International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A LAYERED BLOCKCHAIN-BASED SECURE MESSAGING ARCHITECTURE WITH SMART CONTRACT-DRIVEN CERTIFICATE MANAGEMENT SYSTEM

Rohaila Naaz[1*] ,Dr.Ashendra kumar Saxena[2]

[1]Research scholar Teerthanker Mahaveer University, Moradabad
[2]Professor Teerthanker Mahaveer University, Moradabad

I Introduction

In the contemporary digital landscape, the demand for secure, private, and tamper-resistant communication has never been more critical. Conventional messaging platforms, which predominantly rely on centralized servers, are increasingly vulnerable to data breaches, unauthorized surveillance, and censorship. Even with the adoption of end-to-end encryption, these systems remain susceptible to single points of failure and metadata exposure, undermining user privacy and trust. Blockchain technology has emerged as a transformative solution to these challenges, offering a decentralized, immutable, and transparent infrastructure for secure data exchange. By leveraging distributed ledger technology, Blockchain-based messaging systems eliminate the need for trusted intermediaries, enhance resistance to censorship, and ensure data integrity through consensus-driven validation.

Recent advancements have further enriched the security landscape by integrating smart contracts for automated certificate management and user authentication. Smart contracts, as self-executing code deployed on the Blockchain , facilitate trustless key exchanges, enforce access control, and automate the issuance and revocation of user certificates without relying on third-party authorities. At the cryptographic layer, the employment of Elliptic Curve Diffie-Hellman (ECDH) key exchange and advanced symmetric encryption algorithms such as AES ensures that only authorized parties can decrypt and access message content, while digital signatures and authentication tags provide robust guarantees of data integrity and non-repudiation (Dwivedi, A.et al.)

Despite these innovations, the efficiency and scalability of decentralized messaging platforms are closely tied to the underlying Blockchain consensus mechanism. Traditional consensus algorithms, such as Proof of Work and Proof of Stake, often incur significant computational overhead or centralization risks. To address these limitations, this work introduces a Random Proof of Authority (PoA) consensus protocol, implemented in Solidity, which dynamically selects validators to finalize message transactions(Singh, M. M. Islam 2025). This approach balances security, performance, and decentralization, making it well-suited for real-time secure communication applications.

This paper presents a comprehensive, layered architecture for Blockchain-based secure messaging, encompassing a React-based frontend, a Node.js backend, and an Ethereum-compatible Blockchain network. The system combines ECDH-based key management, smart contract-driven certificate issuance, and a novel Random PoA consensus mechanism to deliver end-to-end encrypted, tamper-proof, and censorship-resistant messaging. Through this design, the proposed framework addresses the pressing need for trustworthy digital communication in an era of escalating cyber threats and privacy concerns.

II Literature Review

Blockchain and smart contracts are revolutionizing secure messaging by building decentralized platforms that prioritize transparency and user control. Unlike old-school centralized apps, which leave users exposed to hacks, censorship, and data leaks due to single weak points, this approach spreads the risk across a tamper-proof network Antonopoulos, A. M., & Wood, G. (2019).. Drawing on blockchain's core strengths—like unchangeable records, open verification, and strong encryption—it creates a reliable backbone for private chats (Carranza et al 2024). Smart contracts add automation

Received: August 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

for things like certificate handling, paired with a random proof-of-authority setup for smoother, safer operations (weng et al. 2022)

The system stacks into clear layers for easy scaling, much like proven designs in the field (Rebello et al 2022). At the top, a simple user interface lets people craft, lock down, send, and unlock messages effortlessly.

A key innovation here is the custom consensus algorithm, shaped by deep dives into the literature break down how these algorithms safeguard IoT data integrity and anonymity(showkar et al. 2023), but warn of PoW's energy drain and PoS's centralization risks—calling for balanced options that fit messaging's speed and privacy needs. (Sandberg, 2023) prove PoS beats PoW for health record sharing, slashing power use and delays with ECDSA signatures and SHA-256—ideal for urgent, secure exchanges. (Abd Rabou, A. (2025) tackle vehicle networks with VBCA, a team-based blockchain that cuts latency and boosts confirmations over PoW. (H.Mao et al.2023) push DAG for 6G's massive data flows, enabling parallel checks to handle high-speed messaging without slowdowns.

Kang et al. (2022) tweak PBFT into ITPBFT for traceability, trimming consensus time amid faulty nodes. Menon et al. (2023) blend blockchain with ML for smart homes, leaning on distributed agreement for rock-solid privacy. Wang et al. (2019) speed up PBFT for energy grids, while Liu et al. (2023) introduce CBFT—grouping fast nodes by credit to skyrocket throughput (3x PBFT) and tolerance (59% better). Ahanger et al. (2022) add reputation scoring for drones, Oliveira et al. (2023) eye Industry 5.0's needs, and Baig et al. (2022) refine Raft for IoT supply chains.

In short, no one-size-fits-all: PoW secures but slows; PoS saves energy but needs safeguards; PBFT shines in trusted groups yet scales poorly. Tailored hybrids—like VBCA or CBFT—nail the security-efficiency tradeoff for messaging, though real-world tests are next (mostly sims so far). Table 1 sums up architectures, consensuses, and comms protocols for blockchain chat systems.

Category	**Key Concepts and Findings**
Blockchain Messaging System Architecture	Layered architecture with modularity and scalability; User Interface Layer for message encryption and decryption; Smart contracts for certificate management and PoA consensus
Consensus Protocols	Proof of Work (PoW); Proof of Stake (PoS) and variants; Practical Byzantine Fault Tolerance (PBFT) and derivatives; Directed Acyclic Graph (DAG)-based asynchronous consensus; Reputation-based consensus; Modified Raft (mRAFT)
Consensus Protocol Trade-offs	PoW: High security but high latency and resource consumption; PoS: Energy efficient but vulnerable to stake centralization; PBFT: Robust but limited scalability due to communication overhead; Trade-offs between throughput, latency, fault tolerance, and privacy
Application-Specific Consensus Algorithms	Vehicular network Based Consensus Algorithm (VBCA) for low latency; Blockchain Network Collaboration Mechanism (BNCM) for energy systems; Credit-Based Byzantine Fault Tolerant (CBFT) for scalability; Reputation-

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	based consensus for UAV networks; Modified Raft (mRAFT) for supply chain IoT
Emerging Communication Paradigms	Integration with 6G networks requiring low latency and high throughput; Asynchronous DAG consensus for parallel transaction validation; Industry 5.0 decentralized access control and device-to-device communication
Consensus Protocol Enhancements	Grouping and credit grading to improve PBFT scalability (CBFT); Reputation systems to balance security and efficiency; Workload redistribution in mRAFT to reduce leader bottleneck
Key Findings and Conclusions	Consensus protocols must be tailored to application context; No single consensus optimizes all parameters,Hybrid and asynchronous models show promise; Need for real-world deployments and standardized benchmarks

Table-1 Concept Table of Blockchain Messaging System

In summary it is shown form literature review that any consensus is not able to give optimal performance so here we applied an asynchronous model, RPoA is applied here for the selection of validators node with the help of permissioned Blockchain Ethereum and PoW Consensus is applied for adding the encrypted information in the form of transactions in the Ethereum Blockchain.

III Methodology

This section outlines the design and implementation methodology of the proposed secure messaging architecture, which integrates a permissionless blockchain, a smart contract-driven certificate management system, and a novel Random Proof of Authority (RPoA) consensus mechanism to ensure secure, decentralized communication.

Our setup breaks down into straightforward layers that make secure, blockchain-powered messaging feel simple and reliable shown in Fig-1.

User Layer- Where the Magic Starts

This is all about you—the person chatting on your phone or laptop. You fire up the app with a clean, everyday interface built using HTML and CSS up front, backed by Python for smooth operation. Sending a message? It gets locked tight on your device with the recipient's public key—think Signal Protocol or AES-256 strength—before heading to a Node.js or Python backend. That backend bridges to Ethereum, using Web3.js or Web3.py to tuck the encrypted message into a smart contract. No one but the recipient can unlock it, and it's stored forever on the blockchain, safe from tampering.

