Resilient Consensus Sustained Collaboratively

Junchao Chen, Suyash Gupta[†], Alberto Sonnino[‡], Lefteris Kokoris-Kogias[§‡], Mohammad Sadoghi
Exploratory Systems Lab
University of California, Davis
[†]University of California, Berkeley [‡] MystenLabs [§] IST Austria

ABSTRACT

Decentralized systems built around blockchain technology promise clients an immutable ledger. They add a transaction to the ledger after it undergoes consensus among the replicas that run a Proof-of-Stake (PoS) or Byzantine Fault-Tolerant (BFT) consensus protocol. Unfortunately, these protocols face a long-range attack where an adversary having access to the private keys of the replicas can rewrite the ledger. One solution is forcing each committed block from these protocols to undergo another consensus, Proof-of-Work (PoW) consensus; PoW protocol leads to wastage of computational resources as miners compete to solve complex puzzles. In this paper, we present the design of our Power-of-Collaboration (PoC) protocol, which guards existing PoS/BFT blockchains against long-range attacks and requires miners to collaborate rather than compete. PoC guarantees fairness and accountability and only marginally degrades the throughput of the underlying system.

ACM Reference format:
Junchao Chen, Suyash Gupta[†], Alberto Sonnino[‡], Lefteris Kokoris-Kogias[§‡], Mohammad Sadoghi. 2018. Resilient Consensus Sustained Collaboratively. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai, Woodstock, NY, June 03–05, 2018 (Conference acronym 'XX),* 7 pages.
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Decentralized systems built using blockchain technology promise their clients an immutable and verifiable ledger [30, 52, 69]. These systems receive client transactions and use state machine replication to add these transactions to the ledger. As these systems are often composed of untrusting nodes, some of which are malicious or Byzantine, establishing state machine replication requires these systems to run a *consensus* protocol that can handle malicious attacks [15, 72]. The two most widely adopted categories of these consensus protocols are: Proof-of-Stake (PoS) protocols [30, 40] and traditional Byzantine Fault-Tolerant (BFT) protocols [15, 72].

Stake-oriented consensus protocols, such as Proof-of-Stake (PoS) protocols, use a probabilistic distribution to decide which node gets to add a new block of client transactions to the ledger; often, the nodes with a higher stake (or wealth) have higher probability of proposing a new block. [40]. Communication-oriented protocols, such as traditional BFT protocols, give each node an equal opportunity (a vote) to add an entry to the ledger; agreement on the next block is reached through successive rounds of vote exchange [15, 45]. Some systems combine PoS and BFT to yield efficient consensus [30]. Despite these differences, these protocols follow the same design: each block added to the ledger includes the *digital signatures* of a quorum of participants to prove that a quorum agreed to update the ledger.

Unfortunately, any decentralized system that employs these protocols suffer from a well-known attack: a *long-range attack* where an adversary attempts to create an alternate ledger and targets clients (or a new participants) that cannot distinguish between the original ledger and the adversarial ledger [2, 8, 66, 68]. An adversary can launch a long-range attack on systems running PoS/BFT consensus protocols due to the following reason: In PoS/BFT protocols, it is *computationally inexpensive* for nodes to add a new block to the ledger. An adversary with *access to the private keys* of the honest nodes can use these keys to create an alternate ledger; the following are the two ways to access the private keys of honest nodes.

(1) *Stealing.* An adversary can attempt to steal the keys of the nodes; stealing private keys is a widespread attack, and such attacks have resulted in losses of up to \$200 million [53, 56, 74].

(2) *Bribing.* An adversary can bribe honest nodes to sell their private keys, especially nodes that once participated in the system and no longer have any stake in it. This bribery attack is feasible because decentralized systems expect to run for years and cannot guarantee that the original set of participants will always run the system. Based on the Tragedy of the Commons [11], rational participants will opt to earn further incentives by selling their keys.

Once an adversary has access to these keys, it can use them to fork the original ledger at a specific block number and create an adversarial ledger with alternate blocks. As nodes of decentralized systems frequently leave/join the system [54, 62], the adversary can use this opportunity to present its adversarial ledger as the authentic ledger to a new node (or client), which unfortunately *cannot distinguish between the two.* Note: even though honest nodes of the system have access to the original ledger if the adversary can access their keys, it can forge their identities, which makes it hard for a new node to distinguish between the two ledgers.

Prior attempts to eliminate long-range attacks follow three directions: (1) Using key-evolving cryptographic techniques and increasing the number of keys an adversary needs to compromise [8, 29], which typically delays the imminent long-range attack. (2) Creating state checkpoints and storing them at all the nodes, assuming that an adversary can only compromise the keys of at most one-third of nodes [14, 66] (3) Periodically appending the ledger state to the Bitcoin blockchain [9, 67]. Indeed, the third direction can guard existing decentralized systems against long-range attacks. For an adversary to present an adversarial chain to the new nodes, it also needs to rewrite the Bitcoin ledger, which is computationally infeasible. Bitcoin employs the PoW consensus protocol, which follows a *computation-oriented* model as it requires all the nodes to compete toward solving a complex puzzle. Whichever node solves the puzzle first adds a new block and receives a reward as compensation for its efforts. As PoW nodes constantly compete with each other, PoW-based solutions lead to the wastage of computational resources, as there is only one winner. [23].

The challenges existing solutions face while eliminating long-range attacks make us conclude that any solution for long-range attacks should: (1) not rely on the long-term safe-keeping of private keys, (2) reduce wastage of computational resources, and (3) be computationally expensive for an attacker to rewrite the ledger.

