ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

PPRCA: Privacy-Preserving of Raft Consensus Algorithm a Next-Generation Consensus for Distributed Network

Wed Kadhim Oleiwi[1*]

[1] Department of Computer Science, Science College for Women, University of Babylon, wsci.wed.kadhum@uobabylon.edu.iq, Hillah 51002, Iraq
***Corresponding author email:** wsci.wed.kadhum@uobabylon.edu.iq
: 07818219571

خوارزمية الإجماع الموزع مع الحفاظ على الخصوصية – الجيل القادم من خوارزميات الإجماع للشبكات الموزعة

ود كاظم عليوي[1*]

[1] كلية العلوم لبنات قسم الحاسبات، جامعة بابل, wsci.wed.kadhum@uobabylon.edu.iq,51002،بابل، العراق

Accepted: **22/1/2025** **Published:** **31/3/2025**

ABSTRACT

Background: Even while traditional Raft is effective at leader election and log replication, it is not appropriate for sensitive applications like supply chains, financial systems, or healthcare because it lacks built-in privacy safeguards.

Materials and Methods: A privacy-preserving Raft consensus method is proposed to solve the privacy issues that occur when private information is transferred between nodes in a distributed system such as a blockchain. Raft itself, by default, does not provide any steps toward ensuring data confidentiality during consensus. By employing privacy-preserving cryptographic techniques like homomorphic encryption and zero-knowledge proofs, nodes can reach consensus while keeping sensitive data private.

Results: Traditional Raft performs much better in scenarios where performance matters, while Privacy-Perving Raft works better in a sensitive application to privacy (the average of write throughput is 5% lower than that of traditional Raft) and CPU is 40-60%.

Conclusion: Based on the gained privacy by some computational costs, it will be valid to draw the conclusion that this works for privacy-sensitive applications within decentralized systems with these performance and security analyses.

Key words: Raft Consensus Algorithm; Privacy Preservation; Zero-Knowledge Proofs (ZKPs); Homomorphic Encryption (HE); Distributed Network.

INTRODUCTION

Applications of distributed ledger technologies, on the other hand, move at warp speed, but require managing private data, including financial transactions, medical records, and identity management, which requires data confidentiality and integrity[1]. Raft has emerged as one of the most adopted protocols in an attempt to realize fault tolerance with consistent states across distributed systems. However, traditional Raft primarily works on log replication efficiency and leader election but possibly at the cost of several critical privacy concerns inherently associated with sharing sensitive data amongst participating nodes themselves[2].

These were designed to guarantee that the majority of network nodes can achieve the required consensus concerning states and also withstand certain kinds of error during progress [3]. They are considered one of the fundamental building blocks of distributed systems since they guarantee fault-tolerance, safety, and liveness [4]. Among them, Raft is one of the well-acknowledged consensus algorithms that balance efficiency with simplicity. Indeed, Raft has become integral in the development of reliable distributed databases and systems due to the fact that it guarantees multiple nodes in a system achieve a consensus about a single state. But with the increasing demand for distributed systems that are safe and ensure privacy, the intrinsic shortcomings of Raft in protecting private information have raised several criticisms[4,5].

In general Raft implementation, all the logs and transactions are replicated on nodes for consistency. While this guarantees dependability, it does expose private information to insider threats, illegal access, and wiretapping [4,5]. Besides, there is no feature in original Raft for encrypting data during replication, nor to guarantee that nodes will be able to validate transactions without revealing private information[5].

This paper proposes an enhanced Raft consensus algorithm that will embed into its architecture advanced cryptographic techniques, such as Zero-Knowledge Proofs and Homomorphic Encryption, as solutions to these challenges in privacy [5,6]. By using ZKPs, the nodes shall be able to prove that some transaction is valid without necessarily revealing the underlying data, thus facilitating the trustless verification across the network[5]. Meanwhile, Homomorphic Encryption enables the computations on encrypted data, making sensitive information kept private while it is being processed and stored [5,6].

The general structure of the proposed privacy-preserving Raft algorithm should include the main composition which involves the structure a Raft node will assume responsibility for node state and its interaction, an encryption module for transaction encryption and decryption, and a zero-knowledge proof module in charge of the validity proof issuance and verification of transactions. The fusion of these modules will greatly improve the confidentiality during the operation of transactions while efficiency and reliability are ensured in reaching consensus.

We demonstrate the viability of our approach in a decentralized environment by giving a detailed analysis and experimental evaluation that underlines the performance implications of cryptographic enhancements on the Raft algorithm. Our results clearly show that, even though at some computational overhead, the privacy provided by the addition of HE and ZKPs makes it particularly fit for applications where data confidentiality is at stake. In other words, this work represents a milestone in the area of distributed Networks, satisfying pressing demands related to privacy-preserving mechanisms in consensus algorithms. In this regard, the integration of recent cryptography with traditional consensus approaches has so far provided unprecedented ways of

ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

strengthening the security and resilience of blockchain solutions capable of protecting sensitive information in a decentralized environment. Herein, a privacy-preserving Raft consensus methodology tries to address the concerns pertaining to privacy in the course of time when sensitive information among the set of nodes is being exchanged over a distributed system, namely blockchain. By default, Raft makes no effort to prevent data from being let out during an agreement. Indeed, sensitive information can remain private while nodes can still achieve consensus regarding a state of matters by embedding cryptography techniques of homomorphic encryption and zero-knowledge proof.

The following are this work's primary contributers:

1- Privacy-Preserving Raft Algorithm: An efficient implementation of the Raft consensus algorithm was made by implementing additional features based on ZKPs and HE in providing good privacy to the transactions but with efficiency regarding the consensus process.
2- Zero Knowledge Proof Integration: The algorithm would implement ZKPs for the nodes to confirm the validity of the transactions without necessarily divulging sensitive data; hence it can permit interactions in a distributed setting.
3- Homomorphic encryption implementation: It utilizes HE that enables it to perform operations over encrypted transactions; therefore, confidentiality about the data has been achieved over the complete consensus process. Data is being processed with no breach in the privacy of the participants.