Locally, the app saves messages encrypted in your browser or a light database like SQLite. Authentication keeps things private, and decryption happens right there on your end. Fig. 1 sketches the full flow—user-friendly on top, bulletproof underneath.

Backend- The Behind-the-Scenes Hero

Running on Node.js or Python, this is the workhorse handling logins (via JWT tokens), message handoffs, and blockchain chats. When you hit send, it calls the smart contract to log your encrypted

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

note immutably. Pulling messages? It grabs them from the chain, passes to the front end for your eyes only. This split lets the user side shine on experience while the backend nails the heavy lifting.

Network Layer- Connecting the Dots

Here, peer devices (you, your friend, relays), servers, and blockchain validators team up for routing, key swaps, and agreement on what's real. We run it local and modular: a Hardhat blockchain node on localhost:8545 mimics Ethereum for testing—perfect for deploying contracts and validating without internet drama.

The Node.js backend hums on port 3001, grabbing front-end requests and piping them to the blockchain. Meanwhile, a React front end on port 3000 delivers the chat UI, hiding the techy bits. Want to test across devices? Just swap localhost for your machine's IP.

This peer-to-peer vibe echoes the OSI model: messages zip securely, keys exchange safely, and consensus locks in truth across nodes. It's flexible—swap blockchains later if needed—keeping things scalable and future-proof.

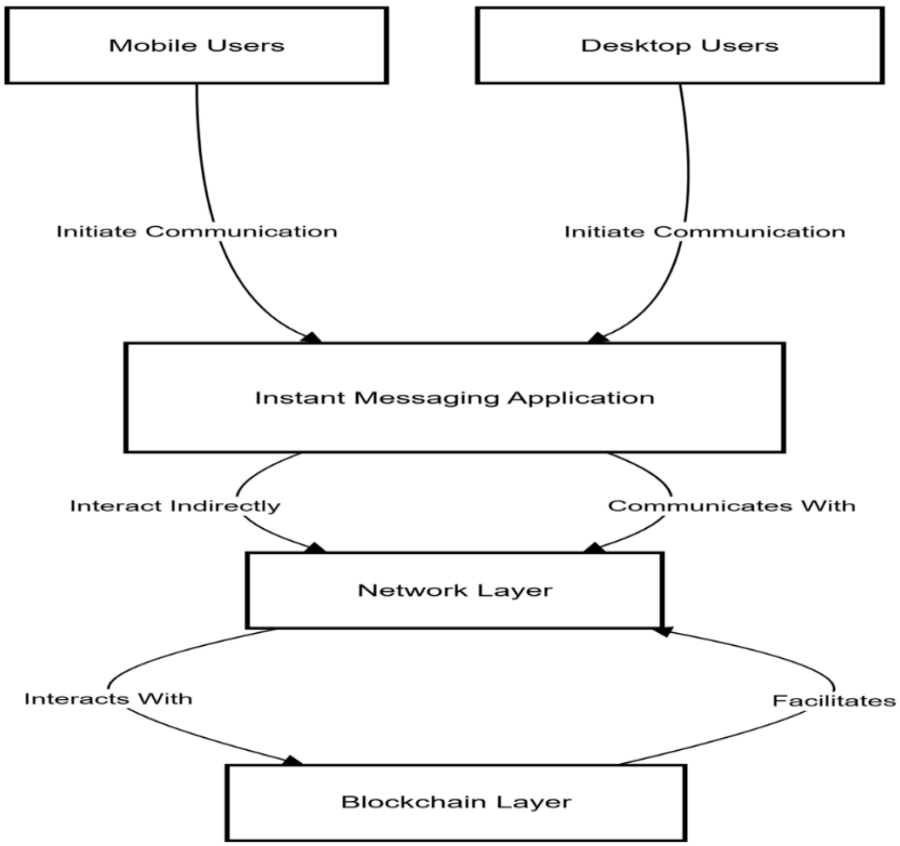

Fig-1 System Architecture

Figure 1 depicts the layered architecture of the proposed secure instant messaging system. Mobile and desktop users initiate communication through a common instant messaging application. This application handles user interactions and forwards communication requests to the network layer. The network layer manages message transmission and acts as an interface between the application and the blockchain. The blockchain layer supports the system by providing decentralized security services such as verification and trusted record management. This layered interaction ensures secure and structured communication between users.

In Fig-2 the message flow communication between Peer nodes, Intermediary Relay nodes, Server nodes and Validators nodes is shown.

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

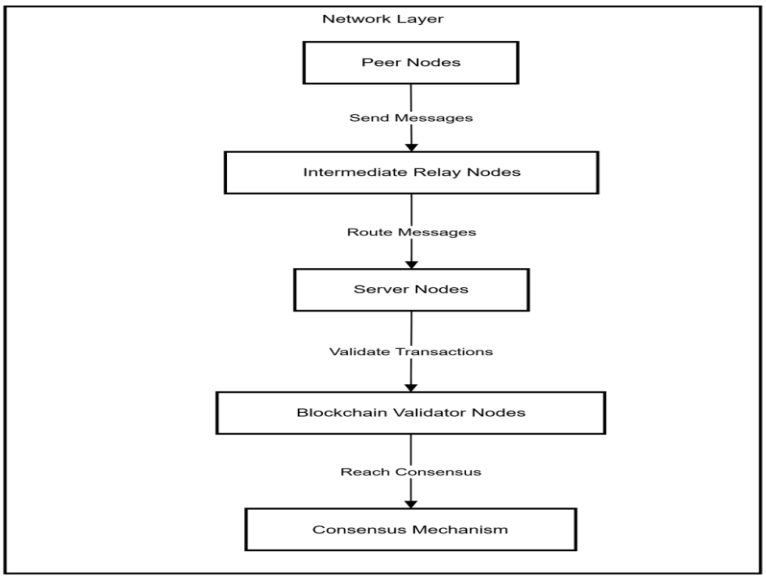

Fig-2 Network Layer consisting of Peer nodes, Server nodes and Validator Nodes

Figure 2 illustrates the internal working of the network layer in the proposed system. Peer nodes generate and send messages, which are first handled by intermediate relay nodes responsible for routing the messages efficiently. These messages are then forwarded to server nodes, where transaction-related checks are performed. Validated transactions are passed to blockchain validator nodes, which participate in the consensus mechanism to confirm and record the transactions. This flow ensures reliable message delivery and secure validation through decentralized consensus.

In summary, the network layer in a multi-server blockchain messaging architecture orchestrates the interaction between frontend, backend, and blockchain nodes, leveraging dedicated communication protocols and cryptographic primitives to ensure secure, decentralized, and efficient message exchange. This separation of concerns not only enhances system robustness and scalability but also aligns with best practices in both blockchain and distributed systems engineering.

At the heart of our trusted blockchain nodes runs RPoA—a smart twist on Proof of Authority. Pre-vetted validators take turns creating blocks, picked at random for fairness and to dodge any sneaky team-ups. It's fast and low-delay like regular PoA, but the surprise element keeps things honest. Every node spins up an Ethereum Virtual Machine (EVM) to run smart contracts smoothly, keeping everyone's data in sync and rules airtight.

For everyday stuff like sending messages or double-checking encryption, we open the door to public-facing nodes. These don't vote on blocks but handle your app chats, pass along locked messages, and run security scans per the smart contract rules—like confirming top-notch encryption and key handling. Permission smart contracts offer simple check-ins (think "transactionAllowed" or "connectionAllowed") so any node can verify if a move is okay.

Only approved accounts can launch or tweak key contracts—enforced right in the protocol, with every client pinging the permission checker first. Updates trigger alerts (like "AccountPermissionsUpdated"), so the whole network stays sharp on who's in or out, adapting to new rules on the fly.(refer Fig-3.)

