Annals of Emerging Technologies in Computing (AETiC)
Vol. 4, No. 1, 2020

Review Article

A Survey on PoW-based Consensus

Alessio Meneghetti*, Massimiliano Sala and Daniele Taufer

University of Trento, Italy
alessio.meneghetti@unitn.it; maxsalacodes@gmail.com; daniele.taufer@unitn.it
*Correspondence: alessio.meneghetti@unitn.it

Received: 10th November 2019; Accepted: 14th December 2019; Published: 1st January 2020

Abstract: We provide a historical overview of proof-of-work techniques and the fields in which it plunges its roots. We are interested in PoW-techniques applied to blockchain technology and therefore we survey the state-of-the-art protocols employing these methods for consensus algorithms, emphasizing the differences between the efficient hashcash systems and the promising bread pudding protocols. Afterwards, the consensus mechanisms are discussed and some interesting known attacks to these algorithms are collected and classified according to their underlying ideas.

Keywords: ***Proof of Work (PoW); Blockchain; Distributed Digital Ledger; Hashcash; Bread Pudding Protocols; Consensus algorithms***

1. Introduction

The notion of proof of work (PoW) encloses a wide number of techniques that have been developed in the past decades and whose finality is to demonstrate that a prover has performed a certain amount of computational work in a specified interval of time.

The formalization of this concept dates back to 1999 [1], although previous examples of delaying functions used for such a purpose has appeared earlier. In 1992 [2] a PoW-technique has been proposed for fighting junk emails by requiring the sender to compute some moderately expensive function of the message, which constitutes a negligible effort for a sober user of this service but serves as a deterrent to unsolicited mail spam. Trapdoors were also designed for pricing functions, allowing trusted parties to grant bulk mailings with ease. Another forerunner of PoW is the notion of uncheatable benchmark, which appeared in 1993 [3] - 1994 [4] and consists of tasks that enforce a certain computational load on the execution entity, but in this case without privileged users that can shortcut the calculation duties. All these constructions, sometimes referred to as timing function, have been employed and refined in 1997 [5] for metering web-sites accesses in a light and verifiable way. Another application of these ideas was studied in 1998 [6] to develop a lottery scheme that works fairly using only internal information, providing at the same time resistance against coalitions of ticket purchasers and detection of (if any) frauds on the part of the lottery agent. Moreover, a wide variety of PoW-puzzles has been contrived for limiting or controlling decryption time [7] or connection depletion attacks [8] by arbitrarily delaying users who want to access specific resources.

The applications mentioned above have subsequently seen thriving research that has highlighted potentialities and limits of PoW-related techniques. As an informative instance, the native idea of developing PoW-systems to face email spam has long been debated and is still an open field of research: in 2004 [9] the PoW-method was proved not to work in its basic formulation and

Alessio Meneghetti, Massimiliano Sala and Daniele Taufer, "A Survey on PoW-based Consensus", *Annals of Emerging Technologies in Computing (AETiC)*, Print ISSN: 2516-0281, Online ISSN: 2516-029X, pp. 8-18, Vol. 4, No. 1, 1st January 2020, Published by International Association of Educators and Researchers (IAER), DOI: 10.33166/AETiC.2020.01.002, Available: http://aetic.theiaer.org/archive/v4/v4n1/p2.html.

under realistic assumptions, as it is often the case, but further research [10, 11] has shown that these ideas are still viable when integrated with reputation models, which dynamically adjust the PoW difficulty.

Beyond these purposes, in 2008 another cutting-edge employment of the PoW saw the light: in his first apparition Bitcoin [12] has laid the foundations for the modern cryptocurrencies introducing a PoW-based consensus algorithm, which basically puts miners in competition for solving a cryptographic challenge. Despite many other solutions for achieving consensus have been proposed since then, this latter approach is still considered relevant and for this reason it is summarized here.

This essay is organized as follows: in Section 2 we review the most prominent techniques employed for the production of PoW-challenges, juxtaposing the widely adopted hashcash concept and the more polished bread pudding protocols. In Section 3 we discuss how blockchain consensus protocols may be designed on top of those challenges, whereas in Section 4 we report their known weaknesses.

2. PoW-methods

In this section we describe some mechanisms of PoW that have been devised over the years, exploiting various types of cryptographic and mathematical challenges.

2.1. Hashcash

Hashcash is probably the most known and widespread PoW-system, ideated in 1997 [13] and formally detailed in 2002 [14]. It relies on Hash Functions, which are deterministic functions

$$\mathcal{H}: \bigcup_{i=1}^{\infty} \{0,1\}^i \rightarrow \{0,1\}^L$$

mapping data of arbitrary size onto data of a fixed size L and satisfying the following properties:

1. **Pre-image resistance:** for every $h \in \{0,1\}^L$ it is hard to find a message m with that given hash, i.e. such that $\mathcal{H}(m) = h$.
2. **Second pre-image resistance:** for every message m it is difficult to find a different one $m' \neq m$ such that $\mathcal{H}(m') = \mathcal{H}(m)$.
3. **Collision resistance:** it is hard to find a pair of different messages $m' \neq m$ with the same hashes, i.e. such that $\mathcal{H}(m') = \mathcal{H}(m)$.
4. **Uniformity:** Every hash value in the output range should be generated with roughly the same probability.

