the period**	**Percentage of share represented**
less than 0.1%	5366	6.1%
at least 0.1%	64	93.90%
at least 0.3%	37	89.25%
at least 0.5%	19	82.21%
at least 0.7%	15	79.99%
at least 1%	13	78.18%

the entire period; in other words, only 64 miners, that represent the 1.17% of all miners that have at least 0.1% share in the entire period, represent 93.90% of the share in the entire period analyzed.

To identify which miners are currently more important, the share of the miners that represent at least 1% share is generated in the period between January 1, 2019, and April 27, 2001, as shown in Fig. **4-13**. The figure indicates that only 11 miners comply with having at least 1% share in the analyzed period and together present an 82.25% total share. Of these 11 miners, it is identified that 2 of these are not present in 2020 or 2021. Of the nine miners that have participated in 2021, 8 have been present since 2019, and one of them since 2020. For the 8 miners that are present in the years 2019, 2020, and 2021, their share is analyzed in the period between January 1, 2019, and April 27, 2001, Fig. **4-14**. Each of these miners presents individually by at least 1% share and together they add up to 77.59% share.

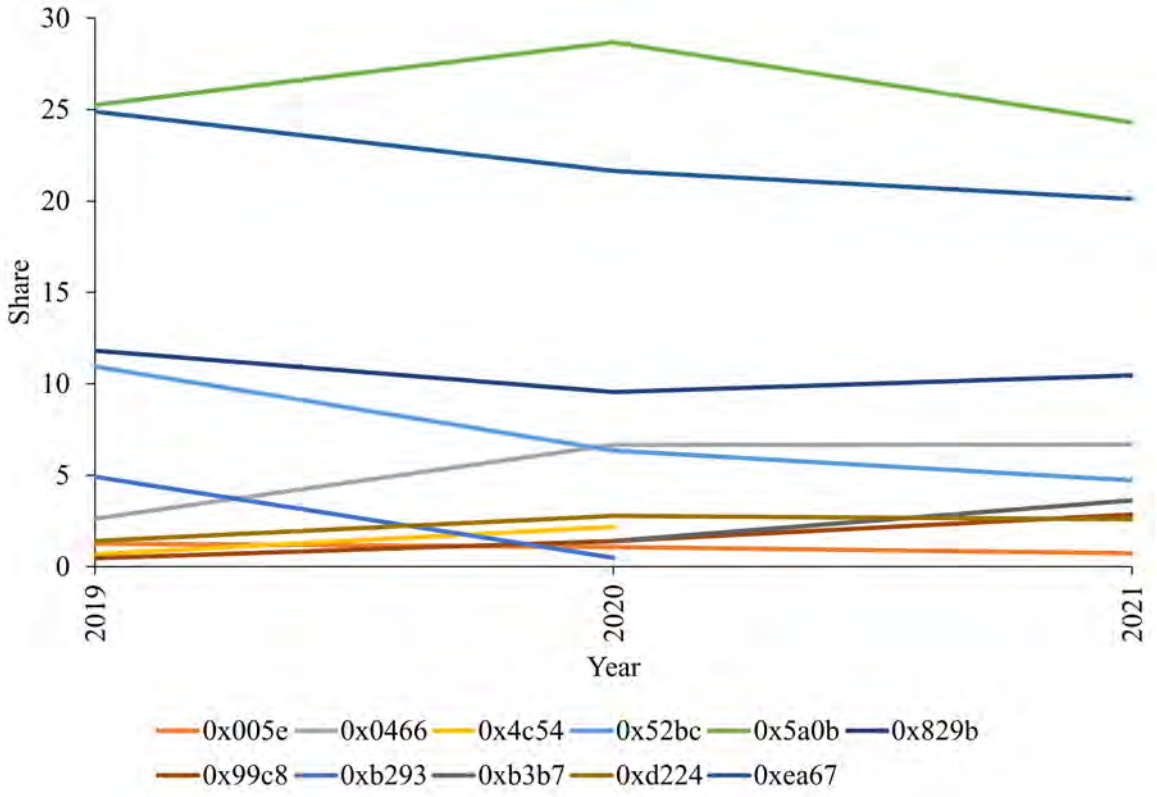


Figure 4-13: Crypto Ethereum share distribution with share of at least 1% from January 1, 2019 until April 27, 2021

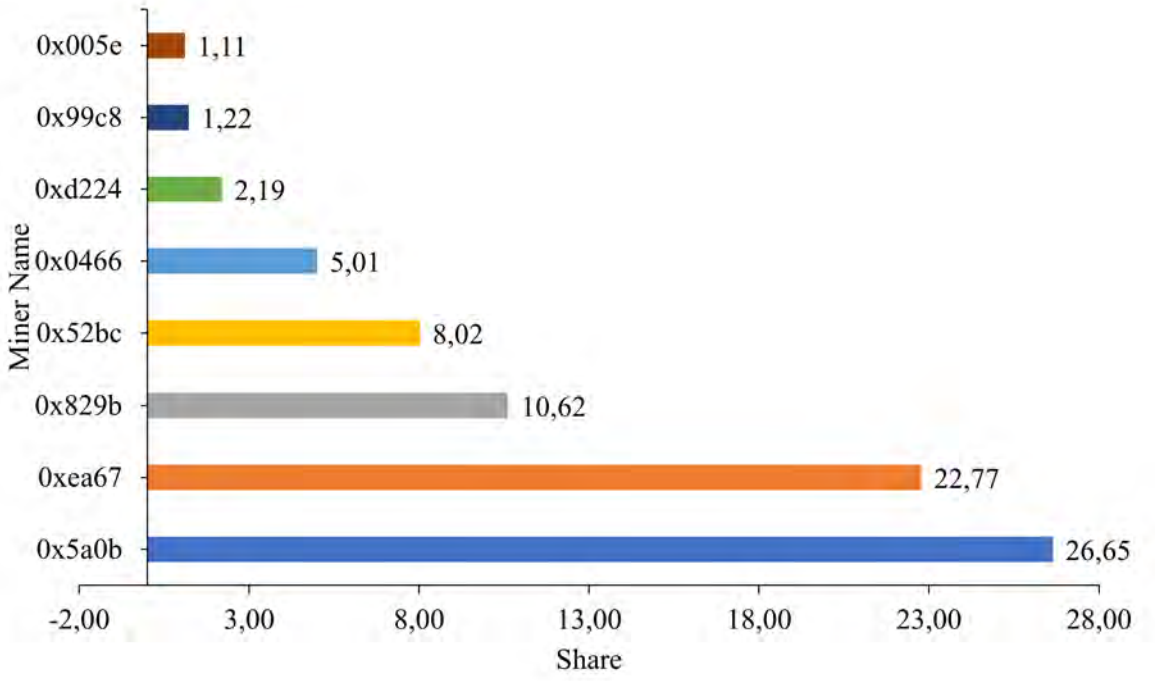


Figure 4-14: Crypto Ethereum share distribution from January 1, 2019, to April 27, 2021

4.2.3 Percentage of blocks mined consecutively

For each of the data sets, an algorithm was applied to identify the percentage of blocks mined consecutively in the entire period of observed time, and for each of the years in the period analyzed.

Bitcoin

In the bitcoin data set, if the probability of mining at least one block in a consecutive manner is 100%, the longest mining chain was 11 blocks, and it had a calculated probability of 0.0001%, in the observed time. Fig. **4-15** shows the percentage of blocks mined consecutively in the whole period with chains of 2 blocks up to 11 consecutive blocks. Fig. **4-15** shows how the probability of mining consecutive blocks decreases as the number of blocks is greater; this trend is presented for each of the years. In this analysis, it can be seen that, for the first four years analyzed (2009-2012), the longest chain occurs in 2012 with 5 blocks and a mining probability of 0.0073%. From the year 2013 to the year 2021, mining chains greater than 5 blocks have been presented, with the year 2013 presenting chains of 11 blocks and the year 2014 of 10 blocks with probabilities of 0.0016% and 0.0017%, respectively. The year 2016 presents mining chains of up to 8 consecutive blocks with a probability of 0.0018%, In the years 2015, 2018, and 2019, there were mining chains of up to 7 blocks, and up to 6 blocks in the years 2017, 2020 and 2021, with average probabilities of 0.0030% and 0.0066% respectively.

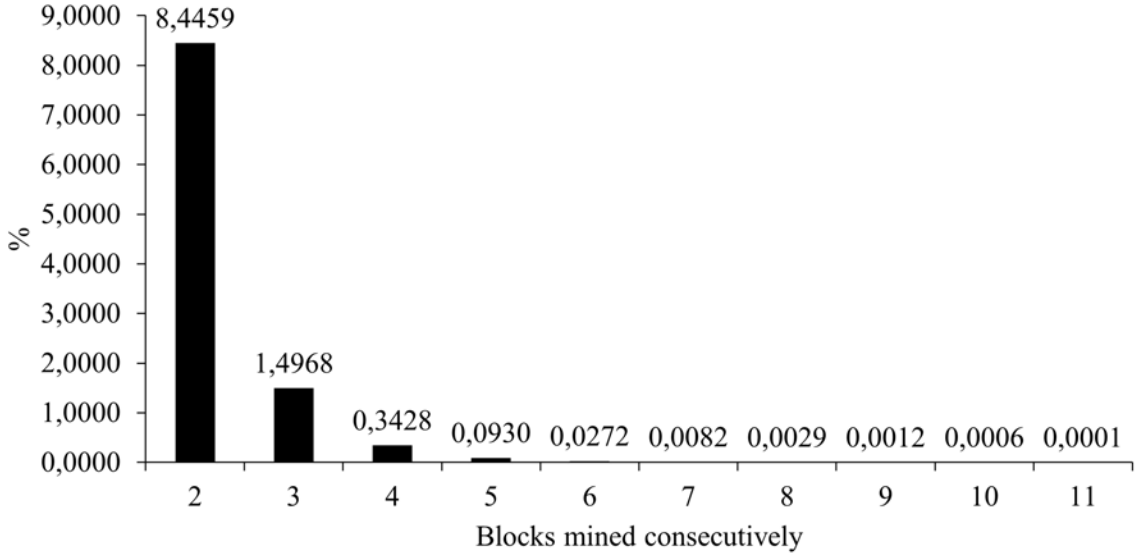


Figure 4-15: Percentage of Bitcoin blocks mined consecutively

Cryto Ethereum

For the crypto ethereum data, it is identified that the longest chain of mined blocks corresponds to 17 blocks, which occurs with a probability of 0.000008% throughout the period, and 0.000128% in 2015 that was when this event happened. Fig.**4-16** shows the percentage of blocks mined consecutively in the whole period, with chains from 2 blocks to 17 consecutive blocks. It can be seen that the probability of mining consecutive blocks decreases as the number of blocks is greater, behavior that is presented for the entire period analyzed as well as in each of the years of that period. It is identified in this analysis that, every years during the observed period, there are mining chains of 10 blocks with an average probability of 0.00126%. Likewise, for the years 2019 and 2020, there are mining chains of 11 blocks with a probability of 0.000044%, and for the year 2016 and 2018 chains of 12 and 14 blocks respectively which present 0.00018745% and 0.00004638% probability of mining.

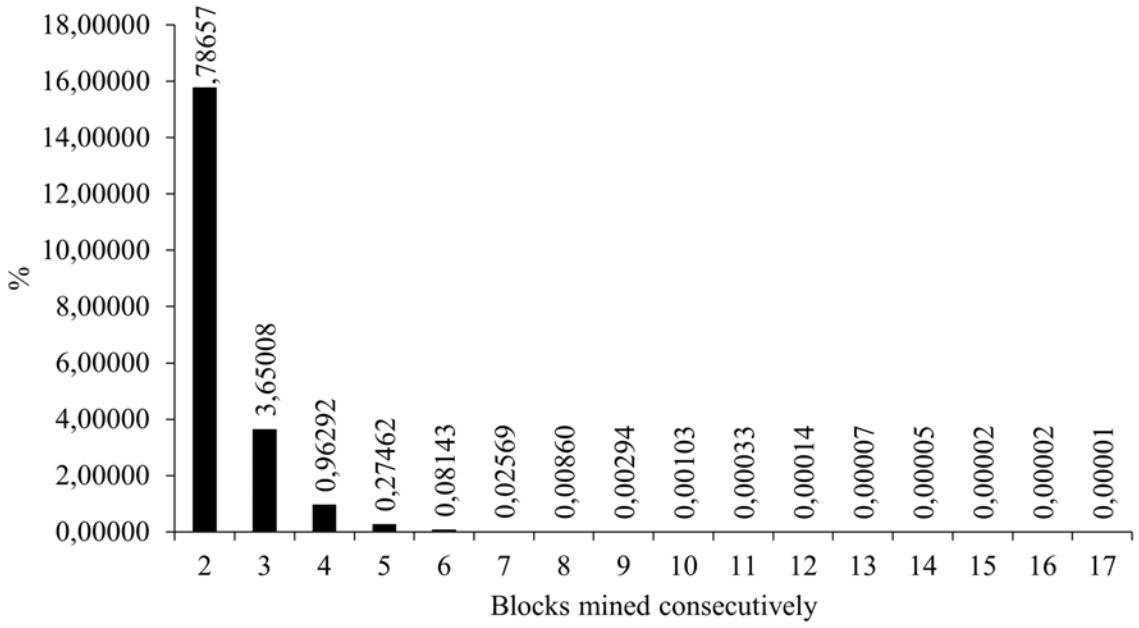


Figure 4-16: Percentage of crypto ethereum blocks mined consecutively

4.2.4 Profile of miners

Bitcoin

To create profiles of the current most representative miners of the bitcoin network, the 10 miners that make a presence on the bitcoin network in 2021 and have at least a 1% hash rate in the period from 1%, are considered from January 1, 2019, to May 8, 2021.

They have a combined hash rate of 76.27% for that period and 73.2% for the period of January 1, 2017, to May 8, 2021, and of 36.13% in 2009-2021. In Fig. **4-17** the percentage of active days for each of the 10 miners in the 2019-2021 period is shown together with the hash rate of each of the miners throughout the period, years 2017 to 2021 and years 2019 to 2021. It is identified that the hash rate of each of the miners is higher in recent periods compared to the entire period of the presence of the bitcoin network (2009-2021). Likewise, it can be seen that 7 of the 10 miners have an active presence ratio of more than 87% in the analyzed period. Only one of those miners has less than 50% presence in the analyzed period with 35.3%

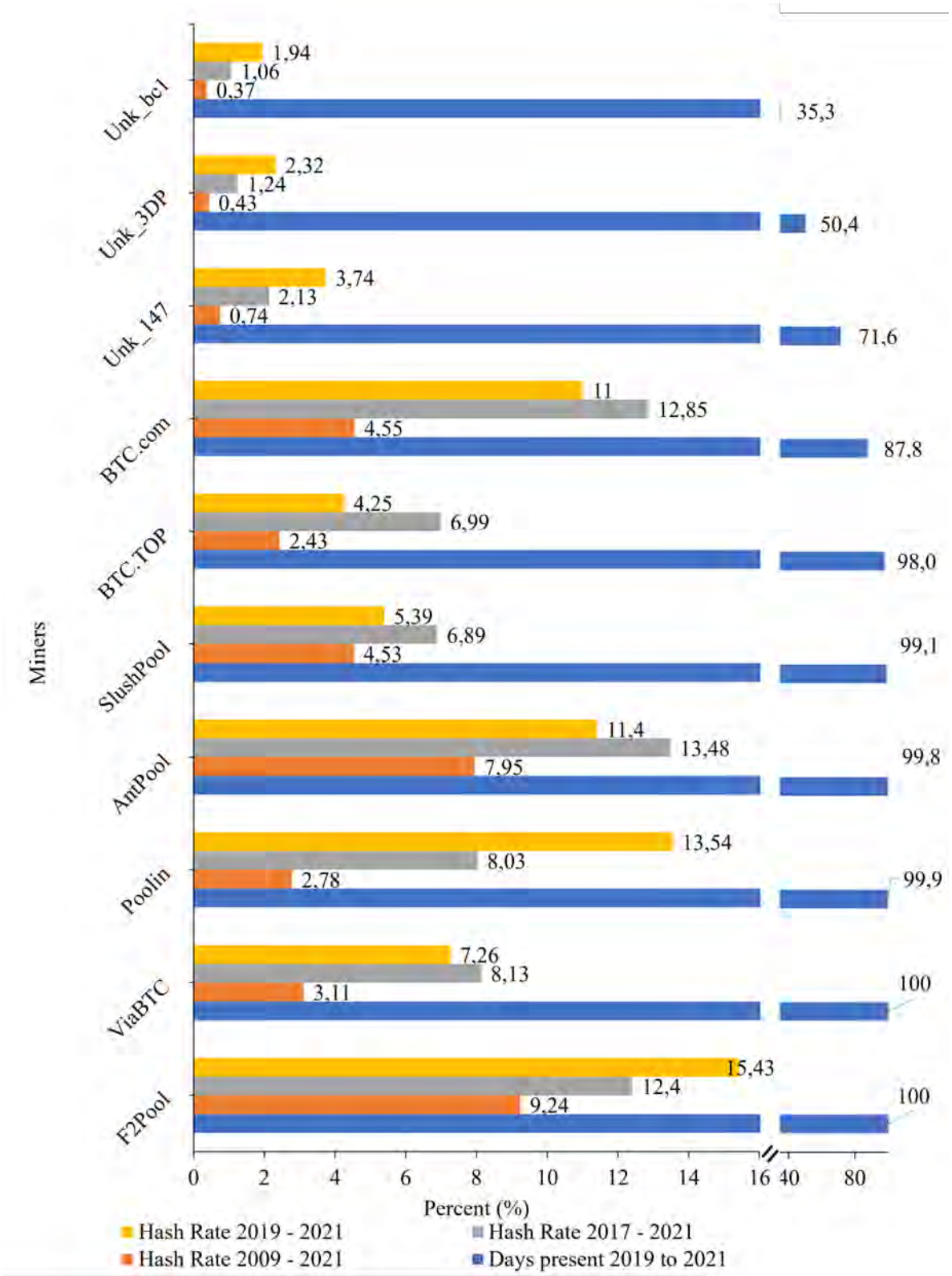


Figure 4-17: Best bitcoin miners in the period January 1, 2019 to May 8, 2021

Table 4-3: Groups formed for the best bitcoin miners with the K-Means, DBScan, and Birch algorithms to generate 3 groups

Group 0		Group 1		Group 2	
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
AntPool	60.65	BTC.TOP SlushPool	83.35 75.09		
F2Pool	66.94	ViaBTC Unk_147	58.21 86.26	BTP.com	NA
Poolin	78.11	Unk_3DP Unk_bc1	76.25 48.20		

Table 4-4: Groups formed for the best bitcoin miners with the K-Means, DBScan, and Birch algorithms to generate 4 groups

Kmeans		DbScan		Birch	
Group 0					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
AntPool F2Pool Poolin	60.65 66.94 78.11	AntPool	NA	AntPool F2Pool Poolin	60.65 66.94 78.11
Group 1					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
Unk_147 Unk_3DP Unk_bc1	76.37 88.71 58.09	BTC.TOP SlushPool ViaBTC Unk_147 Unk_3DP Unk_bc1	83.35 75.09 58.21 86.26 76.25 48.20	BTC.TOP SlushPool ViaBTC Unk_147 Unk_3DP	74.97 76.72 57.74 65.42 50.99
Group 2					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
BTP.com	NA	BTP.com	NA	BTP.com	NA
Group 3					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
BTC.TOP SlushPool	75.90 80.21	F2Pool	NA	Unk_bc1	NA

The day-to-day hash rate of each of the miners is taken from January 1, 2019, to March 8, 2021 [Aponte et al., 2021c], and it is represented in a vector of 859 positions corresponding to the each days in that period. The value of each position of the vector corresponds to the hash rate of the particular miner on each day. The clustering algorithms K-Means, DBScan, and Birch are applied to the set of hash rate vectors of the miners in order to identify a patterns among them. After applying these algorithms, 3 and 4 groups are generated. For the case of 3 groups, the parameters for the algorithms are K-Means (k = 3), DBScan (eps = 1.65 ; min_sample = 1.0; metric = "euclidean") and Birch (branching_factor = 23, threshold = 1.05). In this case, the results of the application of the different algorithms generated the same 3 groups, as shown in Table **4-3**.

To create 4 groups with the clustering algorithms, the parameters for the algorithms are

K-Means (k = 4), DBScan (eps = 1.6; min_sample = 1.0; metric = "euclidean"), and Birch (branching_factor = 23, threshold = 1.25). In this case, the algorithms created different groups, as it can be seen in Table **4-4**. The names of the unknown miners in tables **4-3** and **4-4** were truncated for better visualization. For each of the different groups formed by the clustering algorithms, it was verified that their elements (miners) are in the range of a mean $+/-1$ standard deviation, this value is displayed in the % presence column in the **4-3** and **4-4**.

Cryto Ethereum

To create profiles of the most representative miners currently in the crypto ethereum network, the 8 miners that have an active presence in the years 2019 - 2021 and that individually have at least 1% share in the period, are taken from January 1, 2019, to April 27, 2021. They have a combined share of 77.59% for that period and 69.63% for the period of January 1, 2017, to April 27, 2021, and 57.45% between 2015-2021. Fig. **4-18** shows the percentage of days present for each of the 8 miners, in the 2019-2021 period, together with the share of each of the miners throughout the period, years 2017 to 2021 and years 2019 to 2021. It is identified that the share of each one of the miners is higher in recent periods compared to the entire period of the presence of the crypto ethereum network (2015-2021). Likewise, it can be seen that 7 of the 8 miners have an active presence of the 100% during the analyzed period and the remaining 99.4% presence in the analyzed period.

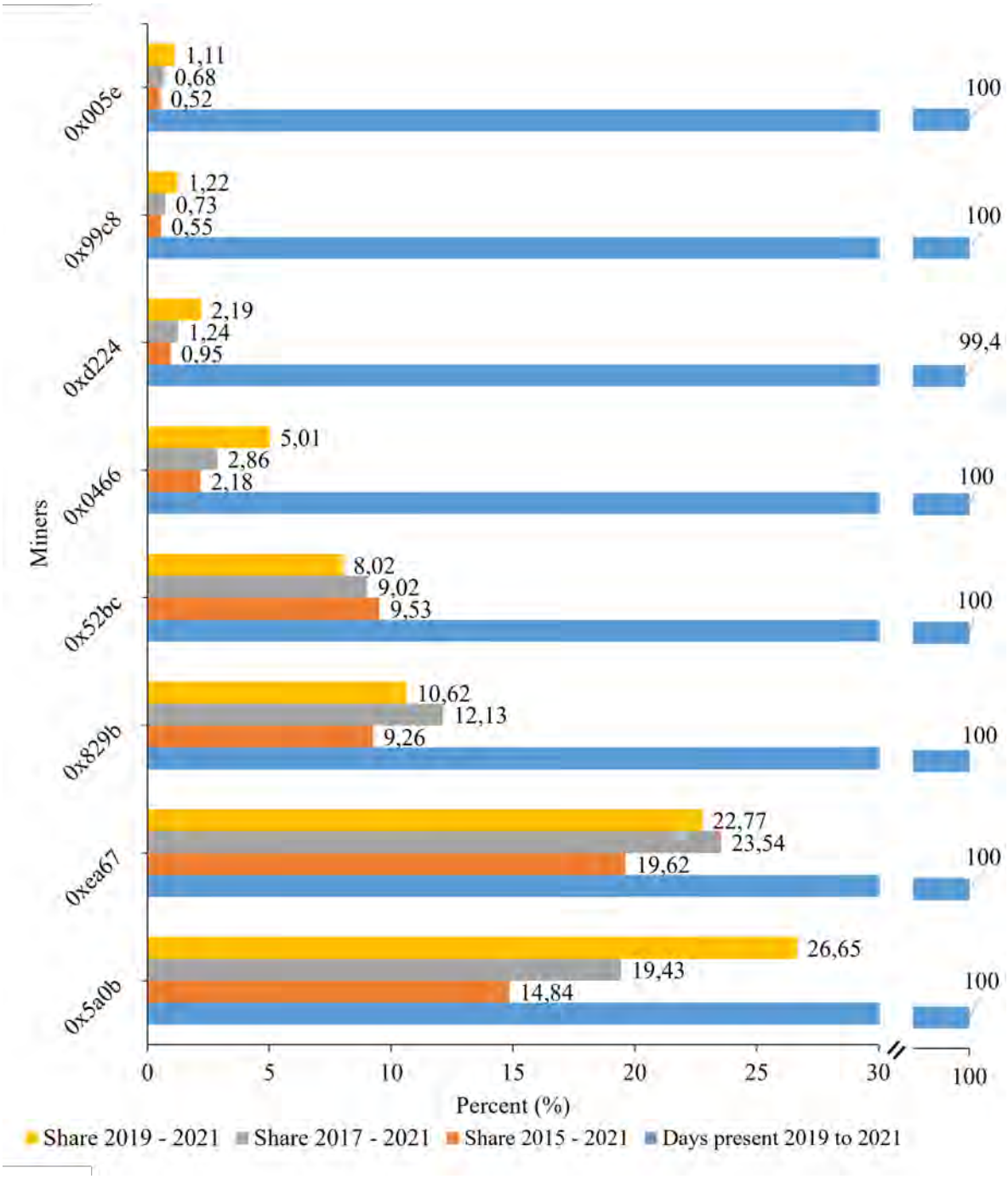


Figure 4-18: Best cryto ethereum miners in the period January 1, 2019 to April 27, 2021

The day-to-day share of each of the miners is taken in the period from January 1, 2019, to April 27, 2021 [Aponte et al., 2021c], and it is represented by a vector of 859 positions

corresponding to the days in that period. The clustering algorithms K-Means, DBScan, and Birch are applied to the set of hash rate vectors of the miners. After applying these algorithms, 3 and 4 groups were generated. The parameters used for the algorithms are K-Means (k = 3), DBScan (eps = 1.9; min_sample = 1; metric = "euclidean") and Birch (branching_factor = 2, threshold = 1.05) and * (branching_factor = 2, threshold = 1.15). The final grouping is shown in Table **4-5**. For the case of 3 groups, Birch generated two different results options.

