Blockchain-based Secure D2D localisation with adaptive precision

Gewu BU
Clermont Ferrand University
LIMOS
France
gewu.bu@uca.fr

Bilel Zaghdoudi
Sorbonne University
LIP6
Paris, France
bilel.zaghdoudi@lip6.fr

Maria Potop-Butucaru
Sorbonne University
LIP6
Paris, France
maria.potop-butucaru@lip6.fr

Serge Fdida
Sorbonne University
LIP6
Paris, France
serge.fdida@lip6.fr

***Abstract*—In this paper we propose a secure best effort methodology for providing localisation information to devices in a heterogenous network where devices do not have access to GPS-like technology or heavy cryptographic infrastructure. Each device will compute its localisation with the highest possible accuracy based solely on the data provided by its neighboring anchors. The security of the localisation is guarantied by registering the localisation information on a distributed ledger via smart contracts. We prove the security of our solution under the adaptive chosen message attacks model. We furthermore evaluate the effectiveness of our solution by measuring the average register location time, failed requests, and total execution time using as DLT case study Hyperledger Besu with QBFT consensus.**

***Index Terms*—Decentralized localisation,D2D,Blockchain, Decentralized Storage**

I. Introduction

Localization is at the core of various applications in wireless networks including routing, clustering, path planning and many others. In maritim applications for instance devices communicate their location via AIS (automatic identification system) subject to a broad range of cyberattacks . Therefore, recent research has explored the integration of blockchain technology into localization systems to address security vulnerabilities and enhance reliability across diverse network environments. In [1], a framework is proposed to mitigate threats in Wireless Sensor Networks (WSNs) operating in hostile settings. By computing dynamic trust values based on metrics like information integrity, reputation, and energy levels, and securely storing these values on a decentralized blockchain, the system ensures transparency and immutability. A Proof-of-Stake (PoS) consensus mechanism selects high-trust beacon nodes for mining and secure localization, resulting in improved accuracy, reduced false positive/negative rates, and enhanced malicious activity detection. Similarly, [2] focuses on vehicular localization in Internet-of-Vehicles (IoV) networks vulnerable to malicious attacks. This work integrates a lightweight blockchain framework with Time Difference of Arrival (TDOA) localization and employs trust evaluation based on node energy consumption and target location estimates. Analytical and simulation studies reveal the scheme's robustness, achieving high accuracy and resilience even under significant malicious activity. Addressing WSN-specific challenges, [3] introduces a range-free localization approach combining trust metrics such as energy levels, reputation, and neighbor data with blockchain's immutability. A PoS mechanism ensures the selection of trustworthy nodes, achieving notable improvements in localization precision and energy efficiency. For IoT applications, [4] presents BlockLoc, a blockchain-based localization system designed to counter threats like Sybil attacks, eavesdropping, and message forging. By securely storing location data and verifying node positions using techniques such as RSSI and DV-Hop, BlockLoc enhances robustness and scalability in dynamic IoT environments. In the domain of smart homes, [5] proposes a secure monitoring framework leveraging private blockchain technology. The system employs RSSI-based trilateration with Kalman filtering to localize devices and detect malicious activity while reducing computational overhead through optimized consensus mechanisms. Simulation results demonstrate increased localization accuracy and security for smart home networks. Further enhancing WSN localization, [6] incorporates behavioral and data trust metrics into a blockchain framework. Trust values are decentralized and immutable, with high-trust nodes selected for localization tasks. The approach achieves significant gains in energy efficiency, accuracy, and attack resilience. A novel Proof-of-Location (PoL) mechanism is introduced in [7] for decentralized location verification using blockchain. This system creates tamper-resistant certificates of user positions, leveraging short-range communication and asymmetric cryptography to combat spoofing and replay attacks. Results highlight its robustness and scalability for applications such as smart cities and IoT ecosystems. Integrating federated learning with blockchain, [8] presents a secure IoT-based WSN localization framework. Through hierarchical trust evaluation and privacy-preserving machine learning models, the system achieves near-perfect malicious node detection and localization accuracy, making it suitable for large-scale deployments. The framework in [9] supports low-power IoT devices by combining blockchain with RF timestamping and TDOA techniques, eliminating reliance on GPS. The Helium network demonstrates efficient geolocation services with reduced energy consumption, ideal for applications like asset tracking and smart cities. In UAV localization, [10] employs blockchain with a Distance Bounding Protocol to ensure secure and private location authentication. Features such as zero-knowledge proofs and tamper-proof verification enhance the system's scalability and security in dynamic UAV environments. Lastly, [11] introduces the Internet of Entities (IoE), a decentralized framework leveraging existing wireless infrastructure to provide scalable, privacy-preserving localization services. This system records location data in tamper-resistant ledgers while minimizing costs, offering robust solutions for e-health, smart cities, and mobility applications.