The general structure of hashcash systems is the following: to prove that a certain amount of work on some data has been performed, one searches for some bits, called the *Nonce*, that appended to the original data produce a hash with easily identifiable properties, such as having a prescribed number δ of zeros in the first positions.

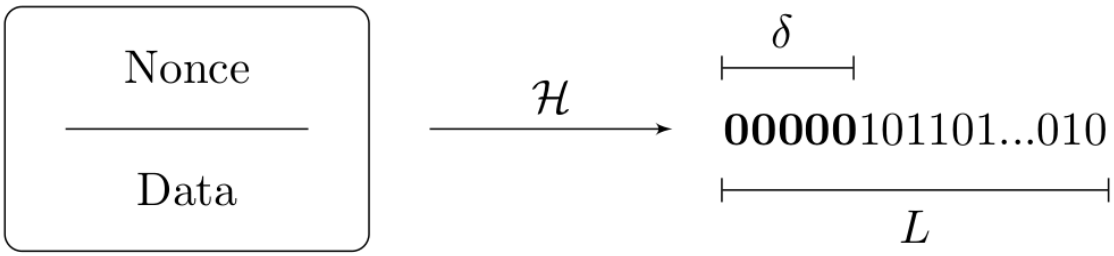

Figure 1. Hashcash basic structure.

Given the above properties of $\mathcal{H}$ there are no strategies better than testing different randomly generated nonces until the required output property is satisfied. Since $\mathcal{H}$ is supposed to be uniform, the probability of achieving the δ-starting-zeros condition is $2^{-\delta}$, thus the difficulty of the hashcash challenge can be easily adjusted by tuning δ, which is called the *difficulty parameter*.

The main reason why hashcash has become so popular is the extremely short time required to verify the result of a challenge: despite the average number of hashes needed to find the desired condition is variable and depends on the difficulty parameter, only one hash is needed to check that the proposed nonce provides an output with the desired properties.

We list now some hashcash-based PoW-protocols employed by modern cryptocurrencies, emphasizing their peculiarities.

- **SHA-256**

This function belongs to the Secure Hash Algorithms (SHA) family designed in 2001 by the American agency NSA and chosen by NIST as the U.S. Federal Information Processing Standard (FIPS). A comprehensive description of these functions may be found in [15]. The SHA-2 family is still considered secure (in contrast to SHA-0 and SHA-1 ones), even though the more recent SHA-3, selected by NIST in 2012, appears to have better security properties. The SHA-256 function produces outputs of length $L = 256$ bits and is used in Bitcoin [12] and related cryptocurrencies such as Bitcoin Cash, Counterparty, MazaCoin, Namecoin and by hybrid systems such as NeuCoin and Peercoin.

- **Ethash**

This function has been specifically designed for the Ethereum consensus protocol, as described in [16]. It is obtained from the Kekkak hash function [17] combined with modified versions of Hashimoto [18] and Dagger [19] algorithms in order to achieve ASIC-resistance by memory-hard computations and light client verifiability. Nonetheless, ASIC miners for Ethash has been announced in 2018 but they are not considered a threat to Ethereum, which is developing a new PoW-system (called ProgPoW) for the near future and a PoS-based consensus algorithm for the long term.

- **Scrypt**

This password-based key-derivation function has been developed to use a large amount of memory compared to other key-derivation functions, by using large vectors of pseudorandom strings that are produced from the last block and accessed in a pseudorandom order to output the derived key [20]. For its memory-hardness it has been adopted as part of the PoW-algorithm of some cryptocurrencies aiming at ASIC-resistance, such as Litecoin and Dogecoin.

- **Cryptonote**

Similar to Scrypt, the Cryptonote [21] protocol relies on random memory access, but it depends on all the previous blocks. It acts as a voting system that is assumed to be egalitarian since it makes use of CryptoNight PoW-algorithm, although dedicated devices are under development. This protocol serves many privacy-oriented PoW-algorithms, such as Monero's.

- **Equihash**

Another memory-hard PoW system is Equihash [22], which was specifically ideated to limit parallel implementations by memory bandwidth. In a nutshell, given a generic hash function $\mathcal{H}$, the prover is required to produce messages $m_1, \ldots, m_k$ such that $\mathcal{H}(m_1) \oplus \ldots \oplus \mathcal{H}(m_k) = 0$, i.e. the bitwise sum of their hashes is the zero vector, and the hash of the concatenated message $\mathcal{H}(m_1 || \ldots || m_k) = 0$ has prescribed properties, such as a given number of leading zeros. Equihash has been adopted by cryptocurrencies such as ZCash, Bitcoin Gold and Aion.

- **Concatenating hashes (e.g. X11)**

Some hashcash solutions are obtained by concatenating several hash functions, such as X11 hash developed for Dash [23]. It applies the eleven hashes Blake, Bmw, Grøstl, Jh, Keccak, Skein, Luffa, CubeHash, SHAvite-3, Simd and Echo, one after the other, obtaining a heavier, but arguably more secure, hash function. Following the latter idea, many other composite hashes have been designed, such as X13, X14, X15 and X17.

