mathematics

MDPI

Article

A New ECDLP-Based PoW Model †

Alessio Meneghetti, Massimiliano Sala and Daniele Taufer *

Department of Mathematics, University of Trento, Via Sommarive 14, 38123 Povo (TN), Italy; Alessio.Meneghetti@unitn.it (A.M.); maxsalacodes@gmail.com (M.S.)

* Correspondence: daniele.taufer@gmail.com

† This paper is an extended version of our paper published in the Proceedings of the 3rd Distributed Ledger Technology Workshop (DLT 2020), Ancona, Italy, 4 February 2020.

Received: 9 July 2020; Accepted: 9 August 2020; Published: 12 August 2020

Abstract: Blockchain technology has attracted a lot of research interest in the last few years. Originally, their consensus algorithm was Hashcash, which is an instance of the so-called Proof-of-Work. Nowadays, there are several competing consensus algorithms, not necessarily PoW. In this paper, we propose an alternative proof of work algorithm which is based on the solution of consecutive discrete logarithm problems over the point group of elliptic curves. At the same time, we sketch a blockchain scheme, whose consensus is reached via our algorithm. In the considered architecture, the curves are pseudorandomly determined by block creators, chosen to be cryptographically secure and changed every epoch. Given the current state of the chain and a prescribed set of transactions, the curve selection is fully rigid, therefore trust is needed neither in miners nor in the scheme proposers.

Keywords: proof of work (PoW); elliptic curve cryptography (ECC); elliptic curve discrete logarithm problem (ECDLP); blockchain; epoch; provable security

1. Introduction

A proof of work (PoW) is a procedure that allows a prover to demonstrate that he is very likely to have performed a specific amount of computational work within a prescribed interval of time [1].

This concept has been formalized in 1999 [2], although previous instances of delaying functions conceived for similar purposes had appeared earlier [3–9].

Since 2008, PoW-methods have been attracting a considerable interest as Bitcoin [10] introduced a PoW-based consensus algorithm, which puts miners in competition for solving a cryptographic challenge. Bitcoin's consensus relies on a hashcash system [11,12], whose workload may be easily adjusted with a fastly verifiable output. Despite their high efficiency and easy implementation, all the hashcash-based protocols share a common limitation: the huge amount of computations employed by nodes becomes useless after the consensus is reached. This aspect has been raising environmental concerns and many solutions have been proposed to reduce these energy-intensive computer calculations.

A promising countermeasure to this issue is the adoption of *bread pudding protocols* [2]. They face the aforementioned problem by performing a computational work that is reusable either for practical [13–16], cryptographical [2,17], or mathematical [18] reasons. Moreover, the latter class of systems encloses several protocols that are meant to be research propellants [19], namely designed to boost the commitment upon the solution of difficult mathematical problems. A notable example is PrimeCoin [18], which appeared in 2013 and is a cryptocurrency whose Proof-of-Work consensus is based on searching for prime numbers.

Along the same line, we have proposed [20], a blockchain architecture with a PoW-consensus algorithm based on the solution of the *Discrete Logarithm Problem* over the point groups of elliptic curves

(ECDLP). The idea of basing a PoW on ECDLP has already appeared in other works [21,22], as this problem is widely studied and applied in cryptographic protocols. In particular, the PoW consensus algorithm proposed in [21] is based on solving the ECDLP on a *fixed* elliptic curve. Although we do appreciate their approach for the novelty of using ellptic curves in their PoW constructions, the choice of a specific fixed curve may look suspicious. In fact, in these previous approaches, the considered curves do not usually fulfill the standard security criteria [23], especially for what concerns the *fully rigidity*: the network has to initially trust an authority that is providing the curve parameters.

In this work, we radically solve this issue by designing a PoW-system based on elliptic curves that are changing over the time. Since the curves are pseudo-randomly constructed and satisfy general security conditions, a malicious user wanting to attack the chain shall efficiently break the ECDLP over an immense class of elliptic curves, a task that may be fairly considered infeasible. We provide the aforementioned scheme [20] with precise mathematical foundations and further implementation details. Concrete choices of the involved parameters are exhibited, and the security of the system is debated in a formal manner.

The paper is organized as follows: a quick summary of the ECDLP is given in Section 2, while the main blockchain architecture is proposed in Section 3, whose block construction is detailed in Sections 3.1 and 3.2. The strong points of this system are discussed in Section 4, where its security is also debated, while, in Section 5, future work directions are presented.

2. ECDLP

The ECDLP is a renowned problem that consists, given two points P and Q of an elliptic curve E, of finding (if existent) an integer $N \in \mathbb{N}$ such that the N-th multiple of P (usually called the *base point*) equals Q, i.e., $Q = N \cdot P$.