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

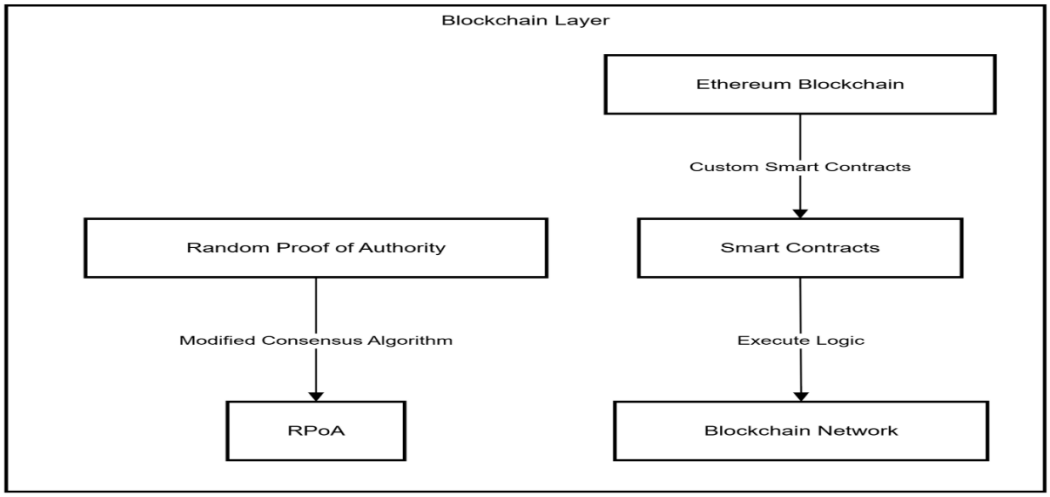

Fig-3 Blockhain Layer with RPoA enablement and Smart contract based message storation in the Blockchain.

Figure 3 represents the blockchain layer of the proposed system. The Ethereum blockchain forms the underlying platform on which custom smart contracts are deployed. These smart contracts execute application-specific logic and interact with the blockchain network to manage secure message-related records. In parallel, a modified consensus approach based on Random Proof of Authority (RPoA) is used, where validator selection is performed through a modified authority-based mechanism. Together, the smart contracts and the RPoA-enabled consensus ensure secure, trusted, and decentralized operation of the blockchain layer.

IV Mathematical Working Model

A. **Private Key Generation**

Elliptic curve cryptography (ECC) is used to generate private key. The process starts with selecting a strong, random number that serve as the foundation of the private key. Cryptographic libraries are used to secure random number generators (RNGs) to produce these values, it is relying on mathematical functions in Eq 1 that incorporate factors like system events or hardware-based randomness.

A cryptographically secure random integer d within the range [1, n−1] where n is the order of the elliptic curve group

Public key Q using the elliptic curve point multiplication

$$Q=d\times G$$

Eq.1

Where G is the base point on the elliptic curve

The private key is a randomly generated 256-bit number. Let's denote it as k_{priv}

$k_{\text{priv}} \in Z_N$

Where N is the order of the elliptic curve used.

B. Public Key Generation:

The public key is derived from the private key using Elliptic Curve Cryptography (ECC). Let G be the generator point on the elliptic curve. The public key k_{pub} is computed as:

$k_{\text{pub}}=k_{\text{priv}}G$

C. Certificate Retrieval

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

When Alice wants to send a message to Bob, she retrievea Bob's certificate from the Organization server Node. This certificate includes Bob's public key
k_{pub}Bob.

D. Sending a Secure Message

Alice generates the shared master secret key K_{shared}using her private key k_{priv}Aliceand Bob's public key k_{pub}Bobvia Elliptic Curve Diffie-Hellman (ECDH)(Fig-4).

K_{shared}=k_{priv}Alice.k_{pub}Bob

Since k_{pub}Bob=k_{priv}Bob·G, the shared secret becomes-

K_{shared}=k_{priv}Alice· (k_{priv}Bob·G)-

A Sequence Key S_K is derived from K_{shared}using a Hash-based Key Derivation Function (HKDF)

E. Application in Diffie-Hellman (ECDH)

In **ECDH key exchange**, HKDF is used to derive SequenceKeys **(SK)** and Message **Keys (MK)**:

- **Sequence Key (SK)** is derived from the shared secret-

$$SK = HKDF(Kshared, "sequence\ key\ info")$$

From the Chain Key, a Message Key M_K is derived:

$$MK = HKDF(CK, "message\ key\ info")$$

- **Ratchet Forward Mechanism-**

For every new message, the Sequence Key is updated using a cryptographic hash function:

$$SKnew = Hash(SK)$$

The Message Key is derived from the updated Sequence key -$MKnew = HKDF(SKnew, "message\ key\ info")$

F. **Receiving a Secure Message-**

Bob's app generates the Message Key using the same mechanism-

$$MK = HKDF(CK"message\ key\ info")$$

Bob updates the Sequence Key after generating the Message Key:

$$SKnew = Hash(SK)$$

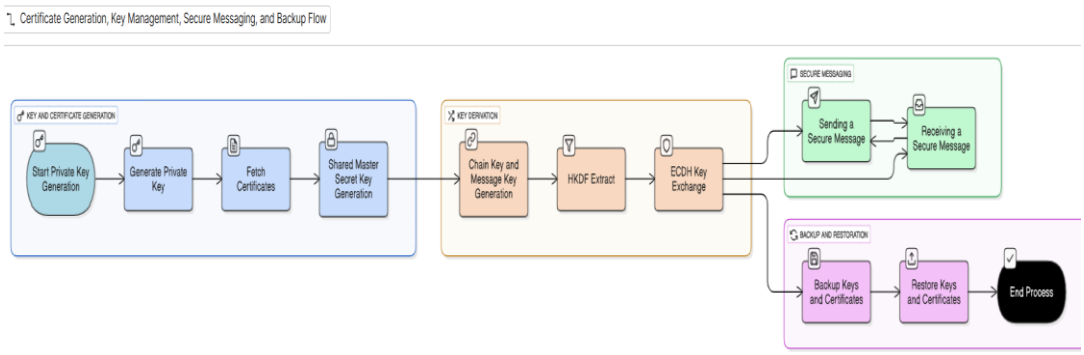

Fig-4 is the showing the procedure of Sending and Securely Receiving Messages over a secure Medium on the Ethereum Blockchain network

Figure 4 illustrates the complete workflow for certificate generation, key management, secure message transmission, and backup and restoration in the proposed system. Initially, cryptographic keys are generated, and user certificates are obtained to establish identity. A shared secret key is then derived using key exchange and key derivation mechanisms. This derived key is used to securely send and receive encrypted messages over the Ethereum blockchain network. Additionally, the figure shows the backup and restoration process, where cryptographic keys and certificates are securely stored and can be recovered when required, ensuring continuity and reliability of communication.