RELATED WORKS

In this review we focuses on research that work on enhancing the performance of consensus algorithms. In order to provide network security, where An enhanced Raft Consensus method known as hhRaft was presented in [7],in order to keep an eye on the participating nodes and spot malicious activity during the leader election and log replication processes, it introduces a monitor role. The findings demonstrate that hhRaft performs better than Raft regarding consensus delay, transaction throughput, and anti-byzantine fault capabilities. Nevertheless, the study doesn't fully investigate the possible flaws or ways of assault that can still impact the hhRaft algorithm in extremely hostile settings. For the Hyperledger Fabric platform, the Raft consensus method is optimized in the article [8] with regard to leader election and log replication. By using the AdRaft method, throughput is increased by 5.8% and latency is decreased by 1.3%. By raising their term value, malevolent nodes can become the leader, compromising the fairness of the Raft cluster leader election. The assault experiment, in which a rogue node gains leader position by inflating its term value, is restored in [9] to address the aforementioned issues. By dynamically modifying the term's growth range, you may detect rogue nodes and produce a protective effect.

A quick, scalable, decentralized blockchain system called Conflux [10] processes concurrent blocks optimistically without rejecting any as forks. By using a direct acyclic graph to depict block connections, the Conflux consensus algorithm reaches an agreement on the overall order of the blocks. An independent division of the block verification, storage and update of the ledger state into three stages, with a PoW consensus algorithm as a model, is achieved through an expansion scheme called Monoxide [11]. A "crossbow mining" mechanism was designed to disperse the computing power of the fragmentation consensus group so as to effectively avoid attacks by 51% of computing power. While, in turn, impossible breakthrough FLP consensus can be performed through the fragmentation technology used. FastBFT algorithm [12]: on the basis of trusted hardware execution environment, adopt some message aggregation techniques to reduce the

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

ISSN: 2312-8135 | Print ISSN: 1992-0652 info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

communication complexity of the PBFT algorithm. HoneyBadger BFT method [13]: it has proposed an apportioned asynchronous consensus transaction group apportionment and threshold encryption technology for guaranteed operation mechanism in the case of BFT trusted operation mechanism. So, in the paper [14], it proposed an enhancement technique on OLR-Raft by improving the log replication parallelly, cache logging, and log addressing optimization. The log parallel replication is not optimized effectively, given the RPC call time is way more than compared to log landing time, and merely by log addressing, optimization of non-conflicting log backtracking efficiency of peer nodes could have been easily handled. A scheme of logging via a cache failed at effective leader node communication bottleneck issues. In [15] presents a privacy-preserving blockchain system that supports transaction auditing while ensuring data confidentiality. By combining zk-SNARKs with Homomorphic Encryption, it ensures that transaction privacy is maintained without sacrificing regulatory compliance.[16] RaBFT, an enhanced Byzantine fault tolerance consensus method based on Raft, is suggested. In order to achieve Byzantine fault tolerance, the secret sharing technique is optimized for the distribution process of log messages. The committee's role is then introduced to share the leader's communication pressure, thereby resolving the leader single node's performance bottleneck issue.[17] This study improves leader election and log replication for the Hyperledger Fabric platform using the Raft consensus method. By integrating peer nodes in the distribution of log information, the apportionment idea-based log replication enhancement seeks to lower the leader node's communication complexity during the log replication phase.

FUNDAMENTAL IDEAS

The private-protecting by protecting critical information beneath transactions or data, the Raft method seeks to enable nodes in a dispersed network to confirm that they are accurate. This can be important in fields where data secrecy is crucial, such as banking, healthcare, or identity verification.

THE RAFT ALGORITHM'S SHORTCOMINGS

To enhance a leader's authority, the Raft approach divides the entire process into two steps: leader election and log replication. During normal conditions of the leader, the Raft algorithm synchronizes logs across leader and follower nodes [18]. The election procedure will choose a new leader node in case of failure of the incumbent. Sync logs across the leader and follower nodes when the leader node is working as planned. The election process selects a replacement leader node when the current one collapses[19,20]. The dependability viewpoint of the algorithm prevents a network's consensus from splitting, and the single leader node ensures determinism for a dispersed network , With the reception of the client's request, the leading node adds the log and simultaneously broadcasts the request for log replication to other follower nodes [20].

Once the status of log application replicates on over 50% of the nodes, it sends this result back to the client. It keeps on retrying for all the log entries from a follower node that has gone down or slow in operation or dropping packets until all log entries are appropriately replicated on each node. Because of its design, which prioritizes consistency, fault tolerance, and ease of implementation, the Raft consensus algorithm lacks built-in privacy measures, while being extensively utilized for its simplicity and efficacy in reaching agreement in distributed systems. Here's why Raft does not meet privacy standards[21,22]:

- Transaction Transparency: All nodes in Raft replicate logs, which frequently include important transactions or data. This implies that all network followers have access to the same transaction data as the network leader. Unless extra privacy safeguards are put in place outside, all nodes that see sensitive data in transactions will see it in plain text.
- Lack of Built-in Encryption: Neither transactional nor log encryption is provided by Raft. Transactions are sent in clear text from the leader to the followers during log replication, leaving them open to illegal access or interception. Raft relies on the assumption of a totally trustworthy network, which is unrealistic for many real-world applications where data secrecy is essential, such financial systems or healthcare systems.
- Leader and Follower Visibility: In Raft, the leader has complete access to all incoming transactions and their contents. Despite being effective, this centralized leadership style has privacy concerns since the leader may serve as a focal point for possible exposure, particularly if it is corrupted or hostile. Complete transaction logs are also sent to followers, therefore node-to-node privacy is not maintained.
- Node State Exposure: During elections and replication procedures, the internal states of nodes are made public by Raft's architecture[23]. For instance, nodes broadcast their log status and term numbers during leader elections, which may reveal details about the transactions or processes they have handled.
- Lack of Privacy-Preserving Consensus: Raft ensures consensus among the majority of the nodes about the current state of the system. However, as opposed to ZKPs or HE, which could prove the soundness of a transaction without revealing its details, the consensus technique itself has no mechanism to anonymize or secretly validate the transactions[24].

Such limitations can be overcome by introducing cryptographic solutions, which include TLS for secure communications, SMPC for decentralized sensitive computation and limited leader exposure, Homomorphic Encryption for encrypted computation on replicated data, and Zero-Knowledge Proof for anonymous transaction validation. It is with such a combination of enhancements that Raft will be more private and secure, thus suitable for use in sensitive applications in a distributed system.