Table 4-5: Groups formed for the best crypto ethereum miners with the k means, dbscan, and birch algorithms together with the percent of the presence of their elements in the range "mean + / - one standard deviation" for 3 groups

Kmeans / Birch		**DbScan**		**Birch**	
Group 0					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
0x0466 0xd224 0x99c8 0x005e	18.04 94.92 77.24 92.09	0x5a0b	NA	0xd224 0x99c8 0x005e	58.72 55.66 85.61
Group 1					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
0x5a0b 0xea67	100 100	0xea67	NA	0x5a0b 0xea67	100 100
Group 2					
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
0x829b 0x52bc	100 100	0x829b 0x52bc 0x0466 0xd224 0x99c8 0x005e	2.83 65.56 88.67 99.64 85.84 61.32	0x829b 0x52bc 0x0466	73.34 77 50

For the generation of 4 groups with the clustering algorithms, the parameters for the algorithms are K-Means (k = 4), DBScan (eps = 1; min_sample = 1), and Birch (branching_factor = 2, threshold = 1). In this case, the results of the 3 different algorithms presented the same results, which are presented in Table **4-6**. The names of the miners in tables **4-5** and **4-6** were truncated for better visualization. For each of the different groups formed with the clustering algorithms, it was verified that their elements (miners) are in the range of mean $+/-1$ standard deviation, this value is displayed in the % presence column in the **4-5** and **4-6**.

4.2.5 Analysis of double-spending

The research by [Rosenfeld, 2014] presents an analytical solution to model the probability of a double-spending attack via a stochastic process. A double-spending attack

Table 4-6: Groups formed for the best crypto ethereum miners with the kmeans, db-scan, and birch algorithms together with the percent of the presence of their elements in the range "mean +/- one standard deviation" for four groups

kmeans / birch							
Group 0		**Group 1**		**Group 2**		**Group 3**	
Miner	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**	**Miner**	**% presence**
0x5a0b	NA	0xea67	NA	0x829b	100	0x0466	18.04
						0xd224	94.92
						0x99c8	77.24
				0x52bc	100	0x005e	92.09

happens when an attacker persuades a seller that a transaction has been confirmed and subsequently convinces the entire network to accept other transactions that make the first transaction invalid. If such an attack occurs, then the merchant is left without the product and the payment, and thus the attacker keeps the product and the value of the payment.

Recall that a transaction included in a block within the valid chain has n confirmations if there are n blocks that follow the block containing the transaction. The model proposed in [Rosenfeld, 2014] assumes that there is a block B_k within a branch known to the honest miners (normally the longest branch) and that such block B_k contains the transaction T_k that credits the payment to the seller and has n confirmations. To perform the attack, the attacker has to construct a branch with additional m blocks starting from the block to which the block B_k points. Both the honest miners and the attacker are in the task of extending their respective branches. This model for the double-spending attack is inspired by a catching up game, in which the attacker's goal is to make its branch longer than the valid chain.

This model also supposes that the hash rate of the honest network and the attacker is constant. Specifically, if the complete hash rate is H, then $p \cdot H$ is the portion that corresponds to the honest miners, and $q \cdot H$ is the remaining portion that corresponds to the attacker, where $p+q = 1$. Also, it supposes that the mining difficulty is unchanging for the hash rate H, and that the average time to mine a block is T_0.

Let $z = n - m$. Whenever a block is found, the value of z changes; if that block was found by the honest network z increases by 1, and if that block was found by the attacker z decreases by 1. If z ever reaches -1, the attacker's chain becomes longer and his attack succeeds.

Let a_z be the probability that the attacker will be able to catch up when he is currently z blocks behind. Following the analysis in [Rosenfeld, 2014], $a_z = min(q/p, 1)^{\max(z+1,0)}$. Additionally, m is regarded as a negative binomial variable; it represents the number of

successes (blocks mined by the attacker) before n failures (blocks mined by the honest network), where q is the success probability. Therefore, the probability for a particular value of m is given by:

$$P(m) = \binom{m+n-1}{m} p^n q^m \tag{4-1}$$

Note that once n blocks are found by the honest network, in a period of time during which $m+1$ blocks are found by the attacker (one block is assumed to be pre-mined by the attacker before commencing the attack), the race starts with $z = n-m-1$. It follows that the probability for the double-spend to succeed is $r := \sum_{m=0}^{\infty} P(m) a_{n-m-1}$.Following the analysis in [Rosenfeld, 2014], it follows that

$$r = \begin{cases} 1 - \sum_{m=0}^{n} \binom{m+n-1}{m} (p^n q^m - p^m q^n) & if q < p \\ 1 & if q \geqslant p \end{cases} \tag{4-2}$$

Following these results, an analysis of the success probability of a double-spending attack for the historical data of bitcoin and Ethereum is presented in this chapter. For both bitcoin and Ethereum, the mining data for the year 2020 is taken from the best miners presented in Fig. **4-17** and Fig. **4-18**. For each of the analyzed blockchain networks, 3 miners with different hash rate/share values are selected. The results are shown in Table **4-7**.

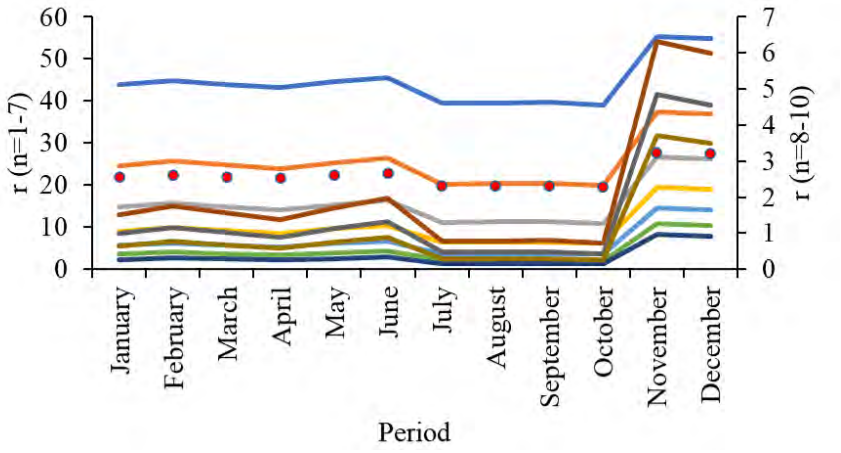
Table 4-7: Three selected miners of both Bitcoin and Ethereum for the year 2020 with different hash rate/share values

Bitcoin			
Miner	**Max hash rate**	**Min hash rate**	**Avg hash rate**
F2Pool	27.67	19.53	22.24
ViaBTC	11.71	6.81	8.63
BTC.TOP	4.43	2.18	3.22
Ethereum			
Miner	**Max share**	**Min share**	**Avg share**
0x5a0b	41.57	30.20	36.68
0x52bc	10.09	6.6	8.12
0x005e	1.85	0.84	1.38

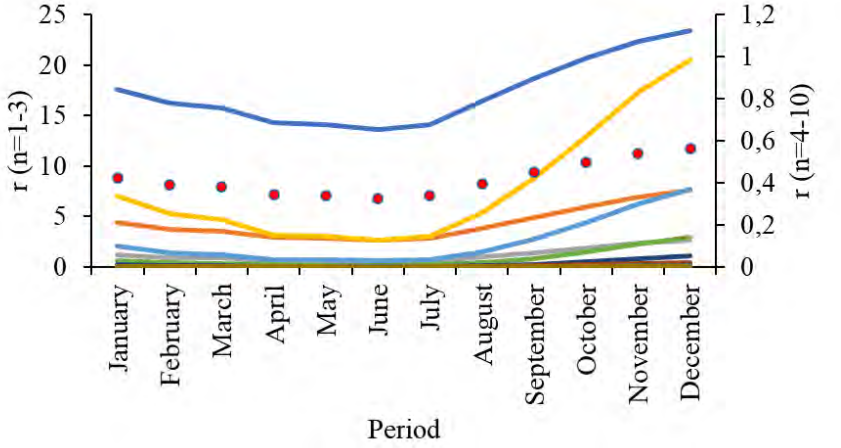
For these miners, the success probability of a double-spending attack, r, is calculated for each month of the year 2020. For a miner, r is computed for each $q \in \{q_1, q_2, \ldots, q_{12}\}$ and $n \in \{1, 2, \ldots, 10\}$, where q_i denotes the hash/share ratio per month for the miner and n the number of confirmations. Particularly, q_i is calculated by computing the

number of blocks mined by the miner in the month i, divided by the total number of blocks mined in the month i. Fig. **4-19** and Fig. **4-20** show the result of this calculation for the bitcoin and crypto ethereum networks, respectively.

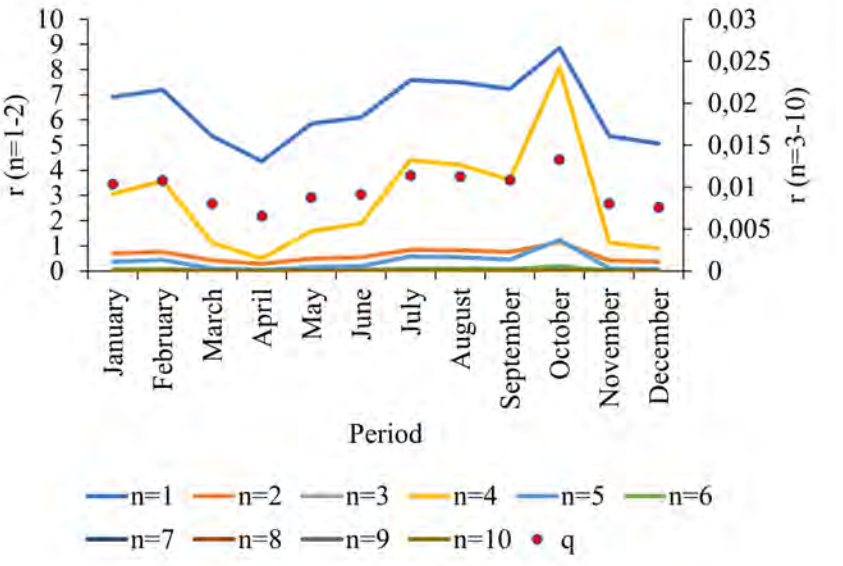
In the case of the bitcoin network, it can be seen that, as the hash rate for a miner grows higher and n assumes the lowest values, the success probability of carrying out a double-spending attack also grows higher, as expected. In particular, the highest success probability for carrying out a double-spending attack was registered for the miner F2Pool, since it has the highest hash rate and $n = 1$, as shown by Fig. **4-19** (a). This probability was 0.55 in November 2020.



(a) F2 Pool



(b) ViaBTC



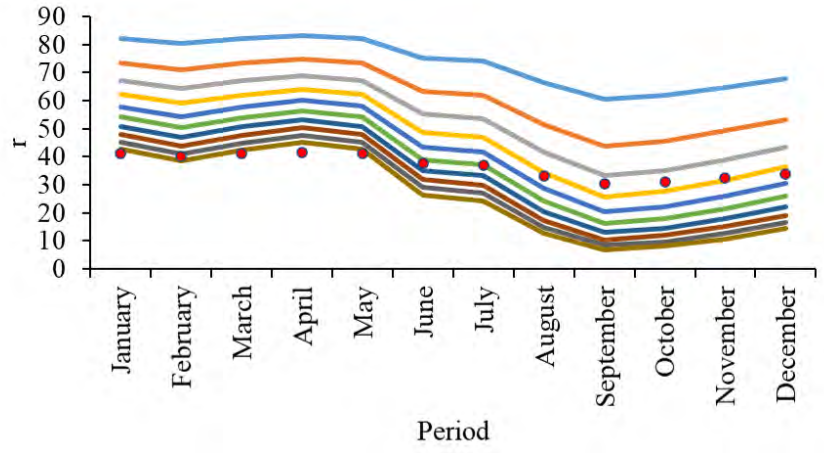
(c) BTC.TOP

Figure 4-19: Probability of Success of a Double Spending Attack on bitcoin Miners

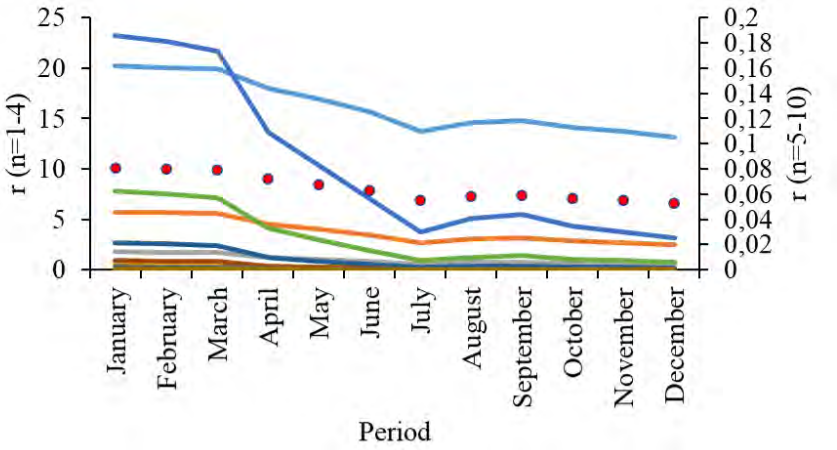
For the other cases, for all miners, it is observed that the probability of performing a double-spending attack is less than 0.4. In fact, most of the computed probabilities are less than 0.2. In Fig. **4-19** (c) it is observed that, in the specific case of the BTC.TOP Miner, which presents a lower hash rate, the probabilities to carry out the attack are very low, being 0.0886 for one confirmation, 0.0114 for two confirmations, and less than 0.0017 for $n \geq 3$.

For the crypto ethereum network, the general trend for the success probability of carrying out the attack is similar to that of the bitcoin network. In particular, the success probability is directly proportional to the share of the miner and inversely proportional to the number of confirmations. It should be noted that for the miner 0x5a0b Fig. **4-20** (a), which has the greatest hash power, the probabilities of performing the double-spending attack are above 0.65, sometimes reaching values above 0.7 when one commit is made.

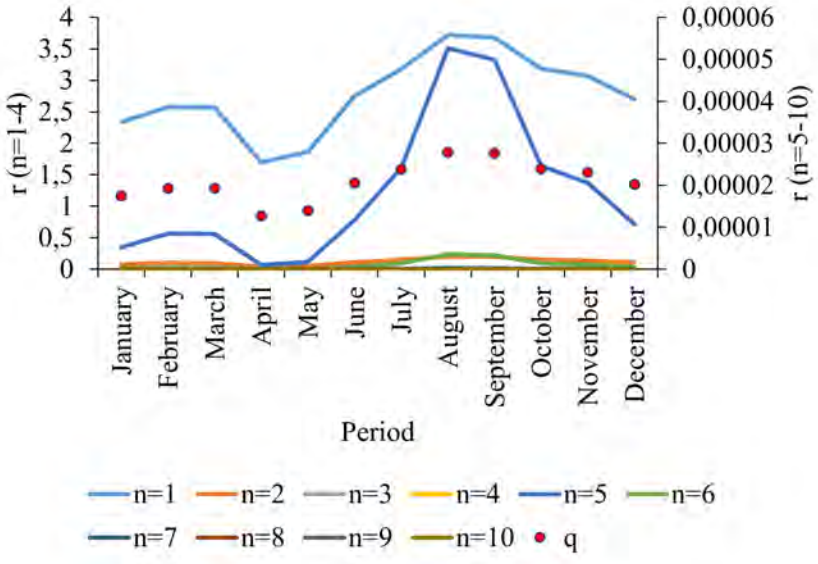
For the miner 0x52bc Fig. **4-20** (b), it was fount that it had success probabilities of performing the attack greater than 0.01 for $n \leq 2$. However, for the other cases, these probabilities are less than 0.01, reaching down to the order of 0.1×10^{-7}. In the case of the miner with the lowest share power, such as 0x005e Fig. **4-20** (c), the success probability of a double-spending attack is greater than 0.01 only for $n = 1$. For $n \geq 2$, on the other hand, the probabilities are less than 0.002, reaching down to the order of 6.2×10^{-16}.



(a) 0x5a0b



(b) 0x52bc



(c) 0x005e

Figure 4-20: Probability of Success of a Double Spending Attack on Crypto Ethereum Miners

4.3 Conclusions

The decrease in the number of miners along with the centralization of the hash rate/share are threats to the security of these blockchains (bitcoin and Ethereum), therefore understanding the behavior of the miners becomes a relevant research topic.

It is concluded that according to the analysis carried out, the centralization of the hash rate seems to be a real threat. On the one hand, for bitcoin, only 18 miners representing 0.00905014% of the total number of miners have 51.01% of the hash rate in the entire period. In other words, 99.9658% of the total miners only reach 35.88% of the total hash rate. On the other hand, for the crypto Ethereum network, only 13 miners representing 0.23% of the total number of miners collectively achieve 78.18% of the share, i.e. 98.82% of miners collectively achieve 6.10% of the total share.

In both scenarios, there is a real possibility that a 51% attack could take place if the most powerful miners get together, which violates the main general assumption of the blockchains. Now, there is a real negative incentive to perform such an attack because the credibility of the blockchain network will go to zero, as well as the value of the crypto assets; thus, a self-protection policy takes place in these public networks by its members. However, there are plenty of other blockchains, public and private, in which the value loss maybe not be too critical to discourage such attacks to obtain a specific asset. This centralization of hash rate/share and its risks imply that all new prevention mechanisms to address this kind of vulnerability must take this new situation seriously in their considerations.

5 On Proof-of-Accuracy Consensus Protocols

Consensus protocols are a fundamental part of any blockchain; although several protocols have been in operation for several years, they still have drawbacks. Proof-of-Work (PoW) and Proof-of-Stake (PoS) protocols are the most popular; however, they still have limitations. In particular, the consensus mechanism PoW is inefficient regarding energy consumed by its participants [Aponte et al., 2021a]. For instance, some may be susceptible to a 51% attack, which may suppose a high risk to the trustworthiness of the blockchains. Although this attack is theoretically possible, executing it in practice is often regarded as arduous because of the premise that, with sufficiently active members, it is not 'straightforward' to have much computing power. However, in the previous chapter it was seen that in current public blockchains, this risk real and makes it imperative to keep exploring new protocols.

The community has made efforts to solve this and other blockchain problems, which has resulted in the birth of alternative consensus protocols [Oyinloye et al., 2021]. This chapter presents a detailed proposal of a Proof-of-Accuracy protocol, which aims to democratize the miners' participation within a blockchain, control the miners' computing power, and mitigate the majority attacks.

As evidenced in Section 2.1, there is little research on Proof-of-Accuracy consensus protocols. In particular, its study and development have been theoretical. According to [Kudin et al., 2019], implementing this algorithm requires some components, such as selection of a coordinator, generation of a secret, generation and distribution in the network of parts of the secret, and competition between the participants to find the parts and reconstruct the secret. However, no proposal has presented a concrete protocol. This chapter presents a detailed proposal for the formalization of a Proof-of-Accuracy protocol.

The protocol proposal is presented progressively, starting with an initial blueprint (based on different components described in [Kudin et al., 2019] and its drawbacks), which is improved in terms of security. Earlier versions of the protocol feature the mentioned phases; however, the new version (see Section 5.1.7) removes the need for a

coordinator and combines the Proof-of-Work feature with access to random locations to improve the protocol's resistance to majority attacks.

The chapter is structured as follows: Section 5.0.1 presents an description of Proof-of-Accuracy consensus protocol. Section 5.1 presents the protocol proposal, introducing it progressively from earlier versions. This section presents the notation we use to describe our protocol and its earlier designs, a general description of the generic protocol, the assumptions, and the description of the proposed protocol. Section 5.2 presents an analysis of the proposed protocol. In particular, it presents a deep analysis of its mining process, computational costs, and security. Section 5.3 presents a prototype implementation of the proposed protocol. Section 5.4 presents a qualitative comparison between the proposed protocol and other consensus protocols proposed in the literature. Section 5.6, shows the conclusions and describes future research directions.

5.0.1 Proof-of-Accuracy Protocol

[Kudin et al., 2019] presented novel ideas for creating new consensus protocols, introducing two possible protocols: protocol simple tickets and Proof-of-Accuracy consensus protocol (PoAc); these ideas are theoretical with no concrete implementation.

PoAc features an effort or work (CPE) component since the selection of the node that wins the right to add the new block to the network is not only based on calculating the solution of a problem with a certain threshold of computational complexity but also should include a proof that the winning node has the necessary data pieces for the calculation of the solution of the problem.

These data pieces must be correctly defined and follow a random distribution to ensure that the task of collecting these pieces is stochastic and feasible in a given interval of time, which may be increased by adding some decoy pieces and distributing them with the genuine data pieces on the network [Oyinloye et al., 2021].

The network participants vie for accessing the data pieces needed to solve the mining problem, which helps participants with little computing skills have the opportunity to win the Proof-of-Work and add a new block to the chain. Figure **5-1** shows the flow of the Proof-of-Accuracy consensus protocol, which features a random selection of a coordinator, who generates a secret, divides it into shares, and distributes them among participants. The mining process then starts and consists of accessing these shareholders to reconstruct the secret. The mining party reconstructing the secret will acquire the right to add a new block to the blockchain.

A recent paper [Kaur et al., 2022] proposed a delegated Proof-of-Accessibility (DPoAC)

protocol, mostly based on the previous idea. It employed secret sharing, PoS with random selection, and an interplanetary file system (IPFS). This protocol is similar to the initial design presented in Section 5.1.5 and follows a similar flow as shown in Figure **5-1**. The main difference is that the coordinator stores the n shares of the secret in different n nodes on the IPFS network. For a mining party to acquire block creation rights, the party has to access these shareholders to reconstruct the secret.

As analyzed in Section 5.1.5, this approach has a drawback in that it heavily relies on the coordinator, meaning this node gains too much knowledge of the secret and may take advantage of it to favor any mining party.

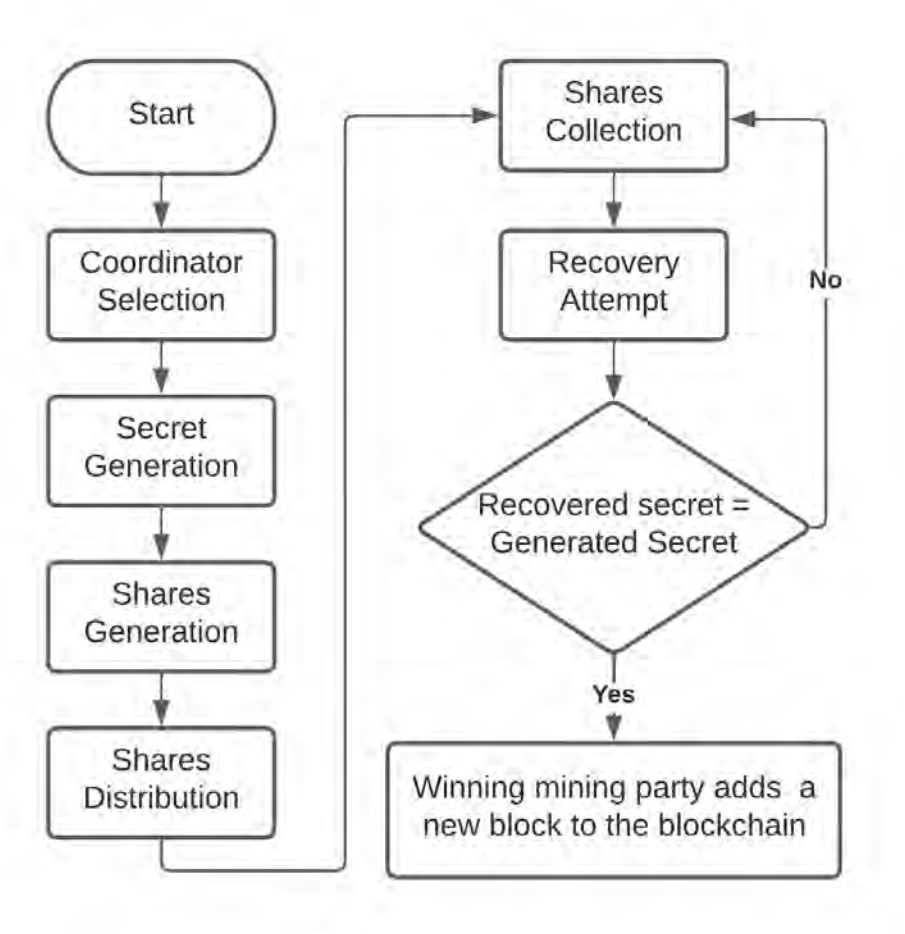


Figure 5-1: Proof of accuracy flowchart.