These contributions collectively underscore the potential of blockchain technology to revolutionize secure localization by addressing vulnerabilities, enhancing accuracy, and ensuring scalability across various domains.

However, while blockchain-based localization systems offer enhanced transparency and security, several challenges and limitations persist. Many frameworks, such as those leveraging trust evaluation or federated learning techniques, are computationally demanding and introduce significant delays, which can be problematic for applications requiring quick responses, such as UAV operations or dynamic IoT networks [1], [2], [8]. Additionally, while decentralization strengthens resilience, it also increases communication overhead and energy consumption, posing challenges for resource-constrained devices within Wireless Sensor Networks (WSNs) [3], [5]. Scalability further becomes an issue in environments like large-scale IoT or vehicular systems, especially when consensus mechanisms such as Proof-of-Work or Proof-of-Stake are involved [4], [9]. Moreover, specific vulnerabilities, such as Sybil attacks or the falsification of location proofs, remain inadequately addressed in some frameworks, potentially undermining the reliability of the localization process [6], [7], [10]. These drawbacks underscore the need for optimization and the adoption of lighter, more efficient security measures to fully realize the potential of blockchain-powered localization solutions.

In this paper, we propose a blockchain-based two layer hierarchical architecture consisting of an upper layer blockchain and a lower layer decentralized D2D network. Proposed system leverages the blockchain layer to provide reliable and flexible localisation service to the D2D network. Our solution outperforms existing works in the following aspects :

First, our solution offers superior scalability compared to non-hierarchical solutions [1], [2], [3], [6], [10], [11]. In non-hierarchical architectures, nodes to be located also function as blockchain participants. As a result, all nodes must perform complex consensus algorithms to maintain the blockchain, which inherently constrains the scalability of these solutions. Furthermore, the communication overhead for blockchain maintenance increases the latency of the localisation process. High latency becomes even more problematic in high-mobility scenarios, such as vehicular or aerial networks mentioned in [2], [10]. In contrast, our hierarchical system allows the lower layer D2D network to function without maintaining the blockchain, which enhances the scalability of the lower layer D2D network.

Second, similar to [4], [7], our solution leverages blockchain to provide localisation services for lower layer decentralized networks. However, the mentioned solutions rely on a fixed localisation computation process, which imposes strict requirements on network environment, such as the network density. For example, these solutions require the locations of at least three anchor neighbors to perform the localisation computation. In contrast, our solution offers flexible and adaptive localisation computation. Proposed localisation service provides high-accuracy localisation for nodes in high-density network environments, while delivering approximate localisation for nodes at the edge of the network. This improves the potential and flexibility for nodes to obtain their location in time.

Finally, in terms of security, we demonstrate that the entire workflow of the proposed localisation service is secure against network-level adversaries and attacks. For other adversaries attempt to insert anomalous data and disrupt the accuracy of localisation computation, recent advancements in machine learning have led to well-developed solutions for detecting and filtering such anomalies [15], [16]. Our solution, therefore, provides modular interfaces to adopt these methods into our localization workflow, preventing the insertion of malicious data and ensuring the accuracy of the localisation computation process.

II. Problem statement

We consider a system operating in an asynchronous communication environment. It features a two-layer hierarchical structure. The upper layer consists of a distributed network maintaining a distributed ledger, and a distributed storage system. The lower layer is a decentralized and dynamic D2D network composed of independent D2D nodes. The two layers communicate via dedicated gateway nodes. Gateway nodes are direct participants of upper layer network and positioned at the edge between the two layers. Gateway nodes communicate with other participants in the upper layer network and interact with the distributed storage system through reliable channels and interfaces. While, gateway nodes communicate with the D2D nodes of the lower layer through lossy wireless channels.