2.2. Bread pudding alternatives

The various hashcash-based PoWs described in the previous section are extremely efficient and easy to implement, although they all share a common limitation: the large computational power

exploited by nodes is used only for convincing the network of their work, therefore it is substantially useless after the consensus is reached. To have an idea of the extent of this issue, it is estimated [24, 25] that bitcoin miners currently perform around 2^{65} hashes per second, i.e. 2^{90} hashes per year, consuming 0.33% of the world's electricity.

Here we examine some proposals that have been conceived to overcome this problem by working only on the PoW side of consensus, namely by achieving mathematically or cryptographically useful results via these PoW-calculations. Protocols of this type are usually referred to as *bread pudding protocols*. Alternatives that do not employ PoW-based consensus are actually viable but will not be discussed in this paper.

- **MicroMint and E-money**

MicroMint [26] is a micropayment protocol in which coins consists of sets of k values $\{m_1, \dots, m_k\}$ with prescribed properties on their hash values $\mathcal{H}(m_i)$. With an appropriate choice of parameters, minting a single coin amounts to finding hash collisions so it is computationally demanding, whereas minting many of them is not much harder. Thus, if the provider of this payment system mints a sufficiently large amount of coins in advance and limits their validity period, it is just not convenient for an attacker to try forging them, unless it has a monstrous computational power, not commensurate to the entity of the micropayments he wants to forge.

The burden of generating those coins may be obtained by bread pudding minting PoW as proposed in [1], entrusting miners the hard job of finding hash collisions and producing coins by gathering their results together. In this work general types of private PoW algorithms are examined but it might be conceivably adapted to blockchain scenarios.

Similar ideas have been pursued in [27] while developing reusable PoW, namely to generate via PoW tokens that can be sequentially reused: rather than being discarded after use, these tokens may be exchanged for new, equal-value, ones.

- **Data handling**

After Bitcoin release many new models have been proposed to improve critical aspects of this cryptocurrency, such as its energy consumption. Several solutions consisting of new types of PoWs have been examined, with a broader concept of work. Alternatives such as Proof of Space [28], Proof of Secure Erasure [29], Proof of Burn [30] and Proof of Elapsed Time [31] arose to make miners prove their work in energy-inexpensive ways.

One of the projects that may also be considered a bread pudding protocol is Permacoin [32], in which miners have to prove to have invested not only computational resources but also memory or storage. By applying a refined version of Proof-of-Retrievability techniques [33], clients' work maintains a highly decentralized file storage of a publicly valuable archive, which can be recovered with overwhelming probability via erasure-coding.

- **Solving scientific problems**

Some attempts have been made to convert the PoW-computations into useful scientific outputs, both from theoretical and numerical sides. As an instance, prime numbers have always been fascinating mathematicians but their determination is still challenging. With this in mind, in 2013 [34] a PoW-protocol aiming at finding Cunningham chains of primes has been proposed. In a nutshell, this PoW consists of certifying a message m by finding an integer N such that $N \cdot \mathcal{H}(m)$ is the center or the origin of such a prime chain. In its easiest formulation, one requires $p = N \cdot \mathcal{H}(m) - 1$ to be prime as well as $p + 2$, i.e. searches for a multiple of the given hash that lies between two twin primes (that is the shortest possible Cunningham chain). The author claims that verifying pseudo-primality suffices, but we notice that also verifying if

$$4[(p-1)! + 1] + p \equiv 0 \pmod{p(p+2)}$$

is enough by Clement's theorem [35].

Another line of work in this direction suggests creating PoWs based on real world problems, such as DNA and RNA sequencing, surface matching, data mining and computational geometry. A model for integrating matricial instances of these problems was studied in the Proof of eXercise

setting [36], although the verification of such computations presents some limits (it is slow and probabilistic).

It is also worth noting that recently PoWs based on distributed computing networks are spreading. Regardless of the consensus type, miners are required to fulfil specific tasks outlined by external platforms for having their blocks accepted, taking part in a large worldwide computation, as in the case of CureCoin [37] that simulates protein folding for medical and biological purposes.

- **Research propellants**

Should people actually use PoW-protocols revolving around mathematical problems of interest, then there would be a boost of scientific research in those fields.

An example of such a protocol is a PoW-system [38] whose hardness is based on the Orthogonal Vectors (OV) problem. It amounts to deciding if given two sets of vectors $U, V \subseteq \{0,1\}^d$, there are two orthogonal vectors belonging to different sets, i.e. there exist $u \in U$ and $v \in V$ such that they satisfy $< u, v >_{\mathbb{Z}} = 0$. Aside from the practical relevance of some instances of this problem, its theoretical complexity is far from being understood and is conjectured to be quadratic in the size of the considered sets, but new study cases might be of help. Moreover, this may provide mathematicians with an algorithmic motivation to reduce other dilemmas to the OV one, motivating a deep investigation of equivalent classes of problems.

Another recent research topic involves the Discrete Logarithm-based PoW [39, 40]: for many groups G and elements $g, h = g^e \in G$, we are not aware of algorithms asymptotically faster than Pollard's Rho to recover the exponent e. Many cryptographic protocols rely on the postulated difficulty of this problem but it is not excluded that further research in this field could lead to discover asymptotically faster algorithms.