Zero-Knowledge Proofs (ZKPs):

Zero-Knowledge Proofs represent the revolution of thinking about security and privacy in the digital era. In this regard, ZKPs hold great potential in facilitating a party in proving the veracity of a statement without disclosing any personal data from the party in domains such as banking, identity management, and secure communications[25]. While the implementation is most often hard and costly, these proofs are indisposable for the future of safe digital interactions, because they manage not to undermine confidence while keeping data privacy. Zero-knowledge proofs will probably play a greater role in creating a more private and secure digital environment as encryption advances[26,27].

All these reasons make consensus techniques dependent on zero-knowledge proof for scalability, efficiency, and privacy concerns of blockchain networks. Any typical consensus protocol, such as PoW or PoS, requires nodes to share either transaction data or computational results, which may leak private information and give rise to security vulnerabilities. Instead, nodes may demonstrate the accuracy of their calculations or transactions by employing ZKPs, all without disclosing the

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

real data. By guaranteeing that sensitive material is kept private, this method lowers the amount of data sent over the network, speeds up verification, and improves security[26,27].

For instance, in PoS systems, ZK-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) can be used to verify block generation while maintaining the privacy of staking amounts. Comparably, ZK-Rollups use ZKPs to combine several transactions into a single proof, allowing for low-cost, high-throughput transactions without compromising consensus integrity. By incorporating ZKPs into consensus, blockchain technology is becoming more and more capable of supporting decentralized systems that are more private and secure. Zero-knowledge proofs operate on a few key principles.as it is explained in figure(1)[27]:

1. Completeness: the verifier will accept the statement if it is true.
2. Soundness: It should be very hard for a cheating prover to convince the verifier of the validity of a statement if that is not the case.
3. Zero-Knowledge: The verifier, apart from the fact that the statement is true, learns nothing else.

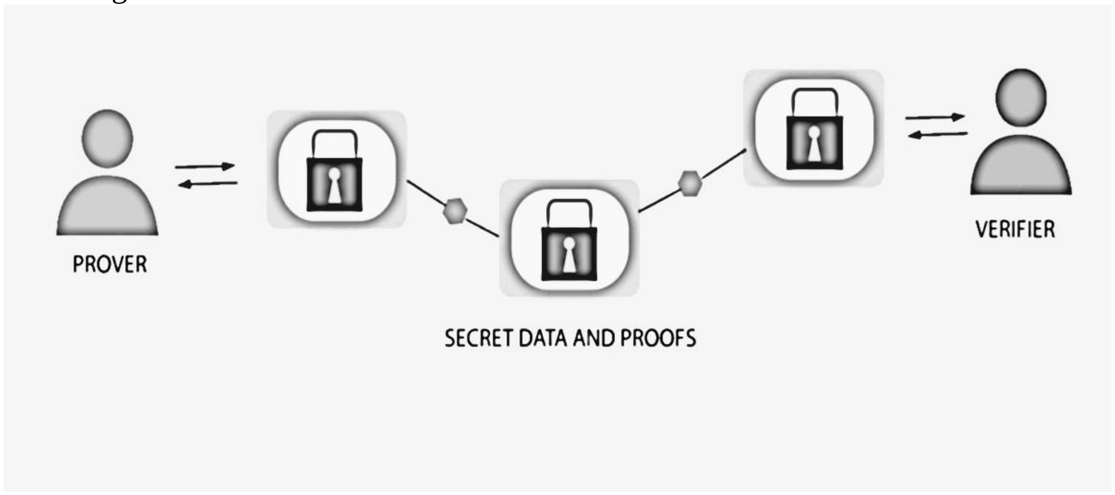


Figure (1): simplified illustration of the concept of Zero-Knowledge Proofs[27].

Homomorphic Encryption

Data privacy is crucial in the current digital era. Security issues are brought up by the massive volumes of sensitive data that organizations gather and handle. Conventional encryption techniques shield data from prying eyes during transmission and storage, but when data has to be processed, they must be decrypted, opening the door to possible security flaws. In this situation, homomorphic encryption, or HE, is useful. It is a type of encryption that makes it possible to process encrypted data without having to first decode it [28]. This implies that even during processing, data is kept private and safe. When these computations are decrypted, the result is equivalent to what would have occurred if the operations had been performed on the plaintext, or unencrypted data. In contrast to ordinary encryption, this approach allows for secure data processing. This technique is especially applicable when other parties, such as cloud service providers, need to access or manipulate sensitive data without exposing the actual data. Homomorphic encryption relies on complex mathematical functions for operations to be conducted directly on ciphertexts, the type of homomorphic encryption will determine the type of mathematical operations that can be supported, such as addition and multiplication. There are three

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

مـجلـة جـامعة بـابل للعلـوم الصـرفة والتطـبيقيـة مـجلـة جـامعة بـابل للعلـوم الصـرفة والتطـبيقيـة مـجلـة جـامعة بـابل

مـــجلة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـــجلة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـــجلة جـــامعة بـــابل

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com ISSN: 2312-8135 | Print ISSN: 1992-0652

main varieties of homomorphic encryption: fully homomorphic, somewhat homomorphic, and partially homomorphic. [28].

METHODOLOGY

The subsequent section depicts the development and test process of the privacy-preserving Raft consensus algorithm, with Homomorphic Encryption and Zero-Knowledge Proofs in place to guarantee safe, trustless verification with privacy in the distributed ledger system. Steps to be included in this approach are mentioned herein:

Core Concepts:

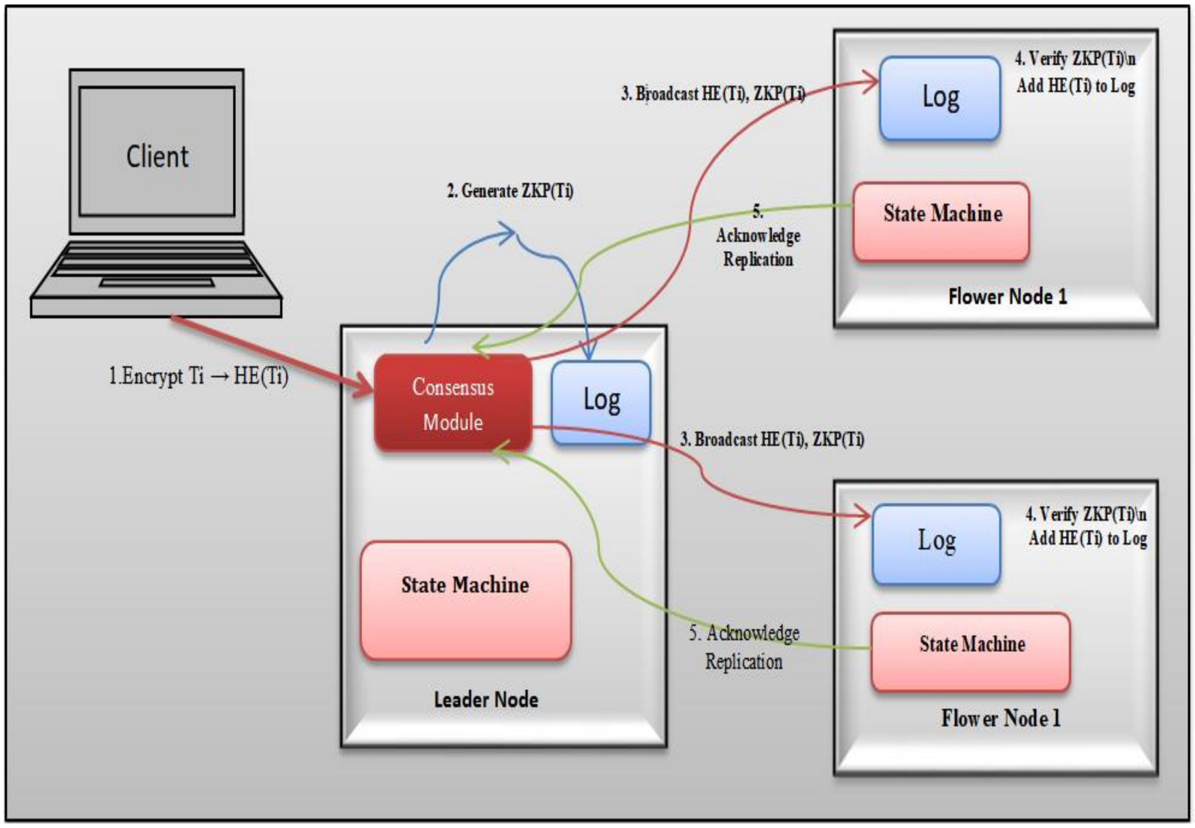
Raft is a method that enables a dispersed network to confirm the veracity of data or transactions without necessarily exposing the sensitive information that may lie underlying; hence, offering protection of privacy.

1. Zero-Knowledge Proofs (ZKPs)[26,27]:
    a. One party, the prover, is able to assure another party, the verifier, that a statement is true using a procedure called ZKPs, which discloses only the validity of the assertion.
    b. Suppose a node in a blockchain system wants to prove that it indeed verified the correctness of some transaction without necessarily revealing its details , such as balance or identities of those transacting .
2. Homomorphic Encryption (HE)[28]:
    a. HE enables doing computations on encrypted data with no need for decryption; the result of the computation is also encrypted and can afterwards be decrypted to reveal the end result.
    b. That is tosay, nodes perform some operation on the encrypted data, be it verifying some transaction or reaching consensus, without allowing sensitive information to get exposed.
3. Hybrid Approach Combining ZKP and HE: The Raft consensus could benefit from a hybrid approach where:
    a. ZKPs are used to prove that a certain operation, as transaction validation was performed correctly.
    b. HE enables the operations on ciphertext across nodes to keep the private information hidden throughout the consensus process.The leader node can send encrypted data to the followers (using HE), and instead of verifying the raw data, followers use a zero-knowledge proof to ensure that the computation or transaction was performed correctly. This provides both privacy and correctness guarantees.

As in a medical blockchain, patient data (encrypted using HE) is processed across nodes to compute a diagnosis. The network can prove the computation's correctness (using ZKPs) without revealing patient records.

Architecture of Privacy-Preserving Raft Consensus Algorithm(PPRCA) :

The privacy-preserving Raft algorithm is built on the traditional Raft consensus protocol, which ensures fault tolerance and consistency across distributed nodes the Figure(2) explain the diagram of the proposed PPRCA, while Figure(3) is explained the system phases . In this enhanced version, we integrate cryptographic techniques to address privacy concerns. The system architecture is explained with the following algorithm(1).