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

IV Implementation

A. Deployment of RPoA smart Contract

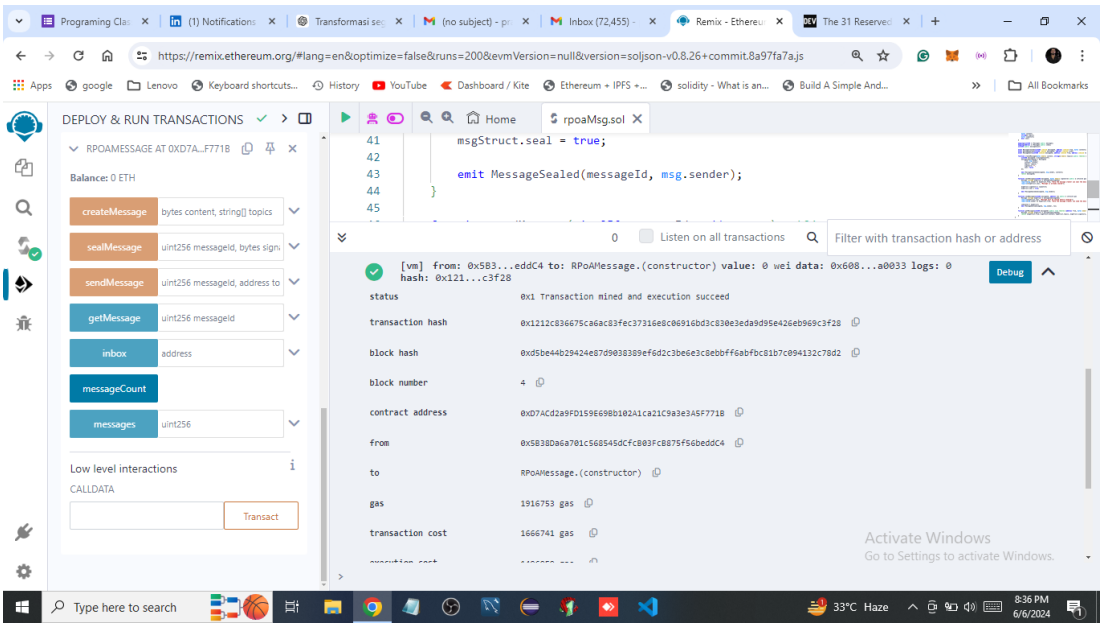

Fig--5 deloying RPoA smart contrcat on Ethereum

Figure 5 shows the deployment of the Random Proof of Authority (RPoA) smart contract on the Ethereum blockchain using the Remix development environment. The figure illustrates the execution of the contract deployment transaction, including transaction status, gas consumption, and successful contract creation. This deployment enables the blockchain network to execute RPoA-based consensus logic and manage secure message-related operations through smart contracts.

B. Message Creation

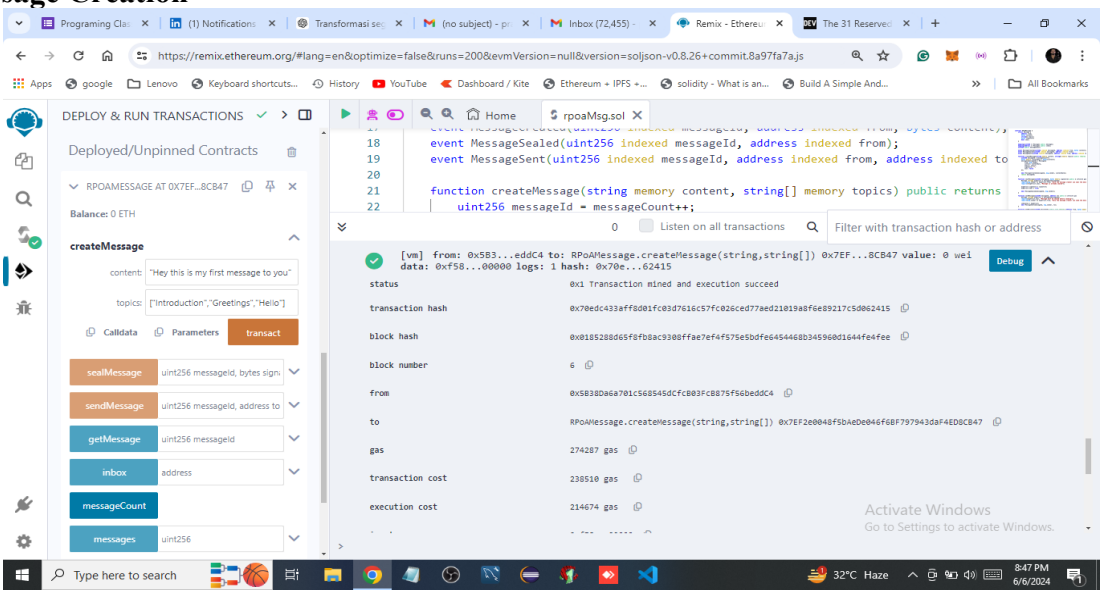

Fig-6 Message creation RPoA smart contrat on Ethereum

Figure 6 illustrates the process of creating a message using the deployed RPoA smart contract on the Ethereum blockchain. The figure shows the invocation of the message creation function through the Remix interface, where message content and associated parameters are submitted as a transaction. Upon execution, the transaction is successfully recorded on the blockchain, confirming secure message creation under the RPoA-enabled smart contract.

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

D . Message Sending

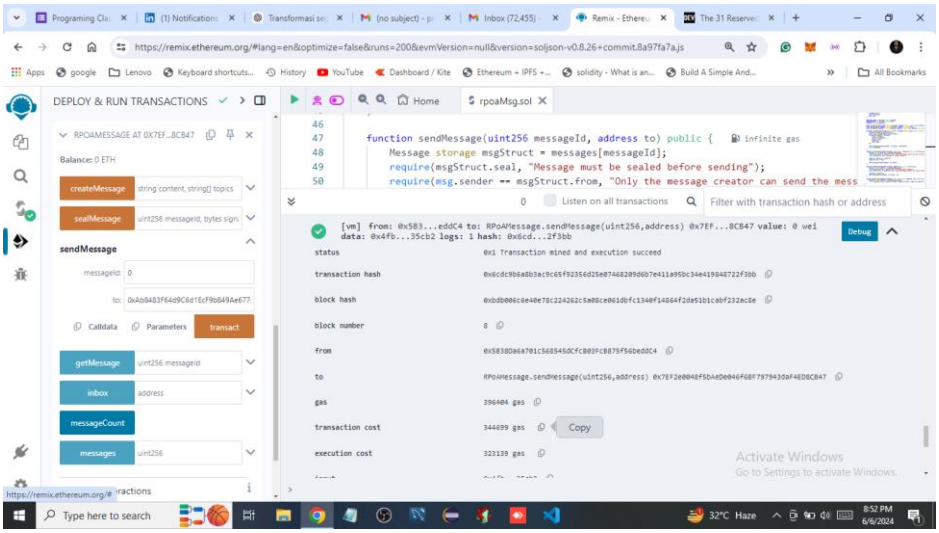

Fig-7 Message sending using RPoA smart contrat on Ethereum

Figure 7 demonstrates the process of sending a message using the RPoA-based smart contract on the Ethereum blockchain. The figure shows the execution of the message sending function, where a previously created and sealed message is transmitted to the intended recipient address. The successful transaction execution confirms that the message is securely processed and recorded through the RPoA-enabled smart contract mechanism.

E. Receiving of Message

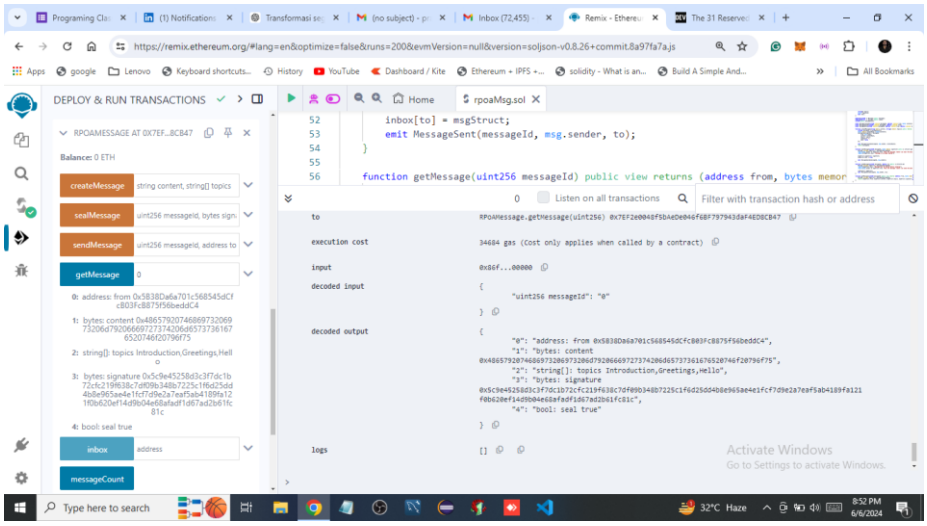

Fig-8 Message Recieving using RPoA smart contrat on Ethereum

Figure 8 illustrates the process of receiving a message through the RPoA-based smart contract on the Ethereum blockchain. The figure shows the invocation of the message retrieval function, where the intended recipient accesses the stored message using its identifier. The successful execution confirms that the message is securely retrieved from the blockchain while maintaining access control enforced by the smart contract.

