This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/SANER56733.2023.00105, 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)

An Optimized Concurrent Proof of Authority Consensus Protocol

A. Nazir[1], M. Singh[1], G. Destefanis[2], J. Memon[1], R. Neykova[2], M. Kassab[3] and R. Tonelli[4]
[1]*Cobe LTD*, U.K. {anjum, michael, jamshed}@cobe.network
[2]*Brunel University London*, U.K. {giuseppe.destefanis, rumyana.neykova}@brunel.ac.uk
[3]*Pennsylvania State University*, muk36@psu.edu
[4]*University of Cagliari, IT*, roberto.tonelli@unica.it

***Abstract*—Security and reliability in Blockchain software systems is a major challenge in Blockchain Oriented Software Engineering. One of the most critical components to address at the architectural level is the consensus protocol, as it serves as the mechanism for accepting valid transactions and incorporating them into the ledger history. Given that this process is executed by specific blockchain nodes, it is crucial to consider them as a key point of focus for ensuring the integrity of the entire blockchain history. This paper addresses the major challenge of security and reliability in Blockchain software systems by proposing a new protocol for Permissioned Concurrent Proof of Authority (CPoA). This protocol involves selecting a group of nodes as authority nodes, responsible for validating new identities, blocks, and transactions. The protocol is integrated with a framework that subjects validators to a unique eligibility criterion and a combination of reputation, security score, online aging, and general performance indicators related to node reliability, significantly reducing the risk of validator misbehavior and enhancing security, reliability and confidentiality of the entire blockchain compared to other existing approaches.**

***Index Terms*—consensus, blockchain, concurrency**

I. Introduction

Blockchain Oriented Software Engineering (BOSE) encompasses various aspects of blockchain technology [1; 5; 6]. One of the main challenges in blockchain systems, as with any distributed and decentralized software system, is to ensure security and reliability in a distributed environment where the attack surface is somewhat out of control. BOSE principles aid in identifying and addressing security issues. In blockchain networks, where the system relies on the participation of multiple actors who share computational load and data storage, various approaches have been developed [4; 9]. We focus on permissioned blockchain, where nodes can only join the network upon authorization, allowing the network to exercise some control over participants at the entry point.

Synchronizing ledger instances across various nodes is a fundamental challenge for distributed ledger technologies, which is addressed through consensus protocols in blockchain technologies. Consensus protocols are crucial for security, scalability, and high throughput. Permissionless (public or open) blockchains allow access to anyone, participants don't know or trust each other. On the other hand, Permissioned blockchains have restricted access to vetted participants who already know and trust each other and are held accountable through off-chain legal contracts and agreements [7].

An additional level of control may also be implemented over node categories, such that only a subset of the network's nodes, known as "validators," can add blocks and transactions to the ledger.

In this configuration, it is crucial to analyze from BOSE principles the best architectural choices at the system level that make the blockchain network more robust against attacks and faults and more reliable.

In this work, we discuss Cobe's proposal for a consensus algorithm that aims to enhance and improve single-node reliability for validators.

Cobe's blockchain is a new technology proposal that aims to create a comprehensive cross-border trade ecosystem while incorporating state-of-the-art innovations in the blockchain field at different levels, including consensus protocols.

Cobe aims to provide a dual blockchain solution, where both native permissioned and permissionless chains are built to work in synergy. Cobe's proposal includes various components such as a dual-sided blockchain architecture that includes both permissioned and permissionless chains working in harmony, a blockchain with Cobe's Concurrent Proof of Authority (CPoA) consensus, a blockchain with Cobe's Concurrent Delegate Proof of Stake (CDPoS) consensus, and the Cobe's Confidentiality Enhancement Framework (CCEF) to ensure high confidentiality and security in a CPoA blockchain, among other unique features.

In this paper, we present Cobe's Permissioned Concurrent Proof of Authority (CPoA) and the Cobe's Confidentiality Enhancement Framework (CCEF) and discuss the expected benefits of their adoption in blockchain technology.

II. Cobe's Permissioned Concurrent Proof of Authority (CPoA) Blockchain

Popular mechanisms to reach consensus in decentralized blockchains include Proof of Work, Proof of Stake, and Proof of Authority. In Proof of Work, miners compete to solve a mathematical puzzle, using computational resources. The first miner to solve the puzzle broadcasts the solution, and the block is added to the ledger. Security is provided by the difficulty of the puzzle. In Proof of Stake, validators are chosen in turns to propose new blocks and must stake cryptocurrency, making the protocol crypto-economically secure. In Proof of Authority, validators are chosen and authorized to propose blocks, and there is a rotation procedure to prevent a single validator from validating consecutive blocks. Incentives to behave well are based on a reputation mechanism. Proof of Authority is a consensus mechanism where nodes provide identity information to become validators, allowing for faster validation of blocks but making the protocol more centralized compared to Proof of Stake blockchains [**?**].