3. Blockchain consensus

Distributed ledger technologies have gained a worldwide interest, mainly due to the sharp increase in cryptocurrencies popularity observed recently. Verifying and maintaining those ledgers requires either central authorities or cryptographic mechanisms, called consensus protocols, which we survey here.

Many such protocols have been proposed, exploiting disparate techniques from mathematics, cryptography, computer science and information theory for achieving high levels of security, throughput, privacy and scalability. Nevertheless, we only focus here on those applied to blockchains and that make use of PoW-methods as discussed in Section 2, sending an interested reader back to other recent surveys on dissimilar types of consensus protocols [41, 42, 43, 44].

Stripped of technical details, the classical realization of a PoW-based consensus on a blockchain involves

- a PoW-challenge C to be solved by miners for creating new blocks;
- a criterion to discern proper blocks from stale ones.

To ensure security and prevent miners advantaging themselves by precomputing instances of C, this challenge has to depend on previous blocks and on the new transitions added to the chain. Therefore the previous block header is typically chosen as input of C, comprising among other things the result of the previous challenge, the Merkle root of new transactions, a timestamp and a nonce, as portrayed below. The challenge is then repeated, by changing the nonce, until the prescribed output conditions are satisfied.

The first miner that solves C broadcasts the new block and receives the correspondent reward. However, forks may occur in the uncommon cases when more miners solve the challenge before being notified of other miners' solutions, or when dishonest miners try to mount malicious attacks, e.g. double-spending by forking. The following criteria have been proposed to settle these occurrences:

- **The oldest fork wins**

According to this rule, in presence of forks an honest miner will append blocks only on the branch that was created first, i.e. the one with the fork block that possesses the lowest timestamp.

This solution might be suitable for not-completely-decentralized blockchains, for which a secure timestamp may be generated, but it is prone to forged stamps and DoS-attacks in blockchains that only rely on geometric properties of the chain.

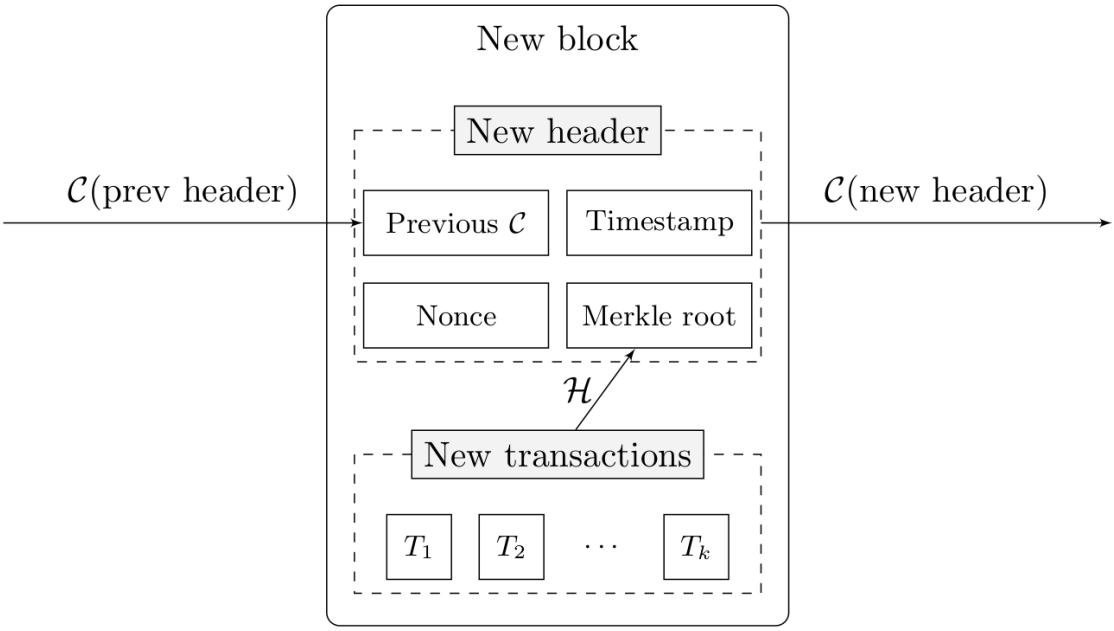

Figure 2. Block structure

- **The longest fork wins**

This was the first solution appeared in Bitcoin, which relies only on the geometry of the chain. It states that the main chain is simply the longest one, so every block that is not included in this chain has to be considered stale and discarded. Despite its elegance, it conveys the risk that an attacker overcomes the honest chain. In fact, in the unlucky case of many forks happening in a short amount of time, a privately mined chain may be chosen as main since the network computational power is distributed among the various branches of the chain.

- **GHOST**

Nowadays, the strategy that is considered the most secure is GHOST [45], which prescribes to consider as main chain the one with the greatest amount of computational power spent on it, keeping in consideration all the blocks (even those that do not belong to the main chain). In this way, to straightforwardly overcome honest miners an attacker should control more than half of the network computational power. In its basic formulation, the number of *uncle* blocks up to the m-th generation is considered (m being a system parameter) and a small reward is granted even to miners that have mined proper uncles.

