A Survey of Probabilistic Micropayment Schemes

Sulyab Thottungal Valapu and Bhaskar Krishnamachari
Viterbi School of Engineering
University of Southern California
{thottung, bkrishna} @ usc.edu

ABSTRACT

Although the earliest electronic micropayment schemes date back to the mid-90s, recent years have witnessed a resurgence of research interest in the field due to the rising popularity of cryptocurrencies and the associated increase in transaction fees. Probabilistic micropayment schemes have shown particular theoretical promise due to their ability to aggregate payments beyond client-merchant pairs. In this paper, we review various probabilistic micropayment protocols proposed in both pre-cryptocurrency and post-cryptocurrency eras and provide an analysis of what the future of research in this field could look like.

I. INTRODUCTION

Micropayments, commonly defined as payments worth pennies or fractions of pennies [1] [10], have numerous potential applications such as pay-as-you-go multimedia streaming, ad-free Internet [2], and IoT data marketplaces [3]. However, processing micropayments as individual transactions is often economically infeasible for merchants since the transaction fees incurred can approach or even exceed the value of the payment itself. For example, the average credit card processing fees in the US include a flat fee ranging from 5 to 10 cents in addition to a percentage fee that varies with the transaction value [4]. Although considered micropayment systems in their own right during their early days due to low transaction fees [5], the average transaction fees of popular cryptocurrencies such as Bitcoin and Ethereum have increased considerably with increasing adoption, touching highs of $63 and $69 respectively [6] [7]. Furthermore, high-frequency, low-value transactions [8] can congest both traditional and cryptocurrency payment networks, potentially driving up transaction fees and confirmation delays. Hence, it is necessary to aggregate multiple transactions into one to make micropayments viable in terms of both economical metrics (for merchants), as well as operational metrics (for payment networks and/or banks.) Various electronic micropayment schemes have been proposed since the mid-90s [5] with this goal, with Millicent [9], NetBill [16], and Agora [17] being some of the first.

We survey probabilistic micropayment schemes, a class of micropayment schemes introduced in the seminal works of Rivest [12] and Wheeler [13]. Although early works assumed a traditional payment infrastructure with bank as a trusted third party, modern schemes [2] are built atop cryptocurrency protocols without centralized trust. Probabilistic micropayment schemes have been a topic of continued research interest due to their low book-keeping overhead and “universal aggregation" capability [19], i.e., the ability to amortize transactions across client-merchant pairs. In particular, they show promise as effective layer 2 solutions to make low-value transactions viable on cryptocurrencies with high transaction fees [15].

The paper is organized as follows: In Section II, we describe the basics of probabilistic micropayments. In Sections III and IV, we describe various schemes proposed in pre-cryptocurrency and post-cryptocurrency eras respectively. In Section V, we analyse the strengths and weaknesses of existing schemes and conclude by listing possible research directions in the area.

II. CONCEPT

A probabilistic micropayment protocol is defined between a client c and a merchant m over the course of a series of rounds $r_1, r_2, .., r_n$. In the pre-processing stage, the parties reach an agreement on the *macropayment value M* and the *winning probability* p. Each round is a Bernoulli trial with outcomes *win* or $lose$ having probabilities p and $1 - p$ respectively. Upon a win outcome, the client has to pay the merchant the macropayment value M. Otherwise, the client has to pay nothing. Thus, the expected payment per round (i.e, *micropayment value*) is $M * p$.

As an illustration, consider the use case of a pay-as-you-go WiFi internet service. Assume that the provider charges $0.01 per MB used. However, since transactions worth 1 cent are infeasible, the provider and users could engage in a probabilistic micropayment protocol with parameters $M = \$10$ and $p = \frac{1}{1000}$ and a round of the protocol executed every time the user consumes one megabyte of data. Thus, although the win outcome is expected only once every 1000 rounds, they result in economically feasible macropayments of $10 each. However, on average, the provider is paid the same micropayment value of $0.01 per round.