Here, we are only interested in elliptic curves over prime fields $\mathbb{F}_p$ and determined by their short Weierstrass equation $y^2 = x^3 + Ax + B$. Solving ECDLP on curves E over large prime fields is considered to be a difficult challenge except for degenerate cases.

2.1. The General Case

The most efficient currently known *general* attacks are Baby-Step Giant-Step [24] and Pollard's Rho—Kangaroo algorithms [25], which have an asymptotic complexity of $O(\sqrt{|E|})$, where $|E|$ is the size of E. These are general parallel collision-finding algorithms, which work over any groups, i.e., no properties of the underlying structure, but the operation definitions are used.

The introduction of Semaev's polynomials [26] has suggested the existence of subexponential algorithms to solve ECDLP; however, no clear evidence has emerged. Pairings-based attacks [27,28], Index calculus [29–31], and Xedni calculus [32] have been recently being studied, but none of them seem to significantly reduce the problem complexity of the general case so far.

2.2. Special Cases

There are some families of curves whose ECDLP is known to be easier than the general case, as there are algorithms for efficiently solving it. Consequently, such curves have to be carefully avoided for designing a ECDLP-based protocol. A concise summary of those particular attacks, the curve on which they may be efficiently applied, and how we avoid them are drawn in Table 1.

Table 1. Attacks on ECDLP over special curves

Attack	It Applies on Curves	To Avoid It: Use	Ref.
Weil-descent	over composite fields	prime fields	[33,34]
Pohlig–Hellman	of composite orders	prime orders	[35]
Semaev, Satoh–Araki, Smart	anomalous	non-anomalous curves	[36–38]
Menezes–Okamoto–Vanstone	low embedding degree	high embedding degree	[28]
Frey–Rück	low embedding degree	high embedding degree	[27]
Wiener–Zuccherato	low CM discriminant	high CM discriminant	[39]

3. A Sample Blockchain Architecture

To show how our PoW-proposal works, we introduce a schematic sample ledger architecture, but the same idea may easily be adapted for other similar architectures. The consensus of our ledger is based on solving ECDLP on non-suspicious curves, hence it needs to address two radically-different closely-linked tasks: finding a strong pseudo-random curve and producing generic instances of ECDLP on it. Therefore, we have chosen a blockchain scheme based on two types of blocks, the standard ones and those defining the involved parameters. Other approaches are still viable, provided they solve the two aforementioned connected tasks.

[EB] An *Epoch Block* contains, aside from the header and a list of transactions, a prime number p, an elliptic curve E defined over $\mathbb{F}_p$, and a base point P of E, all to be determined by the proposing miner via rigid algorithms with a pseudorandom set of input parameters.

Moreover, it encloses as PoW a pair of integers (N_1, N_2) to be discovered by the proposing miner, such that $N_i \cdot P$ are points of E deterministically determined from the block contents and the current chain status.

The frequency of the EBs determines the number of changes in the ECDLP-parameters with respect to the network block construction speed. In the current proposal, we arbitrarily set these EBs to occur once every 2016 blocks, emulating the bitcoin difficulty adjustment rate.

[SB] The *Standard Blocks* are just a light version of the EB blocks; they are constructed in the same way except for p, E, P, which are inherited from the last EB block of the chain.

SBs constitute the vast majority of the blocks of the chain, as portrayed in Figure 1.

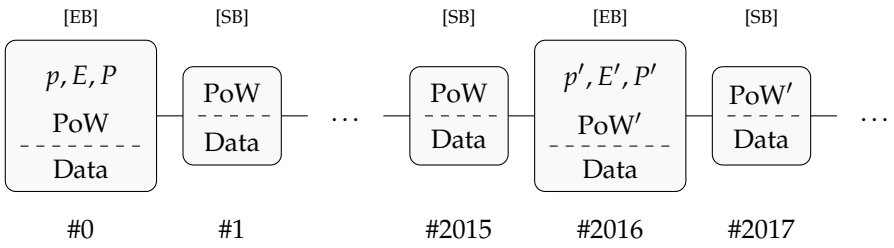

Figure 1. Sample blockchain model.

EBs basically define the setting (curves and base points) on which the discrete logarithm PoWs will have to be solved in the following epoch. They are slightly heavier to be produced and verified but occur rarely (roughly once every two weeks with the proposed frequency and assuming a BTC-like network computational power).

In order to give the specifications of our blocks, we need a deterministic function `P_Gen`, used for constructing a point on a given elliptic curve E from a prescribed hash digest h, which we treat as an integer for simplicity. The following is a concrete example of such a function:

```
function P_Gen(h,E)
i = 0
```

```
while #{points of E with x-coord = h + i} = 0:
i = i + 1
P = (h + i, *) point of E with 0 ≤ * < p/2
return P
```

We notice that the points determined by the above function are affine by construction. The hash $\mathcal{H}$ that we propose to use in the following is SHA3-512 [40], which provides a satisfying collision resistance even against post-quantum attacks, but one might conceivably replace it with another properly constructed one.