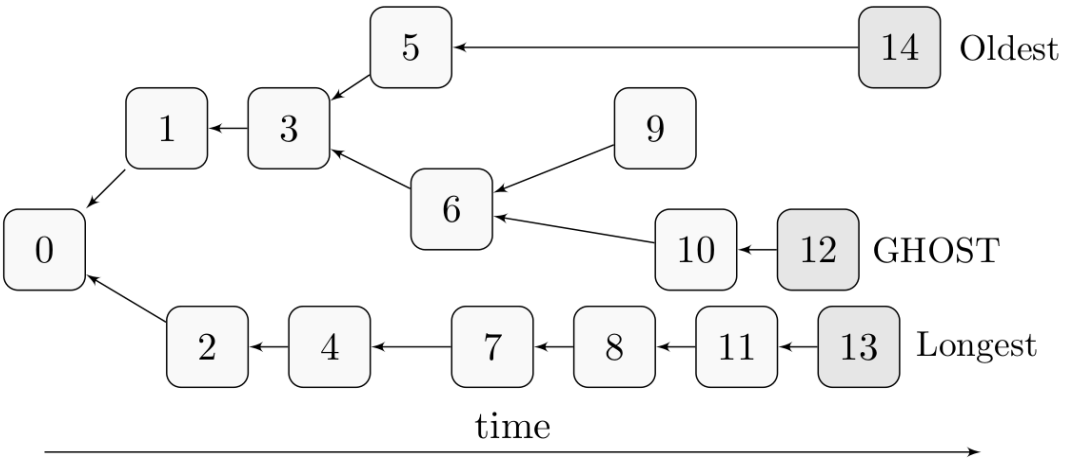

Figure 3. Consensus rules

Additionally, more complex structures of blocks may be considered, whose consensus protocols depend on their geometries and the properties we want the chain to acquire. Lately, directed acyclic graphs have been adopted in place of pure chains of blocks, whose comparison has been deeply studied [46]. These solutions are typically faster in reaching consensus, but their relative security is debated.

4. Known attacks

In this section we survey the most noticeable attacks on blockchains with PoW-based mechanisms of consensus. Most of them actually apply to other consensus protocols, but we do not discuss them here. We also do not consider weak cryptographic primitives, which clearly leave poorly designed cryptosystems open to several types of attacks. However, it is worth pointing out that this assumption is not always satisfied in practice, developers are more and more incentivized to choose high throughput over security. This attitude sometimes translates to deprecated cryptographic functions, of which Curl hash function [47] designed by and for IOTA is a fair representative.

- **51% attacks**

These attacks, already discussed in the BTC original paper [12], are mounted by a user that can (at least temporarily) control more than half of the network's computing power. Such an attacker would be able to double-spend coins by reversing confirmed transactions, to prevent certain transactions from gaining confirmations and to halt payments between some users. This attack, which occurred in 2018 both for BTC Gold and ZenCash, has been widely studied [48] showing that Nakamoto's security analysis may have been too conservative [49]. Variants of the 51% attack such as Block Discarding Attack and Difficulty Raising Attack [50] might succeed even by controlling a smaller portion of the network.

- **Selfish Mining attacks**

With this term we refer to attacks based on strategies that miners can play in order to statistically increase their block creation ratio, hence their revenues. In its first formulation [51] it was shown how miners can constitute selfish pools whose gain, acting co-ordinately and with more than a certain threshold number of users, rises super-linearly with pool size. There are also other attack strategies that might be employed by miners, among which we report (Trail) Stubborn Mining [52], Block Withholding and Fork After Withholding [53]. They have been recently discussed and compared [54, 55, 56], highlighting the system parameters in which each of them excels.

- **Network-level attacks**

This class of attacks aims at exploiting the peer-to-peer network employed by a given cryptocurrency in order to break the perfect information assumption: the nodes cannot legitimately observe the PoW performed by their peers. In an Eclipse Attack [57] the attacker monopolizes the victim's incoming and outcoming connections, filtering its view of the public blockchain. Instead, by transiently disrupting communications between subgroups of similar mining power a vicious miner can delay block propagation, increasing the amount of wasted blocks from a subgroup, which is called Balance Attack [58]. It is also possible to delay blocks by manipulating routing advertisements, usually referred to as Routing Attack [59].

- **Pool related attacks**

The consequences of pools formation, where members aggregate their power and share the rewards, have been widely studied [60], leading to many attempts to exploit this natural aggregation. Some pool awarding schemes allow malicious miners to generate more revenue than a naive miner of the same computational power by strategically spreading their shares across multiple accounts [61]. Pernicious users can also address communication protocols used in blockchains, such as Stratum [62], attacking privacy, security and safety of mining equipment owners. Furthermore, the risk of sabotages between different open pools is effective [63], which may translate to lower earnings for miners, consequently rendering the system unattractive.

