Applying the AuRa Consensus Model for Digital Certificate Management in a Private Ethereum Blockchain

Robiah Arifin[1*], Wan Aezwani Wan Abu Bakar[2*], Mustafa Man[3*],
Mohamad Afendee Mohamed[4], Evizal Abdul Kadir[5]
Department of Big Data, Infostructure and Network Management Centre,
Universiti Sultan Zainal Abidin, 21030 Kuala Nerus, Terengganu, Malaysia[1]
Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, 22200 Besut Campus, Besut, Terengganu, Malaysia[2, 4]
Faculty of Computer Science and Mathematics, Universiti Malaysia Terengganu, 21030 Kuala Nerus, Terengganu, Malaysia[3]
Faculty of Engineering, Universitas Islam Riau, Jl. Kaharuddin Nasution 113, Pekanbaru 28284, Riau, Indonesia[5]

***Abstract*—The issue of fake certificates has been widely identified, and their prevalence has increased significantly in recent years. This growing trend has become a global concern due to its adverse impact on educational standards. A key factor contributing to the problem is the continued reliance on manual processes for issuing and verifying certificates. To address these challenges, this study proposes the use of an authority round (AuRa) consensus algorithm for managing certificate data on the Ethereum blockchain. AuRa, a member of the proof of authority (PoA) family, facilitates consensus among nodes distributed across multiple servers and networks. This mechanism plays a vital role in preserving the integrity and decentralization of the blockchain while ensuring the security of transactional data. Furthermore, the study investigates how AuRa enables efficient certificate data transactions within a private Ethereum environment. It also evaluates the algorithm's performance in terms of transaction speed per second (TPS) and throughput per second (TGS), demonstrating its effectiveness for managing certificate transactions on a blockchain network. Then the TPS and TGS results substantiate the suitability of AuRa for digital certificate generation, evidenced by its stable and efficient performance within a controlled private server environment.**

***Keywords*—Blockchain technology; fake certificate prevention; Authority round (AuRa) algorithm; ethereum private network; Proof of Authority (PoA); certificate verification system**

I. Introduction

Certificates are important for graduate students as proof that they have completed their studies. Unfortunately, numerous job seekers use fake certificates to apply for jobs. Fake certificates were first traced in the 19th century [1]. Such certificates were issued by a nonexistent university.

Fake certificates have become a global issue and have been detected worldwide [2]. In response to the scenario, the research proposes the digital certificate using blockchain technology supported by the authority round (AuRa) consensus algorithm.

Blockchain allows participants to write and read the transactions in the network [3]. To ensure the security of blockchain, this research implements the Ethereum and AuRa consensus algorithms. The consensus algorithm serves as a security mechanism by establishing agreement among nodes to ensure the throughput is consistent, not different, and valid. This research also implemented the smart contract as a logic flow to transact data into the blockchain.

The research aims to create a digital certificate using the AuRa consensus algorithm through a smart contract and to measure the transaction speed per second (TPS) and throughput per second (TGS). The implementation is executed on a private server that consists of three datasets, each containing a different number of records and including data information such as student number, gender, and faculty. This research aims to address the issue of counterfeit certificates by generating a digital certificate using the Ethereum blockchain supported by the AuRa consensus algorithm.

The contribution of this study lies in demonstrating that the AuRa consensus mechanism is capable of supporting blockchain-based transaction processes. The effectiveness of these processes is evaluated using throughput performance indicators, namely TPS and TGS, in the context of digital certificate generation. Furthermore, the findings indicate that AuRa can effectively address issues related to the prevention of counterfeit certificates in the digital certificate issuance process.

Section II outlines the issue of counterfeit certificates and examines the underlying causes. It also provides an overview of blockchain technology, with a focus on the Ethereum platform and the AuRa consensus algorithm, highlighting their potential in addressing the problem of certificate forgery.

Section III details the proposed implementation of the method to mitigate these issues. Section IV presents the results and analysis of the implementation, followed by Section V, which discusses the findings and acknowledges the limitations of the study. Finally, Section VI concludes the study by summarizing the implementation of blockchain technology and the AuRa consensus algorithm for certificate generation, and outlines potential directions for future research.

A. Novelty

This study introduces a novel real-world deployment of the AuRa consensus algorithm for digital certificate generation, offering an empirical assessment that is largely absent in prior

*Corresponding author.

blockchain-based certificate management literature. While existing works commonly focus on conceptual frameworks or simulated environments, this research advances the field by evaluating AuRa within an operational private server setting and by utilizing two concrete performance metrics, TPS and TGS. The use of TPS and TGS as paired performance indicators provides a measurable and systematic way to assess both transactional efficiency and data-handling capability, thereby contributing a practical and reproducible evaluation methodology to the research domain.

B. Contributions

Then the findings demonstrate that AuRa delivers stable and favorable TPS and TGS performance outcomes, substantiating its suitability for real-world digital certificate generation. This work contributes empirical evidence that AuRa can reliably support certificate issuance workflows with predictable throughput and consistent operational behaviour within a private server environment.

Beyond confirming performance viability, the study provides a validated deployment model that institutions can adopt or extend for secure certificate management. Collectively, these contributions highlight the practical applicability of AuRa and position this study as one of the few that bridges the gap between theoretical blockchain designs and deployable certificate management solutions.

II. Literature Review

The main criterion for employers to choose an employee is the certificate. Employers verify certificates manually, which is time-consuming and labor-intensive. The difficulty of verifying the certificate contributes to the issue of the production of fake certificates. The problem of forged certificates persists among the higher education community [4]. Fake certificates have become a global issue, which is associated with certificate scams.