In this section, we present new features that are introduced in Cobe's Concurrent Proof of Authority (CPoA) consensus protocol. After that, the Cobe Confidentiality Enhancement Framework (CCEF) is introduced. This framework ensures high confidentiality and security in a CPoA blockchain.

A. New Features

The CPoA protocol uses a unique eligibility criterion to select nodes for validating blocks. It utilizes the Proof of Turn (PoT) mechanism for block creation and takes into account the reputation score of the node. Note that PoT is another research proposal of Cobe's blockchain and is not covered in this paper. The Cobe's CPoA protocol also introduces a new confidentiality enhancement framework to ensure the high confidentiality and security of users' data. These features will be discussed in more detail in the proceeding sections.

B. Eligibility Criteria (E_c)

The eligibility criterion is one of the most important parameters of any PoA-based blockchain. In Cobe's CPoA chain, node eligibility is calculated using Eq. 1. This involves achieving a minimum threshold value for the parameters presented in the equation.

$$E_c = \Xi * CBE_{Min} * \kappa \tag{1}$$

$where$
Ξ = Validator's identification value (0 or 1)
CBE_{Min} = Minimum (CBE) Account Balance a validator must maintain
κ = Security score of the node.

1) Formal Identification and Authorization (Ξ): All nodes must go through a mandatory authentication process before joining the CPoA chain network. They must submit documentary evidence to verify their identity. After passing a thorough verification process, a node can become a validator. Nodes in the CPoA chain are selected based on compliance with the identity stack. This one-time process assigns a unique identity to the node that serves as an identifier in all communication within the network.

2) Nodes Stake Size (σ_s): A node that wants to become a validator in Cobe's CPoA blockchain must hold a certain minimum number of CBE (Cobe's native coin) in its account. At the beginning a node must have at least 350,000 CBE. Any node that does not meet this criterion will not be eligible.

3) Security Score (κ): The security score is used to measure the trustworthiness of a node. It can be calculated by using Eq. 2. It includes three parameters: (i) Vulnerability Score (ν_s), (ii) Firewall Score (ϕ_s), and (iii) Remote Attestation Score (α_s).

$$\kappa = \omega_1 * \nu_s + \omega_2 * \phi_s + \omega_3 * \alpha_s \tag{2}$$

$where$
ω_1, ω_2 and ω_3 are weights assigned to each parameter, ω_1 + ω_2 + ω_3 = 1

All nodes that fulfil Cobe's CPoA eligibility criteria can become validators. However, Cobe's CPoA blockchain will allow only a limited number of validators initially. It will commence with 21 validators, and the number will increase in the future.

C. Nodes' Reputation Score (ρ)

In the first round, all selected validators will create the same number of blocks in a round-robin fashion. Over time, each validator's reputation score is built. This reputation score will be used to determine how many blocks a node can validate. A lower score will result in a lower block creation rate and thus, a lower reward. If a node's reputation score falls below a minimum threshold, the node may be removed from participating in the network altogether. The reputation score for a validator in Cobe's CPoA chain is calculated by using Eq. 3.

$$\rho = \omega_1 * O + \omega_2 * \kappa - \omega_3 * \beta_{missed} - \omega_4 * \beta_{bad} \tag{3}$$

$where$
ω_1, ω_2, ω_3 and ω_4 are weights assigned to each parameter
O = Online age
κ = Security score
β_{missed} = Blocks missed by the node
β_{bad} = Bad blocks created by the node.

1) Online Age (O): Online age is an important measure that shows a node's reliability and consistency in the blockchain network. Online age can be measured from the time the node has been online during the last 'x' epochs. Online age can be calculated as follows:

$$O = \frac{\tau_o}{\tau_t} \tag{4}$$

$where$
τ_o = No. of epochs a node remain online
τ_t = Total epochs under consideration.

2) Missed Blocks (β_{missed}): Blocks missed by a node affect the throughput of the blockchain network, particularly if a node fails to create a block as per the PoT block creation schedule. This will affect the reputation of node negatively.