In this paper, we introduce *Power-of-Collaboration* (PoC) protocol, which, when appended to decentralized systems running PoS/BFT consensus, helps to meet the aforementioned goals. PoC is noninvasive as it works on the output of underlying PoS/BFT consensus protocol and has minimal impact on the performance of existing decentralized systems. PoC advocates for *collaborative mining*, which, like PoW, requires miners to solve a compute-intensive puzzle, but all the miners are now working together (instead of competing) to solve the same puzzle.

The most closely related work, Bitcoin's *centralized mining pools*, also attempts to reduce the costs associated with mining [1, 20, 50]. As the name indicates, these mining pools are centralized and managed by an organization. The organization sets the rules for the mining pool, decides which node should receive a reward and how much reward, and controls which node can participate in the pool. Not only is the existence of centralized mining pools against the ethos of a decentralized system, but the managing organization charges fees for management without spending any computational resources. Further, attempts to create a decentralized mining pool have been unsuccessful due to nodes not doing designated tasks and lack of accountability: the last block added by any decentralized mining pool in Bitcoin was in 2019 [20, 50].

PoC, in essence, functions as a single decentralized mining pool where all the nodes collaborate to find a solution for the compute-intensive puzzle. Like PoW, nodes are still spending their computational resources to find the nonce, which makes it computationally expensive for the adversary to create an adversarial ledger. However, we need to ensure that, like centralized mining pools, we reduce the wastage of computational resources while also guaranteeing decentralization and *fairness*. We do so by splitting the compute-intensive puzzle into a set of unique sub-problems, and each node works on a unique subset of these sub-problems; the solution to the original compute-intensive problem is present in these subsets. We also need to provide *accountability* and deter malicious nodes from not doing work, as it can delay the discovery of the solution. PoC does so through our slice-shifting protocol, which identifies and penalizes a malicious miner and transfers its work to honest miners.

To show that PoC is effective in practice, we append it to several decentralized systems. In our first set of experiments, we append PoC to Apache's ResilientDB (Incubating) [37] as it provides access to an open-source permissioned blockchain platform and an optimized implementation of Pbft, a BFT consensus protocol. ResilientDB's Pbft implementation adds approximately 1000 blocks per second on a system of 128 replicas, and our experiments illustrate that PoC can sustain this throughput on a system of 128 miners and requires 29× less mining time than Bitcoin's PoW protocol. In our final set of experiments, we use the Diablo [32] benchmarking framework to append PoC to *four* popular blockchain systems, namely Diem [25], Algorand [30], Ethereum [69], and Quorum [17]. Our results illustrate that PoC has a minimal impact (≈ 10×) on the throughput of these blockchains. Next, we list our contributions.

- We present the Power-of-Collaboration (PoC) protocol, which, when appended to existing decentralized systems running PoS and BFT protocols, makes it computationally-expensive for an adversary to launch a long-range attack.
- PoC introduces the notion of collaborative mining, which divides the mining task among all the miners.
- PoC advocates fairness and accountability: rewards are distributed among the miners in proportion to their share of work, and Byzantine behavior is quickly detected and penalized through the slice-shifting mechanism.

2 SYSTEM MODEL

We adopt the standard communication and failure model adopted by most consensus protocols [15, 31, 45]. We assume the existence of a blockchain system $\mathcal{S}$ of the form $\mathcal{S} = \{\mathcal{R}, C\}$. The set $\mathcal{R}$ consists of $\mathbf{n}_{\mathcal{R}}$ replicas (or stakeholders in case of PoS protocols) of which at most $\mathbf{f}_{\mathcal{R}}$ can behave arbitrarily. In a typical blockchain system, these replicas store the state and participate in consensus. The remaining $\mathbf{n}_{\mathcal{R}} - \mathbf{f}_{\mathcal{R}}$ are honest: they follow the protocol and remain live. We also assume the existence of a finite set of clients C, of which arbitrarily many can be malicious.

Miner Staking. Unlike PoW-based systems, we make a similar assumption as common PoS-based systems: PoC requires the knowledge of total number of miners participating in the mining process. Each miner that wishes to participate in PoC mining needs to *stake* some of its resources. This staking also determines the amount of collaborative work a miner has to perform.

We denote the set of miners as $\mathcal{M}$ and the total number of miners as $\mathbf{n}_{\mathcal{M}}$, of which at most $\mathbf{f}_{\mathcal{M}}$ can act maliciously. The roles of both miners and replicas can be played by the same party, however, for the sake of exposition, we denote miners and replicas separately.

Authenticated communication and Adversary model: replicas and miners employ standard cryptographic primitives such as Mac and/or digital signatures (DS) to sign messages. We denote a message m signed by a replica R using DS as $\langle m \rangle_{\mathbf{R}}$. We employ a *collision-resistant* hash function $\mathsf{hash}(\cdot)$ to map an arbitrary value v to a constant-sized digest $\mathsf{hash}(v)$ [38]. Each replica/miner only accepts a message if it is ***well-formed***. We assume the existence of a *standard adversary* which can corrupt arbitrary nodes, delay and reorder messages [15]. Further, there exists a mechanism for the system $\mathcal{S}$ to allow replicas to join or leave $\mathcal{S}$, and this knowledge is percolated to all the existing members of $\mathcal{S}$. Each such system $\mathcal{S}$ should provide the following guarantees:

Safety. If two honest replicas R1 and R2 order transactions T and T' at sequence numbers k, then $T = T'$.

Liveness. If a client sends a transaction T, then it will eventually receive a response for T.

3 BACKGROUND

We begin by presenting the necessary conceptual background.

