Cryptography-Based Bootstrapping Scheme for Permissionless Sharding Blockchain

Ziqiang Xu, Ahmad Salehi S., and Naveen Chilamkurti
Department of Computer Science and Information Technology, La Trobe University, Melbourne, Australia
Email: {z.xu, a.salehishahraki, n.chilamkurti}@latrobe.edu.au

***Abstract*—This paper proposes a secured bootstrapping protocol based on BLS aggregated signatures. The protocol provides a secure, efficient identity endorsement and verification scheme for permissionless sharding blockchain and effectively prevents the Sybil attack during the bootstrapping process. Our results show that our scheme is secure and supports large-scale networks. Our scheme performs better than the current methods in preventing the Sybil attack.**

***Index Terms*—Blockchain, Security, Cryptography**

I. Introduction

Sharding Blockchain is a scalability improvement designed to increase the throughput and performance of the blockchain network [1]. Sharding blockchain achieves parallel processing by dividing the entire network into multiple independent shards. Each shard can ensure the consistency of the whole network through cross-shard communication and coordination mechanisms. This design reduces the weight of the consensus process on each node and provides higher scalability for the development of blockchain technology.

There are three primary phases to consider when building a sharding blockchain: bootstrapping, consensus, and reconfiguration. During the bootstrapping phase, the infrastructure of the entire sharding blockchain must be established. This includes determining the overall structure of the network, creating the initial shards and assigning nodes. The consensus phase is critical to ensuring consistency and security across the shards of the network, including intra-shard consensus and cross-shard communication. After the sharding blockchain runs for a period, reconfiguration is required to fix potential problems and avoid sharding faults.

This paper focuses on a cryptography-based feasible scheme for the bootstrapping phase in a permissionless sharding blockchain. The bootstrapping phase involves node registration and identity verification, ensuring the network nodes are legitimate. The biggest threat to this process is the Sybil attack. The Sybil attack is an attack in distributed systems in which an attacker attempts to control the system or disrupt normal operations by disguising multiple false identities. In sharding blockchain, many false identities may lead to faulty nodes taking the majority share and affecting or controlling the consensus mechanism of the entire network.

In a permissionless sharding blockchain, the current bootstrapping approach utilises PoW and its variants for identity authentication [2]. Nodes prove their workload by solving complex mathematical problems so that the nodes in the network recognize their identity. However, running PoW causes a high load on the nodes as the nodes need to find the hash value that matches a particular condition. If we want to increase the efficiency of PoW, then we need to reduce the computational difficulty. This may lead to selfish mining. Therefore, balancing the security and efficiency of PoW bootstrapping is a difficult problem.

Our scheme aims to address the problem of balancing the security and efficiency of the bootstrapping process in a permissionless sharding blockchain. The scheme uses a deposit as a guarantee of identity and endorses the authenticity of the identity through mutual signatures.

II. Propose Scheme

Our proposed scheme has three key components. 1. A global state ledger. Nodes participating in the network have copies of this ledger. This ledger records all valid endorsement signature information. 2. Efficient Randomized Endorsement Groups. Each node in the network is assigned an exclusive endorsement group. This endorsement group is designed to verify the deposit's authenticity and validate or invalidate the key pair. 3. Verifiable Aggregate Signatures. A protocol based on BLS aggregated signatures is designed to offer low computational complexity and high verification efficiency.

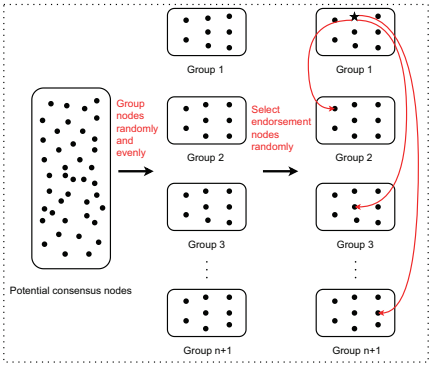

Fig. 1: Grouping Endorsement Groups

Fig. 1 illustrates selecting endorsement groups. All participating nodes within the network are disrupted and evenly divided into $n+1$ candidate subgroups. Each subgroup contains k nodes. Then, for each node, a node is randomly selected

2833-292X/24/$31.00 ©2024 IEEE
DOI 10.1109/DSN-S60304.2024.00043

from each of the other candidate subgroups, which means that each node will have n endorsement nodes.