Gateway nodes are considered to have their accurate GPS locations and sufficient computational power for complex localisation computation, while D2D nodes are dynamic and mobile nodes, lacking their GPS locations, and may not possess the capability for complex localisation computation.

We identify two fundamental techniques well used for wireless source localisation as the building blocks of our proposed solution : *Time Difference of Arrival (TDoA)* and *Time of Arrival (ToA)* [12], [13], [14].

In our context, only gateway nodes can perform *TDoA* computation, as time synchronization among computing nodes is required. During the *TDoA* process, multiple gateway nodes receiving messages from the same target node calculate its location by sharing information through the distributed ledger. While, *ToA* computation can be performed either by gateway nodes, or D2D nodes with sufficient computational power. During the *ToA* process, the target node calculates its location locally by collecting messages from nearby nodes.

The problem we aim to address in this context is to provide reliable and adaptive localisation services to D2D nodes in the lower layer D2D network through interaction with the upper layer distributed ledger and distributed storage system. Our solution allows the localisation service to choose, based on the current network environment of the target D2D node, the most adapted localisation method for computing the location of target D2D node.

III. Blockchain-based adaptive localisation

In this section, we present the proposed localisation service based on adaptive localisation.

A. Adaptive localisation methodology

This algorithm identifies the highest localization accuracy a node can achieve based on data from its neighboring anchors. By sequentially evaluating conditions from the most precise level (`L1`) to the least accurate (`L5`), it ensures that each node can dynamically adapt its localization strategy. This adaptability enhances the overall performance and resilience of the system, enabling robust operation in dynamic or resource-constrained environments. By leveraging varying levels of anchor data, the methodology maintains reliable localization capabilities, even in scenarios where synchronization, responses, or distance measurements are limited, ensuring optimal efficiency and flexibility in the network.

Algorithm 1 Determine Localization Precision Level

```
Require: $node\_data$: A dictionary containing:
    $synchronized\_anchors$: List of synchronized anchor IDs
    $anchor\_responses$: List of anchor IDs responding to requests
    $anchor\_distances$: Dictionary mapping anchor IDs to distances
    $nearby\_anchors$: List of dictionaries with keys:
        $id$: Anchor ID
        $accuracy\_level$: Accuracy level of the anchor
Ensure: The highest achievable accuracy level: $L1$, $L2$, $L3$, $L4$, $L5$, or None
  if $|synchronized\_anchors| \geq 3$ then
      return $L1$
  if $|anchor\_responses| \geq 3$ then
      return $L2$
  if any $anchor\_id \in nearby\_anchors$ for $anchor\_id \in anchor\_distances$ then
      return $L3$
  if any $anchor.accuracy\_level \in \{L1, L2\}$ for $anchor \in nearby\_anchors$ then
      return $L4$
  if any $anchor.accuracy\_level \in \{L3, L4\}$ for $anchor \in nearby\_anchors$ then
      return $L5$
  return None
```

B. Adaptive Localisation Service Workflow

Through the proposed localisation service, any D2D node in the system can request the computation and registration of its location.

The localisation service operates in two phases, involving three processes (see Figure 1). The first phase consists of the *TDoA Location Computation* process. In this phase, the localisation service aims to compute and register an *L1* location for the target D2D node. If successful, the localisation process is complete. Otherwise, the localisation service proceeds to the second phase.

The second phase includes two alternative processes: *Delegated ToA Location Computation* and *Local ToA Location Computation*. In this phase, the service aims to compute and register an *L2* to *L5* location for the target D2D node. If successful, the localisation process is complete. Otherwise, the service is deemed a failure.

Comparing with *Local ToA Location Computation*, *Delegated ToA Location Computation* does not require the node to perform local location computation. Instead, the D2D node must send necessary information for location computation to the gateway node. This reduces the local computational load on the D2D node but, increases the overall communication overhead of the localisation service. The preference between the two processes can be predefined based on the application scenario.