3.1 PoS and BFT Consensus

PoS consensus protocols allow each node to add the next block to the blockchain in proportion to its invested stake [22, 30]. Often, the stake is equivalent to a monetary token or currency. As a result, the higher the stake a node invests, the greater the probability of it add a block. Once a stakeholder proposes the next block, all the other nodes also sign this block, which acts like an agreement among the nodes. Similarly, BFT protocols like PBFT [15] and HotStuff [72] designate in each round a replica as a leader, which proposes a block. Following this, all the replicas work through multiple rounds of message exchange to ensure that the proposed block has the support of a quorum of honest replicas.

3.2 Long-range attack on PoS and BFT

A known attack that affects both PoS [9, 67] and BFT [2, 8] protocols is the long-range attack, where an attacker attempts to create an alternate ledger. As described in Section 1, in PoS/BFT protocols, it is computationally inexpensive for nodes to add a new block to the ledger. Thus, an adversary needs access to the private keys of the honest nodes, which it can do either through stealing or bribing. Once an adversary has access to these keys, it can use them to fork the original ledger at a specific block number or height and create an adversarial ledger with alternate blocks (orthogonal, but in the past, popular blockchains have observed forks due to malicious attacks [18]). As nodes of decentralized systems frequently leave/join the system [54, 62], the adversary can use this opportunity to present its adversarial ledger as the authentic ledger to a new node (or client), which unfortunately cannot distinguish between the two. We illustrate this through the following example.

Example 3.1. Assume that a decentralized system $\mathcal{S}$ has the following PoS blockchain ledger: $\mathfrak{B}_1, \mathfrak{B}_2, ...\mathfrak{B}_k, ...\mathfrak{B}_n$. Say malicious nodes get access to the private keys of all the honest nodes and decide to create an adversarial ledger, starting from the k-th block. Once it is the turn of malicious nodes to propose new blocks, they reveal the following adversarial ledger: $\mathfrak{B}_1, \mathfrak{B}_2, ...\mathfrak{B}'_k, ..., \mathfrak{B}'_n, \mathfrak{B}'_{n+1}$. Any new node joining $\mathcal{S}$ cannot distinguish between these two ledgers and will choose the longest chain. Similarly, some existing honest nodes, if bribed, may decide to forfeit their ledger and switch to the malicious ledger. Moreover, as time passes, with old nodes leaving the system and new nodes unable to distinguish, a hard-working adversary may be able to affirm the adversarial ledger as the original ledger.

In practice, there are a lot of examples where an adversary has successfully stolen the private keys of honest parties [53, 56, 74]. We agree that stealing so many keys, primarily when the nodes are distributed is hard. Hence, a rational attack is where the adversary bribes the honest validators who no longer have a stake in the system. As these validators have nothing to lose, Tragedy of the Commons [11] suggests that these validators will sell their private keys in return for some incentive. However, suppose the system can guarantee a fixed set of nodes. In that case, even if the adversary has access to the private keys of these nodes, it cannot convince the honest nodes to switch to the adversarial ledger as, locally, each of them has a copy of the ledger. Unfortunately, it is hard to prevent old nodes from leaving the system. New nodes will eventually fill those spots, and the adversary needs to target only these new nodes. If it can make it impossible for them to distinguish between the two ledgers, which it can do with access to the private keys of old nodes, the adversary can forge the identities to old nodes.

The challenges make us conclude that any solution for long-range attacks should: (1) not rely on the long-term safe-keeping of private keys, and (2) be computationally expensive for an attacker to rewrite the ledger.

3.3 Proof-of-Work Consensus

We briefly look at the design of PoW consensus protocol, which can help in preventing long-range attacks.

In the PoW protocol, each miner $\text{M} \in \mathcal{M}$ selects some client transactions from the available pool of transactions and packs them in a block $\mathfrak{B}$. This block $\mathfrak{B}$ also includes a header, which contains: (i) hash of the previous block (*prev*), (ii) the digest of all transactions or *Merkle root* $M_{\mathfrak{B}}$, (iii) difficulty D, and (iv) the nonce η, among other fields [50, 55]. As each miner decides which transactions to include in its block, two miners may mine blocks with different transactions that *extend* the same previous block (with the hash *prev*). Computing $M_{\mathfrak{B}}$ of all transactions in the block requires a miner M to compute a pairwise hash from leaves to the root. The difficulty D, also termed as the difficulty of finding the nonce, informs the miner of the range of *desired hash.* Specifically, each miner continuously selects a random nonce η till it satisfies the following equation:

$$\mathsf{hash}(prev \mid\mid M_{\mathfrak{B}} \mid\mid \eta) \ < \ D \tag{1}$$

When M discovers a *valid* nonce, it adds it to its block and broadcasts this block to all the miners. When another miner M′ receives a block with a valid nonce that extends the last block added to the ledger (with hash *prev*), M′ adds the received block to its ledger and starts building/mining the next block that extends the received block. Note: once M′ has added a block to the ledger, if in the future, M′ receives any other block that includes *prev*, it ignores that block. Consequently, the miner who discovers the nonce earliest has the highest probability of adding a new block to the ledger as its block can reach a majority of miners the earliest. Clearly, PoW miners compete with each other in an attempt to find a valid nonce and PoW consensus faces the following two challenges (among many others): (1) All but one miner waste their computational resources and only the winner receives an incentive for finding the nonce. (2) More than one miner can find a valid nonce, which can temporarily fork the ledger; as described above, two miners may start mining subsequent blocks that extend different previous blocks. As there is no longer one ledger and instead multiple forks, PoW protocols define a mechanism to trim all but one fork, leading to further wastage of computational resources.