We also assume that all proposing miners use prescribed signature algorithms and we denote with $\sigma_k(m)$ the signature of the string m obtained by the miner with signing key k.

3.1. Standard Blocks

A minimal model of a SB consists of a list of valid transactions and a header, which comprises their Merkel root $\mathcal{M}$, the hash of the previous header h_{prev} and a pair of integers (N_1, N_2) solving

$$\text{PoW}: \quad \begin{cases} \texttt{P\_Gen}(\mathcal{H}(h_{\text{prev}}), E) = N_1 \cdot P, \\ \texttt{P\_Gen}(\mathcal{H}(\mathcal{M}), E) = N_2 \cdot P, \end{cases}$$

where E and P have been defined in the last EB. A model of the proposed SB is given in Figure 2.

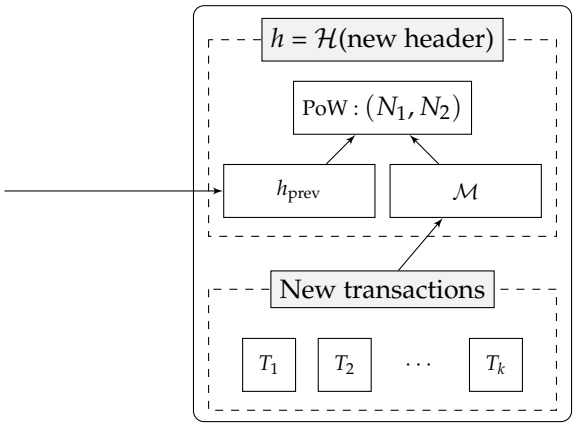

Figure 2. A Standard Block [SB] prototype.

3.2. Epoch Blocks

An EB is a thick version of a SB, namely it is constructed in a similar fashion, but it encodes three additional pieces of data: the prime p, the elliptic curve E over $\mathbb{F}_p$, and the base point P of E.

- Generating p

  The prime number p is the responsibility of the expected runtime of the PoW. Its size is determined by the difficulty parameter d, whose tuning depends on the block production ratio that a designer wants to obtain. Therefore, we do not discuss the choice of d, but we refer to the BTC implementation [41] or to more structured models such as personalized difficulty adjustments [42]. Our goal is to produce a prime number of the prescribed size and which does not allow fast arithmetic for the known cases, i.e. we pick it satisfying the following properties.

  Exceptionality properties

  1. p is not a Crandall prime [43], i.e., not of the from $2^k - c$ for a relatively small and positive integer c.
  2. p is neither a Generalized Mersenne prime [44] nor a More Generalized Mersenne prime [45], i.e., it may not be written as $p(m)$ for some integer m and polynomial p with very small coefficients and number of monomials.

3. p is not Montgomery-friendly [46–48], i.e., it may not be obtained as $2^{\alpha}(2^{\beta} - \gamma) - 1$ for small positive integers α, β, γ.

Given the difficulty parameter d and the hash of the previous header h, we propose the generation of such a prime number p as follows.

```
function p_Gen(d, h)
repeat
h = ℋ(h)
p = Next,Prime(h mod 2^{2d})
until p satisfies exceptionality properties
return p
```

- Generating E

We aim at generating pseudorandom elliptic curves for which no efficient attacks are currently known, i.e., satisfying the following properties:

Security properties

1. The number of points of E is prime and different from p.
2. The *embedding degree* B is greater than 20, i.e., $|E| \nmid p^B - 1$ for every $1 \leq B \leq 20$.
3. Let D be the *CM field discriminant*, defined as

$$D = \begin{cases} \Delta & \text{if } \Delta \equiv 1 \bmod 4, \\ 4\Delta & \text{otherwise,} \end{cases} \qquad \Delta = \text{SquareFreePart}(t^2 - 4p),$$

where t is the trace of E. Then, we require $D > 2^{40}$.

Let h be the previous block header; we suggest generating the curve as follows:

```
function E_Gen(p, h)
i = 0
repeat
i = i + 1
A_E = ℋ(h + i)
B_E = ℋ(A_E)
E defined by y^2 = x^3 + A_E x + B_E over 𝔽_p
until E is an EC satisfying security properties
return E
```

- Generating P

The base point we prescribe for an EB and its subsequent epoch is

$$P = \texttt{P\_Gen}(\mathcal{H}(p \,||\, A_E \,||\, B_E), E),$$

where $x||y$ denotes the concatenation of the bit representations of x and y.

We remark that the new epoch parameters are manufactured before the PoW production, which therefore depends on them. A schematic epoch block is sketched in Figure 3.