III. CLASSICAL SCHEMES (1996-2002)

The concept behind probabilistic micropayments was introduced in 1996 by Wheeler [13], who used the term "bets", and in 1997 by Rivest [12], who used the term "lottery tickets". In particular, Rivest proposed a full-fledged protocol using hash chains similar to PayWord [11]. Lipton and Ostrovsky [18] modified the protocol by incorporating zero-knowledge proofs to prove security properties. In 2002, Micali and Rivest [14] introduced three new schemes. Peppercoin [19] micropayment system, built based on these schemes, appeared in 2004. However, none of the implementations saw significant adoption, due to reasons including the prohibitive scope of changes required to be made to global payment systems to incorporate the schemes [2].

a. Rivest's Lottery Scheme (1997)

In this scheme [12], the bank acts as a trusted third party that provides clients with the credential to issue lottery tickets as payment. In the pre-processing stage, the merchant chooses a random value m_n and creates the hash chain $m_0, m_1, m_2, .., m_n$ where

$$m_i = H(m_{i+1}), \quad \forall i = 0,1,2,..,n-1$$

and commits to the chain by signing a message containing m_0 using their private key. Similarly, the client chooses a random value c_n and creates the hash chain $c_0, c_1, c_2, .., c_n$ defined in the same way and commits to the chain by signing a message containing c_0 using their private key. In the jth round of the protocol, the client discloses c_j. Assuming that the winning probability is $\frac{1}{1000}$, the merchant wins if the last 3 digits of m_j are equal to the last 3 digits of c_i, in which case the merchant discloses m_j to prove that they won. The client can verify the claim by simply repeatedly hashing m_j for a total of j times and checking whether the result equals m_0. This scheme works due to the one-way property of cryptographic hash functions. Given m_j or c_j for any round j, it is computationally hard for the other party to predict what m_{j+1} or c_{j+1} would be. Furthermore, verification of winning claims is easy as it involves only a fixed number of hash calculations.

Nevertheless, the scheme is not without its drawbacks. A malicious client could refuse to pay a macropayment, in which case the merchant would be helpless except to report the incident to the bank with proof. Some legitimate clients may find that too many of the tickets they issued end up winning, thus forcing them to pay more than what they owe. Furthermore, every round of the protocol requires two-way communication between the client and the merchant, limiting its scalability.

b. MR1, MR2 and MR3 Schemes (2002)

In their 2002 paper, Micali and Rivest [14] introduced three new probabilistic micropayment schemes numbered MR1 to MR3, each aimed to address various drawbacks of Rivest's lottery scheme.

MR1 makes Rivest's lottery scheme non-interactive, thereby reducing its communication overhead. It assumes a public function $F(.)$ that takes arbitrary bit strings as input and outputs a value in the interval $[0,1]$. For any transaction T, the client issues their digital signature on T as the lottery ticket C_T. The merchant wins the lottery if $F(SIG_M(C_T)) < p$, where p is the pre-determined chance of winning the lottery. Since $SIG_M(C_T)$ is unpredictable to the client and $SIG_C(T)$ cannot be altered by the merchant, provided that F has a uniform distribution, the winning rate cannot be altered by the client, merchant, or the bank. This scheme is non-interactive to the extent that the merchant need not respond to the lottery ticket messages from the client.

MR2 addresses the possibility of clients getting overcharged due to bad luck. In this scheme, every lottery ticket issued by a client contains a unique serial number starting from 1 and assigned sequentially. The bank keeps track of the serial number of the latest winning check issued by the client. When the client issues a new winning check, the bank credits the full winning amount to the merchant but only charges the client what is owed, based on the winning serial numbers. The scheme effectively shifts the burden of risk from the client to the bank, with the assumption that over many clients issuing large numbers of micropayments, the bank does not lose much. However, the bank needs to deploy additional measures to identify malicious clients who, for example, issue multiple micropayments with the same serial number. The scheme also opens up possibility for clients and merchants to collude to make illicit gains. Finally, correlating the serial number with a unit of payment would require changes at the scheme level to support micropayments for different amounts.

MR3 inherits the benefits of MR1 and MR2, and improves on efficiency by reducing the number of times the merchant contacts the bank. The winning tickets are stored by the merchant and submitted to the bank in chunks.

IV. MODERN SCHEMES (2015-PRESENT)

After a hiatus lasting more than a decade during which no research papers were published, the advent of cryptocurrencies and smart contracts brought revived interest probabilistic micropayments. In 2015, Pass and Shelat [2] introduced three schemes (named MICROPAY1-3) that adapted Rivest's lottery scheme to cryptocurrencies, including one that worked with Bitcoin. Later years saw several schemes proposed, including DAM [20], Hu and Zhang's protocol [22], MicroCash [1] and Randpay [23], each focusing on enhancing scalability, anonymity, and/or decentralization properties.