One solution to reduce the probability of forks is to increase the hardness/difficulty of finding the nonce; decentralized systems dynamically update the difficulty D to fix the rate miners add new blocks to the ledger. Specifically, these systems want to ensure miners spend at least a fixed amount of time searching for the valid nonce. The value of D is a system parameter and D increases if miners are producing blocks at a faster rate than expected or the probability of forks is high and decreases vice versa.

3.4 Centralized Pooled Mining

Several decentralized systems, like Bitcoin, allow miners to work in groups to reduce the cost incurred by miners during PoW consensus; miners pool together their resources to increase their chances of finding a valid nonce [1, 50]. Arguably, almost all the active mining pools today are centralized; they are run by an organization that manages the pool's functioning.

The pool controller creates the block for the pool miners to mine and determines a set of lower-difficulty sub-problems; assume that the expected difficulty for adding a block to the ledger is D, then a miner may need to find a nonce at difficulty $d << D$. If a miner finds a valid nonce for a sub-problem, it submits that nonce to the controller. If a nonce leads to a hash at difficulty D, the controller forwards this block to the miners outside the pool and distributes the rewards proportional to the miners who discovered any valid nonce after deducting a management fee.

Mining pools ensure that each miner receives a regular payout (incentive) even if that individual miner cannot discover the nonce that reaches difficulty D. However, by design, these mining pools sacrifice decentralization for centralized management. The pool controller receives a fee for managing the pool and decides the rewards and punishments for the miners, which miners can join the pool, and who to remove from the pool. Moreover, the existence of mining pools does not eliminate the nature of PoW, as often there is more than one mining pool, and these pools compete with each other, which wastes computational resources.

Alternatively, decentralized pools eliminate the need for a pool controller. However, attempts to create a decentralized mining pool have been unsuccessful due to nodes not doing designated tasks and lack of accountability: the last block added by any decentralized mining pool in Bitcoin was in 2019 [20, 50].

4 POWER-OF-COLLABORATION

PoC aims to guard a decentralized system running PoS/BFT protocols from long-range attacks. It offers the following properties:

(G1) **Computationally expensive ledger re-writing.** PoC makes it computationally expensive (solve complex puzzles) for an adversary to overwrite the original ledger,

(G2) **Reduced wastage of computational resources.** PoC requires miners to collaborate and work on the same block; each miner has to work on a subset of search space. Consequently, miners spend less resources than PoW.

(G3) **Fairness.** PoC ensures fairness by distributing incentives among all the miners; even if there is no valid nonce in a miner's subset of the search space, it receives an incentive for its efforts.

(G4) **Accountability.** PoC penalizes any miner that fails to find a valid nonce, if present, in its subset of the search space.

Before we describe the design of PoC, we discuss some of the possible solutions and their limitations.

Version 1. Let us assume a decentralized system running a PoS/BFT consensus protocol employs a PoW consensus subsystem to guard itself against long-range attacks. Each batch of transactions that the PoS/BFT protocols commit is forwarded to the PoW subsystem to add to the ledger maintained by PoW miners. Each miner M follows the PoW protocol: creates a PoW block that includes one or more committed batches, a transaction that transfers incentive to its account, the hash of the previous block *prev*, and initiates the search for a valid nonce. When M finds the nonce, it broadcasts its block and the nonce to the other miners. If another miner M′ receives a block/nonce, it starts building/mining the next block and includes the hash of the received block/nonce as the previous block. This solution faces the following two limitations: (i) all but one miner waste their computational resources (lack of fairness), and (ii) more than one miner can find a valid nonce, which can temporarily fork the ledger and lead to a subsequent increase in the hardness/difficulty of finding the nonce (§3.3).

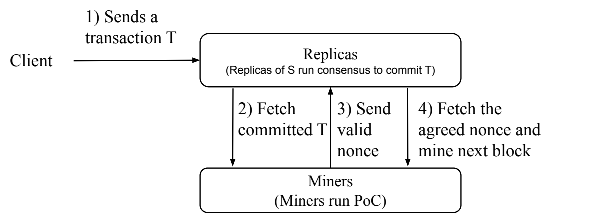

Figure 1: Transactional flow in a system $\mathcal{S}$+PoC.

Version 2. Next, we replace the PoW consensus subsystem with a centralized mining pool, where the pool operator receives the next committed block and creates sub-problems for the pool's miners to mine. This solution does not face any of the above limitations. However, this solution illustrates control by a single operator/organization, which receives fees for its services and decides the incentives/penalties for the pool's miners.

Version 3. Finally, we replace the centralized mining pool with a decentralized mining pool, where miners decide to coordinate with each other without any operator. The following are the challenges for any decentralized mining pool-based solution (i) which miner decides the content of the block, (ii) how to fairly distribute rewards among the miners, and (iii) how to detect and penalize a malicious miner that delays block mining by not performing designated tasks

Overview. Our solution should offer the four appealing properties (G1 to G4). Consequently, we design PoC that requires no centralized organization and guards a PoS/BFT protocol from long-range attacks. In Figure 1, we illustrate the transactional flow.

(1) Transaction ordering and communication. PoC expects that the underlying PoS/BFT protocol reaches consensus on client transactions among its replicas and forwards every committed batch of transactions to the PoC miners.

(2) Block creation and mining. Once miners are ready to mine, they select a set of ordered batches to form a block. To allow miners to collaborate and reduce computational resource wastage, PoC ensures that all the miners are mining identical blocks, and each miner searches for the valid nonce on a unique subset of the search space. This collaboration helps us meet property G2.