Figure 1 shows a detailed workflow. Steps with prefix *TDoA*, *Dlg* and *Local* are dedicated steps for *TDoA Location Computation*, *Delegated ToA Location Computation* and *Local ToA Location Computation*, respectively.

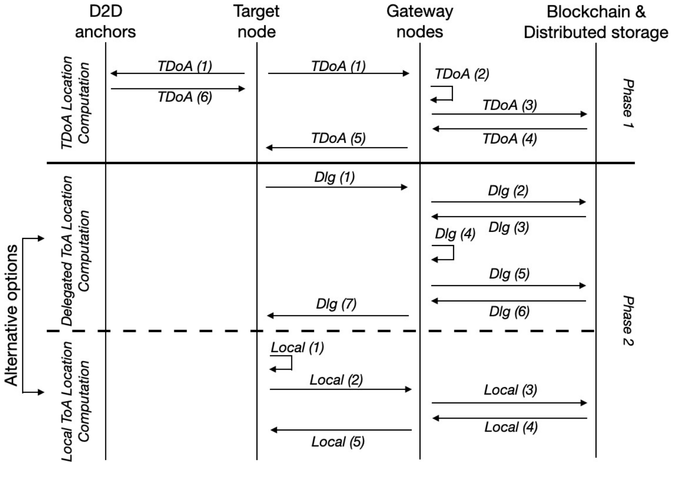

Fig. 1. Localisation Service Workflow

Let's begin with the first phase.

TDoA(1): The D2D node to be located broadcasts a *TDoA Computation Request*, asking to calculate and register its location. This request can be received by any gateway and D2D node.

TDoA(2): If at least three gateway nodes receive the *TDoA Computation Request* within a certain period of time, a receiving gateway node will be able to calculate an *L1* location for the target D2D node and execute steps *TDoA(3)* to *TDoA(5)*. Otherwise, the *TDoA Location Computation* process fails. In this case, receiving gateway nodes skip steps *TDoA(3)* and *TDoA(4)* and directly send a *ToA Response* back to the target D2D node in *TDoA(5)*.

TDoA(3): If the *L1* location is successfully calculated in *TDoA(2)* by a receiving gateway node, the gateway interacts with the distributed ledger and distributed storage system to register this location for the target D2D node.

TDoA(4): The distributed ledger and distributed storage system send the registration result back to the gateway.

TDoA(5): If the registration succeeds, the gateway notifies the target D2D node of its *L1* location and confirms the registration via a *TDoA Computation Response*. Otherwise, the *TDoA Location Computation* process fails, the gateway sends a *ToA Response* to the target D2D node instead.

TDoA(6): During the *TDoA Location Computation* process, any other D2D node that receives the *TDoA Computation Request* will respond to the target D2D node with a *ToA Response*.

If the *TDoA Location Computation* fails, the localisation service proceed the second phase, by beginning *Delegated ToA Location Computation* or *Local ToA Location Computation*.

In *Delegated ToA Location Computation*:

Dlg(1): The target D2D node sends a *Delegated Computation Request*, consisting of all *ToA Responses* received during *TDoA(5)* and *TDoA(6)*, to a gateway node.

Dlg(2): Upon receiving the *Delegated Computation Request*, the gateway node interacts with the distributed ledger and distributed storage system to verify the validity of the *ToA Responses* submitted in the *Delegated Computation Request*.

Dlg(3): The distributed ledger and distributed storage system return a subset of valid *ToA Responses* to the gateway node.

Dlg(4): If the subset contains at least three *ToA Responses*, the gateway node computes an *L2* location for the target D2D node. If only one or two valid *ToA Responses* are available, the gateway assigns an *L3* to *L5* location according to one of these valid *ToA Responses*. If the subset is empty, the *Delegated ToA Location Computation* process fails, and steps *Dlg(5)* and *Dlg(6)* are skipped. The gateway notifies the target D2D node of the failure in *Dlg(7)*.

Dlg(5): If an *L2* to *L5* location is successfully computed during *Dlg(4)*, the gateway node interacts with the distributed ledger and distributed storage system to register this location for the target D2D node.

Dlg(6): The distributed ledger and distributed storage system send the registration result back to the gateway.

Dlg(7): If the registration succeeds, the gateway node notifies the target D2D node of its location and confirms the registration via a *Delegated Computation Response*. Otherwise, it indicates the failure of localisation service in the response.