Numerous factors cause these problems. One of the factors is an inefficient student management system. The highly trustworthy system needs to be implemented to eliminate the fake certificate problem [5]. Another factor is that employers often do not complete the certificate verification process because it requires extra human resource efforts [6]. Manual verification processes also exacerbate these problems [7]. Employers must contact academic institutions manually for certificate verification, which takes significant time and effort [8].

To address the growing issue of fake certificates, blockchain technology can provide a solution. Blockchain technology is an effective method to address these problems. Many applications and methods based on a blockchain framework have been proposed to prevent fake certificate issues, as described in Table I.

Based on Table I, show the implementation of blockchain for a digital certificate. However, these studies face performance issues for the verification and authentication of digital certificates. Unfortunately, this study also did not discuss the measurement and performance results of the digital certificate.

TABLE I. Digital Certificate Management Using Blockchain

Authors	Case Study
Fang et al. (2024)[9]	Credential verification
Feng et al. (2024)[10]	Certificate authentication
Liu et al. (2024)[11]	Certificate verification
Mai et al. (2024)[12]	Certificate verification
Samarai (2024)[13]	Certificate verification

A. Blockchain Technology

Blockchain is a decentralized, distributed database used for storing transaction data and information. Blockchain was introduced as a new method of managing and storing data, replacing traditional centralized systems with decentralized systems [14]. The data storage consists of a chain of records known as blocks.

Data is stored across different nodes on different servers. The blockchain is also known as a peer-to-peer (P2P) network [15] because the nodes communicate with each other through a P2P network. Blockchain uses a storage system to store digital data and protect it with cryptography and a hash function. Cryptographic enforcement in blockchain networks ensures immutability, rendering the recorded logistics data resilient against tampering and hacking [16].

In 1991, the capabilities of blockchain were improved with the secure chain of blocks, which could be implemented through the cryptographically secure method described by Haber and Stornetta [17]. Today, public blockchains are widely available through various readily available platforms to be implemented by researchers. However, the popular and widely used ones are Bitcoin, Ethereum, and Hyperledger [18]. Ethereum was established years back, focusing on encouraging businesses to adopt the blockchain in their business activities [19]. To date, these blockchain platforms share similarities but also retain their unique features.

Both Bitcoin and Ethereum are public blockchains that allow everybody to participate in writing and reading the transactions in the network. This concept is known as the permissionless blockchain. Participants are allowed to connect and disconnect from the network at any time without permission.

B. Ethereum

Ethereum was first introduced by Vitalik Buterin in 2013. It was intended to overcome the limitations of Bitcoin. In 2014, Ethereum's development was crowdfunded in 2014 to improve the technology. Ethereum has implemented distributed data storage that allows everyone to run their own applications on the blockchain [20]. It is capable of storing data with flexible block sizes. The Ethereum engine requires support from Ethereum blockchain clients.

The Ethereum client can be deployed on development and production environments, but it has difficulties and needs expertise to set up and configure the environment. Parity and Geth are the popular and active supporters of the blockchain community [21].

However, the Ethereum clients need to be supported by a consensus algorithm to ensure the security and performance of transactions in the blockchain.

C. Consensus Algorithm

A consensus algorithm is an agreement process among nodes across different servers and networks. It supports and commits a blockchain system to maintain the consistency and decentralization of nodes [22]. Consensus algorithms enhance the security of blockchain transactions by ensuring that throughput remains consistent and valid. The consensus algorithm is functional not only in terms of security aspects, but it also provides services for reducing the execution time of transaction processes and providing the highest throughput.

The performance of consensus depends on its algorithm and protocol for executing transactions in the blockchain. Its performance also directly impacts the processing speed and scalability of the blockchain. A consensus algorithm that commits transactions in less time is more efficient. Faster consensus execution increases throughput. In addition, various consensus algorithms support blockchain transactions. The most popular and widely implemented blockchain applications are proof of authority (PoA), practical byzantine fault tolerance (PBFT), proof of stake (PoS), and proof of work (PoW). Table II shows the consensus and its specialty.

TABLE II. CONSENSUS AND SPECIALTY

Consensus	Specialty
PoS	More spikes in blockchain
PoW	Able to solve a mathematical puzzle
Proof of Luck	Implemented the random selection
Proof of Elapsed Time	Able to set up the scheduling while the timeout.
Proof of Space	Implemented in bigger size of hard disk environment
PoA	Well-suited for public blockchains deployed in private environments.

D. PoA

PoA was introduced by Wood as an alternative to PoW and PoS. PoA was introduced to overcome the limitations of PoW and PoS [23]. Therefore, the PoA has become a popular consensus algorithm because of its high-performance transaction process and security. It is able to process thousands of transactions per second without impacting its security. PoA also provides fault tolerance and efficiency.

Besides that, the PoA consensus also needs less energy and fewer computational resources to implement. It also makes efficient use of network bandwidth. This algorithm is used in permissioned blockchains and is implemented in two versions, namely AuRa and Clique. These versions contain their own advantages and disadvantages.

This research implemented the AuRa because it was originally proposed for a private network environment [24]. That is why this research chose to implement AuRa to overcome the fake certificate issue. Additionally, nearly 4,000 blockchain projects across various domains, including education, management, healthcare, and insurance, have utilized AuRa [25]. As proven, AuRa is more secure compared to Clique, and AuRa throughput is higher than PoW.

Although this study does not formally benchmark alternative consensus algorithms or certificate verification systems, the selection of the AuRa is justified based on its compatibility with permissioned blockchain environments. Unlike PoW, which is energy-intensive and better suited for public networks, or PoS, which introduces complexity in validator selection, AuRa offers deterministic block production, low energy consumption, and predictable performance. These characteristics align well with the operational needs of academic certificate issuance, where trust is centralized, and efficiency is prioritized. Future work may involve empirical benchmarking against other consensus mechanisms to further validate this selection.