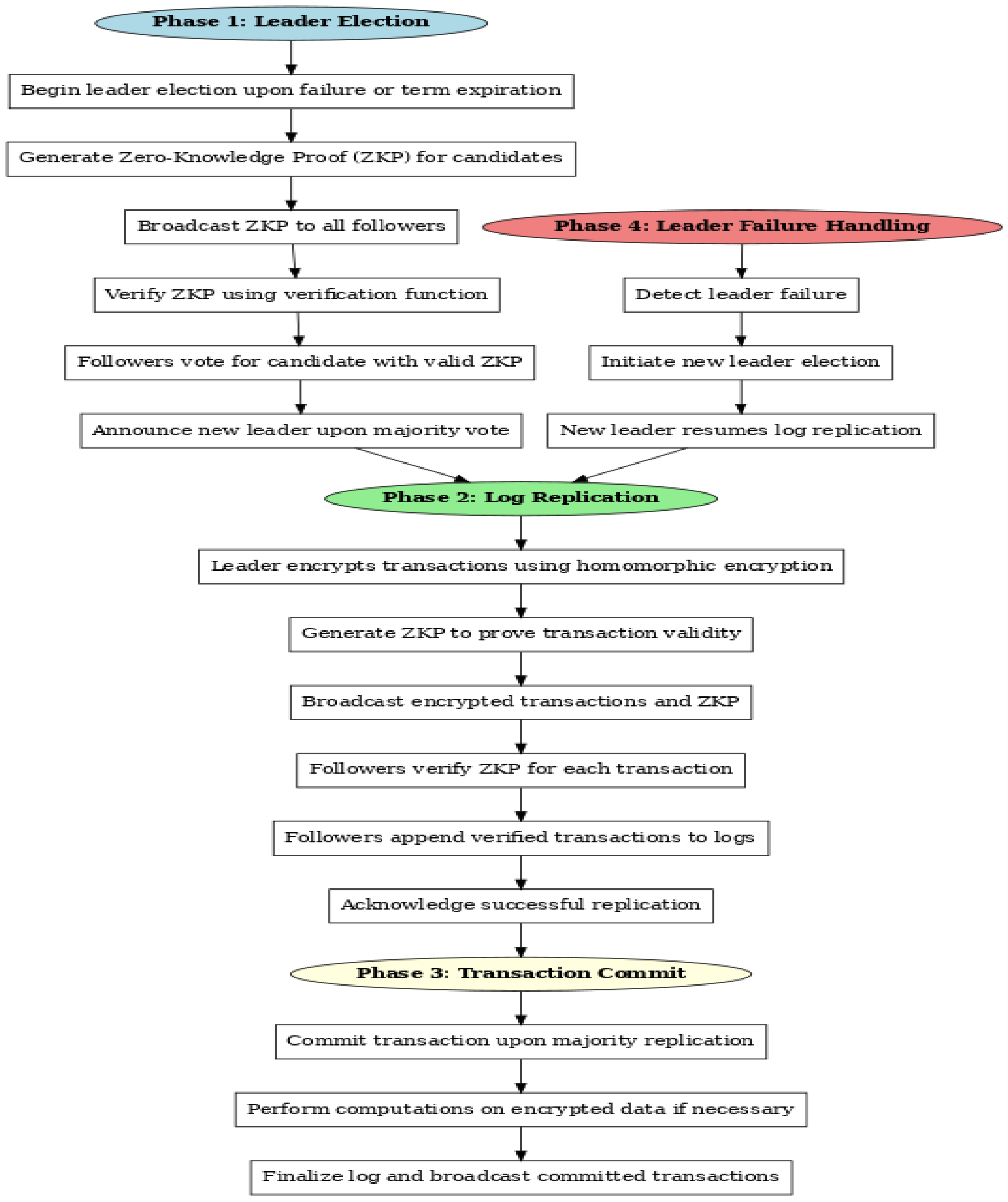


Figure(2): Diagram of Secure Transaction Replication with Privacy-Preserving Raft Consensus Algorithm(PPRCA) in the current study

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com


Phase 1: Leader Election
Begin leader election upon failure or term expiration
Generate Zero-Knowledge Proof (ZKP) for candidates
Broadcast ZKP to all followers
Verify ZKP using verification function
Followers vote for candidate with valid ZKP
Announce new leader upon majority vote
Phase 4: Leader Failure Handling
Detect leader failure
Initiate new leader election
New leader resumes log replication
Phase 2: Log Replication
Leader encrypts transactions using homomorphic encryption
Generate ZKP to prove transaction validity
Broadcast encrypted transactions and ZKP
Followers verify ZKP for each transaction
Followers append verified transactions to logs
Acknowledge successful replication
Phase 3: Transaction Commit
Commit transaction upon majority replication
Perform computations on encrypted data if necessary
Finalize log and broadcast committed transactions

Figure(3): Diagram of the proposed system phases in the current study.

مـجلـة جـــامعة بــــابل للعلـــــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بــــابل للعلـــــوم الصـــرفة والتطـــبيقية مـجلـة جـــامعة بــــابل

ISSN: 2312-8135 | Print ISSN: 1992-0652 info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

```
Algorithm(1): Privacy-Preserving Raft Consensus
Input:
  • Set of transactions $T_1,T_2,...,T_n$ (encrypted with homomorphic encryption)
  • Leader node L (current leader of the Raft cluster)
  • Follower nodes $F_1,F_2,...,F_k$
  • Zero-Knowledge Proof generation function ZKP(T)
  • Homomorphic Encryption function HE(data)
  • Verification function VerifyZKP(ZKP)
  •
Output:
  • Consensus on encrypted transactions without revealing sensitive data.
  • Privacy-preserving log replication and leader election.

Phase 1: Leader Election
  1. Begin
  2. if leader node L fails or the term ends then
           o For each candidate nodes C :
         generates a Zero-Knowledge Proof ZKP(C)  //prove its eligibility without revealing sensitive information.
  3. Broadcast proof:
           o Candidate nodes broadcast ZKP(C)to all followers.
  4. Verify election ZKP:
           o For follower nodes Fi verifies ZKP(C) using the VerifyZKP function.
  5. Vote:
       o Followers cast votes for the candidate with the correct ZKP.
       o If a majority of votes is received, the candidate becomes the leader L.
  6. End election:
       o If a new leader is elected, broadcast the result.

Phase 2: Log Replication
  1. Encrypt transactions:
       o For each  transaction ($T_i$)
           o Homomorphic Encryption HE($T_i$) before replication.
           o Replication  HE($T_i$).
  2. For each transaction($T_i$):
       o generates a Zero-Knowledge Proof ZKP(Ti)   // proves the transaction's validity
  3. Broadcast encrypted transactions and ZKPs:
       o Leader node Lsends HE($T_i$) and ZKP($T_i$)to all follower nodes $F_1,F_2,...,F_k$.
  4. Follower verification:
       o Each follower node Fi verifies ZKP($T_i$) using VerifyZKP(ZKP($T_i$)).
       o If valid, the follower adds HE($T_i$) to its log.
  5. Acknowledge replication:
       o Follower nodes acknowledge the successful replication back to the leader node L.
```

Phase 3: Transaction Commit

1. Reach consensus:
   - Once a majority of followers replicate \`HE($T_i$)\`, the leader commits the transaction..
2. Perform computation :
   - Homomorphic computations on HE(Ti).
   - The result remains encrypted as HE(Result). //**decrypt able only by an authorized entity with the decryption key**
   - The leader finalizes the log and broadcasts the committed transactions to all follower nodes.

Phase 4: Leader Failure Handling

1. Detect failure:
   - If the leader node L fails, follower nodes initiate a new leader election (go to Phase 1).
2. Re-elect new leader:
   - The new leader continues the replication and transaction validation process from the last committed log, maintaining privacy throughout the transition.

Functions Explained:

- ZKP(T_i):
  - A Zero-Knowledge Proof that proves the correctness of transaction T_i without revealing its details.
- HE(T_i):
  - Homomorphic Encryption of transaction T_i, ensuring that computations can be performed on encrypted data.
- VerifyZKP(ZKP(T_i)):
  - Verifies the correctness of ZKP(T_i). If valid, it ensures that T_i is correct without knowing its content.