Despite the verification of SBs being extremely fast, EBs are slower to be checked since verifiers need to test that all the curve parameters involved have been properly constructed, running several types of mathematical algorithms such as primality testing, finite fields operations, and elliptic curve points counting.

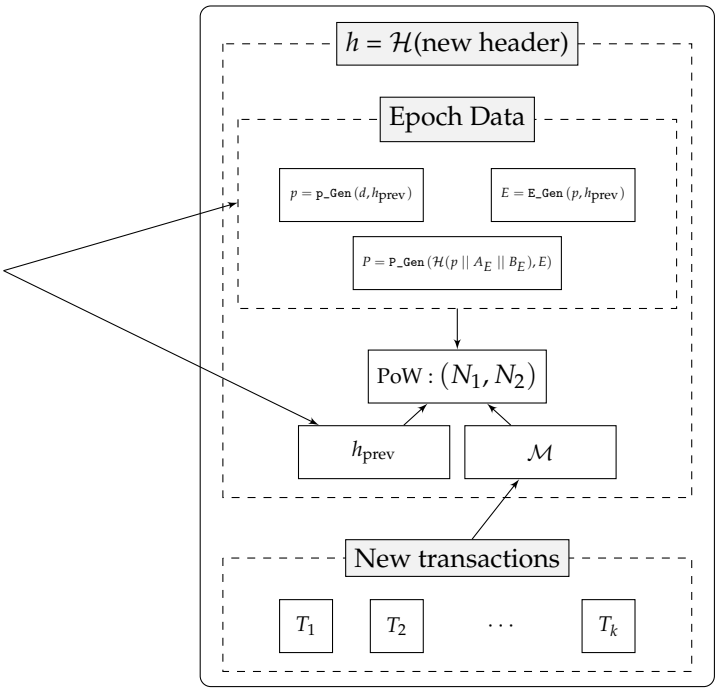

Figure 3. A Epoch Block [EB] prototype.

4. Method Discussion

Here, we discuss motivation and advantages of the presented choices.

First, this PoW model involves many different mathematical algorithms of wide interest, for which this blockchain may represent a concrete research propellant. Furthermore, it might also provide a public collection of cryptographically secure elliptic curves of moderate size.

Apart from its scientific usefulness, it conveys many desirable security properties. The challenges involved do not rely on a given curve of questionable provenance but on the *generic* difficulty of the ECDLP, which is much more fair to be trusted. Thus, we find it aims at embracing the decentralization ideals that lead to cryptocurrencies creation: even the mathematical objects involved are publicly manufactured, and no trust is required even in the authors or the proposing entities. For example, in the Bitcoin case, everybody must trust the strength of the given curve Secp256k1.

The existence of different types of blocks in blockchains has become common, as it is considered suitable for tackling the problem of scalability [49]. Here, we pursue this idea by designing a scheme with two different types of blocks, but possibly more sophisticated solutions may be taken under consideration for obtaining further worthwhile properties.

As for blocks forgery, we point out that both SBs and EBs comprise a PoW which depends on the entire block, together with the previous one. This means that any counterfeit in any position of the chain results into an incorrect final block, which may be easily detected from the network.

Moreover, it is hard to conceive shortcuts for the PoW production: for a given difficulty parameter d, we expect a d-bits secutity of the *general* ECDPL by using $p \approx 2^{2d}$, unless attacks outperforming Pollard's rho are discovered. Moreover, common base field operations speed ups are avoided by making use of not-exceptional primes, ensuring a fair and general problem to be solved equally for every miner. In fact, neither specific algorithms nor dedicated hardware may be used for solving such a general problem, of which easy cases are carefully avoided. In addition, the constructed curves fulfill the known security criteria [23]:

- working over prime fields avoids Weil-descent attacks;
- searching for curves of prime order prevents from Pohlig–Hellman attacks;
- since $p \neq |E|$ the curves are not anomalous so Smart, Semaev, Satoh–Araki attacks do not apply;

- the embedding degree we suggest is greater than 20 as required by SEC1 [50], which prevents pairing attacks such as Menezes–Okamoto–Vanstone (based on Weil Pairing) and Frey–Rück (based on Tate–Lichtenbaum Pairing);
- attacks to curves with low CM discriminant are prevented by requiring it higher than 2^{40}, as for Brainpool Standard Curves [51].

The following theorem shows that, whatever the difficulty of the considered ECDLP is, when the cryptographic primitives are well-constructed, solving an instance of this problem is inescapable for proposing a new valid block.