A. MICROPAY (2015)

This paper [2] introduced three new schemes named MICROPAY 1-3. MICROPAY 1 does not require a trusted third party but cannot be implemented on Bitcoin due to the limitations of the Bitcoin scripting language. MICROPAY 2 overcomes this limitation by requiring a *Verfiable Transaction Service (VTS)* to act as the trusted third party. MICROPAY 3 improves upon MICROPAY 2 by requiring the VTS to intervene only in the case of disputes.

MICROPAY 1 consists of three stages. Firstly, the client transfers the macropayment amount M to a new "escrow" Bitcoin address with a specific release condition (described later.) Once the trade is underway, the merchant requests for payment by choosing a random string r_M, and sending c_M, a commitment to that value (eg: a one-way hash of r_M) and a_M, the payment address to the merchant. The client then makes a probabilistic payment by choosing a random string r_C and signing (c_M, r_C, a_M) using the private key of the escrow address. Assuming that the winning probability is $\frac{1}{100}$, the release condition of the escrow address checks that the first two

digits of $r_M \oplus r_C$ are 00. If this is indeed the case, the merchant can create a new transaction with this information to avail transfer of the macropayment amount from the escrow account. The authors note that although MICROPAY 1 cannot be implemented in Bitcoin, it could be implemented in any cryptocurrency with expressive scripting languages, such as Ethereum.

MICROPAY 2 enables the scheme to be implemented using Bitcoin scripting language with the trade-off of using a trusted third party VTS. The scheme proceeds identical to MICROPAY 1 until the point where the merchant wins the lottery, upon which the merchant forwards the relevant information to the VTS. The release script of the escrow accounts created as per MICROPAY 2 requires a multisig of the client (which is always provided to the merchant) as well as the VTS.

A major drawback of MICROPAY 1 & 2 is that both schemes are susceptible to front-running attacks, i.e, the client could rush to withdraw funds from the escrow account upon learning about the merchant's lottery win. MICROPAY 3 addresses this drawback as well as limits the involvement of VTS to resolving disputes. This is achieved by using two escrow addresses: one in the control of the client, and one in joint control of the client, merchant, and the VTS. When the client issues a lottery ticket as payment, money gets released from the first escrow address to the second escrow address. Money can be released from the second escrow address in two ways: to the merchant, if either the client or the VTS agrees; and to the client, if the VTS agrees. Specifically, if the client agrees with a winning ticket, there is no need to forward the transaction to the VTS; and a malicious client cannot perform a front-running attack because they cannot withdraw money from the second escrow without the VTS agreeing. However, MICROPAY 3 requires an additional on-chain transaction due to the second escrow requirement, which can increase the transaction fees considerably.

B. DAM (2017)

The DAM [20] scheme extends Zerocash [21] as an offline probabilistic micropayment scheme with anonymity guarantees. The protocol uses NIZK (Non-interactive Zero-knowledge Proof) techniques to ensure that merchants cannot identify lottery tickets backed by the same escrow. The paper also provides a thorough economic and game theoretic analysis of worst-case and average-case gains made by malicious clients with double-spending strategies, and the resulting bounds are used to establish penalty escrows.

DAM makes use of fractional message transfer (FMT) as a lottery mechanism. Using FMT, the client sends an encrypted payment message to the merchant, who has a pre-defined winning probability p of being able to decrypt the message. Thus, the merchant wins the lottery only if they are able to decrypt the message. However, since the payment message is encrypted, the client has to prove the validity of the payment message to the merchant by sending a NIZK proof along with it. If the merchant is able to decrypt the message only to find that the ticket has already been spent, they are able to recover the penalty escrow. On the other hand, if the merchant is unable to decrypt the message, the client can "refresh" the ticket for reuse in another micropayment, with the property that the old and the new ticket cannot be linked together by an external observer, thus providing anonymity.

The main drawback of DAM is the expensive cryptographic operations involved. Each micropayment involves FMT and NIZK operations, making the practicality of the scheme questionable for high-frequency micropayments, especially for clients with limited resources.