III. RESEARCH METHODOLOGY

To overcome the fake certificate issues, this study proposed to implement the AuRa consensus algorithm to create the digital certificate. Fig. 1 shows the framework of implementation of AuRa to create the digital certificate.

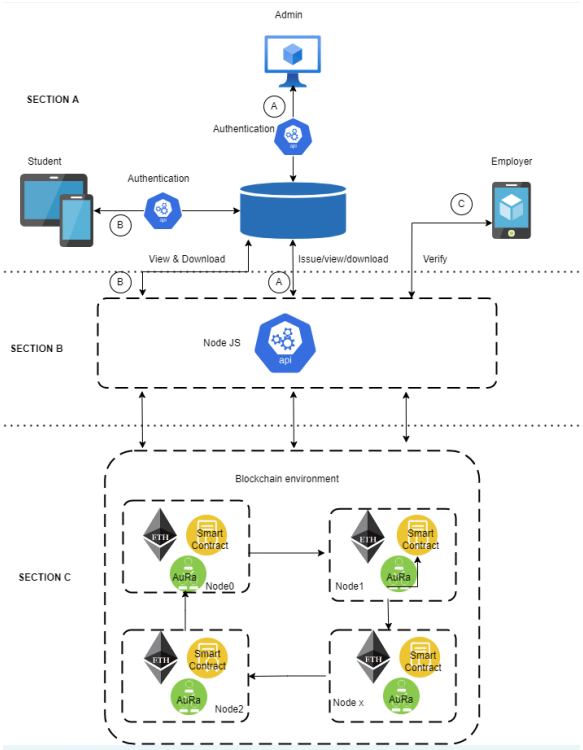

Fig. 1. Implementation of the AuRa framework.

Regarding Fig. 1, the framework to implement the AuRa consensus algorithm is divided into three sections. Section A focuses on the application layer, which comprises three primary roles: administrator, student, and employer. The administrator is responsible for issuing digital certificates through the system, ensuring that each record is correctly generated, authenticated, and securely committed to the blockchain. This role also incorporates access control mechanisms to prevent unauthorized issuance and ensures that certificate data is transmitted and stored in a tamper-resistant manner.

Students, as legitimate certificate holders, are permitted to securely access, view, and download their digital certificates via authenticated sessions within the application. Security controls implemented at this layer, such as identity verification and encrypted communication, ensure that only the rightful student can retrieve their certificate information. Employers function as external verifiers who authenticate the validity of certificates through the application's verification module. This module leverages blockchain-backed immutability to ensure that verification results cannot be falsified, providing a trusted mechanism that prevents manipulation, forgery, or unauthorized alterations of certificate records. Collectively, these security functions establish a secure, transparent, and tamper-resistant application environment.

Section B refers to the backend processes, which serve as the core operational layer supporting the application layer. The backend utilizes an application programming interface (API) to transmit and receive commands and data between the application layer and the blockchain network. These APIs manage the execution of certificate issuance requests, retrieval of certificate data, and verification queries while ensuring secure, authenticated, and structured communication between the frontend and the underlying blockchain infrastructure.

To enhance security, the backend incorporates multiple protective mechanisms. All data transmissions are encrypted to prevent interception or tampering during communication. Input validation and authentication checks are enforced for every request to mitigate unauthorized access and potential injection attacks. Transaction records are logged in an immutable audit trail to provide non-repudiation and support forensic analysis if discrepancies occur. Furthermore, the backend enforces role-based access control, ensuring that only authorized administrators, students, or employers can execute specific operations, thereby maintaining the integrity, confidentiality, and availability of the digital certificate system. These combined security measures ensure that the backend can reliably manage certificate issuance, retrieval, and verification while minimizing potential threats.

Lastly, Section C, the blockchain layer forms the foundational infrastructure of the digital certificate system, providing tamper-resistant storage, consensus validation, and secure transaction recording. This layer leverages the AuRa consensus algorithm to ensure that all certificate issuance and verification transactions are validated by a network of authorized nodes, maintaining integrity and consistency across the distributed ledger. The blockchain layer interacts with the backend via secure APIs, receiving transaction requests for certificate issuance, verification queries, and data retrieval, and returning confirmation or result status to the application layer.

Security functions at the blockchain layer are critical to maintaining trust and reliability. Each transaction is cryptographically signed and recorded in an immutable block, ensuring non-repudiation and protection against data alteration. Consensus mechanisms prevent single-node manipulation, while periodic block validation guarantees that any conflicting or unauthorized transactions are rejected. Access control is enforced at the node level, restricting operations to authorized validators.

Additionally, the blockchain maintains redundancy across nodes, enhancing system availability and resilience against node failures or malicious attacks. Combined with the secure backend and application layers, the blockchain layer ensures a comprehensive, end-to-end security framework for the digital certificate management system, enabling reliable issuance, verification, and auditability of certificates.

Lastly, the framework is comprehensively limited to certificate generation and verification activities. To guarantee the authenticity, security, and reliability of the produced certificates, both activities are required to systematically progress through the processes outlined in Sections A, B, and C.

A. AuRa Consensus Algorithm

The AuRa consensus algorithm was proposed for the Parity client. Parity is one of Ethereum's clients that applied Rust as a programming language. It has been implemented by Lava, VecHain Thor, xDai DPOS network, Microsoft Azure (for deployment only), and Kovan Testnet. This consensus became popular because it was widely adopted in blockchain applications.

This consensus assumes the authorities are honest and the network is synchronous among the nodes. Besides that, AuRa consists of four steps, including transaction pending, proposed block, voting, and as depicted in Fig. 2.