5.1 Proposed Protocol

This section introduces the proposed protocol. The Section 5.1.1 introduces the notation that will be used throughout the section. The Sections 5.1.5 and 5.1.6, describe earlier versions of the proposed protocol, highlighting their weaknesses and disadvantages. These earlier versions serve as a base to introduce the proposed protocol. Finally, the proposed protocol is presented in Section 5.1.7.

5.1.1 Notation

The notation to be used throughout the section is introduced. Specifically, Table **5-1** summarizes the notation.

Table 5-1: Summary of notation.

Symbol	Description
$t \in \mathbb{N}$	A positive integer.
$n \in \mathbb{N}$	A positive integer.
$m \in \mathbb{N}$	A positive integer.
$m_0 \in \mathbb{N}$	A positive integer.
$m_1 \in \mathbb{N}$	A positive integer.
$\mathbb{G} \in \mathbb{N}$	Denotes a cyclic group of order q
$q \in \mathbb{N}$	A prime number
$p = 2q + 1$	A safe prime number
$g \in \mathbb{G}$	A public generator of $\mathbb{G}$
$\mathbb{Z}_p$	The ring of integers modulo p
$\mathbb{ID}_m = \{1, 2, \ldots, m\}$	The set of m identifiers
$i \in \mathbb{ID}_m$	An identifier in $\mathbb{ID}_m$
$s_i(j) : \mathbb{ID}_m \setminus \{i\} \rightarrow \{1, -1\} \subseteq \mathbb{Z}_q$ [Hoogerwerf et al., 2021]	$s_i(j) = \begin{cases} 1 & if \quad i > j \\ -1 & if \quad i < j \end{cases}$ (5-1)
$\mathcal{H}_0 : \{0,1\}^* \rightarrow \{0,1\}^{m_0}$	Denotes a cryptographic hash function
$\mathcal{H}_1 : \{0,1\}^* \rightarrow \{0,1\}^{m_1}$	Denotes a cryptographic hash function
$0 \leq p_{min} \leq 1 \in \mathbb{R}$	Denotes a probability (protocol parameter)

5.1.2 General Description

The progressive versions of the protocol are built upon the cryptographic components required to compose a Proof-of-Accuracy (PoAc) protocol described by [Kudin et al., 2019]. In particular, according to [Kudin et al., 2019], a PoAc protocol features the selection of a coordinator among all the participants, the joint generation of a secret by all the participants, the generation of shares of the generated secret, decoy shares, the distribution of all of them over the network participants, the mining process among

the mining parties to reconstruct the generated secret, and the proof of recovering the secret by the winning party. The cryptographic tools used are described below.

Single Broadcast-Based Joint Random Number Generation Protocol

The proposed protocol uses a joint random number generation protocol as described in [Hoogerwerf et al., 2021, Kursawe et al., 2011]. According to their designers, these protocols do not require a secure network and need one transmission per network node. The protocol [Hoogerwerf et al., 2021] features additive aggregation instead of a multiplicative aggregation as in the protocol presented in [Kursawe et al., 2011].

The arithmetic carried out by m participants in the protocol [Hoogerwerf et al., 2021] works over a cyclic group $\mathbb{G} \subseteq \mathbb{Z}_p$ generated by g with order q, where q and $p = 2q+1$ are prime numbers. Each participant generates a Diffie-Hellman-like key pair $(\kappa_i, pk_i = g^{\kappa_i})$ and shares its public key with the other network participants. With the set of public keys, the participant i computes $R_i = \sum_{j=1, j\neq i}^{m} s_i(j) pk_j^{\kappa_i} \in \mathbb{Z}_p$, generates $c_i \in \mathbb{Z}_p$, and calculates $\gamma_i = c_i + R_i$. At the last step, a randomly chosen coordinator takes the role of the combiner and collects all the γ_i values, and computes $\alpha = \sum_{i=1}^{m} \gamma_i = \sum_{i=1}^{m} c_i + \sum_{i=1}^{m} R_i = \sum_{i=1}^{m} c_i$, since $\sum_{i=1}^{m} R_i = 0$ (see proof in [Hoogerwerf et al., 2021]), which will be the input secret passed to the next sub-protocol.

Shamir's Secret Sharing Scheme

Shamir's secret sharing scheme aims to divide a secret α in s parts (α_1,$\alpha_2 \ldots \alpha_s$) so that with any t of the s parts, α can be reconstructed, but every set of $t-1$ reveals nothing about α [Boneh and Shoup, 2020]. Shamir's secret sharing scheme stems from a general fact of polynomial interpolation: A polynomial of maximum degree $t-1$ defined over a field is fully determined by t points of the polynomial. In our particular case, we work it over $\mathbb{Z}_q$, which is a field when q is a prime number [Boneh and Shoup, 2020].

The s parts are called genuine parts, the valid ones to reconstruct the secret. Also, $n-s$ non-genuine parts are generated and distributed among the participants. Combining genuine and non-genuine parts allows for adjusting the difficulty in collecting t genuine parts to recover the secret. Algorithms 1 and 2 describe the inner workings of Shamir's secret sharing scheme.

Schnorr Non-Interactive Zero-Knowledge Scheme

Schnorr non-interactive zero-knowledge (NIZK) scheme is a non-interactive variant of the three-pass Schnorr identification scheme. The Schnorr NIZK scheme allows a prover

Algorithm 1 generates the t-out-of-s sharing of α

function $\mathtt{G}_{sh}(s, t, \alpha)$
  choose $a_1, \ldots, a_{t-1} \leftarrow \mathbb{Z}_q$ at random and define a polynomial

$$f(x) := a_{t-1}x^{t-1} + a_{t-2}x^{t-2} + \ldots + a_1 x + \alpha \in \mathbb{Z}_q.$$

  for $i \leftarrow 1$ to s **do**
    select x_i randomly from $\mathbb{Z}_q$, such that $x_i \neq x_j$ for all $j \in \{1, \ldots, i-1\}$
    $y_i \leftarrow f(x_i)$
    $\alpha_i \leftarrow (x_i, y_i)$
  end for
  return $(\alpha_1, \alpha_2, \ldots, \alpha_s)$
end function

Algorithm 2 recovers α given t genuine shares

function $\mathtt{C}_{sh}(\alpha_1 = (x_1, y_1), \ldots, \alpha_t = (x_t, y_t))$
  $\alpha \leftarrow 0 \in \mathbb{Z}_q$
  for $i \leftarrow 1$ to t **do**
    $\lambda_i \leftarrow 1 \in \mathbb{Z}_q$
    for $j \leftarrow 1$ to t **do**
      if $i \neq j$ **then**
        $\lambda_i \leftarrow \lambda_i \cdot \frac{-x_j}{x_i - x_j}$
      end if
    end for
    $\alpha \leftarrow \alpha + y_i \cdot \lambda_i$
  end for
  return α
end function

to prove to any verifier their knowledge of a discrete logarithm without leaking any information about its value [Boneh and Shoup, 2020]. Algorithm 3 shows how a proof gets generated, while Algorithm 4 shows how a proof gets verified.

Algorithm 3 generates a proof

1: **function** GENPROOF(α, u, id)
2: $\alpha_t \xleftarrow{R} \mathbb{Z}_q$
3: $u_t \leftarrow g^{\alpha_t}$
4: $c \leftarrow \mathcal{H}_0(g, u_t, u, id)$.
5: $\alpha_z \leftarrow \alpha_t + c \cdot \alpha$
6: **return** (id, u_t, α_z)
7: **end function**

Algorithm 4 verifies a proof

1: **function** VERIFYPROOF$(id, u_t, \alpha_z, u))$
2: $c^{'} \leftarrow \mathcal{H}_0(g, u_t, u, id)$.
3: $u_z \leftarrow g^{\alpha_z}$.
4: **if** $u_z = u_t u^{c^{'}}$ **then**
5: **return** 1
6: **else**
7: **return** 0
8: **end if**
9: **end function**

Digital Signatures

A digital signature scheme SS consists of the following algorithms [Boneh and Shoup, 2020].

- G is a probabilistic algorithm that takes a security parameter. It outputs a pair $(\texttt{vk}, \texttt{sk})$, where sk is a secret signing key, and vk is a public verification key.
- sign is a probabilistic algorithm that is invoked as $\sigma \leftarrow \texttt{sign}(\texttt{sk}, m)$, where sk is a secret key (as output by the key generation algorithm) and m is a message. The algorithm outputs a signature σ.
- verify is a deterministic algorithm invoked as $\texttt{b} \leftarrow \texttt{verify}(\texttt{vk}, m, \sigma)$, where b is a bit. If $\texttt{b} = 1$, then it means the signature is accepted, or else it is rejected.

5.1.3 Threat Model

We assume a semi-honest adversary, i.e., one who corrupts parties but follows the protocol as specified. Under this threat model, the corrupt parties follow the rules of the protocol honestly but they may attempt to learn as much as possible from the messages they receive from other parties to control the creation of blocks in the chain. Furthermore, there may be several colluding corrupt parties combining their partial views to learn information. Semi-honest adversaries are regarded as passive since they do not take any active actions other than attempting to learn private information by observing a view of the protocol execution. Semi-honest adversaries are also commonly called honest-but-curious.

The view of a party is regarded as its private inputs, its memory data, and the list of all messages received during the protocol. In this sense, the view of an adversary is composed of the combined views of all corrupt parties. Therefore, under this threat model, any information the adversary learns from the run of the protocol must be a computable function on the input of its combined view [Evans et al., 2018].

5.1.4 Initial Assumptions

We assume that each participant has access to a long-term key pair ($\texttt{sk}, \texttt{vk}$) generated by a signature scheme $\texttt{SS}$. Furthermore, each participant has access to a digital certificate that proves the validity of the corresponding public verification key $\texttt{vk}$. Additionally, when two participants want to communicate, a secure channel is established between them via a protocol such as Transport Layer Security (TLS) [Boneh and Shoup, 2020].

5.1.5 Initial Design

Here, we present our first attempt to build a proof of accuracy protocol. In particular, this initial design is a proof-of-concept based on the cryptographic components required to compose a Proof-of-Accuracy (PoAc) protocol described by [Kudin et al., 2019]. Specifically, [Kudin et al., 2019] presents the main cryptographic constituents to build such a protocol, but they do not present a concrete cryptographic construction of the protocol.

It is assumed that there are $m + 1$ participants. One of them assumes the role of coordinator with identifier 0. Additionally, each of the m remaining participants is given a unique identifier $i \in \mathbb{ID}_m$. The arithmetic works over a cyclic group $\mathbb{G} \subseteq \mathbb{Z}_p$ generated by g and whose order is a prime number q with $p = 2q+1$ (p a prime number). The initial design runs as follows.

1. Participant i generates an ephemeral key pair by selecting the private key $\kappa_i \in \mathbb{Z}_q$ at random and computing the public key $pk_i \leftarrow g^{\kappa_i} \in \mathbb{G}$
2. Participant i sends the ephemeral public key pk_i to the other $m-1$ participants.
3. Once participant i receives other participants' ephemeral public keys, the participant computes R_i as $R_i = \sum_{j=1, j\neq i}^{m} s_i(j) pk_j^{\kappa_i}$, where s is the function defined in Equation (5-1).
4. Each participant i selects $c_i \in \mathbb{Z}_q$ at random and computes $\gamma_i = c_i + R_i$. The participant then sends its γ_i to the coordinator.
5. Once the coordinator receives γ_i's from all participants, the coordinator will compute the secret α as

$$\alpha = \sum_{i=1}^{m} \gamma_i = \sum_{i=1}^{m} c_i + \sum_{i=1}^{m} R_i = \sum_{i=1}^{m} c_i,$$

   since $\sum_{i=1}^{m} R_i = 0$.
6. The coordinator generates s shares of α, $(\alpha_1, \ldots, \alpha_s) \leftarrow \mathtt{G}_{sh}(s, t, \alpha)$, and $n-s$ random points $\alpha_{s+1}, \alpha_{s+2}, \ldots \alpha_n$ from $\mathbb{Z}_q \times \mathbb{Z}_q$. The coordinator then computes $u = g^{\alpha}$ and shuffles the genuine and non-genuine points, forming the list $A = [\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_n}]$, with $1 \leq i_1, i_2, \ldots, i_n \leq n$. The coordinator now computes $\sigma_0 = \mathtt{sign}(\mathtt{sk}_0, \mathcal{H}_1(A \parallel u \parallel \mathtt{B}_l))$ where $\mathtt{B}_l$ is the last block in the blockchain. The coordinator now makes A, u, σ_0 publicly accessible to all participants.
7. At this point, the mining process begins. A mining party will attempt to reconstruct the secret α by finding t genuine points from A. In particular, the participant first collects A, u, σ_0 from the coordinator, and may check whether σ_0 is a valid signature for $\mathcal{H}_1(A \parallel u \parallel \mathtt{B}_l)$. The participant then selects t points $\alpha_1, \ldots, \alpha_t$, computes $\alpha' \leftarrow \mathtt{C}_{sh}(\alpha_1, \ldots, \alpha_t)$, and checks whether u is equal to $g^{\alpha'}$. Once the participant finds t suitable points, the participant will proceed with step 8. Otherwise, the participant will attempt to find t genuine points.
8. A mining party with identifier id proves its knowledge of the correct α' to other participants by using the Schnorr non-interactive zero-knowledge scheme. Specifically,

   a) The participant computes $\mathtt{proof} = (id, u_t, \alpha_z) \leftarrow \mathtt{genProof}(\alpha', u, id)$. He then publishes (m_{id}, σ_{id}) to the network, where

$$m_{id} \leftarrow (\mathtt{proof}, A, u, \sigma_0)$$

   and $\sigma_{id} \leftarrow \mathtt{sign}(\mathtt{sk}_{id}, \mathcal{H}_1(m_{id}))$.

b) Any verifier can check a solution (m_{id}, σ_{id}) by calling the function `check` shown by Algorithm 5.

Algorithm 5 checks a solution

```
1: function CHECK(m_id, σ_id, vk_id, vk_0, B_l)
2:     b_0 ← verify(vk_id, H_1(m_id), σ_id)
3:     if b_0 = 1 then
4:         (proof, A, u, σ_0) ← m_id
5:         b_1 ← verify(vk_0, H_1(A ‖ u ‖ B_l), σ_0)
6:         (id, u_t, α_z) ← proof
7:         b_2 ← verifyProof(id, u_t, α_z, u)
8:         if b_1 = 1 and b_2 = 1 then
9:             return Accept
10:        else
11:            return Reject
12:        end if
13:    else
14:        return Reject
15:    end if
16: end function
```

Analysis of the Initial Design

Here, the initial design is analyzed

Correctness Step 7 is analyzed of the initial design. Note that what the coordinator does is to call $\mathtt{G}_{sh}(s, t, \alpha)$, creating s genuine shares for α, any t of which serves to reconstruct α. Hence, any mining party that finds t genuine shares among the n entries in A will successfully reconstruct the secret.

Drawbacks The major drawback of the initial design is that it heavily relies on the coordinator since this coordinator aggregates all γ_i to obtain the secret α and then computes the s shares of α and $n - s$ random points, which means the coordinator gains too much knowledge of α and, hence, may take advantage of this knowledge to favor any mining party.

Ideally, this coordinator must not know α, neither the s genuine shares of α nor the $n - s$ random points, i.e., the coordinator only should serve as an aggregator of data.

The following design improves upon the initial one by exploiting further the aggregating protocols [Hoogerwerf et al., 2021, Kursawe et al., 2011] to compute the genuine and non-genuine shares securely.

5.1.6 An Improved Design

Let us assume that there are t participants. One of them assumes the role of the coordinator with identifier 0. Each participant other than the coordinator has a unique identifier $i \in \mathbb{ID}_{t-1}$. The arithmetic works over a cyclic group $\mathbb{G} \subseteq \mathbb{Z}_p$ generated by g, whose order is a prime number q with $p = 2q + 1$ (p a prime number). The improved design runs as follows:

1. Participant i generates $n + 1$ ephemeral key pairs

   $$(\kappa_{i,0}, g^{\kappa_{i,0}}), (\kappa_{i,1}, g^{\kappa_{i,1}}), \ldots, (\kappa_{i,n}, g^{\kappa_{i,n}})$$

   by randomly selecting $\kappa_{i,k} \in \mathbb{Z}_q$ and computing $g^{\kappa_{i,k}}$ for each $k \in \{0, 1, \ldots, n\}$.

2. Participant i sends its ephemeral public keys

   $$(g^{\kappa_{i,0}}, g^{\kappa_{i,1}}, g^{\kappa_{i,2}}, \ldots, g^{\kappa_{i,n}})$$

   to all other participants.

3. After receiving the ephemeral public keys from each other participant, the participant i computes the following vector

   $$R_i = (R_{i,0}, R_{i,1}, \ldots, R_{i,n}),$$

   where

   $$R_{i,0} = \prod_{j=1, j\neq i}^{t-1} (g^{\kappa_{j,0}})^{s_i(j)\kappa_{i,0}}$$

   and

   $$R_{i,k} = \sum_{j=1, j\neq i}^{t-1} s_i(j)(g^{\kappa_{j,k}})^{\kappa_{i,k}}$$

   for each $k \in \{1, 2, \ldots, n\}$.

4. Participant i selects $\gamma_i, c_i \xleftarrow{R} \mathbb{Z}_q$ and a probability $p_i \xleftarrow{R} [p_{min}, 1]$. This participant computes the vector $C_i = (g^{\gamma_i} R_{i,0}, e_{i,1} + R_{i,1}, \ldots, e_{i,n} + R_{i,n})$, where

   $$e_{i,k} = \begin{cases} c_i k^i + \gamma_i \bmod q \text{ with probability } p_i \\ z \xleftarrow{R} \mathbb{Z}_q \quad \text{with probability } 1 - p_i \end{cases}$$

   and sends C_i to the coordinator

5. Once the coordinator receives each C_i for $i = 1, \ldots, t-1$, the coordinator will compute

$$A = \left(\prod_{i=1}^{t-1} C_{i,0}, \sum_{i=1}^{t-1} C_{i,1}, \ldots, \sum_{i=1}^{t-1} C_{i,n} \right) \tag{5-2}$$

The coordinator now computes $\sigma_0 = \texttt{sign}(\texttt{sk}_0, \mathcal{H}_1(A \parallel \texttt{B}_l))$, where $\texttt{B}_l$ is the last block in the blockchain. The coordinator now makes A, σ_0 publicly accessible to all participants.

6. At this point, the mining process begins. A mining party first collects A and σ_0 from the coordinator, and may check whether σ_0 is a valid signature for $\mathcal{H}_1(A \parallel \texttt{B}_l)$. It then will attempt to find t unique indices $1 \leq k_1, k_2, \ldots, k_t \leq n$, such that $A_0 = g^{\alpha'}$, where

$$\alpha' \leftarrow \texttt{C}_{sh}((k_1, A_{k_1}), (k_2, A_{k_2}), (k_3, A_{k_3}), \ldots, (k_t, A_{k_t})).$$

Once the participants find t suitable points, they will proceed with step 7. Otherwise, they will attempt to find t genuine points.

7. A mining party with identifier id proves its knowledge of the correct α' to other participants by using the Schnorr non-interactive zero-knowledge scheme. Specifically,

   a) The participant computes $\texttt{proof} = (id, u_t, \alpha_z) \leftarrow \texttt{genProof}(\alpha', A_0, id)$. He then publishes (m_{id}, σ_{id}) to the network, where $m_{id} \leftarrow (\texttt{proof}, A, \sigma_0)$ and $\sigma_{id} \leftarrow \texttt{sign}(\texttt{sk}_{id}, \mathcal{H}_1(m_{id}))$.

   b) Any verifier can check a solution (m_{id}, σ_{id}) by calling the function `check` shown by Algorithm 6.

Analysis of the Improved Design

Here, the improved design is analyzed. Before diving into the details, the following claim is presented, which will be useful during the analysis.

Lemma 5.1.1 *Let us assume there are t participants and let $\mathbb{G}$ be a cyclic group, generated by g. Each participant i generates*

$$(\kappa_{i,1}, g^{\kappa_{i,1}}), \ldots, (\kappa_{i,n}, g^{\kappa_{i,n}}),$$

where $\kappa_{i,k}$ is taken randomly from $\mathbb{Z}_q$. For a fixed k, let participant i collect $g^{\kappa_{j,k}}$ for all $j \neq i$ and compute $R_{i,k} = \prod_{j=1, j\neq i}^{t} (g^{\kappa_{j,k}})^{s_i(j)\kappa_{i,k}}$, then

Algorithm 6 checks a solution

```
1: function CHECK(m_id, σ_id, vk_id, vk_0, B_l)
2:     b_0 ← verify(vk_id, H_1(m_id), σ_id)
3:     if b_0 = 1 then
4:         (proof, A, σ_0) ← m_id
5:         b_1 ← verify(vk_0, H_1(A ‖ B_l), σ_0)
6:         (id, u_t, α_z) ← proof
7:         b_2 ← verifyProof(id, u_t, α_z, A_0)
8:         if b_1 = 1 and b_2 = 1 then
9:             return Accept
10:        else
11:            return Reject
12:        end if
13:    else
14:        return Reject
15:    end if
16: end function
```

$$\prod_{i=1}^{t} R_{i,k} = 1$$

Proof.

Note that $\prod_{i=1}^{t} R_{i,k} = \prod_{i=1}^{t} \prod_{j=1, j\neq i}^{t} (g^{\kappa_{j,k}})^{s_i(j)\kappa_{i,k}}$. Consider the term $(g^{\kappa_{i,k}})^{s_j(i)\kappa_{j,k}}$. Since $-s_i(j) = s_j(i)$ for $i \neq j$, the term $(g^{\kappa_{i,k}})^{s_j(i)\kappa_{j,k}} = (g^{\kappa_{j,k}})^{-s_i(j)\kappa_{i,k}}$. Therefore both $(g^{\kappa_{j,k}})^{s_i(j)\kappa_{i,k}}$ and $(g^{\kappa_{j,k}})^{-s_i(j)\kappa_{i,k}}$ appear in the product. Therefore, the product is 1.

Correctness We analyze Steps 5 and 6 of the improved design. Let us assume that each C_i obtained from participant i was created with a fixed probability p_i. Note that since $\prod_{i=1}^{t-1} R_{i,0} = 1$, (by Lemma 5.1.1)

$$A_0 = \prod_{i=1}^{t-1} \mathtt{C}_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} R_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} \prod_{i=1}^{t-1} R_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} = g^{\sum_{i=1}^{t-1} \gamma_i} = g^{\alpha} = u$$

where $\alpha = \sum_{i=1}^{t-1} \gamma_i \bmod q$.

Let us fix a k with $1 \leq k \leq n$. Let us assume that $C_{i,k} = c_i k^i + \gamma_i + R_{i,k} \bmod q$ for all $1 \leq i \leq t-1$. By construction, this occurs with probability $\rho = p_1 p_2 \dots p_{t-1}$. Since

$\sum_{i=1}^{t-1} R_{i,k} = 0$ (By Lemma 5.1.1) then

$$A_k = \sum_{i=1}^{t-1} C_{i,k} = \sum_{i=1}^{t-1} c_i k^i + \sum_{i=1}^{t-1} \gamma_i + \sum_{i=1}^{t-1} R_{i,k} = P(k)$$

where $P(x) = c_{t-1}x^{t-1} + c_{t-2}x^{t-2} + \ldots + c_1 x + \alpha$.

If the mining party finds t suitable $k_1, k_2, \ldots, k_t$, then

$$A_{k_1} = P(k_1), A_{k_2} = P(k_2) \ldots, A_{k_t} = P(k_t).$$

Therefore, $A_0 = g^{\alpha'}$, where

$$\alpha' = \mathtt{C}_{sh}((k_1, A_{k_1}), (k_2, A_{k_2}), \ldots, (k_t, A_{k_t})).$$

Drawbacks The improved design still relies on a coordinator, but now this coordinator does not know α, neither the s genuine shares of α nor the $n - s$ random points, i.e., the coordinator only serves as an aggregator of data; if the coordinator wants to know the secret α, it will have to perform step 6 of the improved design as any other mining party would. However, having a coordinator still presents a unique point of failure for this approach. Also, it solely relies on the Proof-of-Work performed by a mining party at step 6, which may not have a solution. To reduce power concentration, hence, the possibility of a 51% attack [Aponte-Novoa et al., 2021], and increase the fairness of the mining process, a new version should complement the Proof-of-Work with other approaches; for example, access to random locations, similar to PoC [Oyinloye et al., 2021].

Another drawback is that any mining party will attempt to recover α from A at step 6. Ideally, any mining party should only know how to reconstruct g^α rather than α. Another issue may arise if the cyclic group $\mathbb{G}$ is set to another one (e.g., a subgroup of the points of an elliptic curve). If that is the case, a mapping to associate each element of $\mathbb{G}$ with an element of $\mathbb{Z}_q$ will be required. In this version, this is not a problem since we are assuming the arithmetic works over a cyclic group $\mathbb{G} \subseteq \mathbb{Z}_p$ generated by g with order q, where $p = 2q + 1$ and p, q are prime numbers. Hence, the computation of

$$R_{i,k} = \sum_{j=1, j\neq i}^{t-1} s_i(j)(g^{\kappa_{j,k}})^{\kappa_{i,k}}$$

for each $k \in \{1, 2, \ldots, n\}$ does not present any inconvenient.