- **Goldfinger attacks**

This family [64] comprises the attacks whose main goal is to undermine a given cryptocurrency system, even if this does not directly produce an economic benefit for the attacker. The motivations behind such an attack range from political to ideological reasons, or they may be of economic interest by exploiting short market positions or by eliminating crypto-competitors. These outbreaks may be realized by obtaining a significant percentage of the decision-making power of the network, which may be obtained by renting, bribing or buying other miner's computational power (called Hostile Takeover Attack [65]). One can also destabilize the network consensus protocol with DoS-attacks to single users or the whole mining pools [66, 67].

Other types of attacks have been studied lately, exploiting new features of modern blockchains, such as smart contracts. As an example, double-spending attacks capitalizing on insufficient blocks verification or inconsistent execution of state machines have been proposed [68] for certain cryptocurrencies.

5. Conclusion and further research

We have surveyed the development of the principal PoW-techniques, their current use in cryptocurrencies consensus algorithms and the attacks that have been conceived (and mounted) against such protocols in the last decades. We believe that the amount of work put in a PoW at a global scale, such as in Bitcoin, cannot be overcome by attackers, however smart they are, as long the cryptographic and mathematical assumptions underlying the algorithm are sound. We see with high interest the rise of alternative PoW-algorithms, that promise to simultaneously give security to the network and produce mathematically interesting statements [69], while we feel that any attempt to reduce the amount of computational power (used in PoW) cannot leave the security of the system unaffected. The following table summarizes our findings:

Table 1: Comparison of PoW-methods employed in blockchains

	Hashcash	MicroMint	Data Handling	Research-related
Energy Consumption	Very high	High	**Low**	Very high
External Usefulness	Low	Fair	Low	**High**
Security	**Very high**	**Very high**	Fair	High
Adoption	**Large**	Small	Small	Small

As comprehensive as we have tried to be, new consensus algorithms have been recently developed, more are coming in these days, and they all would deserve further investigation. A comparison of these new schemes and the techniques mentioned in this paper surely deserves to be carried on, comparing the respective efficiency, consumes and scalability, as we plan to do in future papers.

Acknowledgments

The results presented here have been carried on within the EU-ESF activities, call "PON Ricerca e Innovazione 2014-2020", project "Distributed Ledgers for Secure Open Communities". We thank the Quadrans Foundation for its support.

References

[1] M. Jakobsson, A. Juels, "Proofs of Work and Bread Pudding Protocols (Extended Abstract)", Secure Information Networks, 1999, Available: https://doi.org/10.1007/978-0-387-35568-9_18

[2] C. Dwork, M. Naor, "Pricing via Processing or Combatting Junk Mail", Annual International Cryptology Conference, 1992, Available: https://doi.org/10.1007/3-540-48071-4_10

[3] J. Cai, R. J. Lipton, R. Sedgewick, A. C. Yao, "Towards uncheatable benchmarks", IEEE, 1993, Available: https://doi.org/10.1109/SCT.1993.336546

[4] S. Ar, J. Cai, "Reliable benchmarks Using Numerical Instability", Proceeding SODA '94, 1994, Available: https://dl.acm.org/citation.cfm?id=314476