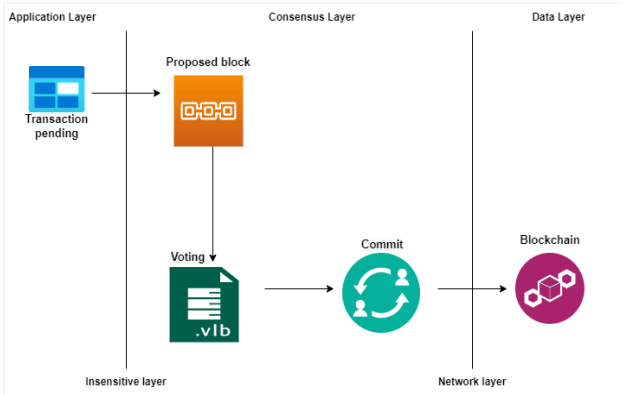

Fig. 2. The steps of the AuRA process.

In Fig. 2, the AuRa algorithm starts by assigning a leader; each authority has its turn to assign as a leader using the Round Robin Algorithm. Then the leader will propose a block, and the authorities will decide if the proposed block is accepted or not. The decision or voting process is based on Eq. (1).

$$2f + 1 \leq n \quad (1)$$

Regarding Eq. (1), where f is a faulty node, n is the number of nodes. In that case, the number of faulty nodes must be at least as many as the number of nodes 50%. Additionally, only trusted authorities are able to add the block to the blockchain. Trusted authorities that call n number and implement the majority among the authorities able to create a node.

After the leader is assigned, the leader will propose the block in the block proposal. Then the voting process will execute on block acceptance, an authority's members will vote on the block proposed by a leader, whether that block is able to commit or

not. If the leader or node is not valid, the process assign new leader will reprocess.

B. Create and Configure AuRa Engine

The AuRa engine comprises the Ethereum blockchain. Once the AuRa engine was installed, the Ethereum blockchain was automatically installed. Each AuRa engine was installed on a different server and is called a node. This research created the nodes at private servers. Each node will act as an authority on the network and issue blocks in the transaction process. Fig. 3 shows the creation of the AuRa engine. AuRa engine created at port xxxx:xxxx, located in the parity folder. These engines are connected with the xxxx.toml file.

Coding 1: Create the AuRa engine

```
[1]  run –rm -ti  -p xxxx:xxxx
[2]  -v ~/.local/share/parity/docker/:/home/parity/
[3]  .local/share/parity/ parity/parity:v3.0.0
[4]  --config /home/parity/.local/share/parity/xxxx.toml
[5]  --jsonrpc-interface all
```

Fig. 3. Create AuRa engine.

Then, after completing the coding to create the AuRa engine, the next step is to create the authority account, as in Fig. 4. The authority account created at port xxxx:xxxx is similar to the port at the AuRa engine. The location and toml file are also similar to the AuRa engine.

Coding 2: Create authority account

```
[1]  run –rm -ti  -p xxxx:xxxx -v
[2]  ~/.local/share/parity/docker/:/home/parity/
[3]  .local/share/parity/ parity/parity:v3.0.0
[4]  --config/home/parity/.local/share/parity/xxxx.toml
```

Fig. 4. Create an authority account.

Based on the execution of Fig. 4, the results in Fig. 5 illustrate the authority account successfully created. The authority account that was created is 0x008ce3c5316c23c8fdee747b0ae59188879xxxxx.

Result : Create authority account

```
[1]  {"jsonrpc":"2.0","result":"0x008ce3c5316c23c8fdee747b0ae59188879xxxxx","id":0}
```

Fig. 5. Result achieved.

C. Smart Contract

Smart contracts were proposed by Nick Szabo in the early 1990s. A smart contract is a platform and a high-level programming language to execute the mining process automatically [26]. The few available programming languages for smart contracts are Solidity, Mutan, Viper, and Chaincode.

The function of smart contracts is to support the Ethereum blockchain. It was developed and stored in the Ethereum blockchain. A smart contract is a programmable code that executes predefined commands stored on a blockchain. These commands include attributes, conditions, and rules that must be met before data can be recorded on the blockchain. Smart contracts leverage programmed logic to autonomously enforce access control lists and system constraints defined within the blockchain infrastructure [27]. In this research, a smart contract was implemented to facilitate the execution of commands for storing data on the blockchain.

Additionally, this research implemented Solidity as a programming language to develop the smart contract code. The smart contract was developed and tested on the Remix integrated development environment (IDE) platform. After the smart contract was successfully tested, it was rewritten on the Truffle framework. At the truffle framework, the smart contract was running using the truffle command to migrate the smart contract and the AuRa consensus algorithm.

Before executing the migration process, the configuration between nodes and the truffle framework must be considered. Fig. 6 captures the coding of the configuration of the nodes and the truffle framework. Line 1 indicates the server to create the AuRa engine. It consists of information about the host (line 3), port (line 4), and network ID at line 5.

Configuration nodes and truffle framework

```
[1]  Networks:{
[2]  Development: {
[3]  Host:"xxx.xx.x.x"
[4]  Port: xxxx,
[5]  Network_id:"xxxxx"
[6]  },
[7]  }
```

Fig. 6. Configuration of nodes and truffle.

Additionally, after configuring the nodes and the truffles framework as depicted in Fig. 6, the next step is to integrate both nodes and truffles. Used the truffle init command, the nodes and truffles will migrate as illustrates at Fig. 7.

Fig. 7. The migration nodes and truffles.

The smart contract address is crucial for integrating transaction data into the blockchain. Additionally, the migration process generates the application binary interface (ABI) file, which is essential for the transaction process.