Our final version deals with the drawbacks of the improved design by introducing the following features:

- Permit a mining party to access C_i at different locations and compute a new vector A independently with the collected C_i's.
- Use only the multiplicative version of the aggregating protocol [Kursawe et al., 2011] to compute the ciphertexts of the shares.
- Exploit the homomorphic properties of ElGamal-based cryptographic schemes to allow any mining party to reconstruct g^{α} from the ciphertexts of the shares.

5.1.7 Our Proposed Protocol

Let us assume that there are $t-1$ participants. Each participant has a unique identifier $i \in \mathbb{ID}_{t-1}$. The arithmetic works over a cyclic group $\mathbb{G}$ generated by g and whose order is a prime number q. Our proposed protocol runs as follows.

1. Participant i generates $n+1$ ephemeral key pairs

   $$(\kappa_{i,0}, g^{\kappa_{i,0}}), (\kappa_{i,1}, g^{\kappa_{i,1}}), \ldots, (\kappa_{i,n}, g^{\kappa_{i,n}})$$

   by selecting $\kappa_{i,k} \in \mathbb{Z}_q$ at random and computing $g^{\kappa_{i,k}}$ for each $k \in \{0, 1, \ldots, n\}$.

2. Participant i sends its ephemeral public keys

   $$(g^{\kappa_{i,0}}, g^{\kappa_{i,1}}, g^{\kappa_{i,2}}, \ldots, g^{\kappa_{i,n}})$$

   to all other participants.

3. After receiving the ephemeral public keys from the other $t-2$ participants, participant i computes the following vector

   $$R_i = (R_{i,0}, R_{i,1}, \ldots, R_{i,n}),$$

   where

   $$R_{i,k} = \prod_{j=1, j \neq i}^{t-1} (g^{\kappa_{j,k}})^{s_i(j)\kappa_{i,k}}$$

   for each $k \in \{0, 1, \ldots, n\}$

4. At this point, the mining process begins. A mining party will have to contact each participant i and request a $\mathtt{C}_i$ from it. Specifically, upon request, the participant i executes $(\mathtt{C}_i, \sigma_i) \leftarrow \mathtt{generateC}(\mathtt{sk}_i, R_i, \mathtt{B}_l)$, shown by Algorithm 7, where $\mathtt{B}_l$ is the last block in the blockchain.

   The participant i then sends $(\mathtt{C}_i, \sigma_i)$ to the requesting mining party. Note that the mining party may check whether σ_i is a valid signature for $\mathcal{H}_1(\mathtt{C}_i \parallel \mathtt{B}_l)$. Once

Algorithm 7 generates the pair $(\mathtt{C}_i, \sigma_i)$

function GENERATEC($\mathtt{sk}_i, R_i, \mathtt{B}_l$)
  $p_i \xleftarrow{R} [p_{min}, 1]$.
  $c_i \xleftarrow{R} \mathbb{Z}_q$
  $\gamma_i \xleftarrow{R} \mathbb{Z}_q$
  $\mathtt{C}_{i,0} \leftarrow g^{\gamma_i} R_{i,0}$
  for $k \leftarrow 1$ to n **do**
    $p \xleftarrow{R} [0, 1]$
    if $p \leq p_i$ **then**
      $e_{i,k} \leftarrow c_i k^i + \gamma_i \bmod q$
    else
      $e_{i,k} \xleftarrow{R} \mathbb{Z}_q$
    end if
    $\mathtt{C}_{i,k} \leftarrow g^{e_{i,k}} R_{i,k}$
  end for
  $\sigma_i \leftarrow \mathtt{sign}(\mathtt{sk}_i, \mathcal{H}_1(\mathtt{C}_i \parallel \mathtt{B}_l))$.
  return $(\mathtt{C}_i, \sigma_i)$
end function

the mining party contacts each participant i and collects the corresponding $\mathtt{C}_i$, the party then computes A as

$$A = \left(\prod_{i=1}^{t-1} \mathtt{C}_{i,0}, \prod_{i=1}^{t-1} \mathtt{C}_{i,1}, \ldots, \prod_{i=1}^{t-1} \mathtt{C}_{i,n} \right).$$

The mining party's goal is to find unique indices $1 \leq k_1, k_2, \ldots, k_t \leq n$, such that

$$A_0 = w$$

with

$$w = (A_{k_1})^{\lambda_{k_1}} \cdot (A_{k_2})^{\lambda_{k_2}} \cdot \ldots \cdot (A_{k_t})^{\lambda_{k_t}}$$

and

$$\lambda_{k_j} = \prod_{\substack{r=1 \\ r \neq j}}^{t} \frac{-k_r}{k_j - k_r} \text{ for } 1 \leq j \leq t.$$

Once the participant finds t unique indices, the participant will proceed with step 5. Otherwise, this mining party may attempt step 4 again.

5. When a mining party with identifier id reconstructs w, i.e., finds suitable unique indices $1 \leq k_1, k_2, \ldots, k_t \leq n$, it will publish (m_{id}, σ_{id}) to the network, where

$$m_{id} = (id, k_1, k_2, \ldots, k_t, (\mathtt{C}_1, \sigma_1), (\mathtt{C}_2, \sigma_2), \ldots, (\mathtt{C}_{t-1}, \sigma_{t-1})),$$

and

$$\sigma_{id} = \mathtt{sign}(\mathtt{sk}_{id}, \mathcal{H}_1(m_{id})).$$

6. Any verifier can check a solution (m_{id}, σ_{id}) by calling the function `check` shown by Algorithm 8.

Algorithm 8 checks a solution

function CHECK($m_i, \sigma_{id}, \mathtt{vk}_{id}, \mathtt{vk}_1, \mathtt{vk}_2, \ldots, \mathtt{vk}_{t-1}, \mathtt{B}_l$)
  $b_0 \leftarrow \mathtt{verify}(\mathtt{vk}_{id}, \mathcal{H}_1(m_{id}), \sigma_{id})$.
  if $b_0 = 1$ **then**
    $(id, k_1, k_2, \ldots, k_t, (\mathtt{C}_1, \sigma_1), (\mathtt{C}_2, \sigma_2), \ldots (\mathtt{C}_{t-1}, \sigma_{t-1})) \leftarrow m_{id}$
    for $i \leftarrow 1$ to $t-1$ **do**
      if $\mathtt{verify}(\mathtt{vk}_i, \mathcal{H}_1(\mathtt{C}_i \parallel \mathtt{B}_l), \sigma_i) = 0$ **then**
        return Reject
      end if
    end for
    Compute

$$A = \left(\prod_{i=1}^{t-1} \mathtt{C}_{i,0}, \prod_{i=1}^{t-1} \mathtt{C}_{i,1}, \ldots, \prod_{i=1}^{t-1} \mathtt{C}_{i,n} \right).$$

    With the given indices $1 \leq k_1, k_2, \ldots, k_t \leq n$, compute $\lambda_{k_j} = \prod_{\substack{r=1 \\ r \neq j}}^{t} \frac{-k_r}{k_j - k_r}$ for $1 \leq j \leq t$.
    Compute $w = (A_{k_1})^{\lambda_{k_1}} \cdot (A_{k_2})^{\lambda_{k_2}} \ldots \cdot (A_{k_t})^{\lambda_{k_t}}$
    if $w = A_0$ **then**
      return Accept
    else
      return Reject
    end if
  else
    return Reject
  end if
end function

5.2 Protocol Analysis

In this section, a deeper analysis of the proposed protocol is done.

5.2.1 Correctness of Our Proposed Protocol

We now analyze step 4 of our proposed protocol. Let us assume that each $\mathtt{C}_i$ obtained from participant i was created with a fixed probability p_i. Note that $\prod_{i=1}^{t-1} R_{i,k} = 1$ for any $0 \leq k < n$ (by Lemma 5.1.1), then

$$A_0 = \prod_{i=1}^{t-1} \mathtt{C}_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} R_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} \prod_{i=1}^{t-1} R_{i,0} = \prod_{i=1}^{t-1} g^{\gamma_i} = g^{\sum_{i=1}^{t-1} \gamma_i} = g^{\alpha} = u$$

where $\alpha = \sum_{i=1}^{t-1} \gamma_i \bmod q$.

Let us fix a k with $1 \leq k \leq n$. Let us assume that $\mathtt{C}_{i,k} = g^{e_{i,k}} R_{i,k}$ with $e_{i,k} = c_i k^i + \gamma_i \bmod q$ for all $1 \leq i \leq t-1$. By construction, this occurs with probability $\rho = p_1 p_2 \dots p_{t-1}$. Therefore,

$$A_k = \prod_{i=1}^{t-1} \mathtt{C}_{i,k} = \prod_{i=1}^{t-1} g^{e_{i,k}} R_{i,k} = \prod_{i=1}^{t-1} g^{e_{i,k}} \prod_{i=1}^{t-1} R_{i,k} = g^{\sum_{i=1}^{t-1} c_i k^i + \sum_{i=1}^{t-1} \gamma_i} = g^{P(k)}$$

where $P(X) = c_{t-1} X^{t-1} + c_{t-2} X^{t-2} + \ldots + c_1 x + \alpha$ with $\alpha = \sum_{i=1}^{t-1} \gamma_i$.

If the mining party finds t suitable $k_1, k_2, \ldots, k_t$, then

$$A_{k_1} = g^{P(k_1)}, A_{k_2} = g^{P(k_2)} \ldots, A_{k_t} = g^{P(k_t)}.$$

Hence,

$$A_0 = g^{\alpha} = g^{P(0)} = (A_{k_1})^{\lambda_{k_1}} \cdot (A_{k_2})^{\lambda_{k_2}} \ldots \cdot (A_{k_t})^{\lambda_{k_t}},$$

where

$$\lambda_{k_j} = \prod_{\substack{r=1 \\ r \neq j}}^{t} \frac{-k_r}{k_j - k_r} \text{ with } 1 \leq j \leq t.$$

The section 5.2.2, extend the analysis of this mining process.

5.2.2 Mining Process

Here further analyzed step 4 of the proposed protocol.

Estimating n and s

Let t be fixed and let $\mathtt{C}_i$ be the array obtained from participant i at step 4 of the proposed protocol for $1 \leq i \leq t-1$. Assume that $\mathtt{C}_{i,k}$ for $1 \leq k \leq n$ has been created with probability p_i by the participant i. Note that the value A_k depends on $\mathtt{C}_{i,k}$ for $1 \leq i \leq t-1$. Therefore, $\rho = \prod_{i=1}^{t-1} p_i$ is the probability of obtaining a genuine entry A_k in A. If the random variable X is defined as the number of genuine A_k's in the sequence of values $A_1, A_2, A_3, \ldots, A_n$, then X follows a binomial distribution with success probability ρ in the sequence of n independent trials. Therefore, the expected number of genuine A_k's is given by $E[X] = n \cdot \rho$. By choosing n, such that $E[X] \geq t$, s is expected to be greater than t. Since p_{min} is known and $0 \leq p_{min} \leq p_1, p_2, \ldots, p_{t-1} \leq 1$, then $n \cdot p_{min}^{t-1} \leq n \cdot \rho = E[X]$, and so

$$E[X] \geq s_{min} = n \cdot p_{min}^{t-1} \geq t$$

when $n \geq \frac{t}{p_{min}^{t-1}}$.

Expected Number of Attempts for Recovering A_0

The expected number of attempts is estimated to solve the Proof-of-Work assuming that the number of genuine A_i's is at least t, i.e. $s \geq t$ (otherwise, the PoW cannot be solved). Let $\mathcal{Z}$ be the set of all combinations $[k_1, k_2, \ldots, k_t]$ out of $[1, 2, \ldots, n]$.

A first strategy by the miner is to select $\Omega \subseteq \mathcal{Z}$ and call $\mathtt{mining}_0(\Omega, A)$ as shown by Algorithm 9. ω is called a genuine combination if the reconstructed w is equal to A_0 (i.e., line 7 of the function $\mathtt{mining}_0$ satisfies). Otherwise, ω is a non-genuine combination.

Algorithm 9 describes the first mining strategy

```
1:  function MINING_0(Ω, A)
2:      while True do
3:          ω ←R Ω
4:          [k_1, k_2, ..., k_t] ← ω
5:          Compute λ_{k_j} = ∏_{r=1, r≠j}^{t} −k_r / (k_j − k_r) for 1 ≤ j ≤ t.
6:          Compute w = (A_{k_1})^{λ_{k_1}} · (A_{k_2})^{λ_{k_2}} ... · (A_{k_t})^{λ_{k_t}}
7:          if w = A_0 then
8:              return ω
9:          end if
10:     end while
11: end function
```

Let $N = |\Omega|$ be the total number of combinations in Ω. Let K_1 be the number of

genuine combinations in Ω. Therefore, $K_2 = N - K_1$ is the number of non-genuine combinations in Ω. When $\Omega = \mathcal{Z}$, $N = \binom{n}{t}$ and $K_1 = \binom{s}{t}$

Let Y_0 be the random variable that counts the number of iterations to find a genuine combination in Ω using the strategy $\texttt{mining}_0$. Therefore, Y_0 follows a geometric distribution with success probability $\frac{K_1}{N}$. Hence,

$$Pr[Y_0 = y_0] = (1 - \frac{K_1}{N})^{y_0 - 1} \frac{K_1}{N}$$

for $y_0 = 1, 2, 3, \ldots$ and $= E[Y_0] = N/K_1$.

Note that at any iteration of $\texttt{mining}_0$, nothing is learned about a particular A_i, i.e., it does not learn whether A_i is genuine or non-genuine, but instead, it does learn whether a particular combination ω is genuine or not.

A seemingly better strategy is to pick $\Omega \subseteq \mathcal{Z}$ and call $\texttt{mining}_1(\Omega, A)$ as shown by Algorithm 10. Furthermore, if the mining party has access to computational resources, the party then may make a partition of $\mathcal{Z}$, i.e. pick $\Omega_1, \Omega_2, \ldots, \Omega_{n_p}$ with $\Omega_i \cap \Omega_j = \emptyset$ for $1 \leq i, j \leq n_p$ with $i \neq j$, and $\mathcal{Z} = \bigcup_{i=1}^{n_p} \Omega_i$; and sets and runs n_p parallel processing tasks, where the processing task i searches over Ω_i, viz. executes $\texttt{mining}_1(\Omega_i, A)$.

Algorithm 10 describes the second mining strategy strategy

```
1:  function MINING_1(Ω, A)
2:      for each ω ∈ Ω do
3:          [k_1, k_2, ..., k_t] ← ω
4:          Compute λ_{k_j} = ∏_{r=1, r≠j}^{t} −k_r / (k_j − k_r) for 1 ≤ j ≤ t.
5:          Compute w = (A_{k_1})^{λ_{k_1}} · (A_{k_2})^{λ_{k_2}} ... · (A_{k_t})^{λ_{k_t}}
6:          if w = A_0 then
7:              return ω
8:          end if
9:      end for
10:     return ⊥
11: end function
```

Let Y_1 be the random variable that counts the number of iterations to find a genuine combination in Ω using the strategy $\texttt{mining}_1$. According to [Ahlgren, 2014],

$$Pr[Y_1 = y_1] = \frac{\binom{N - y_1 + 1}{K_1}}{\binom{N}{K_1}} \frac{K_1}{N - y_1 + 1}$$

for any $y_1 \in \{1, \ldots, K_2 + 1\}$. Hence, $E_{Y_1} = E[Y_1] = \sum_{y_1=1}^{K_2+1} y_1 Pr[Y = y_1] = \frac{N+1}{K_1+1}$. Since $K_1 \leq N$, then $K_1 N + K_1 \leq K_1 N + N$ and, therefore, $E_{Y_1} = \frac{N+1}{K_1+1} \leq \frac{N}{K_1} = E_{Y_0}$.

Experimental Results

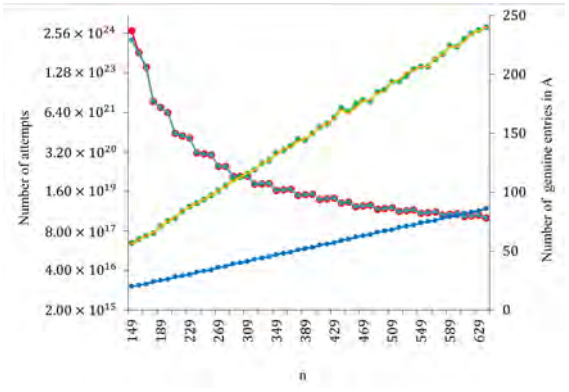
Let $\mathcal{P}$ be the set $\{0.9, 0.93, 0.95, 0.98, 1\}$ and $\mathcal{T}$ be the set $\{20, 30, 40, 50, 60\}$. The following experiment is carried out, in which a trial consists of the following steps.

1. Choose $p_{min} \in \mathcal{P}$, $t \in \mathcal{T}$. For the selected pair (p_{min}, t), choose a value for n, such that $n \geq \frac{t}{p_{min}^{t-1}}$.

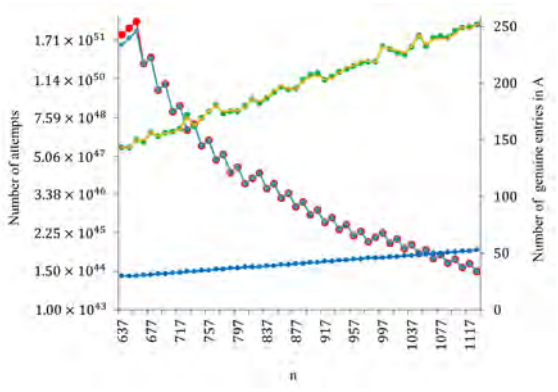
2. For the selected three-tuple (p_{min}, t, n), perform 100 runs of a modified mining phase (step 5 of the proposed protocol). At each run, the corresponding A is constructed; the numbers of genuine A_i (i.e., the value of s), $E[X]$, and s_{min} are computed. Moreover, $E[Y_0]$ and $E[Y_1]$ are computed, assuming $\Omega = \mathcal{Z}$.

3. At the end of the trial, the means of all values s, $E[X]$, s_{min}, $E[Y_0]$, and $E[Y_1]$ are respectively computed.

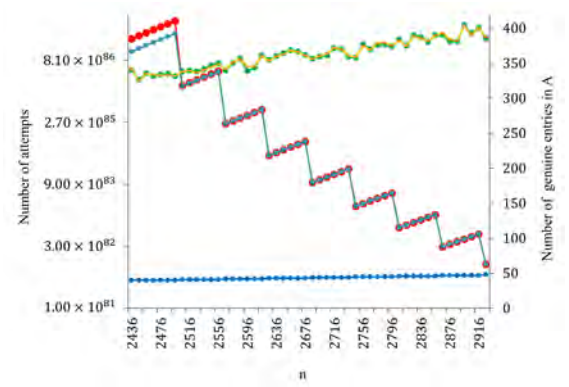
Figures **5-2**–**5-6** show the results obtained for $p_{min} \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.



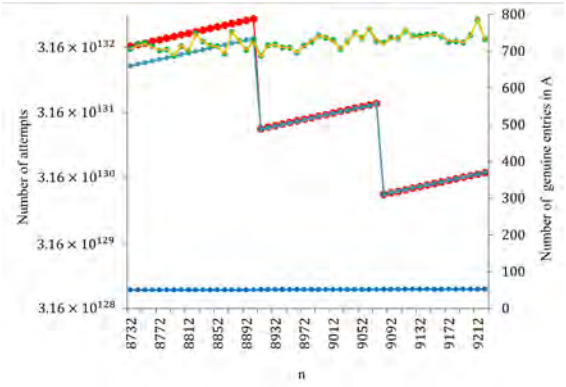
(a) t = 20



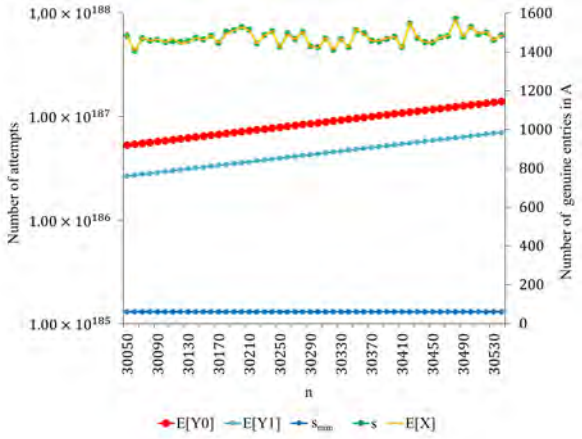
(b) t = 30



(c) t = 40

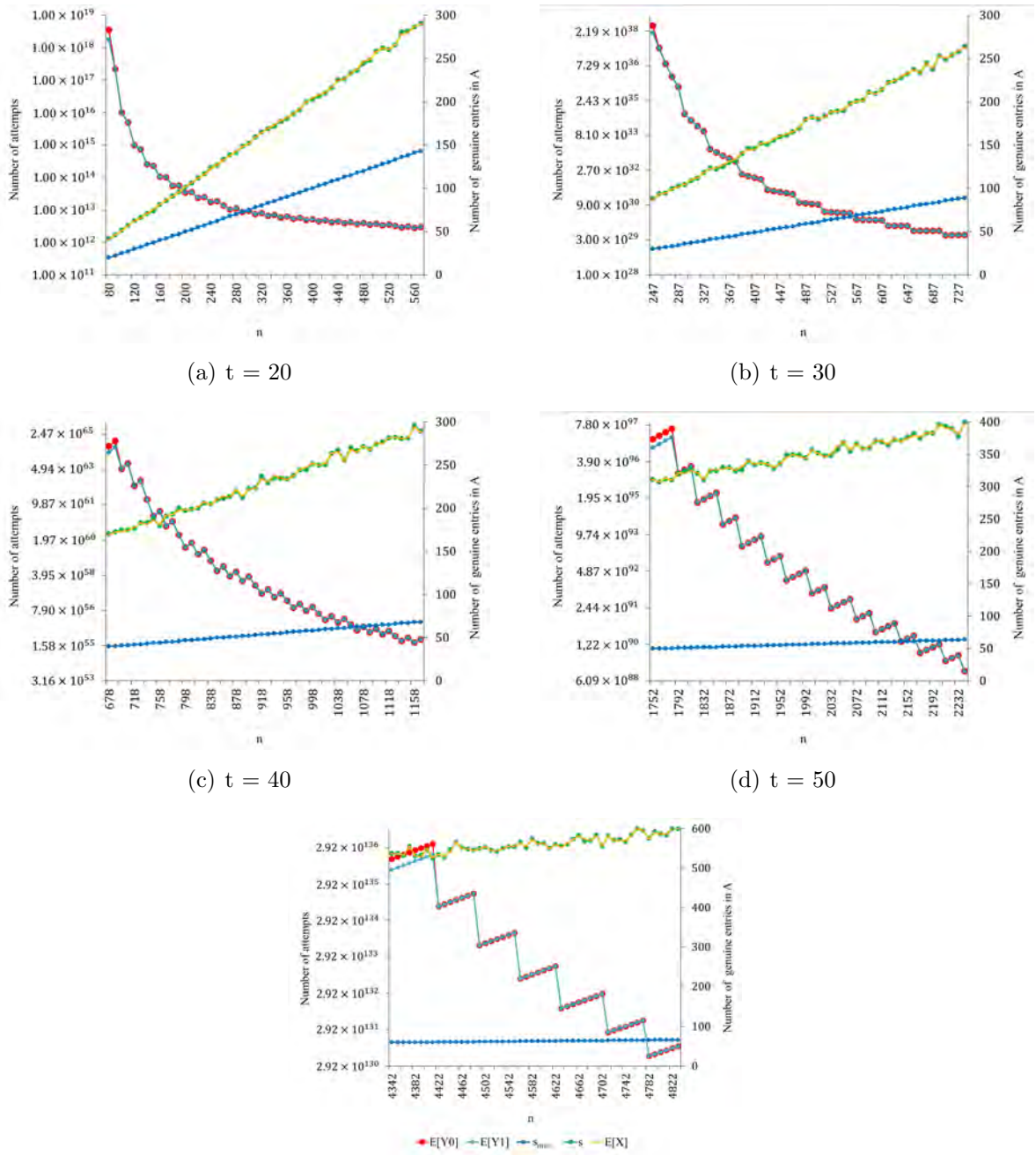


(d) t = 50



(e) t = 60

Figure 5-2: Results obtained for $p_{min} = 0.90 \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.



(a) t = 20

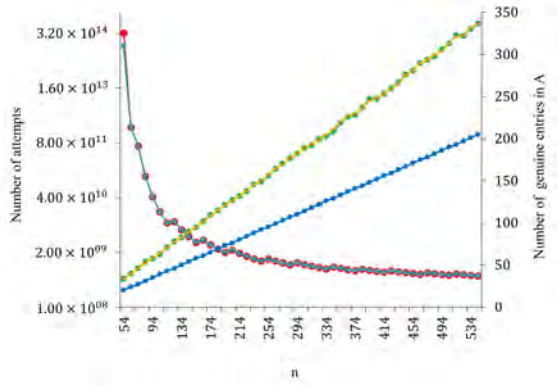
(b) t = 30

(c) t = 40

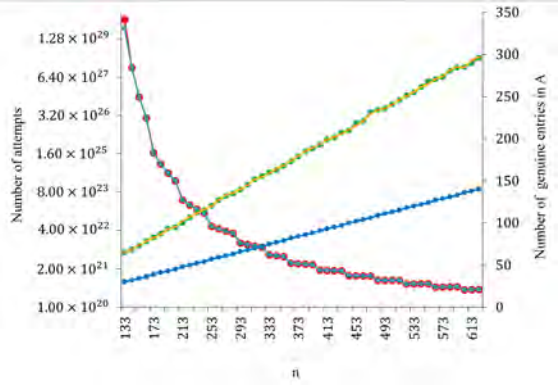
(d) t = 50

(e) t = 60

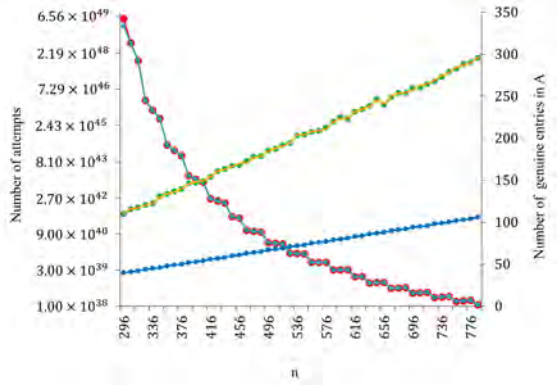
Figure 5-3: Results obtained for $p_{min} = 0.93 \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.