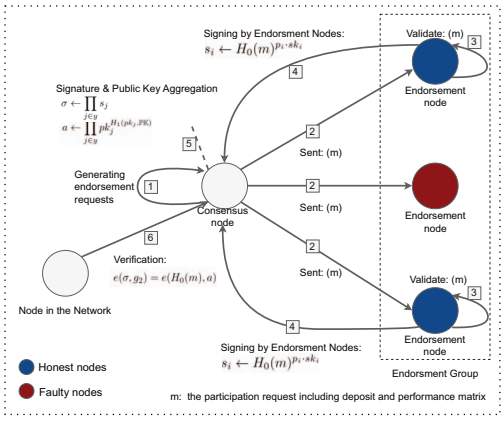

Fig. 2: Endorsing process

Fig. 2 illustrates the endorsement process. During the endorsement process, each consensus node submits a participation request $Tx : \{pk, TokenAddress\}$ to their n endorsement nodes, where pk is the node's public key waiting to be activated. The token address is the deposit. Upon receiving Tx, the endorsement nodes verify the token's authenticity. The Tx that passes the verification will be signed by the endorsement node. When the consensus node collects endorsement signatures from more than $\frac{2n}{3}$, its public key will be activated, and the aggregate signature will be published to the global state ledger. The specific scheme for endorsement signature is as follows:

Preliminaries. Assuming a non-degenerate, efficiently computable, bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_t$ are groups of prime order q. Let g_1 and g_2 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively.

Definition 1. Hash functions: $H_0 : \{0,1\}^* \to \mathbb{G}_1$, and $H_1 : \{0,1\}^* \to \mathbb{Z}_q$.

Definition 2. View $V_i : f(pk_i, \mathbb{PK})$ represents a view associated with a specific public key pk_i, where $\mathbb{PK} = \{pk_1, pk_2, \ldots, pk_n\}$ constitutes a set of public keys.

Definition 3. Vector $c_i : [pk_1 = 0, pk_2 = 1, \ldots, pk_n = 0]$, where the set of pk are all elements of $\mathbb{PK}$. Each entry of vector c_i is assigned by the mapping function $f : \{\text{true}, \text{false}\} \to \{1, 0\}$, wherein true is mapped to 1, and false is mapped to 0.

Key Generation. The key generator algorithm, $KeyGenerator()$, generates sk and pk through $sk \leftarrow \text{random}(\mathbb{Z}_q)$ and $pk \leftarrow g_2^{sk} \in \mathbb{G}_2$.

Signing. The signature algorithm, $Sign(sk_i, m, PK)$, produces the signature s_i during the signing process, with its computation outlined as $s_i \leftarrow H_0(m)^{p_i \cdot sk_i}$.

Where $p_i \leftarrow H_1(pk_i, \mathbb{PK})$, m represents the message signed, while sk_i and pk_i denote the signer's key pair.

Signature Aggregation. The aggregated signature, $SignAgg(c_i, \{s_1, s_2, \ldots, s_n\})$, combines all collected signatures s_i and produces the aggregated signature $\sigma \leftarrow \prod_{j \in y} s_j$.

where $y = c_i : \{pk_i | pk_i = 1\}$ denotes the set of all pk in c_i with $pk_i = 1$.

Public Key Aggregation. Aggregating public keys is a critical component for verifying the authenticity of signatures. $PKAgg(c_i, \mathbb{PK})$ generates the aggregated public key $a \leftarrow \prod_{j \in y} pk_j^{H_1(pk_j, \mathbb{PK})}$.

Signature Verification. $Verify(\sigma, a, m)$ outputs success when $e(\sigma, g_2) = e(H_0(m), a)$ holds.

III. Evaluation

In this section, we describe the security and efficiency of the scheme. For signature security, we consider a scenario where the total number of nodes is $n = 3f + 1$, and f adversary nodes exist. Several f adversary nodes collude to assist one of them in forging the signatures of the remaining honest nodes. Suppose we aim to forge the signature of sk_1, requiring us to find an s_1 such that $e(s_1, g_2) = e(H_0(m), pk_1^{H_1(pk_1, \mathbb{PK})})$. Where $pk_1 = g_2^{sk_1 H_1(pk_1, \mathbb{PK})}$, rearranging the equation, we arrive at $s_1 = H_0(m)^{sk_1 H_1(pk_1, \mathbb{PK})}$. When $H_1(pk_1, \mathbb{PK})$ is known, forging s_1 necessitates finding an integer z that satisfies $z \cdot H_1(pk_1, \mathbb{PK}) \equiv 1 \pmod q$. However, discovering an integer z to ensure that $s_1 \in G_1$ and satisfies $e(s_1, g_2) = e(H_0(m), pk_1^{H_1(pk_1, \mathbb{PK})})$ is the Computational Diffie-Hellman (CDH) problem. Consequently, accomplishing this task within a given time is virtually infeasible. Therefore, the signature scheme is secure.

For efficiency, the complexity of each node's communication during our scheme's bootstrapping has a fixed constant $O(c)$, where c is determined by the number of candidate groups. Compared to PoW, which requires the selection of a reference committee, the nodes in the committee will be responsible for verifying the PoW results, with a complexity of $O(n2)$. Therefore, our bootstrapping process is efficient.

IV. Conclusion

This paper proposes a secure and efficient cryptography-based permissionless sharding blockchain bootstrapping method. Compared to PoW, this method reduces the demand for device performance for the bootstrapping process. In large-scale networks, this research overcomes the efficiency and sustainability issues when using PoW bootstrapping and provides an innovative approach for the widespread use of permissionless sharding blockchains.

References

[1] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 17–30, 2016.

[2] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: Scaling blockchain via full sharding," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 931–948, 2018.

[3] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 514–532, Springer, 2001.