In *Local ToA Location Computation* :

Local(1): The target D2D node collects all the *ToA Responses* received during *TDoA(5)* and *TDoA(6)*. The D2D node try to compute an *L2* to *L5* location locally as described in *Dlg(4)*. If no *ToA Responses* is received within a certain period of time, the *Local ToA Location Computation* process and the localisation service end with failure.

Local(2): If an *L2* to *L5* location is computed, the target D2D node sends, to a gateway node, both the location and the identities of the nodes whose *ToA Responses* were used for the computation. This is done through a *Location Registration Request*.

Local(3): The receiving gateway then interacts with the distributed ledger and distributed storage system. It verifies whether the locations of the nodes, whose identities are submitted in the *Location Registration Request*, have been registered. If all these locations are verified, the distributed ledger and distributed storage system register the received location of the target D2D node.

Local(4): The distributed ledger and distributed storage system send the registration result back to the gateway.

Local(5): If the registration succeeds, the gateway node notifies the target D2D node of its location and confirms the registration via a *Location Registration Response*. Otherwise, it indicates the failure of localisation service in the response.

C. Location proof flows

The proposed proof of location uses registered GPS gateway locations stored and secured on the Blockchain. An example of the flow to collect data of a device position is as follows :

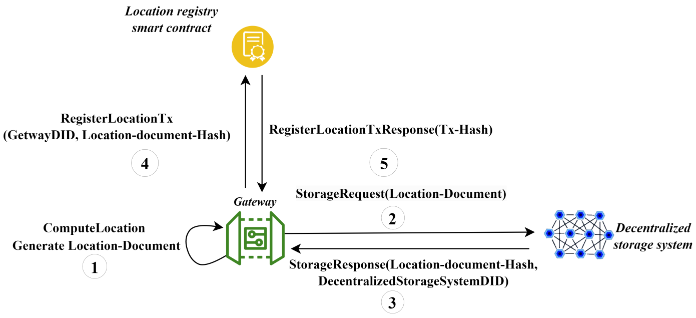
Location registry
smart contract
RegisterLocationTx
(GetwayDID, Location-document-Hash)
4
RegisterLocationTxResponse(Tx-Hash)
5
Gateway
StorageRequest(Location-Document)
2
ComputeLocation
Generate Location-Document
1
Decentralized
storage system
StorageResponse(Location-document-Hash,
DecentralizedStorageSystemDID)
3

Fig. 2. Register location

The diagram 2 illustrates a five-step process involving a smart contract and the interaction between a gateway and a decentralized storage system. The process begins with step 1, where the gateway computes a location and generates a location document. In step 2, this location document is sent to a decentralized storage system as a storage request. The decentralized storage system responds with a storage response, which includes the location document hash and the decentralized storage system's DID (Decentralized Identifier), as indicated in step 3. The gateway then initiates a register location transaction with the Location Registry smart contract, including the gateway/the located device DID and the location document hash, in step 4. Finally, the smart contract processes the transaction and returns a response containing the transaction hash back to the gateway in step 5, completing the interaction loop.

IV. Security analysis

In this section, we present the security analysis of the proposed localisation service, starting with the adversary model and the security assumptions.

A. Adversary Model

When discussing attacks on localisation in wireless networks, two primary categories of adversaries can be identified. The first category targets the system itself. These adversaries launch attacks at the communication layer, aiming to compromise the system's availability, confidentiality, or integrity. The second category focuses on disrupting the accuracy of the localisation computations. These adversaries aim to inject anomalous data into the localisation computation process by providing falsified timestamps or location information. Since the common way to inject anomalous data is by submitting falsified local information of the adversary, it's therefore challenging for others to detect such behavior using traditional cryptographic methods.

In recent research, developed solutions have been proposed to enhance resistance against the anomalous data injection, by leveraging machine learning techniques to detect anomalous data within the system [15], [16]. In this paper, we assume that our system employs appropriate measures to resist anomalous data injection targeting the localisation computations. These measures can be integrated directly into the workflow of section III-B, where localisation computation is required. More specifically, *TDoA (2)* in *TDoA Location Computation*, *Dlg(4)* in *Delegated ToA Location Computation*, and *Local(1)* in *Local ToA Location Computation*. Consequently, our analysis will focus exclusively on the security in the context of the first category of adversaries.