3) Bad Blocks (β_{bad}): Each wrong or incorrect block created by an authorized validator effects its reputation score and eligibility negatively. At the end of each cycle/round, all incorrect blocks created by a validator are counted. Only a certain number of incorrect blocks are allowed, attributed to external causes, e.g., network delay. Each wrong/incorrect block and blocks missed by a node lead to a significant reduction in the earned reward.

4) Security Score (κ): Security is an important criterion not only for validators' eligibility but also to measure their reputation. Cobe's CPoA protocol uses the security score presented, which is calculated by using Eq. 2.

5) Nodes' Validation Share Score (VSS): In Cobe's CPoA chain, a node's validation share score can be calculated by using the reputation of the node (ρ) and the node's stake size (σ_s). The reputation score and the validation share score can be calculated by using Eq. 3 and Eq. 5 respectively. The node's validation share can be calculated by using Eq. 6.

$$VSS = \sigma_s + RS \tag{5}$$

$where$
VSS = Validation Share Score.

$$VS = \left\lfloor \left(\frac{VSS}{\sum_{i=1}^{n} VSS} \right) \beta_{Max} \right\rfloor \tag{6}$$

$where$
VS= Validation Share
n= Total number of elected validators
β_{Max} = Maximum blocks can be created in an epoch.

D. Cobe's Confidentiality Enhancement Framework (CCEF)

Cobe's CPoA chain uses Homomorphic Encryption (HE) to ensure data confidentiality. Homomorphic encryption allows for operations on encrypted data without the need for decryption, thus validators can validate transactions in their encrypted state, maintaining data confidentiality. In this section the working of the Cobe Confidentiality Enhancement Framework (CCEF) is presented. Cobe's CPoA blockchain has introduced two novel techniques to ensure data confidentiality and privacy.

1) Homomorphic Encryption (HE): Homomorphic Encryption (HE) allows for computations on encrypted data without the need for decryption, preventing unauthorized information disclosure. Different types of HE include Partially Homomorphic Encryption (PHE), which allows for one computational operation an unlimited number of times, Somewhat Homomorphic Encryption (SHE), which allows for several types of operations but only a certain number of times, and Fully Homomorphic Encryption (FHE), which supports arbitrary computation any number of times, but has poor performance [8]. FHE can eliminate the trade-off between data usability and data privacy, but requires no trusted third parties and is quantum safe.

2) Zero Knowledge Proof (ZKP): Zero Knowledge Proof (ZKP) allows a party to prove to another party that a given statement is true without revealing any additional information. In blockchain, it enables a payer to prove the validity of a payment without disclosing the payment's details. For example, Zcash [3] uses Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs) [2] to preserves a user's payment privacy. Cobe's CPoA chain uses zk-SNARKs to verify that the payer has sufficient balance in their account.

3) Framework Operation: Cobe CPoA chain uses FHE and ZKP together to ensure the confidentiality and privacy of user data. The fundamental operations of encrypted transactions are presented below.

1) Let's say Alice wants to pay some CBE *'m'* to Bob. She creates a transaction and submits it to the transaction pool.
2) This transaction includes (i) a non-interactive zero knowledge proof about her account balance *'y'* and (ii) the encrypted transaction amount. The encrypted transaction amount is generated with the help of Bob's homomorphic public key.
3) The transaction is submitted to the transaction pool, from where a validator will pick and process it.
4) The validator will validate the transaction in phases.
   - It will verify the account balance claim Alice is making by using zk-SNARKs.
   - After verifying the available balance in Alice's account, the validator will perform the required operation, e.g., credit the amount into Bob's account. This will be achieved with the help of the homomorphic encryption scheme discussed above.
5) After performing the necessary operations on the ciphertext, the validator will create the block, add it to the ledger, and forward it to other validators on the network.
6) Bob can verify the amount and can perform aggregate operations on the credited amount directly on the wallet information. The complete process is demonstrated in Figure 1.

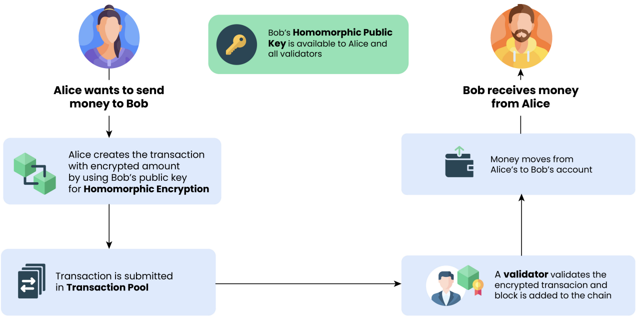
Bob's **Homomorphic Public Key** is available to Alice and all validators
Alice wants to send money to Bob
Bob receives money from Alice
Alice creates the transaction with encrypted amount by using Bob's public key for **Homomorphic Encryption**
Money moves from Alice's to Bob's account
Transaction is submitted in **Transaction Pool**
A **validator** validates the encrypted transacion and block is added to the chain

Figure 1: Transaction processing flow in CPoA.