(3) Nonce discovery and attestation. Once a miner discovers a nonce, it broadcasts the nonce to everyone. To ensure that there are no forks of the ledger, PoC requires the decentralized system's PoS/BFT protocol to attest a nonce. Note: this recursive dependency only helps to select a valid nonce without requiring multiple rounds of communication among the miners; the same task can be done by running a consensus on the nonce among the miners.

(4) Reward distribution. Once a miner finds a valid nonce, each miner receives an incentive. This reward distribution allows PoC to ensure fairness (property G3);

(5) Failure detection and progress. Byzantine miners may not search for nonce in their subset of the search space. If the valid nonce is present in this subset, then no miner will ever discover it. PoC allows miners to independently discover such malicious attacks and switches honest miners to different subsets to facilitate the discovery of the nonce.

(6) Penalty. PoC guarantees accountability by penalizing malicious miners that failed to find a valid nonce (property G4).

5 RELATED WORK

BFT has been studied extensively in the literature [6, 7, 15, 16, 35, 45, 47–49, 57, 58, 61, 64, 70, 75]. A sequence of efforts [12, 15, 33, 42, 46, 51, 59, 60, 63, 72, 73] have been made to reduce the communication cost of the BFTprotocols: (1) linearizing BFT consensus [31, 72], (2) optimizing for geo-replication [3, 36], and (3) sharding [4, 5, 21, 27, 65]. Nevertheless, all of these protocols face long-range attacks [24].

Alternatively, prior works have focussed on designing PoS protocols that permit the node with the highest stake to propose the next block [22, 30, 39, 41, 43] However, even these protocols suffer from long-range attacks if adversary has access to the private keys.

Existing work to protect against long-range attacks includes: (1) checkpointing the state through a trusted committee [8, 10, 13, 19], (2) key-evolving cryptographic techniques [26, 30, 39], (3) verifiable delay functions [71], and (4) appending the state to Bitcoin [9, 66].

(1) *Trusted Committee* solutions aim to periodically checkpoint the state of a PoS blockchain on another canonical chain, which is maintained by a committee of trusted members [8, 10, 13, 19]. These systems assume that the trusted members cannot be compromised, and thus, new nodes that wish to join the system can distinguish between the PoS blockchain and the canonical chain. Moreover, small size committees mimic a centralized system, while large committees increase latency for checkpoints.

(2) *Key-evolving cryptographic techniques* force participants to periodically discard old keys and generate new keys [26, 30, 39]. These works assume that honest nodes will discard their old keys after they generate a new pair; the onus is on the honest nodes.

(3) *Verifiable delay functions* provide proof that helps differentiate between a ledger created long ago versus a recently created adversarial ledger [71]. However, nothing prevents an adversary from initiating the creation of the adversarial ledger at the time of genesis. Once it has access to the private keys of other nodes, it can build blocks on top of this ledger, which makes it impossible for a new node to distinguish between the two ledgers.

(4) *Appending the state to Bitcoin* is a popular solution against long-range attacks for many recent papers [9, 66, 67]. Bitcoin employs PoW consensus, which requires miners to compete and thus wastes computational resources. Prior solutions to improve PoW or Bitcoin [28, 34, 44] do not eliminate this competition.

6 CONCLUSIONS

In this paper, we presented our novel PoC protocol, which, when appended to existing PoS/BFT protocols, guards them against long-range attacks. Like PoW, PoC makes it computationally expensive for an adversary to rewrite the ledger. However, unlike PoW, PoC introduces collaborative mining that requires miners to work with each other instead of competing.

REFERENCES

[1] 2011. P2Pool. http://p2pool.in/

[2] Marcos K. Aguilera, Idit Keidar, Dahlia Malkhi, Jean-Philippe Martin, and Alexander Shraer. 2010. Reconfiguring Replicated Atomic Storage: A Tutorial. *Bull. EATCS* 102 (2010), 84–108. http://eatcs.org/beatcs/index.php/beatcs/article/view/156

[3] Yair Amir, Claudiu Danilov, Jonathan Kirsch, John Lane, Danny Dolev, Cristina Nita-Rotaru, Josh Olsen, and David Zage. 2006. Scaling Byzantine Fault-Tolerant Replication to Wide Area Networks. In *International Conference on Dependable Systems and Networks (DSN'06)*. 105–114. https://doi.org/10.1109/DSN.2006.63

[4] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2019. Caper: a cross-application permissioned blockchain. *Proceedings of the VLDB Endowment* 12, 11 (2019), 1385–1398.

[5] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2021. *SharPer: Sharding Permissioned Blockchains Over Network Clusters.* Association for Computing Machinery, 76–88.

[6] Karolos Antoniadis, Antoine Desjardins, Vincent Gramoli, Rachid Guerraoui, and Igor Zablotchi. 2021. Leaderless Consensus. In *41st IEEE International Conference on Distributed Computing Systems.* IEEE, 392–402. https://doi.org/10.1109/ICDCS51616.2021.00045

[7] Pierre-Louis Aublin, Rachid Guerraoui, Nikola Knezevic, Vivien Quéma, and Marko Vukolic. 2015. The Next 700 BFT Protocols. *ACM Trans. Comput. Syst.* 32, 4 (2015), 12:1–12:45. https://doi.org/10.1145/2658994

[8] Sarah Azouvi, George Danezis, and Valeria Nikolaenko. 2020. Winkle: Foiling Long-Range Attacks in Proof-of-Stake Systems. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies.* Association for Computing Machinery, New York, NY, USA, 189–201. https://doi.org/10.1145/3419614.3423260

[9] Sarah Azouvi and Marko Vukolić. 2022. Pikachu: Securing PoS Blockchains from Long-Range Attacks by Checkpointing into Bitcoin PoW using Taproot. *arXiv preprint arXiv:2208.05408* (2022).