C. Hu and Zhang's Scheme (2018)

Hu and Zhang [22] improves upon MICROPAY 3 by reducing the number of on-chain transactions required from two to one while retaining the advantages of the scheme, namely protection from double-spending and front-running attacks, and a trusted third party that is invoked only in the case of dispute. To achieve this, the scheme makes use of a cryptographic primitive called accountable assertions [25], and the timelock functionality of Bitcoin. Informally, accountable assertions allow a user to "assert" statements to a pre-defined number of

"contexts." The assertions are publicly verifiable. More importantly, if two or more distinct statements are asserted to the same context by the user, the private key of the user becomes publicly extractable.

In the pre-processing phase of the protocol, the client creates two escrows: one for making macropayments, and the other a penalty escrow. The penalty escrow can be reused by the client for a fixed number of transactions d. The money from the penalty escrow can be released in two ways: by the client themselves after a fixed amount of time T (thus preventing front-running attacks), or with a multisig of the merchant and the trusted third party before time T (thus ensuring that the merchant gets paid if the client refuses to pay.) A payment round of the scheme starts similar to MICROPAY with the merchant choosing a random number r_M, creating a commitment c_M and sending (c_M, a_M) to the client where a_M is the payment account of the merchant. The client then picks a random number r_C, creates signatures and sends them back along with an "assertion" on the serial number of the ticket, denoted by s. Upon validating the information, the merchant publishes the assertion on a public bulletin board. Thus, if the client tries to double spend by reusing the same serial number, anyone can extract their private key and they end up losing the money in the penalty escrow. If the client acts honestly, the penalty escrow is untouched, and can be reused for another $d - s$ rounds. Finally, if the client does not double spend but refuses to honor a winning ticket, the merchant gets paid from the penalty escrow using a multisig of the merchant and the trusted third party.

It could be argued that the reduction of on-chain transactions by one compared to MICROPAY 3 compensates for the higher number of expensive cryptographic computations. A minor drawback of the scheme is that although accountable assertions ensure that a double-spending client gets punished, it does not ensure that the merchants get paid in those cases.

D. MicroCash (2020)

All cryptocurrency micropayment schemes discussed so far requires an honest client to wait for the merchant's response before issuing a new ticket backed by the same escrow, since escrow reuse is possible only if the merchant does not win the lottery. MicroCash [1] overcomes this limitation by making concurrent issuance of tickets using a single escrow possible. Furthermore, the protocol selects an exact pre-defined number of winning tickets, thus addressing the (although improbable) risk of issuing too many winning tickets.

MicroCash defines "rounds" in a specific way, with the duration of a round equal to the mining interval between two consecutive Bitcoin blocks. In the pre-processing stage of MicroCash, the client sets up a payment escrow and a penalty escrow. During payment rounds, the client issues tickets with unique sequence numbers to the merchants. The merchants hold on to the tickets until the winning tickets are declared, which happens after a fixed number of rounds R. The winning ticket numbers are determined using the result of applying a Verifiable Delay Function (VDF) to the block mined after R rounds. Since exact number of winning tickets are determined based on parameters defined by the client during escrow creation, the client is never overcharged. The winning merchants then submit their tickets to the mining pool for redemption. Since the winning tickets are identified based on their sequence numbers, double spending can be detected in the case of winning tickets, and is penalized by the forfeiture of penalty escrows. Furthermore, the client is not allowed to withdraw funds from their escrows for a fixed time interval, making front-running attacks unlikely.

The main drawback of MicroCash is its lack of flexibility. Since parameters such as winning probability, macropayment amount and the list of merchants are pre-defined at the time of escrow creation, any departure from these parameters would require the creation of a new escrow. Furthermore, unless the entire space of lottery tickets as defined by escrow parameters is used up, there is still a slim chance of the client issuing too many winning tickets. Thus, the scheme does not completely succeed at either of its stated objectives.

E. Randpay (2020)

Randpay [23] takes a significantly different approach from other schemes by not requiring escrows or a trusted third party. Instead, the lottery is directly played between the merchant and the client in a single round of communication. However, due to the requirement of a special cryptocurrency address called RandpayUTXO, Randpay is compatible only with Emercoin [24] as of now.