Smart contracts play a pivotal role in the blockchain ecosystem by enabling decentralized automation that eliminates the reliance on traditional intermediaries. Through this cryptographic execution, these protocols secure data integrity and ensure absolute immutability against unauthorized

tampering. Furthermore, the embedded algorithmic logic allows for the precise enforcement of operational conditions and strict access control mechanisms across the network.

This automated governance enhances system transparency and auditability, providing an immutable ledger of verifiable transaction histories. Consequently, smart contracts serve as the foundational backend infrastructure that drives the functionality of decentralized applications and modern digital economies. Any modifications made to the smart contract logic do not affect system operations, provided that the updated contract undergoes the requisite node reconfiguration and truffle deployment processes.

D. Integration and Implementation

Before certificate issues, Python was implemented as a programming language to generate the certificate that embeds the AuRa consensus algorithm and smart contract. To ensure the AuRa consensus algorithm and smart contract are able to communicate with each other, the Web3 library is implemented. Web3, commonly known as web3.py, is a Python library that is able to interact with smart contracts and Blockchain.

It's commonly implemented in decentralized applications (dApps) to assist the sending transactions process, communication with smart contracts, writing and reading the block data in JavaScript Object Notation (JSON) format, to incorporate the solution for fake certificate issues. The certificate information was converted into JSON format to transact data into the blockchain through the AuRa consensus algorithm. The important data, such as student number, name, result, program, and faculty, were stored in a blockchain database. These data are important and must be accurate and reliable to avoid the fake certificate issues.

To evaluate the framework's practical applicability, the experimental setup simulates a multi-institution infrastructure by deploying a private Ethereum network across several servers. These nodes are strategically distributed across two geographically distinct locations to measure crucial network metrics, including reliability, synchronization overhead, and transaction speeds under realistic network latencies. This distributed private network closely mirrors a real-world, cross-organizational deployment where multiple independent authorities participate in the certificate verification process. Within this environment, digital certificates were generated and evaluated using three distinct datasets, each encompassing varying data types to test the system's robustness against realistic workloads.

Additionally, to ensure integrity and immutability, a private key is utilized in the generation of a digital certificate. A private key is a randomly generated set of numbers used in cryptographic processes. In this research, private keys are generated for selected users who possess the authority to create digital certificates. This research also grants administrators the explicit authority to revoke private keys upon theft or loss. To further prevent unauthorized administrative exploitation, key deployment is restricted to a single-user concurrency model.

E. Hardware and Software

To implement the proposed method in this research, several servers are required. Specifically, four servers were utilized to establish individual nodes, with each server physically located at a distinct on-premises location. The implementation relies on the AuRa consensus algorithm and the Ethereum blockchain engine, both of which were installed and configured on every node across all servers.

Additionally, a smart contract was developed, which was installed and configured on only one of the nodes. Other supporting software tools, including Truffle, Remix IDE, and Ganache, were accessed via their online versions, eliminating the need for local installation on the servers.

IV. Result and Analysis

This study aims to demonstrate that the AuRa algorithm is executable and effective for resolving certificate management. Within the context of digital certificate management, TPS and TGS serve as sufficient and primary evaluation metrics; TPS directly indicates the system's throughput capacity during high-volume issuance periods, while TGS reflects the responsiveness and latency of real-time certificate verification. For a rigorous performance, AuRa was implemented on the myCert platform using authentic graduation certificate data to generate and validate its operational suitability under realistic workloads.

MyCert is an in-house application that manages digital certificates, including the generation and verification process. It uses Python as its programming language and PyCharm as its Integrated Development Environment (IDE). The AuRa consensus algorithm is installed and embedded on myCert platform.

A. Geographic Location

The study deployed identical datasets across three geographically dispersed locations connected through a wide area network (WAN) topology, with each site hosting a data instance linked through intermediary network nodes. This distributed configuration reflects real-world environments in which data must be transmitted and validated across multiple infrastructures. The datasets were processed during peak operational periods to evaluate system behaviour under high-load conditions, specifically to measure the performance of the AuRa consensus mechanism in terms of TPS and TGS. This approach enables a more precise assessment of how network congestion and distributed processing influence overall performance within a WAN-based architecture.

TPS quantifies the number of successful transactions processed per second. It aims to measure the performance of AuRa, focused on transaction speed per second. This study used three sets of data: set 1 consists of 1,021 data, set 2 includes 2,435 data, and set 3 comprises 3,422 data. Three sets of data refer to the number of graduation student starting 2023 to 2025.

Based on three sets of data, the AuRA is used to generate the certificate. The performance of AuRa is measured by TPS and TGS. Fig. 8 illustrates the result of TPS. Based on the TPS result at different locations, the Data Set 1 is 31.59, Data Set 2 is 29.69, and Data Set 3 is 18.59. The same location for TPS result is 40.32,32.03 and 19.25.

Regarding Fig. 8, it is observed that the number of transactions significantly impacts the system's throughput performance, measured in TPS. However, the TPS results across

these datasets are not consistent, indicating that factors beyond transaction volume, such as network latency or processing overhead, may influence overall performance. This research also measured the TGS. The TGS represents the total size of data that transactions process successfully per second. This research measures the TGS for the size of the data.

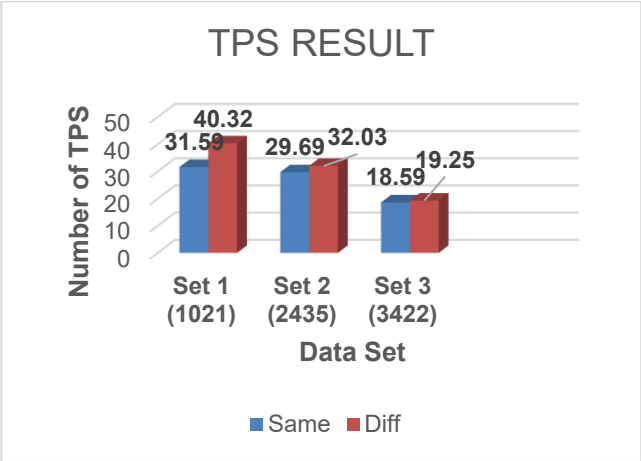

Fig. 8. The TPS result.