[10] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. 2012. Bitter to Better — How to Make Bitcoin a Better Currency. In *Financial Cryptography and Data Security.* Springer Berlin Heidelberg, Berlin, Heidelberg, 399–414.

[11] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. 2014. Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake [Extended Abstract]y. *SIGMETRICS Perform. Eval. Rev.* 42, 3 (dec 2014), 34–37. https://doi.org/10.1145/2695533.2695545

[12] Erik-Oliver Blass and Florian Kerschbaum. 2020. BOREALIS: Building Block for Sealed Bid Auctions on Blockchains. In *ASIA CCS '20: The 15th ACM Asia Conference on Computer and Communications Security.* ACM, 558–571. https://doi.org/10.1145/3320269.3384752

[13] Vitalik Buterin. 2014. Proof of Stake: How I Learned to Love Weak Subjectivity. https://blog.ethereum.org/2014/11/25/proof-stake-learned-love-weak-subjectivity

[14] Vitalik Buterin, Diego Hernandez, Thor Kamphefner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang, and Yan X. Zhang. 2020. Combining GHOST and Casper. *CoRR* abs/2003.03052 (2020). arXiv:2003.03052

[15] Miguel Castro and Barbara Liskov. 2002. Practical Byzantine Fault Tolerance and Proactive Recovery. *ACM Trans. Comput. Syst.* 20, 4 (2002), 398–461. https://doi.org/10.1145/571637.571640

[16] Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. 2023. How To Optimize My Blockchain? A Multi-Level Recommendation Approach. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–27.

[17] JPMorgan Chase. 2019. Quorum Whitepaper. https://github.com/ConsenSys/quorum/blob/master/docs/Quorum%20Whitepaper%20v0.2.pdf

[18] Cryptopedia Staff. 2023. What Was The DAO? https://www.gemini.com/cryptopedia/the-dao-hack-makerdao

[19] Phil Daian, Rafael Pass, and Elaine Shi. 2019. Snow White: Robustly Reconfigurable Consensus and Applications to Provably Secure Proof of Stake. In *Financial Cryptography and Data Security: 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18–22, 2019, Revised Selected Papers.* Springer-Verlag, Berlin, Heidelberg, 23–41. https://doi.org/10.1007/978-3-030-32101-7_2

[20] Nicholas Dana Troutman and Aron Laszka. 2021. PoolParty: Efficient Blockchain-Agnostic Decentralized Mining Pool. In *2021 The 3rd International Conference on Blockchain Technology.* Association for Computing Machinery, New York, NY, USA, 20–27. https://doi.org/10.1145/3460537.3460554

[21] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. 2019. Towards Scaling Blockchain Systems via Sharding. In *Proceedings of the 2019 International Conference on Management of Data.* ACM, 123–140. https://doi.org/10.1145/3299869.3319889

[22] Bernardo David, Peter Gaži, Aggelos Kiayias, and Alexander Russell. 2018. Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain. In *Advances in Cryptology – EUROCRYPT 2018*, Jesper Buus Nielsen and Vincent Rijmen (Eds.). Springer International Publishing, Cham, 66–98.

[23] Alex de Vries. 2018. Bitcoin's Growing Energy Problem. *Joule* 2, 5 (2018), 801–805. https://doi.org/10.1016/j.joule.2018.04.016

[24] Evangelos Deirmentzoglou, Georgios Papakyriakopoulos, and Constantinos Patsakis. 2019. A survey on long-range attacks for proof of stake protocols. *IEEE Access* 7 (2019), 28712–28725.

[25] Diem Association. 2022. Diem BFT. https://www.diem.com/en-us/

[26] Manu Drijvers, Sergey Gorbunov, Gregory Neven, and Hoeteck Wee. 2020. Pixel: Multi-signatures for Consensus. In *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2093–2110.

[27] Muhammad El-Hindi, Carsten Binnig, Arvind Arasu, Donald Kossmann, and Ravi Ramamurthy. 2019. BlockchainDB: A shared database on blockchains. *Proceedings of the VLDB Endowment* 12, 11 (2019), 1597–1609.

[28] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. 2016. Bitcoin-NG: A Scalable Blockchain Protocol. In *Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation* (Santa Clara, CA) *(NSDI'16)*. USENIX Association, USA, 45–59.

[29] Matthew K. Franklin. 2006. A survey of key evolving cryptosystems. *Int. J. Secur. Networks* 1, 1/2 (2006), 46–53. https://doi.org/10.1504/IJSN.2006.010822

[30] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*. Association for Computing Machinery, New York, NY, USA, 51–68. https://doi.org/10.1145/3132747.3132757

[31] Guy Golan Gueta, Ittai Abraham, Shelly Grossman, Dahlia Malkhi, Benny Pinkas, Michael Reiter, Dragos-Adrian Seredinschi, Orr Tamir, and Alin Tomescu. 2019. SBFT: A Scalable and Decentralized Trust Infrastructure. In *49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE. https://doi.org/10.1109/DSN.2019.00063

[32] Vincent Gramoli, Rachid Guerraoui, Andrei Lebedev, Chris Natoli, and Gauthier Voron. 2012. Diablo-v2: A Benchmark for Blockchain Systems. https://infoscience.epfl.ch/record/294268

[33] Guy Golan Gueta, Ittai Abraham, Shelly Grossman, Dahlia Malkhi, Benny Pinkas, Michael Reiter, Dragos-Adrian Seredinschi, Orr Tamir, and Alin Tomescu. 2019. Sbft: a scalable and decentralized trust infrastructure. In *2019 49th Annual IEEE/IFIP international conference on dependable systems and networks (DSN)*. IEEE, 568–580.