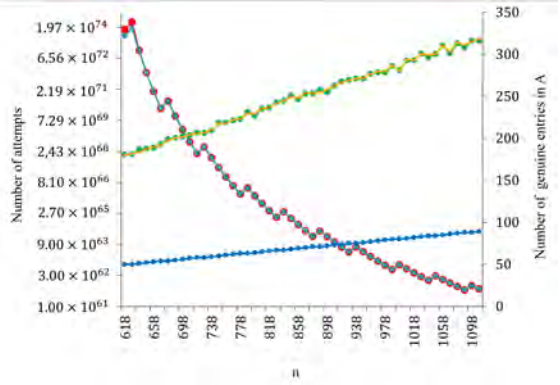
(a) t = 20



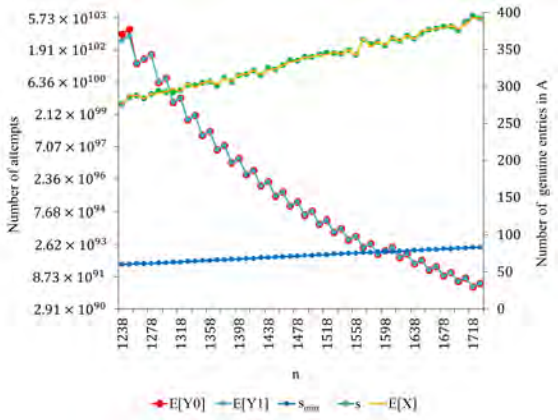
(b) t = 30



(c) t = 40

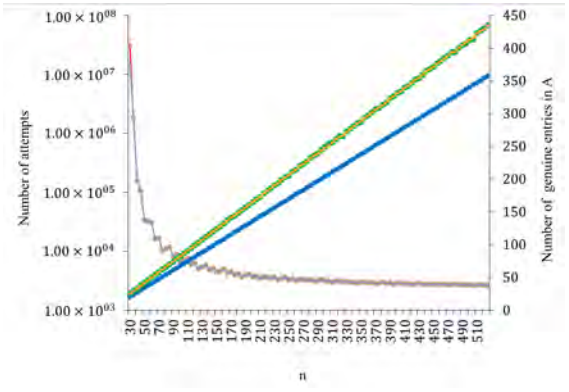


(d) t = 50

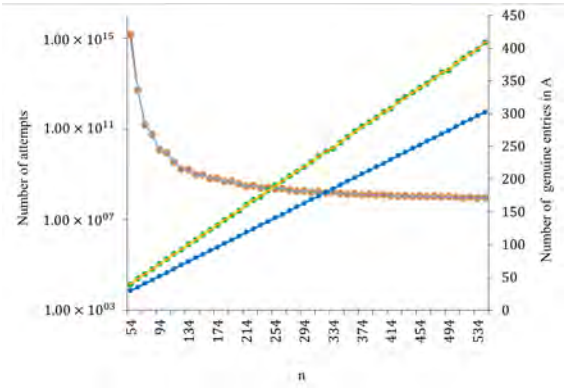


(e) t = 60

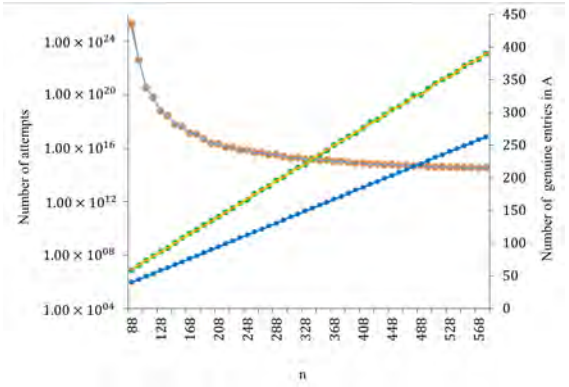
Figure 5-4: Results obtained for $p_{min} = 0.95 \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.



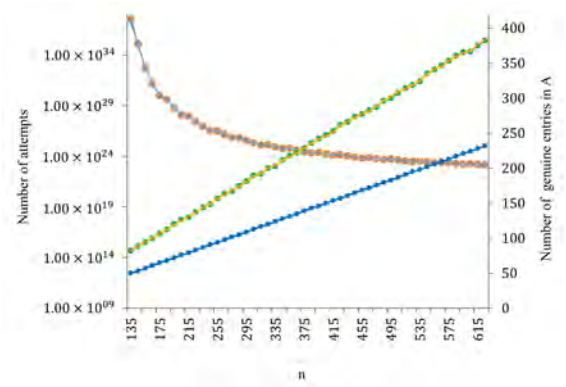
(a) t = 20



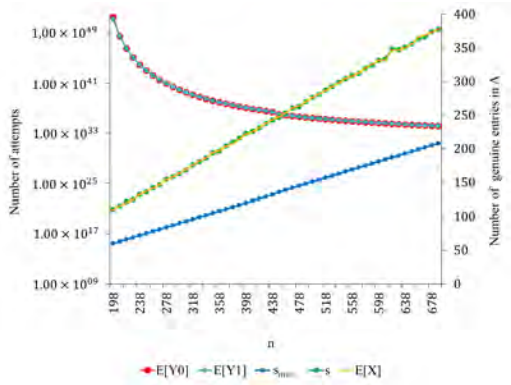
(b) t = 30



(c) t = 40

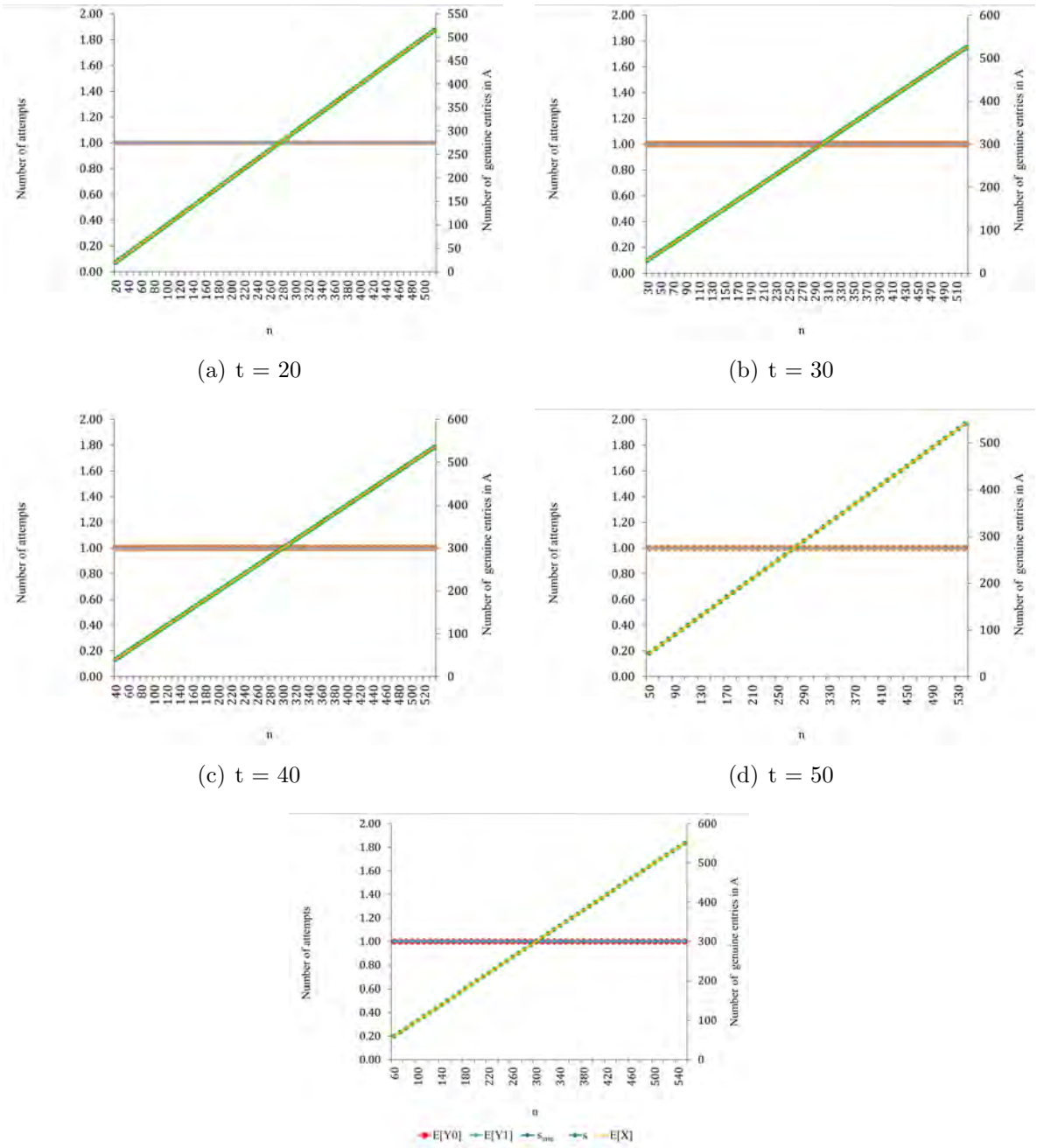


(d) t = 50



(e) t = 60

Figure 5-5: Results obtained for $p_{min} = 0.98 \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.



(a) t = 20

(b) t = 30

(c) t = 40

(d) t = 50

(e) t = 60

Figure 5-6: Results obtained for $p_{min} = 1 \in \mathcal{P}$, $t \in \mathcal{T}$, and proper values of n.

5.2.3 Computational Cost Analysis

In this section, the computation costs related to some critical steps of the protocol are analyzed. Let $\mathtt{C}_{GO}$ denote the cost of a group operation, and let $\mathtt{C}_{FA}$, $\mathtt{C}_{FM}$, and $\mathtt{C}_{FI}$ denote the costs of a field addition, field multiplication, and field inversion, respectively. $\mathtt{C}_{GE}$ denote the cost of an exponentiation in the group, i.e., a^m with $a \in \mathbb{G}$. Using a generic fast exponentiation algorithm, then $\mathtt{C}_{GE}$ will have a cost of, at most, $r \cdot \mathtt{C}_{GO}$, where $r = \lceil \log_2(m) \rceil$.

Computational Costs of Step 4

Once a participant collects all C_i, the participant needs to compute $A_k = \prod_{i=1}^{t-1} C_{i,k}$ for $k = 0, \ldots, 1, \ldots n$. Note that computing A_j has a cost of $(t-2)\mathtt{C}_{GO}$; hence, computing A has a cost of $(n+1)(t-2)\mathtt{C}_{GO}$.

Once the participant computes A, the mining process begins. Until completing the challenge, the participant will keep on selecting t distinct indices $1 \leq k_1, k_2, \ldots, k_t \leq n$ out of $\{1, 2, \ldots, n\}$, computing $w = (A_{k_1})^{\lambda_{k_1}} \cdot (A_{k_2})^{\lambda_{k_2}} \ldots \cdot (A_{k_t})^{\lambda_{k_t}}$ with $\lambda_{k_j} = \prod_{\substack{r=1 \\ r \neq j}}^{t} \frac{-k_r}{k_j - k_r}$ for $1 \leq j \leq t$, and checking whether $A_0 = w$.

So the cost of computing w, denoted as $\mathtt{C}_w$, is $\mathtt{C}_w = t \cdot \mathtt{C}_{GE} + (t-1) \cdot \mathtt{C}_{GO} + t(t-1)(2 \cdot \mathtt{C}_{FM} + \mathtt{C}_{FI})$. Therefore, the whole mining process cost is roughly $E_{Y_1}(\mathtt{C}_w + \delta)$ with δ being a constant.

5.2.4 Security Analysis

The goal of the adversary is to gain control over the creation of new blocks in the chain. To that end, he must generate solutions to the Proof-of-Works of the protocol and prove their validity to the other participants of the network. Since we assume the adversary is semi-honest, he follows the rules of the protocol but may want to learn as much as possible from the messages he receives from other parties to gain control over the creation of blocks in the chain.

First, note that the proposed protocol in steps 1–3 uses the joint random number generation protocol in parallel. At step 3 of the proposed protocol, what participant $1 \leq i \leq t-1$ does is to create $n+1$ ElGamal public keys $R_{i,k}, 0 \leq k \leq n$ from the other participant's public $g^{\kappa_{j,k}}$ for $1 \leq j \leq t-1$. Since the $R_{i,k}$ values are constructed from random Diffie-Hellman public-key and cannot be distinguished from random R's, the security of this protocol can be reduced to the decisional Diffie-Hellman problem on the group $\mathbb{G}$ [Hoogerwerf et al., 2021]. Furthermore, note that the private key

associated with the public key $R_{i,k}$ is given by $\sum_{j=1,j\neq i}^{t-1} \kappa_{j,k} \cdot s_i(j) \cdot \kappa_{i,k} \bmod q$ for each $k \in \{1, 2, \ldots, n\}$ and is unknown to any participant, even colluding corrupt parties.

At step 4 of the protocol, a participant, say i, will send back $\mathtt{C}_i$ with the corresponding signature σ_i to any requesting mining party. Note that each $\mathtt{C}_{i,k}$ is of the form $g^{\epsilon} R_{i,k}$ for some $\epsilon \in \mathbb{Z}_q$, i.e., $\mathtt{C}_{i,k}$ represents the ElGamal ciphertext of the message g^{ϵ} under the public key $R_{i,k}$.

Claim 1. A mining party can obtain a proper and fresh $A = [A_0, A_1, A_2, \ldots, A_n]$ if and only if the mining party has access to the corresponding $t-1$ $\mathtt{C}_i$ and computes

$$A = \left(\prod_{i=1}^{t-1} \mathtt{C}_{i,0}, \prod_{i=1}^{t-1} \mathtt{C}_{i,1}, \ldots, \prod_{i=1}^{t-1} \mathtt{C}_{i,n} \right).$$

Proof.

In a direction (only if), in section 5.2.1 proved it. In the other direction, if the requesting mining party has access to l_0 different $\mathtt{C}_{i,k}$ (possibly previous ones) with $l_0 \neq t-1$, then the computed $A_k = \prod_{i=1}^{l_0} \mathtt{C}_{i,k}$ will be of the form g^{β} for some random $\beta \in \mathbb{Z}_q$. Hence, the adversary only learns g^{β} and nothing else.

Claim 2. A mining party cannot reuse a $\mathtt{C}_i$ for future challenges.

Proof. Upon request to participant i at step 4, a mining party obtains $(\mathtt{C}_i, \sigma_i)$, where $\sigma_i \leftarrow \mathtt{sign}(\mathtt{sk}_i, \mathcal{H}_1(\mathtt{C}_i \parallel \mathtt{B}_l))$, with $\mathtt{B}_l$ being the last block. Hence, the mining party cannot reuse any $(\mathtt{C}_i, \sigma_i)$ as part of a solution to a future challenge. In particular, by calling the function `check`(), as shown by Algorithm 8, any verifier can discover any cheater.

Moreover, the mining process is fair. Let us now assume a mining party, possibly an attacker, constructs a proper and fresh A, then such a party may start the mining process. By construction, such a party does not know whether he will find t suitable $k_1, k_2, \ldots, k_t$ in such A, with

$$A_{k_1} = g^{P(k_1)}, A_{k_2} = g^{P(k_2)} \ldots, A_{k_t} = g^{P(k_t)}.$$

such that

$$A_0 = g^{\alpha} = g^{P(0)} = (A_{k_1})^{\lambda_{k_1}} \cdot (A_{k_2})^{\lambda_{k_2}} \ldots \cdot (A_{k_t})^{\lambda_{k_t}},$$

where

$$\lambda_{k_j} = \prod_{\substack{r=1 \\ r \neq j}}^{t} \frac{-k_r}{k_j - k_r} \text{ for } 1 \leq j \leq t.$$

If the mining party does not find suitable indices in A, the mining party may contact any of the $t-1$ participants to construct a proper and fresh A and start the mining process again. Furthermore, the party may request multiple C_i's from a participant i, and then pick only a C_i from the available ones for each i to obtain proper and fresh A's, and finally search suitable indices in each created A.

From the previous analysis, it is concluded that the proposed protocol can be regarded as secure and fair, assuming a semi-honest adversary.

5.3 Implementation

A proof-of-concept implementation of the proposed protocol was coded in the Python programming language. To simulate the protocol's behavior on a peer-to-peer network, the implementation of a decentralized peer-to-peer network by [Snoeren, 2021] was used. The protocol code was adjusted and some framework classes were modified for the correct implementation of the protocol and its peer-to-peer simulation. After making adjustments to the decentralized peer-to-peer network implementation, the logic of the proposed protocol was written in the following python classes:

- Generator class contains methods to create generators for $\mathbb{G}$.
- Zq class contains methods to carry out operations in $\mathbb{Z}_q$.
- MyRsa class contains methods for generating and validating RSA digital signatures.
- Participant class encloses the logic of the proposed protocol. It internally makes calls to P2P methods to send and receive messages over a P2P network.
- Protocol class contains the main method. It deals with instantiating the participants and calling functions for each protocol's phase.

The protocol code was published on GitHub [Aponte-Novoa and Villanueva-Polanco, 2022c] and can be executed in a Google Colab notebook [Aponte-Novoa and Villanueva-Polanco, 2022a].

5.4 Comparison with Other Approaches

Following a similar approach as in [Aponte et al., 2021a], a qualitative comparison between the proposed protocol and the following proof-based consensus protocols is carried out: Pure PoW [Nakamoto, 2008], Cuckoo hash function-based PoW [Pagh

and Rodler, 2001], Prime number finding-based PoW [Sunny, 2013], Double puzzles-based PoW [Eyal and Sirer, 2014], Non-outsourceable puzzles [Miller et al., 2015], Bitcoin-NG [Eyal et al., 2016], GHOST strategy [Sompolinsky and Zohar, 2013], Generalized PoW [Tang et al., 2017], Pure PoS (Nextcoin) [Nxt,], State of the block-based PoS [Nguyen and Kim, 2018a], PoS by coin flipping from many nodes [DBL, 2019], Delegated PoS [Del,], Coin age-based PoW difficulty re-designation (Ppcoin) [Sunny and Scott, 2012], Stake-based PoW difficulty re-designation (Blackcoin) [Vasin, 2014], Coin age with an exponential decay function [Ren, 2014], Combining PoW and PoS to append blocks sequentially [Duong et al., 2020], with difficulty adjustment [Chepurnoy et al., 2018], Proof of activity [Bentov et al., 2014], Puzzles designed for human PoW [Blocki and Zhou, 2016], Proof of burn [P4Titan, 2014], Proof of space [Park et al., 2018], Proof of elapsed time [saw,], Proof of luck [Milutinovic et al., 2016], Multichain [Greenspan et al., 2015] ; and the following Vote-based consensus protocols: Hyperledger with practical Byzantine fault tolerance [Hyp, b], Symbiont, R3 Corda with BFT-SMaRt [Sym, b, Cor,], Iroha with Sumeragi [iro, , Hyp, a], Ripple [Schwartz et al., 2014], Stellar [Mazieres, 2015], Quorum with Raft [Lamport, 2001], Chain [Fed,].

To perform a qualitative comparison, A set of characteristics that all these protocols shared are defined. Additionally, for each feature, a rank of values with their corresponding numeric values are defined. Table **5-2** shows the defined features with the values they can assume.

Table 5-2: Features and values used to cluster.

Feature	Description	Value
Energy efficiency	Efficient energy used to perform the tasks of the protocol	Yes = 0 No = 1
Modern hardware	Modern hardware requirements	No need = 0, Low need = 1, Need = 2, High = 3
Forking	Possibility to perform a forking of the main chain	Never = 0, Very difficult = 1, Difficult = 2, Probably = 3, Very Probably = 4, Never = 0
Double-spending attack	Possibility of a double-spending attack	Difficult = 1, More or less = 2, Easy = 3
Block creation speed	Speed to create a block	Very fast = 0, Fast = 1, Low = 2, Very low = 3
Mining Pool	Amenable to mining pool creation	Never = 0, Very difficult = 1, Can be prevented = 2, Difficult to prevent = 3, It occurs = 4
Number of participants	Number of participants in the protocol	Mostly unlimited = 0, Limited = 1
Decentralization	Decentralization of participants	Mostly high = 0, Low = 1
Trust	Trust of the network	More trustful = 0, Less trustful = 1
Node Identities	Node identity management	No = 0, Yes = 1
Security threat	Security threat to network	More serious = 0, Less serious = 1
Award-giving	Award-giving to miner nodes	Yes = 0, Mostly no = 1

Based on what is found in the literature, a value per feature per protocol is assigned, creating a dataset whose rows represent the names of protocols and columns represent the features. then hierarchical clustering over the dataset to group the protocols is applied; worked with the method of least variance (Ward's method); which seeks to obtain the least variability intra-cluster to ensure that each group is the most homogeneous possible. Finally the group is identified (and, hence, the features) in which the proposed protocol (PoAc) was located. Figure **5-7** shows the groups created by the clustering algorithm.

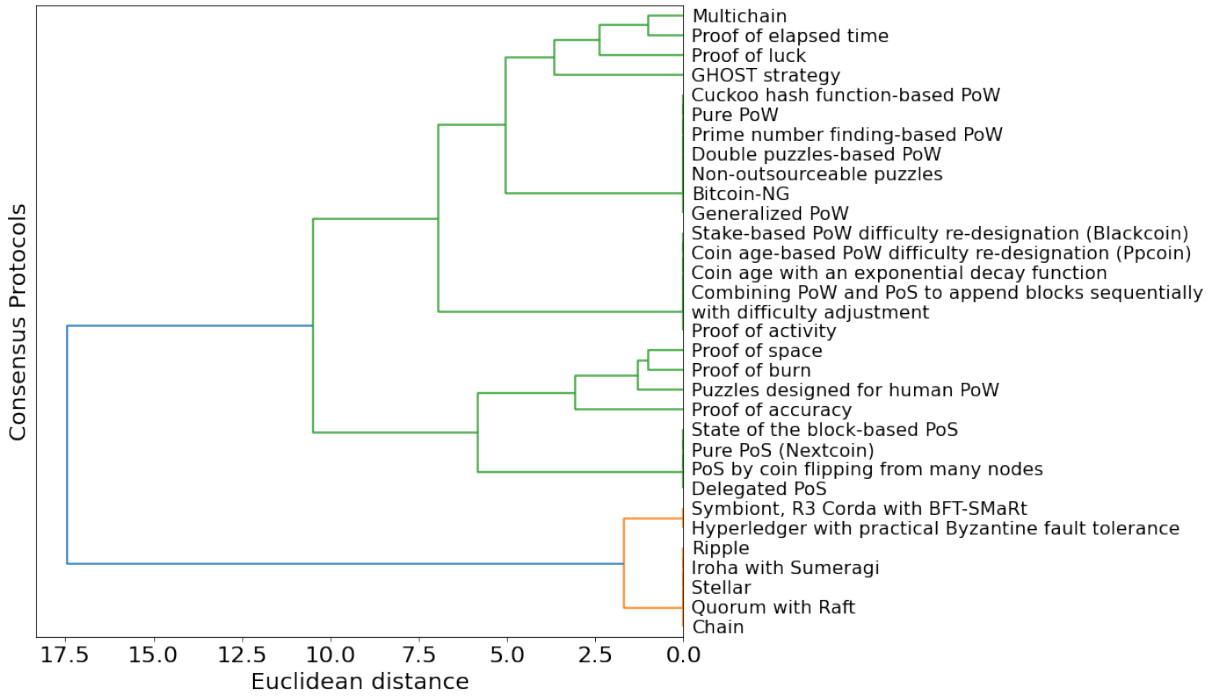


Figure 5-7: Hierarchical clustering dendrogram.

As a result of the hierarchical clustering, it is identified that PoAc is part of a group made up of Proof-of-Space, Proof-of-Burn, and puzzle designed for human PoW. Based on the previously defined features, it can be said that these protocols share the following features:

- Non-energy-efficient.
- Their need for modern hardware for its execution is low.
- They may present forking.
- Resistant to double-spending attack.
- Strategies may be deployed to prevent mining pools.
- High decentralization.
- Trustful
- Award-giving to a node adding a block to the chain.

It consider that the proposed protocol may be implemented and deployed in any blockchain, which decides the selection of the winning participant; that is, who wins the right to add a new block to the main chain, through a Proof-of-Work based protocol

(PoW). Additionally, the proposed protocol does not require the participants to have any particular hardware, making it implementable in blockchains where the hardware of the participants features any computing power.