Regarding data in Table II, the TGS results were measured and compared with the execution at different and the same locations. Fig. 9 states the result of a number of TGS. Based on the TGS results, it showcases that set 1 has successfully transacted 1,064,266.14 KB per second, while set 2 is 992,814.83 KB per second, and lastly, set 3 successfully transacted 624,364.33 KB per second for a different location. The same location for TGS result is 1,358,194.53 KB per second, 1071194.95 KB per second, and 655,649.11 KB per second.

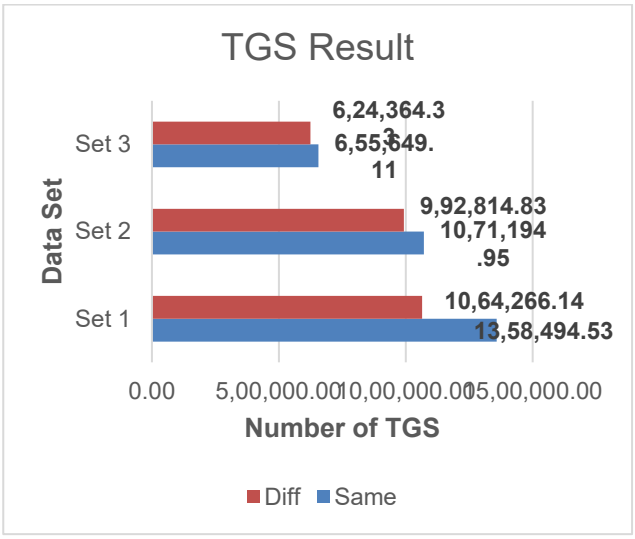

Fig. 9. Number of throughputs per second.

B. Number of Nodes

The number of nodes can significantly impact the performance of a consensus algorithm. In this study, the AuRa consensus algorithm was executed to generate digital certificates under three different scenarios using the same dataset.

In the first scenario, three nodes were deployed; in the second, four nodes; and in the third scenario, five nodes were implemented. To ensure a controlled testing environment, all three configurations were evaluated using uniform network bandwidth, identical server capacities, and a standardized dataset volume. This setup was designed to evaluate the effect of node scalability on system performance.

The TPS results presented in Fig. 10 show that the first scenario, which deployed three nodes, achieved a throughput of 53.32 transactions per second, outperforming the second scenario with four nodes, which recorded a TPS of 50.92. In the third scenario, where five nodes were implemented, the TPS increased to 52.25, indicating a partial recovery of performance relative to the four-node setup. These observations demonstrate that node count exhibits a non-linear influence on system throughput, suggesting that merely increasing the number of authority nodes does not guarantee improved performance within the AuRa consensus environment.

However, this experiment is limited to evaluating only three configurations, including three, four, and five nodes, under a three-dataset, which restricts the depth of performance interpretation. The analysis does not yet account for several critical factors that could further affect TPS behaviour.

Specifically, the study does not evaluate system performance when an authority node fails or behaves maliciously, which is essential for understanding AuRa's robustness in adversarial conditions. Similarly, the experiment does not explore the impact of increased block time or varying network latency, both of which can significantly influence throughput in distributed blockchain deployments.

Furthermore, the scalability of the system beyond five nodes, especially in multi-institution or multi-university settings, remains unexamined. Addressing these dimensions in future work would provide a more comprehensive assessment of AuRa's performance characteristics and operational resilience in large-scale, heterogeneous environments.

The TPS results presented in Fig. 10 indicate that the performance in the first scenario, which utilized three nodes, achieved a throughput of 53.32 transactions per second. This is higher compared to the second scenario, which involved four nodes and recorded a TPS of 50.92. Interestingly, in the third scenario, where five nodes were implemented, the TPS increased to 52.25, demonstrating improved performance relative to the second scenario. These findings suggest that the number of nodes has a non-linear effect on system performance, where increasing the number of nodes does not necessarily result in higher throughput.

Based on the analysis presented in Fig. 10, it can be concluded that the number of nodes has a measurable impact on the performance of the AuRa consensus algorithm. Inconsistencies in the empirical results are largely driven by synchronization delays inherent in multi-node deployments. A critical factor is the temporal overhead required for all nodes to achieve state synchronization, which inherently influences overall throughput and latency.

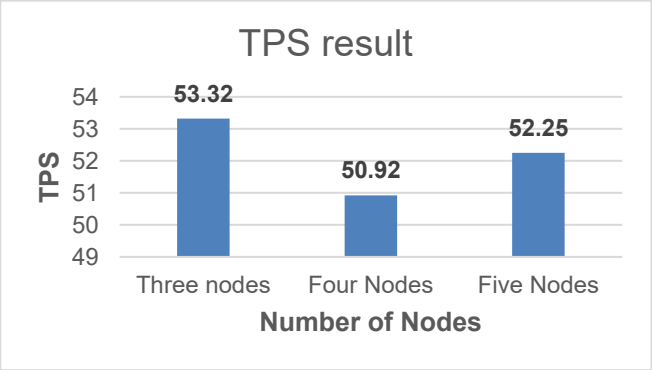

Fig. 10. Comparison of TPS results based on the number of nodes.