F. Securing Message over the Blockchain Network

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

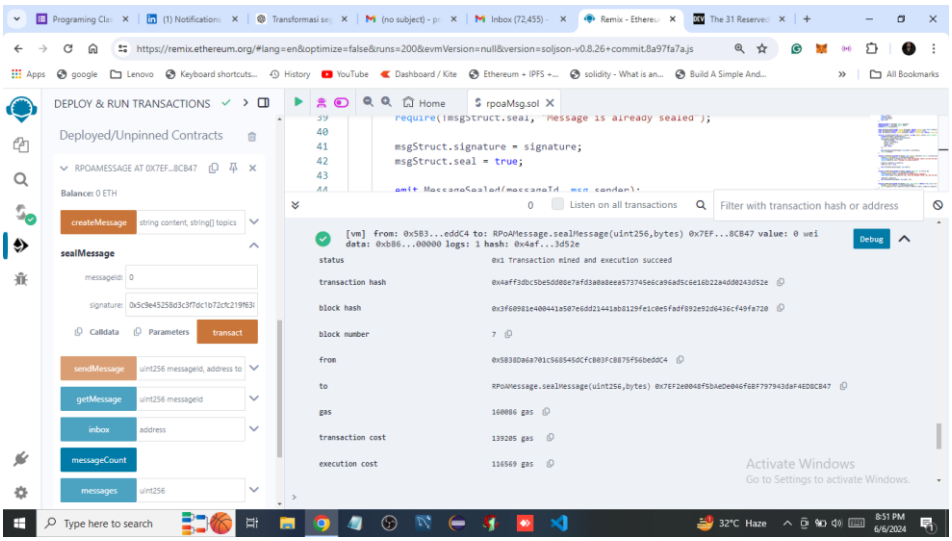

Fig-9 Securing Message using RPoA smart contrat on Ethereum

Figure 9 demonstrates the process of securing a message using the RPoA-based smart contract on the Ethereum blockchain. The figure shows the execution of the message sealing function, where the message is cryptographically secured before transmission. The successful transaction confirms that the message is marked as sealed, ensuring that only authorized actions such as sending or retrieval can be performed on the secured message.

G. Acknowledgement of Sealed Message on the Blockchain

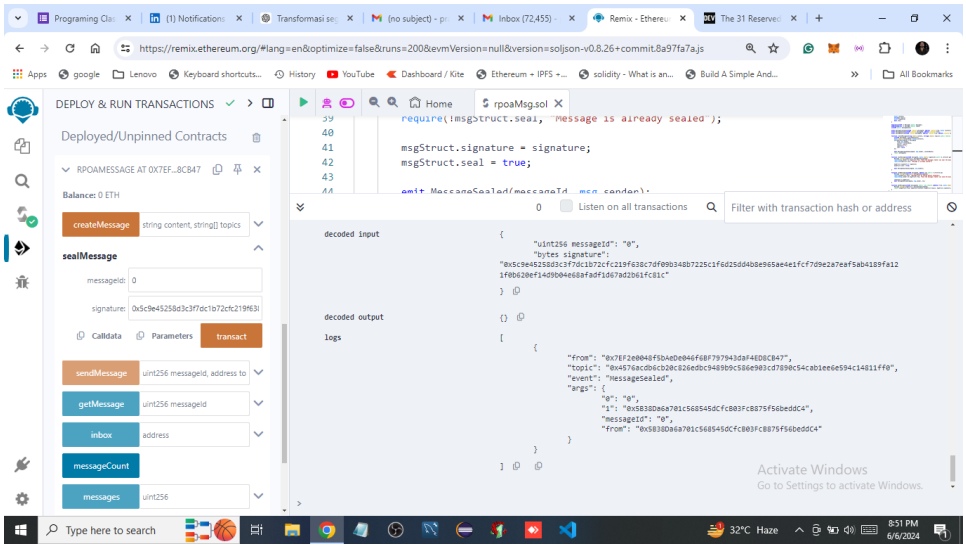

Fig-10 Message sealing using RPoA smart contrat on Ethereum

Figure 10 illustrates the message sealing operation performed using the RPoA-based smart contract on the Ethereum blockchain. The figure shows the execution of the sealing function, where a digital signature is applied to the message to finalize its secure state. Once sealed, the message becomes immutable and ready for controlled transmission or retrieval as enforced by the smart contract logic.

V Results and Discussion

For real-time oversight of the De-chat platform, we employed Prometheus, an open-source monitoring solution, integrated with Node Exporter. Node Exporter collects hardware and kernel metrics—such as CPU utilization, memory consumption, and power metrics—from Linux hosts via an HTTP endpoint (default port 9100). Prometheus scrapes these time-series data by configuring the

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

prometheus.yml file to include localhost:9100 in the scrape_configs section, enabling customizable performance insights.
Smart Contract Testing and Deployment Workflow-Testing commenced with network initialization using selected validator nodes, followed by selection of De-chat smart contracts. A controlled environment was established on the NSC Lab testbed with four server instances, deploying an Ethereum network via Remix IDE comprising four full nodes and seven light nodes.
Integration with MetaMask involved wallet installation, configuration to the Sepolia testnet, address capture, and deployment through Remix's "Deploy & Run Transactions" panel. The complete De-chat contract was subsequently deployed on the permissionless Sepolia testnet, validating scalability and feasibility in a live Ethereum environment.
SHA-256 excels in blockchain-scale ops, uniquely identifying transactions to thwart double-spends, aligning with objectives to benchmark faster, safer E2E against traditional methods.Time trials (up to 500-char messages) pit AES, RSA, and SHA-256; SHA-256 shows lowest latency and memory footprint across scaling nodes.Memory benchmarks confirm SHA-256's efficiency in bytes, ideal for high-volume social platforms.
Proposed RPoA refines PoA, stakeholders vote validators by stake-weighted authority (e.g., tiered org roles), limiting 4-10 per block; rule-compliant leaders earn rank-boosting incentives, enabling server-side E2E on permissioned DLT for secure social prototyping.

A. Comparison of different End 2 End Encryption Algorithms in Time
To compare the E2EE, with lesser processing time and more secure method with Blockchain technology.
Comparison of End 2 End Encryption Algorithms
AES,RSA and SHA -256 are considered for messages length upto 500 characters

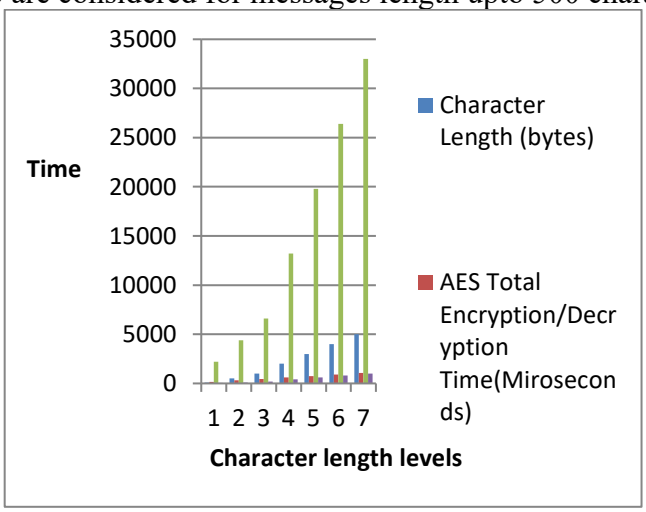

Fig-11 Comparison of different E2EE Algorithms Time

Figure 11 presents a comparison of the processing time required by different end-to-end encryption (E2EE) algorithms for varying message lengths up to 500 characters. The figure illustrates how encryption and decryption time increases with message size, allowing evaluation of the computational efficiency of AES, RSA, and SHA-256. This comparison highlights the relative performance of each algorithm in terms of time overhead for secure message processing.