[34] Diksha Gupta, Jared Saia, and Maxwell Young. 2018. Proof of Work Without All the Work. In *Proceedings of the 19th International Conference on Distributed Computing and Networking* (Varanasi, India) *(ICDCN '18)*. ACM, New York, NY, USA, Article 6, 10 pages. https://doi.org/10.1145/3154273.3154333

[35] Suyash Gupta, Jelle Hellings, and Mohammad Sadoghi. 2021. *Fault-Tolerant Distributed Transactions on Blockchain*. Morgan & Claypool Publishers. https://doi.org/10.2200/S01068ED1V01Y202012DTM065

[36] Suyash Gupta, Sajjad Rahnama, Jelle Hellings, and Mohammad Sadoghi. 2020. ResilientDB: Global Scale Resilient Blockchain Fabric. *Proc. VLDB Endow.* 13, 6 (2020), 868–883. https://doi.org/10.14778/3380750.3380757

[37] Apache Incubator. 2023. ResilientDB: Global-Scale Sustainable Blockchain Fabric. https://github.com/apache/incubator-resilientdb

[38] Jonathan Katz and Yehuda Lindell. 2014. *Introduction to Modern Cryptography* (2nd ed.). Chapman and Hall/CRC.

[39] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In *Advances in Cryptology – CRYPTO 2017*. Springer International Publishing, Cham, 357–388.

[40] Sunny King and Scott Nadal. 2012. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. https://www.peercoin.net/whitepapers/peercoin-paper.pdf

[41] Sunny King and Scott Nadal. 2012. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August* 19, 1 (2012).

[42] Eleftherios Kokoris Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. 2016. Enhancing bitcoin security and performance with strong consistency via collective signing. In *25th usenix security symposium (usenix security 16)*. 279–296.

[43] Markulf Kohlweiss, Varun Madathil, Kartik Nayak, and Alessandra Scafuro. 2021. On the Anonymity Guarantees of Anonymous Proof-of-Stake Protocols. In *42nd IEEE Symposium on Security and Privacy*. IEEE, 1818–1833. https://doi.org/10.1109/SP40001.2021.00107

[44] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. 2016. Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing. In *Proceedings of the 25th USENIX Conference on Security Symposium* (Austin, TX, USA) *(SEC'16)*. USENIX Association, USA, 279–296.

[45] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong. 2009. Zyzzyva: Speculative Byzantine Fault Tolerance. *ACM Trans. Comput. Syst.* 27, 4 (2009), 7:1–7:39. https://doi.org/10.1145/1658357.1658358

[46] Lucas Kuhring, Zsolt István, Alessandro Sorniotti, and Marko Vukolić. 2021. StreamChain: Building a Low-Latency Permissioned Blockchain For Enterprise Use-Cases. In *2021 IEEE International Conference on Blockchain (Blockchain)*. 130–139. https://doi.org/10.1109/Blockchain53845.2021.00027

[47] Ahmed Lekssays, Giorgia Sirigu, Barbara Carminati, and Elena Ferrari. 2022. MalRec: A Blockchain-Based Malware Recovery Framework for Internet of Things. In *Proceedings of the 17th International Conference on Availability, Reliability and Security* (Vienna, Austria) *(ARES '22)*. Association for Computing Machinery, New York, NY, USA, Article 99, 8 pages. https://doi.org/10.1145/3538969.3544446

[48] Shengyun Liu, Paolo Viotti, Christian Cachin, Vivien Quéma, and Marko Vukolic. 2016. XFT: Practical Fault Tolerance beyond Crashes. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*. USENIX Association, USA, 485–500.

[49] Dumitrel Loghin, Tien Tuan Anh Dinh, Aung Maw, Chen Gang, Yong Meng Teo, and Beng Chin Ooi. 2022. Blockchain Goes Green? Part II: Characterizing the Performance and Cost of Blockchains on the Cloud and at the Edge. *arXiv preprint arXiv:2205.06941* (2022).

[50] Loi Luu, Yaron Velner, Jason Teutsch, and Prateek Saxena. 2017. SmartPool: Practical Decentralized Pooled Mining. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 1409–1426.

[51] Mads Frederik Madsen, Mikkel Gaub, Malthe Ettrup Kirkbro, and Søren Debois. 2019. Transforming Byzantine Faults using a Trusted Execution Environment. In *15th European Dependable Computing Conference*. 63–70. https://doi.org/10.1109/EDCC.2019.00022

[52] Satoshi Nakamoto. 2009. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf

[53] Rahul Nambiampurath and Kyle Baird. 2023. 2,000 Crypto Private Keys Stolen From Edge Wallet. https://beincrypto.com/2000-crypto-private-keys-stolen-edge-wallet/

[54] Margaux Nijkerk. 2023. Ethereum Unstaking Requests Now Face About a 17-Day Wait. https://www.coindesk.com/tech/2023/04/18/ethereum-unstaking-requests-now-face-about-a-17-day-wait/

[55] Krzysztof Okupski. 2016. Bitcoin Developer Reference. https://github.com/minium/Bitcoin-Spec