[5] M. K. Franklin, D. Malkhi, "Auditable Metering with Lightweight Security", Financial Cryptography, 1997, Available: https://doi.org/10.1007/3-540-63594-7_75
[6] D. M. Goldschlag, S. G. Stubblebine, "Publicly Verifiable Lotteries: Applications of Delaying Functions", Financial Cryptography, 1994, Available: http://dl.acm.org/citation.cfm?id=647502.728319
[7] R. L. Rivest, A. Shamir, D. A. Wagner, "Time-lock Puzzles and Timed-release Crypto", Massachusetts Institute of Technology Cambridge, 1996, Available: https://dl.acm.org/citation.cfm?id=888615
[8] A. Juels, J. Brainard, "Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks", Network and Distributed System Security Symposium, 1999, Available: https://www.ndss-symposium.org/ndss1999/cryptographic-defense-against-connection-depletion-attacks
[9] B. Laurie, R. Clayton, ""Proof-of-Work" Proves Not to Work", 2004, Available: https://www.cl.cam.ac.uk/~rnc1/proofwork.pdf
[10] P. Gardner-Stephen, "Escalating The War On SPAM Through Practical PoW Exchange", IEEE, 2007, Available: https://doi.org/10.1109/ICON.2007.4444132
[11] D. Liu, L. J. Camp, "When Proof of Work Works", SSRN, 2006, Available: http://dx.doi.org/10.2139/ssrn.941190
[12] S. Nakamoto, Bitcoin: "A Peer-to-Peer Electronic Cash System", 2008, Available: https://bitcoin.org/bitcoin.pdf
[13] A. Back, "Hashcash", 1997, Available: http://www.cypherspace.org/hashcash
[14] A. Back, "Hashcash - A Denial of Service Counter-Measure", 2002, Available: http://www.hashcash.org/papers/hashcash.pdf
[15] W. Penard, T. van Werkhoven, "On the Secure Hash Algorithm family", Cryptography in Context, Chapter 1, 2007, Available: https://www.staff.science.uu.nl/~tel00101/liter/Books/CrypCont.pdf
[16] Ethereum team, "Ethash", 2018, Available: https://github.com/ethereum/wiki/wiki/Ethash
[17] G. Bertoni, J. Daemen, M. Peeters, G. van Assche, R. van Keer, "Keccak implementation overview", 2012, Available: https://keccak.team/files/Keccak-implementation-3.2.pdf
[18] T. Dryja, "Hashimoto: I/O bound proof of work", 2017, Available: http://diyhpl.us/~bryan/papers2/bitcoin/meh/hashimoto.pdf
[19] V. Buterin, "Dagger: A Memory-Hard to Compute, Memory-Easy to Verify Scrypt Alternative", 2013, Available: http://www.hashcash.org/papers/dagger.html
[20] C. Percival, "Stronger Key Derivation via Sequential Memory-Hard Functions", 2009, Available: http://www.tarsnap.com/scrypt/scrypt.pdf
[21] N. van Saberhagen, "Cryptonote v 2.0", 2013, Available: https://cryptonote.org/whitepaper.pdf
[22] A. Biryukov, D. Khovratovich, "Equihash: Asymmetric Proof-of-Work Based on the Generalized Birthday Problem", Ledger, 2017, Available: https://doi.org/10.5195/ledger.2017.48
[23] E. Duffield, E.Diaz, "Dash: A Payments-Focused Cryptocurrency", 2018, Available: https://github.com/dashpay/dash/wiki/Whitepaper
[24] A. de Vries, "Bitcoin's Growing Energy Problem", Joule, 2018, Available: http://doi.org/10.1016/j.joule.2018.04.016
[25] Blockchain.com, https://www.blockchain.com/charts/hash-rate
[26] R. L. Rivest, A. Shamir, "PayWord and MicroMint: Two simple micropayment schemes", International Workshop on Security Protocols, 1997, Available: http://doi.org/10.1007/3-540-62494-5_6
[27] H. Finney, "RPOW - Reusable Proofs of Work", 2004, Available: https://nakamotoinstitute.org/finney/rpow/index.html
[28] S. Dziembowski, S. Faust, V. Kolmogorov, K. Pietrzak, "Proofs of Space", IACR, 2013, Available: https://eprint.iacr.org/2013/796.pdf
[29] D. Perito, G. Tsudik, "Secure Code Update for Embedded Devices via Proofs of Secure Erasure", IACR, 2010, Available: https://eprint.iacr.org/2010/217.pdf
[30] P4Titan, "Slimcoin, A Peer-to-Peer Crypto-Currency with Proof-of-Burn", 2014, Available: https://github.com/slimcoin-project/slimcoin-project.github.io/blob/master/whitepaperSLM.pdf
[31] Sawtooth team, "PoET 1.0 Specification", 2016, Available: https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/poet.html
[32] A. Miller, A. Juels, E. Shi, J. Katz, "Permacoin: Repurposing Bitcoin Work for Data Preservation", IEEE, 2014, Available: https://www.microsoft.com/en-us/research/publication/permacoin
[33] A. Juels, B. S. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files", ACM, 2007, Available: http://doi.acm.org/10.1145/1315245.1315317
[34] K. Sunny, "Primecoin: Cryptocurrency with Prime Number Proof-of-Work", 2013, Available: http://primecoin.io/bin/primecoin-paper.pdf