Theorem 1. *Let us assume that the current epoch is endowed with the curve E and its base-point P, and let h_{prev} be the hash of the previous header's block. Let σ be a uniform and deterministic digital signature algorithm and $\mathbb{M}$ be a proposing miner with fixed signing key k. If $\mathcal{H}$ is uniform and $\mathbb{M}$ exhibits a valid block, then $\mathbb{M}$ has solved at least one generic instance of ECDLP on E.*

Proof. By definition of our PoW, the given block is valid if and only if it contains (N_1, N_2) such that

$$\begin{cases} Q_1 = \texttt{P\_Gen}(\mathcal{H}(\sigma_k(h_{\text{prev}})), E) = N_1 \cdot P, \\ Q_2 = \texttt{P\_Gen}(\mathcal{H}(\mathcal{M}), E) = N_2 \cdot P. \end{cases}$$

Since E and P are determined by the epoch and h_{prev} is determined by the previous block, the proposing miner has no control on them. Moreover, σ is deterministic and k is fixed, so that $Q_1 = \texttt{P\_Gen}(\mathcal{H}(\sigma_k(h_{\text{prev}})), E)$ cannot be influenced by the miner. Therefore, if $\mathbb{M}$ has proposed a valid block, it must have solved $Q_1 = N_1 \cdot P$. Since both σ_k and $\mathcal{H}$ are uniform, then also $\texttt{P\_Gen}(\mathcal{H}(\sigma_k(*)), E)$ is, so the instance of the ECDLP that $\mathbb{M}$ has solved was generic. □

We notice that the hypotheses of Theorem 1 are satisfied whenever σ and $\mathcal{H}$ are cryptographically well-designed, e.g., in a random oracle model [52]. Moreover, in such model, a solution of the equation $Q_2 = N_2 \cdot P$, which ensures transactions integrity but is not directly employed in the previous proof, may only be discovered by solving another generic instance of ECDLP. Indeed, if $\mathcal{H}$ behaves as a random oracle, then finding a pair of integers (X, Y) that solves

$$\texttt{P\_Gen}(\mathcal{H}(X), E) = Y \cdot P$$

is as difficult as computing a generic discrete logarithm on E. However, the same argument might hold under much weaker assumptions (e.g., $\mathcal{H}$ to be preimage resistant).

Besides security, the curves we propose are *fully rigid* as defined in [23]: their construction is entirely explained in terms of the previous block, which cannot be controlled by a malicious actor since there is no room for miner choices (such as nonces). Even assuming that the transactions of the previous block might be chosen ad hoc, an attacker who wants to impose a particular curve during the next epoch has to brute-force invert the hash $\mathcal{H}$ at the cost of one ECDLP solution for each attempt, until a desired hash digest is obtained, within the time needed for the entire network to solve a single ECDLP. We consider this scenario unachievable under realistic assumptions.

As regards the difference between EBs and SBs, we point out that the bulk of miner's work consists of the ECDLP solution: we expect good parameters to be generated in EBs in a time that is linear in the difficulty parameter [53], whereas the asymptotic difficulty of ECDLP solution is exponential in it. This difference in complexity is concretely presented in Figure 4, where with a Magma [54] test we have compared the time used for generating the parameters p, E, P and the time needed for solving a generic ECDLP on them.

Since the curve creation appears not to be computationally demanding when compared to the actual PoW, then lazy miners do not have any substantial advantage in skipping it.

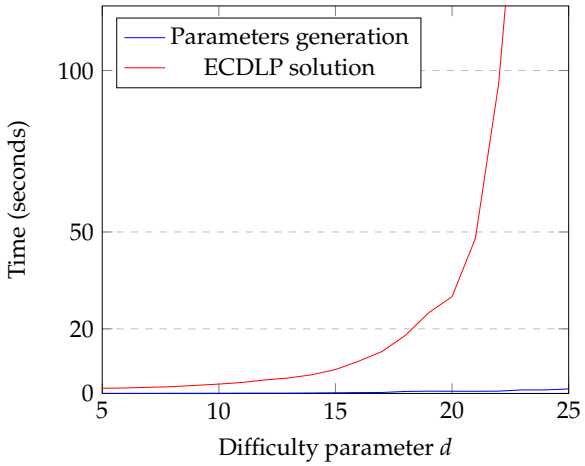

Figure 4. Time comparison between parameters creation and ECDLP solution.

Remark 1. *As regards forks, they may occur if two miners find at nearly the same time a valid solution. This is possible, but unlikely, for two reasons. The first is that each miner is trying to solve a distinct ECDLP mathematical problem (since its input data include the miner's digital signature), so the difficulty of these problems may differ in practice, although they have the same average complexity. The second is that all known efficient algorithms to solve ECDLP on a generic strong curve are randomized algorithms, so that, even if two miners had the same identical initial data, their running time would be different. In any case, if the computational power of the miner network grows significantly before the parameter d is adjusted, then forks will be more likely to occur, exactly as it happens with Bitcoin. In conclusion, we cannot exclude the event of forks, but its probability will be about the same as forks in Bitcoin, and, when they happen, the chain that survives is the one that is supported by the majority of the network computational power (again, as it happens for Bitcoin).*

5. Conclusions

We have proposed a new PoW-based blockchain model based on *general* ECDLP, highlighting the desirable properties that such a scheme provides in terms of scientific relevance, security, and pure decentralization ideals. In particular, even though the scheme employs a well-known underlying mathematical problem, it does not rely on specific choices of curve parameters, so that its security depends on the *average complexity* of the ECDLP solution. This removes possible suspects on the security of a given fixed curve.