In the preprocessing stage of Randpay, the merchant and the client agree upon a winning probability p and a macropayment amount M. In a round of the scheme, the merchant generates a new asymmetric key pair and calculates the corresponding cryptocurrency address (eg: `0xabcdabcd`). The merchant then sends the client an address range depending on the address generated in the first step and the winning probability. For example, if the winning probability is $\frac{1}{256}$, the merchant may send the range `[0xabcdab00,0xabcdabff]`. The client then randomly picks one cryptocurrency address in the range, creates a cryptocurrency transaction paying that address the sum M, signs it and sends it back to the merchant. The merchant wins the lottery if the client chooses the payment address generated by the merchant. However, a winning merchant needs to prove that they indeed have the private key corresponding to the winning address, since otherwise a malicious merchant could falsely claim that they won, effectively "burning" the client's money. To prove his claim, the merchant adds the special cryptocurrency address called RandpayUTXO as an input to the lottery transaction, and signs RandpayUTXO with the private key corresponding to the winning payment address. The merchant then submits the transaction to the mining pool as usual. Transactions having RandpayUTXO as one of the inputs are validated by miners only if the RandpayUTXO input is signed by the private key corresponding to the output address.

While Randpay benefits from its simplicity and not requiring escrows or third parties, it is susceptible to double spending and front-running attacks. Furthermore, implementing it on popular cryptocurrencies such as Ethereum is not possible without redesigning them, which could hinder its adoption.

V. DISCUSSION

The probabilistic micropayment schemes described in this paper use different metrics to compare between themselves. Some prioritize scalability properties by minimizing escrows [22], removing them altogether [23], supporting concurrent micropayments [1] or making the scheme non-interactive [14][2]; while some others focus on decentralization properties by improving anonymity [20] or removing the need for a third party [23]. It is seen that most schemes lay emphasis on security properties by ensuring protection from double spends and front-running attacks, with Randpay [23] being a notable exception. However, it is a common theme across schemes that improving one property almost always requires making compromises in one or more other properties. For example, in providing anonymity guarantees while preserving double-spend protection, DAM [20] sacrifices scalability; and in increasing scalability by not requiring escrows, Randpay [23] compromises on protection from double spending and front-running attacks. As of today, there is no probabilistic scheme that ticks all the boxes, indicating the vast research potential in this area. Making cryptocurrency micropayments viable is a key hurdle to crypto-economy becoming ubiquitous, and with its universal aggregation capability, probabilistic micropayments could be the best bet towards scaling it.

REFERENCES

[1] G. Almashaqbeh, A. Bishop, and J. Cappos, "MicroCash: Practical Concurrent Processing of Micropayments," in Financial Cryptography and Data Security, Cham, 2020, pp. 227–244.

[2] R. Pass and Abhi Shelat, "Micropayments for Decentralized Currencies," in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, Oct. 2015, pp. 207–218. doi: 10.1145/2810103.2813713.

[3] J. Robert, S. Kubler, and S. Ghatpande, “Enhanced Lightning Network (off-chain)-based micropayment in IoT ecosystems,” Future Generation Computer Systems, vol. 112, pp. 283–296, Nov. 2020, doi: 10.1016/j.future.2020.05.033.

[4] L. Daly, “Average credit card processing fees and costs in 2021,” The Motley Fool, 13-Jan-2022. [Online]. Available: **https://www.fool.com/the-ascent/research/average-credit-card-processing-fees-costs-america**

[5] S. T. Ali, D. Clarke, and P. McCorry, “The Nuts and Bolts of Micropayments: A Survey,” arXiv:1710.02964 [cs], Oct. 2017. [Online]. Available: **http://arxiv.org/abs/1710.02964**

[6] “Bitcoin Avg. Transaction Fee Chart,” BitInfoCharts. **https://bitinfocharts.com/comparison/bitcoin-transactionfees.html**.

[7] “Ethereum Avg. Transaction Fee Chart,” BitInfoCharts. **https://bitinfocharts.com/comparison/ethereum-transactionfees.html**.

[8] M.-S. Hwang, I.-C. Lin, and L.-H. Li, “A simple micro-payment scheme,” Journal of Systems and Software, vol. 55, no. 3, pp. 221–229, Jan. 2001, doi: 10.1016/S0164-1212(00)00072-8.

[9] S. Glassman, M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro, “The Millicent Protocol for Inexpensive Electronic Commerce,” 1995. Available: **https://www.w3.org/Conferences/WWW4/Papers/246/**

[10] H. Beadle, R. Gonzalez, R. Safavi-Naini, and S. Bakhtiari, “A Review of Internet Payments Schemes,” in Proceedings of the Australian Telecommunication Networks & Applications Conference 1996, Sep. 1996.