E2EE with large no of Nodes(Objective 2)

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

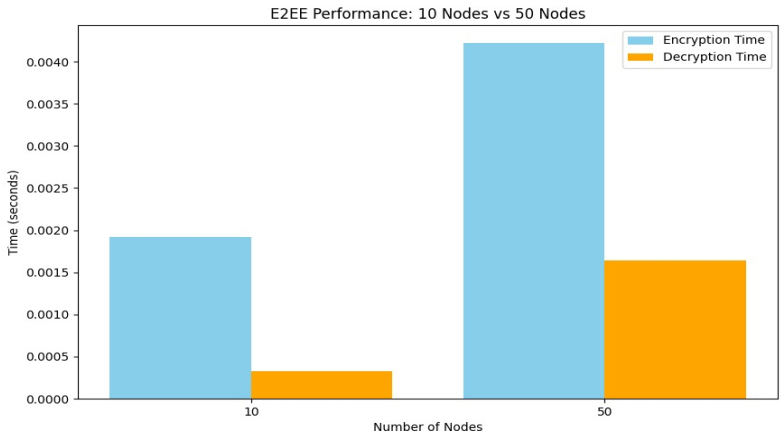

Fig-12 E2EE with large no of Nodes

Figure 12 illustrates the performance of end-to-end encryption when the number of participating nodes increases. The figure compares encryption and decryption time for scenarios with fewer nodes and a larger number of nodes. The results indicate that as the number of nodes increases, the overall processing time also increases, reflecting the scalability impact of E2EE operations in a multi-node blockchain-based communication environment.

Comparison of Different E2EE Algorithm memory usage in Bytes

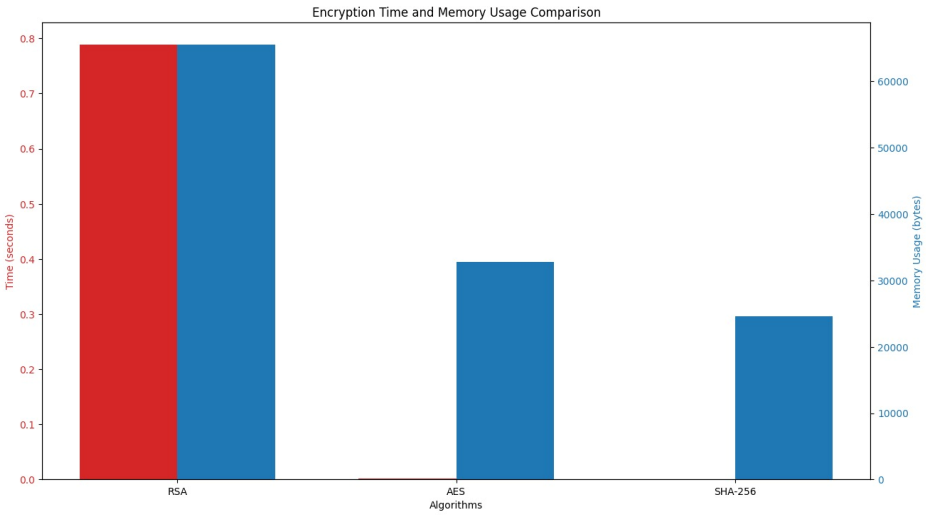

Fig-13 Comparison of Different E2EE Algorithm's Memory usage in Bytes

Figure 13 presents a comparison of memory usage for different cryptographic algorithms used in the end-to-end encryption process. The figure illustrates the memory consumption associated with RSA, AES, and SHA-256 during encryption-related operations. The results show that RSA requires higher memory due to its asymmetric key operations, while AES and SHA-256 exhibit lower memory usage, indicating their suitability for resource-constrained and large-scale secure messaging environments.

Our 3rd objective is to implement End 2 End Encryption using Blockchain Technology on social media platform simulated environment. The protocol uses elliptic curve point multiplication to achieve secure data transfer nodes. Proposed architecture prototype would make E2EE on social network platform at server-side end for authenticating and manipulating the user information by means of permission based Distributed ledger.A limited number of stake holders (limited to 4 to 10) are present for each new block, so the appointed stake holders are there to validate all transactions

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

in a new Block, The stake holders who have the highest priority will be considered the one who follows All written rule in the smart contract and it Rank got increased by this Procedure, this rank would be measured by Incentive fees he gets by following the rules of the network as reward.

B. Energy Consumption

Gas- is an abstract unit, Each transaction in a smart contract consumes gas, which represents the computational effort required to execute the transaction.the Energy taken in a time interval monitored on Testbed is recorded in Excel in Two scenarios.First with 3 server nodes and 7 light nodes and second with 4 Server nodes and 7 light nodes

1. Gas consumption to actual power usage by considering the computational resources required per gas unit, gas consumption to actual power usage by considering the computational resources required per gas unitTo calculate Difference between power consumption with 3 server and 7 Light nodes.

$$\text{Energy(Joules)} = \text{GasUsed} \times \text{Power(W)} \times \text{ExecutionTime(s)}$$

The analysis shows the amount of gas required for executing 120 transactions is .0000316 ether while the gas required for 120 transactions with 4 server nodes and 7 light nodes is 0.0000000003822 ether this shows a small change in server nodes will significantly lower the gas price.

- To calculate Difference between Gas consumption

Difference between Gas consumption= Gas Consumption (with 3 server and 7 light nodes)-Gas Consumption (with 4 servers and 7 light nodes) =0.01229877656

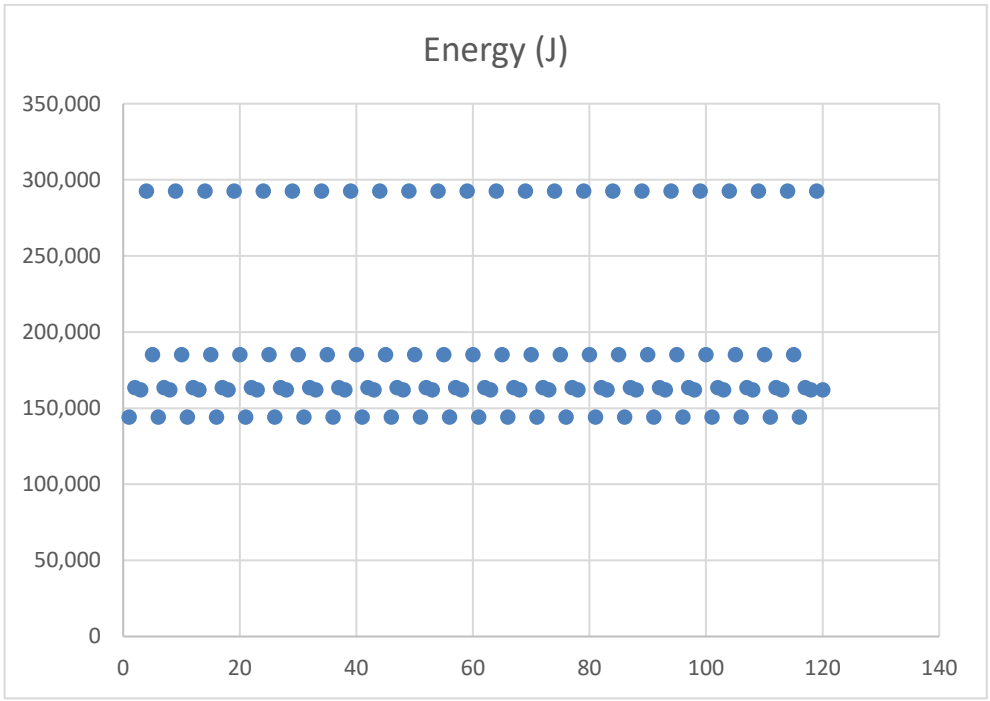

Fig-14 Energy Consumption In Joules For 2 Minutes With 3 Server Nodes And 7 Light Nodes(X-Axis-Time, Y-Axis-Energy)

Figure 14 illustrates the energy consumption measured in joules over a duration of two minutes when the network operates with three server nodes and seven light nodes. The X-axis represents time, while the Y-axis represents energy consumption. The plotted values show the variation in energy usage across different nodes during continuous operation of the blockchain-based messaging system.