It may be interesting to produce an actual implementation of the proposed scheme, obtaining practical time measurements and efficiency considerations. A subsequent engaging project might address the resistance of such a protocol to the known attacks under real-world assumptions, comparing the obtained results with outcomes of existing cryptocurrencies. Other types of curve models may be also considered for such schemes, such as Edwards or Montgomery curves, which may have faster point arithmetic. More generally, every family of groups on which the DLP is considered hard may be exploited to conceive a similarly fair PoW-schemes. The choice of basing our proposal on elliptic curves that are considered secure is motivated by their intrinsic interest and worldwide adoption.

Finally, different types of PoW might be conceived in a similar fashion, possibly employing problems that are thought to be resistant even to quantum attacks.

Author Contributions: Conceptualization, M.S. and D.T.; Investigation, A.M. and D.T.; Methodology, D.T.; Project administration, D.T.; Supervision, M.S.; Validation, A.M.; Writing—original draft, D.T.; Writing—review and editing, A.M., M.S. and D.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Meneghetti, A.; Sala, M.; Taufer, D. A Survey on PoW-Based Consensus. *AETiC* **2020**, *4*, 1. [CrossRef]
2. Jakobsson, M.; Juels, A. Proofs of Work and Bread Pudding Protocols (Extended Abstract). Secure Information Networks. 1999. [CrossRef]
3. Ar, S.; Cai, J. Reliable Benchmarks Using Numerical Instability. SODA '94: Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms. 1994; pp. 34–43. Available online: https://dl.acm.org/citation.cfm?id=314476 (accessed on 3 January 2020).
4. Cai, J.; Lipton, R.J.; Sedgewick, R.; Yao, A.C. Towards Uncheatable Benchmarks. In Proceedings of the Eigth Annual Structure in Complexity Theory Conference, San Diego, CA, USA, 18–21 May 1993. [CrossRef]
5. Dwork, C.; Naor, M. Pricing via Processing or Combatting Junk Mail. In *Annual International Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 1992; Volume 740. [CrossRef]
6. Franklin, M.K.; Malkhi, D. Auditable Metering with Lightweight Security. In *International Conference on Financial Cryptography*; Springer: Berlin/Heidelberg, Germany, 1997; Volume 1318. [CrossRef]
7. Goldschlag, D.M.; Stubblebine, S.G. Publicly Verifiable Lotteries: Applications of Delaying Functions. In *International Conference on Financial Cryptography*; Springer: Berlin/Heidelberg, Germany, 1994. Available online: http://dl.acm.org/citation.cfm?id=647502.728319 (accessed on 5 January 2020).
8. Juels, A.; Brainard, J. Client Puzzles: A Cryptographic Countermeasure against Connection Depletion Attacks. NDSS Symposium 1999. Available online: https://www.ndss-symposium.org/ndss1999/cryptographic-defense-against-connection-depletion-attacks (accessed on 5 January 2020).
9. Rivest, R.L.; Shamir, A.; Wagner, D.A. *Time-Lock Puzzles and Timed-Release Crypto*; Massachusetts Institute of Technology: Cambridge, MA, USA, 1996. Available online: https://dl.acm.org/citation.cfm?id=888615 (accessed on 5 January 2020).
10. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 5 January 2020).
11. Back, A. *Hashcash*. 1997. Available online: http://www.cypherspace.org/hashcash (accessed on 5 January 2020).
12. Back, A. *Hashcash—A Denial of Service Counter-Measure*. 2002. Available online: http://www.hashcash.org/papers/hashcash.pdf (accessed on 5 January 2020).
13. CureCoin Team. *2019 Curecoin Model (White Paper Draft)*. 2019. Available online: https://curecoin.net/white-paper (accessed on 5 January 2020).
14. Finney, H. *RPOW—Reusable Proofs of Work*. 2004. Available online: https://nakamotoinstitute.org/finney/rpow/index.html (accessed on 5 January 2020).
15. Miller, A.; Juels, A.; Shi, E.; Katz, J. Permacoin: Repurposing Bitcoin Work for Data Preservation. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 18–21 May 2014. Available online: https://www.microsoft.com/en-us/research/publication/permacoin-repurposing-bitcoin-work-for-data-preservation (accessed on 5 January 2020).
16. Shoker, A. Sustainable Blockchain through Proof of Exercise. In Proceedings of the 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 30 October–1 November 2017. Available online: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8171383 (accessed on 5 January 2020).
17. Rivest, R.L.; Shamir, A. PayWord and MicroMint: Two Simple Micropayment Schemes. In *International Workshop on Security Protocols*; Springer: Berlin/Heidelberg, Germany, 1997; Volume 1189. [CrossRef]
18. Sunny, K. *Primecoin: Cryptocurrency with Prime Number Proof-of-Work*. 2013. Available online: http://primecoin.io/bin/primecoin-paper.pdf (accessed on 5 January 2020).
19. Ball, M.; Rosen, A.; Sabin, M.; Vasudevan, P.N. Proofs of Useful Work. Cryptology ePrint Archive (IACR). 2017. Available online: https://eprint.iacr.org/2017/203.pdf (accessed on 4 January 2020).