[11] Rivest R.L., Shamir A. (1997), “PayWord and MicroMint: Two simple micropayment schemes.” In: Lomas M. (eds) Security Protocols. Security Protocols 1996. Lecture Notes in Computer Science, vol 1189. Springer, Berlin, Heidelberg.

[12] R. L. Rivest, “Electronic lottery tickets as micropayments,” in Proceedings of Financial Cryptography ’97, volume 1318 of Lecture Notes in Computer Science, pages 307–314. Springer, 1997.

[13] D. Wheeler, “Transactions using bets,” in Security Protocols, volume 1189 of Lecture Notes in Computer Science, pages 89–92. Springer, 1996.

[14] Micali S., Rivest R.L., “Micropayments Revisited,” In: Preneel B. (eds) Topics in Cryptology — CT-RSA 2002. CT-RSA 2002. Lecture Notes in Computer Science, vol 2271. Springer, Berlin, Heidelberg.

[15] Takahashi T., Otsuka A., “Probabilistic Micropayments with Transferability,” In: Bertino E., Shulman H., Waidner M. (eds) Computer Security – ESORICS 2021. ESORICS 2021. Lecture Notes in Computer Science, vol 12972. Springer, Cham.

[16] M. Sirbu and J. D. Tygar, “NetBill: an Internet commerce system optimized for network-delivered services,” in IEEE Personal Communications, vol. 2, no. 4, pp. 34-39, Aug. 1995, doi: 10.1109/98.403456.

[17] E. Gabber and A. Silberschatz, “Agora: a minimal distributed protocol for electronic commerce,” in Proceedings of the 2nd conference on Proceedings of the Second USENIX Workshop on Electronic Commerce - Volume 2, USA, Nov. 1996, p. 13.

[18] Lipton R.J., Ostrovsky R. (1998) Micro-payments via efficient coin-flipping. In: Hirchfeld R. (eds) Financial Cryptography. FC 1998. Lecture Notes in Computer Science, vol 1465. Springer, Berlin, Heidelberg. **https://doi.org/10.1007/BFb0055469**

[19] Rivest R.L. (2004) Peppercoin Micropayments. In: Juels A. (eds) Financial Cryptography. FC 2004. Lecture Notes in Computer Science, vol 3110. Springer, Berlin, Heidelberg. **https://doi.org/10.1007/978-3-540-27809-2_2**

[20] Chiesa A., Green M., Liu J., Miao P., Miers I., Mishra P. (2017) Decentralized Anonymous Micropayments. In: Coron JS., Nielsen J. (eds) Advances in Cryptology – EUROCRYPT 2017. EUROCRYPT 2017. Lecture Notes in Computer Science, vol 10211. Springer, Cham. **https://doi.org/10.1007/978-3-319-56614-6_21**

[21] E. Ben Sasson et al., “Zerocash: Decentralized Anonymous Payments from Bitcoin,” 2014 IEEE Symposium on Security and Privacy, 2014, pp. 459-474, doi: 10.1109/SP.2014.36.

[22] Hu K., Zhang Z. (2018) Fast Lottery-Based Micropayments for Decentralized Currencies. In: Susilo W., Yang G. (eds) Information Security and Privacy. ACISP 2018. Lecture Notes in Computer Science, vol 10946. Springer, Cham. **https://doi.org/10.1007/978-3-319-93638-3_38**

[23] Oleksii Konashevych, Oleg Khovayko, Randpay: The technology for blockchain micropayments and transactions which require recipient’s consent, Computers & Security, Volume 96, 2020, 101892, ISSN 0167-4048, **https://doi.org/10.1016/j.cose.2020.101892**.

[24] “Official website of Emercoin.” **https://emercoin.com/en/** (accessed Feb. 14, 2022).

[25] Tim Ruffing, Aniket Kate, and Dominique Schröder. 2015. Liar, Liar, Coins on Fire! Penalizing Equivocation By Loss of Bitcoins. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS ’15). Association for Computing Machinery, New York, NY, USA, 219–230. DOI: **https://doi.org/10.1145/2810103.2813686**

[26] Boneh D., Bonneau J., Bünz B., Fisch B. (2018) Verifiable Delay Functions. In: Shacham H., Boldyreva A. (eds) Advances in Cryptology – CRYPTO 2018. CRYPTO 2018. Lecture Notes in Computer Science, vol 10991. Springer, Cham. **https://doi.org/10.1007/978-3-319-96884-1_25**