We consider adversaries capable of initialing *Adaptive Chosen Message Attacks (A-CMA)* [17]. In this scenario, an adversary node can not only choose a set of messages and obtain their corresponding ciphertexts or plaintexts, but also select subsequent messages based on the outcomes of the previously chosen messages.

B. System model and security analysis

We consider that our system is an asynchronous and decentralized network, consisting of a large, finite, yet unbounded set of nodes from the networks of two layers. To focus on the security analysis of the proposed localisation service, we assume that the wireless communication channels between D2D nodes and gateway nodes are designed to prevent data loss through re-transmission and to protect against *Physical* and *MAC* layer attacks, such as *Flooding Attacks*, *Replay Attacks*, and *DoS Attacks*. We exclusively outline the security assumptions for the localisation service on the *Network* and *Application* layers.

1) The connected distributed ledger and distributed storage system satisfies the *ACID* properties - *Atomicity*, *Consistency*, *Isolation* and *Durability*.
2) All Gateways are connected to the distributed storage system via dedicated interfaces. Through the interfaces, the communication between gateways and distributed storage system satisfies the *CIA* properties - *Confidentiality*, *Integrity*, and *Availability*.
3) Both D2D and Gateway nodes have their own IDs, bound to unique public-private key pairs.
4) In the context of *A-CMA*, the public-private key encryption scheme is considered secure, and the digital signature satisfies *Existential Unforgeability* [18]. This ensures that the adversary cannot successfully crack the encryption or forge the signature of any message.
5) Gateway nodes are honest.

C. Security Properties and Proofs

We conclude that our localisation service satisfies the following properties under *A-CMA*:

P1 Any computation or registration request from a D2D node will eventually be accurately received and processed by a gateway node. And that D2D node will eventually receive a correct response corresponding to its request.

P2 Registered locations cannot be tampered with.

P1 is guaranteed by Lemma 1 and Lemma 2. P2 is guaranteed by Lemma 3 The proof of this property consists of the following Lemmas.

Lemma 1. *Any request or response in the system cannot be tampered with or falsified.*

Proof. All requests and responses are required to be digitally signed to ensure their integrity. Therefore, according to *Hypothesis 4* and *Hypothesis 5*, requests or responses in the system cannot be tampered with or falsified. □

Lemma 2. *No adversary node can disrupt the message flow within the system, nor disable the process flow of localisation service.*

Proof. In the system, any computation or registration request, as well as any response, will eventually be delivered to the destination. Even if messages are lost due to network congestion or adversary node behavior, the underlying retransmission mechanism will be triggered. Therefore, all messages will eventually be successfully delivered and received.

As described in Section III-B, all three localisation computation processes have a maximum waiting time. If the waiting time is exceeded, the process fails. The workflow will then proceed to the next step or terminate the computation. Therefore, no behavior of an adversary node can cause the localisation service to wait indefinitely or become disabled. □

Lemma 3. *No D2D node can interact with the distributed ledger or distributed storage system without bypassing a gateway node.*

Proof. As described in Section III-B, only gateway nodes have the access to interact with the distributed ledger and the interface of the distributed storage system. Meanwhile, according to Lemma 1, no D2D node can impersonate as a gateway node. Therefore, D2D nodes cannot directly interact with the distributed ledger or the distributed storage system, nor tamper with the data in it. □

V. Experimental analysis

In this section, we present and analyze the results of the location registration process for the `sm_reg` smart contract. Three key performance metrics are discussed: Average Register Location Time vs. Number of Clients, Total Successful vs. Failed Transactions, and Total Execution Time vs. Number of Clients. Each chart provides insights into the scalability and resilience of the smart contract across different client configurations.

We utilized the Hyperledger Besu client version 24.5.1, configured with the QBFT consensus algorithm. The system was set to operate with a block period of 1 second and deployed on a virtual machine running Linux Ubuntu 22.04, with specifications of 8 vCPUs, 80 GB of storage, and 16 GB of memory.