4) Cobe Homomorphic Encryption Scheme: Proc. `ParamGen()` is used to generate the necessary parameters required for encryption keys.

This article has been accepted for publication in a future proceedings of this conference, but has not been fully edited. Content may change prior to final publication. Citation information: DOI: 10.1109/SANER56733.2023.00105, 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)

```
Procedure ParamGen()
  Data: Security parameter $\lambda$, $PT$, $K$, $B$
  Result: Params
```

$where$
λ: denotes the desired security level of the scheme; for instance, 128-bit or 256-bit security
PT: denotes plaintext
K: represents the dimension of the vectors to be encrypted
B: represents auxiliary parameter that is used to control the complexity of the program.

Encryption parameters generated via Proc. `ParamGen()` will be used to generate encryption keys as shown in Proc. `KeyGen(Params)`.

```
Procedure KeyGen(Params)
  Data: Key generation parameters $Params$
  Result: SK, PK, EK
```

$where$
SK: represents secrete key. SK will be used to decrypt the ciphertext
PK: represents public key. PK will be used to encrypt the PT
EK: represents evaluation key. EK will be used to perform homomorphic operations over the ciphertext.

Encryption of the given message will be performed by the encryption procedure (see Proc. `Encrypt(m,PK)`).

```
Procedure Encrypt(m,PK)
  Data: Encryption parameters $m, PK$
  Result: c
```

$where$
m: represents the message to be encrypted
PK: represents the homomorphic public key of the recipient
c: represents ciphertext.

When a transaction is received by a validator, it may be required to perform various computations over the ciphertext. A generic evaluation method is represented in Proc. `Eval(c,EK,Params)`.

```
Procedure Eval(c,EK,Params)
  Data: Parameters $c, EK, Params$
  Result: $c^{'}$
```

$where$
$c^{'}$: represents new ciphertext generated after performing some operation by the validator.

When the new ciphertext is received by the recipient, the recipient will decrypt it by using the decryption algorithm presented in Proc. `Decrypt(`$c^{'}$`,SK)`.

```
Procedure Decrypt($c^{'}$,SK)
  Data: Parameters $c, SK$
  Result: $m^{'}$
```

$where$
$m^{'}$: is the new message generated after computing over the ciphertext. It will be equivalent to PT generated after performing same operation.

III. Conclusion

Blockchain Oriented Software Engineering (BOSE) principles are crucial for ensuring the security and reliability of blockchain systems, particularly in permissioned networks. Cobe's proposal for a dual blockchain solution incorporates novel features, such as a dual-sided architecture and consensus algorithms, to enhance confidentiality and security. The proposed solution is expected to yield improved single node reliability for validators and increased confidentiality and security for the overall network.

References

[1] Bartolucci, S., Destefanis, G., Ortu, M., Uras, N., Marchesi, M., Tonelli, R.: The butterfly "affect": Impact of development practices on cryptocurrency prices. EPJ Data Science **9**(1), 21 (2020)

[2] Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M.: Succinct {Non-Interactive} zero knowledge for a von neumann architecture. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 781–796 (2014)

[3] Hopwood, D., Bowe, S., Hornby, T., Wilcox, N.: Zcash protocol specification. GitHub: San Francisco, CA, USA p. 1 (2016)

[4] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review p. 21260 (2008)

[5] Pierro, G.A., Rocha, H., Ducasse, S., Marchesi, M., Tonelli, R.: A user-oriented model for oracles' gas price prediction. Future Generation Computer Systems **128**, 142–157 (2022)

[6] Porru, S., Pinna, A., Marchesi, M., Tonelli, R.: Blockchain-oriented software engineering: challenges and new directions. In: 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C). pp. 169–171. IEEE (2017)

[7] Rauchs, M., Hileman, G., et al.: Global cryptocurrency benchmarking study. Cambridge Centre for Alternative Finance Reports (2017)

[8] Regueiro, C., Seco, I., de Diego, S., Lage, O., Etxebarria, L.: Privacy-enhancing distributed protocol for data aggregation based on blockchain and homomorphic encryption. Information Processing & Management **58**(6), 102745 (2021)

[9] Wood, G., et al.: Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper **151**(2014), 1–32 (2014)