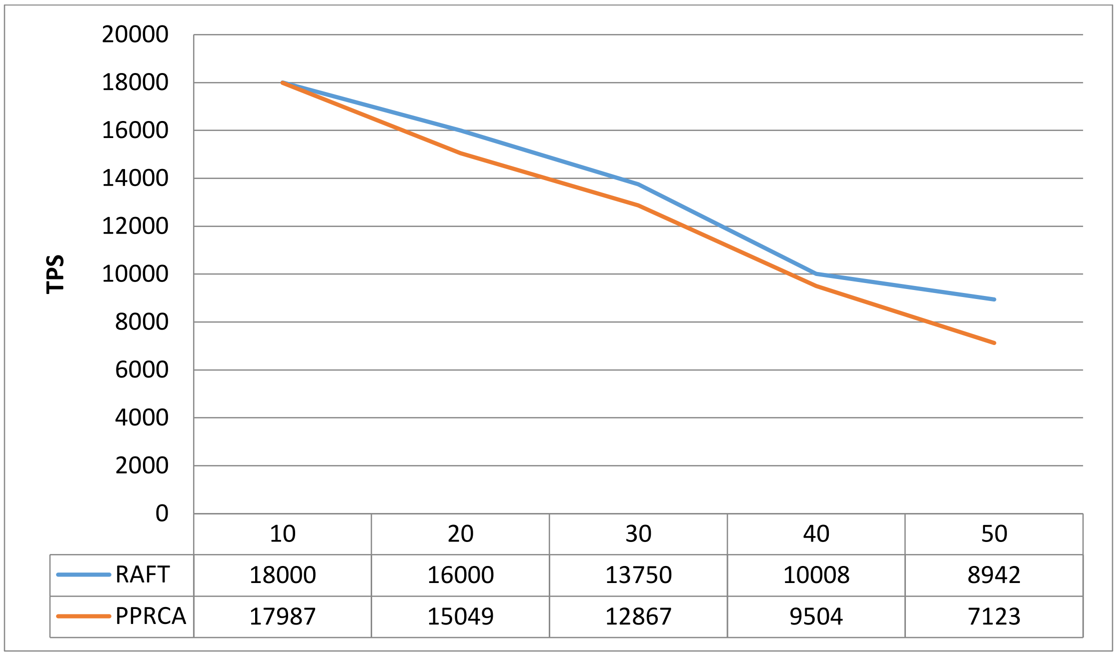
Evaluation of the Privacy-Preserving Raft Consensus Algorithm(PPRCA)

We conducted performance testing and functional verification of the PPRCA algorithm. To confirm the efficacy of the PPRCA algorithm, these software are run on a HP DESK-TOP-F5259A5, with a Processor Intel (R) Core (TM) i7-10750H CPU @2.60GHz, RAM 16.0GB. we created a series of servers for the functional verification use a Python 3.8.10. There are 50 servers in the cluster, and they are all referred to as nodes 1, 2,..., 50. The performance of the Raft and PPRCA was evaluated in the performance test. In the normal condition, Figure 5 displays the two methods' performance with varying numbers of nodes. It can be concluded from the experimental results that there is no extra overhead, the deterioration in performance did not happen, and PPRCA could work as well as the original Raft method when conducting a TPS performance test in the scale of 10 to 50 devices. While the traditional RAFT algorithm consistently outperforms PPRCA in terms of throughput (TPS) across various number of nodes as shown in Figure(4), the enhanced security and data protection mechanisms offered by PPRCA may justify the marginal performance disparity. These features position PPRCA as a more suitable option for applications where data security and integrity are of paramount importance, such as financial systems, governmental platforms, or other sensitive domains. Although RAFT remains the preferred choice for scenarios prioritizing optimal performance, the advanced security capabilities of PPRCA provide a

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

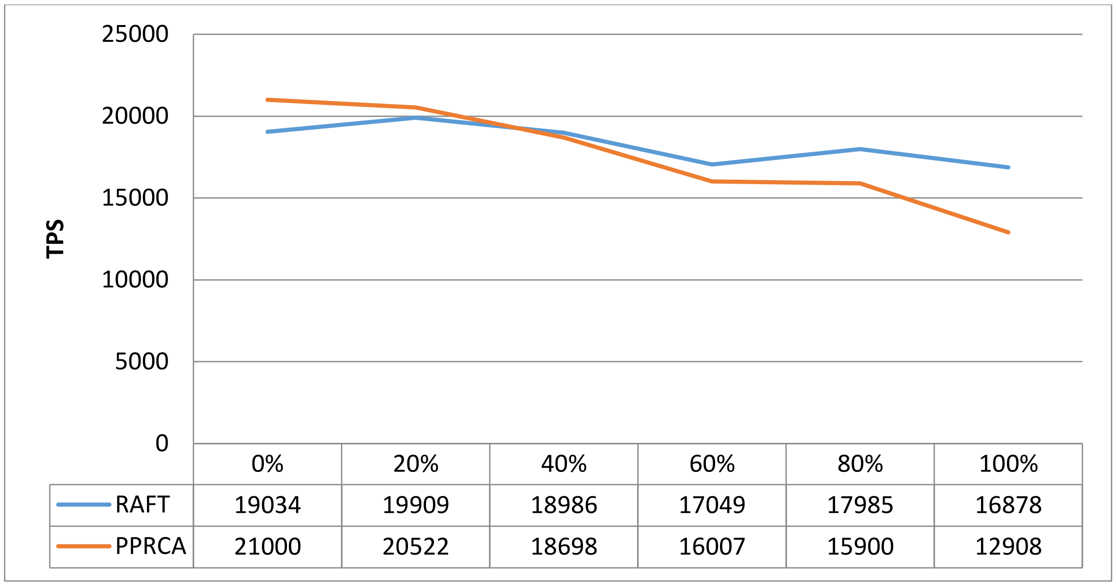
compelling argument for its adoption in security-critical environments, where the trade-off between performance and protection is more acceptable.



	10	20	30	40	50
RAFT	18000	16000	13750	10008	8942
PPRCA	17987	15049	12867	9504	7123

Figure (4) Average throughput vs number of nodes.