[56] Elena R. 2024. Private Key Breaches Surge in 2024 with Over $239 Million Stolen! Is Your Crypto Safe? https://coinpedia.org/news/private-key-thefts-skyrocket-in-q1-2024-with-over-239-million-stolen-in-just-three-months/

[57] Christian Rondanini, Barbara Carminati, Federico Daidone, and Elena Ferrari. 2020. Blockchain-based controlled information sharing in inter-organizational workflows. In *2020 IEEE International Conference on Services Computing (SCC)*. 378–385. https://doi.org/10.1109/SCC49832.2020.00056

[58] Pingcheng Ruan, Tien Tuan Anh Dinh, Qian Lin, Meihui Zhang, Gang Chen, and Beng Chin Ooi. 2021. LineageChain: a fine-grained, secure and efficient data provenance system for blockchains. *VLDB J.* 30, 1 (2021), 3–24. https://doi.org/10.1007/s00778-020-00646-1

[59] Vasily A. Sartakov, Stefan Brenner, Sonia Ben Mokhtar, Sara Bouchenak, Gaël Thomas, and Rüdiger Kapitza. 2018. EActors: Fast and flexible trusted computing using SGX. In *Proceedings of the 19th International Middleware Conference*, Paulo Ferreira and Liuba Shrira (Eds.). ACM, 187–200. https://doi.org/10.1145/3274808.3274823

[60] Youren Shen, Hongliang Tian, Yu Chen, Kang Chen, Runji Wang, Yi Xu, Yubin Xia, and Shoumeng Yan. 2020. *Occlum: Secure and Efficient Multitasking Inside a Single Enclave of Intel SGX*. Association for Computing Machinery, New York, NY, USA, 955–970.

[61] Peiyao Sheng, Gerui Wang, Kartik Nayak, Sreeram Kannan, and Pramod Viswanath. 2021. BFT Protocol Forensics. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1722–1743. https://doi.org/10.1145/3460120.3484566

[62] Myles Sherman. 2021. Nodes on Bitcoin's Lightning Network Double in 3 Months. https://www.coindesk.com/tech/2021/07/14/nodes-on-bitcoins-lightning-network-double-in-3-months/

[63] Man-Kit Sit, Manuel Bravo, and Zsolt István. 2021. An Experimental Framework for Improving the Performance of BFT Consensus for Future Permissioned Blockchains. In *Proceedings of the 15th ACM International Conference on Distributed and Event-Based Systems* (Virtual Event, Italy) *(DEBS '21)*. Association for Computing Machinery, New York, NY, USA, 55–65. https://doi.org/10.1145/3465480.3466922

[64] Chrysoula Stathakopoulou, Matej Pavlovic, and Marko Vukolić. 2022. State Machine Replication Scalability Made Simple. In *Proceedings of the Seventeenth European Conference on Computer Systems*. Association for Computing Machinery, New York, NY, USA, 17–33. https://doi.org/10.1145/3492321.3519579

[65] Florian Suri-Payer, Matthew Burke, Zheng Wang, Yunhao Zhang, Lorenzo Alvisi, and Natacha Crooks. 2021. Basil: Breaking up BFT with ACID (Transactions). In *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles (SOSP '21)*. Association for Computing Machinery, 1–17. https://doi.org/10.1145/3477132.3483552

[66] E. Tas, D. Tse, F. Gai, S. Kannan, M. Maddah-Ali, and F. Yu. 2023. Bitcoin-Enhanced Proof-of-Stake Security: Possibilities and Impossibilities. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 126–145. https://doi.org/10.1109/SP46215.2023.10179426

[67] Ertem Nusret Tas, David Tse, Fisher Yu, and Sreeram Kannan. 2022. Babylon: Reusing Bitcoin Mining to Enhance Proof-of-Stake Security. *arXiv preprint arXiv:2201.07946* (2022).

[68] Yongge Wang, Jingwen Sun, Xin Wang, Yunchuan Wei, Hao Wu, Zhou Yu, and Gillian Chu. 2020. Sperax: An Approach To Defeat Long Range Attacks In Blockchains. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 574–579. https://doi.org/10.1109/INFOCOMWKSHPS50562.2020.9163036

[69] Gavin Wood. 2015. Ethereum: A secure decentralised generalised transaction ledger. http://gavwood.com/paper.pdf

[70] Zihang Xiang, Tianhao Wang, Wanyu Lin, and Di Wang. 2023. Practical Differentially Private and Byzantine-resilient Federated Learning. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–26.

[71] Anatoly Yakovenko. 2019. Solana: A new architecture for a high performance blockchain v0.8.13. https://solana.com/solana-whitepaper.pdf

[72] Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan Gueta, and Ittai Abraham. 2019. HotStuff: BFT Consensus with Linearity and Responsiveness. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*. ACM, 347–356. https://doi.org/10.1145/3293611.3331591

[73] Rui Yuan, Yubin Xia, Haibo Chen, Binyu Zang, and Jan Xie. 2018. ShadowEth: Private Smart Contract on Public Blockchain. *J. Comput. Sci. Technol.* 33, 3 (2018), 542–556. https://doi.org/10.1007/s11390-018-1839-y

[74] Yohan Yun. 2024. Lazarus Group's favorite exploit revealed — Crypto hacks analysis. https://cointelegraph.com/magazine/north-korean-hackers-private-keys-flash-loan-attacks/

[75] Ce Zhang, Cheng Xu, Jianliang Xu, Yuzhe Tang, and Byron Choi. 2019. GEM^2-Tree: A Gas-Efficient Structure for Authenticated Range Queries in Blockchain. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. 842–853. https://doi.org/10.1109/ICDE.2019.00080