[35] P. A. Clement, "Congruences for Sets of Primes", The American Mathematical Monthly, 1949, Available: https://www.jstor.org/stable/2305816
[36] A. Shoker, "Sustainable Blockchain through Proof of eXercise", IEEE, 2017, Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8171383
[37] CureCoin Team, "2019 Curecoin Model (White Paper draft)", 2019, Available: https://curecoin.net/white-paper
[38] M. Ball, A. Rosen, M. Sabin, P. N. Vasudevan, "Proofs of Useful Work", IACR, 2017, Available: https://eprint.iacr.org/2017/203.pdf
[39] M. Hastings, N. Heninger, E. Wustrow, "The Proof is in the Pudding: Proofs of Work for Solving Discrete Logarithms", IACR, 2018, Available: https://eprint.iacr.org/2018/939.pdf
[40] M. Lochter, "Blockchain as cryptanalytic tool", IACR, 2018, Available: https://eprint.iacr.org/2018/893.pdf
[41] L. M. Bach, B. Mihaljević, M. Žagar, "Comparative Analysis of Blockchain Consensus Algorithms", MIPRO, 2018, Available: https://ieeexplore.ieee.org/document/8400278
[42] N. Giang-Truong, K. Kyungbaek, "A Survey about Consensus Algorithms Used in Blockchain", IACR, 2018, Available: http://jips-k.org/file/down?pn=530
[43] A. M. Wahab, W. Mahmood, "Survey of Consensus Protocols", 2018, Available: https://www.researchgate.net/publication/328160285_Survey_of_Consensus_Protocols
[44] W. Wang, H. Dinh Thai, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, D. In Kim, "A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks", 2019, Available: http://doi.org/10.1109/ACCESS.2019.2896108
[45] Y. Sompolinsky1, A. Zohar, "Secure High-Rate Transaction Processing in Bitcoin", IACR, 2013, Available: https://eprint.iacr.org/2013/881.pdf
[46] F. M. Benčić, I. P. Žarko, "Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph", IEEE, 2018, Available: https://ieeexplore.ieee.org/document/8416434
[47] E. Heilman, N. Narula, T. Dryja, M. Virza, "IOTA Vulnerability Report: Cryptanalysis of the Curl Hash Function Enabling Practical Signature Forgery Attacks on the IOTA Cryptocurrency", 2017, Available: https://github.com/mit-dci/tangled-curl/blob/master/vuln-iota.md
[48] J. Jang, H. Lee, "Profitable Double-Spending Attacks", ArXiv, 2019, Available: https://arxiv.org/abs/1903.01711
[49] C. Grunspan, R. Pérez-Marco, "Double spend races", ArXiv, 2017, Available: https://arxiv.org/abs/1702.02867
[50] L. Bahack, "Theoretical Bitcoin Attacks with less than Half of the Computational Power (draft)", ArXiv, 2013, Available: https://arxiv.org/abs/1312.7013
[51] I. Eyal, E. G. Sirer, "Majority is not Enough: Bitcoin Mining is Vulnerable", 2013, Available: https://www.cs.cornell.edu/~ie53/publications/btcProcFC.pdf
[52] K. Nayak, S. Kumar, A. Miller, E. Shi, "Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack", IACR, 2015, Available: https://eprint.iacr.org/2015/796.pdf
[53] Y. Kwon, D. Kim, Y. Son, E. Vasserman, Y. Kim, "Be Selfish and Avoid Dilemmas: Fork After Withholding (FAW) Attacks on Bitcoin", ArXiv, 2017, Available: https://arxiv.org/abs/1708.09790
[54] C. Grunspan, R. Pérez-Marco, "On profitability of selfish mining", ArXiv, 2018, Available: https://arxiv.org/abs/1805.08281
[55] C. Grunspan, R. Pérez-Marco, "On profitability of stubborn mining", ArXiv, 2018, Available: https://arxiv.org/abs/1808.01041
[56] C. Grunspan, R. Pérez-Marco, "On Profitability of Trailing Mining", ArXiv, 2018, Available: https://arxiv.org/abs/1811.09322
[57] E. Heilman, A. Kendler, A. Zohar, S. Goldberg, "Eclipse Attacks on Bitcoin's Peer-to-Peer Network", IACR, 2015, Available: https://eprint.iacr.org/2015/263.pdf
[58] C. Natoli, V. Gramoli, "The Balance Attack Against Proof-Of-Work Blockchains: The R3 Testbed as an Example", ArXiv, 2016, Available: https://arxiv.org/abs/1612.09426
[59] M. Apostolaki, A. Zohar, L. Vanbever, "Hijacking Bitcoin: Routing Attacks on Cryptocurrencies", ArXiv, 2016, Available: https://arxiv.org/abs/1605.07524
[60] P. Daian, I. Eyal, A. Juels, E. G. Sirer, "(Short Paper): PieceWork: Generalized Outsourcing Control for Proofs of Work", Financial Cryptography, 2017, Available: https://fc17.ifca.ai/bitcoin/papers/bitcoin17-final24.pdf
[61] J. Holland, R. J. Connor, J. P. Diamond, J. M. Smith, M. Schuchard, "Not So Predictable Mining Pools: Attacking Solo Mining Pools by Bagging Blocksand Conning Competitors", Financial Cryptography, 2018, Available: https://fc18.ifca.ai/preproceedings/79.pdf
[62] R. Recabarren, B. Carbunar, "Hardening Stratum, the Bitcoin Pool Mining Protocol", ArXiv, 2017, Available: https://arxiv.org/abs/1703.06545

[63] I. Eyal, "The Miner's Dilemma", ArXiv, 2014, Available: https://arxiv.org/abs/1411.7099
[64] J. A. Kroll, I. C. Davey, E. W. Felten, "The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries", 2013, Available: https://www.econinfosec.org/archive/weis2013/papers/KrollDaveyFeltenWEIS2013.pdf
[65] M. Vasek, M. Thornton, T. Moore, "Empirical Analysis of Denial-of-Service Attacks in the Bitcoin Ecosystem", Financial Cryptography, 2014, Available: https://fc14.ifca.ai/bitcoin/papers/bitcoin14_submission_17.pdf
[66] B. Johnson, A. Laszka, J. Grossklags, M. Vasek, T. Moore, "Game-Theoretic Analysis of DDoS Attacks Against Bitcoin Mining Pools", Financial Cryptography, 2014, Available: https://fc14.ifca.ai/bitcoin/papers/bitcoin14_submission_16.pdf
[67] M. Vasek, M. Thornton, T. Moore, "Empirical Analysis of Denial-of-Service Attacks in the Bitcoin Ecosystem", Financial Cryptography, 2014, Available: https://fc14.ifca.ai/bitcoin/papers/bitcoin14_submission_17.pdf
[68] Z. Peng, Y. Chen, "All roads lead to Rome: Many ways to double spend your cryptocurrency", ArXiv, 2018, Available: https://arxiv.org/abs/1811.06751
[69] A. Meneghetti, M. Sala, D. Taufer, "A new ECDLP-based PoW model", ArXiv 2019, Available: https://arxiv.org/abs/1911.11287