We analyze the effectiveness of the Raft and PPRCA algorithms in the abnormal state and model the Leader node's performance deterioration at a scale of 50 nodes. Figure 5: Results of the experiment. Within the network quality test range, the node network bandwidth is from 50 to 550 kbps, latency is from 0 to 100 ms, packet loss rate is from 0 to 12%, and Raft's TPS falls. The performance degrades rapidly the networks quality reaches 50% when the packet loss rate surpasses 5%. On the other hand, PPRCA maintained near or slightly worse performance. Thus, the average writen throughput of PPRCA is about 5% lower than that of traditional Raft.



	0%	20%	40%	60%	80%	100%
RAFT	19034	19909	18986	17049	17985	16878
PPRCA	21000	20522	18698	16007	15900	12908

Figure (5): Raft and PPRCA algorithms on network quality and TPS comparison.

Strong privacy assurances are introduced via the Privacy-Preserving Raft method, albeit at the expense of increased CPU utilization, as it show in table 1. Traditional Raft performs well in contexts where performance is crucial, while Privacy-Preserving Raft is more appropriate for applications that are sensitive to privacy. This trade-off highlights the importance of carefully considering the application's requirements. Performance bottlenecks could be reduced by optimizations such as batch processing of ZKPs or using less complex cryptography techniques.

Table-1: Traditional Raft vs. Privacy-Preserving Raft Consensus Algorithm(PPRCA) s on CPU Usage

Metric	Traditional Raft	Privacy-Preserving Raft
CPU Usage (%)	10-15%	40-60%

Security Discussion: Traditional Raft vs. Privacy-Preserving Raft Consensus Algorithm(PPRCA) :

Traditional Raft provides consistency and fault tolerance; however, it lacks inherent mechanisms that guarantee privacy and confidentiality. Each node replicates the transactions and logs in plaintext; therefore, it exposes sensitive information to all participants and relies on a trusted network environment. This makes traditional Raft vulnerable to eavesdropping, data manipulation, and insider attacks. It also does not provide protection against unauthorized access and hence cannot be used in apps that need ptivate data such as financial transactions or personal records.

In contrast, the Raft consensus algorithm with privacy enhancement offers improved security and privacy by embedding encryption and Zero-Knowledge Proofs. Data encryption ensures that data remains confidential even during replication and consensus, thus preventing data exposure to unauthorized parties even in untrusted environments. Zero-Knowledge Proofs enable followers to

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

مـــجلة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـــجلة جـــامعة بـــابل للعلـــوم الصـــرفة والتطـــبيقية مـــجلة جـــامعة بـــابل

validate transactions without necessarily accessing the contents, hence protecting sensitive information while at the same time maintaining the consensus process's integrity. These mechanisms contribute to defending against other attacks, such as eavesdropping and tampering, because the encrypted data and the ZKPs ensure that only valid transactions will be accepted and replicated. In general, the privacy-preserving Raft offers strong security and privacy guarantees, making it suitable for sensitive and regulated applications as explained in Table 2.

Table-2: Traditional Raft vs. Privacy-Preserving Raft Consensus Algorithm(PPRCA)

Aspect	Traditional Raft	PPRCA
Confidentiality	No inherent confidentiality; data is in plaintext.	High confidentiality through encryption and ZKPs.
Integrity	Relies on majority agreement; susceptible to tampering.	Ensures cryptographic integrity with ZKP-based validation.
Resilience to Attacks	Vulnerable to eavesdropping and data manipulation.	Strong resilience with encrypted logs and ZKPs.
Privacy	No privacy mechanisms; all nodes see transaction details.	Robust privacy; no plaintext data is shared or stored.

CONCLUSIONS

In distributed networks, the privacy-preserving Raft consensus method offers a solid basis for protecting private information while sustaining consensus. It integrates advanced cryptographic techniques, including ZKPs and HE, which will guarantee the anonymity of all information even in decentralized settings. However performance overhead issues are yet to be resolved with further optimizations and enhancements. This has great potential for application in areas that are very sensitive to data security and privacy.

Conflict of interests.

Non conflict of interest

References

[1] M. Zhang, “Implementation of Improved Raft Consensus Algorithm in IoT Information Security Management,” *Int. J. Adv. Comput. Sci. Appl. (IJACSA)*, vol. 15, no. 6, 2024.

[2] Y. Li, Y. Fan, L. Zhang, and J. Crowcroft, "RAFT Consensus Reliability in Wireless Networks: Probabilistic Analysis," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12839–12853, Jul. 2023.

[3] X. Xu, L. Hou, Y. Li, and Y. Geng, "Weighted RAFT: An Improved Blockchain Consensus Mechanism for Internet of Things Application," in *Proc. 2021 7th Int. Conf. Comput. Commun. (ICCC)* , 2021.

[4] D. Huang, X. Ma, and S. Zhang, "Performance Analysis of the Raft Consensus Algorithm for Private Blockchains," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 1, pp. 172–181, Jan. 2020.

[5] S. R., R. Chirakarotu Nair, and P. K. Panakalapati, “Promise of Zero-Knowledge Proofs (ZKPs) for Blockchain Privacy and Security: Opportunities, Challenges, and Future Directions,” *Security Privacy*, Sep. 2024.

[6] D. Cao, B. Li, and J. Cai, "A Privacy-Preserving Method for Cross-Chain Interoperability Using Homomorphic Encryption," *Peer-to-Peer Netw. Appl.*, vol. 18, pp. 1–16, 2025.

[7] Y. Wang, S. Li, L. Xu, and L. Xu, "Improved Raft Consensus Algorithm in High Real-Time and Highly Adversarial Environment," in *Proc. 18th Int. Conf. Web Inf. Syst. Appl. (WISA)*, Kaifeng, China, 2021.

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com | ISSN: 2312-8135 | Print ISSN: 1992-0652

[8] X. Ding and L. Rajamanickam, “Improvement of Raft Consensus Algorithm Based on Credit Values on Electronic Medical Record (EMR) Blockchain,” in *Soc. Photo-Opt. Instrum. Eng. (SPIE) Conf. Ser.*, 2023.

[9] Y. Wang, M. Tian, Y. Zhang, X. Liu, Y. Xiao, and X. Hei, "A Security Enhancement Scheme for Raft Consensus Algorithm Against Term Forgery Attacks," in *Proc. 2022 IEEE 21st Int. Conf. Ubiquitous Comput. Commun. (IUCC/CIT/DSCI/SmartCNS)*, Chongqing, China, 2022.