20. Meneghetti, A.; Sala, M.; Taufer, D. A Note on an ECDLP-Based PoW Model. In Proceedings of the 3rd Distributed Ledger Technology Workshop (DLT 2020), Ancona, Italy, 4 February 2020. Available online: http://ceur-ws.org/Vol-2580/ (accessed on 6 February 2020).
21. Hastings, M.; Heninger, N.; Wustrow, E. The Proof Is in the Pudding: Proofs of Work for Solving Discrete Logarithms. Cryptology ePrint Archive (IACR). 2018. Available online: https://eprint.iacr.org/2018/939.pdf (accessed on 5 January 2020).
22. Lochter, M. Blockchain as Cryptanalytic Tool. Cryptology ePrint Archive (IACR). 2018. Available online: https://eprint.iacr.org/2018/893.pdf (accessed on 7 January 2020).
23. Bernstein, D.J.; Lange, T.L. Safecurves: Choosing Safe Curves for Elliptic-Curve Cryptography. Available online: https://safecurves.cr.yp.to/ (accessed on 7 January 2020).
24. Shanks, D. Class Number, a Theory of Factorization and Genera. In *Proceedings of Symposia in Pure Mathematics*; American Mathematical Society: Providence, RI, USA, 1969; Volume 20, pp. 415–440. Available online: http://www.ams.org/books/pspum/020/ (accessed on 7 January 2020).
25. Pollard, J.M. Monte Carlo Methods for Index Computation (mod p). *Math. Comput.* **1978**, *32*, 918–924. [CrossRef]
26. Semaev, I.A. Summation Polynomials and the Discrete Logarithm Problem on Elliptic Curves. Cryptology ePrint Archive (IACR). 2004. Available online: https://eprint.iacr.org/2004/031.pdf (accessed on 7 January 2020).
27. Frey, G.; Rück, H. A Remark Concerning M-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves. *Math. Comput.* **1994**, *62*, 865–874. Available online: https://www.jstor.org/stable/2153546?seq=1 (accessed on 7 January 2020).
28. Menezes, A.J.; Okamoto, T.; Vanstone, S.A. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *IEEE Trans. Inf. Theory* **1993**, *39*, 5. Available online: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=259647 (accessed on 7 January 2020). [CrossRef]
29. Amadori, A.; Pintore, F.; Sala, M. On the Discrete Logarithm Problem for Prime-Field Elliptic Curves. *Finite Fields Their Appl.* **2018**, *51*, 168–182. [CrossRef]
30. McGuire, G.; Mueller, D. A New Index Calculus Algorithm for the Elliptic Curve Discrete Logarithm Problem and Summation Polynomial Evaluation. Cryptology ePrint Archive (IACR). 2017. Available online: https://eprint.iacr.org/2017/1262.pdf (accessed on7th January 2020).
31. Silverman, J.H.; Suzuki, J. Elliptic Curve Discrete Logarithms and the Index Calculus; In *International Conference on the Theory and Application of Cryptology and Information Security*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 110–125. [CrossRef]
32. Silverman, Silverman , J.H. The Xedni Calculus and the Elliptic Curve Discrete Logarithm Problem. *Des. Codes Cryptogr.* **2000**, *20*, 5–40. [CrossRef]
33. Frey, G.; Gangl, H. How to Disguise an Elliptic Curve (Weil Descent). In *Talk at ECC '98*; University of Waterloo: Waterloo, ON, Canada, 1998. Available online: http://www.cacr.math.uwaterloo.ca/conferences/1998/ecc98/frey.ps (accessed on 7 January 2020).
34. Gaudry, P.; Hess, F.; Smart, N.P. Constructive and Destructive Facets of Weil Descent on Elliptic Curves. *J. Cryptol.* **2002**, *15*, 19–46. [CrossRef]
35. Pohlig, S.; Hellman, M. An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic Significance. *IEEE Trans. Inf. Theory* **1978**, *24*, 1. Available online: https://ee.stanford.edu/~hellman/publications/28.pdf (accessed on 7 January 2020). [CrossRef]
36. Satoh, T.; Araki, K. Fermat Quotients and the Polynomial Time Discrete Log Algorithm for Anomalous Elliptic Curves. *Comm. Math. Univ. Sancti Pauli* **1998**, *47*, 81–92.
37. Semaev, I.A. Evaluation of Discrete Logarithms in a Group of p-Torsion Points of an Elliptic Curve in Characteristic p. *Math. Comput.* **1998**, *67*, 353–356. [CrossRef]
38. Smart, N. The Discrete Logarithm on Elliptic Curves of Trace One. *J. Math. Cryptol.* **1999**, *12*, 193–196. [CrossRef]
39. Wiener, M.J.; Zuccherato, R.J. Faster Attacks on Elliptic Curve Cryptosystems. In *International Workshop on Selected Areas in Cryptography*; Springer: Berlin/Heidelberg, Germany, 1998. Available online: https://link.springer.com/chapter/10.1007/3-540-48892-8_15 (accessed on 7 January 2020).
40. Bertoni, G.; Daemen, J.; Peeters, M.; van Assche, G.; van Keer, R. Keccak Implementation Overview. 2012. Available online: https://keccak.team/files/Keccak-implementation-3.2.pdf (accessed on 7 January 2020).