C. Security Analysis

In the context of security, low TPS performance in the AuRa consensus mechanism introduces several potential security vulnerabilities that merit careful consideration. When each transaction requires extended processing time, the prolonged block finalisation window increases exposure to adversarial interference. Specifically, a malicious leader may exploit this delay to insert falsified data, manipulate timestamps, or execute equivocation attacks before finality is reached.

Such conditions weaken the integrity guarantees of the protocol and elevate the probability of inaccurate or unauthorized certificate records being committed to the blockchain. In extreme cases, these delays may disrupt the synchronous assumptions underlying AuRa, thereby increasing the risk of block production inconsistencies.

Similarly, TGS performance indicates inefficiencies in handling data throughput, which may further exacerbate security risks within the certificate generation workflow. When the system struggles to transfer or commit data at a stable rate, node desynchronization becomes more likely, particularly in networks with multiple validators. Such desynchronization can create opportunities for adversaries to exploit temporary inconsistencies between nodes, potentially resulting in chain forks or interrupted block propagation.

If a malicious node initiates an attack during these vulnerable intervals, such as injecting altered certificate information, the resulting chain divergence could compromise system reliability and undermine trust in the blockchain-based certificate infrastructure. Therefore, maintaining adequate TPS and TGS performance is not merely an efficiency requirement but a critical component of ensuring the security and correctness of the AuRa-driven system

V. Discussion and Limitation

This study implemented AuRa to create digital certificates. The implementation focused on the measurement of TPS and TGS in only two situations: the same location and different locations of nodes. However, the study did not record the average gas usage per transaction during certificate creation.

A. Consensus Algorithm

This research implements only one consensus algorithm, which causes a limitation in evaluating and measuring the TPS and TGS results. If more than one consensus algorithm had been implemented in this study, comparisons could provide a stronger and more comprehensive evaluation.

B. Server

This study implements the AuRa algorithm to generate certificates on a private server. Thus, at least three servers are needed to establish the nodes. The private server implementation might incur extra costs and resources due to distinct configurations and maintenance requirements.

VI. Conclusion and Future Work

This work contributes a real-world deployment of a blockchain-enabled digital certificate management system built on a private Ethereum network using the AuRa consensus algorithm. Moving beyond conventional blockchain implementations, this study offers empirical evidence of its practical viability in strengthening certificate integrity and resolving verification-related challenges. Crucially, the core technical contribution lies in optimizing block generation and transaction throughput within a private network architecture, thereby effectively mitigating certificate forgery while addressing the performance and scalability bottlenecks inherent in standard consensus mechanisms.

Blockchain allows read and write access to the data. To ensure the security of blockchain, it is supported by the AuRa consensus algorithm. The AuRa consensus algorithm is able to create a secure environment within the transaction process until the transaction is committed to the blockchain. This research also implemented the AuRa consensus algorithm, Ethereum, and smart contracts, which have been embedded with Python coding.

The implementation of the certificate data is converted into JSON format. Then, execute the experiment, including three sets of data on a private server. However, the evaluation reveals inconsistent results on TPS and TGS. Regarding these results, geolocation, size of data, and number of nodes affected the TPS and TGS results.

Moreover, enhanced performance in TPS and TGS strengthens the system's capability to detect and prevent the use of fake or invalid certificates. By enabling real-time verification and rapid detection of discrepancies within the blockchain-based certificate infrastructure, the system reduces the risk of certificate forgery or duplication. This efficiency allows for continuous monitoring and immediate rejection of suspicious or tampered credentials, thereby upholding data integrity and trust in the digital certification ecosystem. Ultimately, the improved performance ensures not only operational effectiveness but also reinforces the security and reliability of the entire certificate management process. Regrettably, a direct comparative analysis with prior studies within the same domain could not be established. This limitation arises because existing literature fails to explicitly detail its empirical results and the specific performance metrics employed.

For future work, this study attempts to generate certificates under various conditions, including different numbers of nodes. The results are then compared and analyzed to determine whether the number of nodes has a significantly affects TPS and TGS.

ACKNOWLEDGMENT

We express sincere gratitude to the Center of Research and Innovation Management (CREIM) at UniSZA for their invaluable financial support towards the publication and the Research Group of Intelligent Data (ID). Our heartfelt appreciation also goes to all team members from UniSZA, including Pn. Robiah Arifin, the PhD candidate for the technical configuration of the project, and Dr. Wan Aezwani Wan Abu Bakar, and Assoc. Prof. Dr. Mohamad Afendee as her Supervisor and Co-Supervisor for the conceptual and technical proofreading, as well as the UMT member, Assoc Prof. Ts. Dr. Mustafa Man, for the network linkages and the reviewer's suggestions. Also, we are grateful for the international contributions and collaboration provided by Dr. Evizal Abdul Kadir, a research fellow from Universitas Islam Riau, Indonesia.

REFERENCES

[1] Carmichael, J. J., & Eaton, S. E. (2023). Fake degrees and fraudulent credentials in higher education: Conclusions and future directions. In Fake Degrees and Fraudulent Credentials in Higher Education (pp. 269-285). Cham: Springer International Publishing.

[2] Arifin, R., Wan Abu Bakar, W. A., Man, M., & Kadir, E. A. (2024). Performance Evaluation of the AuRa Consensus Algorithm for Digital Certificate Processes on the Ethereum Blockchain. International Journal of Advanced Computer Science & Applications, 15(11).

[3] Huynh-The, T., Gadekallu, T. R., Wang, W., Yenduri, G., Ranaweera, P., Pham, Q. V., ... & Liyanage, M. (2023). Blockchain for the metaverse: A review. Future Generation Computer Systems, 143, 401-419.