[10] C. Li, P. Li, D. Zhou, W. Xu, F. Long, and A. Yao, "Scaling Nakamoto Consensus to Thousands of Transactions Per Second," *Comput. Sci.*, 2018.

[11] J. Wang and H. Wang, "Monoxide: Scale Out Blockchains with Asynchronous Consensus Zones," in *Proc. 16th USENIX Symp. Netw. Syst. Des. Implementation (NSDI)*, 2019.

[12] J. Liu, W. Li, G. O. Karame and N. Asokan, "Scalable Byzantine Consensus via Hardware-Assisted Secret Sharing," in *IEEE Transactions on Computers*, vol. 68, no. 1, pp. 139-151, 2019.

[13] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The Honey Badger of BFT Protocols," in *Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Security (CCS '16)*, Association for Computing Machinery, New York, NY, USA, 2016.

[14] K. Dai, "Design of Optimized Log Replication Distributed Consensus Algorithm OLR-RAFT Based on Raft," Master’s thesis, Huazhong Univ. Sci. Technol., 2017.

[15] L. Xu, Y. Zhang, and L. Zhu, "Regulation-Friendly Privacy-Preserving Blockchain Based on zk-SNARK," in *Proc. Adv. Inf. Syst. Eng. Workshops*, 2023.

[16] F. Bai, F. Li, T. Shen, K. Zeng, X. Zhang, and C. Zhang, "RaBFT: An Improved Byzantine Fault Tolerance Consensus Algorithm Based on Raft," *J. Supercomput.*, vol. 80, pp. 21533–21560, 2024.

[17] W. Fu, X. Wei, and S. Tong, "An Improved Blockchain Consensus Algorithm Based on Raft," *Arab J. Sci. Eng.*, vol. 46, pp. 8137–8149, 2021.

[18] R. Wang, L. Zhang, Q. Xu, and H. Zhou, "K-Bucket Based Raft-Like Consensus Algorithm for Permissioned Blockchain," in *Proc. 2019 IEEE 25th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, 2019.

[19] S. Zhang and J. H. Lee, "A Group Signature and Authentication Scheme for Blockchain-Based Mobile-Edge Computing," *IEEE Internet Things J.*, vol. 7, pp. 4557–4565, 2019.

[20] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A Survey on Privacy Protection in Blockchain Systems," *J. Netw. Comput. Appl.*, 2019.

[21] Y. Wang, S. Li, L. Xu, and L. Xu, "Improved Raft Consensus Algorithm in High Real-Time and Highly Adversarial Environment," in *Springer*, 2021.

[22] C. Copeland and H. Zhong, "Tangaroa: A Byzantine Fault Tolerant Raft," 2016.

[23] Z. Wang, T. Li, H. Wang, A. Shao, Y. Bai, S. Cai, Z. Xu, and D. Wang, "CRaft: An Erasure-Coding-Supported Version of Raft for Reducing Storage Cost and Network Cost," in *Proc. 18th USENIX Conf. File Storage Technol. (FAST '20)*, 2020.

[24] S. Tian, Y. Liu, Y. Zhang, and Y. Zhao, "A Byzantine Fault-Tolerant Raft Algorithm Combined with Schnorr Signature," in *Proc. IEEE*, 2021.

[25] S. Anthoniraj, R. Mishra, S. Loonkar, T. Agarwal, G. Ahluwalia, and A. Gill, "Design of Novel Cryptographic Model Using Zero-Knowledge Proof Structure for Cyber Security Applications," *J. Cybersecurity Inf. Manage.*, vol. 14, no. 1, 2024.

[26] X. Tang, L. Shi, X. Wang, K. Charbonnet, S. Tang, and S. Sun, "Zero-Knowledge Proof Vulnerability Analysis and Security Auditing," *Cryptology ePrint Archive*, pp. 514,2024.

[27] K. Munjal and R. Bhatia, "A Systematic Review of Homomorphic Encryption and Its Contributions in Healthcare Industry," *Complex Intell. Syst.*, vol. 9, no. 4, pp. 3759–3786, 2023.

[28] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A Survey on Homomorphic Encryption Schemes: Theory and Implementation," *ACM Comput. Surv. (CSUR)*, vol. 51, no. 4, pp. 1–35, 2018.

ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

Vol.33; No.1.| 2025

الخلاصة

المقدمة: تقدم هذه الورقة تحسينًا على طريقة التوافق المعروفة باسم "Raft" لمعالجة مخاوف الخصوصية من خلال دمج تقنيات الإثبات الصفري (Zero-Knowledge Proofs – ZKPs) والتشفير المتماثل (Homomorphic Encryption). يهدف هذا التحسين إلى ضمان سرية البيانات مع الحفاظ على التوافق في الشبكات الموزعة. ورغم أن خوارزمية "Raft" التقليدية تتميز في عمليات انتخاب القائد وتكرار السجل، إلا أنها تفتقر إلى آليات مدمجة لحماية الخصوصية، مما يجعلها غير مناسبة للتطبيقات الحساسة مثل إدارة سلاسل التوريد والأنظمة المالية وأنظمة الرعاية الصحية.

طرق العمل: تم اقتراح ال PPRCA لحل مشكلات الخصوصية التي تحدث عند نقل المعلومات الخاصة بين العقد في نظام موزع مثل blockchain. لا توفر خوارزميه ال Raft التقليديه ا خطوات لحماية سرية البيانات عند التوصل إلى إجماع. يمكن أن يساعد استخدام طرق التشفير التي تحافظ على الخصوصية، مثل التشفير المتجانس (HE) وإثباتات المعرفة الصفرية (ZKPs)، العقد في إنشاء اتفاق مع حماية البيانات الحساسة.

الاستنتاجات: أظهرت تحليلات الأداء والأمان أن الطريقة المقترحة فعالة وقابلة للتطبيق في الأنظمة الموزعة الحساسة للخصوصية، رغم وجود بعض التكاليف الحسابية. تمهد هذه الورقة الطريق نحو طرق توافق موزعة آمنة وقابلة للتوسع وتدعم حماية الخصوصية.

الكلمات المفتاحية: خوارزمية التوافق Raft، حفظ الخصوصية، الإثباتات الصفرية (ZKPs)، التشفير المتماثل (HE)، الشبكات الموزعة.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com