5.5 Discussion of a potential application

A particular use case can be a decentralized application for registering anonymous touristic information where each participant in the network corresponds to a touristic operator. For this application, the integrity and availability of the data are crucial, and so there must be a control on registering data in the blockchain, making a consensus necessary. The touristic operators feature pieces of hardware that may not be homogeneous and typically present a low level of computation.

Additionally, this scenario may use its cryptocurrency for the consortium of touristic operators. A tourist can use it to carry out monetary transactions with the different operators, requiring trust between the participants. In addition to the touristic operators and their resources, tourists would participate in the network, even in the protocol, with mobile devices featuring varying computing power.

5.6 Conclusions

Although the study of alternative consensus protocols has taken an interest in the scientific blockchain community recently, most of the related works are still theoretical ideas. Proof-of-Accuracy protocols are a particular case since there are few papers about them in the literature, presenting neither concrete proof-of-accuracy protocols nor implementations.

This chapter introduced a new Proof-of-Accuracy protocol, starting with an initial and insecure design, which progressively improved regarding security. The proposed protocol removed the need for a coordinator and combined the proof of work component with access to random locations to improve the protocol's resistance to majority attacks. The analysis pointed out that it is secure and fair assuming a semi-honest adversary.

6 On Detecting Cryptojacking on Websites: Revisiting the Use of Classifiers

Cryptojacking is an illegal and unauthorized mining activity on the victim's computer, using the computational power of the victim's device to extract cryptocurrencies, which generates large computational consumption, reducing the computational efficiency of the victim's computer. Moreover, this attack may be used by a powerful attacker to increment their computationally power, posing a risk to any blockchain based on mining [Carlin et al., 2020, Aponte et al., 2021a, Tayyab et al., 2022, Wu et al., 2022, Bijmans et al., 2019, Aponte-Novoa et al., 2021, Aponte-Novoa and Villanueva-Polanco, 2022b].

Cryptojacking on websites uses JavaScript code to mine cryptocurrencies. This technique does not require installing javascript code to perform the mining process. All it takes is for the user to load the infected website in their browser for the illegal mining code to execute in the browser of the victim's computer [Cry, a]. According to [Cry, b], many websites have been infected by cryptojacking, such as personal blogs up to Alexa-ranking websites. Also, it noted that as of January 2022, there were around 3000 sites websites that offered online cryptojacking scripts.

Cryptojacking detection techniques such as browser extensions and antiviruses provide a partial solution to the cryptojacking problem since attackers can avoid them by employing obfuscation techniques or renewing domains or malicious scripts relatively frequently. [Tekiner et al., 2021]. Cryptojacking boomed with the birth of service providers that offer ready-to-use implementations of mining scripts in web browsers. Therefore, attackers can reach many more victims via websites. These service providers are Coinhive [Coi,] and Cryptoloot [Cry, c].

As seen in [Ying et al., 2022, Hernandez-Suarez et al., 2022, Naseem et al., 2021, Om Kumar and Sathia Bhama, 2019, Liu et al., 2018, Sivaraju, 2022, Petrov et al., 2020], there is a recent trend of applying specialized deep learning models to detect cryptojacking in websites. However, specialized deep learning models might pose challenges with ref-

erence to their deployment and performance. So this chapter seeks to explore machine learning models (comparatively simpler than deep learning models) for cryptojacking classification to identify which of these machine learning models may render similar or better results. Hence, this chapter takes previous works as a reference, particularly the recent specialized deep learning model and the dataset presented and collected in [Hernandez-Suarez et al., 2022]. This dataset was used to train and validate multiple machine learning classification algorithms to detect cryptojacking on websites and make a comparison between the machine learning models and the specialized deep learning model.

This chapter is organized as follows: Section 6.1.1, presents three different cases of cryptojacking. Section 6.2 presents different techniques for the detection of cryptojacking, Section 6.3 describes the dataset's selection and content, and presents an exploratory data analysis. This section shows a correlation and clustering of data, and also describes feature selection and split and normalization process. Some machine learning models for detecting cryptojacking on the website are presented in the section 6.3.3. Next, Section 6.4 shows the results, and the Section 6.5 an idea of implementing a hybrid, lightweight, usable, privacy-preserving mechanism added to a web browser for blocking websites that potentially may be infected by cryptojacking is presented. Then, some conclusions are given.

6.1 Some cases of cryptojacking

6.1.1 Cryptojacking on websites

Websites of different kinds have been victims of strong cryptojacking type malicious software actions. In early 2018, the online video-sharing platform YouTube was illegally compromised where the CoinHive miner ran on its ads [You, , kas,], and the Russian Nuclear Weapon Research Center [Rus,]

6.1.2 Cryptojacking with advanced techniques

Other attacks have used advanced techniques in the spread of cryptojacking; an example of this was when the botnet called Vollgar attacked all Microsoft SQL (MSSQL) databases servers to take control of administrative accounts and inject malicious miners into those servers [MSS,]. Another example of the use of these advanced techniques was presented with the Zoom video conferencing software. In this case, the attackers merged cryptojacking malware with the main zoom application and published it on different file-sharing platforms [Zoo,]. Similarly, this technique was used in the Nintendo

Switch consoles [nin,]. Another use case of these advanced techniques was presented in MikroTik routers between July and August of 2018, where a cryptojacking campaign managed to compromise more than 200,000 MikroTik routers, primarily located in Brazil. In the same way, researchers observed that routers that were not MikroTik were also compromised [tre,]. Moreover, a recent study explores cryptojacking and its impact on Electric Vehicles [Malik and Anwar, 2022].

In 2019, 8 applications were detected and removed from the Microsoft Store in Windows 10. When the user installed and opened these applications, they secretly downloaded cryptojacking's JavaScript code, which carried out mining tasks for the Monero cryptocurrency, notably affecting the user device performance [Mic, ,Sym, a].

In 2018, researchers from RedLock dedicated to computer security discovered that at least one unknown computer criminal broke into an Amazon Cloud account associated with Tesla and used it to mine cryptocurrencies. In this attack, the Stratum mining protocol was used, and the true IP address of the mining pool was hidden and kept the CPU consumption low [Tes,].

6.1.3 Cryptojacking in industrial control systems or Critical Servers

The impact of cryptojacking has surpassed the borders of traditional websites, affecting industrial control systems and critical servers, In January 2020, following a report on the bug bounty website www.hackerone.com, the US Department of Defense discovered that its government and military servers were affected by cryptojacking to mine the currency Monero illegally [ZDN, ,Dec, a]. In 2019, a Russian nuclear warhead facility employee was fined around $7000 for illegally mining bitcoin using the facility's servers [Dec, b]

6.2 Related works

The literature presents different techniques for the detection of cryptojacking on websites; however, these solutions present some limitations related to performance, transparency and effectiveness.

A hardware-based approach to cryptojacking detection is presented in [Ying et al., 2022]. This method takes advantage of the Intel Processor Trace mechanism to collect control flow information at runtime from the web browser. This technique uses two optimization approaches based on library functionality and information gain to preprocess control flow information. It also takes advantage of a Recurrent Neural Network (RNN) for cryptojacking detection.

A method that performs a fingerprinting technique to detect possible malicious sites is presented in [Hernandez-Suarez et al., 2022], which is then characterized with an autoencoding algorithm that keeps the best information of the infection vestiges to maximize the classification power by means of a deep dense neural network.

A lightweight cryptojacking detection system that uses deep learning techniques to accurately detect the presence of unwarranted mining activity based on emerging WebAssembly (Wasm) based cryptojacking malware in real-time is introduced at [Naseem et al., 2021]. This system employs an image-based classification technique to distinguish between benign web pages and those using Wasm. Specifically, the classifier implements a convolutional neural network (CNN) model.

A detection and control method for IoT botnets is presented in [Om Kumar and Sathia Bhama, 2019], which uses a deep learning model and cryptojacking activities carried out by the bot. This method performs malicious attack detection by implementing a sparse autoencoder composed of an input layer, a hidden layer, and an output layer.

A method for detecting silent browser mining behavior is presented in [Liu et al., 2018]. This method drives known malicious mining samples, extracts heap snapshots and stack code functions of a dynamically running browser, and performs automatic detection based on a recurrent neural network (RNN).

A method called CapJack to identify illicit bitcoin mining activity in a web browser using cutting-edge CapsNet technology is presented in [Sivaraju, 2022]. Deep learning framework CapsNet employs heuristics based on system behavior to detect malware effectively.

A detection method called CoinPolice is presented in [Petrov et al., 2020]; that method flips throttling against cryptojackers, artificially varying the browser's CPU power to observe the presence of throttling. Based on a deep neural network classifier, CoinPolice can detect hidden miners

6.3 Methodology

This section, describes the methodology whose phases are shown in Fig. **6-1**.

1. Selection of the dataset.
2. Exploratory data analysis. In this phase, a correlation matrix is calculated and perform clustering of the dataset. Additionally, the feature selection process, dataset normalization, and dataset splitting are performed.

Figure 6-1: Methodology

3. Exploration of Classification Models. In this phase, the Sparse Autoencoder + Deep Dense Neural Network model for Cryptojacking detection [Hernandez-Suarez et al., 2022] is reproduced. Additionally, different classifiers for Cryptojacking detection were trained and tested.

4. Results Presentation. The results of applying K-means clustering to the dataset, as well as the results of cross-validation and evaluation of the different classifier models are presented.

The development of these phases was carried out in a notebook in the Google Colaboratory (Google Colab) environment using the Python programming language and libraries such as NumPy, Pandas, Seaborn, MatPlotLib, TensorFlow, Keras and scikit-learn. The interested reader can see the source code here [Aponte-Novoa et al., 2022a]. The different classification models were executed in the virtual machine offered by Google Colab, which has the following characteristics: 2.20GHz Intel Xeon processor, 12G of RAM, and Ubuntu 18.04.6 LTS Operating system.

6.3.1 Selection of Dataset

The dataset that was collected in [Hernandez-Suarez et al., 2022] is used, which contains records of host-based and network-based features associated with websites. Also, each record is labeled whether a website is infected or not by cryptojacking. According to [Hernandez-Suarez et al., 2022], the procedure to capture these samples includes three layers: the first of these is called fingerprinting, in which the sites that may contain signs of cryptomining are captured. After this, data based on the network is captured, which includes the information flows of the traffic that transits through the HTTP, HTTPS, and TCP protocols and finally the host-based data is captured, which consists of the tracking of the website once that interacts with the browser. The dataset used in this chapter comprises 9292 benign sites and 3434 sites labeled and validated as cryptojacking infected. The host-based and network-based features that compose the dataset are indicated in Table **6-1**. In addition to these features, the dataset presents an attribute called Label, representing the class or classification of the sample where 1 indicates a sample of a site labeled as cryptojacking infected and 0 a benign site.

Table 6-1: Host-based and Network-based Features [Hernandez-Suarez et al., 2022]

Host-based	
Feature	**Description**
C1	Time in C1 is a subset of the total processor idle time
C2	Time at C2 is a state of lower energy and higher output latency than Time at C1
C3	Time at C3 is a lower energy state and higher output latency than Time at C2
I/O Data Operations	Speed at which the process is issuing read and write I/O operations
I/O Data Bytes	Speed at which the process is reading and writing bytes in I/O operations
Number of subprocesses	Number of sub-processes that are currently active in a parent process
Time on processor	The total time, in seconds, that a process has been running
Disk Reading/sec	Speed of disk reading operations
Disc Writing/sec	Speed of writing operations to disk
Confirmed byte radius	The ratio of Memory/Bytes committed and Memory/Confirmation limit
Percentage of processor usage	Elapsed time on the processor when an active thread is running
Pages Read/sec	Speed rate at which the disk was read in order to resolve hard page errors
Pages Input/sec	Speed at which pages are written to disk to free up space in physical memory
Page Errors/sec	This is the average number of pages with faults per second
Network-based	
Feature	**Description**
Bytes Sent	The rate at which bytes leave the browser's HTTP requests
Received Bytes (HTTP)	Speed of bytes arriving to the browser's HTTP responses
Network packets sent	Speed of sending packets in the TCP protocol
Network packets received	Packet reception speed over the TCP protocol

6.3.2 Exploratory data analysis

This section shows a correlation and clustering of the data, and also describes feature selection and split and normalization process of the dataset.

Correlation

It was verified that the dataset does not present missing cells and duplicate rows. As expected, the dataset does not present these cases thanks to the fact that after its collection, the collectors went through a pre-processing process. To identify the existing correlation between the different features that make up the dataset and the classification class, a correlation matrix is built by employing the method `corr` of `pandas.DataFrame` with the default parameters (method='pearson', min_periods=1, numeric_only=_NoDefault.no_default) (see Fig **6-2**).

This matrix tells us that the feature Percentage of processor usage is the one that

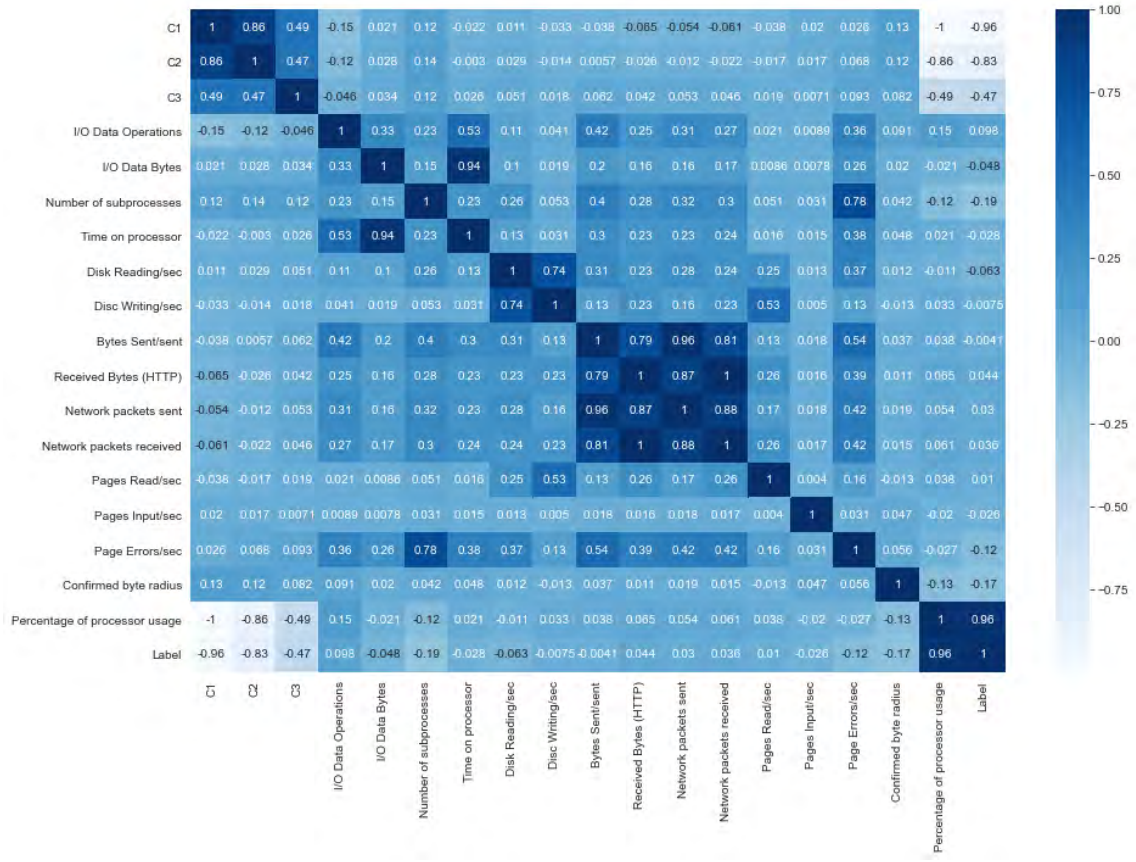


Figure 6-2: Feature Correlation Matrix

presents the highest positive correlation with the classification of the website (label). On the other hand, the features C1, C2, C3, Number of subprocesses, confirmed byte radius and Page Errors/sec have, in their order, the highest negative correlation with website ranking.

Clustering

Due to the essence of the dataset, it is known that its records are divided into two categories: the first is the records of sites infected with cryptojacking (labeled with 1) and the records of benign sites (labeled with 0). To verify and analyze how well these records are grouped, according to their features, the unsupervised learning algorithm K-Means Clustering is used, which groups the unlabeled dataset into K different clusters. First, the elbow method to determine the number of clusters in the dataset is used. Fig. **6-3** shows the elbow of the curve at three, but for the reason explained previously, two is given as the number of clusters parameter to the K-Means algorithm.

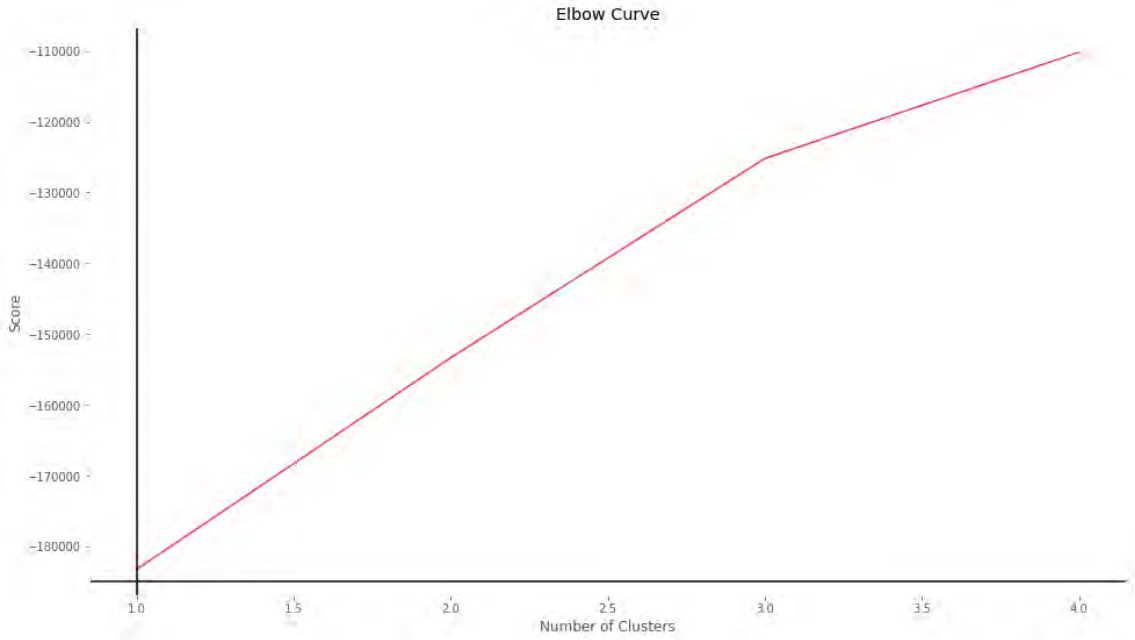


Figure 6-3: Elbow Curve

Next, the elements on the dataset were grouped using the K-Means algorithm with the parameters $n\_clusters = 2$, $random\_state = 0$ and $init = "k - means + +"$. "k-means++" is a method that chooses the first centroid to the location of a randomly selected data point and then chooses subsequent centroids of the remaining data points, based on a probability proportional to the square of the distance from the nearest existing centroid of a given point. It is helpful to choose the centroids to be as far away as possible from each other, to try to cover the data space occupied since the initialization [Arthur and Vassilvitskii, 2007].

The K-Means algorithm was applied to all the dataset entries and generated a figure to visualize the group assignment for each sample of the dataset group, only taking C1, C2, and C3 features. Fig. **6-4** shows that the application of clustering to the dataset results in two well-defined groups corresponding to the samples of benign sites and infected websites.

Feature Selection

Extracting relevant features from the raw data is paramount for Intrusion Detection System (IDS) classification [Prashanth et al., 2022]. For feature selection, first, the features with a p-value less than or equal to 0.05 were select and later statistical methods with the f_classif function and the univariate filtering function "Anova Scores" for 10

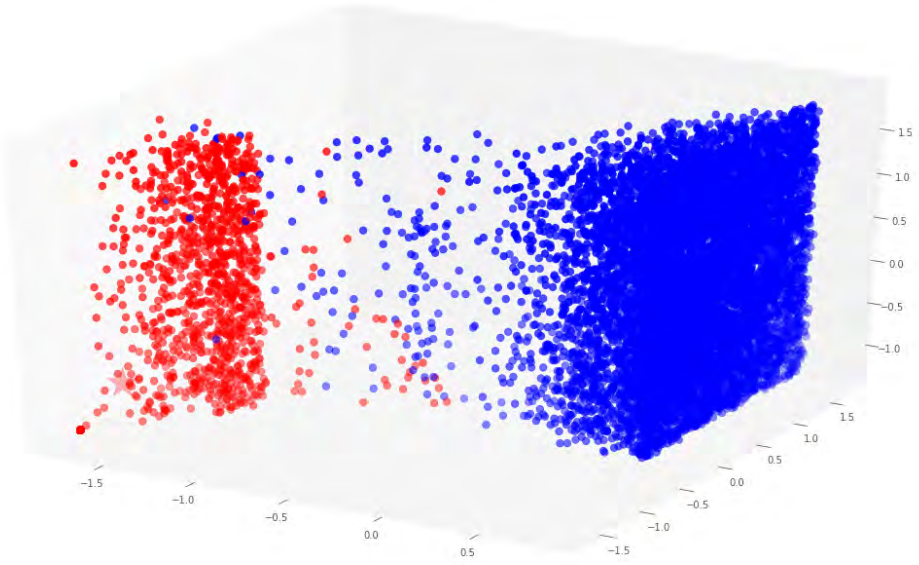


Figure 6-4: Clustering of dataset entries

Table 6-2: Parameters used in the feature selection methods

Method	Paramethers
p-value	*p*-value ¡= 0.05
Anova scores	n_splits=5, n_repeats=5, random_state=12345, score_func = f_classif, k= 5/10
RFECV wrapper	estimator=LogisticRegression, step=1, cv= StratifiedKFold(n_splits=5, random_state=12345, shuffle=True), min_features_to_select=1, scoring='f1', verbose=2, n_jobs=−1

and 4 features was applied. "Anova Score" fits a simple linear model between a single feature and the outcome, then the *p*-value for the whole model Ftest is returned [car,]. In addition, an RFECV wrapper method was applied, which is a feature ranking with recursive feature elimination, and cross-validated selection of the best number of features [RFE,]. The parameters used in the feature selection are in Table **6-2**. For each of these filters, a subset of data was selected to be used in the cross-validation.

Split and Normalization of the dataset

After exploring the dataset, the features are separated in a X set and the label category in a Y set. The records of the dataset are split for training (80% of them) and testing (20% of them). For this partition, the parameters $random\_state = 42$ and $stratify = y$ are used. The stratify parameter makes a split so that the proportion of values in the sample produced will be the same as the proportion of values provided to parameter

stratify. Because the ranges for the different data features are very different, the Standard Scaler to the training and testing datasets is applied for the original dataset and the other datasets created in the Feature Selection step.

6.3.3 Exploration of Classification Models

Having the dataset, a subset of training data and a subset of test data are prepared. Also, normalization to the data is applied. With the normalized data, the Sparse AutoEncoder (SAE) + Deep Dense Neural Network (DDNN) model is replicated, as proposed in [Hernandez-Suarez et al., 2022], to have a point of comparison for the proposed models. Later some techniques are applied for the selection of features. After this, 6 reference models are defined for classification, such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, K-Nearest Neighbor, and XGBoost. With these reference models, the different datasets obtained in the selection of features are tested, performing cross-validation for each of these sets. Finally, the performance of the classification models is measured using precision, recall, and F1-score.

Sparse Autoencoder + Deep dense Neural Network

According to the results shown in [Hernandez-Suarez et al., 2022], the Sparse AutoEncoder + Deep Dense Neural Network model is an state-of-the-art model for cryptojacking detection. In [Hernandez-Suarez et al., 2022] the reader can see the comparison between this model and other proposed models. In this chapter, this model is reproduced to validate the results against the proposed methods, using the mentioned dataset.

Classification Models

This section considers the following classification models.

- Logistic Regression or LR is a standard probabilistic statistical classification model widely used in different disciplines, such as computer vision, and marketing, among others [Feng et al., 2014].
- A Decision Tree is a hierarchical structure built using a data set's features (independent variables). In a Decision Tree, each node is partitioned according to a measure associated with a subset of features [Suthaharan, 2016]. This algorithm repeatedly divides the data set according to a criterion that seeks to maximize the separation of the data, resulting in a tree-like structure [Breiman et al., 2017].

Table 6-3: Models parameters

Model	Parameters
Logistic regression	default
Decision Tree	criterion= 'gini', max_depth=5, min_samples_leaf=20, random_state=12345
Random Forest	n_estimators=25
Gradient Boosting	n_estimators=20, learning_rate=0.5, max_features=2, max_depth=2, random_state=0
k-Nearest Neighbor	n_neighbors = 36, p = 1 (manhattan_distance)
XGBoost	max_bin=255

- A Random Forest is a collection of decision trees associated with a set of bootstrap samples, generated from the original data set. The nodes are partitioned based on the entropy or Gini index of a selected subset of the features [Suthaharan, 2016].
- Gradient Boosting is a widely used machine learning algorithm due to its efficiency, accuracy, and interoperability [Friedman, 2001]. This algorithm achieves state-of-the-art performance in many machine learning tasks, such as multi-class classification [Li, 2012], click prediction [Richardson et al., 2007] and learning to rank [Burges, 2010].
- K-Nearest Neighbor classifier, unlike other methods, uses the data directly for classification, without first building a model [Dasarathy, 1991,Ripley, 2007]. One of the advantages of the K-nearest neighbors algorithm over other algorithms is the fact that the neighbors can provide an explanation for the classification result [Dreiseitl and Ohno-Machado, 2002].
- XGBoost is a scalable ensemble machine learning and gradient boosting technique focusing on performance and speed. This technique allows for solving problems of ranking, classification, and regression. [Omer and Shareef, 2022]

The classification models are instantiate and trained. Table **6-3** shows the parameters used in these algorithms. With the different models, cross-validation is performed with the different datasets, and the mean and standard deviation of the accuracy are calculated.