[4] Thakare, N., Narad, S., & Surysvanshi, Y. (2024, December). Fake certificate detection by using blockchain. In AIP Conference Proceedings (Vol. 3188, No. 1). AIP Publishing.

[5] Aini, Q., Harahap, E. P., Santoso, N. P. L., Sari, S. N., & Sunarya, P. A. (2023). Blockchain based certificate verification system management. APTISI Transactions on Management, 7(3), 191-200.

[6] Tumati, T. V., Tian, Y., & Jiang, X. (2024, January). A soulbound token certificate verification system (sbtcert): Design and implementation. In 2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0345-0350). IEEE.

[7] Pu, Shuyi, and Jasmine Siu Lee Lam. "The benefits of blockchain for digital certificates: A multiple case study analysis." Technology in Society 72 (2023).

[8] Rustemi, A., Dalipi, F., Atanasovski, V., & Risteski, A. (2023). A systematic literature review on blockchain-based systems for academic certificate verification. IEEE Access.

[9] Fang, J., Feng, T., Guo, X., Ma, R., & Lu, Y. (2024). Blockchain-cloud privacy-enhanced distributed industrial data trading based on verifiable credentials. Journal of Cloud Computing, 13(1), 30.

[10] Feng, X., Wang, L., Bai, X., & Yang, P. (2024). Distributed identity management mechanism based on improved block-chain certificateless encryption algorithm. Physical Communication, 65, 102341.

[11] Liu, H., Ming, Y., Wang, C., Zhao, Y., Zhang, S., & Lu, R. (2024). Blockchain-assisted verifiable certificate-based searchable encryption against untrusted cloud server for Industrial Internet of Things. Future Generation Computer Systems, 153, 97-112.

[12] Mai, C. K., Iqbal, M. S., Rohith, A., Suchetan, T. C. K., & Shinde, P. C. (2024). Applicant Credentials Tracker for Employment Using Blockchain Technology. International Journal of Intelligent Systems and Applications in Engineering, 12(3s), 320-327.

[13] Samarai B.A. (2024).Use of Blockchain Technology in Educational Field.International Journal on Technical and Physical Problems of Engineering,15(4),140-151

[14] Rani, P., Sachan, R. K., & Kukreja, S. (2024). Educert-chain: a secure and notarized educational certificate authentication and verification system using permissioned blockchain. Cluster Computing, 27(7), 10169-10196.

[15] Howell, A., Saber, T., & Bendechache, M. (2023). Measuring node decentralisation in blockchain peer to peer networks. Blockchain: Research and Applications, 4(1), 100109.

[16] Villegas-Ch, W., Gutierrez, R., Govea, J., & García-Ortiz, J. (2025). Integrated AI, IoT, and blockchain for enhancing security and traceability in perishable logistics. Emerg. Sci. J, 9, 2471-2496.

[17] Tripathi, G., Ahad, M. A., & Casalino, G. (2023). A comprehensive review of blockchain technology: Underlying principles and historical background with future challenges. Decision Analytics Journal, 9, 100344.

[18] Pancari, S., Rashid, A., Zheng, J., Patel, S., Wang, Y., & Fu, J. (2023). A systematic comparison between the ethereum and hyperledger fabric blockchain platforms for attribute-based access control in smart home IoT environments. Sensors, 23(16), 7046.

[19] Mai, C. K., Iqbal, M. S., Rohith, A., Suchetan, T. C. K., & Shinde, P. C. (2024). Applicant Credentials Tracker for Employment Using Blockchain Technology. International Journal of Intelligent Systems and Applications in Engineering, 12(3s), 320-327.

[20] Arifin, R. B., Bakar, W. A. W. A., Man, M. B., & Kumar, B. P. (2024). Tackling counterfeit certificate problems with blockchain technology: a review. International Journal of Advanced Technology and Engineering Exploration, 11(119), 1392.

[21] Arifin, R., Bakar, W. A. W. A., Man, M., & Kadir, E. A. (2025). Comparative Performance Analysis of Original AuRa and Improved AuRa Consensus Algorithms in Chain Hammer Digital Certificate Simulation. International Journal of Advanced Computer Science & Applications, 16(10).

[22] Islam, M. M., & In, H. P. (2023). Decentralized Global Copyright System Based on Consortium Blockchain with Proof of Authority. IEEE Access. https://doi.org/10.1109/SCC55611.2022.00054

[23] Hu, Y., Tian, G., Jiang, A., Liu, S., Wei, J., Wang, J., & Tan, S. (2023). A practical heartbeat-based defense scheme against cloning attacks in PoA blockchain. Computer Standards & Interfaces, 83, 103656.

[24] Sarfaraz, A., Chakrabortty, R. K., & Essam, D. L. (2023). The implications of blockchain-coordinated information sharing within a supply chain: A simulation study. Blockchain: Research and Applications, 4(1), 100110.

[25] Zhang, Xinrui, Rujia Li, Qin Wang, Qi Wang, and Sisi Duan. "Time-manipulation attack: Breaking fairness against proof of authority Aura." In Proceedings of the ACM Web Conference 2023, pp. 2076-2086. 2023. https://doi.org/10.1145/3543507.3583252

[26] Wu, G., Wang, H., Lai, X., Wang, M., He, D., & Chan, S. (2024). A comprehensive survey of smart contract security: State of the art and research directions. Journal of Network and Computer Applications, 226, 103882.

[27] Zaw, T. O. K., Anbananthen, K. S. M., Muthaiyah, M. S., Balasubramaniam, B., Kannan, R., Kalid, K. S., ... & Mohammad, S. (2025). A Systematic TOGAF-Driven Framework for Blockchain-Based Food Traceability with Access Control Lists. HighTech and Innovation Journal, 6(2), 461-475.