The implemented experimental study includes two main components: a smart contract written in Solidity and a JavaScript script using the web3 library to interact with the deployed blockchain. The Solidity contract `sm_reg` provides functions to register and retrieve device locations, storing them securely on the blockchain. It defines methods like registerLocation and getRegisteredLocations, allowing nodes to store and query location data, ensuring the immutability and integrity of this data. In this study, we implemented a basic version of the location storage system that registers locations directly on the blockchain without integrating a decentralized storage network. This simplified version enables us to focus solely on measuring the blockchain's performance.

The JavaScript script implements a worker-based system to test the scalability of the blockchain. Using the cluster module, the script forks multiple worker processes, each responsible for sending 100 transactions to the blockchain to register locations. It logs each transaction's result and response time, aiming to measure the time taken to store device locations under various load conditions. By omitting the decentralized storage network, this experiment strictly evaluates the system's performance when handling a large number of direct blockchain transactions, providing insight into the blockchain's scalability in its current configuration.

The implementation, deployment, and testing of the `sm_reg` smart contract were conducted on an HP EliteBook 850 G8 Notebook laptop. This machine was configured with Windows 11 Professional Version 23H2, an 11th Gen Intel(R) Core(TM) i7-1185G7 processor running at 3.00GHz, and 64 GB of memory. The APIs were developed using JavaScript (npm 10.7.0 and Node.js 20.15.0), while Solidity 0.8.0 was chosen for implementing and compiling the `sm_reg` contract.

A. Average Register Location Time vs. Number of Clients

Figure 3 visualizes the relationship between the average register location time (in milliseconds) and the number of clients involved in the registration process. As the client load increases from 1 to 128, the average time required to register a device's location steadily grows.

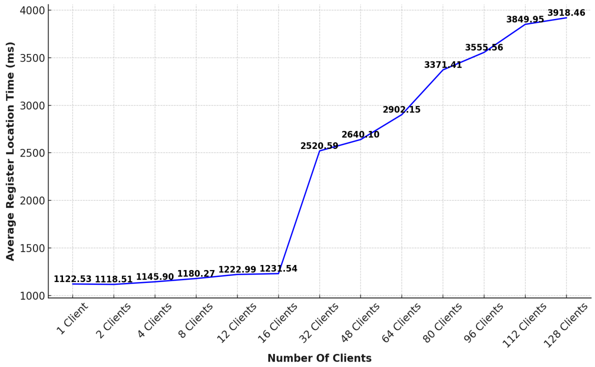

Fig. 3. Average Register Location Time vs. Number of Clients

Up to around 32 clients, the increase in average registration time is relatively moderate, with times ranging from 1122 ms to 2520 ms. However, beyond this point, especially with configurations involving 64 clients (2902 ms) and 128 clients (3918 ms), the registration time rises significantly.

This indicates that while the blockchain can handle a moderate number of clients efficiently, there are signs of stress as the number of clients exceeds 32. The sharp increase in time suggests that the resources allocated for the blockchain are insufficient, leading to bottlenecks tied to transaction processing speed and block confirmation times under higher load.

B. Total Execution Time vs. Number of Clients

The final chart (Figure 4) presents the total execution time (in seconds) required to process all transactions for each client configuration. As the client number increases, the execution time rises accordingly.

Execution time remains under 300 seconds for up to 64 clients (517 seconds), but spikes significantly for 128 clients, reaching 915 seconds.

This sharp rise indicates that the system's overall efficiency degrades as the client load increases. The growth in execution time is non-linear, suggesting that beyond a certain number of clients, the system becomes increasingly inefficient. This inefficiency is likely due to increased block propagation times and queuing delays in processing registration transactions.

References

[1] Goyat, R., Kumar, G., Alazab, M., Saha, R., Thomas, R., & Rai, M. K. (2021). A secure localization scheme based on trust assessment for WSNs using blockchain technology. Future Generation Computer Systems, 125, 221-231.