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

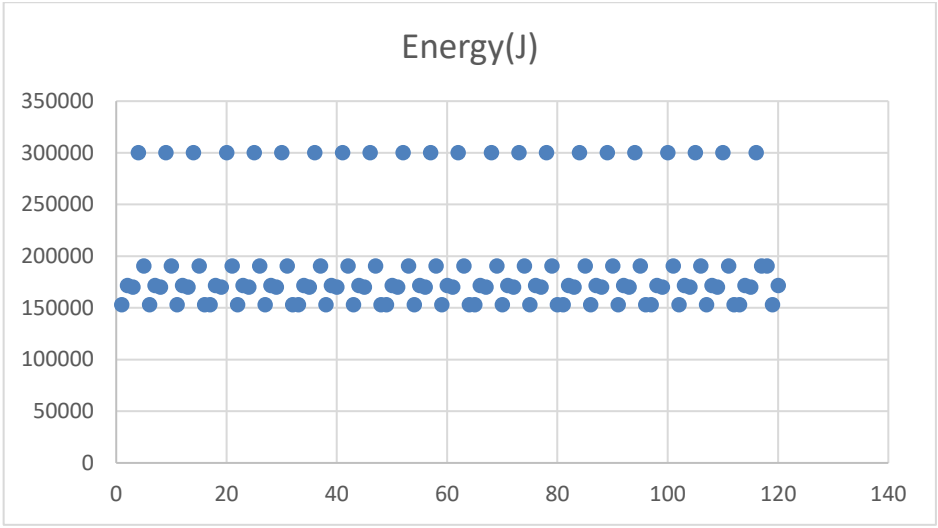

Fig-15 Energy Consumption in Joules for 2 Minutes With 4 Server Nodes And 7 Light Nodes(X-Axis- Time, Y-Axis-Energy)

Figure 15 shows the energy consumption in joules over two minutes when the number of server nodes is increased to four while keeping seven light nodes constant. The figure highlights the impact of adding an additional server node on overall energy consumption, as observed through the recorded energy values over time.

C. Comparison of power consumption of Blockchain network with 3,4 servers 7 light nodes

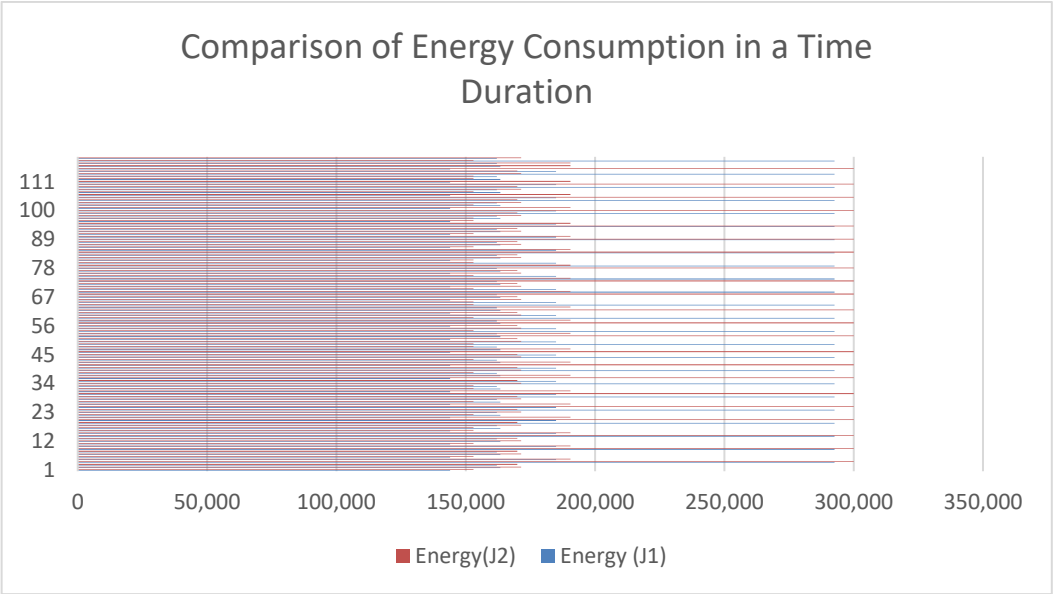

Fig-16 Comparison of power consumption of Blockchain network with 3,4 servers and 7 light nodes

Figure 16 compares the power (energy) consumption of the blockchain network operating with three server nodes and four server nodes, while keeping seven light nodes constant. The figure illustrates the variation in energy usage over the same time duration for both configurations. The comparison highlights the impact of increasing the number of server nodes on the overall power consumption of the blockchain-based network.

VI Conclusion

Received: November 10, 2025

International Journal of Applied Mathematics
Volume 38 No. 10s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

In proposed Reliable Proof of Authority(RPoA) is a concept where users of the network vote and elect the validators with one delegated Authority weightage of votes among stakeholders, the more the stake one is having in the network the more weightage he will get to vote, suppose in any organization there are 4 level of users from Top management to Employee level then weightage of vote will be given from top to bottom respectively.A limited number of stake holders (limited to 4 to 10) are present for each new block, so the appointed stake holders are there to validate all transactions in a new Block, The stake holders who have the highest priority will be considered the one who follows All written rule in the smart contract and it Rank got increased by this Procedure, this rank would be measured by Incentive fees he gets by following the rules of the network as reward.One important feature of this system is that the receiver can always verify exactly who sent a dedicated message. This prevents "denial of service" situations where a sender might falsely claim, "I did not send this message." Since the sender's private key is required to encrypt and sign the message, it is impossible for anyone else to have sent it without access to that key. This ensures that the sender cannot deny their involvement, maintaining strong message integrity and non-repudiation.

In summary, the authentication tag acts as a robust safeguard for message integrity. Only if the authentication tag, IV, and encrypted message all match correctly will decryption succeed. Otherwise, the message remains unreadable and secure. This process ensures that only the intended receiver can access the original message, and any attempt to tamper with the message will be instantly detected and rejected by the system.

Reference

1. Abd Rabou, A. (2025). The impact of Web3 on platform economy: A transaction cost economics perspective (Master's thesis, Politecnico di Torino). Politecnico di Torino Institutional Repository.
2. Ahanger, T. A., Aljumah, A., Perumal, T., & Khan, F. A. (2022). Reputation-based blockchain framework for secure UAV communication networks. IEEE Access, 10, 77102–77115.
3. Antonopoulos, A. M., & Wood, G. (2019). Mastering Ethereum: Building smart contracts and DApps. Sebastopol, CA: O'Reilly Media.
4. Baig, M. I., Qureshi, K. N., Bashir, A. K., & Jeon, G. (2022). Improved Raft consensus for blockchain-enabled IoT supply chain systems. IEEE Transactions on Industrial Informatics, 18(9), 6225–6235.
5. Carranza, H., Bustamante, M., Carranza, A., & Muniganti, S. (2024). Secure chat application with an end-to-end encryption. In Proceedings of the 10th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'24) (Paper No. CIST 140). https://doi.org/10.11159/cist24.140.
6. Mao, T. Nie, H. Sun, D. Shen and G. Yu, "A Survey on Cross-Chain Technology: Challenges, Development, and Prospect," in IEEE Access, vol. 11, pp. 45527-45546, 2023, doi: 10.1109/ACCESS.2022.3228535.
7. Kang, Y., Li, Q., & Liu, Y. (2022). Trusted data analysis and consensus mechanism of product traceability based on blockchain. Computational Intelligence and Neuroscience, 2022, Article 3035231. https://doi.org/10.1155/2022/3035231.