6.4 Results

6.4.1 K-Means clustering

Once the clustering model has been trained, the dataset is taken and look for which element is the closest to the centroid of each of the two groups; these elements will be

Table 6-4: Values of the features of representatives of clusters

Feature	Cluster 1 (y=0)	Cluster 2 (y=1)
C1	82.2145	8.2148
C2	73.1083	25.4516
C3	3.9767	6.4803
I/O Data Operations	32.7094	18.8409
I/O Data Bytes	121124.1414	39403.4743
Number of subprocesses	30	31
Time on processor	0.4967	0.1733
Disk Reading/sec	16.6267	4.9044
Disc Writing/sec	1.3781	0
Bytes Sent/sent	915.6518	210.0906
Received Bytes (HTTP)	12258.8161	3844.3897
Network packets sent	6.1794	1.6865
Network packets received	10.1138	3.6172
Pages Read/sec	0.6446	0.0665
Pages Input/sec	0	0
Page Errors/sec	13351.0499	1633.1676
Confirmed byte radius	28.9634	27.1788
Percentage of processor usage	17.7854	91.7852

taken as the representatives of each of the groups. For each of these representatives, the values of your features and their category are verified (Table. **6-4**). It can be seen that the category for each representative is different (benign/infected), and the values of some features are distant from each other.

With the clustering model created and trained, it is applied to the test dataset, obtaining the two clusters. With the help of the representative elements of each cluster, the clusters corresponds to the benign sites and which to the infected ones are built. With this information, the cluster assigned to each element of the test dataset and the set of labels (y) corresponding to the test dataset, a classification report is calculated, where it is assumed that the cluster assignment to each sample of the dataset corresponds to classification between benign site and infected site. In this report, it obtains an accuracy = 0.9902 and precision = 0.988201.

6.4.2 Feature Selection

Table **6-5** presents the results of applying the methods for selecting features. It shows the method used and its selected features. 15 and 12 features are obtained with p-value and "RFECV wrapper" methods respectively.

Table 6-5: Selected features

Method	Selected features
***p*-value**	C1, C2, C3,I/O Data Operations, I/O Data Bytes, Number of subprocesses, Time on processor, Disk Reading/sec, Received Bytes (HTTP), Network packets sent, Network packets received,Pages Input/sec, Page Errors/sec, Confirmed byte radius, Percentage of processor usage
anova scores 4 features	Percentage of processor usage, C1, C2, C3
anova scores 10 features	Percentage of processor usage, C1, C2, C3, Number of subprocesses, Confirmed byte radius, Page Errors/sec, I/O Data Operations,Disk Reading/sec , I/O Data Bytes
RFECV wrapper	Percentage of processor usage, Network packets sent, I/O Data Bytes, I/O Data Operations, Received Bytes (HTTP), Disc Writing/sec, Network packets received , C2, Bytes Sent/sent , Confirmed byte radius, Time on processor, C1

6.4.3 Cross-Validation

For the cross-validation and evaluating the models, the datasets are described in Table **6-6**. The results of this process is shown in Table **6-7** and Table **6-8**

The results of the cross-validation of the different classification models with each of the datasets are presented in Table **6-7**

Table 6-6: Datasets Description

Dataset Name	Description
All features dataset	complete dataset
15 features dataset	dataset of 15 features selected with *p*-value
4 features dataset	dataset of 4 features selected with anova Scores method
10 features dataset	dataset of 10 features selected with anova Scores method
12 features dataset	dataset of 12 features selected with RFECV wrapper method

Table 6-7: Cross-Validation models

Logistic regression		
dataset	**mean accuracy**	**standard deviation accuracy**
All features dataset	0.9917	0.0020
15 features dataset	0.9911	0.0020
4 features dataset	0.9911	0.0023
10 features dataset	0.9912	0.0021
12 features dataset	0.9919	0.0018
Decision Tree		
All features dataset	0.9914	0.0018
15 features dataset	0.9921	0.0022
4 features dataset	0.991	0.0018
10 features dataset	0.991	0.0018
12 features dataset	0.9915	0.0020
Random Forest		
All features dataset	0.9944	0.0015
15 features dataset	0.9944	0.0014
4 features dataset	0.9906	0.0017
10 features dataset	0.9943	0.0015
12 features dataset	0.9947	0.0013
Gradient Boosting		
All features dataset	0.9926	0.0018
15 features dataset	0.9919	0.0019
4 features dataset	0.9905	0.0022
10 features dataset	0.9919	0.0020
12 features dataset	0.9932	0.0016
k-Nearest Neighbor		
All features dataset	0.9878	0.0018
15 features dataset	0.9878	0.0020
4 features dataset	0.9913	0.0021
10 features dataset	0.9891	0.0019
12 features dataset	0.9889	0.0019
XGBoost		
All features dataset	0.9947	0.0016
15 features dataset	0.9946	0.0016
4 features dataset	0.9908	0.0018
10 features dataset	0.9942	0.0017
12 features dataset	0.9950	0.0017

6.4.4 Model Selection and Evaluation

Table **6-8** shows the main metrics obtained with the classification models.

Table 6-8: Results of evaluating the models

SAE + DDNN [Hernandez-Suarez et al., 2022]							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9893	0.9941	0.9911	0.9926	0.9823	0.9712	0.9891
Logistic regression							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9937	0.9957	0.9957	0.9957	0.9884	0.9884	0.9884
15 features dataset	0.9937	0.9957	0.9957	0.9957	0.9884	0.9884	0.9884
4 features dataset	0.9918	0.9946	0.9941	0.9944	0.984	0.9854	0.9847
10 features dataset	0.9925	0.9957	0.9941	0.9949	0.9841	0.9884	0.9862
12 features dataset	0.9941	0.9957	0.9962	0.996	0.9898	0.884	0.9891
Decision Tree							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9902	0.9904	0.9962	0.9933	0.9896	0.9738	0.9817
15 features dataset	0.9918	0.993	0.9957	0.9944	0.9883	0.9811	0.9847
4 features dataset	0.9906	0.9925	0.9946	0.9936	0.9854	0.9796	0.9825
10 features dataset	0.9906	0.9946	0.9925	0.9935	0.9797	0.9854	0.9826
12 features dataset	0.9918	0.993	0.9957	0.9944	0.9883	0.9811	0.9847
Random Forest							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9957	0.9957	0.9984	0.997	0.9956	0.9884	0.992
15 features dataset	0.9953	0.9962	0.9973	0.9968	0.9927	0.9898	0.9913
4 features dataset	0.9906	0.9925	0.9946	0.9936	0.9854	0.9796	0.9825
10 features dataset	0.9949	0.9952	0.9978	0.9965	0.9941	0.9869	0.9905
12 features dataset	0.9949	0.9952	0.9978	0.9965	0.9941	0.9869	0.9905
Gradient Boosting							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9937	0.9957	0.9957	0.9957	0.9884	0.9884	0.9884
15 features dataset	0.9929	0.9952	0.9952	0.9952	0.9869	0.9869	0.9869
4 features dataset	0.991	0.9935	0.9941	0.9938	0.984	0.9825	0.9832
10 features dataset	0.9925	0.9946	0.9952	0.9949	0.9869	0.9854	0.9862
12 features dataset	0.9929	0.9946	0.9957	0.9952	0.9883	0.9854	0.9869
k-Nearest Neighbor							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9906	0.9904	0.9968	0.9936	0.9911	0.9738	0.9824
15 features dataset	0.9906	0.9909	0.9962	0.9936	0.9897	0.9753	0.9824
4 features dataset	0.9921	0.9946	0.9946	0.9946	0.9854	0.9854	0.9854
10 features dataset	0.9894	0.9909	0.9946	0.9928	0.9853	0.9753	0.9802
12 features dataset	0.9902	0.992	0.9946	0.9933	0.9853	0.9782	0.9817
XGBoost							
		benign			malign		
Dataset	Accuracy	Precision	Recall	F1-score	Precision	Recall	F1-score
All features dataset	0.9965	0.9973	0.9978	0.9976	0.9942	0.9927	0.9934
15 features dataset	0.9965	0.9973	0.9978	0.9976	0.9942	0.9927	0.9934
4 features dataset	0.9910	0.9925	0.9952	0.9938	0.9868	0.9796	0.9832
10 features dataset	0.9965	0.9973	0.9978	0.9976	0.9942	0.9927	0.9934
12 features dataset	0.9965	0.9973	0.9978	0.9976	0.9942	0.9927	0.9934

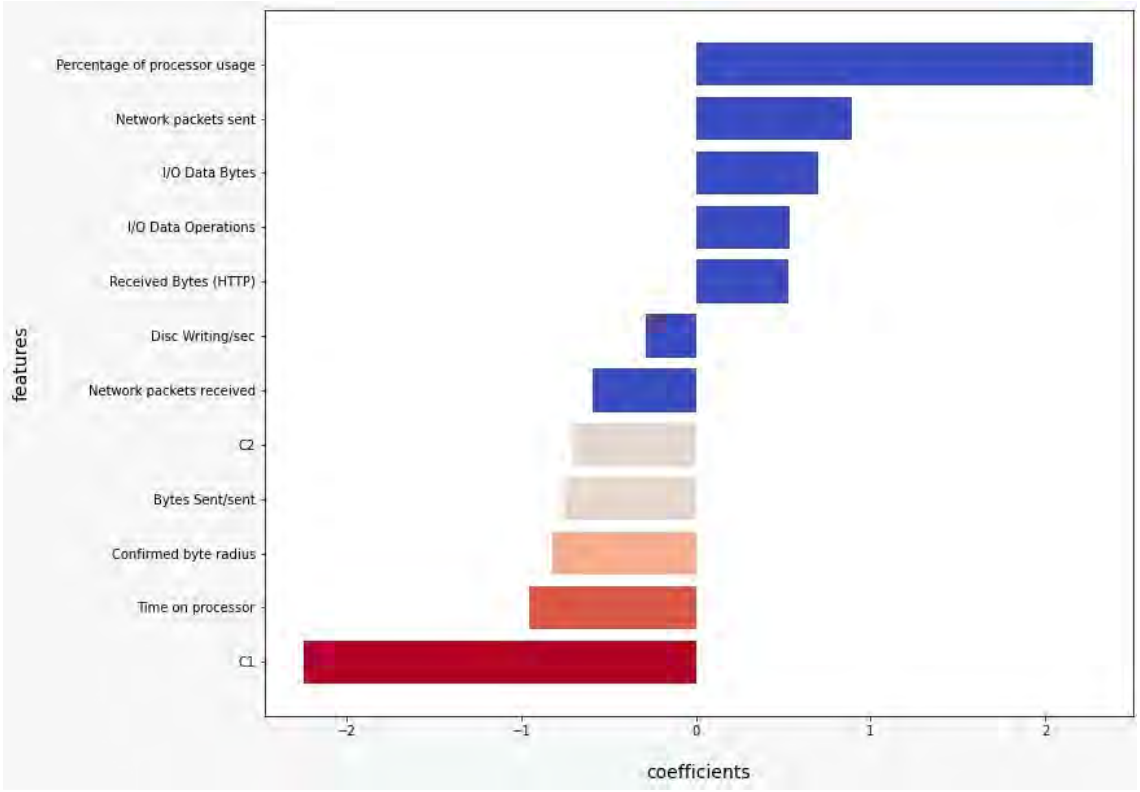


Figure 6-5: Feature Importance logistic regression Model

The simple model Logistic Regression and the advanced model XGBoost are used as a reference, and to search for the features with the most importance in the classification process. As seen in Fig. **6-5**, positive coefficients indicate that the event (malign) is more likely at that level of the predictor than at the reference level. Negative coefficients indicate that the event (malign) is less likely at that level of the predictor than at the reference level. It can be seen that the features "Percentage of processor usage" or "Network packets sent" indicate that the event is more likely, while the variables C1 or "Confirmed byte radius" suggest that the event is much less likely. As seen in Fig. **6-6**, it can be noticed that the variables C1, C2, "Percentage of processor usage", and "Disc Writing/sec" are the most important in the classification process of XGBoost model.

6.5 Integration Scenario

Taking the ideas described in [Cozza et al., 2020,Guarino et al., 2022,Ikram et al., 2016], the feasibility of a hybrid, lightweight, usable, privacy-preserving mechanism added to a web browser for blocking websites that potentially may be infected by Cryptojacking

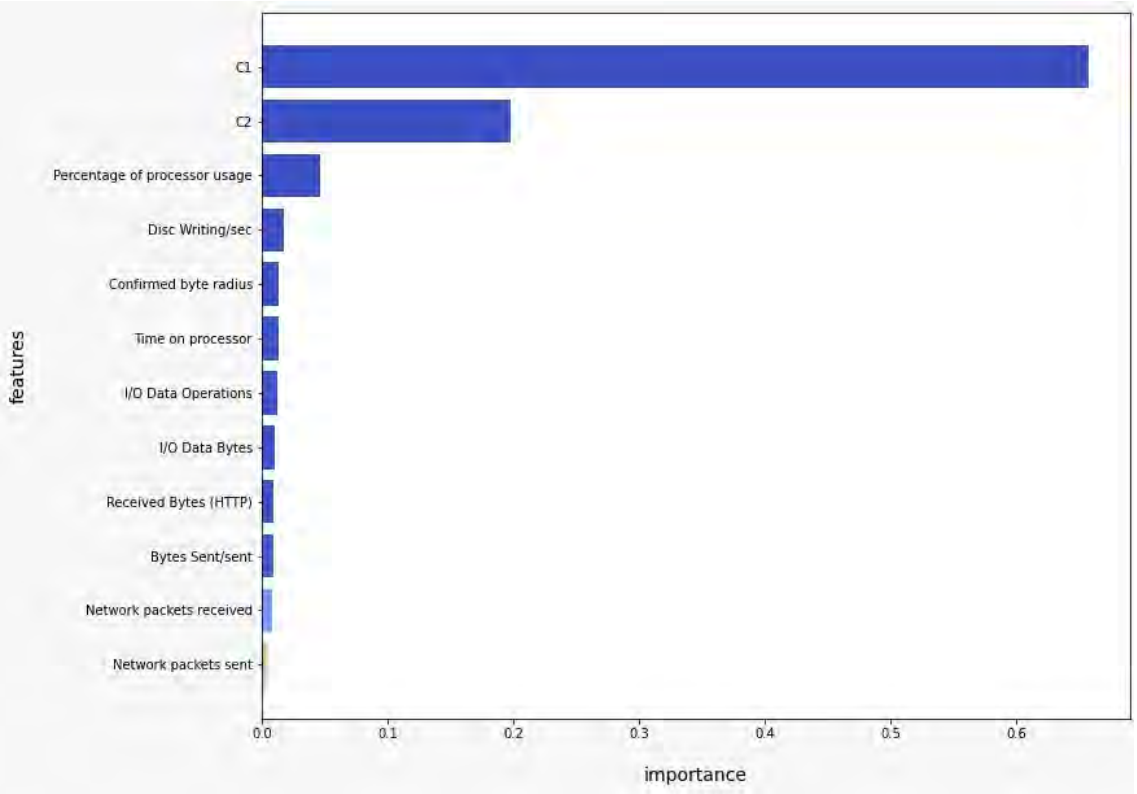


Figure 6-6: Feature Importance XGBoost Model

is clear. The envisioned approach exploits the blacklisting technique, widely used in this field, and a machine learning classifier to classify websites as benign or malign. The output from a classifier allows for updating the blacklists used to filter/block blacklisted websites. Additionally, this approach can be enhanced by introducing an ML-Based model to detect JavaScript malicious code inserted in websites or content shared with the user. To improve the usability of the mechanism, live alerts may be generated for the users for providing them a comprehensive awareness and full control of potential cryptojacking threats.

6.6 Conclusions

In this chapter, six Machine Learning models are explored for detecting cryptojacking on websites. The exploration started with a simple model as Logistic Regression, and then moved to more advanced algorithms in terms of tabular data classification, such as XGBoost, Decision Trees, Random Forest, Gradient Boosting, and K-Nearest Neighbor models. Furthermore, various feature selection methods were used, such as those based

on statistical methods, e.g. Test Anova, and other methods called Wrappers, in order not only to reduce the complexity of the built models but also to know the features with greater predictive power.

From the results, these conclusions could be drawn:

1. With 12 of the 18 features obtained with the RFECV method, an accuracy similar to that of other works [Hernandez-Suarez et al., 2022, Liu et al., 2018, Naseem et al., 2021, Om Kumar and Sathia Bhama, 2019, Gomes and Correia, 2020] based on Deep Learning techniques was reached. Even, as observed in Table **6-7**, with a dataset of only 4 features, an accuracy of 99.11% was obtained using Logistic Regression, and an accuracy of 99.13 was obtained with k-Nearest Neighbor.

2. The most relevant features in the case of Logistic Regression were C1, "Percentage of processor usage", "I/O Data Bytes" and, "Network packets sent", while the most important features in the case of XGBoost were "Percentage of processor usage", "Network packets sent", "Time on processor" and C1.

It is concluded that by using simpler models such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, and K-Nearest Neighbor models, a ML-based classification component can be built, with a success rate similar to or greater than that of advanced algorithms such as XGBoost and even those of other works based on Deep Learning. Additionally, the simplicity of these models help the evaluator interpret the results and know the inner-working of these models in comparison with other advanced models based on Deep Learning, which are regarded as black boxes.

7 Conclusions

In this work we revisited the Majority attack, also named 51% vulnerability, one of the main drawbacks of blockchain networks. As discussed previously, this problem occurs because a blockchain network is based on a distributed consensus mechanism to establish trust in the network, and if an attacker, or group of malintentioned users, can carry out an attack by owning more than 51% of the blockchain network's computing power (hash power), they can rewrite the blockchain and violate its main principles.

Therefore, this thesis work proposed objective was to "Design and implement detection and mitigation strategies of the majority attack (51% attack) in a distributed blockchain system, based on the characterization of the miners' behavior." Four significant contributions are presented in this work, to contribute to the main goal of this research.

First of all, a new characterization of consensus algorithms was presented, which can be used to find families of mechanisms using cluster based classification. The results show the identification of new clusters of consensus mechanisms, and describes the behavioral patterns not seen before in the literature.

The second contribution was a detailed characterization of the miners in the bitcoin and crypto ethereum blockchains, to prove the computing distribution assumption is not completely alligned with the reality of these blockchain networks and that the creation of profiles may allow the detection of anomalous behaviors and prevent 51% attacks. The analysis results show that, in the last years, there has been an increasing concentration of hash rate power in a very small set of miners, which generates a real risk for current blockchains. Also, there is a pattern in mining among the main miners, which makes it possible to identify out-of-normal behavior.

The third and main contribution of this thesis is a detailed proposal of a Proof-of-Accuracy protocol. The proposed protocol improves the original proposal by removing the need for a coordinator and combines the Proof-of-Work feature with access to random locations to improve the protocol's resistance to majority attacks. It aims to democratize the miners' participation within a blockchain, control the miners' computing power, and mitigate the 51% vulnerability.

Finally, an exploration of some machine learning classification models for detecting cryptojacking, are presented. The results suggest that simple models, such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, and k-Nearest Neighbor models, can achieve success rates similar to or greater than that of advanced algorithms such as XGBoost and even those of other works based on Deep Learning.

Based on the conclusions of this work, the following ideas for future works are proposed:

1. An area that can be explored is the definition of the requirements for new consensus algorithm; that is, according to each characteristic such as energy consumption, modern hardware, and bifurcation described in Table (**3-1**. Another proposal is to include the type of problem to be solved, given that some of these characteristics may not be relevant for all particular cases and thus, it may change the decision of what type of algorithm to use. Likewise, in future work must consider the conditions for applying certain consensus algorithms by analyzing characteristics that can be more or less efficient depending on the scenario in which they work.

2. A tool to identify anomalous behavior can be implemented to detect a possible attack being performed by a miner or group of miners and generate a general alert to protect the integrity of the blockchain.

3. The proposed protocol can be analyzed and improved in a scenario with dishonest participants, where they may not follow its rules or carry out attacks compromising its secret information. Another research topic is to make the protocol quantum-resistant since it is a Diffie-Hellman-like cryptographic construction, which is not quantum-resistant.

Bibliography

[MSS,] A crypto-mining botnet has been hijacking MSSQL servers for almost two years — ZDNet.

[ZDN,] Bug hunter finds cryptocurrency-mining botnet on DOD network — ZDNet.

[car,] caretSBF: Selection By Filtering (SBF) Helper Functions in caret: Classification and Regression Training.

[Coi,] Coinhive – Monero Mining Club.

[Cor,] Corda. `https://www.corda.net/`. Accessed 2022-07-05.

[Dec, a] Crypto mining botnet found on Defense Department web server - Decrypt.

[Cry, a] Cryptojacking campaign impacts nearly 1.500 websites - noticias de seguridad - trend micro es.

[Cry, b] Cryptojacking: What is it and what you need to know in 2022.

[Cry, c] CryptoLoot - Earn More From Your Traffic.

[Mic,] Detected Cryptojacking Prompts Microsoft to Remove Eight Free Apps from Microsoft Store.

[Fed,] Federated Consensus. Accessed: 2022-07-05.

[Hyp, a] Hyperledger Iroha Documentation. Accessed: 2022-07-05.

[Hyp, b] Hyperledger – Open Source Blockchain Technologies. `https://www.hyperledger.org/`. Accessed 2022-07-05.

[iro,] Iroha Whitepaper. Accessed: 2022-07-05.

[Dec, b] Man fined $7.000 for using Russian supercomputer to mine Bitcoin - Decrypt.

[nin,] Nintendo Switch Game Pulled Over Cryptojacking Concerns.

[Nxt,] Nxt Whitepaper - Introduction: Nxt Whitepaper. `https://nxtdocs.jelurida.com/Nxt_Whitepaper`. Accessed 2022-07-05.

[tre,] Over 200,000 MikroTik Routers Compromised in Cryptojacking Campaign.

[Del,] Proof of Stake vs. Delegated Proof of Stake — Gemini. https://www.gemini.com/cryptopedia/proof-of-stake-delegated-pos-dpos#section-delegated-proof-of-stake. Accessed 2022-07-05.

[RFE,] Recursive Feature Elimination (RFE) for Feature Selection in Python.

[Rus,] Russian Scientists Arrested for Crypto Mining at Nuclear Lab - CoinDesk.

[saw,] sawtooth. https://sawtooth.hyperledger.org/docs/1.2/. Accessed 2022-07-05.

[Sym, a] Several Cryptojacking Apps Found on Microsoft Store — Symantec Blogs.

[Sym, b] Symbiont - Enterprise Fintech Using Blockchain Technology. https://www.symbiont.io/. Accessed: 2022-07-05.

[Tes,] Tesla's Cloud Hacked, Used to Mine Cryptocurrency.

[You,] YouTube ads have been secretly mining cryptocurrency — Mashable.

[Zoo,] Zoomed In: A Look into a Coinminer Bundled with Zoom Installer.

[kas,] ¿Qué es el cryptojacking y cómo funciona?

[DBL, 2019] (2019). *ICBCT 2019: Proceedings of the 2019 International Conference on Blockchain Technology*, New York, NY, USA. Association for Computing Machinery.

[Ahlgren, 2014] Ahlgren, J. (2014). The Probability Distribution for Draws Until First Success Without Replacement. *arXiv: Probability*.

[Aponte et al., 2021a] Aponte, F., Gutierrez, L., Pineda, M., Meriño, I., Salazar, A., and Wightman, P. (2021a). Cluster-based classification of blockchain consensus algorithms. *IEEE Latin America Transactions*, 19(4):688–696.

[Aponte et al., 2021b] Aponte, F., Villanueva, R., and Wightman, P. (2021b). bitcoin miners with individualized unknown miners, mendeley data, v1.

[Aponte et al., 2021c] Aponte, F., Villanueva, R., and Wightman, P. (2021c). Daily computing power bitcoin and cryto ethereum, mendeley data, v1.

[Aponte-Novoa et al., 2022a] Aponte-Novoa, F. A., Daniel, P.-A., and Villanueva-Polanco, R. (2022a). Detecting cryptojacking on web sites use classifiers.

[Aponte Novoa et al., 2021] Aponte Novoa, F. A., Jabba-Molinares, D., and Wightman-Rojas, P. M. (2021). Uso y aplicaciones de la integración entre com-

putación cuantica y blockchain: Revisión sistemática exploratoria. *Mundo FESC*, 11(21):156–165.

[Aponte-Novoa et al., 2021] Aponte-Novoa, F. A., Orozco, A. L. S., Villanueva-Polanco, R., and Wightman, P. (2021). The 51% attack on blockchains: A mining behavior study. *IEEE Access*, 9:140549–140564.