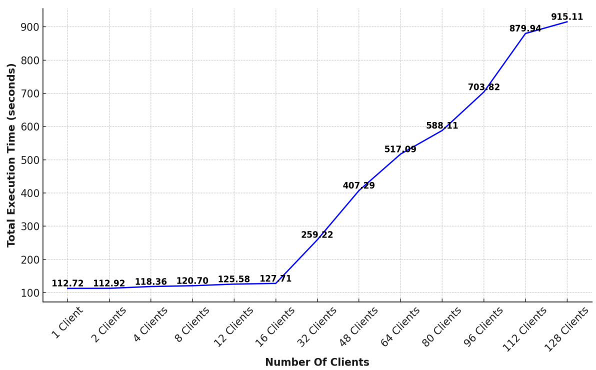

Fig. 4. Total Execution Time vs. Number of Clients

[2] He, J., Chun, Y. J., So, H. C., & EURASIP, M. (2023). Modeling and performance analysis of blockchain-aided secure TDOA localization under random internet-of-vehicle networks. Signal Processing, 206, 108904.

[3] Goyat, R., Kumar, G., Rai, M. K., Saha, R., Thomas, R., & Kim, T. H. (2020). Blockchain powered secure range-free localization in wireless sensor networks. Arabian Journal for Science and Engineering, 45, 6139-6155.

[4] Cheikhrouhou, O., & Koubâa, A. (2019, June). Blockloc: Secure localization in the internet of things using blockchain. In 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC) (pp. 629-634). IEEE.

[5] Baucas, M. J., Gadsden, S. A., & Spachos, P. (2021). IoT-based smart home device monitor using private blockchain technology and localization. IEEE Networking Letters, 3(2), 52-55.

[6] Kim, T. H., Goyat, R., Rai, M. K., Kumar, G., Buchanan, W. J., Saha, R., & Thomas, R. (2019). A novel trust evaluation process for secure localization using a decentralized blockchain in wireless sensor networks. IEEE access, 7, 184133-184144.

[7] Amoretti, M., Brambilla, G., Medioli, F., & Zanichelli, F. (2018, July). Blockchain-based proof of location. In 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C) (pp. 146-153). IEEE.

[8] Gebremariam, G. G., Panda, J., & Indu, S. (2023). Blockchain-Based Secure Localization against Malicious Nodes in IoT-Based Wireless Sensor Networks Using Federated Learning. Wireless communications and mobile computing, 2023(1), 8068038.

[9] Haleem, A., Allen, A., Thompson, A., Nijdam, M., & Garg, R. (2018). A decentralized wireless network. Helium Netw, 3-7.

[10] Pan, H., Wang, Y., Wang, W., Cao, P., Ye, F., & Wu, Q. (2024). Privacy-preserving location authentication for low-altitude UAVs: A blockchain-based approach. Security and Safety, 3, 2024004.

[11] Saia, R., Podda, A. S., Pompianu, L., Reforgiato Recupero, D., & Fenu, G. (2021). A blockchain-based distributed paradigm to secure localization services. Sensors, 21(20), 6814.

[12] Li, X., Deng, Z. D., Rauchenstein, L. T., & Carlson, T. J. (2016). Contributed Review: Source-localization algorithms and applications using time of arrival and time difference of arrival measurements. Review of Scientific Instruments, 87(4).

[13] Zekavat, R., & Buehrer, R. M. (Eds.). (2019). Handbook of position location: theory, practice, and advances.

[14] Pourkhaatoun, M., & Zekavat, S. A. (2018). TOA estimation techniques: A comparison. Handbook of Position Location: Theory, Practice, and Advances, Second Edition, 237-267.

[15] Gebremariam, G. G., Panda, J., & Indu, S. (2023). Secure localization techniques in wireless sensor networks against routing attacks based on hybrid machine learning models. Alexandria Engineering Journal, 82, 82-100.

[16] Delwar, T. S., Aras, U., Mukhopadhyay, S., Kumar, A., Kshirsagar, U., Lee, Y., ... & Ryu, J. Y. (2024). The Intersection of Machine Learning and Wireless Sensor Network Security for Cyber-Attack Detection: A Detailed Analysis. Sensors, 24(19), 6377.

[17] Katz, J., & Lindell, Y. (2007). Introduction to modern cryptography: principles and protocols. Chapman and hall/CRC.

[18] Goldwasser, S., & Bellare, M. (1996). Lecture notes on cryptography. Summer course “Cryptography and computer security” at MIT, 1999, 1999.