Received: November 10, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

8. King, Nancy J., and V.T. Raja. 2012. "Protecting the Privacy and Security of Sensitive Customer Data in the Cloud." Computer Law & Security Review 28(3): 308–19. doi:10.1016/j.clsr.2012.03.003.
9. Kiran Kumar Nalla. 2024. "Securing Chat Applications: Strategies for End-to-End Encryption and Cloud Data Protection." World Journal of Advanced Engineering Technology and Sciences 13(2): 528–40. doi:10.30574/wjaets.2024.13.2.0634.
10. Knodel, Mallory, Andrés Fábrega, Daniella Ferrari, Jacob Leiken, Betty Li Hou, Derek Yen, Sam de Alfaro, Kyunghyun Cho, and Sunoo Park. 2024. "How To Think About End-To-End Encryption and AI: Training, Processing, Disclosure, and Consent." doi:10.48550/ARXIV.2412.20231.
11. Krishna Prakasha, K., and U. Sumalatha. 2025. "Privacy-Preserving Techniques in Biometric Systems: Approaches and Challenges." IEEE Access 13: 32584–616. doi:10.1109/ACCESS.2025.3541649.
12. Kumar, Sanjay, and Deepmala Sharma. 2023. "Key Generation in Cryptography Using Elliptic-Curve Cryptography and Genetic Algorithm." In RAiSE-2023, MDPI, 59. doi:10.3390/engproc2023059059.
13. Lampropoulos, Konstantinos, Apostolis Zarras, Eftychia Lakka, Polyanthi Barmpaki, Kostas Drakonakis, Manos Athanatos, Herve Debar, et al. 2023. "White Paper on Cybersecurity in the Healthcare Sector. The HEIR Solution." doi:10.48550/ARXIV.2310.10139.
14. Laroiya, Chetna, Deepika Saxena, and C. Komalavalli. 2020. "Applications of Blockchain Technology." In Handbook of Research on Blockchain Technology, Elsevier, 213–43. doi:10.1016/B978-0-12-819816-2.00009-5.
15. Lee, Edward A., Soroush Bateni, Shaokai Lin, Marten Lohstroh, and Christian Menard. 2021. "Quantifying and Generalizing the CAP Theorem." doi:10.48550/ARXIV.2109.07771.
16. Li, Yijing, Ran Bi, Nan Jiang, Fengqiu Li, Mingsi Wang, and Xiangping Jing. 2024. "Methods and Challenges of Cryptography-Based Privacy-Protection Algorithms for Vehicular Networks." Electronics 13(12): 2372. doi:10.3390/electronics13122372.
17. Liberati, Alessandro, Douglas G. Altman, Jennifer Tetzlaff, Cynthia Mulrow, Peter C. Gøtzsche, John P.A. Ioannidis, Mike Clarke, et al. 2009. "The PRISMA Statement for Reporting Systematic Reviews and Meta-Analyses of Studies That Evaluate Health Care Interventions: Explanation and Elaboration." Annals of Internal Medicine 151(4): W-65-W-94. doi:10.7326/0003-4819-151-4-200908180-00136.
18. Limniotis, Konstantinos. 2021. "Cryptography as the Means to Protect Fundamental Human Rights." Cryptography 5(4): 34. doi:10.3390/cryptography5040034.
19. Liu, H., Zhang, Y., Chen, X., & Wang, Z. (2023). CBFT: Credit-based Byzantine fault tolerance consensus for high-performance blockchain systems. Future Generation Computer Systems, 139, 125–138.
20. M. M. Islam, M. M. Merlec and H. P. IN, "Proof of Random Leader: A Fast and Manipulation-Resistant Proof-of-Authority Consensus Algorithm for Permissioned Blockchains Using Verifiable Random Function," in IEEE Transactions on Services Computing, vol. 18, no. 3, pp. 1655-1668, May-June 2025, doi: 10.1109/TSC.2025.3536315.

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

21. Manginte, Shofia Yunus. 2024. "Fortifying Transparency: Enhancing Corporate Governance through Robust Internal Control Mechanisms." Advances in Management & Financial Reporting 2(2): 72–84. doi:10.60079/amfr.v2i2.173.
22. Mannan, Mamun Abdul. 2024. "Data Privacy in E-Commerce: Challenges and Best Practices." In Advances in Information Security, Privacy, and Ethics, eds. Dina Darwish and Kali Charan. IGI Global, 415–40. doi:10.4018/979-8-3693-9491-5.ch017.
23. Menegay, Peter, Jason Salyers, and Griffin College. 2018. "Secure Communications Using Blockchain Technology." In MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM), Los Angeles, CA: IEEE, 599–604. doi:10.1109/MILCOM.2018.8599771.
24. Menon, V. G., Anand, M., Alazab, M., & Ravi, V. (2023). Blockchain-assisted machine learning framework for secure smart home data privacy. IEEE Internet of Things Journal.
25. Mittelstadt, Brent Daniel, Patrick Allo, Mariarosaria Taddeo, Sandra Wachter, and Luciano Floridi. 2016. "The Ethics of Algorithms: Mapping the Debate." Big Data & Society 3(2): 2053951716679679. doi:10.1177/2053951716679679.
26. Mohammed Abdul, Shezon Saleem. 2024. "Navigating Blockchain's Twin Challenges: Scalability and Regulatory Compliance." Blockchains 2(3): 265–98. doi:10.3390/blockchains2030013.
27. Mosteiro-Sanchez, Aintzane, Marc Barcelo, Jasone Astorga, and Aitor Urbieta. 2020. "Securing IIoT Using Defence-in-Depth: Towards an End-to-End Secure Industry 4.0." Journal of Manufacturing Systems 57: 367–78. doi:10.1016/j.jmsy.2020.10.011.
28. Mssassi, Souhail, and Anas Abou El Kalam. 2024. "Leveraging Blockchain for Enhanced Traceability and Transparency in Sustainable Development." In International Conference on Advanced Intelligent Systems for Sustainable Development (AI2SD'2023), Lecture Notes in Networks and Systems, eds. Mostafa Ezziyyani, Janusz Kacprzyk, and Valentina Emilia Balas. Cham: Springer Nature Switzerland, 162–77. doi:10.1007/978-3-031-54318-0_14.
29. Munjal, Kundan, and Rekha Bhatia. 2023. "A Systematic Review of Homomorphic Encryption and Its Contributions in Healthcare Industry." Complex & Intelligent Systems 9(4): 3759–86. doi:10.1007/s40747-022-00756-z.
30. Oliveira, A. M., Rocha, Á., & Pérez-Cota, M. (2023). Blockchain and Industry 5.0: Opportunities, challenges, and future research directions. Computers & Industrial Engineering, 176, 108944.
31. Rebello, G.A.F., Camilo, G.F., Guimarães, L.C.B. et al. A security and performance analysis of proof-based consensus protocols. Ann. Telecommun. 77, 517–537 (2022). https://doi.org/10.1007/s12243-021-00896-2.
32. Sandberg, K. (2023, February). Web3 and sustainability: How we can reduce the climate impact of blockchains, how blockchains can help reduce our own. The Linux Foundation.
33. Showkat, S., & Qureshi, S. (2023). Securing the Internet of Things through blockchain approach: Security architectures, consensus algorithms, enabling technologies, open issues, and research directions. International Journal of Computing and Digital Systems, 13(1). https://doi.org/10.12785/ijcds/130109.
34. Wang, Q., Li, R., Wang, Q., Chen, S., & Xiang, Y. (2022). Exploring unfairness on proof of authority: Order manipulation attacks and remedies. In Proceedings of the 2022 ACM Asia

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Conference on Computer and Communications Security (ASIA CCS '22) (pp. 123–137). Association for Computing Machinery. https://doi.org/10.1145/3488932.3517394.

35. Wang, S., Wang, J., Wang, X., Qiu, T., & Liu, A. (2019). Accelerating practical Byzantine fault tolerance for smart grid systems. IEEE Transactions on Industrial Informatics, 15(10), 5623–5634.

Received: November 10, 2025