[Aponte-Novoa et al., 2022b] Aponte-Novoa, F. A., Povedano Álvarez, D., Villanueva-Polanco, R., Sandoval Orozco, A. L., and García Villalba, L. J. (2022b). On detecting cryptojacking on websites: Revisiting the use of classifiers. *Sensors*, 22(23).

[Aponte-Novoa and Villanueva-Polanco, 2021] Aponte-Novoa, F. A. and Villanueva-Polanco, R. (2021). Notebook of data analysis bitcoin and crypto ethereum. `https://github.com/faan03/BC_CE_DataAnalysis`.

[Aponte-Novoa and Villanueva-Polanco, 2022a] Aponte-Novoa, F. A. and Villanueva-Polanco, R. (2022a). Notebook proof of accuracy consensus protocol.

[Aponte-Novoa and Villanueva-Polanco, 2022b] Aponte-Novoa, F. A. and Villanueva-Polanco, R. (2022b). On proof-of-accuracy consensus protocols. *Mathematics*, 10(14).

[Aponte-Novoa and Villanueva-Polanco, 2022c] Aponte-Novoa, F. A. and Villanueva-Polanco, R. (2022c). Proof of accuracy consensus protocol. `https://github.com/faan03/proof_of_accuracy_consensus_Protocol`.

[Arjun and Suprabha, 2020] Arjun, R. and Suprabha, K. R. (2020). Innovation and Challenges of Blockchain in Banking: A Scientometric View. *International Journal of Interactive Multimedia and Artificial Intelligence*, InPress(InPress):1.

[Arthur and Vassilvitskii, 2007] Arthur, D. and Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In *SODA '07*.

[Bach et al., 2018] Bach, L. M., Mihaljevic, B., and Zagar, M. (2018). Comparative analysis of blockchain consensus algorithms. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1545–1550.

[Bentov et al., 2014] Bentov, I., Lee, C.-T., Mizrahi, A., and Rosenfeld, M. (2014). Proof of activity: Extending bitcoin's proof of work via proof of stake [extended abstract]. *IACR Cryptol. ePrint Arch.*, 2014:452.

[Bijmans et al., 2019] Bijmans, H. L., Booij, T. M., and Doerr, C. (2019). Inadvertently making cyber criminals rich: A comprehensive study of cryptojacking campaigns at internet scale. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1627–1644, Santa Clara, CA. USENIX Association.

[Bitcoinwiki, 2016] Bitcoinwiki (2016). SHA-256. `https://en.bitcoin.it/wiki/SHA-256`. Accessed: 2021-1-20.

[BiXBiT, 2018] BiXBiT (2018). Merged mining - collective benefit and a panacea for 51% attack? https://medium.com/@bixbit.official/merged-mining-collective-benefit-and-a-panacea-for-51-attack-373404106a9. Accessed: 2021-06-20.

[Block, 2018] Block, A. (2018). Mitigating 51% attacks with llmq-based chainlocks — by alexander block — dash blog. https://blog.dash.org/mitigating-51-attacks-with-llmq-based-chainlocks-7266aa648ec9.

[Blockchain ETL, 2021] Blockchain ETL (2021). GitHub - blockchain-etl/ethereum-etl. `https://github.com/blockchain-etl/ethereum-etl#readme`. Accessed: 2021-05-09.

[Blockchain.com, 2021] Blockchain.com (2021). Blockchain Explorer - Search the Blockchain — BTC — ETH — BCH. `https://www.blockchain.com/explorer`. Accessed: 2021-04-09.

[Blocki and Zhou, 2016] Blocki, J. and Zhou, H.-S. (2016). Designing Proof of Human-Work Puzzles for Cryptocurrency and Beyond. In *Theory of Cryptography*, pages 517–546, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Boneh and Shoup, 2020] Boneh, D. and Shoup, V. (2020). *A Graduate Course in Applied Cryptography*. Self-publishing.

[Breiman et al., 2017] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (2017). *Classification and regression trees*. Routledge.

[Burges, 2010] Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81.

[Carlin et al., 2020] Carlin, D., Burgess, J., O'Kane, P., and Sezer, S. (2020). You could be mine(d): The rise of cryptojacking. *IEEE Security & Privacy*, 18(2):16–22.

[Carmen and Rafael, 2013] Carmen, X. G. M. and Rafael, S. M. C. (2013). *Fundamentos de las Técnicas Multivariantes*. AULA ABIERTA. UNED.

[Catena et al., 2003] Catena, A., Alvarez, M., and Trujillo Mendoza, H. (2003). *Análisis multivariado. Un manual para investigadores.* Biblioteca Nueva.

[ChainZilla, 2019] ChainZilla (2019). Blockchain Security and How to Mitigate. https://medium.com/chainzilla/solutions-to-51-attacks-and-double-spending-71526be4bb86. Accessed: 2021-02-17.

[Chaudhry and Yousaf, 2019] Chaudhry, N. and Yousaf, M. M. (2019). Consensus

Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities. *ICOSST 2018 - 2018 International Conference on Open Source Systems and Technologies, Proceedings*, (December 2018):54–63.

[Chepurnoy et al., 2018] Chepurnoy, A., Duong, T., Fan, L., and Zhou, H.-S. (2018). Twinscoin: A cryptocurrency via proof-of-work and proof-of-stake. *IACR Cryptol. ePrint Arch.*, 2017:232.

[Cox and Donnelly, 2011] Cox, D. R. and Donnelly, C. A. (2011). *Principles of Applied Statistics*. Cambridge University Press.

[Cozza et al., 2020] Cozza, F., Guarino, A., Isernia, F., Malandrino, D., Rapuano, A., Schiavone, R., and Zaccagnino, R. (2020). Hybrid and lightweight detection of third party tracking: Design, implementation, and evaluation. *Computer Networks*, 167:106993.

[Cryptocompare.com, 2015] Cryptocompare.com (2015). What is merged mining – Bitcoin & Namecoin – Litecoin & Dogecoin? https://www.cryptocompare.com/mining/guides/what-is-merged-mining-bitcoin-namecoin-litecoin-dogecoin/.

[Dasarathy, 1991] Dasarathy, B. V. (1991). Nearest neighbor (nn) norms: Nn pattern classification techniques. *IEEE Computer Society Tutorial*.

[Dean, 2015] Dean (2015). 51% attack. `http://cryptorials.io/glossary/51-attack/`. Accessed: 2019-07-14.

[Dey, 2018] Dey, S. (2018). Securing majority-attack in blockchain using machine learning and algorithmic game theory: A proof of work. *arXiv*, pages 7–10.

[Dreiseitl and Ohno-Machado, 2002] Dreiseitl, S. and Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, 35(5):352–359.

[Du et al., 2017] Du, M., Ma, X., Zhang, Z., Wang, X., and Chen, Q. (2017). A review on consensus algorithm of blockchain. *2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, 2017-Janua:2567–2572.

[Duong et al., 2020] Duong, T., Fan, L., Katz, J., Thai, P., and Zhou, H.-S. (2020). 2-hop Blockchain: Combining Proof-of-Work and Proof-of-Stake Securely. In Chen, L., Li, N., Liang, K., and Schneider, S., editors, *Computer Security – ESORICS 2020*, pages 697–712, Cham. Springer International Publishing.

[Duong et al., 2017] Duong, T., Fan, L., and Zhou, H.-S. (2017). 2-hop Blockchain: Combining Proof-of-Work and Proof-of-Stake Securely. *Cryptol. ePrint Arch.*

[Evans et al., 2018] Evans, D., Kolesnikov, V., and Rosulek, M. (2018). *A Pragmatic Introduction to Secure Multi-Party Computation*, volume 2. NOW Publishers Inc.

[Eyal et al., 2016] Eyal, I., Gencer, A. E., Sirer, E. G., and Van Renesse, R. (2016). Bitcoin-NG: A scalable blockchain protocol. In *13th USENIX symposium on networked systems design and implementation (NSDI 16)*, pages 45–59.

[Eyal and Sirer, 2014] Eyal, I. and Sirer, E. G. (2014). How to disincentivize large bitcoin mining pools. *Blog post: http://hackingdistributed. com/2014/06/18/how-to-disincentivize-large-bitcoin-mining-pools*.

[Feng et al., 2014] Feng, J., Xu, H., Mannor, S., and Yan, S. (2014). Robust logistic regression and classification. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

[Franco et al., 2014] Franco, J., Rodríguez, A., and Jiménez, E. (2014). *Estadística Aplicada II: Estadística en Administración para Toma de Decisiones*. Económico Administrativo. Grupo Editorial Patria.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232.

[Gao et al., 2018] Gao, J., Asamoah, K. O., Sifah, E. B., Smahi, A., Xia, Q., Xia, H., Zhang, X., and Dong, G. (2018). Gridmonitoring: Secured sovereign blockchain based monitoring on smart grid. *IEEE Access*, 6:9917–9925.

[Garoffolo et al., 2018] Garoffolo, A., Stabilini, P., Vigliano, R., and Stav, U. (2018). Proposal to modify Satoshi consensus to enhance protection against 51% attack. https://www.horizen.global/assets/files/A-Penalty-System-for-Delayed-Block-Submission-by-Horizen.pdf.

[Gomes and Correia, 2020] Gomes, F. and Correia, M. (2020). Cryptojacking detection with cpu usage metrics. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, pages 1–10. IEEE.

[Google Cloud Platform, 2021] Google Cloud Platform (2021). BigQuery – Google Cloud Platform Crypto Ethereum dataset. `https://bit.ly/2Yl61iI`. Accessed: 2021-05-09.

[Greenspan et al., 2015] Greenspan, G. et al. (2015). Multichain private blockchain-white paper. *URl: http://www.multichain.com/download/MultiChain-White-Paper.pdf*, 85.

[Guarino et al., 2022] Guarino, A., Malandrino, D., and Zaccagnino, R. (2022). An

automatic mechanism to provide privacy awareness and control over unwittingly dissemination of online private information. *Computer Networks*, 202:108614.

[Hajdarbegovic, 2014] Hajdarbegovic, N. (2014). Bitcoin Miners Ditch Ghash.io Pool Over Fears of 51% Attack. `https://www.coindesk.com/bitcoin-miners-ditch-ghash-io-pool-51-attack`. Accessed: 2021-05-02.

[Hammer Øyvind, Harper David A.T, 2001] Hammer Øyvind, Harper David A.T, R. P. D. (2001). Palaeontologia Electronica. Accedido 20-10-2019.

[Hao et al., 2018] Hao, Y., Li, Y., Dong, X., Fang, L., and Chen, P. (2018). Performance Analysis of Consensus Algorithm in Private Blockchain. *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018-June(Iv):280–285.

[Hernandez-Suarez et al., 2022] Hernandez-Suarez, A., Sanchez-Perez, G., Toscano-Medina, L. K., Olivares-Mercado, J., Portillo-Portilo, J., Avalos, J.-G., and García Villalba, L. J. (2022). Detecting cryptojacking web threats: An approach with autoencoders and deep dense neural networks. *Applied Sciences*, 12(7).

[Hoogerwerf et al., 2021] Hoogerwerf, E., van Tetering, D., Bay, A., and Erkin, Z. (2021). Efficient joint random number generation for secure multi-party computation. In di Vimercati, S. and Samarati, P., editors, *Proceedings of the 18th International Conference on Security and Cryptography, SECRYPT 2021*, Proceedings of the 18th International Conference on Security and Cryptography, SECRYPT 2021, pages 436–443. SciTePress. 18th International Conference on Security and Cryptography, SECRYPT 2021 ; Conference date: 06-07-2021 Through 08-07-2021.

[Ikram et al., 2016] Ikram, M., Asghar, H. J., Kâafar, M. A., Krishnamurthy, B., and Mahanti, A. (2016). Towards seamless tracking-free web: Improved detection of trackers via one-class learning. *CoRR*, abs/1603.06289.

[Javed et al., 2021] Javed, I. T., Alharbi, F., Bellaj, B., Margaria, T., Crespi, N., and Qureshi, K. N. (2021). Health-id: A blockchain-based decentralized identity management for remote healthcare. *Healthcare*, 9(6).

[Kaur et al., 2022] Kaur, M., Gupta, S., Kumar, D., Verma, C., Neagu, B.-C., and Raboaca, M. S. (2022). Delegated proof of accessibility (dpoac): A novel consensus protocol for blockchain systems. *Mathematics*, 10(13).

[Komodo, 2018] Komodo (2018). Komodo: An Advanced Blockchain Technology, Focused on Freedom. https://cryptorating.eu/whitepapers/Komodo/2018-02-14-Komodo-White-Paper-Full.pdf. Accessed 2021-06-18.

[Kudin et al., 2019] Kudin, A. M., Kovalenko, B. A., and Shvidchenko, I. V. (2019).

Blockchain Technology: Issues of Analysis and Synthesis. *Cybernetics and Systems Analysis*, 55(3):488–495.

[Kurpjuweit et al., 2021] Kurpjuweit, S., Schmidt, C. G., Klöckner, M., and Wagner, S. M. (2021). Blockchain in additive manufacturing and its impact on supply chains. *Journal of Business Logistics*, 42(1):46–70.

[Kursawe et al., 2011] Kursawe, K., Danezis, G., and Kohlweiss, M. (2011). Privacy-friendly aggregation for the smart-grid. In *Privacy Enhancing Technologies*, volume 6794, pages 175–191.

[Lamport, 2001] Lamport, L. (2001). Paxos made simple. *Sigact News - SIGACT*, 32.

[Le and Hsu, 2021] Le, T.-V. and Hsu, C.-L. (2021). A systematic literature review of blockchain technology: Security properties, applications and challenges. *Journal of Internet Technology*, 22:789–801.

[Le et al., 2022] Le, T.-V., Hsu, C.-L., and Chen, W.-X. (2022). A hybrid blockchain-based log management scheme with nonrepudiation for smart grids. *IEEE Transactions on Industrial Informatics*, 18(9):5771–5782.

[Legendre, 2012] Legendre, P. (2012). *Numerical Ecology: Numerical Ecology.* Issn Ser. ELSEVIER SCIENCE B.V, 3rd ed.. edition.

[Li, 2012] Li, P. (2012). Robust logitboost and adaptive base class (abc) logitboost.

[Li et al., 2020] Li, X., Jiang, P., Chen, T., Luo, X., and Wen, Q. (2020). A survey on the security of blockchain systems. *Future Generation Computer Systems*, 107(November 2020):841–853.

[Liu et al., 2018] Liu, J., Zhao, Z., Cui, X., Wang, Z., and Liu, Q. (2018). A novel approach for detecting browser-based silent miner. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pages 490–497. IEEE.

[Liu et al., 2020] Liu, Y., Liu, J., Zhang, Z., and Yu, H. (2020). A Fair Selection Protocol for Committee-Based Permissionless Blockchains. *Computers & Security*, 91:101718.

[Malik and Anwar, 2022] Malik, A. W. and Anwar, Z. (2022). Do charging stations benefit from cryptojacking? a novel framework for its financial impact analysis on electric vehicles. *Energies*, 15(16).

[Mazieres, 2015] Mazieres, D. (2015). The stellar consensus protocol: A federated model for internet-level consensus. *Stellar Development Foundation*.

[Miller et al., 2015] Miller, A., Kosba, A., Katz, J., and Shi, E. (2015). Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 680–691.

[Milutinovic et al., 2016] Milutinovic, M., He, W., Wu, H., and Kanwal, M. (2016). Proof of luck. In *Proceedings of the 1st Workshop on System Software for Trusted Execution*. ACM.

[Minchev, 2018] Minchev, R. (2018). PirlGuard — Innovative Solution against 51% Attacks. https://medium.com/pirl/pirlguard-innovative-solution-against-51-attacks-87dd45aa1109.

[N and M, 2019] N, A. and M, V. (2019). Blockchain Security Attack: A Brief Survey. In *2019 10th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2019*, pages 6–11. IEEE.

[Nakamoto, 2008] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash SyNakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Consulted, 1–9. doi:10.1007/s10838-008-9062-0stem. *Journal for General Philosophy of Science*, 39(1):53–67.

[Naseem et al., 2021] Naseem, F., Aris, A., Babun, L., Tekiner, E., and Uluagac, S. (2021). Minos: A lightweight real-time cryptojacking detection system. In *28th Annual Network and Distributed System Security Symposium, NDSS*.

[Ng, 2018] Ng, E. (2018). Dash to mitigate 51% attacks with chainlocks. https://blockchainreporter.net/dash-to-mitigate-51-attacks-with-ChainLocks/. Accessed: 2021-06-20.

[Nguyen and Kim, 2018a] Nguyen, G. T. and Kim, K. (2018a). A survey about consensus algorithms used in Blockchain. *Journal of Information Processing Systems*.

[Nguyen and Kim, 2018b] Nguyen, G.-T. and Kim, K. (2018b). A survey about consensus algorithms used in blockchain. *Journal of Information Processing Systems*, 14(1):101–128. cited By 93.

[Om Kumar and Sathia Bhama, 2019] Om Kumar, C. and Sathia Bhama, P. R. (2019). Detecting and confronting flash attacks from iot botnets. *The Journal of Supercomputing*, 75(12):8312–8338.

[Omer and Shareef, 2022] Omer, Z. M. and Shareef, H. (2022). Comparison of decision tree based ensemble methods for prediction of photovoltaic maximum current. *Energy Conversion and Management: X*, page 100333.

[O'Shea and Nash, 2015] O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks.

[Oyinloye et al., 2021] Oyinloye, D. P., Teh, J. S., Jamil, N., and Alawida, M. (2021). Blockchain consensus: An overview of alternative protocols. *Symmetry*, 13(8).

[P4Titan, 2014] P4Titan (2014). Slimcoin. a peer-to-peer crypto-currency with proof-of-burn "mining without powerful hardware". Accessed: 2021-1-20.

[Pagh and Rodler, 2001] Pagh, R. and Rodler, F. F. (2001). Cuckoo hashing. *BRICS Report Series*, 8(32).

[Park et al., 2018] Park, S., Kwon, A., Fuchsbauer, G., Gaži, P., Alwen, J., and Pietrzak, K. (2018). SpaceMint: A Cryptocurrency Based on Proofs of Space. In *Financial Cryptography and Data Security*, pages 480–499, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Petrov et al., 2020] Petrov, I., Invernizzi, L., and Bursztein, E. (2020). Coinpolice:detecting hidden cryptojacking attacks with neural networks.

[Piscini et al., 2016] Piscini, E., Guastella, J., Rozman, A., and Nassim, T. (2016). Innovating in the digital era.

[Prashanth et al., 2022] Prashanth, S. K., Shitharth, S., Praveen Kumar, B., Subedha, V., and Sangeetha, K. (2022). Optimal feature selection based on evolutionary algorithm for intrusion detection. *SN Computer Science*, 3(6):439.

[Ren, 2014] Ren, L. (2014). Proof of Stake Velocity: Building the Social Currency of the Digital Age. Accessed: 2021-1-20.

[Richardson et al., 2007] Richardson, M., Dominowska, E., and Ragno, R. (2007). Predicting clicks: Estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, page 521–530, New York, NY, USA. Association for Computing Machinery.

[Ripley, 2007] Ripley, B. D. (2007). *Pattern recognition and neural networks*. Cambridge university press.

[Rosenfeld, 2014] Rosenfeld, M. (2014). Analysis of Hashrate-Based Double Spending. *ArXiv*, abs/1402.2009.

[Saad et al., 2021] Saad, M., Qin, Z., Ren, K., Nyang, D., and Mohaisen, D. (2021). e-pos: Making proof-of-stake decentralized and fair. *IEEE Transactions on Parallel and Distributed Systems*, 32(8):1961–1973.

[Saberi et al., 2019] Saberi, S., Kouhizadeh, M., Sarkis, J., and Shen, L. (2019).

Blockchain technology and its relationships to sustainable supply chain management. *International Journal of Production Research*, 57(7):2117–2135.

[Sankar et al., 2017] Sankar, L. S., Sindhu, M., and Sethumadhavan, M. (2017). Survey of consensus protocols on blockchain applications. *2017 4th International Conference on Advanced Computing and Communication Systems, ICACCS 2017*.

[Sarabia Alegría et al., 2018] Sarabia Alegría, J., Prieto Mendoza, F., and Jordá Gil, V. (2018). *Prácticas de estadística con R*. Economía Y Empresa. Ediciones Pirámide.

[Sayeed and Marco-Gisbert, 2020] Sayeed, S. and Marco-Gisbert, H. (2020). Proof of Adjourn (PoAj): A novel approach to mitigate blockchain attacks. *Applied Sciences (Switzerland)*, 10(18).

[Schwartz et al., 2014] Schwartz, D., Youngs, N., Britto, A., et al. (2014). The ripple protocol consensus algorithm. *Ripple Labs Inc White Paper*, 5(8):151.

[Scicchitano et al., 2020] Scicchitano, F., Liguori, A., Guarascio, M., Ritacco, E., and Manco, G. (2020). A deep learning approach for detecting security attacks on blockchain. In *CEUR Workshop Proceedings*, volume 2597, pages 212–222.

[Sivaraju, 2022] Sivaraju, S. (2022). An insight into deep learning based cryptojacking detection model. *Journal of Trends in Computer Science and Smart Technology*, 4(3):175–184.

[Snoeren, 2021] Snoeren, M. (2021). python-p2p-Network: Framework to easily implement decentralized peer-to-peer Network Applications in Python. `https://github.com/macsnoeren/python-p2p-network`.

[Sompolinsky and Zohar, 2013] Sompolinsky, Y. and Zohar, A. (2013). Accelerating bitcoin's transaction processing. fast money grows on trees, not chains. *IACR Cryptol. ePrint Arch.*, 2013:881.

[Sunny, 2013] Sunny, K. (2013). Primecoin: Cryptocurrency with prime number proof-of-work.

[Sunny and Scott, 2012] Sunny, K. and Scott, N. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. Technical report, self-published paper.

[Suthaharan, 2016] Suthaharan, S. (2016). Chapter 6 - a cognitive random forest: An intra- and intercognitive computing for big data classification under cune condition. In Gudivada, V. N., Raghavan, V. V., Govindaraju, V., and Rao, C., editors, *Cognitive Computing: Theory and Applications*, volume 35 of *Handbook of Statistics*, pages 207–227. Elsevier.

[Tang et al., 2017] Tang, S., Liu, Z., Chow, S. S. M., Liu, Z., and Long, Y. (2017). Forking-free hybrid consensus with generalized proof-of-activity. *IACR Cryptol. ePrint Arch.*, page 367.

[Tayyab et al., 2022] Tayyab, U.-e.-H., Khan, F. B., Durad, M. H., Khan, A., and Lee, Y. S. (2022). A survey of the recent trends in deep learning based malware detection. *Journal of Cybersecurity and Privacy*, 2(4):800–829.

[Tekiner et al., 2021] Tekiner, E., Acar, A., Uluagac, A. S., Kirda, E., and Selcuk, A. A. (2021). Sok: Cryptojacking malware. In *2021 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 120–139.

[Vasin, 2014] Vasin, P. (2014). BlackCoin's Proof-of-Stake Protocol v2 Pavel.

[Ventrone, 2021a] Ventrone, J. (2021a). Bitcoin Blockchain Data — Kaggle. `https://bit.ly/3DVOr5b`. Accessed: 2021-05-09.

[Ventrone, 2021b] Ventrone, J. (2021b). GitHub JuanVentrone/bitcoin_blockchain_scrapper. `https://bit.ly/2Vtch72`. Accessed: 2021-05-09.

[Wang and Kogan, 2018] Wang, Y. and Kogan, A. (2018). Designing confidentiality-preserving blockchain-based transaction processing systems. *International Journal of Accounting Information Systems*, 30:1–18. 2017 Research Symposium on Information Integrity & Information Systems Assurance.

[Wessel and Olson, 2016] Wessel, D. and Olson, P. (2016). The Hutchins Center Explains: How blockchain could change the financial system (part 1) — Brookings Institution. https://www.brookings.edu/blog/up-front/2016/01/11/the-hutchins-center-explains-how-blockchain-could-change-the-financial-system-part-1/.

[Wu et al., 2017] Wu, H., Li, Z., King, B., Ben Miled, Z., Wassick, J., and Tazelaar, J. (2017). A distributed ledger for supply chain physical distribution visibility. *Information*, 8(4).

[Wu et al., 2022] Wu, M.-H., Lai, Y.-J., Hwang, Y.-L., Chang, T.-C., and Hsu, F.-H. (2022). Minerguard: A solution to detect browser-based cryptocurrency mining through machine learning. *Applied Sciences*, 12(19).

[Yang et al., 2019] Yang, X., Chen, Y., and Chen, X. (2019). Effective scheme against 51% attack on proof-of-work blockchain with history weighted information. In *Proceedings - 2019 2nd IEEE International Conference on Blockchain, Blockchain 2019*, pages 261–265. IEEE.

[Ying et al., 2022] Ying, Q., Yu, Y., Tian, D., Jia, X., Ma, R., and Hu, C. (2022).

Cjspector: A novel cryptojacking detection method using hardware trace and deep learning. *Journal of Grid Computing*, 20:31.

[Yu et al., 2019] Yu, B., Liu, J., Nepal, S., Yu, J., and Rimba, P. (2019). Proof-of-QoS: QoS Based Blockchain Consensus Protocol. *Computers & Security*, 87:101580.

[Zheng et al., 2017] Zheng, Z., Xie, S., Dai, H., Chen, X., and Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE International Congress on Big Data (BigData Congress)*, pages 557–564.