41. Bitcoin Team. PoW Implementation. The Bitcoin Core Developers. 2018. Available online: https://github.com/bitcoin/bitcoin/blob/master/src/pow.cpp (accessed on 8 January 2020).
42. Chou, C.; Lin, Y.; Chen, R.; Chang, H.; Tu, I.; Liao, S. Personalized Difficulty Adjustment for Countering the Double-Spending Attack in Proof-of-Work Consensus Protocols. *arXiv* **2018**, arXiv:1807.02933. Available online: https://arxiv.org/abs/1807.02933 (accessed on 8 January 2020).
43. Crandall, R.E. Method and Apparatus for Public Key Exchange in a Cryptographic System. U.S. Patent 5159632A, 27 October 1992. Available online: https://patents.google.com/patent/US5159632A/en (accessed on 8 January 2020).
44. Solinas, J.A. Generalized Mersenne Numbers. 1999. Available online: http://www.cacr.math.uwaterloo.ca/techreports/1999/corr99-39.pdf (accessed on 8 January 2020).
45. Chung, J.; Hasan, A. More Generalized Mersenne Numbers. 2003. Available online: http://cacr.uwaterloo.ca/techreports/2003/corr2003-17.ps (accessed on 8 January 2020).
46. Acar, T.; Shumow, D. *Modular Reduction without Pre-Computation for Special Moduli*; Microsoft Research: Redmond, WA, USA, 2016. Available online: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/modmul_no_precomp.pdf (accessed on 8 January 2020).
47. Bos, J.W.; Costello, C.; Hisil, H.; Lauter, K. Fast Cryptography in Genus 2. Cryptology ePrint Archive (IACR). 2012. Available online: https://eprint.iacr.org/2012/670.pdf (accessed on 8 January 2020).
48. Hamburg, M. Fast and Compact Elliptic-Curve Cryptography. Cryptology ePrint Archive (IACR). 2012. Available online: https://eprint.iacr.org/2012/309.pdf (accessed on 8 January 2020).
49. Meneghetti, A.; Sala, M.; Taufer, D. A survey on Efficient Parallelization of Blockchain-Based Smart Contracts. *AETiC* **2019**, *3*, 5. Available online: http://aetic.theiaer.org/archive/v3/v3n5/p2.html (accessed on 8 January 2020). [CrossRef]
50. Certicom Research. *SEC 1: Elliptic Curve Cryptography*; Certicom Research, Information-Technology Promotion Agency: Toronto, ON, Canada, 2000. Available online: https://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1_01sec1.pdf (accessed on 8 January 2020).
51. Lochter, M.; Merkle, J. Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation. 2010. Available online: https://tools.ietf.org/html/rfc5639 (accessed on 8 January 2020).
52. Bellare, M.; Rogaway, P. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. CCS '93: Proceedings of the 1st ACM Conference on COMPUTER and Communications Security. 1993; pp. 62–73. Available online: http://cseweb.ucsd.edu/~mihir/papers/ro.html (accessed on 8 January 2020).
53. Galbraith, S.D.; Mckee, J. The Probability That the Number of Points on an Elliptic Curve over a Finite Field Is Prime. *J. Lond. Math. Soc.* **2000**, *62*, 671–684. [CrossRef]
54. Bosma, W.; Cannon, J.; Playoust, C. The Magma algebra system. I. The user language. *J. Symb. Comput.* **1997**, *24*, 235–265. Available online: http://magma.maths.usyd.edu.au/magma (accessed on 8 January 2020). [CrossRef]